SEMIDEFINITE PROGRAMMING APPROACHES TO DISTANCE GEOMETRY PROBLEMS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Pratik Biswas

June 2007

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Yinyu Ye) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Stephen Boyd)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Leonidas Guibas
Computer Science)

Approved for the University Committee on Graduate Studies.

Abstract

Given a subset of all the pair-wise distances between a set of points in a fixed dimension, and possibly the positions of few of the points (called anchors), can we estimate the (relative) positions of all the unknown points (in the given dimension) accurately? This problem is known as the Euclidean Distance Geometry or Graph Realization problem, and generally involves solving a hard non-convex optimization problem. In this thesis, we introduce a polynomial-time solvable Semidefinite Programming (SDP) relaxation to the original formulation.

The SDP yields higher dimensional solutions when the given distances are noisy. To alleviate this problem, we adopt ideas from dimensionality reduction and use local refinement to improve the estimation accuracy of the original relaxation. We apply this algorithm to Wireless Sensor Network Localization; *i.e.*, estimating the positions of nodes in a wireless network using pair-wise distance information (acquired from communications between the nodes within 'hearing' range of each other). Using this 'cooperative' approach, we can estimate network structures with very few anchors, and low communication range, offering a distinct advantage over schemes like GPS triangulation.

As the number of unknown points increases, the problem starts becoming computationally intractable. We describe an iterative and distributed approach to realize the structure of a graph, even in the absence of anchor points. This algorithm is implemented for Molecule Conformation; *i.e.*, finding the 3-D structure of a molecule, given just the ranges on a few interatomic distances (acquired by Nuclear Magnetic Resonance measurements). We are able to derive accurate structures of molecules with thousands of atoms, with sparse and noisy interatomic distance data.

We also extend these formulations to solve the problem of position estimation using angle information. Once again, we demonstrate the working of the algorithm for real life systems, in this case, image based sensor networks.

Acknowledgements

I hope to express my gratitude towards at least some of the individuals and institutions that, through their kindness and encouragement, have led me to the point where I can claim to be a certified 'Doctor of Philosophy'.

First and foremost, I would like to thank my adviser Professor Yinyu Ye. He has done a lot for me in the last 5 years, ever since I signed up with him for an independent study in my first year as a masters student, curious to learn more about convex optimization. With his endless supply of fresh ideas and openness to looking at new problems in different areas, his guidance has proved to be indispensable to my research. On a personal level, I will always remember the enthusiasm and friendliness with which he received anyone who would stop by his office. I hope to imbibe his sense of humility and good humor into my life.

A great debt is also due to Professor Stephen Boyd, for his patient reading of this thesis, and his courses on linear algebra and convex optimization. It was his unique style of teaching that initially helped shape my research interests. Thanks to Professor Leonidas Guibas who took time out of his busy schedule to be in my reading committee. The suggestions offered by Professors Boyd and Guibas greatly helped improve the quality and presentation of this thesis. Professor Fabian Pease was very kind in agreeing to chair my orals committee and I am very grateful to him for it.

I would like to acknowledge the contributions of my co-authors and collaborators: Professor Toh-Kim Chuan from NUS, for the work on noise handling and molecule conformation, Tzu-Chen Liang and Ta-Chung Wang for their work on gradient descent, Anthony So for his theoretical work on localizability and tensegrity theory, and Siddharth Joshi for his advice with the truncated Newton methods. The members of Professor Ye's research group: Arjun, Ben, Dongdong, John, Mark, Shipra, Zhisu and others also gave invaluable comments and suggestions during the course of this research.

I am very grateful to Dr. Hamid Aghajan, Director of the Wireless Sensor Networks Laboratory at Stanford, for his help with the work on using angle information, and for the opportunity to be his teaching assistant in one of his courses. I also appreciate the help and guidance offered to me by Dr. Renate Fruchter, Director of the Project Based Learning Laboratory, during my first couple of years at Stanford.

I would like to express my gratitude towards the Automatic Control Laboratory at ETH Zurich, the Sensor Networks Operation at Intel Research and Amaranth Advisors, for inviting me to summer internships and giving me interesting problems to work on.

My deepest thanks to Stanford University, which may be considered a slightly amorphous entity in this context. For all the wonderful people I have met here, and all the opportunities I was allowed to learn and appreciate the good things in life. I feel incredibly lucky to have been a part of this great institution, and I will cherish the memories of my time here.

My family has been a strong presence in my life. Apart from their unquestioning love and support, they have given me the gift of a deep sense of wonder for all creation and an open mind with which to explore it. Whatever pursuits I have undertaken in my life, I have always been inspired by their words and actions, and I will be forever indebted to them for it.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The study of distances to infer structure is an important one, and is gaining increasing applicability in fields as varied as wireless network position estimation [11, 25, 77], molecule structure prediction [12, 27, 59], data visualization [34, 66], internet tomography [24] and map reconstruction [28]. More recently, the theory is being applied also to manifold learning and dimensionality reduction, particularly in the computer graphics and image processing domains. Examples include face recognition [49] and image segmentation [92].

One essential question in this domain is: given a noisy and incomplete set of Euclidean distances between a network of points (in a given dimension), can we find robust, efficient and scalable algorithms to find their (relative) positions? We will focus on two particular applications of this problem which have received great attention from researchers, and also are the examples considered in detail in this thesis.

1.1.1 Wireless sensor network localization

Advances in micro-electro-mechanical systems (MEMS) and wireless communications have made the sensor network a highly efficient, low cost and robust technology. A typical sensor network consists of a large number of sensors which are densely deployed in a geographical area. Sensors collect the local environmental information such as temperature or humidity and can communicate with each other to relay the gathered data to a decision making authority. Based on the gathered data, inferences can be made about the status of the environment being monitored and appropriate action taken. Examples of areas which are attracting large research efforts in this direction and in prototypes networks have been developed include the habitat monitoring system in the Great Duck Island (GDI) experiment [57], environment monitoring for detecting volcano eruptions [3], the Code-Blue project for emergency medical care [55], industrial control in semiconductor manufacturing plants and oil tankers [51], structural health monitoring [23, 99], military and civilian surveillance and moving object tracking [38] and asset location [32].

The information collected through a sensor network can be interpreted and relayed far more effectively if we know where it is coming from and where it needs to be sent, *i.e.*, there is some spatial information corresponding to each sensor. For example, in an environmental monitoring application, just having access to the raw data does not yield much information about the variation of the environmental factors within the space considered. In contrast, having a geographical map and environmental information corresponding to points in the map is much more insightful, in terms of providing a clearer picture of the how the environmental parameters change across the observed region. In fact, some sensor network applications may simply be required to perform the task of target tracking or asset location. The positions of sensor nodes may also be used for intelligent geographical routing purposes, thus reducing the burden on the network's communication and power resources.

For densely deployed networks spread over a large area, it is often very expensive and infeasible to manually administrate the setup of the network, especially when sometimes the setup is ad-hoc in nature. The initialization of such networks should be done automatically by the network itself. Furthermore, the sensor devices are designed to be low power and low bandwidth, in order to minimize the operating costs and maximize operating lifetime. Therefore, it would be too expensive, even infeasible, to use solutions such as a Global Positioning System (GPS) [46] device on every sensor, compounded by the fact that it would have difficulty in indoor scenarios. It might be possible to have at least a few anchor nodes, however, whose positions are known either by GPS or manual placement.

For the other sensor nodes, relative position information like distances and angles between sensors may be available. A wireless sensor can have the capability to communicate with other sensors which are within a communication radius of it. The communication radius depends mainly on the transmission and measurement schemes used and their respective power and communication constraints. Using different range and angle measurement schemes such as received signal strength (RSS), time of arrival (ToA), angle of arrival (AoA), relative distance and angle estimates can be made about the neighboring sensors.

The problem of inferring global position information for all sensor nodes using this pair-wise distance information is referred to as the localization problem.

Some excellent surveys on both the hardware and algorithm aspects of the localization problem include [40], [60],[72]. The different strategies pursued vary in terms of the kind of measurements (distance, angle, hop information) and hardware and sensing mechanisms (RF, acoustic, camera based) used. Many current localization solutions use triangulation with multiple anchors to find unknown sensor positions. This approach, however, calls for high anchor density and high communication radius, both of which are costly propositions. Using global distance information between all sensors, not just anchor-unknown, but unknown-unknown as well, would allow the entire network to 'cooperatively' compute positions for all its nodes in a far more cost-effective manner. The challenge is to develop such schemes that are scalable to large networks, in terms of cost and power, but at the same time, to not compromise on estimation accuracy, inspite of the noise and incompleteness in the distance measurements.

1.1.2 Molecule conformation

Significant research efforts are being directed into the field of structural genomics, *i.e.*, the study of mapping the 3-D structure of molecules, proteins in particular [17, 91]. Creating a database of protein structures, like in the case of the Protein Data Bank [8], provides the opportunity to recognize homology in different proteins that may not be detectable by merely comparing sequences. This is quite likely because protein structure is often well conserved over evolutionary time. By developing basic 3-D structures of proteins, we can build up a family of proteins with similar characteristics. In fact, such structure is also crucial in determining the molecular mechanism and function of the protein in question, and can be used in directing efforts in drug design. Combined with other molecular modeling tools, molecule structure determination (or conformation) can prove to be an indispensable component in the study of understanding proteins.

Kurt Wüthrich and his co-researchers started a revolution in this field by introducing the use of Nuclear Magnetic Resonance (NMR), as a convenient method by which to measure interatomic distances for proteins in solution [97]. The nuclei of some atoms resonate at particular frequencies when placed in a magnetic field of a particular intensity. However, depending on the local chemical environment (influenced by neighboring atoms and electron density), the frequency shifts slightly. By performing a series of experiments on the molecule, an NMR spectrum can be constructed. By observing the peaks in the NMR spectra, distance ranges can be computed between the nuclei of corresponding to a particular peak. Having recovered distance ranges between some of the atoms, valid configurations of the molecule that satisfy these distance constraints need to be found, a task in which distance geometry techniques play a huge role.

The basic statement of the problem is as follows: Given upper and lower bounds on a sparse subset of all interatomic distances, can we find configurations of the entire molecule that satisfy the given distance constraints? The book by Crippen and Havel [27] provides a comprehensive background to the links between molecule conformation and distance geometry. The problem sizes in this space (often thousands of atoms) call for parallel or distributed approaches, without which the problem may be intractable.

1.2 Distance geometry background

The distance geometry problem has been studied extensively, and for a long time, in the framework of Euclidean distance matrix (EDM) completion. Schoenberg [74] and Young and Householder [103] established some basic properties of distance matrices which have set the stage for future research in distance geometry.

In particular, they considered the case when all pair-wise distances between a set of n points in $\mathcal{R}^d$ are known. Let $X = [x_1, x_2, \ldots, x_n]$ be the $d \times n$ matrix of points, and D be the distance matrix, where $D_{ij} = \|x_i - x_j\|^2$. The Gram matrix or inner product matrix is defined as $Y = X^T X$. Note that

$$D_{ij} = Y_{ii} + Y_{jj} - 2Y_{ij}.$$

By assuming $x_n = 0$, without loss of generality(since this corresponds to a simple translation), the Gram matrix can also be expressed in terms of the distance matrix as

$$Y_{ij} = \frac{1}{2}(D_{in} + D_{jn} - D_{ij}).$$

It was shown that the distance matrix D can correspond to a realization in dimension d if and only if the corresponding Gram matrix Y is positive semidefinite(psd), and has a rank equal to d.

Therefore, in the case of incomplete distance matrices, the task would be to complete the distance (or inner product matrix) in a manner such that a realization in dimension d is possible. It is also worth noting that in a factorization of the completed Gram matrix with rank d, where $Y = \tilde{X}^T \tilde{X}$, $\tilde{X}$ would correspond to a realization of the points in dimension d, and if there are enough distances in the incomplete matrix to ensure a unique realization to begin with, this would only be a translated and rotated version of the original set of points.

These ideas form the basis of a class of algorithms known as multidimensional scaling [26, 85]. Algorithms based on MDS sometimes use objective functions (such as the STRESS criterion) [87, 88] that ensure low-dimensional solutions for the given incomplete distance matrix. The problem with these techniques is that there is a possibility of getting stuck in local minima due to the nonconvex nature of the problem, and there is no guarantee of finding a desired realization in polynomial time.

With this in mind, researchers have also attempted to pin down the exact computational complexity of this problem. Saxe [73] proved that the EDM problem is NP-hard in general. Aspnes et al. [4] proved a similar result in the context of sensor localization. More and Wu [58], who used global optimization techniques for distance geometry problems in the molecule conformation space, established that finding an ϵ optimal solution, when ϵ is small, is also NP-hard. Other general global optimization approaches which employ techniques like pattern search [65] face similar problems.

Chapter 8 of the book by Boyd and Vanderberghe [16] investigates the use of convex optimization for a wide range of geometric problems. The book by Dattorro [28] explores more specifically the theoretical links between convex optimization and Euclidean distance geometry.

Semidefinite programming (SDP) relaxation techniques have been exploited for a large variety of NP-hard problems, many of which are concerning Euclidean distances, such as data compression, metric-space embedding, covering and packing, chain folding [22, 42, 54, 101], manifold learning and nonlinear dimensionality reduction [93]. Not surprisingly, it also finds application in Euclidean distance matrix completion. Alfakih [1] and Laurent [52] have discussed before the use of SDP relaxations in EDM completion, focussing on issues such as realizability and interior point algorithms for these SDPs. Barvinok [5] has used SDP theory to study this problem in the context of quadratic maps and to analyze the dimensionality of the solutions.

1.3 Contributions of the thesis

While there has been a lot of theoretical work in using SDP for distance geometry, its application to real life problems has so far been limited. This thesis presents a novel SDP relaxation for the distance geometry problem, and analyzes its theoretical properties in the context of the uniqueness and realizability of solutions in a fixed dimension. It then addresses actual scenarios where the SDP relaxations for Euclidean distance geometry may be applied, and in doing so, attempts to tackle the issues of achieving good estimation accuracy with noisy measurements and scalability for large scale implementation.

Chapter 2 introduces the basic distance geometry problem with exact distance information and its SDP relaxation. The properties of this relaxation, such as its tightness and computational complexity are discussed. The material in this chapter is based on the papers:

Pratik Biswas and Yinyu Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, pages 46–54. ACM Press, 2004, and

Pratik Biswas, Tzu-Chen Liang, Ta-Chung Wang, and Yinyu Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks*, 2(2):188–220, 2006.

The duality analysis in Section 2.3.2 of the chapter is borrowed from the paper:

Anthony Man-Cho So and Yinyu Ye. Theory of semidefinite programming relaxation for sensor network localization. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 405-414, 2005 and in *Mathematical Programming, Series B*, 109:367-384, 2007.

The basic SDP model is extended to deal with noisy distance information and its application to sensor network localization is discussed in Chapter 3. In particular, methods are developed to compute low-dimensional solutions for the SDP relaxation, such as adding a regularization term to the objective function of the optimization problem and using a postprocessing gradient descent for local refinement. The material in this chapter is based on the paper:

Pratik Biswas, Tzu-Chen Liang, Kim-Chuan Toh, Ta-Chung Wang, and Yinyu Ye. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineeering, Special Issue on Distributed Sensing*, 3(4):360–371, October 2006.

The distributed algorithm with anchor nodes is based on the paper:

Pratik Biswas and Yinyu Ye. A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization. Technical report, Dept of Management Science and Engineering, Stanford University, *appeared in Multiscale Optimization Methods and Applications, Springer*, October 2003.

Chapter 4 applies the SDP relaxation to molecule structure prediction. The challenges in molecule structure prediction are that the molecules are very large, consisting of thousands of atoms, and there are no nodes whose positions are previously known. Therefore, a distributed algorithm, that solves smaller clusters of points, and stitches them together using common points, is introduced. This material is based on the paper:

Pratik Biswas, Kim-Chuan Toh and Yinyu Ye. A distributed SDP approach for large-scale noisy anchor-free 3d graph realization (with applications to molecular conformation). Technical report, Dept of Management Science and Engineering, Stanford University, *submitted to SIAM Journal on Scientific Computing*, February 2007.

Finally Chapter 5 deals with localization using angle information. Formulations that maintain the Euclidean distance geometry flavor of the problem are developed. Issues relating to its computational efficiency are also resolved. This material is based on the papers:

Pratik Biswas, Hamid Aghajan, and Yinyu Ye. Semidefinite programming algorithms for sensor network localization using angle of arrival information. In *to appear, in Proceedings of Thirty-Ninth Annual Asilomar Conference on Signals, Systems, and Computers*, October 2005, and

Pratik Biswas, Hamid Aghajan, and Yinyu Ye. Integration of angle of arrival information for multimodal sensor network localization using semidefinite programming. Technical report, Wireless Sensor Networks Lab, Stanford University,, May 2005.

The thesis states its conclusions and lists directions for future research in Chapter 6.

Chapter 2

SDP relaxation of the distance geometry problem

2.1 The distance geometry problem

We now describe the distance geometry problem in its most basic form, with exact distance information. Consider a set of points in $\mathcal{R}^d$ with m anchors and n unknown points. The points whose locations are yet to be determined are denoted by $x_i \in \mathcal{R}^d$, $i = 1, \dots, n$. An anchor is a point whose location $a_k \in \mathcal{R}^d$, $k = n+1, \dots, n+m$, is already known. For a pair of unknown points x_i and x_j, the exact Euclidean distance between them is denoted as $\hat{d}_{ij}$. Similarly, for an unknown point x_i and an anchor a_k, the exact Euclidean distance between them is denoted as $\hat{d}_{ik}$. In general, not all pairs of unknown/unknown and unknown/anchor distances are known. So the pairs of unknown/unknown points and unknown/anchor points for which distances are known are denoted as $(i,j) \in \mathcal{N}_u$ and $(i,k) \in \mathcal{N}_a$ respectively.

Mathematically, the distance geometry problem in $\mathcal{R}^d$ can be stated as follows: Given m anchor locations $a_k \in \mathcal{R}^d, k = n+1, \dots, n+m$ and some distance measurements $\hat{d}_{ij}, (i,j) \in \mathcal{N}_u \cup \mathcal{N}_a$, find the locations of the n unknown $x_i \in \mathcal{R}^d, i = 1, \dots, n$.

We can also look at it as a graph realization problem. Let $\mathcal{V} := \{1, \dots, n\}$ and $\mathcal{A} := \{a_{n+1}, \dots, a_{n+m}\}$. The realization problem in $\mathcal{R}^d$ for the graph $(\mathcal{V}, \mathcal{A}; \widehat{D})$ is to determine the coordinates of the unknown points $x_1, \dots, x_n$ from the partial distance data $\widehat{D} = \{\hat{d}_{ij} : (i,j) \in \mathcal{N}_u \cup \mathcal{N}_a\}$. Since the distance information is exact, the realization $x_1, \dots x_n$ must satisfy

$$\begin{aligned} \|x_i - x_j\|^2 &= (\hat{d}_{ij})^2, \ \forall\ (i,j) \in \mathcal{N}_u, \\ \|x_i - a_k\|^2 &= (\hat{d}_{ik})^2, \ \forall\ (i,k) \in \mathcal{N}_a. \end{aligned} \tag{2.1}$$

In general, the problem of determining a valid realization is a non-convex optimization problem. One closely related approach is described by [29] wherein the proximity constraints between points which are within 'cutoff distance' of each other are modeled as convex constraints. Then a feasibility problem can be solved by efficient convex programming techniques. Suppose 2 nodes x_1 and x_2 are within a distance R of each other, the proximity constraint can be represented as a convex second order cone constraint of the form

$$\|x_1 - x_2\|_2 \le R.$$

This can be formulated as a matrix linear inequality [15]:

$$\begin{bmatrix} I_2 R & x_1 - x_2 \\ (x_1 - x_2)^T & R \end{bmatrix} \succeq 0. \tag{2.2}$$

Alternatively, if the exact distance $\hat{d}_{12} \le R$ is known, we could set the constraint

$$\|(x_1 - x_2)\|_2 \le \hat{d}_{12}.$$

The second-order cone method for solving Euclidean metric problems can be also found in Xue and Ye [100] where its superior polynomial complexity efficiency is presented.

However, this technique yields good results only if the anchors are placed on the outer boundary, since the estimated positions of their convex optimization model all lie within the convex hull of the anchors. So if the anchors are placed in the interior of the network, the position estimation of the unknown points will also tend to the interior, yielding highly inaccurate results. One also may ask why, if d_{12} is known, another 'bounding away' constraint of the type

$$\|(x_1 - x_2)\|_2 \ge \hat{d}_{12},$$

cannot be added. These constraints would make the problem much tighter and would yield more accurate results. The issue is that this type of constraint is not convex. Therefore, the efficient convex optimization techniques cannot apply. In the next section, we introduce a relaxation that is able to incorporate tighter convex constraints.

2.2 Semidefinite programming relaxation

We will use the following notation. For two symmetric matrices A and B, $A \succeq B$ means $A - B \succeq 0$, i.e. $A - B$ is a positive semidefinite matrix. The inner product between two $m \times n$ matrices A and B, defined as

$$\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij},$$

is denoted by $A \bullet B$. We use I_d and $\mathbf{0}$ to denote the $d \times d$ identity matrix and the vector of all zeros, whose dimensions will be clear in the context. The 2-norm of a vector x is denoted as $\|x\|$. We use the notation $[A\,;\,B]$ to denote the matrix obtained by appending B to the last row of A.

Let $X = [x_1\ x_2\ \ldots\ x_n] \in \mathcal{R}^{d\times n}$ be the position matrix that needs to be determined. It is readily shown that

$$\begin{aligned} \|x_i - x_j\|^2 &= e_{ij}^T X^T X e_{ij} \\ \|x_i - a_k\|^2 &= (e_i; -a_k)^T [X \quad I_d]^T [X \quad I_d](e_i; -a_k), \end{aligned}$$

where e_i is the ith unit vector in $\mathcal{R}^n$ and $e_{ij} = e_i - e_j$. Thus, the Equations in (2.1) can be represented in matrix form as finding a symmetric matrix $Y \in \mathcal{R}^{n\times n}$ and a matrix $X \in \mathcal{R}^n$ such that

$$\begin{aligned} & e_{ij}^T Y e_{ij} = (\hat{d}_{ij})^2,\ \forall\ (i,j) \in \mathcal{N}_u \\ & (e_i; -a_k)^T \begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} (e_i; -a_k) = (\hat{d}_{ik})^2,\ \forall\ (i,k) \in \mathcal{N}_a \\ & Y = X^T X. \end{aligned} \tag{2.3}$$

Our method is to relax the problem (2.3) to a semidefinite program by relaxing the constraint $Y = X^T X$ to $Y \succeq X^T X$. From [15], we know that the last matrix inequality is equivalent to the condition that

$$\begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} \succeq \mathbf{0}. \tag{2.4}$$

Let $\mathcal{K} = \{Z \in \mathcal{R}^{(n+d)\times(n+d)} \ : \ Z = \begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} \succeq \mathbf{0}\}$. The SDP relaxation can now be written as the following feasibility problem:

$$\begin{array}{rl} \text{Find} & Z \\ \text{subject to} & (e_{ij};\mathbf{0})(e_{ij};\mathbf{0})^T \bullet Z = (\hat{d}_{ij})^2, \ \forall\ (i,j) \in \mathcal{N}_u \\ & (e_i;-a_k)(e_i;-a_k)^T \bullet Z = (\hat{d}_{ik})^2, \ \forall\ (i,k) \in \mathcal{N}_a \\ & Z_{(n+1:n+d,n+1:n+d)} = I_d \\ & Z \succeq \mathbf{0}. \end{array} \tag{2.5}$$

Note that according to the context, $\mathbf{0}$ is used to represent the zero vector or matrix of appropriate dimension.

2.3 SDP model analyses

In this section, we probe deeper into the SDP model to answer questions like: When will the solution be unique? Is the SDP relaxation exact? What is the computational complexity of the algorithm, and how does it improve over the weaker relaxation?

2.3.1 Uniqueness

The matrix Z of the SDP (2.5) has $nd + n(n+1)/2$ unknown variables. Consider the case that among $\{i, j, k\}$, there are $nd + n(n+1)/2$ pairs in $\mathcal{N}_u$ and $\mathcal{N}_a$, Then we have at least $nd+n(n+1)/2$ equality constraints. Moreover, if these equalities are linearly independent, then Z has a unique solution.

Proposition 2.1. *If there are $nd+n(n+1)/2$ distance pairs each of which has an accurate distance measure and the SDP* (2.5) *has a unique feasible solution*

$$\bar{Z} = \begin{pmatrix} Y & \bar{X}^T \\ \bar{X} & I_d \end{pmatrix},$$

then we must have $\bar{Y} = (\bar{X})^T\bar{X}$ and $\bar{X}$ equal true positions of the unknown sensors, i.e., the SDP relaxation solves the original problem exactly.

Proof. Let X^* be the true locations of the n points, and

$$Z^* = \begin{pmatrix} (X^*)^T X^* & (X^*)^T \\ X^* & I_d \end{pmatrix}.$$

Then Z^* is a feasible solution for (2.5). On the other hand, since $\bar{Z}$ is the unique solution to satisfy the $nd + n(n+1)/2$ equalities, we must have $\bar{Z} = Z^*$ so that $\bar{Y} = (X^*)^T X^* = \bar{X}^T\bar{X}$. $\square$

We present a simple case to show what it means for the system has a unique solution. Consider $d = 2$, $n = 1$ and $m = 3$. The accurate distance measures from unknown b_1 to known a_1, a_2 and a_3 are d_{11}, d_{21} and d_{31}, respectively. Therefore, the three linear equations are

$$\begin{aligned} y - 2x^T a_1 &= (d_{11})^2 - \|a_1\|^2, \\ y - 2x^T a_2 &= (d_{21})^2 - \|a_2\|^2, \\ y - 2x^T a_3 &= (d_{31})^2 - \|a_3\|^2. \end{aligned}$$

This system has a unique solution if it has a solution and the matrix

$$\begin{pmatrix} 1 & 1 & 1 \\ a_1 & a_2 & a_3 \end{pmatrix}$$

is nonsingular. This essentially means that the three points a_1, a_2 and a_3 are not on the same line, and then $\bar{x} = b_1$ can be uniquely determined. Here, the SDP method reduces to the so-called triangular method. Proposition 2.1 and the example show that the SDP relaxation method has the advantage of the triangular method in solving the original problem.

2.3.2 Analysis of dual and exactness of relaxation

The dual of the SDP relaxation is as follows:

$$\begin{array}{ll} \text{Minimize} & I_d \bullet V + \sum_{(i,j)\in\mathcal{N}_u} y_{ij}(\hat{d}_{ij})^2 + \sum_{(i,k)\in\mathcal{N}_a} w_{ik}(\hat{d}_{ik})^2 \\ \text{subject to} & \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & V \end{pmatrix} + \sum_{(i,j)\in\mathcal{N}_u} y_{ij}(e_{ij};\mathbf{0})(e_{ij};\mathbf{0})^T + \sum_{(i,k)\in\mathcal{N}_a} w_{ik}(e_i;-a_k)(e_i;-a_k)^T \succeq 0. \end{array} \tag{2.6}$$

Let U be the the $(n+d)$ dimensional dual slack matrix, *i.e.*

$$U = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & V \end{pmatrix} + \sum_{(i,j)\in\mathcal{N}_u} y_{ij}(e_{ij};\mathbf{0})(e_{ij};\mathbf{0})^T + \sum_{(i,k)\in\mathcal{N}_a} w_{ik}(e_i;-a_k)(e_i;-a_k)^T.$$

We will employ duality theory to investigate when the SDP (2.5) is an exact relaxation of problem (2.1). Firstly, observe that the dual problem (2.6) is always feasible, since $V = \mathbf{0}$, $y_{ij} = 0 \ \forall \ (i,j) \in \mathcal{N}_u$, $w_{ik} = 0 \ \forall \ (i,k) \in \mathcal{N}_a$ is a feasible solution. Now suppose that the primal problem (2.5) is also feasible. This will actually be the case for the exact distance values that are being considered so far. Then, from duality theory for SDP [2], we can conclude that there is no duality gap and the optimal value of the dual must be 0. Furthermore, we can state the following theorem.

Theorem 2.1. *Let $\bar{Z}$ be a feasible solution to the primal* (2.5)*, and $\bar{U}$ be an optimal slack matrix for the dual* (2.6)*. Then*

1. *Complementary slackness holds,* i.e., $\bar{Z} \bullet \bar{U} = 0$ *or* $\bar{Z}\bar{U} = \mathbf{0}$.
2. *Rank($\bar{Z}$) + Rank($\bar{U}$) $\leq d + n$.*
3. *Rank($\bar{Z}$) $\geq d$, and Rank($\bar{U}$) $\leq n$.*

This leads to the following corollary:

Corollary 2.1. *If an optimal dual slack matrix has rank n, then every feasible solution of the primal* (2.5) *must have rank d. In this case, the problems* (2.1) *and* (2.5) *are equivalent and the relaxation is exact.*

We also state the following technical result to be used in proofs later:

Proposition 2.2. *If every point is connected, directly or indirectly, to an anchor in problem* (2.1)*, then any solution to problem* (2.5) *must be bounded.*

Proof. If point x_i is connected to an anchor point a_k, then:

$$\|x_i^2\| + \|a_k^2\| - 2a_k^T x_i \leq Y_{ii} + \|a_k^2\| - 2a_k^T x_i = (\hat{d}_{ik})^2,$$

which implies that

$$Y_{ii} \leq (\hat{d}_{ik})^2 - \|a_k^2\| + 2\|a_k\|\|x_i\|.$$

$\|x_i\|$ is bounded by the triangle inequality, therefore Y_{ii} is bounded too. Now if x_j is connected to x_i and Y_{ii} is bounded,

$$Y_{ii} + Y_{jj} - 2\sqrt{Y_{ii}Y_{jj}} \leq Y_{ii} + Y_{jj} - 2Y_{ij} = (\hat{d}_{ij})^2.$$

Again, by a simple application of the triangle inequality, we conclude that Y_{jj} is also bounded. □

The discussion about rank is particularly important because the interior point algorithms that are used to solve the SDP compute the max-rank solutions for both the primal and dual [36]. A primal (dual) max-rank solution is one that has the highest rank among all solutions for the primal (dual). Keeping this in mind, we move to the following definition:

Definition 2.1. *The problem* (2.1) *is said to be uniquely localizable if there is a unique realization $\bar{X} \in \mathcal{R}^{d\times n}$ and there is no $x_i \in \mathcal{R}^h$, $i = 1, \ldots, n$, where $h > d$, such that:*

$$\begin{aligned}
&\|x_i - x_j\|^2 = (\hat{d}_{ij})^2, \ \forall\ (i,j) \in \mathcal{N}_u \\
&\|x_i - (a_k; \mathbf{0})\|^2 = (\hat{d}_{ik})^2, \ \forall\ (i,k) \in \mathcal{N}_a \\
&x_i \neq (\bar{x}_i; \mathbf{0}) \ \ \textit{for some } i \in 1, \ldots, n.
\end{aligned}$$

This definition basically implies that there should exist no non-trivial realization in a higher dimensional space (The trivial realization corresponds to setting the $x_i = (\bar{x}_i; 0)$ for $j = 1, \ldots, n$). The following theorem links the idea of unique localizability to the SDP relaxation.

Theorem 2.2. *Suppose that there are enough distance measures between network of points such that the underlying graph is connected. Then, the following statements are equivalent:*

1. *The problem is uniquely localizable.*

2. *The max-rank solution matrix of the corresponding SDP* (2.5) *has rank d.*

3. *The solution matrix of the SDP* (2.5)*, satisfies $Y = X^T X$.*

Proof. The equivalence between (2) and (3) is straightforward. In proving that (2) implies (1), we note that any rank d solution of (2.5) is also a solution to (2.1). However, we need to prove that if the max-rank solution has rank d, it must be unique. If not, then suppose that there exist 2 rank d feasible solutions,

$$Z_1 = \begin{pmatrix} X_1^T X_1 & X_1^T \\ X_1 & I_d \end{pmatrix} \text{ and } Z_2 = \begin{pmatrix} X_2^T X_2 & X_2^T \\ X_2 & I_d \end{pmatrix}.$$

Then the matrix $Z = \alpha Z_1 + \beta Z_2$, where $\alpha + \beta = 1$, $\alpha, \beta > 0$ is also a feasible rank d solution, since all feasible solutions must have at least rank d, but the max-rank is assumed to be d. Therefore,

$$Z = \begin{pmatrix} \alpha X_1^T X_1 + \beta X_2^T X_2 & \alpha X_1^T + \beta X_2^T \\ \alpha X_1^T + \beta X_2^T & I_d \end{pmatrix} = \begin{pmatrix} B^T B & B^T \\ B & I_d \end{pmatrix},$$

where $B = \alpha X_1 + \beta X_2$. This implies that $(X_1 - X_2)^T (X_1 - X_2)^T = \mathbf{0}$, or $\|X_1 - X_2\| = 0$, or $Z_1 = Z_2$, which is a contradiction.

To prove the opposite direction, we must show that the max-rank solution of (2.5) has rank d. Suppose there is a feasible solution Z with rank higher than d. Then, we must have $Y \succ X^T X$. In fact, we can write $Y = X^T X + (X')^T (X')$, where $X' = [x'_1, \ldots, x'_n] \in \mathcal{R}^{r \times n}$ and r is the rank of $Y - X^T X$. Now, consider the points

$$\tilde{x}_i = \begin{pmatrix} x_i \\ x'_i \end{pmatrix} \text{ for } i = 1, \ldots, n.$$

We have $\|\tilde{x}_i\|^2 = Y_{jj}$, $(\tilde{x}_i^T)\tilde{x}_j = Y_{ij}\ \forall\, i, j$. We also know from Proposition 2.2 that Y_{ii} and Y_{ij} are bounded for all i, j. Therefore,

$$\begin{aligned} \|\tilde{x}_i - \tilde{x}_j\|^2 = &\quad (\hat{d}_{ij})^2, \quad \forall\, (i,j) \in \mathcal{N}_u \\ \|\tilde{x}_i - (a_k; \mathbf{0})\|^2 = &\quad (\hat{d}_{ik})^2, \quad \forall\, (i,k) \in \mathcal{N}_a, \end{aligned}$$

which is a contradiction of (1). Therefore, (1) implies (2). □

The importance of this theorem is that it states that if the problem is uniquely localizable, then the relaxation problem (2.5) solves problem (2.1) exactly. To tell whether a problem instance is uniquely localizable or not before solving it is not easy. But once we solve the SDP relaxation and observe whether $Y = X^T X$ in the solution, we immediately know if the problem is uniquely localizable or not. In fact, in general, the problem of computing a realization of points, even when the instance is unique is NP-complete. But this shows that there exists a family of graphs for which a realization can be computed in polynomial time. It should be kept in mind however, that the assumption in the above analyses is that the distance measures are exact. We will deal with noisy data in the following chapter. We will now extend this analysis to introduce a new notion called strong localizability.

