ABSTRACT

Title of Dissertation: DISTRIBUTED SOURCE LOCALIZATION
AND TRACKING ALGORITHMS FOR
AD-HOC ACOUSTIC SENSOR NETWORKS

Tien Pham, Doctor of Philosophy, 2006

Dissertation directed by: Assistant Professor Haralabos C. Papadopoulos
Department of Electrical and Computer Engineering

In this dissertation, we construct an algorithmic framework for systematic tracking of moving sources in large-scale sensor networks. The tracking algorithms we developed generate the estimates of the tracking locations from fusion of space-time data by first fusing the data in space and subsequently by fusing the data in time. Fusion in space is performed by fusing current sensed data that is sufficiently high-quality from the sensor nodes to produce the current source location estimate. These location estimates are indexed as they become available and subsequently fused iteratively in time to produce tracking estimates. Both fusion in space and fusion in time are performed distributively over the ad-hoc sensor network by exploiting distributed algorithms of computation of averages. The distributed tracking algorithms are locally constructed at each participating sensor node exploiting only locally available sensor observations and local available network connectivity information. These algorithms we developed are also resource efficient, scalable, fault-tolerant and can readily adapt to local changes in network topologies. We present methods for optimizing and characterizing the performance of the algorithms as a function of the quality of the sensor measurements, the source dynamics, the sensor density and the network connectivity.

DISTRIBUTED SOURCE LOCALIZATION AND TRACKING ALGORITHMS FOR AD-HOC ACOUSTIC SENSOR NETWORKS

by

Tien Pham

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2006

Advisory Committee:

Professor Haralabos C. Papadopoulos, Chair
Professor Shihab Shamma
Professor Sennur Ulukus
Professor P.S. Krishnaprasad
Professor John Benedetto

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my sincere appreciation and gratitude to the all people who have given me the inspiration, motivation and support that guided me through my PhD journey that began in 1998. Thanks to their constant support and encouragement, I was able to complete this enlightening and rewarding journey. Since it would be impossible to thank everyone, I would therefore acknowledge those without whom this PhD research could never have been accomplished.

First, with deep gratitude, I would like to thank my thesis advisor, mentor, and friend, Dr. Babis Papadopoulos. Babis, you are truly a great mentor, and I am grateful for your inspiration, enthusiasm, keen technical insights, and unwavering support that you have provided to me as a student over the last five years. The confidence and dynamism with which you guided the research work is truly inspiring. I also appreciate and value our friendship. Many times, our weekly meetings would start and end with discussions about our families, politics and/or current events. I do treasure those moments because they put the seemingly important research work in proper context of daily life. I hope that our friendship will continue and grow.

I would like to acknowledge and thank my Committee members Prof. S. Shamma, Prof. P.S. Krishnaprasad, Prof. S. Ulukus and Prof. J. Benedetto.

I would like to thank my research colleagues, office mates and friends at the ECE Department at the University of Maryland. Mohamed Abdallah, Dzul Scherber, and Yongsun Hwang, you have made me feel welcome and part of a special research group. Your exuberant energy made me feel younger than my age and your dedication to your individual research inspired me to do the same. I have learned to understand and appreciate our different cultural backgrounds and our common scientific interests. I'm glad I was able to offer advices and share my family and work experiences with all

of you. Thank you Rannia, Renga, Renju and Prasanth for being great officemates, sharing your tasty food and treats, and lively conversations. To all my fellow students, I will always treasure your company and I wish you the very best in your future endeavors.

Next, I would like to sincerely thank the US Army Research Laboratory for sponsoring my research work and supporting my graduate studies at the University of Maryland. I'm grateful to my supervisors, John Eicke and Nino Srour. Thank you for your long-standing support and for allowing me to take time from work to focus on my research studies. I would like to also thank my colleagues in the Acoustic Signal Processing Branch, Jerry Gerber, Steve Tenney, Mike Scanlon and many others for their cooperation and continue encouragement. In addition, I'm very grateful to Brian Sadler. Thanks for working and mentoring me prior to my PhD journey. Your research insight, encouragement and confidence in me really paved the road for me to go back to school. I look forward to working and collaborating with you again in the near future.