Definition 2.2. *We say that the problem instance is strongly localizable if the dual of its SDP relaxation* (2.6) *has an optimal dual slack matrix with rank n.*

Note that if a network is strongly localizable, then it is uniquely localizable from Theorems 2.1 and 2.2, since the rank of all feasible solution of the primal is d. Using this definition, we develop the following theorem:

Theorem 2.3. *If a problem (graph) contains a subproblem (subgraph) that is strongly localizable, then the submatrix solution corresponding to the subproblem in the SDP solution has rank d, i.e., the SDP relaxation computes a solution that localizes all possibly localizable unknown points.*

Proof. Let the subproblem have n_s unknown points and they are indexed as $1, \ldots, n_s$. Since it is strongly localizable, an optimal dual slack matrix U_s of the SDP relaxation for the subproblem has rank n_s. Then, in the dual problem of the SDP relaxation for the whole problem, we set V and those w_{kj}'s associated with the subproblem to the optimal slack matrix U_s and set all other w_{kj}'s to 0. Then, the slack matrix

$$U = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & U_s \end{pmatrix} \succeq 0$$

must be optimal for the dual of the (whole-problem) SDP relaxation, and it is complementary to any primal feasible solution of the (whole-problem) SDP relaxation

$$Z = \begin{pmatrix} * & * \\ * & Z_s \end{pmatrix} \succeq 0 \text{ where } Z_s = \begin{pmatrix} Y_s & X_s^T \\ X_s & I_d \end{pmatrix}.$$

However, we have $0 = Z \bullet U = Z_s.U_s$ and $Z_s, U_s \succeq 0$. The rank of U_s is n_s implies that the rank of Z_s is exactly d, *i.e.*, $Y_s = (X_s)^T X_s$. So X_s is the unique realization of the subproblem. □

The above theorems indicate that the SDP relaxation is able to estimate exactly all the localizable points. For those points that are not localizable, the SDP solution yields observable error measures. In particular, each individual trace

$$\bar{Y}_{jj} - \|\bar{x}_j\|^2 \tag{2.7}$$

helps us to detect the errors in estimation and isolate exactly those points which are incorrectly estimated given the incomplete distance information.

2.3.3 Computational complexity

A worst-case complexity result to solve the SDP relaxation can be derived from employing interior-point algorithms, e.g., [7].

Theorem 2.4. *Let $k = 3 + |\mathcal{N}_u| + |\mathcal{N}_a|$, the number of constraints. Then, the worst-case number of total arithmetic operations to compute an ϵ-solution of (2.5), meaning its objective value is at most $\epsilon(> 0)$ above the minimal one, is bounded by $O(\sqrt{n+k}(n^3 + n^2k + k^3)\log\frac{1}{\epsilon})$, in which $\sqrt{n+k}\log\frac{1}{\epsilon}$ represents the bound on the worst-case number of interior-point algorithm iterations.*

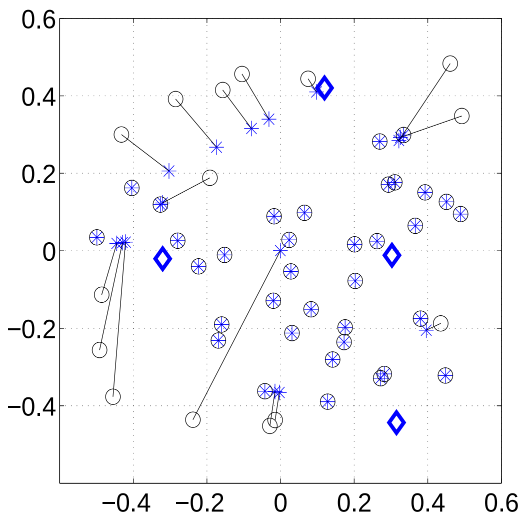
Practically, the number of interior-point algorithm iterations to compute a fairly accurate solution is a constant, $20 - 30$, for semidefinite programming, and k is bounded by $O(n^2)$. Thus, the worst case complexity is bounded by $O(n^6)$. However, in practice, as the number of points increases, the required radio range and number of constraints required to solve for all positions scales more typically with $O(n)$. Therefore the number of operations is typically bounded by $O(n^3)$ in solving a localization problem with n sensors.

2.3.4 Examples

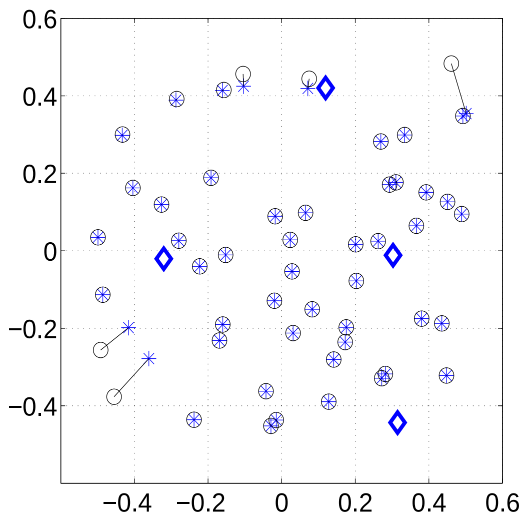
For the purpose of our examples, we generated random networks of points uniformly distributed in a square area of size 1×1. Distances were computed between points that are within a radius R of each other. (The radius indicates that the distance values between any two nodes are known to the solver if they are below the range; otherwise they are unknown.) The original and the estimated point positions were plotted. The (blue) diamond nodes refer to the positions of the anchors; the (black) circle nodes to the original locations of the unknown points; and the (blue) asterisk nodes to their estimated positions from $\bar{X}$. The discrepancies in the positions can be estimated by the offsets between the original and the estimated points as indicated by the solid lines. Unless stated otherwise, the same setup and notations will be used in all examples of this thesis.

Example 2.1. *A network of 50 points and 3 anchors was considered. The effect of varying R and as a result, connectivity, was observed in Figure 2.1. R was varied from 0.2 to 0.35. For Figures 2.1(a) and 2.1(b), there are some points for which there is not enough distance information, and therefore, they have larger errors in estimation. Their individual trace errors 2.7 are also high, and they accurately correlate with the real estimation errors. See Figure 2.2 for the correlation between the individual estimation error and trace error for each unknown sensor for the cases in Figure 2.1(a) and 2.1(b). The (black) diamonds indicate the (normalized) actual estimation error and the (blue) squares indicate the (normalized) trace error.*

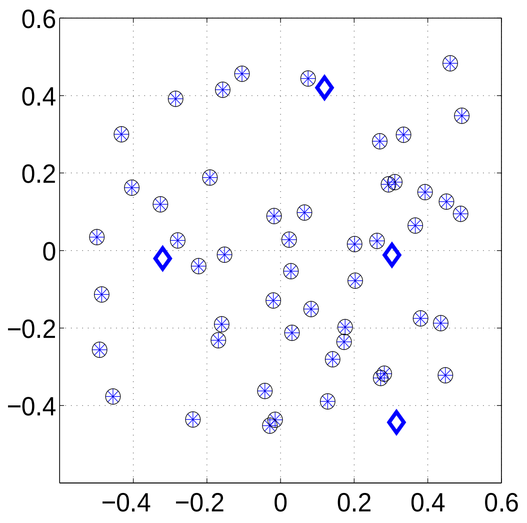
In comparison, for the same case as Figure 2.1, we computed the results from the Doherty et al. [29] method, but with more points as anchors (10 and 25). Figure 2.3 shows the estimation results. As we expected, the estimated positions were all in the convex hull of the anchors.



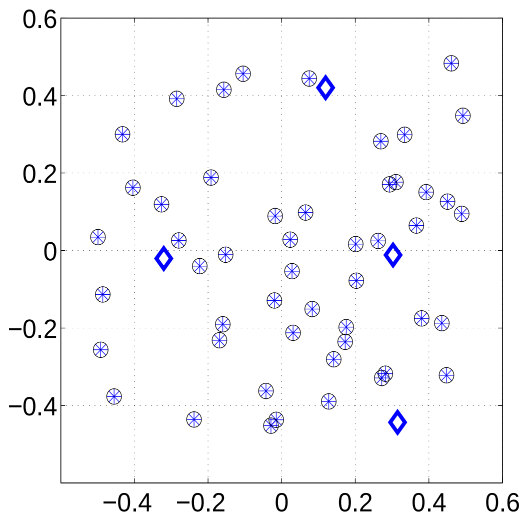
(a) R=0.2



(b) R=0.25

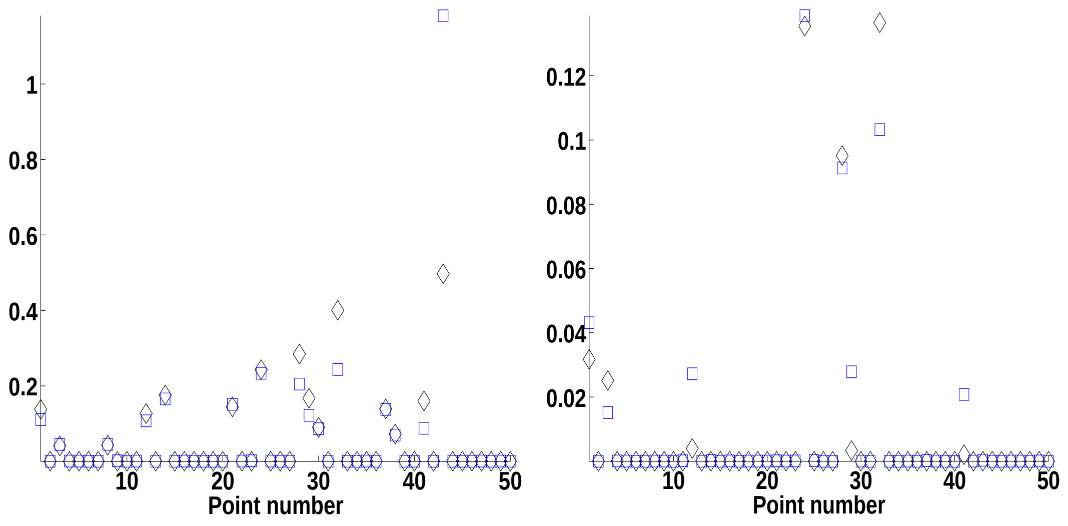


(c) R=0.3



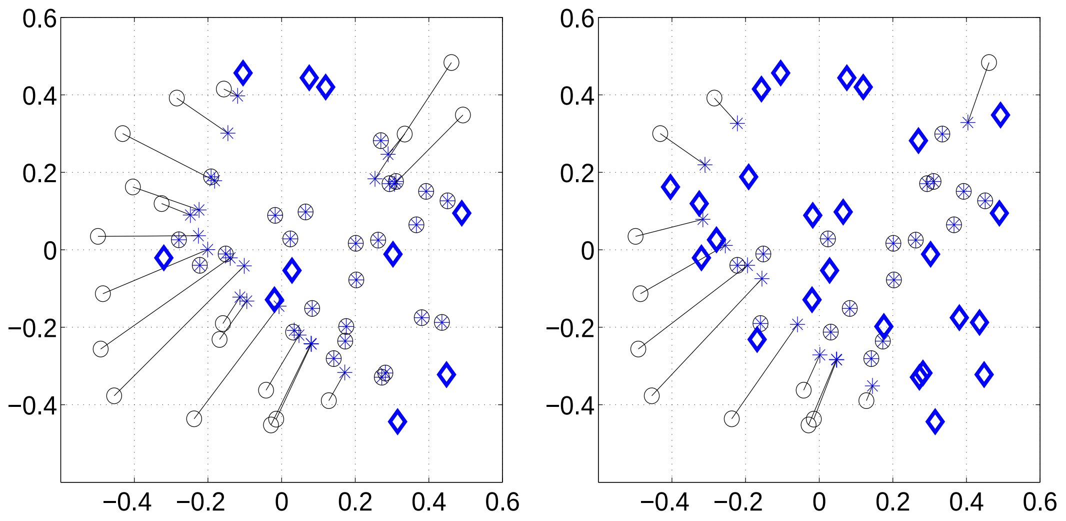
(d) R=0.35

Figure 2.1: Position estimations with 3 anchors, accurate distance measures and varying R.



(a) Error and trace correlation in Figure 2.1(a) (b) Error and trace correlation in Figure 2.1(b)

Figure 2.2: Diamond: the offset distance between estimated and true positions, Box: the square root of individual trace $\bar{Y}_{jj} - \|\bar{x}_j\|^2$.



(a) 10 anchors (b) 25 anchors

Figure 2.3: Position estimations by Doherty et al., R=0.3, accurate distance measures and various number of anchors.

Chapter 3

Dealing with noise: The sensor network localization problem

3.1 Introduction

The sensor network localization problem is: assuming the accurate positions of some nodes (called anchors) are known, how do we use them and partial pair-wise distance measurements or angle measurements to locate the positions of all sensor nodes in the network? In this chapter, we will only deal with distance measurements. The difficulty in this problem arises from two factors. First, the distance measurements always have some noise or uncertainty. Depending on the accuracy of these measurements and processor, power and memory constraints at each of the nodes, there is some degree of error in the distance information. Since the effect of the measurement uncertainty usually depends also on the geometrical relationship between sensors and is not known a priori, an unbiased model is not easy to build. Second, even if the distance measurements are perfectly accurate, a sufficient condition for this sensor network to be localizable cannot be identified easily, see [31, 43].

In this chapter, we extend the general graph realization problem and SDP model described in Chapter 2 to formulate the sensor network localization problem with noisy measurements, and derive results on the upper and lower bounds on the SDP objective function. The SDP objective function gives us an idea of the accuracy of the technique and also provides a 'proof of quality' against which we can measure any estimation. In the SDP approach, noisy distance measurements typically yield higher dimensional solutions. The projection of these solutions which into $\mathcal{R}^2$ generally do not produce satisfactory results. We have developed 2 new techniques to obtain better estimation accuracy. One approach is to add a regularization term to the objective function to force the SDP solution to lie close to a lower dimensional space, by penalizing folding between the points and maximizing the spacing between them. The other approach is to improve the SDP estimated solution using a local refinement method (such as gradient descent). One should note that these methods generally do not deliver a global optimal solution when the problem is non-convex. However, our numerical results show that the SDP estimated solution generally gives a good starting point for the local refinement method to converge to the global optimal solution. It is demonstrated that the improved solution is very close to the optimal one by comparing the minimal objective value to the SDP lower bound.

Unfortunately, the existing SDP solvers scale very poorly and the resulting SDP is often intractable for large networks with hundreds or thousands of points given current SDP solvers. An iterative distributed SDP computation scheme, first demonstrated in [13], has been developed to overcome this difficulty. Our distributed approach is designed so as to exploit the the advantages of both the centralized and distributed paradigms in localization, by not being completely distributed, and using an optimal amount of global information.

Section 3.2 discusses previous work in this domain, particularly at the intersection of sensor network localization and distance geometry. Section 3.3 describes the problem setup and a variety of formulations for the sensor network localization problem, using the SDP relaxation model. In Section 3.4, we derive upper and lower bounds for the SDP objective function. The estimation accuracy of SDP based algorithms for noisy distance data is also discussed. Section 3.5 introduces the use of regularization terms in the SDP objective function in order to obtain lower dimensional solutions. Section 3.6 deals with the use of post processing local refinement methods using the SDP estimated solution as a starting point. In Section 3.7, we present the iterative distributed SDP method for solving localization problems that arise from networks consisting of a large number of sensors. Finally, experimental results that show that these methods significantly improve the performance of SDP based localization algorithms are presented in Section 3.8.

3.2 Related work

A great deal of research has been done on the topic of position estimation in ad-hoc networks, see [40, 72] for surveys. Most techniques use distance or angle measurements from a fixed set of reference or anchor nodes; see [29], [61], [69, 70, 71, 76]; or employ a grid of beacon nodes with known positions; see [19, 41].

Niculescu and Nath [61] proposed the "DV-Hop" and related "DV-Distance" and Euclidean approaches which is quite effective in dense and regular topologies. The anchor nodes flood their position information to all the nodes in the network. Each node then estimates its own position by performing a triangulation using this information. For more irregular topologies however, the accuracy can deteriorate to the radio range. The authors also investigate the use of angle information ("DV-Bearing") in for localization [62] using the information forwarding techniques described above.

Savarese et al. [69] present a 2 phase algorithm in which the start-up phase involves finding the rough positions of the nodes using a technique similar to the "DV-Hop" approach. The refinement phase improves the accuracy of the estimated positions by performing least squares triangulations using its own estimates and the estimates of the nodes in its own neighborhood. This method can accurately estimate points within one third of the radio range.

When the number of anchor nodes is high, the "iterative multilateration" technique proposed by Savvides et al. [70] yields good results. Nodes that are connected to 3 or more anchors compute their position by triangulation and upgrade themselves to anchor nodes. Now their position information can also be used by the other unknown nodes for their position estimation in the next iteration.

Howard et al. [41] and Priyantha et al. [67] have discussed the use of spring based relaxations that initially try to find a graph embedding that resembles the actual configuration and then modify the embedding to approach the actual one using a mass-spring based optimization to correct and balance errors.

While the above methods can be run in a distributed fashion, there also exist some centralized methods that offer more precise location determination by using global information. For centralized techniques, the available distance information between all the nodes must be present on a single computer. The distributed approach has the advantage that the techniques can be executed on the sensor nodes themselves thus removing the need to relay all the information to a central computer. This also affects the scalability and accuracy of the problems under consideration. Distributed techniques can handle much larger networks whereas centralized approaches yield higher accuracy by using global information.

Shang et al. [77] demonstrate the use of "multidimensional scaling" (MDS) in estimating positions of unknown nodes. Firstly, using basic connectivity or distance information, a rough estimate of relative node distances is made. Then classical MDS (which basically involves using a Singular Value decomposition) is used to obtain a relative map of the node positions. Finally an absolute map is obtained by using the known node positions. This technique works well with few anchors and reasonably high connectivity. For instance, for a connectivity level of 12 and 2% anchors, the error is about half of the radio range. The centralized version of the above mentioned technique does not work well when the network topology is irregular. So an alternative distributed MDS and patching technique is explored in [75, 76]. Here local clusters with regular topologies are solved separately and then stitched together subsequently.

Some other methods are based on minimizing some global error functions, which can be different when the model of uncertainty changes. Depending on the kind of optimization model being formulated, the characteristic and computation complexity varies. For example, the maximum likelihood estimation for sensor network localization problem is a non-convex optimization problem. Existing algorithms have limited success, even for small problem sizes, see [59, 58]. On the other hand, if we relax the distance constraints, the problem can be formulated as a second order cone program (SOCP) and cheap and scalable algorithms can be applied [29, 89]. However, the solution is acceptable only when the anchors are densely deployed on the boundary of the sensor network.

The use of convex optimization for sensor localization was first described by Doherty et al. [29]. However, as mentioned in the previous chapter, this technique yields good results only if the anchor nodes are placed on the outer boundary, since the estimated positions of their convex optimization model all lie within the convex hull of the anchor nodes. So if the anchor nodes are placed in the interior of the network, the position estimation of the unknown nodes will also tend to the interior, yielding highly inaccurate results. For example, with just 5 anchors in a random 200 node network, the estimation error is almost twice the radio range.

3.3 The SDP model with noise

Consider a sensor network in $\mathcal{R}^2$ with m anchors and n sensors. An anchor is a node whose location $a_k \in \mathcal{R}^2$, $k = n+1, \ldots, n+m$, is known, and a sensor is a node whose location $x_i \in \mathcal{R}^2$, $i = 1, \ldots, n$ is yet to be determined. In general, not all pairs of sensor/sensor and sensor/anchor distances can be measured, so the set of sensor/sensor and sensor/anchor pairs for which distances are known are denoted as $(i,j) \in \mathcal{N}_u$ and $(i,k) \in \mathcal{N}_a$, respectively. For a pair of sensors $(i,j) \in \mathcal{N}_u$, the distance estimate is denoted as d_{ij}. Similarly, for a sensor/anchor pair $(i,k) \in \mathcal{N}_a$, the distance measured is denoted as d_{ik}.

Mathematically, the localization problem can be stated as follows: given m anchor locations $a_k \in \mathcal{R}^2$, $k = n+1, \ldots, n+m$, and some inaccurate distance measurements d_{ij}, $(i,j) \in \mathcal{N}_u$ and d_{ik}, $(i,k) \in \mathcal{N}_a$, estimate as accurately as possible the locations of the n sensors $x_i \in \mathcal{R}^2$, $i = 1, \ldots, n$. One should note that this problem is in fact a special case of general distance geometry and graph realization problems in $\mathcal{R}^d$ where $d = 2$ (as discussed in Chapter 2), although in this case, we will have to deal with noisy distance data. We will discuss with the general noisy graph realization problem in $\mathcal{R}^d$ and illustrate its applicability to the sensor network localization problem.

3.3.1 Error minimization

A reasonable solution for the noisy graph realization problem should be to try to minimize the discrepancy in the estimated point locations to fit the given distance data. We will illustrate how the SDP relaxation can be extended to the following error minimization model:

$$\text{Minimize}_{x_1,\ldots,x_n \in \mathcal{R}^d} \sum_{(i,j)\in\mathcal{N}_u} \gamma_{ij} \left| \|x_i - x_j\|^2 - d_{ij}^2 \right| + \sum_{(i,k)\in\mathcal{N}_a} \gamma_{ik} \left| \|x_i - a_k\|^2 - d_{ik}^2 \right|, \qquad (3.1)$$

where $\gamma_{ij} > 0$ are given weights. By adding different weights to the distance constraints, we are essentially giving extra emphasis to those distance measurements for which we have higher confidence and vice versa. This would be particularly useful in cases where the distance measures are noisy but there is some prior information about which of those are more reliable. In such cases, we can assign higher weights to the distance constraints corresponding to the more reliable measures.

It should be borne in mind that the SDP approach is extensible to a larger class of distance measures, formulations and cost functions. Different types of cost functions have been described in weighted multidimensional scaling literature, see [26]. In fact, weighted multidimensional scaling has been applied to the sensor network localization problem in [25], and the cost function described above is a specific case of the general cost function described therein. The key lies in how to use SDP relaxations in solving this general class of (nonconvex) problems involving Euclidean distances. This particular cost minimization problem also admits an SDP relaxation, which we will use for illustrating our general relaxation technique. The effect of the choice of the cost function and weights is discussed in more detail in the next subsection.

For convenience, we let $g_{ij} = [e_i; -a_j]$ for $(i,j) \in \mathcal{N}_a$ and $g_{ij} = [e_{ij}; \mathbf{0}]$ for $(i,j) \in \mathcal{N}_u$. We further let $\mathcal{E} = \mathcal{N}_u \cup \mathcal{N}_a$. Then (3.1) can be rewritten in matrix form as:

$$\begin{array}{ll} \text{Minimize} & \displaystyle\sum_{(i,j)\in\mathcal{E}} \gamma_{ij} \left| g_{ij}^T \begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} g_{ij} - d_{ij}^2 \right| \\ \text{subject to} & Y = X^T X. \end{array} \tag{3.2}$$

Once again, problem (3.2) is a non-convex optimization problem and so, our method is to relax it to a semidefinite program by relaxing the constraint $Y = X^T X$ to $Y \succeq X^T X$. From [15], we know that the last matrix inequality is equivalent to the condition that $\begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} \succeq 0$. Let $\mathcal{K} = \{Z \in \mathcal{R}^{(n+d)\times(n+d)} \;:\; Z = \begin{pmatrix} Y & X^T \\ X & I_d \end{pmatrix} \succeq \mathbf{0}\}$. Then the SDP relaxation of problem (3.2) can be written as the following SDP:

$$\begin{array}{ll} \text{Minimize} & g(Z; D) := \displaystyle\sum_{(i,j)\in\mathcal{E}} \gamma_{ij} \left| g_{ij}^T Z g_{ij} - d_{ij}^2 \right| \\ \text{subject to} & Z \in \mathcal{K}. \end{array} \tag{3.3}$$

3.3.2 Choice of cost functions and weights

It is important to recognize that the choice of the cost/penalty function is crucial in providing the statistically 'best' estimate of the point positions. Chapter 7 of the book [16] provides a discussion of statistical estimation in the context of convex optimization. In a maximum likelihood framework, the cost function would be dictated by the prior noise distribution that is assumed for the given distance measurements. In other words, we try to find the point positions that minimize the log-likelihood function given the distances, and the log-likelihood function is determined by the kind of noise model we assume. For example, in the formulation (3.1), the noise model being considered would be an additive Laplacian noise on the squares of the distances:

$$\begin{aligned} d_{ij}^2 &= \|x_i - x_j\|^2 + \epsilon_{ij} \\ p(\epsilon_{ij}) &= e^{-\gamma_{ij}|\epsilon_{ij}|}, \end{aligned} \tag{3.4}$$

where $p(z)$ is the probability density function of random variable z.

The noise model used mostly in this chapter is multiplicative normal noise:

$$\begin{aligned} d_{ij} &= \|x_i - x_j\||1 + \epsilon_{ij}| \\ \epsilon_{ij} &\sim N(0, \sigma_{ij}^2). \end{aligned} \tag{3.5}$$

For $1 + \epsilon_{ij} \geq 0$, which is usually the case, the noise model can be expressed as additive Gaussian noise

$$\begin{aligned} d_{ij} &= \|x_i - x_j\| + \epsilon_{ij}^m \\ \epsilon_{ij}^m &\sim N(0, \|x_i - x_j\|^2 \sigma_{ij}^2). \end{aligned} \tag{3.6}$$

The MLE problem in this case would be

$$\begin{aligned} \text{Minimize}_{x_1,\ldots,x_n \in \mathcal{R}^d} \quad & \sum_{(i,j)\in\mathcal{N}_u} \frac{1}{\|x_i - x_j\|^2 \sigma_{ij}^2} (\|x_i - x_j\| - d_{ij})^2 \quad + \\ & \sum_{(i,k)\in\mathcal{N}_a} \frac{1}{\|x_i - a_k\|^2 \sigma_{ik}^2} (\|x_i - a_k\| - d_{ik})^2. \end{aligned} \tag{3.7}$$

Since the exact distances (or maybe even the noise variances are not known), suitable approximations of them may be used in choosing the weights. For example, in the above problem, we put weights $\gamma_{ij} = \frac{1}{\sigma_{ij}^2 d_{ij}^2}$, to approximate the actual weights applied to the different terms of the cost function.

In fact, by using these approximate weighting schemes, we can also develop convex approximations of log-likelihood functions for more sophisticated noise models. Consider the case of a log-normal noise model where

$$\begin{aligned} d_{ij} &= \|x_i - x_j\| e^{\epsilon_{ij}} \\ \epsilon_{ij} &\sim N(0, \sigma_{ij}^2). \end{aligned} \tag{3.8}$$

The MLE problem corresponding to this noise model is

$$\begin{aligned} \text{Minimize}_{x_1,\ldots,x_n \in \mathcal{R}^d} \quad & \sum_{(i,j)\in\mathcal{N}_u} \frac{1}{\sigma_{ij}^2} (\log\|x_i - x_j\| - \log d_{ij})^2 \quad + \\ & \sum_{(i,k)\in\mathcal{N}_a} \frac{1}{\sigma_{ik}^2} (\log\|x_i - a_k\| - \log d_{ik})^2. \end{aligned} \tag{3.9}$$

At first glance, this is again a complicated nonconvex function, but we can approximate it by using its first order Taylor expansion. The term $\log\|x_i - x_j\| - \log d_{ij}$ can be expressed as $\log\left(1 + \frac{\|x_i - x_j\| - d_{ij}}{d_{ij}}\right)$. Assuming that the error $\|x_i - x_j\| - d_{ij} << d_{ij}$,

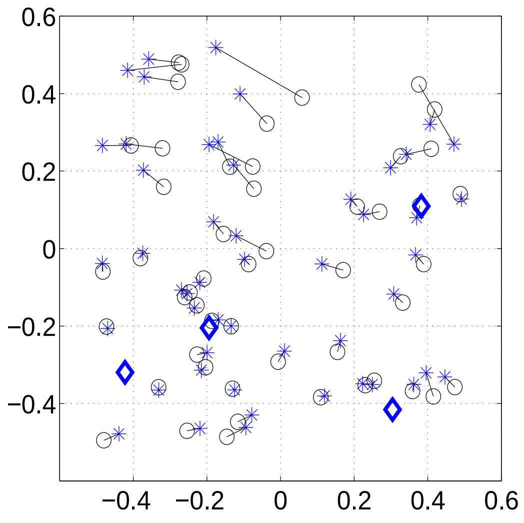
$$\log\left(1 + \frac{\|x_i - x_j\| - d_{ij}}{d_{ij}}\right) \approx \frac{\|x_i - x_j\| - d_{ij}}{d_{ij}}.$$

So the cost function in problem (3.9) can be approximately expressed as

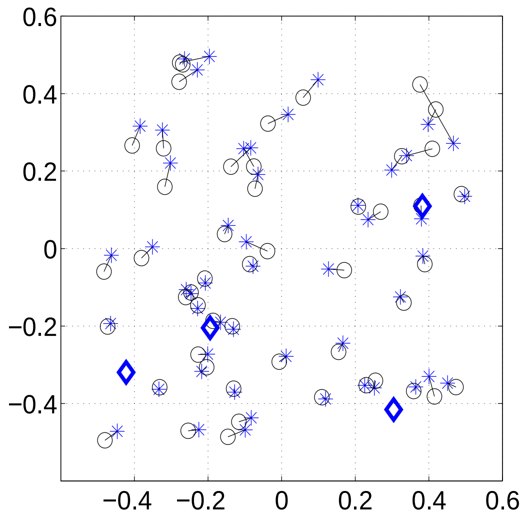
$$\sum_{(i,j)\in\mathcal{N}_u} \frac{1}{\sigma_{ij}^2 d_{ij}^2} (\|x_i - x_j\| - d_{ij})^2 + \sum_{(i,k)\in\mathcal{N}_a} \frac{1}{\sigma_{ik}^2 d_{ik}^2} (\log\|x_i - a_k\| - \log d_{ik})^2. \tag{3.10}$$

It turns out that with this approximation, we end up with the same weights for both the multiplicative (3.7) and log-normal (3.10) noise models. This can be reconciled by the fact that the multiplicative and log-normal noises are approximately the same (if we neglect the higher order terms in the Taylor expansion of $e^{\epsilon_{ij}}$ for log-normal noise).

Example 3.1. *In Figure 3.1, we illustrate the improvement that choosing intelligent weights can provide as opposed to the naive scheme of using equal weights on all terms. The problem setup is the same as for the Example 2.1, except that the distances are noisy as described in model* (3.5)*. In the intelligent weighting case, the terms are scaled based on the distance. As a result, long range distances, which are less reliable are also given less weight. The intelligent weighting scheme provides better estimation for most of the points.*



(a) Naive Weighting



(b) Intelligent Weighting

Figure 3.1: Effect of intelligent weighting in the multiplicative noise model, 50 points, 4 anchors, 20% multiplicative noise.

3.4 Effect of noisy distance data

In this section, we analyze the effect of noisy distance measures on the SDP objective function value, and subsequently, on the point estimations. The SDP objective gives us an insight into the actual estimation error that can be introduced and its dependence on the noise. We assume that the distances d_{ij} and d_{ik} are perturbed by random noises ϵ_{ij} and ϵ_{ik}:

$$\begin{aligned} d_{ij} &= \hat{d}_{ij}|1+\epsilon_{ij}|, \quad (i,j) \in \mathcal{N}_u \\ d_{ik} &= \hat{d}_{ik}|1+\epsilon_{ik}|, \quad (i,k) \in \mathcal{N}_a, \end{aligned}$$

where $\hat{d}_{ij}$ is the true distance between points i and j, and $\hat{d}_{ik}$ is the true distance between point i and anchor k. We further assume that $E(\epsilon_{ij}) = 0$, $E(\epsilon_{ik}) = 0$. We take the absolute value because the noise maybe higher than 1, and so we would get a negative value for distance, which is not meaningful.

Let $\mathcal{I} = \{(i,j) : n+1 \le i \le j \le n+d\}$ and $E_{ij} = (e_i e_j^T + e_j e_i^T)/2$ for $(i,j) \in \mathcal{I}$. Also let δ_{ij} be the Kronecker delta. The dual of SDP (3.3) is as follows:

$$\begin{array}{ll} \text{Maximize} & \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} d_{ij}^2 + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij}\delta_{ij} \\ \text{subject to} & \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} g_{ij} g_{ij}^T + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij} E_{ij} \ \preceq \ 0, \\ & |\lambda_{ij}| \ \le \ \gamma_{ij} \ \forall \ (i,j) \in \mathcal{N}_u. \end{array} \tag{3.11}$$

Let $\hat{v}$ and $\widehat{Z}$ be the optimal objective value and minimizer of (3.3) for the problem $(\mathcal{V},\mathcal{A};\widehat{D})$, respectively. Let $\hat{\lambda}$ be the maximizer of (3.11) for the problem $(\mathcal{V},\mathcal{A};\widehat{D})$. We assume that the problem $(\mathcal{V},\mathcal{A};\widehat{D})$ is localizable, *i.e.*, $\hat{v}=0$ and $g_{ij}^T\widehat{Z}g_{ij} = \hat{d}_{ij}^2$ for all $(i,j)\in\mathcal{E}$.

Proposition 3.1. *Consider the problem $(\mathcal{V},\mathcal{A};D)$ and assume that Assumption 1 holds. Let v_* be the optimal objective value of (3.3) for this problem. We have the following results:*

1.

$$E[v_*] \ \le \sum_{(i,j)\in\mathcal{E}} \gamma_{ij}\hat{d}_{ij}^2 E\left|2\epsilon_{ij}+\epsilon_{ij}^2\right|. \tag{3.12}$$

2.

$$E[v_*] \ge \sup\left\{ \begin{array}{l} \sum_{(i,j)\in\mathcal{E}} \lambda_{ij}\hat{d}_{ij}^2(1+E[\epsilon_{ij}^2]) + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij}\delta_{ij} : \\ \lambda \ \ \textit{is feasible for} \ \ (3.11) \end{array} \right\}.$$

3. If $\epsilon_{ij}\sim N(0,\tau^2)$ for $(i,j)\in\mathcal{E}$, then

$$E[v_*] \ \le \ (1.6\tau+\tau^2)\sum_{(i,j)\in\mathcal{E}} \gamma_{ij}\hat{d}_{ij}^2. \tag{3.13}$$

Proof. 1. Since $\widehat{Z}\in\mathcal{K}$, it is clear that

$$v_* \ \le \ g(\widehat{Z};D) \ \le \sum_{(i,j)\in\mathcal{E}} \gamma_{ij}\left|\hat{d}_{ij}^2(2\epsilon_{ij}+\epsilon_{ij}^2)\right|,$$

and (3.12) follows readily.

2. For a fixed λ that is feasible for (3.11), we have by weak duality that

$$v_* \;\geq\; \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} d_{ij}^2 + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij}\delta_{ij}.$$

By taking expectation on both sides, we get

$$\begin{aligned} E[v_*] \;&\geq\; \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} E[d_{ij}^2] + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij}\delta_{ij} \\ &=\; \sum_{(i,j)\in\mathcal{E}} \lambda_{ij} \hat{d}_{ij}^2(1+E[\epsilon_{ij}^2]) + \sum_{(i,j)\in\mathcal{I}} \lambda_{ij}\delta_{ij}. \end{aligned}$$

Since the above inequality holds for any λ that is feasible for (3.11), the required inequality follows.

3. The inequality in (3.13) follows from (3.12) by noting that $E\left|2\epsilon_{ij}+\epsilon_{ij}^2\right| \leq \frac{4}{\sqrt{2\pi}}\tau+\tau^2 \leq 1.6\tau+\tau^2$.

□

Proposition 3.1(c) indicates that the expected optimal SDP objective value $E[v_*]$ can potentially grow as fast as the standard deviation τ of the noises ϵ_{ij}. The following example and Figure 3.2(a) shows empirically that $E[v_*]$ indeed can grow as fast as the upper bound predicted in Proposition 3.1(c). We will examine the behavior of the bounds through an example. Before doing so, we describe the simulation setup that is used throughout this chapter.

First, n points $\{\hat{x}_i : i=1,\dots,n\}$ are generated from a uniform random distribution in the unit square $[-0.5,0.5]\times[-0.5,0.5]$ via the MATLAB command: `rand('state',0); x = rand(2,1) - 0.5`. Then the edge set $\mathcal{N}_u$ is generated by considering only pairs of points that have distances less than a given Radio range R, *i.e.*,

$$\mathcal{N}_u \;=\; \{(i,j) \,:\, \|\hat{x}_i-\hat{x}_j\| \leq R,\; 1\leq i<j\leq n\}.$$

If there are anchors, the edge set $\mathcal{N}_a$ is similarly defined by

$$\mathcal{N}_a \;=\; \{(i,k) \,:\, \|\hat{x}_i-a_k\| \leq R,\; 1\leq i\leq n,\; n+1\leq k\leq n+m\}.$$

We assume that the distances d_{ij} and d_{ik} are perturbed by random noises ϵ_{ij} and ϵ_{ik} as follows:

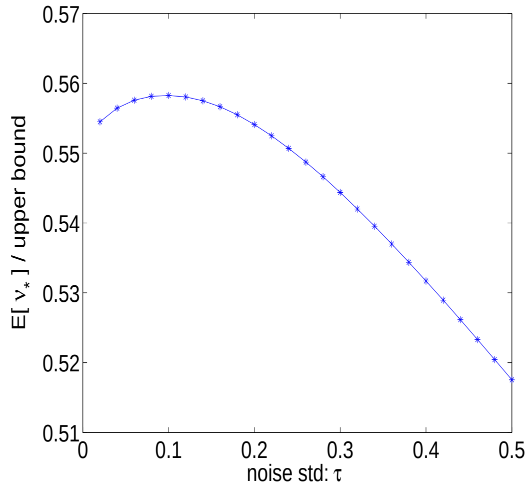
$$\begin{aligned} d_{ij} &= \hat{d}_{ij}|1+\tau\epsilon_{ij}|, \quad (i,j)\in\mathcal{N}_u \\ d_{ik} &= \hat{d}_{ik}|1+\tau\epsilon_{ik}|, \quad (i,k)\in\mathcal{N}_a, \end{aligned}$$

where $\hat{d}_{ij}$ is the true distance between point i and j, and $\hat{d}_{ik}$ is the true distance between point i and anchor k. We assume that ϵ_{ij}, ϵ_{ik} are independent standard Normal random variables. In future examples, we will express the noise in terms of percentage, *i.e.*, a value of 0.1 for τ corresponds to 10% noise. We also define the Root Mean Square Distance (RMSD) error metric to measure the accuracy of the estimated positions $\{x_i : i=1,\dots,n\}$:

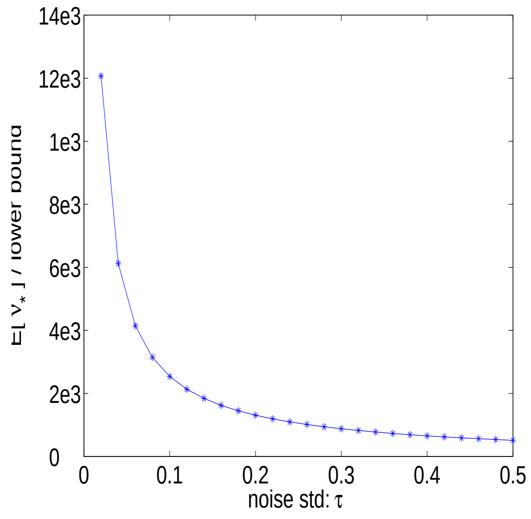
$$RMSD = \frac{1}{\sqrt{n}}\left(\sum_{i=1}^{n}\|\hat{x}_i - x_i\|^2\right)^{1/2}. \tag{3.14}$$

Throughout this section, the weights γ_{ij} in (3.1) are set to 1.

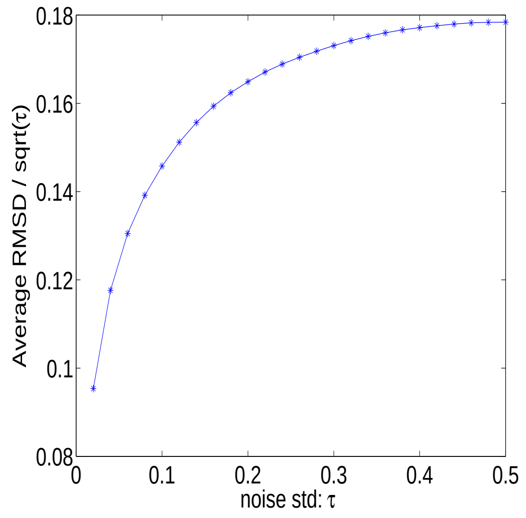
Example 3.2. *For the example in Figure 3.2, we use a network of 60 points, 4 anchors placed at the positions* $(\pm 0.45, \pm 0.45)$. *and a radio range* $R = 0.3$. *For each given* τ, *we generated* 50 *samples of* $\{d_{ij} : (i,j)\in\mathcal{E}\}$. *We solve the corresponding SDP* (3.3) *for these* 50 *samples to estimate the the expected optimal SDP objective value* $E[v_*]$. *In Figure 3.2(a), the ratio between the estimated* $E[v_*]$ *and the upper bound given in Proposition 3.1 is plotted against the standard deviation* τ *of the noises. Figure 3.2(b) plots the ratio between the estimated* $E[v_*]$ *and the corresponding lower bound for this model. As can be seen from the graphs, the upper bound is reasonably tight and is a good indicator of the expected objective value. The lower bound however is very loose and does not reveal too much information. The construction of tighter lower bounds is being investigated. Figure 3.2(c) plots the ratio between the average RMSD error and* $\sqrt{\tau}$ *obtained from the SDP relaxation of* (3.1). *Observe that the average RMSD error roughly grows like* $\tau^{1.5}$ *when* τ *is small and* $\sqrt{\tau}$ *when* τ *is large. The RMSD grows at a higher rate when* τ *is large, because the distance measures become so erroneous that the structure of the graphs is altered significantly and they become non-uniquely localizable.*



(a) $E[v_*]$ versus the upper bound



(b) $E[v_*]$ versus the lower bound



(c) Average RMSD versus $\sqrt{\tau}$

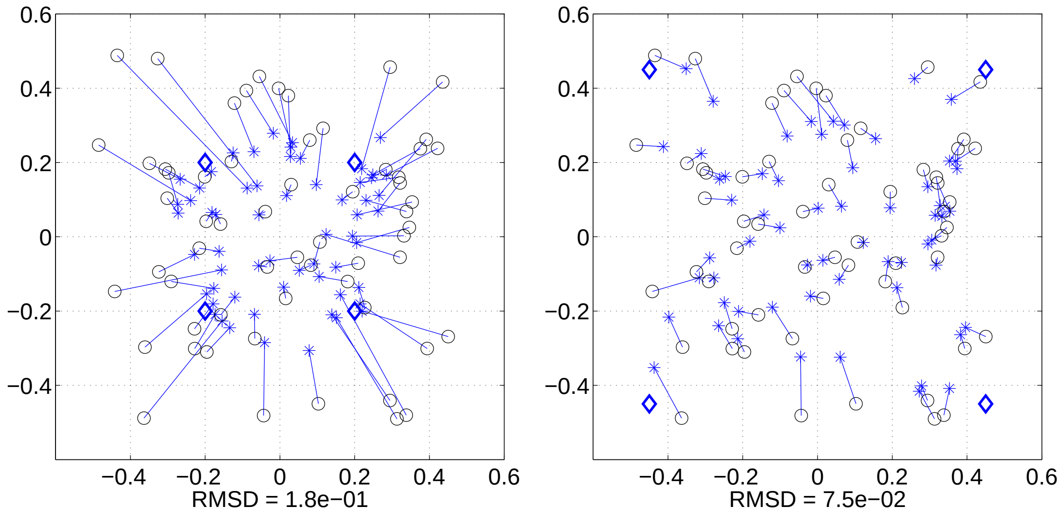
Figure 3.2: Ratio between $E[v_*]$ and the upper and lower bounds in Proposition 3.1; and the ratio between average RMSD and $\sqrt{\tau}$ obtained from the SDP relaxation of (3.1).

3.4.1 The high rank property of SDP relaxations

It was proved that when there is no noise in the measured distances, the SDP approach can be used to solve the localization problem in polynomial time under a uniquely localizable assumption [79]. When the distance measurements have errors, this is no longer the case. The distance constraints usually contradict each other and there is no localization in $\mathcal{R}^d$. In other words, $Y \neq X^T X$. However, since the SDP approach relaxes the constraint $Y = X^T X$ into $Y \succeq X^T X$, it is still possible to locate the sensor in a higher dimensional space (or choose a Y with a higher rank) and to make the objective function value zero. The optimal solution in a higher dimensional space always results in a smaller objective function value than the one constrained in $\mathcal{R}^d$. Furthermore, the 'max-rank' property [36] implies that solutions obtained through interior point methods for solving SDPs converge to the maximum rank solutions. Hence, because of the relaxation of the rank requirement, the solution is 'lifted' to a higher dimensional space. For example, imagine a rigid structure consisting of set of points in a plane (with the points having specified distances from each other). If we perturb some of the specified distances, the configuration may need to be readjusted by setting some of the points outside the plane.

A main research topic is how to round the higher-dimensional (higher rank) SDP solution into a lower-dimensional (rank-2) solution. One way is to ignore the augmented dimensions and use the projection X^* as a suboptimal solution, which is the case in [14]. However, the projection typically leads to points getting 'crowded' together as shown in Figure 3.3(a) (Imagine the projection of the top vertex of a pyramid onto its base). This is because a large contribution to the distance between 2 points could come from the dimensions we choose to ignore. This problem can be reduced to some extent by placing anchors at the perimeter like in Figure 3.3(b). Since these anchors are constrained to be in a plane and are stretched out, the other points cannot fold together into higher dimensions and still maintain the distance constraints with these anchors.

Example 3.3. *An example of the effect of anchor placement is demonstrated in Figure 3.3 where for the same network of points and different anchor positions, the estimations are drastically different. The (blue) diamonds refer to the positions of the anchors; (black) circles to the original locations of the unknown sensors; and (blue) asterisks to their estimated positions from the solution of the SDP* (3.3)*. The discrepancies between the original and the estimated points are indicated by solid lines.*



(a) 4 inner anchors: $(\pm 0.2, \pm 0.2)$ (b) 4 outer anchors: $(\pm 0.45, \pm 0.45)$

Figure 3.3: 60 sensors, 4 anchors, Radio range=0.3, 20% Normal multiplicative noise. The positions are estimated from the solution of the SDP (3.3).

The use of bounding away constraints (that correspond to points that must be far away from each other) can also reduce the problem of crowding. However, there may be too many such constraints for all pairs of points thus increasing the computational complexity of the technique. Warm start approaches have been discussed in [13] where bounding away constraints are added to a second SDP if they are being violated in the initial SDP (without any bounding constraints).

An intelligent anchor placement design that ensures lower dimensional embedding and intelligent selection of bounding away constraints are part of our future research. But, we often may not have the luxury of being able to place anchors strategically, and the high cost of solving multiple SDPs with extra constraints may be undesirable. This motivates the use of regularization methods. In [93], regularization terms have been incorporated into SDPs arising from kernel learning in nonlinear dimensionality reduction. The purpose is to penalize folding and try to find a stretched map of the set of points, while maintaining local distance relations. This idea is also formally connected to the tensegrity theory for graph realization, *i.e.*, certain stretched graphs can always be realized in a lower dimensional space than those without stretching; see [80]. In the next section, we describe a similar technique to encourage lower dimensional solutions. We will also describe the use of a local refinement method to improve the SDP estimated solution. These methods turn out to be extremely effective and efficient.

3.5 Regularization

It has been observed by many researchers that when the distance data is noisy, the points estimated from (3.1) tend to crowd together towards the center of the configuration. Here we propose a strategy to ameliorate this difficulty. Let e be the n-dimensional vector of all ones, $\hat{e} = e/\sqrt{n+m}$, and $\hat{a} = \sum_{k=1}^{m} a_k/\sqrt{n+m}$. Let $\boldsymbol{a} = [\hat{e}; \hat{a}]$ and $A = [a_1, \ldots, a_m]$. Our strategy is to subtract the following regularization term from the objective function of (3.3) to prevent the estimated points from crowding together:

$$
\begin{aligned}
&\frac{\lambda}{n+m}\Big(\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|x_i - x_j\|^2 + \sum_{i=1}^{n}\sum_{k=1}^{m}\|x_i - a_k\|^2\Big) \\
&= \lambda\Big(\sum_{i=1}^{n}\|x_i\|^2 - \|X\hat{e} + \hat{a}\|^2\Big) + \lambda\Big(\frac{n}{n+m}\|A\|_F^2 + \|\hat{a}\|^2\Big) \\
&= \lambda(I - \boldsymbol{a}\boldsymbol{a}^T) \bullet Z + \lambda\Big(\frac{n}{n+m}\|A\|_F^2 + \|\hat{a}\|^2 - d\Big),
\end{aligned}
$$

where λ is a positive regularization parameter that is to be chosen. Ignoring the constant term, the regularized form of (3.3) is given by

$$
\begin{aligned}
\text{Minimize} \quad & g(Z; D) - \lambda(I - \boldsymbol{a}\boldsymbol{a}^T) \bullet Z \\
\text{subject to} \quad & Z \in \mathcal{K}.
\end{aligned}
\tag{3.15}
$$

The idea of regularization can be linked to tensegrity theory and realizability of graphs in lower dimensions. The notion of stress is used to explain this. Let us imagine the edges (distance measures) between the points as a set of springs through which the neighboring points exert forces on each other. By strategically placing some extra springs between chosen vertices in the graph, the force exerted on the different points can be varied. With a non-zero stress induced on the edges in the graph, we can stretch out the graph (It turns out that the dual multipliers of this problem are closely related to the non-zero stress values). The idea behind using non-zero stress for stretching is the following: for the configuration to remain in equilibrium, the total stress on a vertex must sum to zero. In order for the overall stress to cancel out completely, the graph must be in a low dimensional space, so that all the forces are resolved completely. For more details on this theory, see [80]. The choice of which particular edges to stretch is important in achieving low dimensional realizations, but it is a difficult, since it depends on the particular instance of the graph configuration provided. Techniques to select such edges can improve the robustness of the regularization technique and are a part of future research.

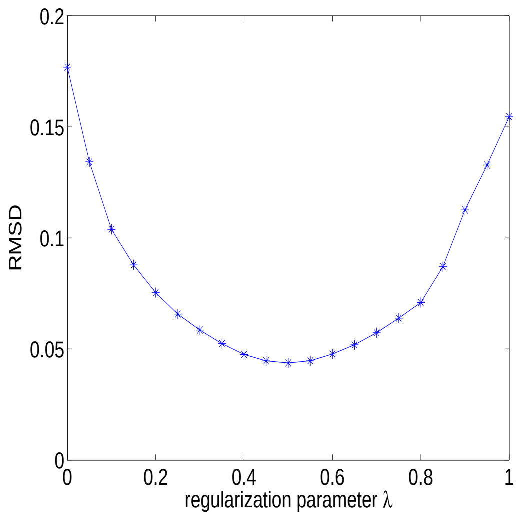
3.5.1 Choice of regularization parameter

An important issue in SDP (3.15) is the optimum choice of the parameter λ. If it is too low, the regularization term will not affect the SDP solution significantly enough. However, if it is chosen to be too high, the regularization term will overwhelm the error term and cause the SDP to either be infeasible or to yield a solution where the points are stretched out too far apart. The optimum choice of the parameter λ is dependent upon the size and geometry of the network and the distance information available. Currently, it is not known how to determine such an optimum value prior to solving the SDP for a particular network. We can start with $\lambda = 1$ and if the SDP is infeasible or the objective function has a high negative value below a predefined threshold, we scale λ by a factor less than 1 and solve a new SDP. The factor could be chosen to balance the magnitudes of the error term and the regularization term in SDP (3.15). While this method works satisfactorily, it suffers the disadvantage of possibly having to solve several SDPs.

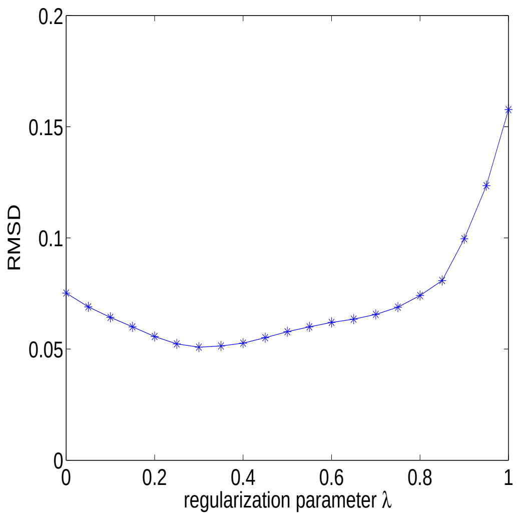
A heuristic choice of λ that we found to be reasonably effective is the following. Suppose Z^* is obtained from the non-regularized SDP (3.3). Let

$$\lambda^* \;=\; \frac{g(Z^*; D)}{(I - \boldsymbol{a}\boldsymbol{a}^T) \bullet Z^*}\ . \tag{3.16}$$

Our numerical experience show that the actual choice of λ should not be larger than λ^*, and $\lambda = \lambda^*$ typically works reasonably well. The heuristic choice of λ just described requires the solution of two SDPs. It was observed from simulations that within a certain range of network sizes and radio ranges, a particular range of λ tends to be ideal. By setting the value of λ to be in this range prior to localization, it is very likely that an accurate solution can be obtained from the SDP (3.15) in the first attempt. However this behavior needs to be investigated in more detail in order to obtain a suitable choice of λ that guarantees a good estimate. Figure 3.4 shows the RMSD error versus the regularization parameter λ for the problem in Figure 3.3. Observe that for a reasonably wide range of λ, the RMSD error is smaller than the RMSD error obtained from the non-regularized SDP in (3.3), which corresponds to the special case $\lambda = 0$.



(a) 4 inner anchors



(b) 4 outer anchors

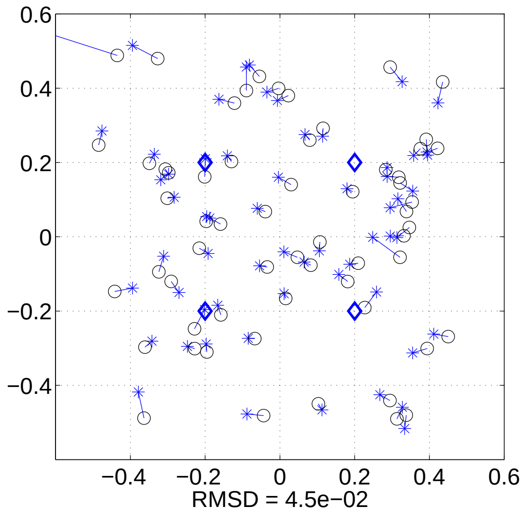
Figure 3.4: RMSD error of the estimated positions obtained from SDP with regularization parameter λ for the examples in Figure 3.3.

Example 3.4. *In the following example, we show the result of adding a regularization term for the cases discussed in Figure 3.3(a). The problem of crowding at the center as was seen before is eliminated as seen in Figure 3.5(a) for anchors placed in the interior. In Figure 3.5(b), we see that by choosing an appropriate value of λ, the regularization can be used to improve the solution even for the case where the anchors are in the exterior.*

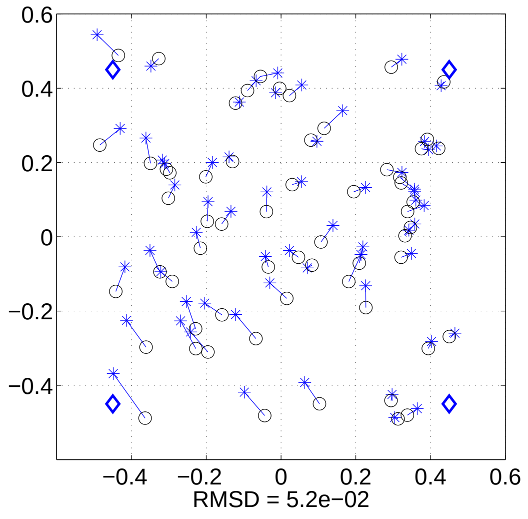
3.6 Local refinement

3.6.1 A gradient descent method

The positions estimated from the SDP relaxation (3.3) or (3.15) can further be refined by applying a local optimization method to (3.1). However, as the objective function in (3.1) is non-smooth, it is more convenient to illustrate the working of the local refinement method to the model described below. This is similar to the approximate minimization model for



(a) 4 inner anchors



(b) 4 outer anchors

Figure 3.5: Same as Figure 3.3, but for SDP with regularization. The regularization parameter λ is chosen to be value in (3.16), which equals to 0.551 and 0.365, respectively.

multiplicative (3.7) or log-normal noise (3.10).

$$\min_{X\in\mathcal{R}^{d\times n}} \left\{ \begin{array}{rcl} f(X) & := & \sum_{(i,j)\in\mathcal{N}_u} \gamma_{ij}\left(\|x_i - x_j\| - d_{ij}\right)^2 \\ & & + \sum_{(i,k)\in\mathcal{N}_a} \gamma_{ik}\left(\|x_i - a_k\| - d_{ik}\right)^2 \end{array} \right\}. \tag{3.17}$$

The method we suggest to improve the SDP estimated solution is to move every sensor location along the negative gradient direction of the function $f(X)$ in (3.17) to reduce the error function value. Recall that $\mathcal{N}_{ui} = \{j : (i,j) \in \mathcal{N}_u\}$ and $\mathcal{N}_{ai} = \{k : (i,k) \in \mathcal{N}_a\}$. By using the fact that $\nabla_x\|x - b\| = (x - b)/\|x - b\|$ if $x \neq b$, it is easy to show that for the objective function $f(X)$ in (3.17), the gradient $\nabla_j f$ with respect to sensor x_i is given by:

$$\begin{aligned} \nabla_i f(X) \; = \; & 2\sum_{j\in\mathcal{N}_{ui}} \gamma_{ij}\left(1 - \frac{d_{ij}}{\|x_i - x_j\|}\right)(x_i - x_j) \\ & + 2\sum_{k\in\mathcal{N}_{ai}} \gamma_{ik}\left(1 - \frac{d_{ik}}{\|x_i - a_k\|}\right)(x_i - a_k). \end{aligned} \tag{3.18}$$

Notice that $\nabla_i f$ only involves sensors and anchors that are connected to x_i. Thus $\nabla_i f$ can be computed in a distributed fashion. Let

$$X(\alpha) \;=\; [x_1 - \alpha\nabla_1 f(X), \,\ldots,\, x_n - \alpha\nabla_n f(X)]. \tag{3.19}$$

By choosing the step size $\alpha \in (0,1]$ appropriately, the function value $f(X(\alpha))$ can be reduced. In our method, the step size is chosen by a simple back-tracking procedure: starting with $\alpha = 1$, if $f(X(\alpha)) < f(X)$, set the next iterate to be $X(\alpha)$ and stop; else, reduce α by half, and recurse.

Example 3.5. *Consider the example presented in Figure 3.6. These are the results of the gradient method applied to the network of 60 sensors and 4 anchor nodes, a radio range of 0.30 and 20% noise, as discussed in Figure 3.3(b). The SDP estimated positions shown in Figure 3.3(b) are used as the initial solution. The anchor points are indicated by the (blue) diamonds. The SDP estimated positions are indicated by the (blue) crosses in Figure 3.6(a). Figure 3.6(a) also shows the update trajectories in* 50 *iterations. The (blue) asterisks indicate the final positions of the sensors; the trajectories are indicated by (blue) lines. It can be observed clearly that most sensors are moving toward their actual locations marked by the (black) circles. The final positions after* 50 *gradient steps are plotted in Figure 3.6(b) which is a more accurate localization. To demonstrate that the refined localization is indeed better, we can compute the objective function at every iteration to see if its value is reduced. In Figure 3.6(c) the objective function values vs number of gradient steps are plotted. One can see that in the first* 15 *steps the objective function value drops rapidly, and then the curve levels off. This demonstrates that the gradient search method does improve the overall localization result. A natural question is how good the new localization is. To answer this question we need a lower bound for the objective function value. One trivial lower bound of the objective function value is* 0*, but a tighter lower bound is given by the optimal objective value of the SDP relaxation of* (3.17)*. In this case, the SDP objective value is about* 0.376*, plotted in Figure 3.6(c) in a dashed line, and the gradient search method produces an objective function value of about* 0.462*. Thus an error gap of* 0.086 *of sub-optimality is obtained, which is about than 23% of the lower bound provided by the optimal SDP objective value. Finally, we observe that the gradient descent method is a local optimization method that generally does not deliver the global optimal solution of a non-convex problem, unless a good starting iterate is available. The sensor network localization problem based on* (3.17) *is a non-convex optimization problem. Hence a pure gradient descent method would not work. To see this, another experiment is performed. We use random starting points as the initial sensor locations and update them by the same rule* (3.19)*. The updated trajectories are shown in Figure 3.6(d). Most of these sensors do not converge to their actual positions. Comparing it to the result in Figure 3.6(b), we can see that the use of the SDP solution as the initial localization makes all the difference. The effect of the initial position is also illustrated through the results of applying the gradient method to the case where the anchors are in the interior (as seen in Figure 3.3(a)). The results of the gradient method are shown in Figure 3.7. Clearly, due to bad initialization, some points are unable to converge close to their actual locations.*

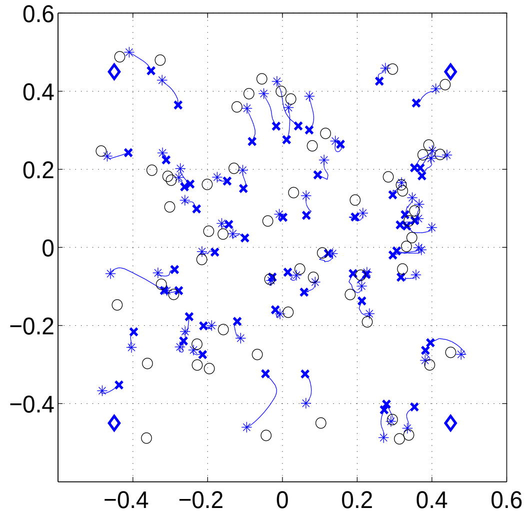
Some finer points need to be mentioned. First, since the regularized SDP is already quite accurate, it is a better starting point for the refinement and the gradient method offers some minor improvement. For the non regularized case, the initial estimate is highly erroneous to begin with and the gradient method can offer significant improvement but still may have high error. A combination of the regularized and gradient method yields very accurate estimates even in the presence of high noise. Secondly, because the distance measurements have noises, convergence to the true location should not be expected. Hence, although some of the trajectories do not converge to their corresponding true locations, this does not mean that the gradient method is not working. Actually, from the objective function value, we know that the updates work. We can also use an exact line-search method to choose the step size α to guarantee a drop in the objective value.

3.6.2 Other local refinement methods

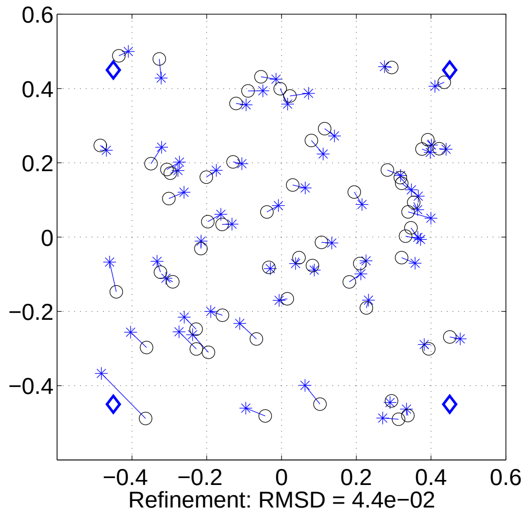
The use of more sophisticated optimization techniques such as approximate or truncated Newton methods (using Preconditioned Conjugate Gradient) can provide far better performance than a simple gradient descent technique. By exploiting the structure in the problem, these methods are able to find better descent directions which can be computed efficiently and take far fewer iterations to converge.

The Newton method attempts to find the descent direction which is the solution of the following system of equations:

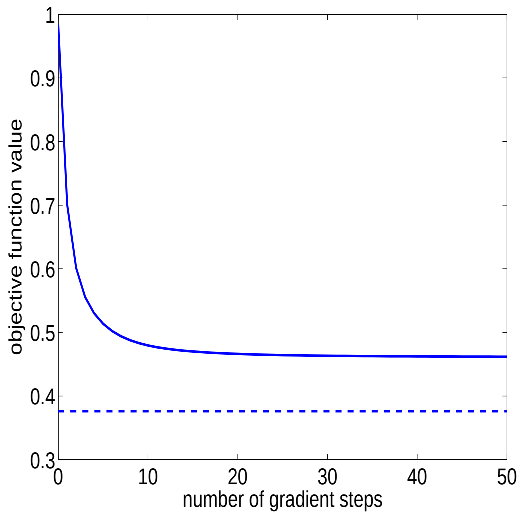
$$\nabla^2 f(X)\delta X_{nt} = -\nabla f(X), \tag{3.20}$$



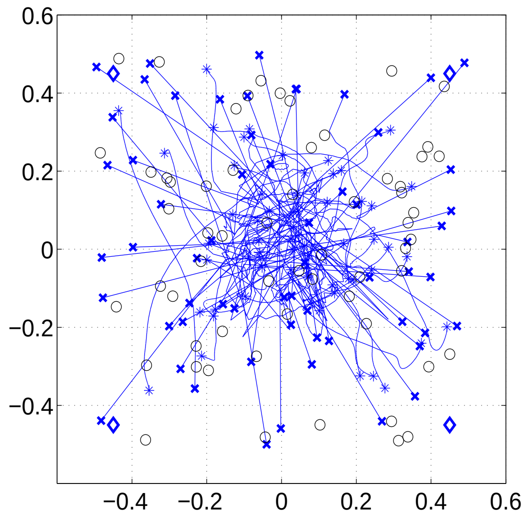
(a) Gradient search trajectories



(b) Gradient result, 50 iterations

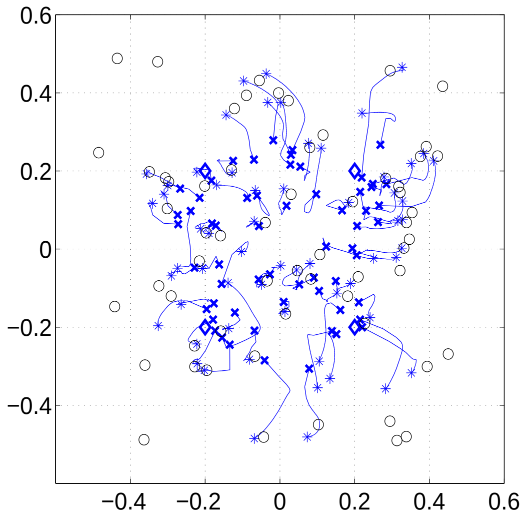


(c) The sum of error squares vs. number of gradient search steps

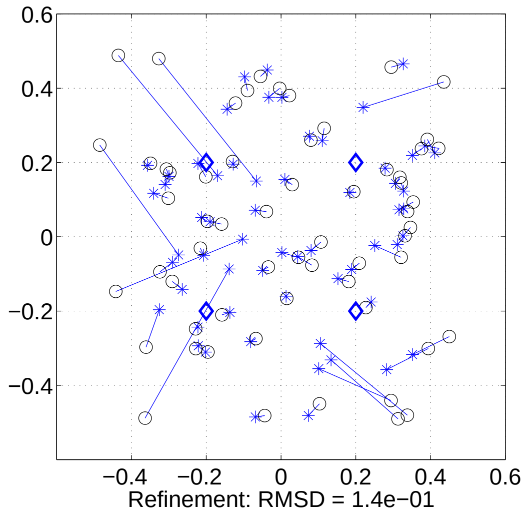


(d) Result of gradient search with random initial starting point

Figure 3.6: Refinement through gradient descent method, for the example in Figure 3.3(b).



(a) Gradient search trajectories



(b) Gradient result, 50 iterations

Figure 3.7: Refinement through gradient method, for case in Figure 3.3(a).

where $\nabla^2 f(X)$ is the Hessian of the cost function $f(X)$. Consider the Hessian in this case:

$$\begin{aligned}\nabla^2_{ii} f(X) &= 2\sum_{j\in\mathcal{N}_{ui}} \gamma_{ij} + 2\sum_{k\in\mathcal{N}_{ai}} \gamma_{ik} \\ \nabla^2_{ij} f(X) &= -2I_{j\in\mathcal{N}_{ui}}\gamma_{ij},\end{aligned} \tag{3.21}$$

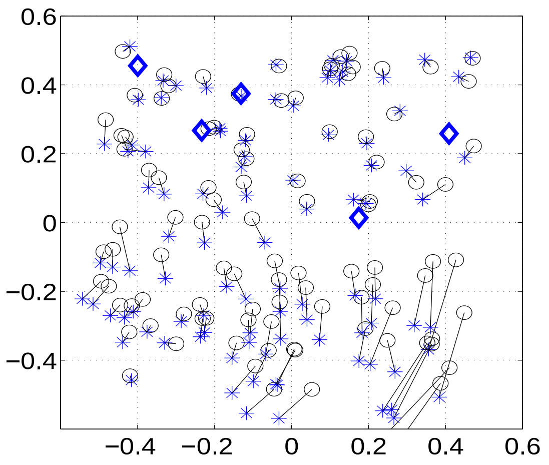
where $I_{j\in\mathcal{N}_{ui}} = 1$ if $j \in \mathcal{N}_{ui}$ and zero otherwise. Clearly in this case, the Hessian (3.21) is constant over all X. It is almost a weighted Laplacian of the underlying graph (extra terms corresponding to anchors are added to the diagonal terms), and very localized and nearly diagonally dominant in nature. Problems with this kind of structure in the Hessian lend themselves very well to approximate or truncated Newton type methods described in [56, 64]. Such methods have been applied extensively to solving large scale problems in statistics, signal processing, and circuit design, see [45, 48, 50].