I would like to thank my parents for their love and support and instilling the value of higher education. Thank you "Bô´ Mẹ". Needless to say, without your love and sacrifice, I would not be where I am today. Next, I would like to thank my mother-in-law. Thank you "Mẹ" for helping take such good care of the family while I'm pursuing my studies. We could not have managed without you. Also, I would like to express my sincere gratitude to Mr. and Mrs. Jack Hanson for your teaching, generosity, love, and support that you and your family provided to me and my family through grade schools. Your kindness, help, and love toward others less fortunate have inspired all of us to do the same.

As always, I'm truly grateful and thankful to my family – my wife Chi, my son

Quân, my daughter Minh and my son An – for all their loving support and sacrifice. Chi, thank you so much for your love, patience and understanding. Many times I want to give up because the journey proved to be quite challenging and difficult. However, I was able to continue because of your constant encouragement and unwavering faith in me. Quân, Minh and An, thank you for bringing me so much joy and love. Thank you for always running to the door, greeting and hugging "Daddy" when I come home late from studying. Your love and affection is very self-assuring and revitalizing. I look forward to coming home early from now on...

Contents

List of Figures

Chapter 1

Introduction

Ad-hoc networks of small, low-power, and densely populated sensor nodes are being envisioned and developed for a wide range of applications due to a confluence of technological advances in areas such as micro-electromechanical system (MEMS) sensor devices, micro-processors, wireless communications, and battery sources. Inexpensive remote sensing devices deployable in large numbers and networked via wireless links and/or the internet provide unparalleled capabilities for instrumenting and monitoring the surrounding environment [1, 2] over large-coverage areas. Such ad-hoc sensor networks (AHSN) find use in military and civilian applications including target tracking [3], classification [4, 5], and source localization [5, 6] applications, weather forecasting and environmental monitoring applications [7–9], inventory warehouse tagging and tracking applications [10], patient monitoring [11], and biological monitoring and animal behaviorial tracking applications [12, 13]. Typically, these AHSN's are comprised of wireless sensor devices with integrated on-board sensing, processing and communication capabilities. Each sensor node is equipped with a single or a combination of sensing modalities such as acoustic, seismic, magnetic, infrared, chemical, biological and other low-power/low-cost modalities. However, with very few exceptions, they have limited communication and computational capabilities as they are typically battery-powered. As interest in distributed sensor networks has proliferated in recent years, the emphasis is increasingly placed on the design and development of efficient algorithms and architectures for network routing, wireless communications and sensor (data) fusion in AHSN's.