The descent direction can be computed by a truncated Newton method, which corresponds to solving the Newton equation using the Preconditioned Conjugate Gradient method [47, 64]. The PCG method is iterative in nature, and as the term 'truncate' suggests, the PCG method is terminated after only a few iterations, sometimes far earlier than a particularly accurate solution of the system of equations has been obtained. A good search direction can also still be found quite often if instead of using the actual Hessian for computing the Newton step, an approximation of it is used (which is why it is called an approximate or inexact Newton method). In other words, we solve $\tilde{H}\delta\tilde{X} = -\nabla f$, where $\tilde{H}$ is a suitable approximation of $\nabla^2 f$. The simplest such approximations could simply be a diagonal or band of $\nabla^2 f$. Using these techniques, not only can the search directions be computed fast, but they are still good enough that the solution quickly converges to an optimal point in very few iterations. The idea is that while solving the exact Newton system may require fewer iterations, if we use the approximate or truncated Newton methods, the effort per iteration may be smaller. In cases where the Hessian has special structure, we can compute good search directions very quickly and efficiently. Using these ideas, we can solve much larger problems which are otherwise intractable for the exact Newton method. This is also better than the gradient descent technique, where many iterations might be needed, when the initial point provided is not close to the global optimum.

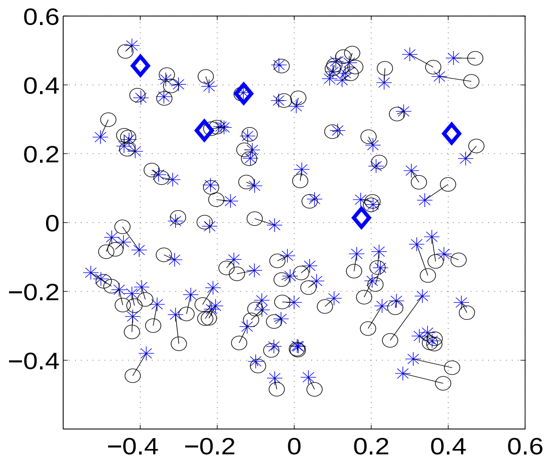
In our experience, simple approximate Newton methods (like using just the diagonal approximation of the Hessian) often do not provide good search directions in this problem, and the estimation can even be worse than using the simple gradient approach. A better approach is to use truncated PCG to solve the exact Newton equation 3.20. Choosing a good preconditioner applies a change of coordinates to the Newton system being solved so as to make the conjugate gradient step converge faster. By preconditioning with the matrix $M = \operatorname{diag}(H)$ and truncating the number of PCG iterations, a truncated Newton method can provide very good search directions.

Example 3.6. *Figure 3.8 is an example of a network of 100 points and 10 anchors, with a radio range of 0.175 and 20% multiplicative noise, where the truncated Newton method offers significant improvement over a gradient descent method.*

Figure 3.9 shows how the objective function value falls for this problem instance in Example 3.6 with every iteration of the different methods. The truncated Newton method can be tuned by controlling the number of iterations and the tolerance (both in the PCG and backtracking search steps), and the tradeoff between accuracy and computation time can be balanced as required. This is particularly encouraging for large problem sizes, where careful tuning can allow us to quickly and accurately solve the original nonconvex problem, given a good starting point from the SDP. To observe how the truncated Newton method behaves, the PCG method is truncated at both 10 steps and 50 steps. It is observed in this case that



(a) Gradient, 40 iterations, RMSD = 0.0778



(b) Truncated Newton, 50 PCG iterations, 40 iterations, RMSD =0.0491

Figure 3.8: Comparison of truncated Newton and Gradient Descent for a network of 100 points and 5 anchors, Radio range=0.175, 20% multiplicative noise.

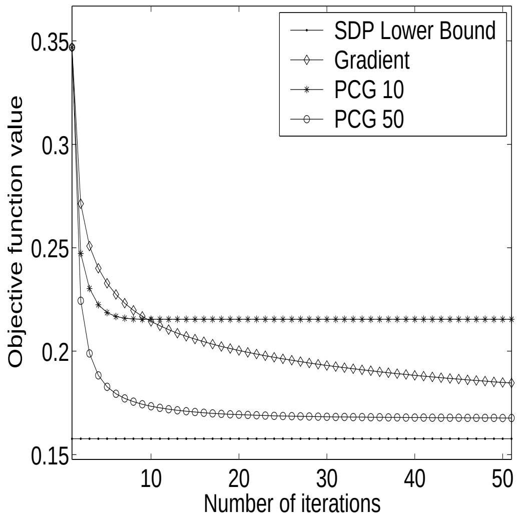


Figure 3.9: Convergence of local refinement methods.

when the PCG step is truncated at just 10 steps, the final estimation accuracy is inferior compared to gradient descent as well. Typically, the PCG step takes only about 30 steps to converge, so keeping the truncation limit at 50 means that there is no real truncation in this case. Tailoring these methods further for application in the nonlinear least squares problem posed by the noisy distance geometry problem, such as using a good Hessian approximation or choosing an effective preconditioner or deciding the number of iterations at which to terminate the PCG method, are promising directions for future research.

3.7 Distributed localization

Unfortunately, the existing SDP solvers have very poor scalability. They can only handle SDP problems with the number of constraints up to a few thousands. The difficulty is that each iteration of interior-point algorithm SDP solvers needs to factorize and solve a dense matrix linear system whose dimension is the number of constraints. While we could solve localization problems with 50 sensors in few seconds, we have tried to use several off-the-shelf codes to solve localization problems with 400 sensors and often these codes quit either due to memory shortage or having reached the maximum computation time.

We describe an iterative distributed SDP computation scheme to overcome this difficulty. We first partition the anchors into many clusters according to their physical positions, and assign some sensors into these clusters if a sensor has a direct connection to one of the anchors. We then solve SDPs *independently* at each cluster, and fix those sensors' positions that have high accuracy measures according the SDP computation. These positioned sensors become 'ghost anchors' and are used to decide the remaining un-positioned sensors. The distributed scheme then repeats. The distributed scheme is highly scalable and we have solved randomly generated sensor networks of thousands of sensors in few minutes for a sequential implementation (*i.e.*, the cluster SDP problems are solved sequentially on a single processor), while the solution quality remains as good as that of using the centralized method for solving small networks.

3.7.1 Iterative algorithm

A round of the distributed computation method is straightforward and intuitive:

1. Partition the anchors into a number of clusters according to their geographical positions. In our implementation, we partition the entire sensor area into a number of

equal-sized squares and the anchors in one square form a regional cluster.

2. Each (unpositioned) sensor sees if it has a direct connection to an anchor (within the communication range to an anchor). If it does, it becomes an unknown sensor point in the cluster to which the anchor belongs. Note that a sensor may be assigned into multiple clusters and some sensors are not assigned into any cluster.

3. For each cluster of anchors and unknown sensors, formulate the error minimization problem for that cluster, and solve the resulting SDP model if the number of anchors is more than 2. Typically, if each cluster has less than 100 sensors and the model can be solved efficiently. The number and sizes of the clusters should therefore be chosen accordingly.

4. After solving each SDP model, check the individual trace (2.7) for each unknown sensor in the model. If it is below a predetermined small tolerance, label the sensor as *positioned* and its estimation $\bar{x}_j$ becomes an 'anchor'. If a sensor is assigned in multiple clusters, we choose the $\bar{x}_j$ that has the smallest individual trace. This is done so as to choose the best estimation of the particular sensor from the estimations provided by solving the different clusters.

5. Consider positioned sensors as anchors and return to Step 1 to start the next round of estimation.

Note that the solution of the SDP problem in each cluster can be carried out at the cluster level so that the computation is highly distributive. The only information that needs to be passed among the neighboring clusters is which of the unknown sensors become positioned after a round of SDP solutions.

It is expected that the noisy cases will take more iterations since the number of 'ghost' anchors added at each iteration will be lower due to higher errors in estimation. The final rounds mainly refine position estimations for a small number of sensors and each round runs in a few seconds. We can choose to truncate the number of SDP iterations early, and initiate the local refinement method. Typically, a few iterations of the SDP yield good enough starting points for the local refinement method to converge quickly to an accurate solution.

In solving the SDP model for each cluster, even if the number of sensors is below 100, the total number of constraints could be in the range of thousands. However, many of those 'bounding away' constraints, *i.e.*, the constraints between two remote points, are inactive or redundant at the optimal solution. Therefore, we adapt an iterative active constraint generation method. First, we solve the problem including only partial equality constraints and completely ignoring the bounding-away inequality constraints to obtain a solution. Secondly we verify the equality and inequality constraints and add those violated at the current solution into the model, and then resolve it with a 'warm-start' solution. We can repeat this process until all of the constraints are satisfied. Typically, only about $O(n+m)$ constraints are active at the final solution so that the total number of constraints in the model can be controlled at $O(n+m)$.

Two SDP codes are used in our method, the primal-dual homogeneous and self-dual interior-point algorithm SeDuMi of [82] and the dual interior-point algorithm DSDP of [7]. Typically, DSDP is 3 to 4 times faster than SeDuMi, but SeDuMi is often more accurate and robust. DSDP is faster due to the fact that the sparse data structure of the problem is more suitable for DSDP.

3.7.2 Examples

Example 3.7. *The first simulation is carried out for solving a network localization with* $2,000$ *sensors, where the iterative distributed SDP method terminates in three rounds, see Figure 3.10 . When a sensor is not positioned, its estimation is typically at the origin. In this simulation, the entire sensor region is partitioned into* 7×7 *equal-sized squares,* i.e., 49 *clusters, and the radio range is set at* 0.06.

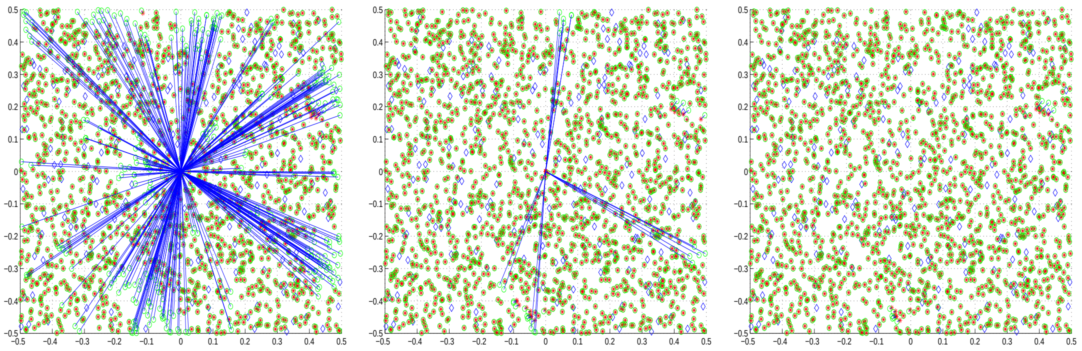
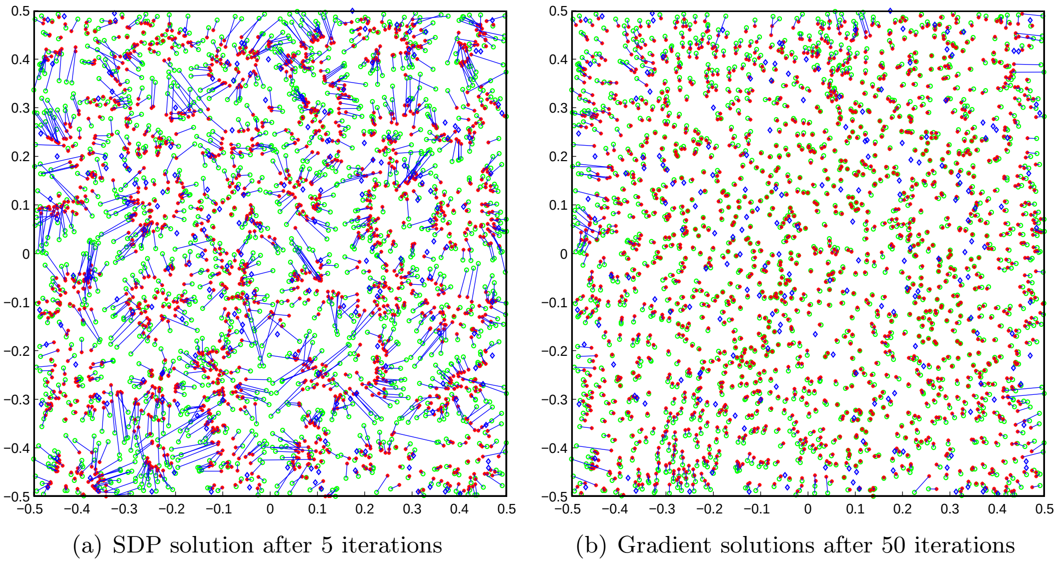

(a) First iteration (b) Second iteration (c) Final iteration

Figure 3.10: Position estimations for the 2,000 node sensor network, 200 anchors, no noise, Radio range=0.06, and the number of clusters=49.

As can be seen from the Figure 3.10 , it is usually the outlying sensors at the boundary or the sensors which do not have many anchors within the radio range that are not estimated in the initial stages of the method. The (blue) lines indicate that these kind of points are not well estimated in the initial stages, and so the discrepancy in actual and estimated positions is high. Gradually, as the number of well estimated sensors or 'ghost' anchors grows, more and more of these points are estimated. It is interesting to note that the erroneous points are concentrated within particular regions. This clearly indicates that the clustering approach prevents the propagation of errors to other clusters. This is because the estimated points within a cluster are used to estimate other points only if their estimation error is below a threshold.

Example 3.8. *Now consider another localization problem with 1800 sensors, 200 anchors, radio range 0.05 but also introduce a 10% multiplicative Normal noise. The sensor network is decomposed into 36 equal-sized domains. Figure 3.11(a) shows the SDP solution after 3 iterations. One can see that the SDP algorithm fails to find accurate solution in certain small areas. But after 50 gradient search steps, the final localization, shown in Figure 3.11(b), is improved.*



(a) SDP solution after 5 iterations (b) Gradient solutions after 50 iterations

Figure 3.11: Position estimations for the $2,000$ node sensor network, 200 anchors, 10% multiplicative noise, Radio range=.05, and the number of clusters=36.

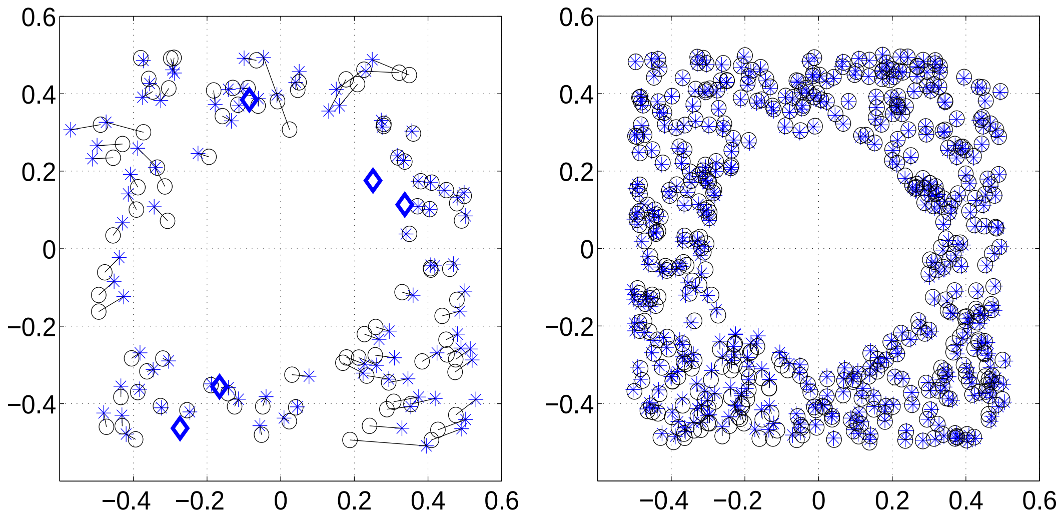
Our program is implemented with MATLAB *and it uses SEDUMI [82] or DSDP [7] as* *the SDP solver. It costs 165 seconds of SEDUMI or 63 seconds of DSDP to get the SDP localization in Figure 3.11(a) on a Pentium IV 1.2 GHz and 512 MB RAM computer. Then, after 50 gradient search steps (which cost less than 20 seconds), the objective function is reduced from 12.81 to 0.230. It can be seen from Figure 3.11(b) that most of the sensors are located very close to their true positions, although few of them, most of which are close to the boundary of the network, are solved inaccurately.*

3.7.3 Extensions

The distributed SDP approach solves with great accuracy and speed very large estimation problems which would otherwise be extremely time consuming in a centralized approach. Also due to smaller independent clusters, the noise or error propagation is quite limited as opposed to centralized algorithms. This idea of breaking down a large problem into smaller subproblems is also useful in other scenarios, and also has further scope for improvement.

- **Geodesic Distances**

  The distributed method for sensor networks would also find application in networks where long range distances diverge from actual Euclidean distances, but the smaller range distances are much more reliable, such as geodesic distances. This can often be the case in the sensor network scenario, where long range distances are often just measured through hop information. The divergence can be quite marked especially when the network structure is irregular and there are 'holes' in the network. Up to a certain limit, determined by the size and placement of the holes in the network, the given geodesic distances may be reliable, in the sense that they correspond closely with the Euclidean distances. By breaking up the network into clusters with sizes up to that limit, and then using the SDP approach in each cluster, we can ensure accurate localization even using dense geodesic distance information instead of Euclidean distance information. Figure 3.12 provides examples of cases where such geodesic information is available, and the geometry of the network is such that such geodesic information does not correlate well with the Euclidean distance information beyond a certain point. For both the 100 point and 1000 point networks, only the noisy distances up to a short range are used. The distance measurements for the short range are only corrupted by 10% multiplicative noise. In these examples, the presence of a hole in the network can severely distort the long range information when hop information is used to compute distances between nodes 2 or more hops away, and they are therefore ignored. By only regarding the distance information up to the number of hops where it corresponds closely with the Euclidean distance information, we are able to estimate positions of points quite accurately.



(a) 100 point network (b) 1000 point network

Figure 3.12: Irregular Networks with Geodesic Distances.

- **Intelligent clustering and error measures**

  The current clustering approach also assumes that the anchor nodes are more or less uniformly distributed over the entire space. So by dividing the entire space into smaller sized square clusters, the number of anchors in each cluster is also more or less the same. However this may or may not be the case in a real scenario. A better approach would be to create clusters more intelligently based on local connectivity information. Keeping this in mind, we try and find for each sensor its immediate neighborhood, *i.e.*, points within radio range of it. It can be said that such points are within one hop of each other. Higher degrees of connectivity between different points can also be evaluated by calculating the minimum number of hops between the 2 points. Using the hop information, we propose to construct clusters which are not necessarily of any particular geometric configuration but are defined by its connectivity with neighborhood points. Such clusters would yield much more efficient SDP models and faster and more accurate estimations. Furthermore, the trace error measure (2.7) is not completely reliable as the distance measures start becoming more and more noisy. Alternate error measures also need to be developed which can do a better job of detecting badly estimated points. Some ideas to address these are explored in the context of molecule structure determination in the next chapter. Building on the ideas in [13], the authors in [21, 44] proposed an adaptive rule based clustering strategy to sequentially divide a global anchored graph localization problem in $\mathcal{R}^2$, into a sequence of subproblems. The technique in localizing each subproblem is similar to that in [13], but the clustering and stitching strategies are different. It is reported that the improved techniques can localize anchored graphs very efficiently and accurately.

3.8 Experimental results

We repeat the simulation setup used for our experiments. A network of uniform randomly distributed unknown points is generated in the square area $[-0.5, 0.5] \times [-0.5, 0.5]$. Anchors are also generated in the same manner. The number of anchors will be denoted by m. The distances between the nodes is calculated. If the distance between 2 nodes was less than the specified radio range R, the distance is included in the edge set for solving the SDP after adding a random error to it in the following manner:

$$d_{ij} = \hat{d}_{ij}|1 + nfN(0,1)|,$$

where $\hat{d}_{ij}$ is the actual distance between the 2 nodes, nf (noise factor) is a given number between $[0, 1]$ that is used to control the amount of noise variance and $N(0, 1)$ is a standard normal random variable. We will also refer to he noise in terms of percentages. For example, 10% noise corresponds to $nf = 0.1$. The radio range R and noise factor nf are varied and the effects on the RMSD error as defined in (3.14) are observed. For each of the graphs shown below, experiments are performed on 30 independent configurations and the average is taken. Our program is implemented with Matlab and it uses either DSDP [7], SEDUMI [82] or SDPT3 [84, 90] as the SDP solver.

We will consider the performance of the SDP relaxation, by performing 25 independent trials on random networks of 60 points, and varying the radio range, measurement noise and number of anchors. The improvement offered by regularization and local refinement becomes clear from these results. We will also explore the performance of the distributed algorithm in terms of estimation error and computation time. We will conclude this section with a comparison against an existing method.

3.8.1 Effect of varying radio range and noise factor

Figure 3.13 shows the variation of average estimation error (normalized by the radio range R) when the noise in the distance measurements increases and with different radio ranges. 6 anchors are placed randomly in the network area.

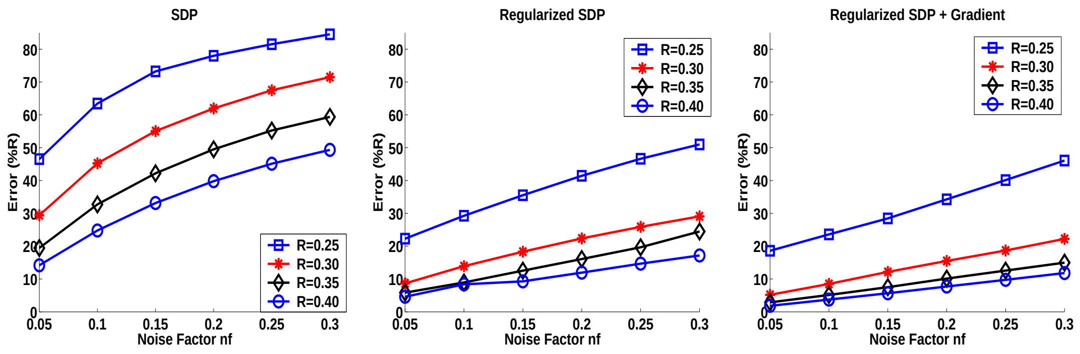


Figure 3.13: Variation of estimation error (RMSD) with measurement noise, 6 anchors randomly placed.

The results for SDP (3.3) which is the model without regularization will serve as comparisons against which the effectiveness of the regularization technique can be gauged. As can be seen from the results, for random anchor placement, the use of regularization provides about 30% R improvement even for very high noise and low radio ranges. This is a significant improvement over the previous methods as far as estimation accuracy for random anchor placement is concerned. For reasonably large radio ranges, the estimation error stays under control even in highly noisy scenarios. Even for 30% noise, the estimation error can be kept below 20% R for $R \geq 0.3$. It can also be observed that the gradient method offers some minor improvement over the regularized SDP. In fact, it is because the regularized SDP provides an excellent starting point that the gradient method error is also lowered significantly as compared to results for the unregularized SDP (3.3).

3.8.2 Effect of number of anchors

Figure 3.14 shows the variation of estimation error by increasing the radio range while varying the number of randomly placed anchors from 4 to 8, and the nf was fixed at 0.2. For the same networks, Figure 3.15 shows the variation of estimation error by increasing the measurement noise (nf) while varying the number of anchors from 4 to 8, where the radio range is fixed at 0.3. We remind the reader that the number of anchors is denoted by m.

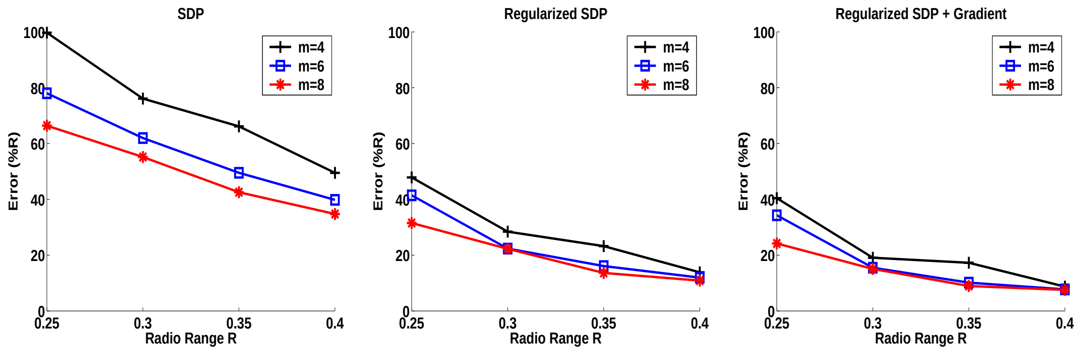


Figure 3.14: Variation of estimation error (RMSD) with number of anchors (randomly placed) and varying radio range, 20% multiplicative noise.

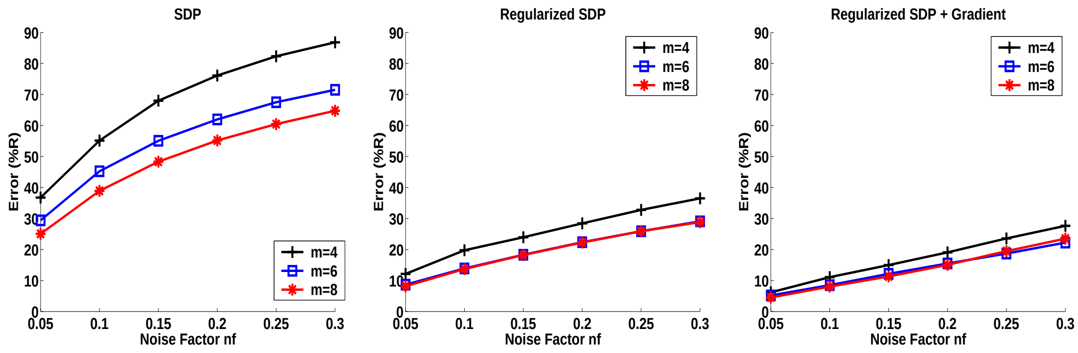


Figure 3.15: Variation of estimation error (RMSD) with number of anchors (randomly placed) and varying measurement noise, $R = 0.3$.

From these results, it becomes clear that beyond a certain number of anchors, the regularized SDP (3.15) provides nearly the same accuracy irrespective of the number of anchors as opposed to the SDP model (3.3) which is more sensitive to the number of anchors. This is encouraging since the regularized SDP can therefore be used to provide better estimation accuracy with fewer anchors.

3.8.3 Computational effort

The computational results presented here were generated using the interior-point algorithm SDP solvers DSDP [7] and SeDuMi [82] with their interfaces to Matlab on a Pentium IV 1.2 GHz and 512 MB RAM PC. DSDP is faster due to the fact that the data structure of the problem is more suitable for DSDP. However SeDuMi is often more accurate and robust.

The number of constraints and solution time was analyzed. Figure 3.16 shows how the number of constraints grows with the number of points for the formulation (3.1) with regularization. Figure 3.17 illustrates how the solution time increases as the number of points in the network increases. Note that the gradient solution time just takes into account the time taken for the local gradient optimization. The SDP solution needs to be computed before the gradient method can be applied. Therefore, the total time would be the combined time for the SDP and gradient methods. However, in our graphs, we do not show the combined times. Instead, times for each of the individual steps are shown in order to also compare the relative times taken by each of the steps. It should be noted that a smaller radio range will be required for denser networks. Since the networks get denser with more points, the connectivity increases as well. However, beyond a certain value, increasing the connectivity only adds more redundant constraints to the problem and increases the computational effort required. So we also progressively decrease the radio range with the number of points so as to keep connectivity within a certain range. The radio range for 40 points is set at 0.4. For each increase of 20 points, the radio range is decreased by 0.05. From the results, it can be observed that the number of constraints and solution time is highly dependent not only on the number of points but upon the radio range selected as well. For example, the connectivity for 100 points and a radio range 0.25 is higher than that for 120 points and radio range 0.2, so the number of constraints and the solution time are also higher. But overall, it can be said that the number of constraints does not scale with $O(n^2)$ but more typically with $O(n)$. Therefore as discussed in Section 2.3.3, the computational effort is more typically $O(n^3)$ as opposed to the worst case $O(n^6)$.

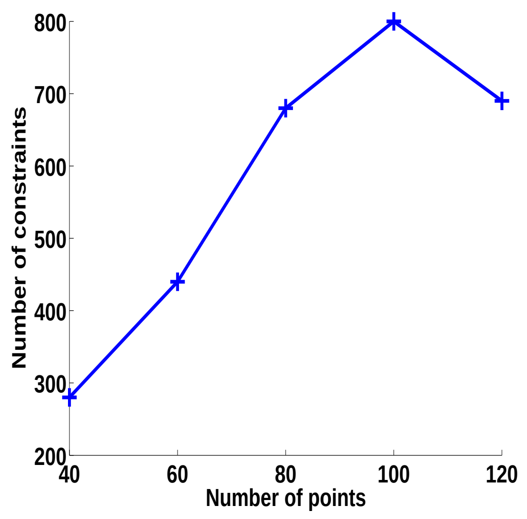


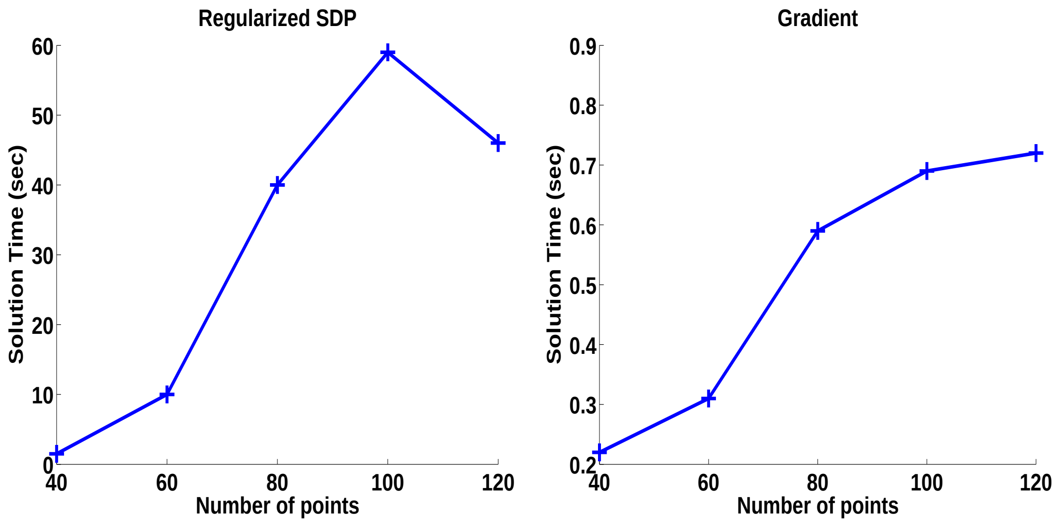
Figure 3.16: Number of constraints vs. number of points.

3.8.4 Distributed method

The solution time and error for the distributed method for larger networks is presented in Figure 3.18(a) and 3.18(b) respectively. As can be seen from the results, a combination of the SDP and gradient method is particularly useful in these cases in reducing the error and is also very computationally efficient. Even for a 4000 point network, the combined solution time is about 400 seconds for a sequential implementation. The estimation error is also extremely low because the networks have a high connectivity and this approach exploits this fact while keeping the solution time low. In fact, the 1000 point network which has a radio range of 0.1 has the highest connectivity, so the error is the least. For the 2000 point network with radio range 0.05, the connectivity is lower, so the relative average estimation error is higher. But a higher connectivity is also reflected in the increase in solution time from going to the 500 to 1000 points. This case has a sharper rate of increase than the other transitions.

3.8.5 Comparison with existing methods

Partial comparison with existing methods is also presented. For this purpose, Multidimensional Scaling, another centralized approach is used. The comparisons are made against the simulation results reported in [76] with networks of 200 points uniformly distributed

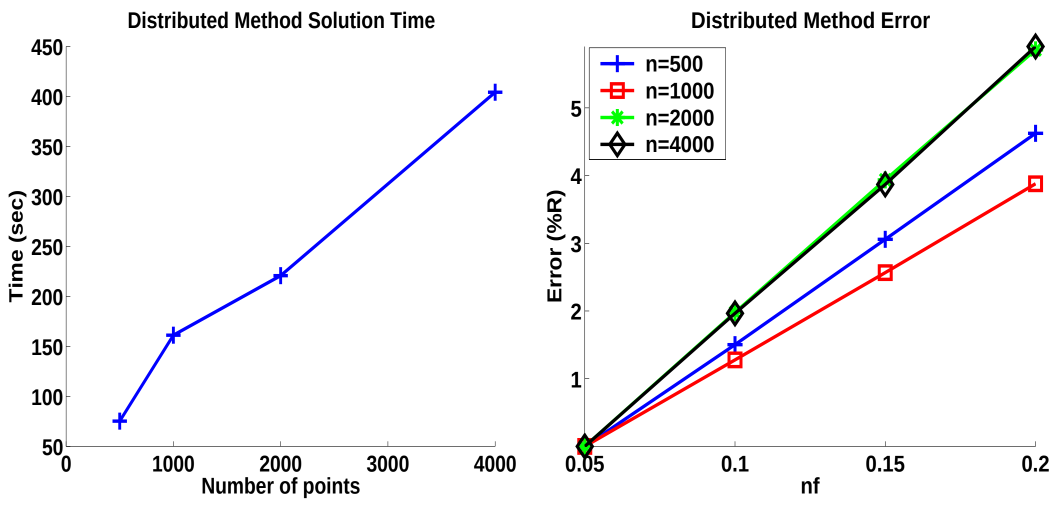


(a) (b)

Figure 3.17: Solution time vs. number of points.

in a square area of $10r \times 10r$ and the radio range is varied from $1.25r$ to $2.5r$ with only 5% measurement noise. For detailed results obtained by MDS, refer to the experimental results section in [76] for random uniform networks. While their results are also presented for localization using only simple connectivity data, our comparisons will only be against the results presented for localization using distance data as well. The results from the paper show that MDS outperforms other methods such as [61] and so is a good choice for comparison.

Figure 3.19 shows the results for the above mentioned setup as reported in [76] for 4 different MDS based methods(MDS-MAP(C), MDS-MAP(C,R), MDS-MAP(P), MDS-MAP(P,R)) which differ in whether a central method is used or alternatively broken into smaller patches and also whether a post processing refinement step is applied or not. The radio range and number of anchors is varied. Figure 3.20 shows the results obtained by SDP. From the results, it appears that the high radio range performance is better than the performance of the MDS methods (between $0-5\%$ R). Only in this lower radio range regime, the MDS estimation error reportedly drops to about $10-20\%$ R after a local refinement is applied, as opposed to $15-25\%$ R for our SDP based methods. The refined MDS methods appear to have slightly better performance for uniform networks with low radio range and low noise factor. Overall, however, the performances of the competing techniques are quite



(a) (b)

Figure 3.18: Distributed method simulations.

similar for this particular problem.

3.9 Conclusion

In this chapter, robust techniques to solve the noisy graph realization problem were discussed and their performance demonstrated on the sensor network localization problem. The results show significant improvement in the performance of the SDP based algorithms in highly noisy scenarios by adding regularization terms followed by a local refinement method. Future work should concentrate on making the algorithms more efficient and robust. The links with tensegrity theory need to examined more closely in order to develop more effective regularization terms through intelligent edge selection. By choosing optimal weights within the objective function with regard to both the error term and the regularization term, the accuracy of the estimates provided by a single SDP solution can potentially be improved significantly.

As the size of the networks grows, solving one large SDP may become intractable. There emerges a need to extend the ideas developed so far to more distributed approaches. Such an approach was discussed in the context of sensor network localization. However when there are no anchors, and the distance measures have a lot of uncertainty, this clustering

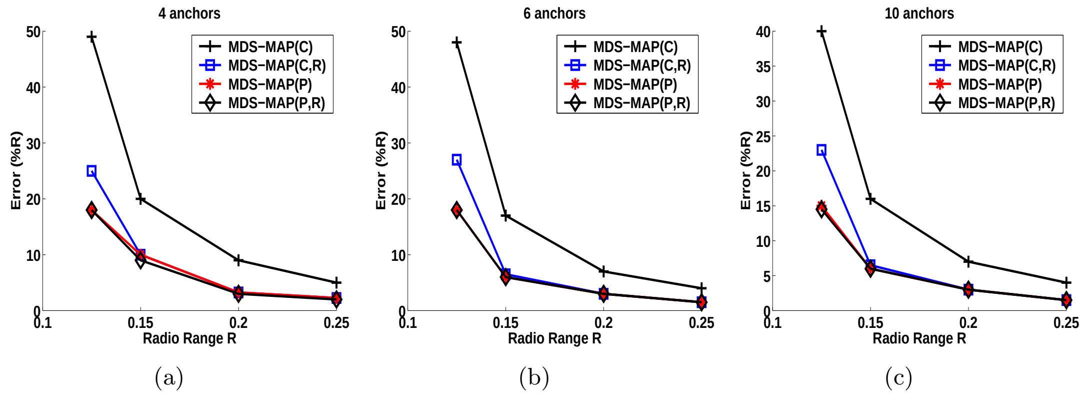


Figure 3.19: MDS results, Estimation error for 200 point networks with varying radio range, 5% multiplicative noise.

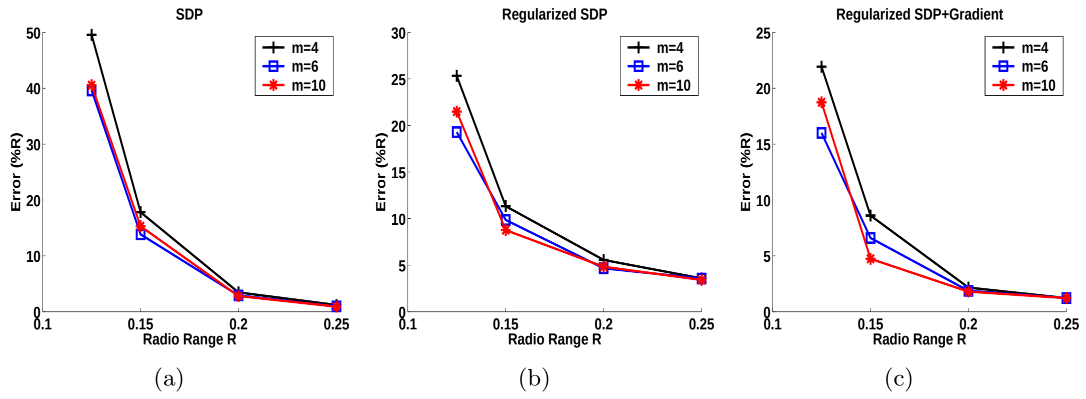


Figure 3.20: Comparison with MDS, Estimation error for 200 point networks with varying radio range, 5% multiplicative noise.

and stitching approaches discussed before are no longer applicable. More advanced algorithms are required. These are the subject of the following chapter, where distributed graph realization is explored in the context of molecule structure determination.

Chapter 4

Anchor free distributed graph realization: molecule structure prediction

4.1 Introduction

Another instance of the graph realization problem arises in molecular conformation, specifically, protein structure determination. It is well known that protein structure determination is of great importance for studying the functions and properties of proteins. In order to determine the structure of protein molecules, nuclear magnetic resonance (NMR) experiments are performed to estimate lower and upper bounds on interatomic distances [27, 37]. Additional knowledge about the bond angles and lengths between atoms also yield information about relative positions of atoms. In the simplest form, given a subset of all the pairwise distances between the atoms of a molecule, the objective is to find a conformation of all the atoms in the molecule such that the distance constraints are satisfied.

Since this problem is also an abstraction of graph realization, most of the concepts that were developed for the sensor network localization problem can be used directly for the molecular conformation problem. In fact, the terms localization and realization will be used interchangeably throughout this chapter. However, some crucial improvements to the basic algorithm have to be made before it can be applied to the molecular conformation problem. In Chapter 3, the problem was described in 2-D although it can be extended to higher dimensions, as is illustrated in this chapter. The improvements suggested in Sections 3.5 and 3.6 also provide much better performance for noisy distance data. However, these SDP methods can only handle problems of relatively small sizes with the number of points typically below 200 or so. A distributed version of the algorithm was described in Section 3.7 for larger sets of points, in which the larger problem was broken down in smaller subproblems corresponding to local clusters of points. The assumption made in the large scale sensor network problem was the existence of anchor nodes all across the network, *i.e.*, points whose positions are known prior to the computation. These anchor nodes play a major role in determining the positions of unknown nodes in the distributed algorithm, since the presence of anchors in each cluster is crucial in facilitating the clustering of points and the process of stitching together different clusters. The anchor nodes are used to create clusters by including all sensors within one or two hops of their radio range. When the positions for unknown nodes are computed, they are already in the global coordinate system since the anchor positions have been incorporated in the computation. Furthermore, the presence of anchors help to dampen the propagation of errors in the estimated positions to other clusters when the problem is solved by a distributed algorithm.

Without anchors, we need alternative methods to cluster the points and to stitch the clusters together. In this chapter, we propose a distributed algorithm for solving large scale noisy anchor-free Euclidean metric realization problems arising from 3-D graphs, to address precisely the issues just mentioned. In the problem considered here, there are no a priori determined anchors, as is the case in the molecular conformation problem. Therefore the strategy of creating local clusters for distributed computation is different. We perform repeated matrix permutations on the sparse distance matrix to form local clusters within the structure. The clusters are built keeping in mind the need to maintain a reasonably high degree of overlap between adjacent clusters, *i.e.*, there should be enough common points considered between adjacent clusters. This is used to our advantage when we combine the results from different clusters during the stitching process.

Within each cluster, the positions of the points are first estimated by solving an SDP relaxation of a non-convex minimization problem seeking to minimize the sum of errors between given and estimated distances. Again, a gradient descent based local refinement method, similar to the one described in Section 3.6, is used as a postprocessing step, after the SDP computation to further reduce the estimation errors. After the gradient descent postprocessing step, poorly estimated points within each cluster are isolated and they are recomputed when more points are correctly estimated.

The solution of each individual cluster yields different orientations in their local coordinate systems since there are no anchors to provide global coordinate information. The local configuration may be rotated, reflected, or translated while still respecting the distance constraints. This was not a problem in the case when anchors were available, as they would perform the task of ensuring that each cluster follows the same global coordinate system. Instead, we now use a least squares based affine mapping between local coordinates of common points in overlapping clusters to create a coherent conformation of all points in a global coordinate system.

We test our algorithm on protein molecules of varying sizes and configurations. The protein molecules, all with known molecular configurations, are taken from the Protein Data Bank [8]. Our algorithm is able to reliably reconstruct the configurations of large molecules with thousands of atoms quite efficiently and accurately based on given upper and lower bounds on limited pairwise distances between atoms. To the best of our knowledge, there are no computational results reported in existing literature for determining molecular structures of this scale by using only sparse and noisy distance information. However, there is still room for improvement in our algorithm in the case of very sparse or highly noisy distance data.

For simplicity, our current SDP based distributed algorithm do not incorporate the lower constraints generated from van der Waals (VDW) interactions between atoms. But such constraints can naturally be incorporated into the SDP model. Given that our current algorithm performs quite satisfactorily without the VDW lower bound constraints, we are optimistic that with the addition of such constraints and other improvements in the future, our algorithm would perform well even for the very difficult case of highly noisy and very sparse distance data.

Section 4.2 describes related work in distance geometry and molecular conformation, and attempts to situate our work in that context. Section 4.3 extends the SDP relaxation models for molecule conformation. In particular, we introduce the inequality based distance geometry model. A preliminary theory for anchor-free graph realization is also developed. The intelligent clustering and cluster stitching algorithms are introduced in Section 4.6 and 4.7 respectively. Section 4.8 describes the complete distributed algorithm. Section 4.9 discusses the performance of the algorithm on protein molecules from the Protein Data Bank [8]. Finally in Section 4.10, we conclude with a summary of the chapter and outline some work in progress to improve our distributed algorithm.

4.2 Related work

Many approaches have been developed for the molecular distance geometry problem. An overall discussion of the methods and related software is provided in [102]. Some of the approaches are briefly described below.

As mentioned before in Section 1.2, when the exact distances between all points are given, a valid configuration can be obtained by computing the eigenvalue decomposition of the inner product matrix (which can obtained through a linear transformation of the distance matrix). A valid inner product matrix in d dimensions must have rank d, and the eigenvectors corresponding to the d nonzero eigenvalues give a valid configuration. So a decomposition can be found and a configuration constructed in $O(n^3)$ arithmetic operations, where n is the number of points. The EMBED algorithm, developed by Crippen and Havel [27], exploits this idea for sparse and noisy distances by first performing bound smoothing, *i.e.*, preprocessing the available data to remove geometric inconsistencies and finding valid estimates for unknown distances. Then a valid configuration is obtained through the eigenvalue decomposition of the inner product matrix and the estimated positions are then used as the starting iterate for local optimization methods on certain nonlinear least squares problems.

Classical multidimensional scaling (MDS) is the general class of methods that takes inexact distances as input, and extracts a valid configuration from them based on minimizing the discrepancy between the inexact measured distances and the distances corresponding to the estimated configuration. The inexact distance matrix is referred to as a dissimilarity matrix in this framework. Since the distance data is also incomplete, the problem also involves completing the partial distance matrix. The papers by Trosset [86, 87, 88] consider this problem of completing a partial distance matrix, as well as the more general problem of finding a distance matrix of prescribed embedding dimension that satisfies specified lower and upper bounds, for use in MDS based algorithms.

Also worth noting in this regard is the distance geometry program APA described by Glunt et al. [68], that applies the idea of a 'data box', a rectangular parallelepiped of dissimilarity matrices that satisfy some given upper and lower bounds on distances. An alternating projection based optimization technique is then used to solve for both a dissimilarity matrix that lies within the data box, and a valid embedding of the points, such that the discrepancy between the dissimilarity matrix and the distances from the embedding are minimized.

The ABBIE software package, developed by Hendrickson [39], on the other hand, exploits the concepts of graph rigidity to solve for smaller subgraphs of the entire graph defined by the points and distances and finally combining the subgraphs to find an overall configuration. It is especially advantageous to solve for smaller parts of the molecule and to provide certificates confirming that the distance information is not enough to ascertain certain atoms. Our approach tries to retain these advantages by solving the molecule in a distributed fashion, *i.e.*, solving smaller clusters and later assembling them together.

Some global optimization methods attempt to attack the problem of finding a conformation which fits the given data as a large nonlinear least squares problem. For example, a global smoothing and continuation approach is used in the DGSOL algorithm by More and Wu [59]. To prevent the algorithm from getting stuck at one of the large number of possible local minimizers, the nonlinear least squares problem (with an objective that is closely related to the refinement stage of the EMBED algorithm) is mollified to smoother functions so as to increase the chance of locating the global minimizer. However, it can still be difficult to find the global minimizer from various random starting points, especially with noisy distance information. More refined methods that try to circumvent such difficulties have also been developed in [58], though with limited success. takes special care not to violate the physically inviolable Minimum Separation (MSD), or Van Der Waals (VDW) constraints, which In fact become more and more crucial in reducing the search space in case of very sparse distance data.

Another example is the GNOMAD algorithm, developed by Williams et al [95], also a global optimization method, which takes special care to satisfy the physically inviolable minimum separation distance, or van der Waals (VDW) constraints. For GNOMAD, the VDW constraints are crucial in reducing the search space in the case of very sparse distance data. Obviously, the VDW constraints can easily be incorporated into any molecular conformation problem that is modelled by an optimization problem, and are also used in the other approaches.

Besides optimization based methods, there are geometry based methods proposed for the molecular conformation problem. The effectiveness of simple geometric build up (also known as triangulation) algorithms has been demonstrated in work by Zhijun Wu and others [30] and [96], for molecules when exact distances within a certain cut-off radius are all given. Basically, this approach involves using the distances between an unknown atom and previously determined neighboring atoms to find the coordinates of the unknown atom. The algorithm progressively updates the number of known points and uses them to compute points that have not yet been determined. However, the efficacy of such methods for large molecules with very sparse and noisy data has not yet been demonstrated.

In this chapter, we will attempt to find the structures of molecules with sizes varying from hundreds to several thousands of atoms, given only upper and lower bounds on some limited pairwise distances between atoms. The approach described here also performs distance matrix completion, similar to some of the methods described above. The extraction of the point configuration after matrix completion is still the same as the MDS methods. The critical difference lies in the use of an SDP relaxation for completing the distance matrix. Furthermore, our distributed approach avoids the issue of intractability for very large molecules by splitting the molecule into smaller subgraphs, much like the ABBIE algorithm [39] and then stitching together the different clusters. Some of the atoms which are incorrectly estimated are solved separately using the correctly estimated atoms as anchors. The latter bears some similarities to the geometric build up algorithms. In this way, we adapted and improved some of the techniques used in previous approaches, but also introduced new ideas generated from recent advances in SDP to attack the twin problems of dealing with noisy and sparse distance data, and the computational intractability of large scale molecular conformation.

4.3 The inequality based SDP model

Consider a molecule with n atoms with unknown coordinates $x_i \in \mathcal{R}^d$, $i = 1, \ldots, n$ in a local coordinate system. Suppose that we know the upper and lower bounds on the Euclidean distances between some pairs of unknown points specified in the edge set $\mathcal{N}_u$. For the rest of the point pairs, the upper and lower bounds would be the trivial bounds, ∞ and 0. We define the lower bound distance matrices $\underline{D} = (\underline{d}_{ij})$ where $\underline{d}_{ij}$ is specified if $(i, j) \in \mathcal{N}_u$, and $\underline{d}_{ij} = 0$ otherwise. The upper bound distance matrix $\overline{D} = (\overline{d}_{ij})$ is defined similarly. We let $D = (d_{ij})$ be the mean of $\underline{D}$ and $\overline{D}$, *i.e.*, $d_{ij} = (\underline{d}_{ij} + \overline{d}_{ij})/2$. The realization problem for the graph $(\{1, \ldots, n\}, \mathcal{N}_u)$ is to determine the coordinates of the unknown points $x_1, \ldots, x_n$

given the upper and lower bound distance matrices, $\underline{D}$ and $\overline{D}$.

Bear in mind that since there are no anchors, these point positions are only relative, since the entire coordinate system can be translated, rotated and reflected while still maintaining the distance constraints. However the positions of the points relative to each other will remain fixed and our objective is to find that relative structure. Without loss of generality, we will assume the points to be centered around the origin. The realization problem just mentioned can be formulated as the following feasibility problem:

$$\begin{array}{rl} \text{Find} & x_1, \dots, x_n \\ \text{subject to} & \underline{d}_{ij}^2 \leq \|x_i - x_j\|^2 \leq \overline{d}_{ij}^2 \quad \forall (i,j) \in \mathcal{N}_u. \end{array} \tag{4.1}$$

Let $X = [x_1\ x_2\ \dots\ x_n]$ be the $d \times n$ matrix that needs to be determined. Problem (4.1) can be rewritten in matrix form as as:

$$\begin{array}{rl} \text{Find} & Y \\ \text{subject to} & \underline{d}_{ij}^2 \leq e_{ij}^T Y e_{ij} \leq \overline{d}_{ij}^2, \quad \forall (i,j) \in \mathcal{N}_u \\ & Y = X^T X. \end{array} \tag{4.2}$$

Problem (4.2) is unfortunately non-convex. Note that this is just a version of the SDP models considered in previous chapter such as (2.5) and (3.3) except that the constraints are in the form of inequalities and that there are no anchors. So $\mathcal{N}_a = \emptyset$, and the $(1,2)$ and $(2,1)$ block of Z in (2.5), (3.3) are always equal to zero if the starting iterate for the interior-point method used to solve it is chosen to be so. Also note that we must add the extra constraint $Y\mathbf{e} = \mathbf{0}$ to eliminate the translational invariance of the configuration by putting the center of gravity of the points at the origin, *i.e.*, $\sum_{i=1}^n x_i = \mathbf{0}$. Thus, the SDP relaxation of (4.2) can be written as the following standard SDP problem:

$$\begin{array}{rl} \text{Find} & Y \\ \text{subject to} & \underline{d}_{ij}^2 \leq e_{ij}^T Y e_{ij} \leq \overline{d}_{ij}^2, \quad \forall\ (i,j) \in \mathcal{N}_u, \\ & Y\mathbf{e} = \mathbf{0}, \\ & Y \succeq \mathbf{0}. \end{array} \tag{4.3}$$

If there are additional constraints of the form $\|x_i - x_j\| \geq L$ coming from knowledge about the minimum separation distance between any 2 points, such constraints can be included in the SDP (4.3) by adding inequality constraints of the form: $e_{ij}^T Y e_{ij} \geq L^2$. In molecular conformation, the minimum separation distances corresponding to the VDW interactions are used in an essential way to reduce the search space in the atom-based constrained optimization algorithm (GNOMAD) described in [95]. The minimum separation distance constraints are also easily incorporated in the MDS framework [68, 87].

In the anchor-free case, if the graph is localizable (defined in the next subsection), a realization $X \in \mathcal{R}^{d\times n}$ can no longer be obtained from the $(2,1)$ block of Z but needs to be computed from the inner product matrix Y by factorizing it to the form $Y = X^T X$ via eigenvalue decomposition (as has been done in previous methods discussed in the literature review). In the noisy case, the inner product matrix Y would typically have rank greater than d. In practice, X is chosen to be the best rank-d approximation, by choosing the eigenvectors corresponding to the d largest eigenvalues. The configuration so obtained is a rotated or reflected version of the actual point configuration.

4.4 Theory of anchor-free graph realization

In order to establish the theoretical properties of the SDP relaxation, we will consider the cases where all the given distances in $\mathcal{N}_u$ are exact, *i.e.*, without noise. A graph $G = (\{1,\dots,n\}, D)$ is localizable in dimension d if (i) it has a realization X in $\mathcal{R}^{d\times n}$ such that $\|x_i - x_j\| = d_{ij}$ for all $(i,j) \in \mathcal{N}_u$; (ii) it cannot be realized (non-trivially) in a higher dimensional space. We let $D = (d_{ij})$ be the $n \times n$ matrix such that its (i,j) element d_{ij} is the given distance between points i and j when $(i,j) \in \mathcal{N}_u$, and zero otherwise. It is shown for the exact distances case in [79] that if the graph with anchors is localizable, then the SDP relaxation will produce a unique optimal Z such that $Y = X^T X$. For the anchor free case where $\mathcal{N}_a = \emptyset$, it is clear that the realization cannot be unique since the configuration may be translated, rotated, or reflected, and still preserve the same distances. To remove the translational invariance, we will add an objective function to minimize the norm of the solution in the problem formulation:

$$\begin{array}{ll} \text{Minimize} & \sum_{j=1}^n \|x_j\|^2 \\ \text{subject to} & \|x_i - x_j\|^2 = d_{ij}^2, \quad \forall\ (i,j) \in \mathcal{N}_u. \end{array} \tag{4.4}$$

What this minimization does is to translate the center of gravity of the points to the origin, *i.e.*, if $\bar{x}_j$, $j = 1, ..., n$, is the realization of the problem, then the realization generated from (4.4) will be $\bar{x}_j - \bar{x}$, $j = 1, ..., n$, where $\bar{x} = \frac{1}{n}\sum_{j=1}^{n}\bar{x}_j$, subject to only rotation and reflection. The norm minimization also helps the following SDP relaxation of (4.4) to have bounded solutions:

$$\begin{array}{ll} \text{Minimize} & \text{Trace}(Y) = I \bullet Y \\ \text{subject to} & e_{ij}^T Y e_{ij} = d_{ij}^2, \quad \forall\ (i,j) \in \mathcal{N}_u, \\ & Y \succeq \mathbf{0}, \end{array} \tag{4.5}$$

where $Y \in \mathcal{S}^n$ (the space of $n \times n$ symmetric matrices), I is the identity matrix, and $\bullet$ denotes the standard matrix inner product. The dual of the SDP relaxation is given by:

$$\begin{array}{ll} \text{maximize} & \sum_{(i,j)\in\mathcal{N}_u} w_{ij} d_{ij}^2 \\ \text{subject to} & I - \sum_{(i,j)\in\mathcal{N}_u} w_{ij} e_{ij} e_{ij}^T \succeq \mathbf{0}. \end{array} \tag{4.6}$$

Note that the dual is always feasible and has an interior, since $w_{ij} = 0$ for all $(i,j) \in \mathcal{N}_u$ is an interior feasible solution. Thus, from the duality theorem for SDP, we have:

Proposition 4.1. *Let* $\bar{Y} \succeq 0$ *be an optimal solution of* (4.5) *and* $\bar{U} = I - \sum_{(i,j)\in cN} \bar{w}_{ij} e_{ij} e_{ij}^T \succeq 0$ *be an optimal slack matrix of* (4.6). *Then,*

1. *complementarity condition holds:* $\bar{Y} \bullet \bar{U} = 0$ *or* $\bar{Y}\bar{U} = \mathbf{0}$*;*

2. *Rank*$(\bar{Y})$ + *Rank*$(\bar{U}) \leq n$.

In general, a primal (dual) max–rank solution is a solution that has the highest rank among all solutions for primal (4.5) (dual (4.6)). It is known that various path–following interior–point algorithms compute the max–rank solutions for both the primal and dual in polynomial time.

We now investigate when the SDP (4.5) will have an exact relaxation, given that the partial distance data (d_{ij}) is exact. For the anchored case, it was proved in [79] that the condition of the relaxation being exact is equivalent to that the rank of the SDP solution $\bar{Y}$ is d. However, for the anchor-free case, we are unable to prove this. Instead, we derive an alternative result:

Definition 4.1. *Problem* (4.4) *is d-localizable if there is no $x_j \in \mathcal{R}^h$, $j = 1, \ldots, n$, where $h \neq d$, such that:*

$$\|x_i - x_j\|^2 = d_{ij}^2 \quad \forall\ (i,j) \in \mathcal{N}_u.$$

For $h > d$, the condition should exclude the trivial case when we set $x_j = (\bar{x}_j; \mathbf{0})$ for $j = 1, \ldots, n$.

The d-localizability indicates that the distances cannot be embedded by a non–trivial realization in higher dimensional space, and cannot be 'flattened' to a lower dimensional space either. We now develop the following theorem.

Theorem 4.1. *If problem* (4.4) *is d-localizable, then the solution matrix, $\bar{Y}$, of* (4.5) *is unique and its rank equals d. Furthermore, if $\bar{Y} = \bar{X}^T\bar{X}$, then $\bar{X} = (\bar{x}_1, ..., \bar{x}_n) \in \mathcal{R}^{d\times n}$ is the unique minimum-norm localization of the graph with $\sum_{j=1}^n \bar{x}_j = 0$ (subject to only rotation and reflection).*

Proof. Note that problem (4.4) is d-localizable, by the definition that the only feasible solution matrix Y to (4.5) has rank d. Thus, there is a rank-d matrix $\bar{V} \in \mathcal{R}^{d\times n}$ such that any feasible solution matrix Y can be written as $Y = \bar{V}^T P \bar{V}$, where P is a $d \times d$ symmetric positive definite matrix. We show that the solution is unique by contradiction. Suppose that there are two feasible solutions

$$Y^1 = \bar{V}^T P^1 \bar{V} \quad \text{and} \quad Y^2 = \bar{V}^T P^2 \bar{V},$$

where $P^1 \neq P^2$ and, without loss of generality, $P^2 - P^1$ has at least one negative eigenvalue (otherwise, if $P^1 - P^2$ has at least one negative eigenvalue, we can interchange the role of P^1 and P^2; the only case left to be considered is when all the eigenvalues of $P^2 - P^1$ are equal to zero, but this case is not possible since it implies that $P^1 = P^2$). Let

$$Y(\alpha) = \bar{V}^T(P^1 + \alpha(P^2 - P^1))\bar{V}.$$

Clearly, $Y(\alpha)$ satisfies all the linear constraints of (4.5), and it has rank d for $0 \leq \alpha \leq 1$. But there is an $\alpha' > 1$ such that $P^1 + \alpha'(P^2 - P^1)$ is positive semidefinite but not positive definite, *i.e.*, one of eigenvalues of $P^1 + \alpha'(P^2 - P^1)$ becomes zero. Thus, $Y(\alpha')$ has a rank less than d but feasible to (4.5), which contradicts the fact that the graph cannot be 'flattened' to a lower dimensional space. Let $\bar{U}$ be an optimal dual matrix of (4.6). Then, any optimal solution matrix $\bar{Y}$ satisfies the complementarity condition $\bar{Y}\bar{U} = \mathbf{0}$. Note that $\bar{U}\mathbf{e} = \mathbf{e}$, where $\mathbf{e}$ is the vector of all ones. Thus, we have $\bar{X}^T\bar{X}\mathbf{e} = \bar{Y}\mathbf{e} = \bar{Y}\bar{U}\mathbf{e} = \mathbf{0}$, which further implies that $\bar{X}\mathbf{e} = \mathbf{0}$. □

Theorem 4.1 has an important implication in a distributed graph localization algorithm. It suggests that if a subgraph is d-localizable, then the sub-configuration is the same (up to translation, rotation and reflection) as the corresponding portion in the global configuration. Thus, one may attempt to localize a large graph by finding a sequence of d-localizable subgraphs. Theorem 4.1 also says that if the graph G is d-localizable, then the optimal solution of the SDP is given by $\bar{Y} = \bar{X}^T\bar{X}$ for some $\bar{X} = [\bar{x}_1, \ldots, \bar{x}_n] \in \mathcal{R}^{d\times n}$ such that $\bar{X}\mathbf{e} = 0$. It is now clear that when G is d-localizable, we have $\bar{Y} = \bar{X}^T\bar{X}$, and hence $\bar{Y}_{jj} = \|\bar{x}_j\|^2$ for $j = 1, \ldots, n$. But in general, when the given distances are not exact but corrupted with noises, we only have $\bar{Y} - \bar{X}^T\bar{X} \succeq 0$. This inequality however, may give a measure of the quality of the estimated positions. For example, the individual trace

$$T_j \;:=\; \bar{Y}_{jj} - \|\bar{x}_j\|^2, \tag{4.7}$$

may give an indication on the quality of the estimated position $\bar{x}_j$, where a smaller trace indicates more accuracy in the estimation. This is the same idea as was discussed for the case with anchors in (2.7).

4.5 Regularization term

As mentioned before in Chapter 3.5, we add additional terms to the objective function that force the SDP solution to lie in a low dimension, in order to counter the property of interior point algorithms to compute high-rank solutions. Our strategy is to convert the feasibility problem (4.1) into an maximization problem using the following regularization term as the objective function,

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \|x_i - x_j\|^2. \tag{4.8}$$

The new optimization problem is

$$\begin{array}{ll} \text{Maximize} & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} \|x_i - x_j\|^2 \\ \text{subject to} & \underline{d}_{ij}^2 \leq \|x_i - x_j\|^2 \leq \overline{d}_{ij}^2 \quad \forall (i,j) \in \mathcal{N}_u \\ & \displaystyle\sum_{i=1}^{n} x_i = 0. \end{array} \tag{4.9}$$

Note that the regularization term can also be expressed as $(I - \mathbf{e}\mathbf{e}^T/n) \bullet Y$, where $\mathbf{e}$ is the vector of all ones. The constraint $\sum_{i=1}^{n} x_i = 0$, which is added to remove the translational invariance of the configuration of points by putting the center of gravity at the origin, can also be expressed as $Y\mathbf{e} = 0$. So the objective function reduces to $\text{Trace}(Y) = I \bullet Y$. Finally, the SDP relaxation is

$$\begin{array}{ll} \text{Maximize} & I \bullet Y \\ \text{subject to} & \underline{d}_{ij}^2 \leq e_{ij}^T Y e_{ij} \leq \overline{d}_{ij}^2, \quad \forall\ (i,j) \in \mathcal{N}_u, \\ & Y\mathbf{e} = 0, \quad Y \succeq 0. \end{array} \tag{4.10}$$

The regularization idea works even better in the inequality setting (4.10) than in the model considered previously in (3.15). There is no error minimization term in the objective function and so we no longer need to worry about finding the right balance in weighting the different terms of the objective function. But one important point to be careful about is that in the case of very sparse distance data, often there may be 2 or more disjoint blocks within the given distance matrix, *i.e.*, the graph represented by the distance matrix may not be connected, and have more than 1 component. In that case, using the regularization term leads to an unbounded objective function since the disconnected components can be pulled as far apart as possible. Therefore, care must be taken to identify the disconnected components before applying model (4.10) to the individual components.

4.6 Clustering

The SDP (4.10) is computationally not tractable when there are several hundreds points. Therefore we divide the entire molecule into smaller clusters of points and solve a separate

SDP for each cluster. The clusters need to be chosen such that there should be enough distance information between the points in a cluster for it to be localized accurately, but at the same time only enough that it can also be solved efficiently. In order to do so, we make use of matrix permutations that reorder the points in such a way that the points which share the most distance information amongst each other are grouped together. In the problem described in this chapter, we have an upper and a lower bound distance matrix, but for simplicity, we will describe the operations in this section on just the partial distance matrix D. In the actual implementation, the operations described are performed on both the upper and lower bound distance matrices. This does not make a difference because the operations performed here basically exploit information only about the connectivity graph of the set of points.

We perform a symmetric permutation of the partial distance matrix D to aggregate the non-zero elements towards the main diagonal. Let $\tilde{D}$ be the permuted matrix. In our implementation, we used the function `symrcm` in MATLAB to perform the symmetric reverse Cuthill-McKee permutation [33] on D. Then the matrix $\tilde{D}$ is partitioned into a quasi-block-diagonal matrix with variable block-sizes. Let the blocks be denoted by $D_1, \ldots, D_L$. A schematic diagram of the quasi-block-diagonal structure is shown in Figure 4.1. The size of each block (except the last) is determined as follows. Starting with a minimum block-size, say 50, we extend the block-size incrementally until the number of non-zero elements in the block is above a certain threshold, say 1000. We start the process of determining the size of each block from the upper-left corner and sequentially proceed to the lower right-corner of $\tilde{D}$. Geometrically, the above partitioning process splits the job of determining the global configuration defined by $\tilde{D}$ into L smaller jobs, each of which try to determine the sub-configuration defined by D_k, and then assemble the sub-configurations sequentially from $k = 1$ to L to reconstruct the global configuration.

Observe that there are overlapping sub-blocks between adjacent blocks. For example, the second block overlaps with the first at its upper-left corner, and overlaps with the third at its lower-right corner. The overlapping sub-blocks serve an important purpose. For convenience, consider the overlapping sub-block between the second and third blocks. This overlapping sub-block corresponds to points that are common to the configurations defined by their respective distance matrices, D_2 and D_3. If the third block determines a localizable configuration X_3, then the common points in the overlapping sub-block can be used to stitch the localized configuration X_3 to the current global configuration determined by the first two blocks. In general, if the k^{th} block is localizable, then the overlapping sub-block between the $k-1^{st}$ and k^{th} blocks will be used to stitch the k^{th} localized configuration to the global configuration determined by the aggregation of all the previous blocks.

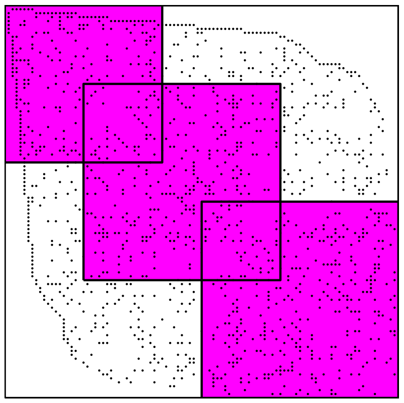


Figure 4.1: Schematic diagram of the quasi-block-diagonal structure considered in the distributed SDP algorithm.

As the overlapping sub-blocks are crucial in stitching a sub-configuration to the global configuration, it should have as high connectivity as possible. In our implementation, we find the following strategy to be reasonably effective. After the blocks $D_1, \ldots, D_L$ are determined, starting with D_2, we perform a symmetric reverse Cuthill-McKee permutation to the (2,2) sub-block of D_2 that is not overlapping D_1, and repeat the same process sequentially for all subsequent blocks. To avoid introducing excessive notation, we still use D_k to denote the permuted k^{th} block. It is also worth noting that in the case of highly noisy or very sparse data, the size of the overlapping sub-blocks needs to be set higher for the stitching phase to succeed. The more common points there are between 2 blocks, the more robust the stitching between them. This is also true as the number of sub-blocks which need to be stitched is large (*i.e.*, the number of atoms in the molecules is large). However, increasing the number of common points also has an impact on the runtime, and therefore we must choose the overlapping sub-block sizes judiciously. Experiments indicated that a sub-block size of 15-20 was sufficient for molecules with less than 3000 atoms but sub-block sizes of 25-30 were more suitable for larger molecules.

4.7 Stitching

After all the individual localization problems corresponding to $D_1, \ldots, D_L$ have been solved, we have L sub-configurations that need to be assembled together to form the global configuration associated with $\tilde{D}$. Suppose the current configuration determined by the blocks D_i, $i = 1, \ldots, k-1$ is given by the matrix $X^{(k-1)} = [U^{(k-1)}, V^{(k-1)}]$. Suppose also that $F^{(k-1)}$ records the global indices of the points that are currently labeled as localized in the current global configuration. Let $X_k = [V_k, W_k]$ be the points in the sub-configuration determined by D_k. (For $k = 1$, V_k is the null matrix.) Here $V^{(k-1)}$ and V_k denote the positions of the points corresponding to the overlapping sub-block between D_{k-1} and D_k, respectively. Let I_k be the global indices of the points in W_k. Note that the global indices of the unlocalized points for the blocks $D_1, \ldots, D_{k-1}$ is given by $J^{(k-1)} = \bigcup_{i=1}^{k-1} I_i \setminus F^{(k-1)}$.

We will now concentrate on stitching the sub-configuration D_k with the global index set I_k. Note that the points in the sub-configuration D_k, which have been obtained by solving the SDP on D_k, will most likely contain points that have been correctly estimated and points that have been estimated less accurately. It is essential that we isolate the badly estimated points, so as to ensure that their errors are not propagated when estimating subsequent blocks and that we may recalculate their positions when more points have been correctly estimated. To detect the badly estimated points, we use a combination of 2 error measures. Let x_j be the position estimated for point j. Set $\widehat{F} \leftarrow F^{(k-1)}$. We use the trace error T_j from Equation (4.7) and the local error

$$\widehat{E}_j = \frac{\sum_{i \in \widehat{\mathcal{N}}_{uj}} (\|x_i - x_j\| - d_{ij})^2}{|\widehat{\mathcal{N}}_{uj}|}, \tag{4.11}$$

where $\widehat{\mathcal{N}}_{uj} = \{i \in \widehat{F} \, : \, i < j, \tilde{D}_{ij} \neq 0\}$. We require $\max\{T_j, \widehat{E}_j\} \leq T_\epsilon$ as a necessary condition for the point to be correctly estimated. In the case when we are just provided with upper and lower bounds on distances, we use the mean distance in the local error measure calculations, *i.e.*,

$$d_{ij} = (\overline{d}_{ij} + \underline{d}_{ij})/2. \tag{4.12}$$

The use of multiple error measures is crucial, especially in cases when the distance information provided is noisy. In the noise free case, it is much easier to isolate points which are badly estimated using only the trace error T_j. But in the noisy case, there are no reliable error measures. By using multiple error measures, we hope to identify all the bad points which might possibly escape detection when only a single error measure is used. Setting the tolerances T_ϵ for the error is also an important parameter selection issue. For very sparse distance data and highly noisy cases, where even accurately estimated points may have significant error measure values, there is a tradeoff between selecting the tolerance and the number of points that are flagged as badly estimated. If the tolerance is too tight, we might end up discarding too many points that are actually quite accurately estimated. The design of informative error measures is still an open issue and there is room for improvement. As our results will show, the stitching phase of the algorithm is one which is most susceptible to noise and inaccurate estimation, and we need better error measures to make it more robust.

Now we have two cases to consider in the stitching process:

1. Suppose that there are enough points in the overlapping sub-blocks V_k and $V^{(k-1)}$ that are well estimated/localized, then we can stitch the k^{th} sub-configuration directly to the current global configuration $X^{(k-1)}$ by finding the affine mapping that matches points in $V^{(k-1)}$ and V_k as closely as possible. Mathematically, we solve the following linear least squares problem:

$$\min \left\{ \|B(V_k - \alpha) - (V^{(k-1)} - \beta)\|_F \;:\; B \in \mathcal{R}^{d\times d} \right\}, \tag{4.13}$$

   where α and β are the centroids of V_k and $V^{(k-1)}$, respectively. Once an optimal B is found, set

$$\widehat{X} = [U^{(k-1)}, V^{(k-1)}, \beta + B(W_k - \alpha)], \quad \widehat{F} \leftarrow F^{(k-1)} \bigcup I_k.$$

   We should mention that in the stitching process, it is very important to exclude points in $V^{(k-1)}$ and V_k that are badly estimated/unlocalized when solving (4.13) to avoid destroying the current global configuration. It should also be noted that there may be points in W_k that are incorrectly estimated in the SDP step. Performing the affine transformation on these points is useless, because they are in the wrong position in the local configuration to begin with. To deal with these points, we re-estimate the positions using those correctly estimated points as anchors. This procedure is exactly the same as what is described in case 2.

2. If there are not enough common points in the overlapping sub-blocks, then the stitching process described in (a) cannot be carried out successfully. In this case, the solution obtained from the SDP step for D_k is discarded, *i.e.*, the positions in W_k are discarded and they are to be determined via the current global configuration $X^{(k-1)}$ point-by-point as follows:

   Set $\widehat{X} \leftarrow X^{(k-1)}$, and $\widehat{F} \leftarrow F^{(k-1)}$. Let $T_\epsilon = 10^{-4}$. For $j \in J^{(k-1)} \bigcup I_k$,

   (a) Formulate a new SDP with $\mathcal{N}_u = \emptyset$ and $\mathcal{N}_a = \{(i,j) \, : \, i \in \widehat{F}, \;\; \tilde{D}_{ij} \neq 0\}$, where the anchor points are given by $\{\widehat{X}(:,i) \, : \, (i,j) \in \mathcal{N}_a\}$.

   (b) Let x_j and T_j be the newly estimated position and trace error from the previous step. Compute the local error measure $\widehat{E}_j$.

   (c) If $j \in J^{(k-1)}$, set $\widehat{X}(:,j) = x_j$; else, set $\widehat{X} = [\widehat{X}, x_j]$. If $\min\{T_j, \widehat{E}_j\} \leq T_\epsilon$, then set $\widehat{F} \leftarrow \widehat{F} \cup \{j\}$, end.

   Notice that in attempting to localize the points corresponding to W_k, we also attempt to estimate the positions of those previously unlocalized points, whose indices are recorded in $J^{(k-1)}$. Furthermore, we use previously estimated points as anchors to estimate new points. This not only helps in stitching new points into the current global configuration, but also increases the chances of correcting the positions of previously badly estimated points (since more anchor information is available when more points are correctly estimated).

4.8 A distributed SDP algorithm for anchor-free graph realization

We will now describe the complete distributed algorithm for solving a large scale anchor-free graph realization problem. To facilitate the description of our distributed algorithm for anchor-free graph realization, we first describe the centralized algorithm for solving model (4.1). It is important to note here that the terms localization and realization are used interchangeably.

Centralized graph localization (CGL) algorithm.
Input: $(\underline{D}, \overline{D}, \mathcal{N}_u; \{a_1, \ldots, a_m\}, \mathcal{N}_a)$.

Output: Estimated positions, $[\bar{x}_1, \ldots, \bar{x}_n] \in \mathcal{R}^{d\times n}$, and corresponding accuracy measures; trace errors, $T_1, \ldots, T_n$ and local error measures $\widehat{E}_1, \ldots, \widehat{E}_n$.

1. Formulate the optimization problem (4.9) and solve the resulting SDP. Let $X = [x_1, \ldots, x_n]$ be the estimated positions obtained from the SDP solution.
2. Perform the local refinement algorithm from Section 3.6 using the SDP solution as the starting iterate to get more refined estimated positions $[\bar{x}_1, \ldots, \bar{x}_n]$.
3. For each $j = 1, \ldots, n$, label the point j as localized or unlocalized based on the error measures T_j and $\widehat{E}_j$.

Distributed anchor free graph localization (DAFGL) algorithm.
Input : Upper and lower bounds on a subset of the pairwise distances in a molecule.
Output : A configuration of all the atoms in the molecule that is closest (in terms of the RMSD error described in Section 4.9) to the actual molecule (from which the measurements were taken).

1. Divide the entire point set into sub-blocks using the clustering algorithm described in Section 4.6 on the sparse distance matrices.
2. Apply the CGL algorithm to each sub-block.
3. Stitch the sub-blocks together using the procedure described in Section 4.7. After each stitching phase, refine the point positions again using the gradient descent based local refinement method described in Section 3.6 and update their error measures.

Some remarks are in order for the above algorithm. In Step 3, we can solve each cluster individually and the computation is highly distributive. In using the centralized CGL algorithm to solve each cluster, the computational cost is dominated by the solution of the SDP problem(the SDP cost is in turn determined by the number of given distances). For a graph with n nodes and m given pairwise distances, the computational complexity in solving the SDP is roughly $O(m^3) + O(n^3)$, provided sparsity in the SDP data is fully exploited. For a graph with 200 nodes and the number of given distances is 10% of the total number of pairwise distances, the SDP would roughly have 2000 equality constraints and matrix variables of dimension 200. Such an SDP can be solved on a Pentium IV 3.0 GHz PC with 2GB RAM in about 36 and 93 seconds using the general purpose SDP software

SDPT3-3.1 [90] and SeDuMi-1.05 [82], respectively. The computational efficiency in the CGL algorithm can certainly be improved in various ways. First, the SDP problem need not be solved to high accuracy. It is sufficient to have a low accuracy SDP solution if it is only used as a starting iterate for the local refinement algorithm. There are various highly efficient methods (such as iterative solver based interior-point methods [83] or the SDPLR method of Burer and Monteiro [20]) to obtain a low accuracy SDP solution. Second, a dedicated solver based on a dual scaling algorithm can also speed up the SDP computation. Substantial speed up can be expected if the computation exploits the low rank structure present in the constrain matrices. However, as our focus in this chapter is not on improving the computational efficiency of the CGL algorithm, we shall not discuss this issue further. In the numerical experiments conducted in Section 4.9, we use the softwares SDPT3-3.1 and SeDuMi to solve the SDP in the CGL algorithm. An alternative is to use the software DSDP-5.8 [7], which is expected to be more efficient than SDPT3-3.1 or SeDuMi.

The stitching process in Step 4 is sequential in nature. But this does not imply that the distributed DAFGL algorithm is redundant and that the centralized CGL algorithm is sufficient for computational purposes. For a graph with 10000 nodes and the number of given distances is 1% of the total number of all pairwise distances, the SDP problem that needs to be solved by the CGL algorithm would have 500000 constraints and matrix variables of dimension 10000. Such a large scale SDP is well beyond the range that can be solved routinely on a standard workstation available today. By considering smaller blocks, the distributed algorithm does not suffer from the scalability limitation faced by the CGL algorithm. If there are multiple computer processors available, say p of them, the distributed algorithm can also take advantage of the extra computing power. The strategy is to divide the graph into p large blocks using Step 2 of the algorithm and apply the DAFGL algorithm to localize one large block on each processor.

In our implementation, we use the gradient refinement step quite extensively, both after the SDP step for a single block, and also after each stitching phase between 2 blocks. In the single block case, the calculation does not involve the use of any anchor points, but when used after stitching, we fix the previous correctly estimated points as anchors. The formula used in the gradient calculations changes accordingly. It should be noted that the more sophisticated refinement methods described in Subsection 3.6.2 can also be used, and are expected to provide even better accuracy. Tuning the truncated Newton method for the molecule conformation problem is a part of our future research.

We end this section with 2 observations on the DAFGL algorithm. First, we observed that usually the outlying points which have low connectivity are not well estimated in the initial stages of the method. As the number of well estimated points grow gradually, more and more of these 'loose' points are estimated by the gradient descent algorithm. As the molecules get larger, the frequency of having non-localizable sub-configurations in Step 4 also increase. Thus the point-by-point stitching procedure of the algorithm described in Section 4.7 gets visited more and more often. Second, for large molecules, the sizes of the overlapping-blocks need to be larger for the stitching algorithm in Section 4.7 to be robust (more common points generally lead to more accuracy in stitching). But to accommodate larger overlapping-blocks, each subgraph in the DAFGL algorithm will correspondingly be larger, and that in turns increases the problem size of the SDP relaxation. In our implementation, we apply the idea of dropping redundant constraints to reduce the computational effort in selecting large sub-block sizes of 100-150. This strategy works because many of the distance constraints are for some of the densest parts of the sub-block, and the points in these dense sections can actually be estimated quite well with only a fraction of those distance constraints. Therefore in the SDP step, we limit the number of distance constraints for each point to below 6. If there are more distance constraints, they are not included in the SDP step. This allows us to choose large overlapping-block sizes while the corresponding SDPs for larger clusters can be solved without too much additional computational effort.

4.9 Experimental results

To evaluate our DAFGL algorithm, numerical experiments were performed on protein molecules with the number of atoms in each molecule ranging from a few hundreds to a few thousands. We conducted our numerical experiments in Matlab on a single Pentium IV 3.0 GHz PC with 2GB of RAM. The known 3D coordinates of the atoms were taken from the Protein Data Bank (PDB) [8]. These were used to generate the true distances between the atoms. Our goal in the experiments is to reconstruct as closely as possible the known molecular configuration for each molecule, using only distance information generated from a sparse subset of all the pairwise distances. This information was in the form of upper and lower bounds on the actual pairwise distances.

For each molecule, we generated the partial distance matrix as follows. If the distance between 2 atoms was less than a given cutoff radius R, the distance is kept; otherwise, no distance information is known about the pair. The cutoff radius R is chosen to be 6Å (1Å = 10^{-8}cm), which is roughly the maximum distance that NMR techniques can measure between two atoms. Therefore, in this case, $\mathcal{N}_u = \{(i,j) : \|x_i - x_j\| \leq 6\text{Å}\}$. We then perturb the distances to generate upper and lower bounds on the given distances in the following manner. Assume that $\hat{d}_{ij}$ is the true distance between atom i and atom j, we set

$$\begin{aligned} \overline{d}_{ij} &= \hat{d}_{ij}(1 + |\overline{\epsilon}_{ij}|) \\ \underline{d}_{ij} &= \hat{d}_{ij} \max(0, 1 - |\underline{\epsilon}_{ij}|), \end{aligned}$$

where $\overline{\epsilon}_{ij}, \underline{\epsilon}_{ij} \sim N(0, \sigma_{ij}^2)$. By varying σ_{ij} (which we keep as the same for all pairwise distances), we control the noise in the data. This is a multiplicative noise model, where a higher distance value means more uncertainty in its measurement. For all future reference, we will refer to σ_{ij} in percentage values. For example, $\sigma_{ij} = 0.1$ will be referred to as 10% noise on the upper and lower bounds. Typically, not all the distances below 6Å are known from NMR experiments. Therefore, we will also present results for the DAFGL algorithm when only a fraction of all the distances below 6Å are chosen randomly and used in the calculation.

Let $\mathcal{Q}$ be the set of orthogonal matrices in $\mathcal{R}^{d\times d}$ ($d = 3$). We measure the accuracy performance of our algorithm by the following criteria:

$$\text{RMSD} = \frac{1}{\sqrt{n}} \min \left\{ \|X^{\text{true}} - QX - h\|_F \mid Q \in \mathcal{Q},\ h \in \mathcal{R}^d \right\} \tag{4.14}$$

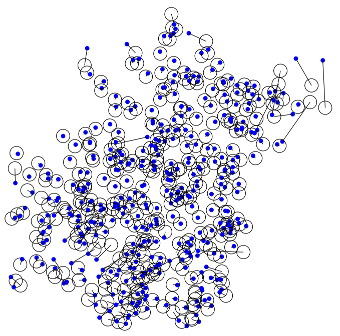
$$\text{LDME} = \left(\frac{1}{|\mathcal{N}_u|} \sum_{(i,j)\in\mathcal{N}_u} \left(\|x_i - x_j\| - d_{ij} \right)^2 \right)^{1/2}. \tag{4.15}$$

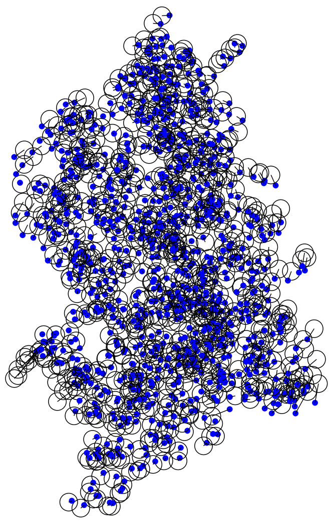
The first criterion requires the knowledge of the true configuration, whereas the second does not. Thus the second criterion is more practical but it is also less reliable in evaluating the true accuracy of the constructed configuration. The practically useful measure LDME gives lower values than the RMSD, and as the noise increases, it is not a very reliable measure.

When 90% of the distances (below 6Å) or higher were considered, and were not corrupted by noise, the molecules considered here were estimated very precisely with RMSD = 10^{-4} – 10^{-6}Å. This goes to show that the algorithm performs remarkably well when there is enough exact distance data. In this regard, our algorithm is competitive compared to the geometric build-up algorithm in [96] which is designed specifically for graph localization with exact distance data. With exact distance data, we have solved molecules that are much larger and with much sparser distance data than those considered in [96]. However, our focus in this work is more on molecular conformation with noisy and sparse distance data. We will only present results for such cases. In Figure 4.2, the original true and the estimated atom positions are plotted for some of the molecules, with varying amounts of distance data and noise. Once again, as in the 2-D case, the open (black) circles correspond to the true positions and solid (blue) dots to their estimated positions from our computation. The error offset between the true and estimated positions for an individual atom is depicted by a solid (black) line. The solid lines, however, are not visible for most of the atoms in the plots because we are able to estimate the positions very accurately.

The plots show that even for very sparse data (as in the case of 1PTQ), the estimation for most of the atoms is accurate. The atoms that are badly estimated are the ones that have too little distance information for them to be localized. The algorithm also performs well for very noisy data, and for large molecules, as demonstrated for 1AX8 and 1TOA. The largest molecule we tried the algorithm on is 1I7W, with 8629 atoms. Figure 4.3 shows that even when the distance data is highly noisy (10 % error on upper and lower bounds), the estimation is close to the original with an RMSD error = 1.3842Å.

However, despite using more common points for stitching, the DAFGL algorithm can sometimes generate estimations with high RMSD error for very large molecules due to a combination of irregular geometry, very sparse distance data and noisy distances. Ultimately, the problem boils down to being able to correctly identify when a point has been badly estimated. Our measures using trace error (4.7) and local error (4.11) are able to isolate the majority of the points that are badly estimated, but do not always succeed when many of the points are badly estimated. In fact, for such molecules, a lot of the computation time is spent in the point-by-point stitching phase, where we attempt to repeatedly solve for better estimations of the badly estimated points, and if that fail repeatedly, the estimations continue to be poor. If the number of badly estimated points is very high, it may affect the stitching of the subsequent clusters as well. In such cases, the algorithm more or less fails to find a global configuration. Examples of such cases are the 1NF7 molecule (5666 atoms) and the 1HMV molecule (7398 atoms) solved with 10% noise. While they are estimated correctly when there is no noise, the 1NF7 molecule estimation returns an RMSD of 25.1061Å and the 1HMV molecule returns 28.3369Å. A more moderate case of stitching failure can


(a) 1PTQ(402 atoms) with 30% of distances below 6Å and 1% noise on upper and lower bounds, RMSD = 0.9858Å


(b) 1AX8(1003 atoms) with 70% of distances below 6Å and 5% noise on upper and lower bounds, RMSD = 0.8184Å

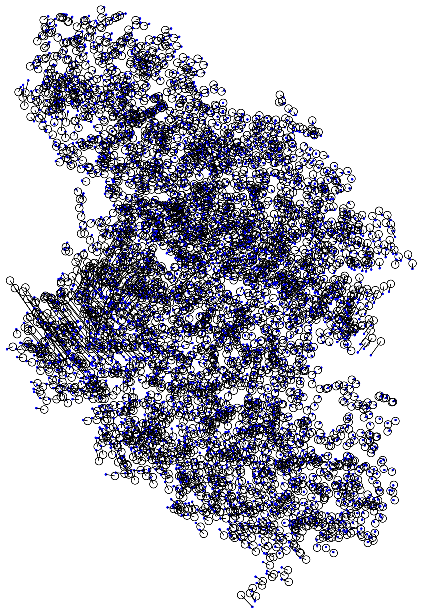

(c) 1TOA(4292 atoms) with 100% of distances below 6Å and 10% noise on upper and lower bounds, RMSD = 1.5058Å

Figure 4.2: Comparison of actual and estimated molecules.

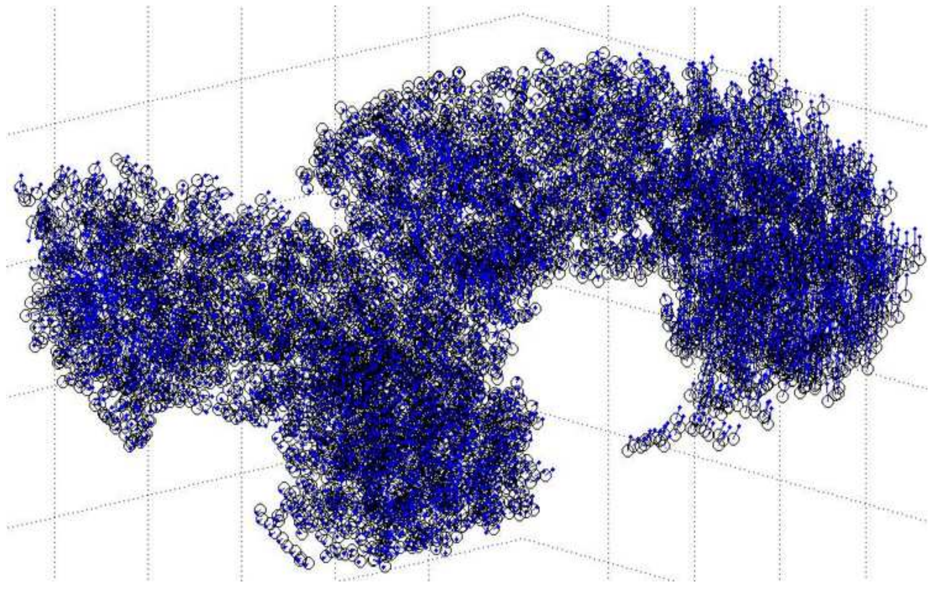


Figure 4.3: 1I7W(8629 atoms) with 100% of distances below 6Å and 10% noise on upper and lower bounds, RMSD = 1.3842Å.

be seen in Figure 4.4 for the molecule 1BPM with 50% of the distance below 6Å, and 5% error on upper and lower bounds, the problem is in particular clusters (which are encircled in the figure). Although they have correct local geometry, their positions with respect to the entire molecule are not. This indicates that the stitching procedure has failed because some of the common points are not estimated correctly, and are then used in the stitching process, thus destroying the entire local configuration. So far, this is the weakest part of the algorithm and future work is heavily focussed on developing better error measures to isolate the badly estimated points and to improve the robustness of the stitching process.

In Figure 4.5, we plotted the 3-dimensional configuration (via Swiss PDBviewer [35]) of some of the molecules to the left and their estimated counterparts (with different distance data inputs) to the right. As can be seen clearly, the estimated counterparts closely resemble the original molecules.

The results shown in the following tables are a good representation of the performance of the algorithm on different size molecules with different types of distance data sets. The numerical results presented in Table 4.1 are for the case when all distances below 6Å are used and perturbed with 10% noise on lower and upper bounds. Table 4.2 contains the results for the case when only 70% of distances below 6Å are used and perturbed with 5%

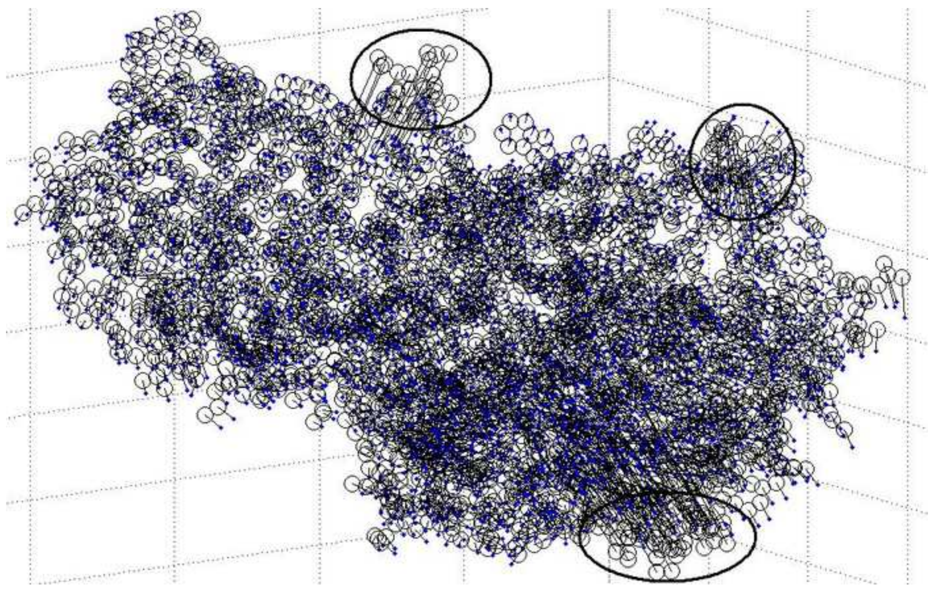
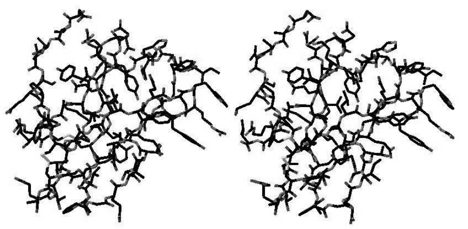


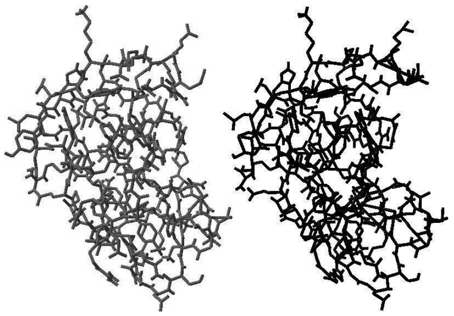
Figure 4.4: 1BPM(3672 atoms) with 50% of distances below 6Å and 5% noise on upper and lower bounds, RMSD = 2.4360 Å.

noise and also for the case when only 50% of distances below 6Å are used and perturbed with 1% noise. The results for 50% distance and 1% noise are representative of cases with sparse distance information and low noise, 100% distance and 10% noise represent relatively denser but highly noisy distance information and 70% distance and 5% noise is a middle ground between the 2 extreme cases.

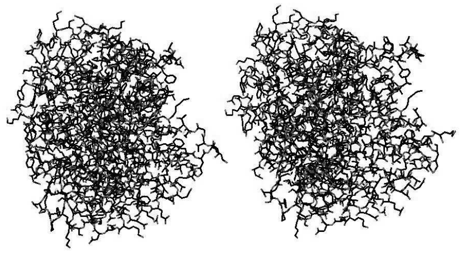
We can see from the values in Table 4.1 that LDME is not a very good measure of the actual estimation error given by RMSD since the former does not correlate well with the latter. Therefore we do not report the LDME values in Table 4.2. From the tables, it can be observed that for relatively dense distance data (100% of all distances below 6Å), the estimation error stays below 2Å even when the upper and lower bounds are very loose. The algorithm is seen to be quite robust to high noise when there is enough distance information. The estimation error is also quite low for most molecules for cases when the distance information is very sparse but much more precise. In the sparse distance cases, it is the molecules that have more irregular geometries that suffer the most from lack of enough distance data and exhibit high estimation errors. The combination of sparsity and noise has a detrimental impact on the algorithm's performance, as can be seen for the results with 70% distances, and 5% noise.



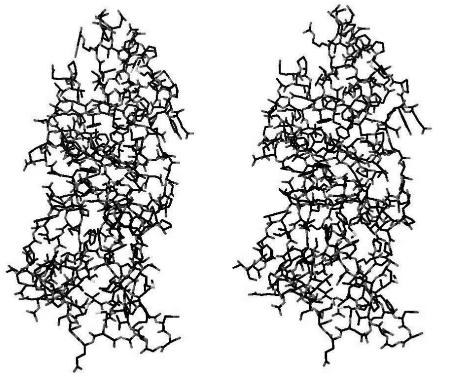
(a) 1HOE(558 atoms) with 40% of distances below 6Å and 1% noise on upper and lower bounds, RMSD = 0.2154Å



(b) 1PHT(814 atoms) with 50% of distances below 6Å and 5% noise on upper and lower bounds, RMSD = 1.2014Å



(c) 1RHJ(3740 atoms) with 70% of distances below 6Å and 5% noise on upper and lower bounds RMSD = 0.9535Å



(d) 1F39(1534 atoms) with 85% of distances below 6Å and 10% noise on upper and lower bounds, RMSD = 0.9852Å

Figure 4.5: Comparison between the original (left) and reconstructed (right) configurations for various protein molecules using Swiss PDB viewer.

Table 4.1: Results for 100% of Distances below 6Å and 10% noise on upper and lower bounds

PDB ID	No. of atoms	% of total pairwise distances used	RMSD(Å)	LDME(Å)	CPU time (secs)
1PTQ	402	8.79	0.1936	0.2941	107.7
1HOE	558	6.55	0.2167	0.2914	108.7
1LFB	641	5.57	0.2635	0.1992	129.1
1PHT	814	5.35	1.2624	0.2594	223.9
1POA	914	4.07	0.4678	0.2465	333.1
1AX8	1003	3.74	0.6408	0.2649	280.1
1F39	1534	2.43	0.7338	0.2137	358.0
1RGS	2015	1.87	1.6887	0.1800	665.9
1KDH	2923	1.34	1.1035	0.2874	959.1
1BPM	3672	1.12	1.1965	0.2064	1234.7
1RHJ	3740	1.10	1.8365	0.1945	1584.4
1HQQ	3944	1.00	1.9700	0.2548	1571.8
1TOA	4292	0.94	1.5058	0.2251	979.5
1MQQ	5681	0.75	1.4906	0.2317	1461.1

In Figure 4.6, we plotted the CPU times required by our DAFGL algorithm to localize a molecule with n atoms versus n (with different types of distance inputs). As the distance data becomes more and more sparse, the number of constraints in the SDP also reduce, and therefore they take less time to be solved in general. However, many points may be incorrectly estimated in the SDP phase and so the stitching phase usually takes longer in these cases. This behavior is exacerbated by the presence of higher noise. Our algorithm is reasonably efficient in solving large problems. The spikes that we see in the graphs for some of the larger molecules also correspond to cases with high RMSD error, in which the algorithm fails to find a valid configuration of points, either due to very noisy or very sparse data. A lot of time is spent in recomputing points in the stitching phase, and many of these points are repeatedly estimated incorrectly. The number of badly estimated points grows at each iteration of the stitching process and becomes more and more time consuming. But, given the general computational performance, we expect our algorithm to be able to handle even larger molecules if the number of badly estimated points can be kept low. On the other hand, we are also investigating methods that discard repeatedly badly estimated

Table 4.2: Results with 70% of distances below 6Å and 5% noise on upper and lower bounds and with 50% of distances below 6Å and 1% noise on upper and lower bounds

PDB ID	No. of atoms	70% Distances, 5% Noise		50% Distances, 1% Noise	
		RMSD(Å)	CPU time (secs)	RMSD(Å)	CPU time (secs)
1PTQ	402	0.2794	93.8	0.7560	22.1
1HOE	558	0.2712	129.6	0.0085	32.5
1LFB	641	0.4392	132.5	0.2736	41.6
1PHT	814	0.4701	129.4	0.6639	53.6
1POA	914	0.4325	174.7	0.0843	54.1
1AX8	1003	0.8184	251.9	0.0314	71.8
1F39	1534	1.1271	353.1	0.2809	113.4
1RGS	2015	4.6540	613.3	3.5416	308.2
1KDH	2923	2.5693	1641.0	2.8222	488.4
1BPM	3672	2.4360	1467.1	1.0502	384.8
1RHJ	3740	0.9535	1286.1	0.1158	361.5
1HQQ	3944	8.9106	2133.5	1.6610	418.4
1TOA	4292	9.8351	2653.6	1.5856	372.6
1MQQ	5681	3.1570	1683.4	2.3108	1466.2

points from future calculations.

4.10 Conclusion and work in progress

An SDP based distributed method to solve the distance geometry problem in 3-D with incomplete and noisy distance data and without anchors is described. The entire problem is broken down into subproblems by intelligent clustering methods. An SDP relaxation problem is formulated and solved for each cluster. Matrix decomposition is used to find local configurations of the clusters and a least squares based stitching method is applied to find a global configuration. Gradient descent methods are also employed in intermediate steps to refine the quality of the solution. The performance of the algorithm is evaluated by using it to find the configurations of large protein molecules with a few thousands atoms. The distributed SDP approach can solve with good accuracy and speed large problems with favorable geometries when 50-70% distances below 6Å are given and they are contaminated

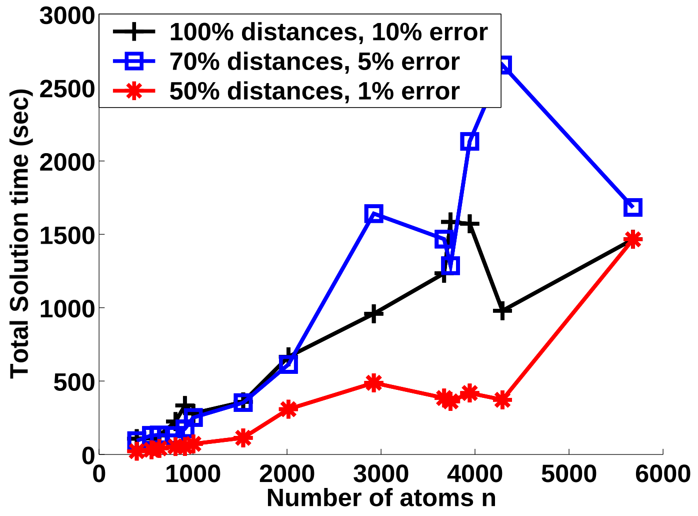


Figure 4.6: CPU time taken to localize a molecule with n atoms versus n.

only with moderate level of noises (0-5%) in both the lower and upper bounds. The current DAFGL algorithm needs to be improved for it to work on very sparse (30-50% of all distances below 6Å) and highly noisy data (10-20% noise), which is often the case for actual NMR data used to perform molecular conformation. For the rest of this section, we outline some possible improvements that can be made to our current algorithm.

One of the main difficulties we encounter in the current distributed algorithm is the propagation of position estimation errors in a cluster to other clusters during the stitching process when the given distances are noisy. Even though we have had some successes in overcoming this difficulty, it is not completely alleviated in the current work. The difficulties faced by our current algorithm with very noisy or very sparse data cases are particularly noticeable for very large molecules (which correspond to a large number of sub-blocks that need stitching). Usually, the estimation for many of the points in some of the sub-blocks from the CGL step is not accurate enough in these cases. This is especially problematic when there are too few common points that can then be used for stitching with the previous block, or if the error measures used to identify the bad points are unable to filter out some of the badly estimated common points. This usually leads to the error propagating to subsequent blocks as well. Therefore, the bad point detection and stitching phase need to made more robust.

To reduce the effect of inadvertently using a badly estimated point for stitching, we can increase the number of common points used in the stitching process, and at the same time, use more sophisticated stitching algorithms that not only stitch correctly estimated points, but also isolate the badly estimated ones. As far as stitching is concerned, currently we use the coordinates of the common points between 2 blocks to find the best affine mapping that would bring the 2 blocks into the same coordinate system. Another idea is to fix the values in the inner product matrix Y in (4.10) that correspond to the common points, based on the values that were obtained for it in solving the SDP for the previous block. By fixing the values, we are indirectly using the common points to anchor the new block with the previous one. The dual of the SDP relaxation also merits further investigation, for possible improvements in the computational effort required and for more robust stitching results. A third method that is worth exploring is the sophisticated stitching algorithm introduced recently in [105] for nonlinear dimensionality reduction.

With regard to error measures, it would be useful to study if the local error measure (4.11) can be made more sophisticated by including a larger set of points to check its distances with, as opposed to just its immediate neighborhood. In some cases, the distances are satisfied within local clusters, but the entire cluster itself is badly estimated (and the local error measure fails to filter out many of the badly estimated points in this scenario). Also, we need to investigate the careful selection of the tolerance T_ε in deciding which points have been estimated correctly and can be used for stitching.

In this chapter. we have only used the gradient descent method for local refinement. In subsection 3.6.2, we introduced the use of truncated Newton methods, which can provide more accurate estimations in fewer iterations. The use of these methods for local refinement should further improve the accuracy of the overall algorithm.

As has been noted before, our current algorithm does not use the VDW minimum separation distance constraints of the form $\|x_i - x_j\| \geq L_{ij}$ described in [68] and [95]. The VDW constraints played an essential role these previous work in reducing the search space of valid configurations, especially in the case of sparse data where there are many possible configurations fitting the sparse distance constraints. As mentioned in Section 3, VDW lower bound constraints can be added to the SDP model, but we would need to keep track of the type of atom each point corresponds to. However, one must be mindful that the VDW constraints should only be added when necessary so as not to introduce too many redundant constraints. If the VDW constraints between all pairs of atoms are added to the SDP model, then the number of lower bound constraints is of the order n^2, where n is the number of points. Thus even if n is below 100, the total number of constraints could be in the range of thousands. However, many of the VDW constraints are for two very remote points, and they are often inactive or redundant at the optimal solution. Therefore, we can adopt an iterative active constraint generation approach. We first solve the SDP problem by completely ignoring the VDW lower bound constraints to obtain a solution. Then we verify the VDW lower bound constraints at the current solution for all the points and add the violated ones into the model. We can repeat this process until all the VDW constraints are satisfied. Typically, we would expect only $O(n)$ VDW constraints to be active at the final solution.

We are optimistic that by combining the ideas presented in previous work on molecular conformation (especially incorporating domain knowledge such as the minimum separation distances derived from VDW interactions) with the distributed SDP based algorithm in this work, the improved distributed algorithm would likely be able to calculate the conformation of large protein molecules with satisfactory accuracy and efficiency in the future. Based on the performance of the current DAFGL algorithm, and the promising improvements to the algorithm we have outlined, a realistic target for us to set in the future is to correctly calculate the conformation of a large molecule (with 5000 atoms or more) given only about 50% of all pairwise distances less than 6Å, and corrupted by 10-20% noise.

Chapter 5

Using angle information

While the application of SDP relaxations to pure distance information has been discussed in previous chapters, a framework does not exist where we can use this technique for angle information. Solving this problem is particularly useful in a scenario where we use sensors that are also able to detect mutual angles, like in image sensor nodes.

There is a move towards towards creating designs and applications of large scale heterogenous sensor networks using multi-modal sensing technologies [81, 104]. So there is also a need to develop localization techniques that can use different types of ranging and proximity information that may have been acquired through different ranging mechanisms (Received Signal Strength, Time of Flight, Ultrasonic, light and image sensors) in an integrated fashion. This chapter takes a step in this direction by extending the SDP model to solve the localization problem using angle information in combination with distance information, or independently, using just angle information. Section 5.1 discusses some other approaches to using angle information for determining sensor positions. Section 5.2.1 describes the scenario envisioned for the problem.It explains the nature of the angle information available. In Section 5.2.2, we present the extension of the SDP model for scenarios where both distance and angle information are present. Using trigonometric relations, we arrive at an optimization problem with a set of quadratic constraints that can then be relaxed into an SDP. A more subtle idea is used in Section 5.2.3 to again obtain an optimization problem with quadratic constraints and subsequently relax it to an SDP. Practical considerations such as the computational effort and accuracy of the algorithm and their dependence upon factors such as communication range and field of view of sensors are discussed in Section 5.3. Extensive experimental results are presented in Section 5.4. Finally, we conclude with the current limitations of the technique and ideas for further research in Section 5.5.

5.1 Related work

The use of angle information for sensor network localization has been investigated by Niculescu and Nath [62]. It is assumed that the network has a few landmark or anchor points whose positions are already known. The algorithm uses information between the anchors and unknown points to set up triangulation equations in order to determine positions of unknown points. The calculated positions are then forwarded to other unknown points and used in further triangulation.

Priyantha et al. [67] describe the Cricket Compass system that uses ultrasonic measurements and fixed beacons to obtain robust estimates of the orientation of mobile devices. This is done by using positions of the fixed beacons and differential distance estimates received from ultrasonic sensors.

Recently, Gao et al. have introduced techniques to use local noisy angle information [18] and a combination of noisy angle and distance information [6] for localization. The basic ideas behind this work is to find bounded regions of feasible positions for all the points, which are obtained by solving a set of linear bounding constraints at each point. Similar bounding constraints have also been investigated by Doherty et al. [29]. We hope to use more global information in our approach, as opposed to a purely distributed point by point approach, which suffers from the drawback of using only the local information.

The use of orientation information for mobile robot localization has been dealt with extensively in Betke and Gurvitz [9]. A variant of this situation is considered in the context of camera sensor networks by Skraba et al. [78]. A moving beacon transmits its position information to sensor nodes that can also measure the angle of arrival of the signal. The sensors can determine their position and camera orientation based on information from the beacon node signals transmitted at different times. This is a purely distributed method with each node being able to configure itself independently.

We consider a slightly different situation where a moving beacon node is not available anymore. In fact, some or maybe all the nodes may not have any information relative to a fixed position. It becomes essential in this scenario to consider relative information between the sensors. In other words, there needs to be a collaborative method that exploits all the mutual information between the nodes. Techniques based on bounding using linear hyperplanes may be too loose to provide a meaningful solution. Our method also attempts to solve the position estimation problem in one step by solving the problem using global information. For very large networks of nodes, such global information may be the collective information shared between local clusters of nodes. This also avoids the issues of forwarding and error propagation as in [62]. Furthermore, by using global information, it is hoped that solutions are more accurate.

5.2 Integrating angle information

5.2.1 Scenario

The scenario for which we will describe our algorithm is very similar to that described in [62]. Consider n unknown points $x_i \in \mathcal{R}^2$, $i = 1, ..., n$. All angle measurements are made at each sensor relative to its own axis. The orientation θ_i of the axis with respect to a global coordinate system, after a random deployment of the network, is not known. There are also a set of m anchor nodes $a_k \in \mathcal{R}^2$, $k = 1, ..., m$, *i.e.*, nodes whose positions are known a priori. The orientations of their axes may or may not be known.

Every sensor can detect neighboring sensors which are within its field of view and a specified communication range of it. The field of view is limited by the maximum angle (on either side of the axis) that the sensor can detect with respect to its own axis. Depending on the type of sensing technology used, these parameters can be accordingly set. For example, the field of view for a typical image sensor is about 20-25 degrees on either side. On the other hand, for omnidirectional sensors with multiple cameras or antenna, the field of view can be made to be 180 degrees on either side. Since every measurement is with respect to the axis of a sensor node, we also have angle measurements corresponding to the angle between 2 other sensors (either anchors or unknown) as seen from the sensor, given that the 2 sensors are within communication distance range of it and within its field of view. The problem setup is described for a set of three points in Figure 5.1. The dark arrows correspond to axis orientations. The measurements given to us are the angles at a particular sensor that other sensors are seen at, with respect to its axis. The angle between sensor B and C, as seen from sensor A, can be obtained by subtracting the angle between sensor B and the axis of A, and the the angle between sensor C and the axis of A. The objective now is to find the positions of the unknown nodes, given many such measurements from all the sensors. From the position information, we can also derive the orientation information.

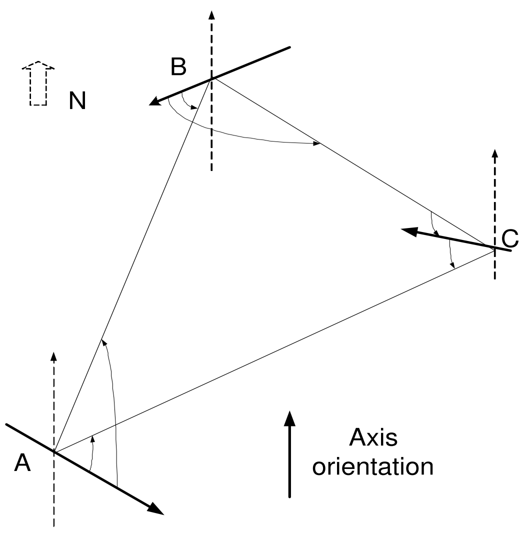


Figure 5.1: Sensor, Axis orientation and field of view.

5.2.2 Distance and angle information

The first case we will consider is when we have both angle and distance information. Consider the situation that we assume that every anchor node is already aware of its axis orientation with respect to the global coordinate system. Then for every anchor, we have an absolute angle measurement (*i.e.*, the angle with respect to the global coordinate system) corresponding to an unknown sensor, if the two are within a communication distance range R. Let the set of all such pairs of points be denoted by N_a. For a pair of two points in N_a, we have an absolute angle θ_{kj} between anchor a_k and unknown sensor x_j. Therefore, the angle information at the anchors can be expressed as a simple linear constraint in the following manner

$$\frac{a_k(2) - x_j(2)}{a_k(1) - x_j(1)} = \tan\theta_{kj}, \ \forall\ k, j \in N_a,$$
$$a_k(2) - x_j(2) = \tan\theta_{kj}(a_k(1) - x_j(1)),$$

where θ_{kj} is the angle at which the sensor j is detected by anchor k with respect to the global coordinate system, $a_k(1)$ and $a_k(2)$ are the x and y coordinates of the point a_k respectively. Similarly $x_j(1)$ and $x_j(2)$ are the x and y coordinates of the point x_j respectively. Since the equality can be written as

$$A_{kj}X = \tan\theta_{kj}a_k(1) - a_k(2), \tag{5.1}$$

where A_{kj} is a $2 \times n$ matrix with $A_{kj}(1,j)$ equal to $\tan\theta_{kj}$, $A_{kj}(2,j)$ equal to -1, and zero everywhere else, and $X = [x_1\ x_2\ \ldots\ x_n]$ is the $2 \times n$ matrix (corresponding to the point positions) that needs to be determined.

Also at every unknown sensor, the measurements of the angle and distance of every sensor within its communication range and field of view is available. Note that all the angle measurements are relative to its sensor axis. By subtracting the axis relative measurements at a sensor at x_j, we can find the angle between 2 other sensors at x_i and x_l as seen from the sensor at x_j. Let the set of all such triplets be denoted by N_b. For a set of three points in N_b, we have the angle θ_{kji} between a_k and x_i as seen from x_j as well as the distance d_{jk} and d_{ji}; or the angle θ_{ijl} between x_i and x_l as seen from x_j as well as the distance d_{ji} and d_{jl}. The angle information between unknown sensors i and l as seen from sensor j can be expressed as

$$\begin{aligned}
&\frac{(x_j - x_i)^T(x_j - x_l)}{\|x_j - x_i\|\|x_j - x_l\|} = \cos\theta_{ijl}, \text{ for } i,j,l \in N_b \text{ or} \\
&\frac{(x_j - x_i)^T(x_j - x_l)}{d_{ji}d_{jl}} = \cos\theta_{ijl}, \text{ or} \\
&\|x_l - x_i\|^2 = d_{ji}^2 + d_{jl}^2 - 2d_{ji}d_{jl}\cos\theta_{ijl},
\end{aligned} \tag{5.2}$$

where θ_{ijl} is the angle between sensor i and sensor l as seen from sensor j, d_{ji} is the distance between sensor j and sensor i and d_{jl} is the distance between sensor j and sensor l. The angle information between anchor k and unknown sensor i as seen from sensor j can be expressed as

$$\begin{aligned}
&\frac{(x_j - a_k)^T(x_j - x_i)}{\|x_j - a_k\|\|x_j - x_i\|} = \cos\theta_{kji}, \text{ for } k,j,i \in N_b, \text{ or} \\
&\frac{(x_j - a_k)^T(x_j - x_i)}{d_{jk}d_{ji}} = \cos\theta_{kji}, \text{ or} \\
&\|x_i - a_k\|^2 = d_{jk}^2 + d_{ji}^2 - 2d_{jk}d_{ji}\cos\theta_{kji},
\end{aligned} \tag{5.3}$$

where θ_{kji} is the angle between anchor k and sensor i as seen from sensor j, d_{ji} is the distance between sensor j and sensor i and d_{jk} is the distance between anchor k and sensor

j.

Now by applying the SDP relaxation described in previous chapters, *i.e.*, using the substitution $Y = X^T X$ and relaxing this constraint to $Y \succeq X^T X$, the problem consisting of the quadratic constraints of the type described above can be reduced once again to an SDP form. Note that the relaxation used is exactly the same as the one introduced in Chapter 2.

5.2.3 Pure angle information

Next, we will consider the localization problem for pure angle information. The problem can be formulated in a variety of ways depending upon how we choose to exploit the angle constraints in our equations. The idea is to maintain the Euclidean distance geometry 'flavor' in our approach where the quadratic term in the formulation is simply relaxed into an SDP. With this aim in mind, the following solution is presented. The same situation (with regard to angle measurements), as considered in the case of combined distance and angle information, will be used here as well. The only difference is that absolutely no distance information between sensors is available.

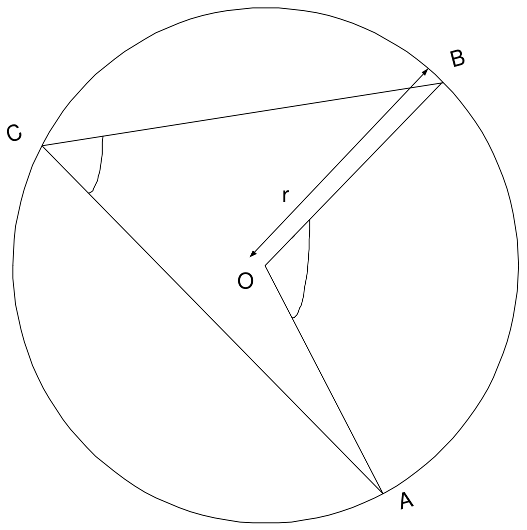


Figure 5.2: 3 points, the circumscribed circle and related angles, $\langle AOB\rangle = 2\langle ACB\rangle$.

Consider 3 points x_i, x_j and x_k that are not located on the same line. One basic property concerning circles is that the angle subtended by 2 of these points, say x_i and x_k, at the center of the circle that circumscribes the 3 points, is twice the angle subtended by the 2 points at the third point x_j, see Figure 5.2. Let the radius of the circumscribing circle be r_{ijk}, which is also unknown. Then by a simple application of the cosine law (similar to the one described in the previous section) on the triangle with the points x_i, x_k and the center of the circle, we get

$$\begin{aligned}\|x_i - x_k\|^2 &= r_{ijk}^2 + r_{ijk}^2 - 2r_{ijk}r_{ijk}\cos 2\phi_{ijk} \\ \|x_i - x_k\|^2 &= 2r_{ijk}^2(1 - \cos 2\phi_{ijk}). \qquad (5.4)\end{aligned}$$

We can create a similar set of equations in the unknowns x_i, x_j, x_k and r_{ijk}^2 using the angles between all such sets of points. These sets of equations can be generated for all sets of points that have mutual angle information with each other. For each set of 3 points, a new unknown r_{ijk}^2 is introduced.

It can be observed that the constraints are quadratic in the matrix $X = [x_1 x_2 \dots x_n]$. By applying the SDP relaxation, *i.e.*, using the substitution $Y = X^T X$ and relaxing this constraint to $Y \succeq X^T X$, the problem is reduced once again to an SDP with one matrix linear inequality and some linear constraints, with the unknowns Y, X, r_{ijk}^2. Furthermore, we can also include the linear constraints introduced by angle measurements at the anchors as described in Equation (5.1).

It is pertinent to ask if the introduction of new unknowns r_{ijk}^2 will cause the problem to be under determined. However, given enough angle relations, there will be fewer unknowns than constraints and the network can be localized accurately. A brief analysis is presented to demonstrate that the number of constraints is larger than the number of unknowns and therefore a unique feasible solution must exist. The matrix Y has $n(n+1)/2$ unknowns and X has $2n$ unknowns. Suppose the number of triplets that have mutual angle information amongst each other is k. The angle measurements considered are exact, *i.e.*, they do not have any error. Then k more unknowns corresponding to the radii of the circumscribing circles are introduced. At the same time, we have constraints for each circle, so a total of $3k$ constraints exist. Hence as long as $3k > k + n(n+1)/2 + 2n$, the set of equations will have a unique feasible solution. The matrix $X^* = [\hat{x}_1 \hat{x}_2 \dots \hat{x}_n]$ and $Y^* = (X^*)^T X^*$, where $\hat{x}_i$ is the true position of the sensor i, will be the solution satisfying these equations. Therefore if k is large enough, the unique solution will exist. The maximum possible value of k is in fact $\binom{n}{3}$. If however, the connectivity is high, then the size of the triplet set k is also large enough for the set of equations to have a unique solution.

5.3 Practical considerations

5.3.1 Noisy measurements

In realistic scenarios, the angle information is not entirely accurate. In our model, we are assuming that all measurements are made relative to the axes of the sensors. These measurements will be noisy. The noise model is described in Section 5.4 where the simulations are presented. Softer SDP models that minimize the errors in the constraints can be developed, based on constraints discussed in the previous section by introducing slack variables. Note that our formulation allows flexibility in choosing any combination of exact or inexact distance and angle constraints depending on the information provided. For example, consider again a constraint of the type (5.4):

$$\|x_i - x_k\|^2 = 2r_{ijk}^2(1 - \cos 2\phi_{ijk}).$$

With noisy angle information, it can be modified to

$$\|x_i - x_k\|^2 = 2r_{ijk}^2(1 - \cos 2\phi_{ijk}) + \alpha_{ijk}, \tag{5.5}$$

where α_{ijk} is the error in the equation. We can now choose to minimize an objective function corresponding to the resulting SDP, the absolute sum of these errors. We can also choose alternative objective functions that penalize the errors in the equations differently. In our presented results, the above mentioned objective function is used. An interesting alternative objective is the sum of the alpha-squares, in which case the SDP formulation becomes a relaxation of a nonlinear least squares problem. The choice of a suitable objective function to minimize that gives the best performance for a particular noise model while being computationally tractable is a further research topic.

The SDP solution can also serve as a good starting point for local refinement through a local optimization method. This idea has been explored for distance based information in Section 3.6. Extending local refinement ideas for angle information is being pursued.

It is also interesting to observe that when there are no anchors used in the pure angle approach, the size of the network can be scaled by any constant α and the angle relations will still hold. But in the absence of any distance information, we do not know what the scaling constant should be. The solution will also be a rotated and translated version of the original network. We still obtain a relative map, which may still be of tremendous use in applications like routing. However, these problems will only present themselves when there are no anchors or distance information at all. Having well placed anchors in the network helps 'ground' the network by providing reference information about a global coordinate system. If enough anchor information is provided, we arrive at the exact position estimations without any rotation or translation.

In the case of noisy angle information, the placement of anchors assumes a critical role for both approaches, as was also the case when just distance information was used in Chapter 3. If the anchors are placed in the interior of the network, the estimations for points outside the convex hull of the anchor points also tend to be towards the interior of the network and are therefore quite erroneous. This problem does not present itself when the anchors are placed on the perimeter of the network. Once again, the addition of a regularization term in the objective function(to pull the points far apart and obtain a low dimensional realization) as described in Section 3.4 should help remove the sensitivity to anchor placement. and is a part of future research.

5.3.2 Computational costs

Because we consider triplets of points, the number of angle constraints of the type (5.2) and (5.4) increase substantially $(O(n^3))$ as the network size(n) and radio range(R) increase. It is very inefficient to include all the angle constraints if only a few of them can be used to obtain the solution. The problem becomes too strictly over-determined when in fact most of the constraints are redundant and the problem can be solved using only a subset of the constraints. For example, even for a network size of 40, the number of constraints for a radio range of 0.3 and omnidirectional angle measurements is about 1100. Clearly, the problem starts becoming intractable for even such small cases. Therefore, we propose an intelligent constraint generation methodology that can help in keeping the number of constraints selected small enough such that the problem is tractable.

In selecting the constraints, we aim to ensure that the constraints selected are well distributed in terms of the points that they correspond to, *i.e.*, there are not too many or too few constraints arising from angle relations for a single point in the network . This can be done by limiting the number of constraints that arise from angle relations for a particular point below a particular threshold. Furthermore, the constraints are between pairs of points. So the constraint selection should not concentrate only on angle constraints corresponding to a particular pair of points as seen from other points. While this can be done by keeping a counter of the number of constraints on every possible pair, clearly this strategy adds a lot of overhead to the actual processing. In order to minimize the cost of the constraint selection, we use a randomized constraint selection technique where a constraint is added to the problem with a certain probability. This probability can be made to depend upon the number of desired constraints as a fraction of the total number of possible constraints. The randomized selection ensures that the constraints take into account the global information of the network as opposed to smaller local clusters of points. The constraint selection scheme may even be enhanced by a subsequent active constraint generation methodology in which we solve the SDP once with a subset of constraints and then solve it again after adding in the constraints that were violated in the initial pass. It may however be expensive to check for all the violated constraints. We have not explored this strategy further in this thesis.

While the above mentioned methods work well for networks of up to 70-80 points, severe scalability issues will arise for very large networks of say, thousands of points. A distributed method for angle information, similar in spirit to that described in Section 3.7 for pure distance information, is a direction for future research. The entire network will be divided into clusters using anchor information and a smaller SDP is solved for each cluster. The cluster idea still takes advantage of the collective information between a set of nodes which are within a cluster. For points in clusters that are well separated from each other, there is little or no mutual information to begin with. So the problem can be solved in a parallel and 'decoupled' fashion in each of the separate clusters. The computations can be made iterative such that points that were well estimated can be used in subsequent iterations to provide estimates for points that were poorly estimated in previous iterations. Information about points that are on the boundary of 2 clusters can even be exchanged between clusters in order to assist in localizing other points within them.

5.4 Simulation results

Simulations were performed on a networks of 40 nodes and 4 anchors in a square region of size 1×1. Each node was given a random axis orientation between $[-\pi, \pi]$. The distances between the nodes was calculated. If the distance between 2 nodes i and j was less than a given radio range R, the angle between the axis of i and the direction pointing to node j was computed. If the angle was within field of view, specified by $maxang$, *i.e.*, the maximum angle on either side of the axis at which a node can see other nodes, the angle measurement was included. Otherwise, all other angle measurements were ignored. The angle measurement mechanism is assumed to be omnidirectional ($maxang = \pi$) in our simulations unless otherwise stated. The measured angles were modeled to be noisy by setting

$$\theta = \hat{\theta} + nfN(0,1),$$

where θ is the measured angle, $\hat{\theta}$ is the true value of the angle, nf is a specified constant, and $N(0,1)$ is a normally distributed random variable. Therefore the noise error in the angle measurements is modeled as additive and can be varied by changing nf. In order to find relative angles between 2 points as seen from a third point, the measured angles corresponding to the 2 points need to be subtracted. Therefore the angle data used in the problem formulation has higher noise variance than the measured angle data. The noisy distance measurements are modeled in the same way as in the previous chapters using a multiplicative noise model:

$$d = \hat{d}(1 + nfN(0,1)),$$

where d is the measured distance and $\hat{d}$ is the actual distance.

The average effect of varying radio range (R), field of view ($maxang$), noise (nf) and number of anchors on the estimation error was studied by performing 25 independent trials each of different configurations. Figure 5.3 shows the variation of average estimation error (normalized by the radio range R) when the noise in the angle measurements increases and with different radio ranges. The $maxang$ is set to π, *i.e.*, the angle measurement mechanism is assumed to be omnidirectional. Four anchors are placed near the corners of the square network. There are no more anchors placed in the interior. Comparing the combined distance-angle approach from Section 5.2.2 to the pure angle approach from Section 5.2.3 for smaller radio ranges, the error is much higher in the pure angle case for a particular setting of radio range (R) and measurement noise (specified by nf). This behavior is expected since the distance-angle approach employs more information to solve for the network as opposed to the pure angle approach. However, as the measurement noise increases, the error in the distance-angle case increases more rapidly. This is because both the distance and angle measurements are deteriorated by the noise measurements as opposed to just the angle measurements for the pure angle case. In fact, for higher radio ranges and high measurement noise, the pure angle approach starts outperforming the distance-angle approach. For example, for a radio range $R = 0.6$ and noise factor $nf = 0.25$, the pure angle case has an error of about 13% R and the distance-angle case of about 17% R.

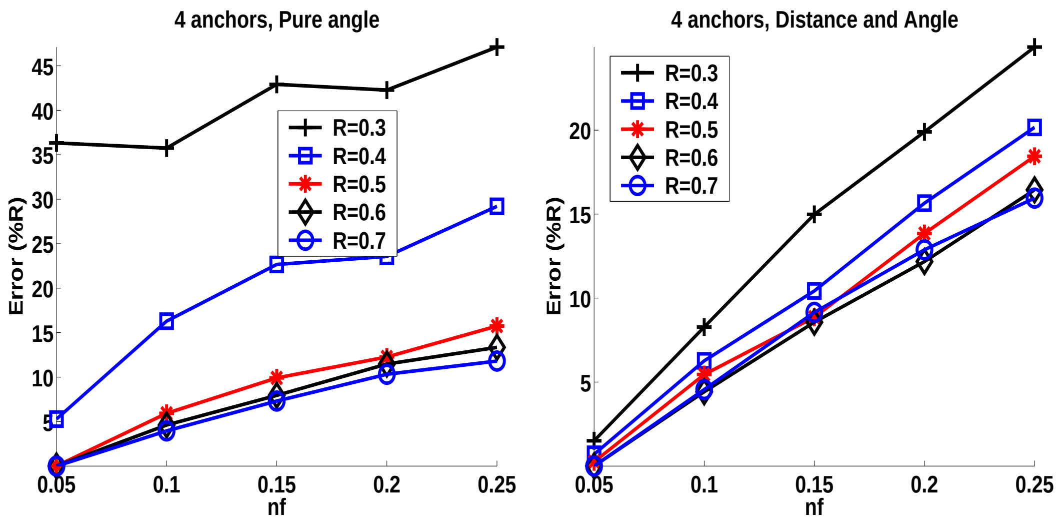


Figure 5.3: Variation of estimation error with measurement noise.

Another observation is that the advantage of having a higher radio range diminishes beyond a particular value. This can be explained by the fact that beyond a certain radio range, there is enough distance and angle information for the network to be solved. The extra information that is obtained for a higher radio range is redundant. We also disregard some of the redundant information by the random constraint sampling technique described in Section 5.3.2 in order to keep the number of constraints small enough so that the problem is tractable. In fact for the relaxation proposed, including the extra information may actually deteriorate the solution quality as well, for the distance-angle case, because with increasing radio range, the multiplicative noise model for distances that is used implies that the distance measures (corresponding to far away points) will have higher noise. This is demonstrated more effectively in Figure 5.4, where we consider average absolute estimation error. For the pure angle case, for a radio range of 0.5 and higher, the error stays more or less the same (about 7-12% R) for varying measurement noises. And for the distance-angle case, by increasing the radio range, while the error does go down in terms of the fraction of the radio range, the absolute error increases. Tuning the radio range such that optimum performance is achieved in terms of computational complexity and accuracy is desirable and it is interesting research question to analyze what such an optimum radio range is for localizing a network and subsequently deriving heuristics to find such an optimum.

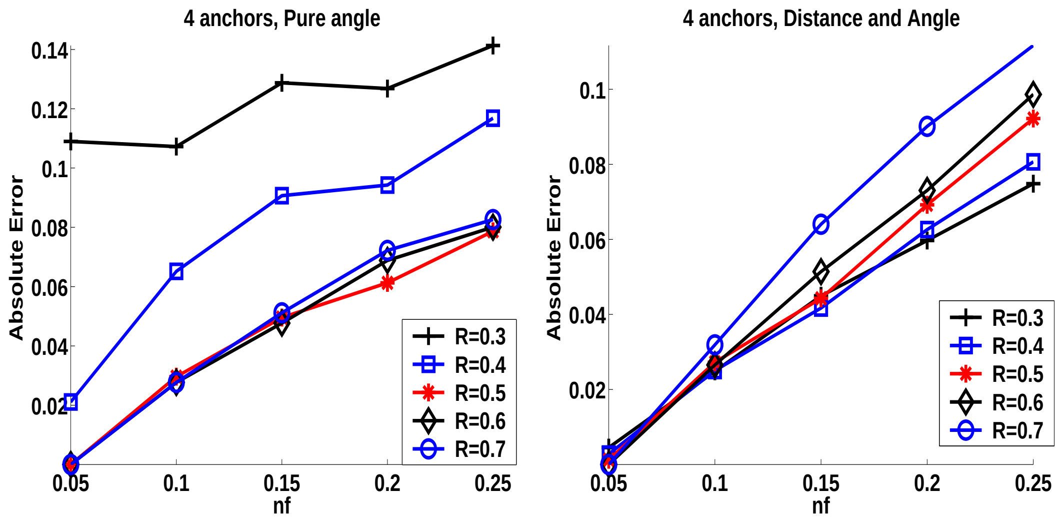


Figure 5.4: Variation of absolute estimation error with measurement noise.

Figure 5.5 shows the variation of estimation error when the radio range is increased and with different measurement noises. With lower radio range, the pure angle case appears to have high error (almost 35-45% R) but with the inclusion of more constraints by increasing the radio range, the error starts diminishing quite rapidly (5-15% R). In contrast, the distance-angle case has lower error with smaller radio ranges to begin with and the drop in error with increasing radio range is not that sharp (about 5% R). The wider spacings between the lines for different nf in the distance-angle case suggests as before that the distance-angle case is more sensitive to change in measurement noise.

Overall, the results suggest that when the radio range and the measurement noises are low, *i.e.*, we have few measurements but they are reliable, it is preferable to use the distance-angle approach. However, if there are lots of measurements but they have higher uncertainty, it is better to use the pure angle approach. An actual example of this behavior is presented in Figure 5.6. For Figures 5.6(a) and 5.6(b) corresponding to low radio range ($R = 0.35$) and low noise ($nf = 0.05$), while the pure angle approach hardly has enough information to solve the network and performs poorly, the distance-angle approach provides low error estimation for all the points. For the second case in Figures 5.6(c) and 5.6(d),

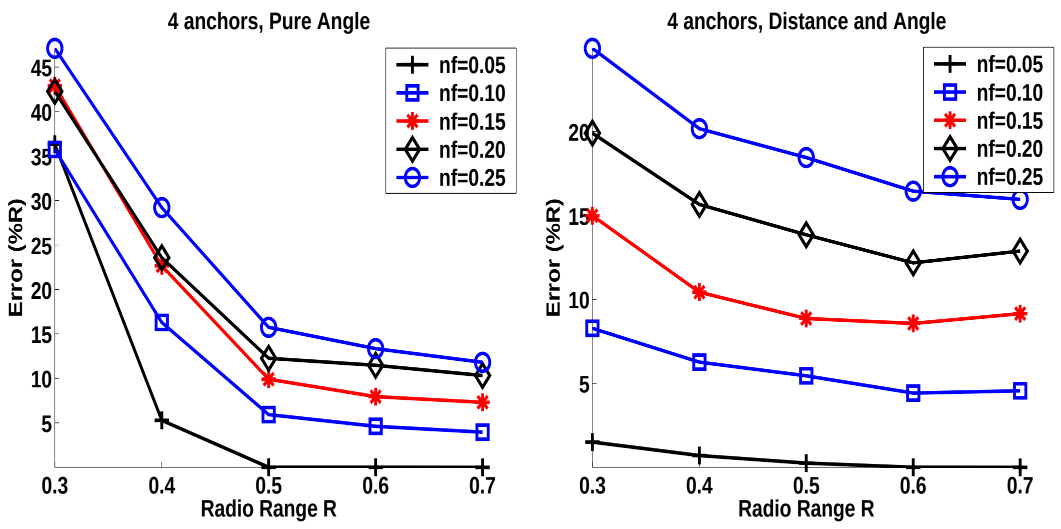
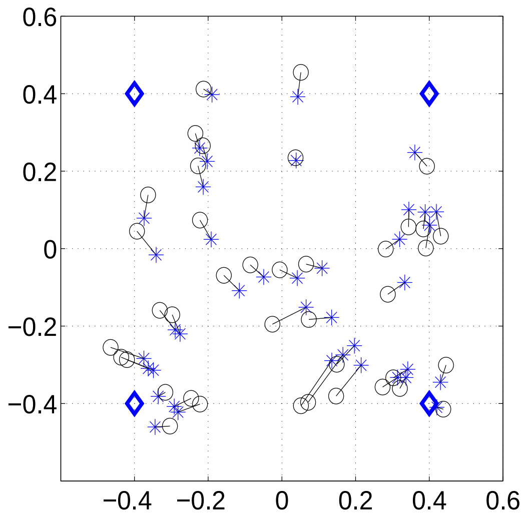


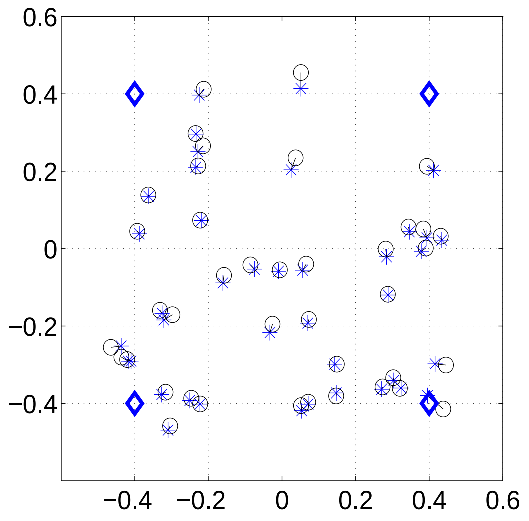
Figure 5.5: Variation of estimation error with radio range.

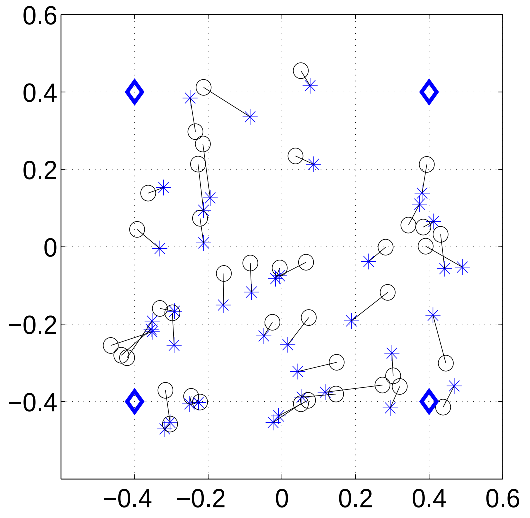
when the radio range is high ($R = 0.7$) and noise is high ($nf = 0.25$), the estimations for quite a few of the points are understandably erroneous owing to the high noise. But the pure angle approach seems to perform slightly better.

Figure 5.7 shows the variation of estimation error by increasing the radio range while varying the number of anchors from 4 to 8. Four anchors were placed at the perimeter and the rest in the interior. nf is fixed at 0.1. The effect of fewer anchors is felt more strongly for low radio ranges and starts becoming irrelevant with higher radio range for the pure angle case. Overall, the effect of more anchors is not extremely significant. For the distance-angle case, the improvement in accuracy by increasing the number of anchors is not very much (only about 1-2%)

The effect of a smaller field of view is presented in Figure 5.8 for varying radio range. For low radio range ($R = 0.3$) and small field of view ($maxang = \pi/4 - \pi/3$), the accuracy is quite poor for both approaches, greater than 100% R for pure angle case, and 50-80% R for distance-angle case. Since distance information is also used in the latter case, the accuracy is better. The lack of more angle information due to limited field of view clearly hurts the accuracy in the pure angle case. Even as the radio range increases, the effect of less angle information goes down, but less dramatically for the pure angle case as for the distance-angle case. It is still about 45% R when $R = 0.7$ for the pure angle case, whereas enough information is available in the distance-angle case for the error to almost vanish.


(a) Pure angle, low radio range and noise


(b) Distance and angle,low radio range and noise


(c) Pure angle, high radio range and noise

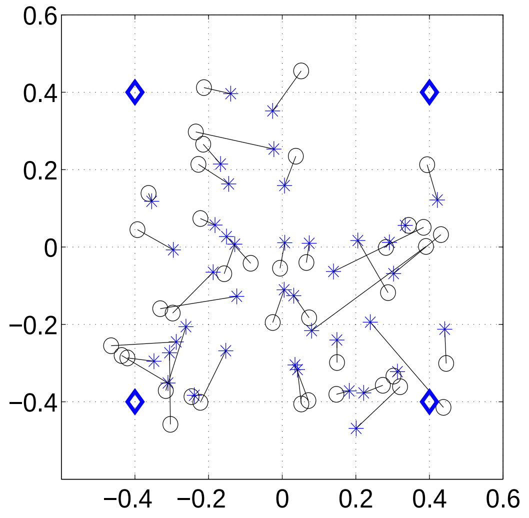

(d) Distance and angle, high radio range and noise

Figure 5.6: Comparisons in different scenarios.

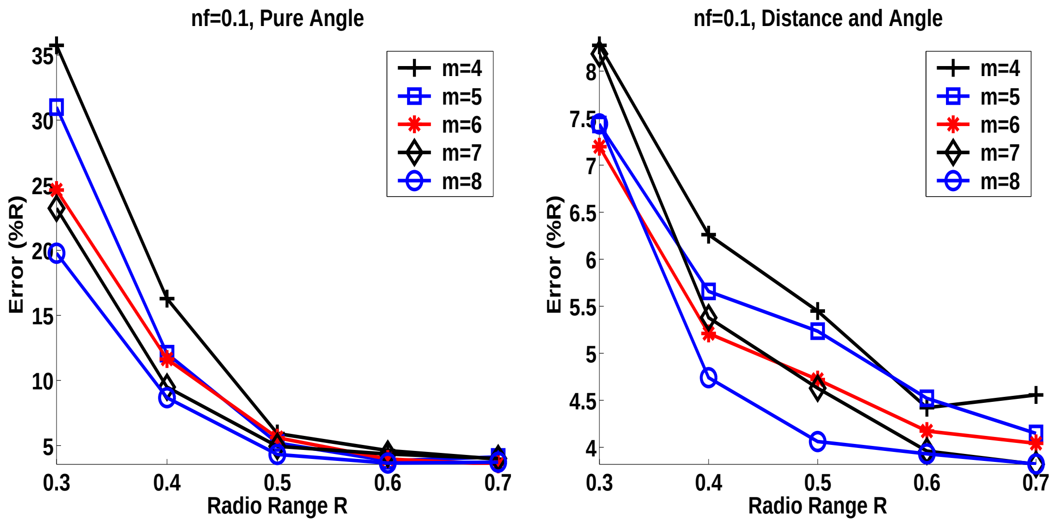


Figure 5.7: Variation of estimation error with more anchors.

The effect of the field of view is also illustrated by means of an actual example depicted in Figure 5.9. Even with measurements with no noise, some points are estimated very poorly when $maxang$ is set to $\pi/3$. Note that the axis orientations of the points are random. By limiting the field of view and having random axis orientations, it is hard to establish with certainty if enough information is available for all the points to be estimated correctly, especially for the pure angle case. Even when points are within radio range of each other, they are not able to detect each other if they are not in each other's field of view. As a result, there is very little information for some of the points and they are estimated badly. With higher radio range, this problem can be reduced but the performance is very sensitive because of the random axis orientations. This also suggests that the random constraint selection may need to be fine-tuned in order to include more constraints for the points that appear to have very little information to begin with.

The number of constraints and solution time was also analyzed. Our simulation program is implemented in MATLAB and it uses SEDUMI [82] as the SDP solver. The simulations were performed on a Pentium IV 2.0 GHz and 512 MB RAM PC. Figures 5.10(a) and 5.10(b) illustrate how the solution time increases as the number of points in the network increases. Even with random constraint selection, the solution time seems to increase at a superlinear rate with the number of points. This indicates the necessity of developing a scalable distributed method that was mentioned in Section 5.3.2. It is interesting that due

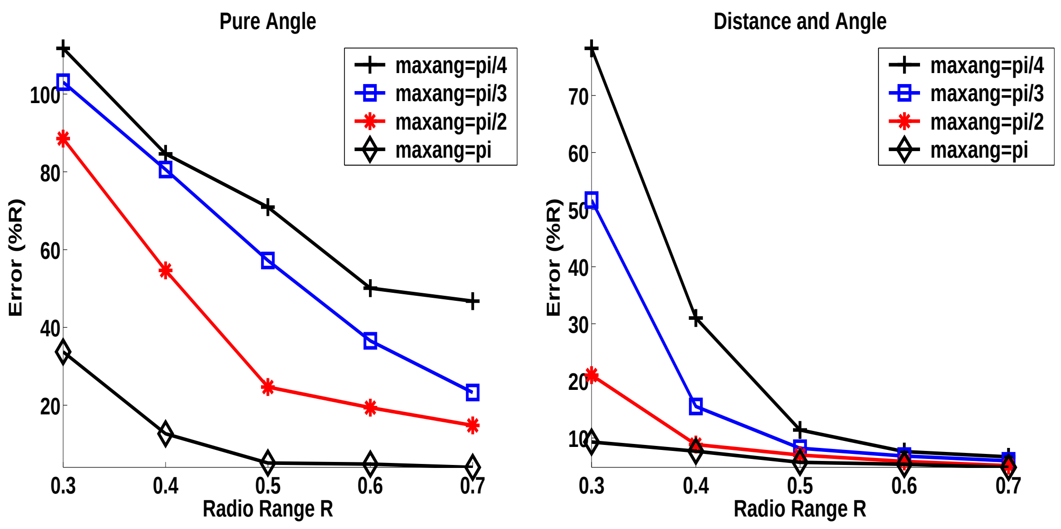


Figure 5.8: Effect of field of view: 40 node network, 4 anchors, 10% noise.

to the random constraint selection strategy, the running time is almost independent of the radio range. For example, for a network of 40 points, the number of constraints are usually around 480-520. This is illustrated in Figure 5.10(c) that shows the change in solution times with change in radio range for a fixed network size of 40 points and 4 anchors. The number of constraints does not change substantially with increasing radio range and therefore the solution time does not change too much either. Furthermore, the pure angle approach uses more constraints and usually takes longer to solve.

5.5 Conclusions

In this chapter, the use of SDP relaxations for solving the distance geometry problem, pertaining to sensor network localization using angle of arrival information, was described. Formulations that use both distance and angle information as well as just angle information were discussed. A random constraint selection method was also described in order to reduce the computational effort. Experimental results show that the method can provide accurate localization. The distance-angle approach performs well in low radio range and low noise scenarios and the pure angle approach performs better when the radio range and measurement noise is high. The performance becomes more unpredictable with limited field of view. In order to rectify this, the random constraint selection method needs to be refined

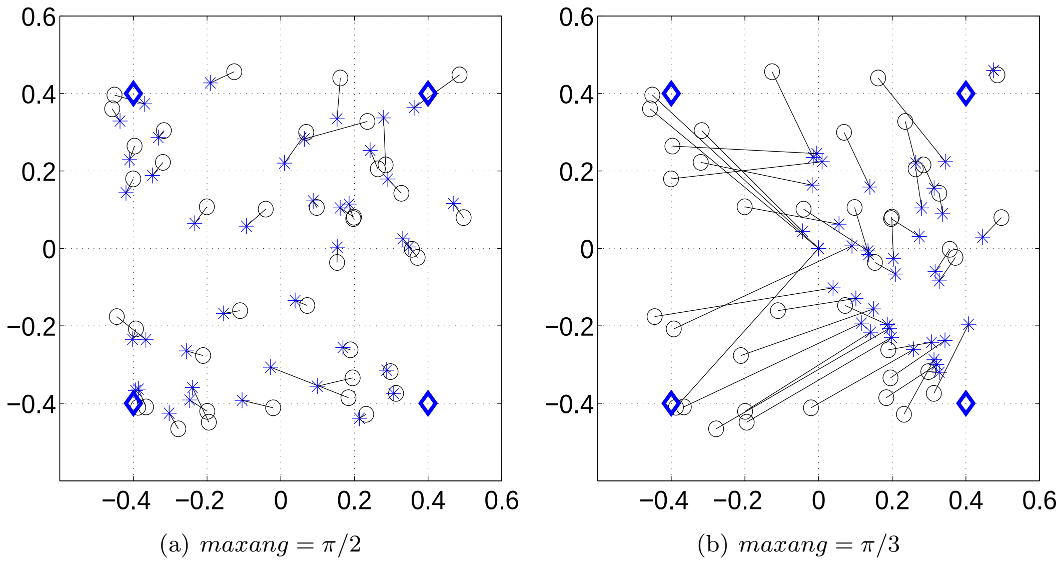
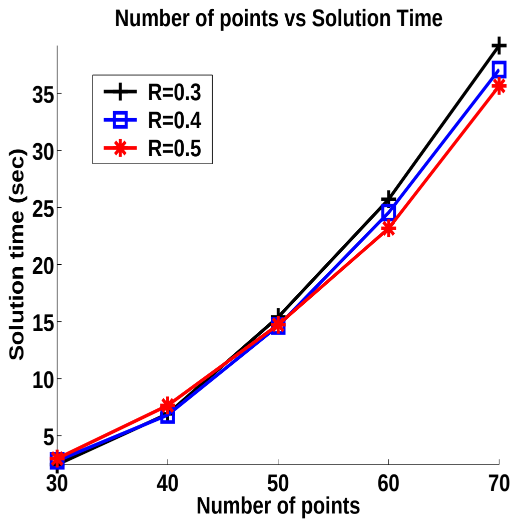


Figure 5.9: Example of effect of field of view on pure angle approach: 40 node network, 4 anchors, $R = 0.5$, no noise.

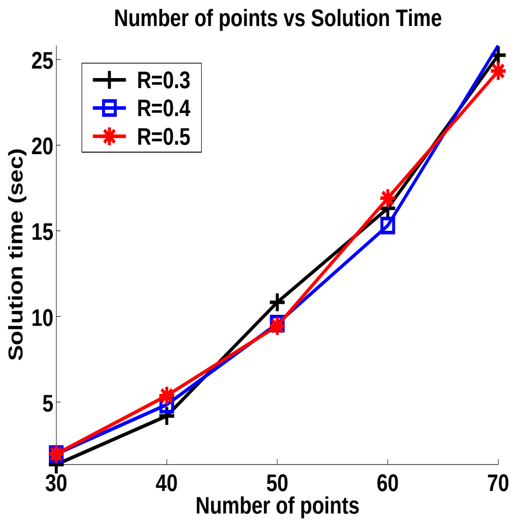
in order to include more constraints for points with very little information.

Due to the random constraint selection, the solution time is more or less independent of radio range. However, the number of constraints and solution time grows substantially as the number of points in the network increases. A distributed method that solves the localization problem as a set of decomposed clusters is being developed in order to make this method tractable for large networks.

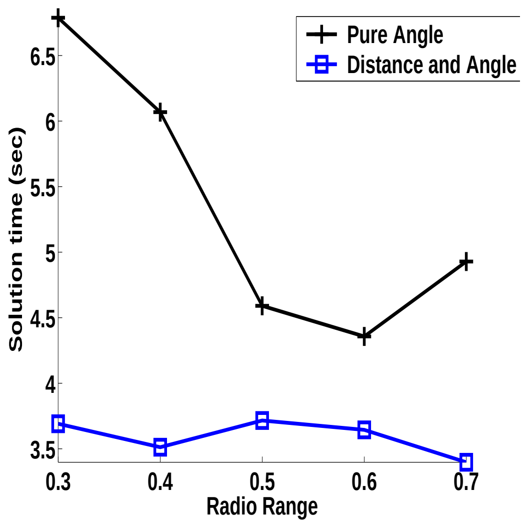
For noisy measurements, more accurate estimation would be provided if the anchor nodes in the network are placed optimally, or a regularization term is used, to ensure low dimensional solutions. These strategies need more thorough investigation. Furthermore, for dealing with high noise cases, local gradient based local optimization methods similar to those described for pure distance information in Section 3.6 must be developed.



(a) Pure Angle



(b) Distance and Angle



(c) Solution Time vs. Radio Range, 40 node network

Figure 5.10: Solution times.

Chapter 6

Conclusions and future research

In this chapter, the contributions of this thesis are restated, and broad research directions discussed. The future research ideas are also discussed in more detail in the corresponding chapters.

6.1 Theoretical aspects

This thesis introduces an SDP relaxation based technique to solve the Euclidean distance geometry problem. By using the global distance information about a graph, we are able to simultaneously estimate the positions of all its vertices accurately. While aspects such as the exactness, complexity and sensitivity of this relaxation have been studied, there are still many possibilities for very interesting theoretical work in this domain.

In particular, tensegrity theory has opened up avenues to further improve the SDP relaxation to achieve realizability of graphs in lower dimensions. Also the analysis of the algorithm under noise can be developed further, such as getting strong bounds on estimation error of all the points, and more importantly, reliable error measures to detect badly estimated points. The key to resolving many of these issues lies in a closer inspection of the dual problem of the SDP relaxation. The analysis of the dual may even uncover newer relaxations that are computationally more efficient and provide accurate low dimensional embeddings. Indications that such an approach is promising can be found in [94], where the graph Laplacian is used to obtain a much lower dimensional SDP formulation. Efforts are also being made to understand the complementarity between the dense distance matrix based methods and sparse Laplacian based methods for dimensionality reduction [98].

There are still significant gaps in understanding the interplay between these approaches and a deeper investigation can yield many theoretical and practical insights not only in distance geometry but in spectral graph theory as well.

Another important theoretical consideration is to characterize the type of ambiguities that exist in the point positions if the solution is not unique. By determining if the ambiguities are discontinuous and discrete, we may be able to significantly narrow down the number of different configurations that the non-unique solution can achieve. The SDP objective function is crucial in deciding which of the non-unique solutions the SDP will converge to. Developing a parameterized objective function for the SDP could allow for such an analysis to be carried out further.

There is also scope to move to tighter relaxations than the SDP. The use of Sum of Squares techniques for polynomial global optimization has already been applied in the sensor localization context in the paper [63] and is often able to deliver solutions that are very close to the global optimal solution of the non-convex problem.

6.2 Sensor network localization

The application of the SDP relaxation to sensor network localization with noisy distance measurements has been demonstrated effectively and has received significant attention from other researchers in the sensor network community. The use of global distance information and the combined use of the regularization term in the SDP and local refinement provide very accurate position estimates for networks with even very noisy distance measurements.

Studying newer formulations with penalty functions that provide the best statistical estimates given a particular distance noise model also need to be investigated further. The algorithm must also be tested using actual measured data from real sensor networks. In such scenarios, the noise models could be different, such as additive, multiplicative or log-normal. In each case, different penalty functions provide the maximum likelihood estimate, and tailoring the objective function of the SDP accordingly could provide significantly better estimations.

The presence of anchors can greatly improve the estimation accuracy. However, the infrastructure cost involved in setting up too many anchors can be overwhelming. As a result, intelligent heuristics or algorithms for effective placement of a limited number of anchors, are worth looking into. A deeper analysis of balance between the regularization and error minimization terms in the noisy distances case is also important. In particular, if we are able to correctly balance the two terms based on just the given network size, anchor positions and incomplete distance matrix, the estimations can be much more accurate. Closer investigation of tensegrity theory would also allow the development of better regularization terms in order to get low dimensional solutions even with very noisy distance measurements and limited anchors.

A distributed iterative algorithm based on the SDP relaxation was also described, to counter the intractability faced in large scale sensor networks. This provides the best opportunity in terms of an actual deployment of such a localization algorithm in a real sensor network. But in doing so, more research needs to be done in deciding the cluster sizes, and in developing a protocol to minimize the cluster setup and communication costs.

6.3 Molecule structure determination

The large scale distributed implementation of the SDP relaxation technique was also demonstrated for molecules with thousands of atoms. We proposed intelligent clustering and stitching algorithms for solving the kind of large scale problems that are encountered in this space. Using these ideas, we were able to solve for molecules of sizes of thousands of atoms.

The algorithm still needs improvement for a combination of very noisy and sparse distance data. In this regard, there is particular scope for improvement in finding reliable error measures for discarding points, and for more robust cluster stitching algorithms. The failure cases of the algorithm indicate that it is in these 2 steps that the error goes out of control, and if there are more reliable techniques for performing them, we can solve for even larger molecules with irregular geometries and with much looser distance constraints. The size of the problems considered also highlights the need for faster and more scalable SDP solvers. In fact, significant improvements are noticed even when more sophisticated local refinement methods such as truncated or inexact Newton methods are used instead of a naive gradient descent algorithm.

The algorithm also needs to be tested with actual NMR data and other data, such as the VDW constraints. Hopefully, proteins whose structures have so far not been determined can be conformed using this algorithm.

6.4 Extensions to angle information

The basic distance geometry model was also extended to handle angle information in $\mathcal{R}^2$. The efficacy of these techniques was demonstrated by simulations on sensor networks (image based sensor networks, for example) that use a mixture of distance and angle information, or pure angle information for localization. However, this approach made more computationally efficient and robust to noise and sparsity. A randomized constraint selection method was used to control the problem size for large networks. However, it is currently still a strongly heuristic method. Better constraint selection would help in improving the accuracy and computational performance, especially when the angle measurement mechanism is very noisy, and the sensors have a very limited field of view. The addition of regularization terms, the development of a distributed method and a local refinement method, much like the ones described for pure distance information in Sections 3.5, 3.7, 3.6 respectively are also required. Deeper theoretical analysis of the uniqueness and exactness of the SDP solution (as was done for distances in Section 2.3), and sensitivity to noise and limited fields of view, could provide insights into how the above mentioned practical issues may be best dealt with.

NMR measurements are also able to measure a few orientation constraints in the form of dihedral angles between the planes defined by 2 sets of 3 points each. Such information may further constrain the solution space, which may be very useful in the case of sparse distance information. Preliminary extensions to account for dihedral angle constraints in molecule structure determination have been developed [28], but they suffer from overwhelming computational costs (due to larger number of variables and far too many constraints). Intelligent schemes that incorporate only the required dihedral angle constraints, while maintaining a manageable problem size need to be developed.

Bibliography

[1] A. Y. Alfakih, A. Khandani, and H. Wolkowicz. Solving Euclidean distance matrix completion problems via semidefinite programming. *Comput. Optim. Appl.*, 12(1-3):13–30, 1999.

[2] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.

[3] G. W. Allen, K. Lorincz, M. Welsh, O. Marcillo, J.Johnson, M. Ruiz, and J. Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10(2):18–25, 2006.

[4] J. Aspnes, D. Goldenberg, and Y. Yang. The computational complexity of sensor network localization, in *p*roceedings of the first international workshop on algorithmic aspects of wireless sensor networks, 2004, 2004.

[5] A. I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13:189–202, 1995.

[6] A. Basu, J. Gao, J. S. B. Mitchell, and G. Sabhnani. Distributed localization using noisy distance and angle information. In *MobiHoc '06: Proceedings of the Seventh ACM International Symposium on Mobile Ad-hoc Networking and Computing*, pages 262–273, New York, NY, USA, 2006. ACM Press.

[7] S. J. Benson, Y. Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM Journal on Optimization*, 10(2):443–461, 2000.

[8] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein data bank. *Nucleic Acids Research*, 28:235–242, 2000.

[9] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13(2), April 1997.

[10] P. Biswas, T. C. Liang, K. C. Toh, T. C. Wang, and Yinyu Ye. Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineeering, Special Issue on Distributed Sensing*, 3(4):360–371, October 2006.

[11] P. Biswas, T. C. Liang, T. C. Wang, and Y. Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks*, 2(2):188–220, 2006.

[12] P. Biswas, K. C. Toh, and Y. Ye. A distributed SDP approach for large-scale noisy anchor-free 3d graph realization (with applications to molecular conformation). Technical report, Dept of Management Science and Engineering, Stanford University, *submitted to SIAM Journal on Scientific Computing*, February 2007.

[13] P. Biswas and Y. Ye. A distributed method for solving semidefinite programs arising from ad hoc wireless sensor network localization. Technical report, Dept of Management Science and Engineering, Stanford University, *to appear in Multiscale Optimization Methods and Applications*, October 2003.

[14] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *Proceedings of the Third International Symposium on Information Processing in Sensor Networks*, pages 46–54. ACM Press, 2004.

[15] S. Boyd, L. E. Ghaoui, E. Feron, and Venkataramanan Balakrishnan. *Linear Matrix Inequalities in System and Control Theory.* SIAM., 1994.

[16] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, March 2004.

[17] S. E. Brenner. A tour of structural genomics. *Nature Reviews Genetics*, 2(10):801–809, October 2001.

[18] J. Bruck, J. Gao, and A. Jiang. Localization and routing in sensor networks by local angle information. In *MobiHoc '05: Proceedings of the Sixth ACM International*

Symposium on Mobile Ad-hoc Networking and Computing, pages 181–192, New York, NY, USA, 2005. ACM Press.

[19] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low cost outdoor localization for very small devices. Technical report, Computer Science Department, University of Southern California, April 2000.

[20] S. Burer and R. D. C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

[21] M. Carter, H. H. Jin, M. A. Saunders, and Y. Ye. Spaseloc: An adaptable subproblem algorithm for scalable wireless network localization. *Submitted to the SIAM J. on Optimization*, January 2005.

[22] B. Chazelle, C. Kingsford, and M. Singh. The side-chain positioning problem: a semidefinite programming formulation with new rounding schemes. In *PCK50: Proceedings of the Paris C. Kanellakis memorial workshop on Principles of computing & knowledge*, pages 86–94. ACM Press, 2003.

[23] K. Chintalapudi, J. Paek, O. Gnawali, T. S. Fu, K. Dantu, J. Caffrey, R. Govindan, E. Johnson, and S. Masri. Structural damage detection and localization using NETSHM. In *IPSN '06: Proceedings of the Fifth International Conference on Information Processing in Sensor Networks*, pages 475–482, New York, NY, USA, 2006. ACM Press.

[24] F. Chung, M. Garrett, R. Graham, and D. Shallcross. Distance realization problems with applications to internet tomography. *Journal of Computer and System Sciences*, 63:432–448(17), November 2001.

[25] J. A. Costa, N. Patwari, and III A. O. Hero. Distributed weighted-multidimensional scaling for node localization in sensor networks. *ACM Transactions on Sensor Networks*, 2(1):39–64, 2006.

[26] T. Cox and M. A. A. Cox. *Multidimensional Scaling.* Chapman Hall/CRC, London., 2001.

[27] G. Crippen and T. Havel. *Distance Geometry and Molecular Conformation.* Wiley, 1988.

[28] J. Dattorro. *Convex Optimization and Euclidean Distance Geometry.* Meboo Publishing, 2006.

[29] L. Doherty, L. E. Ghaoui, and K. S. J. Pister. Convex position estimation in wireless sensor networks. In *Proceedings of IEEE Infocom*, pages 1655 –1663, Anchorage, Alaska, April 2001.

[30] Q. Dong and Z. Wu. A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. *Journal of Global Optimization*, 26(3):321–333, 2003.

[31] T. Eren, D. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. D. O. Anderson, and P. N. Belhumeur. Rigidity, computation, and randomization in network localization. In *Proceedings of IEEE Infocom*, pages 1655 –1663, 2004.

[32] R. J. Fontana, E. Richley, and J. Barney. Commercialization of an ultra wideband precision asset location system. In *Proceedings of 2003 IEEE Conference on Ultra Wideband Systems and Technologies*, pages 369–373, 2003.

[33] A. George and J. W. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall Professional Technical Reference, 1981.

[34] D. Gleich, L. Zhukov, M. Rasmussen, and K. Lang. The world of music: SDP layout of high dimensional data, in *InfoVis 2005.*

[35] N. Guex and M. C. Peitsch. Swiss-model and the Swiss-Pdbviewer: An environment for comparative protein modeling. *Electrophoresis*, 18:2714–2723, 1997.

[36] O. Güler and Y. Ye. Convergence behavior of interior point algorithms. *Mathematical Programming*, 60:215–228, 1993.

[37] T. F. Havel and K. Wüthrich. An evaluation of the combined use of nuclear magnetic resonance and distance geometry for the determination of protein conformation in solution. *Journal of Molecular Biology*, 182:281–294, 1985.

[38] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh. Energy-efficient surveillance system using wireless sensor networks. In *MobiSys '04: Proceedings of the 2nd International Conference on Mobile Systems, Applications, and Services*, pages 270–283, New York, NY, USA, 2004. ACM Press.

[39] B. A. Hendrickson. *The molecular problem: Determining Conformation from pairwise distances.* PhD thesis, Cornell University, 1991.

[40] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, 34(8):57–66, 2001.

[41] A. Howard, M. Matarić, and G. Sukhatme. Relaxation on a mesh: a formalism for generalized localization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1055–1060, Wailea, Hawaii, Oct 2001.

[42] G. Iyengar, D. Phillips, and C. Stein. Approximation algorithms for semidefinite packing problems with applications to Maxcut and graph coloring. In *Eleventh Conference on Integer Programming and Combinatorial Optimization*, Berlin, 2005.

[43] B. Jackson and T. Jordan. Connected rigidity matroids and unique realizations of graphs, 2003.

[44] H. H. Jin. *Scalable Sensor Localization Algorithms for Wireless Sensor Networks.* PhD thesis, Department of Mechanical and Industrial Engineering, University of Toronto, 2005.

[45] S. Joshi and S. Boyd. An efficient method for large-scale gate sizing. Technical report, Dept. of Electrical Engineering, Stanford University *submitted to IEEE Transactions on Circuits and Systems*, 2006.

[46] E. Kaplan. *Understanding GPS: Principles and Applications.* Artech House, 1996.

[47] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* SIAM, 1995.

[48] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky. A method for large-scale l_1-regularized least squares problems with applications in signal processing and statistics. Technical report, Dept. of Electrical Engineering, Stanford University, 2007.

[49] R. Kimmel. *Numerical Geometry of Images: Theory, Algorithms, and Applications.* SpringerVerlag, 2003.

[50] K. Koh, S. J. Kim, and S. Boyd. An interior-point method for large-scale l_1-regularized logistic regression. Technical report, Dept. of Electrical Engineering, Stanford University *To appear, Journal of Machine Learning Research, 2007.*

[51] L. Krishnamurthy, R. Adler, P. Buonadonna, J. Chhabra, M. Flanigan, N. Kushalnagar, L. Nachman, and M. Yarvis. Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the north sea. In *SenSys '05: Proceedings of the Third International Conference on Embedded Networked Sensor Systems*, pages 64–75, New York, NY, USA, 2005. ACM Press.

[52] M. Laurent. Matrix completion problems. *The Encyclopedia of Optimization*, 3:221–229, 2001.

[53] T. C. Liang, T. C. Wang, and Y. Ye. A gradient search method to round the SDP relaxation solution for ad hoc wireless sensor network localization. Technical report, Dept. of Management Science and Engineering, Stanford University, August 2004.

[54] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.

[55] K. Lorincz, D. J. Malan, T. R. F. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. Sensor networks for emergency response: Challenges and opportunities. *IEEE Pervasive Computing*, 3(4):16–23, 2004.

[56] D.G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley, Reading, 1986.

[57] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson. Wireless sensor networks for habitat monitoring. In *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA'02)*, Atlanta, GA, September 2002.

[58] J. Moré and Z. Wu. ϵ-optimal solutions to distance geometry problems via global continuation. Preprint MCS–P520–0595, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., May 1994.

[59] J. Moré and Z. Wu. Global continuation for distance geometry problems. *SIAM Journal on Optimization.*, 7:814–836, 1997.

[60] D. Niculescu. Positioning in ad hoc sensor networks. *IEEE Network Magazine*, 2004.

[61] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *GLOBECOM (1)*, pages 2926–2931, 2001.

[62] D. Niculescu and B. Nath. Ad hoc positioning system (APS) using AoA. In *INFOCOM*, San Francisco, CA., 2003.

[63] J. Nie. Sum of squares method for sensor network localization, 2006.

[64] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, 1999.

[65] P. M. Pardalos and X. Liu. A tabu based pattern search method for the distance geometry problem. In *New trends in mathematical programming*, pages 223–234. Kluwer Academic Publishers, Boston, MA, 1998.

[66] N. Patwari, III A. O. Hero, and A. Pacholski. Manifold learning visualization of network traffic data. In *MineNet '05: Proceedings of the 2005 ACM SIGCOMM Workshop on Mining Network Data*, pages 191–196, New York, NY, USA, 2005. ACM Press.

[67] N. B. Priyantha, A. K. L. Miu, H. Balakrishnan, and S. Teller. The cricket compass for context-aware mobile applications. In *Mobile Computing and Networking*, pages 1–14, 2001.

[68] R. Reams, G. Chatham, W. K. Glunt, D. McDonald, and T. L. Hayden. Determining protein structure using the distance geometry program APA. *Computers & Chemistry*, 23(2):153–163, 1999.

[69] C. Savarese, J. M. Rabaey, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pages 317–327. USENIX Association, 2002.

[70] A. Savvides, C. C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Mobile Computing and Networking*, pages 166–179, 2001.

[71] A. Savvides, H. Park, and M. B. Srivastava. The bits and flops of the n-hop multilateration primitive for node localization problems. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, pages 112–121. ACM Press, 2002.

[72] A. Savvides, M. Srivastava, L. Girod, and D. Estrin. Localization in sensor networks. *Wireless Sensor Networks*, pages 327–349, 2004.

[73] J. B. Saxe. Embeddability of weighted graphs in k-space is strongly np-hard. In *Proceedings of the 17th Allerton Conference on Communication, Control, and Computing*, pages 480–489, 1979.

[74] I. J. Schoenberg. Remarks to m. fr'echet's article 'sur la d'efinition axiomatique d'une classe d'espaces vectoriels distanci'es applicables vectoriellement sur l'espace de hilbert'. *Annals of Mathematics*, 36:724–732, 1935.

[75] Y. Shang and W. Ruml. Improved MDS-based localization. In *Proceedings of the 23rd Conference of the IEEE Communicatons Society (Infocom 2004)*, 2004.

[76] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz. Localization from connectivity in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(11):961–974, 2004.

[77] Y. Shang, W. Ruml, Y. Zhang, and M. P.J. Fromherz. Localization from mere connectivity. In *Proceedings of the Fourth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pages 201–212. ACM Press, 2003.

[78] P. Skraba, L. Savidge, and H. Aghajan. Decentralized node location for wireless image sensor networks. Technical report, Wireless Sensor Networks Lab, Stanford University.

[79] A. M. C. So and Y. Ye. Theory of semidefinite programming for sensor network localization. In *SODA '05: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 405–414, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[80] A. M. C. So and Y. Ye. A semidefinite programming approach to tensegrity theory and realizability of graphs. In *SODA '06: Proceedings of the Aeventeenth Annual*

ACM-SIAM Symposium on Discrete Algorithm, pages 766–775, New York, NY, USA, 2006. ACM Press.

[81] S. Stillman and I. Essa. Towards reliable multimodal sensing in aware environments, 2001.

[82] J. F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11 & 12:625–633, 1999.

[83] K. C. Toh. Solving large scale semidefinite programs via an iterative solver on the augmented systems. *SIAM J. on Optimization*, 14(3):670–698, 2003.

[84] K. C. Toh, M. J. Todd, and R.H. Tütüncü. SDPT3 — a Matlab software package for semidefinite programming. *Optimization Methods and Software*, 11:545–581, 1999.

[85] W. S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17:401–409, 1952.

[86] M. W. Trosset. Applications of multidimensional scaling to molecular conformation. *Computing Science and Statistics*, 29:148–152, 1998.

[87] M. W. Trosset. Distance matrix completion by numerical optimization. *Comput. Optim. Appl.*, 17(1):11–22, 2000.

[88] M. W. Trosset. Extensions of classical multidimensional scaling via variable reduction. *Computational Statistics*, 17(2):147–162, 2002.

[89] P. Tseng. Second-order cone programming relaxation of sensor network localization, textitsubmitted to SIAM Journal of Optimization, August 2005.

[90] R.H. Tütüncü, K. C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical Programming, Series B*, 95:189–217, 2003.

[91] D. Vitkup, E. Melamud, J. Moult, and C. Sander. Completeness in structural genomics. *Nature Structural Biology*, 8(6):559–566, June 2001.

[92] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *Int. J. Comput. Vision*, 70(1):77–90, 2006.

[93] K. Q. Weinberger, F. Sha, and L. K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *ICML '04: Proceedings of the Twenty-first International Conference on Machine Learning*, page 106, New York, NY, USA, 2004. ACM Press.

[94] K. Q. Weinberger, F. Sha, Q. Zhu, and L. Saul. Graph Laplacian methods for large-scale semidefinite programming, with an application to sensor localization. In B. Schölkopf, J. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19*. MIT Press, Cambridge, MA, 2007.

[95] G. A. Williams, J. M. Dugan, and R. B. Altman. Constrained global optimization for estimating molecular structure from atomic distances. *Journal of Computational Biology*, 8(5):523–547, 2001.

[96] D. Wu and Z. Wu. An updated geometric build-up algorithm for molecular distance geometry problems with sparse distance data., submitted 2003.

[97] K. Wüthrich. *NMR of Proteins and Nucleic Acids.* John Wiley & Sons, New York, 1986.

[98] L. Xiao, J. Sun, and S. Boyd. A duality view of spectral methods for dimensionality reduction. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 1041–1048, 2006.

[99] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A wireless sensor network for structural monitoring, 2004.

[100] G. Xue and Y. Ye. An efficient algorithm for minimizing a sum of euclidean norms with applications. *SIAM Journal on Optimization*, 7(4):1017–1036, 1997.

[101] Y. Ye and J. Zhang. An improved algorithm for approximating the radii of point sets. In *RANDOM-APPROX*, pages 178–187, 2003.

[102] J. M. Yoon, Y. Gad, and Z. Wu. Mathematical modeling of protein structure using distance geometry. Technical report, Department of Computational & Applied Mathematics, Rice University, 2000.

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

[104] L. Yuan, C. Gui, C. Chuah, and P. Mohapatra. Applications and design of heterogeneous and/or broadband sensor networks. In *Broadnets*, San Jose, CA, 2004.

[105] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2005.