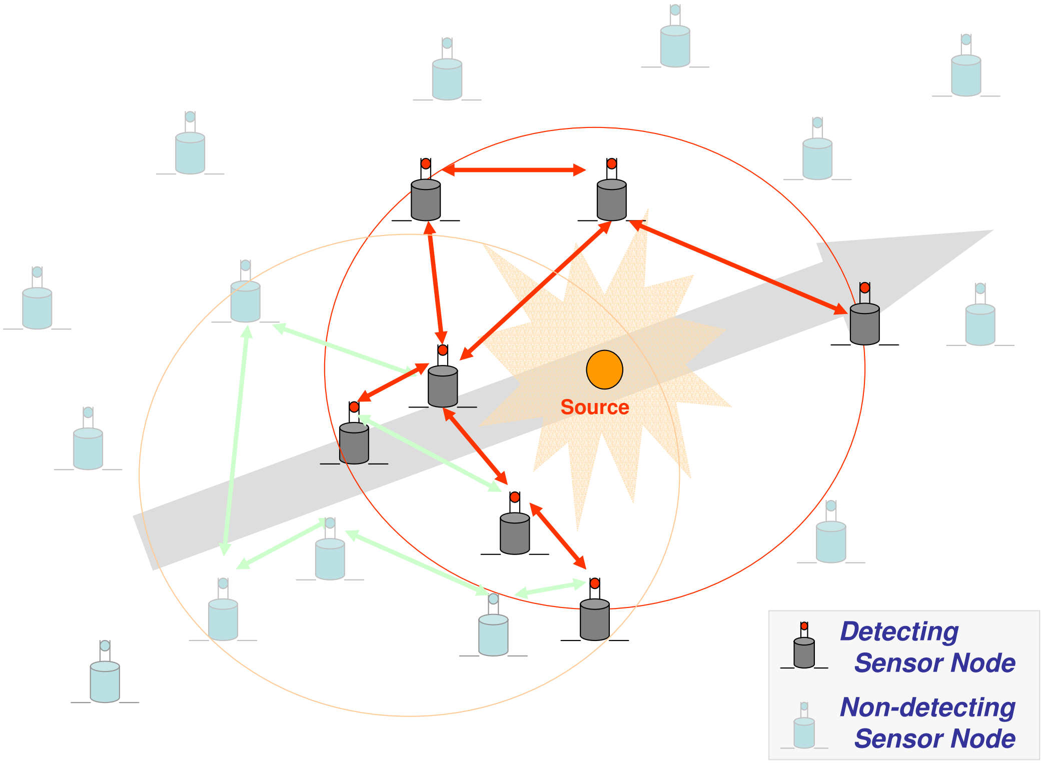


Figure 1.1: Randomly distributed field of low-complexity acoustic sensor nodes with a set of detecting sensor nodes forming an ad-hoc network.

In this dissertation, the problem of interest involves tracking of moving acoustic sources in large-scale sensor networks as shown in Fig. 1.1. We assume that all sensor nodes in the network collect range-based measurements at a fixed rate. However, at any given time, only a subset of detecting nodes, *i.e.*, nodes in proximity to the acoustic source, have range-based measurements of sufficiently high-quality measurements for source localization and tracking.

Fig. 1.2 depicts the tracking problem of interest in further details and illustrates the key challenges that arise in this tracking problem. As shown in the figure, to produce a tracking estimate at any given time t, one key challenge is how to acquire and manage all the distributed sensed data up to and including time t. For instance in Fig. 1.2, tracking at $t = 2$ requires using all available data from $t = 0$ and $t = 1$. Clearly, the size of the required data for performing tracking increases for increasing t. Due to the increasing data set size, there is a need for efficient space-time data fusion methods for fusion in space and fusion in time in a systematic manner. In addition, the figure shows that both sensor fusion in space and in time have to be implemented over the underlying network topology of the AHSN. The additional challenges that arise from the problem of interest include: (i) where the processing takes place, (ii) which nodes should be part of an *active computation network* (ACN) to perform space-time sensor fusion, and (iii) which fusion algorithms are to be used and implemented over the underlying network topology.

One approach is to employ a centralized algorithmic framework for tracking in AHSN's. A generalization of the centralized framework is a tree-structure framework where a tree-like network topology is formed throughout the network. Data, routed and fused as information, is propagated up the tree to the root node. Although the tree-structure methods perform well in static networks, challenges remain for these methods in dynamic networks where the network size and connections can change due node failure and/or node mobility. For example, when nodes fail and/or connections change in the network, the network tree-topology may need to be reconstructed and data/information from the "affected" nodes are either lost or need to be rerouted through the other “non-affected” nodes. Rerouting in tree-structure networks can be quite cumbersome as it requires additional associated overhead information and additional processing. In addition, tree-structure methods suffer from scalability issues as the sensor networks increase in size. As a result, decentralized algorithm frameworks are becoming more attractive than their centralized counterparts for sensor fusion in AHSN’s [14–17].

In this dissertation, we develop a decentralized algorithmic framework for systematic tracking of moving acoustic sources in large-scale ad-hoc sensor networks. The tracking algorithms we developed perform computationally efficient and iterative space-time processing. Fusion in space is performed by fusing current sensed data that is sufficiently high-quality from the sensor nodes to produce the current source location estimate. For example, at any given time each node in the sensor network only needs to communicate with its neighbors if it has detected the source; if it has detected the source, it participates with other “detecting” nodes in forming an ad-hoc detecting subnetwork to perform location estimation as shown in Fig. 1.1. Subsequently, the location estimates are indexed as they become available and fused iteratively in time to produce tracking estimates as shown in Fig. 1.2. Both fusion in space and fusion in time are performed distributively over the ad-hoc networks by exploiting distributed algorithms of computation of averages [3,17–20]. The distributed algorithms we developed are locally-constructed at each participating sensor node in the AHSN exploiting only the locally available observations and local network connectivity information. These distributed algorithms are inherently progressive in that the estimates they generate progressively improve with the number of iterations. In particular, the algorithms we developed are also resource efficient, scalable and fault-tolerant asymmetry. A key advantage of these algorithms is their ability to readily

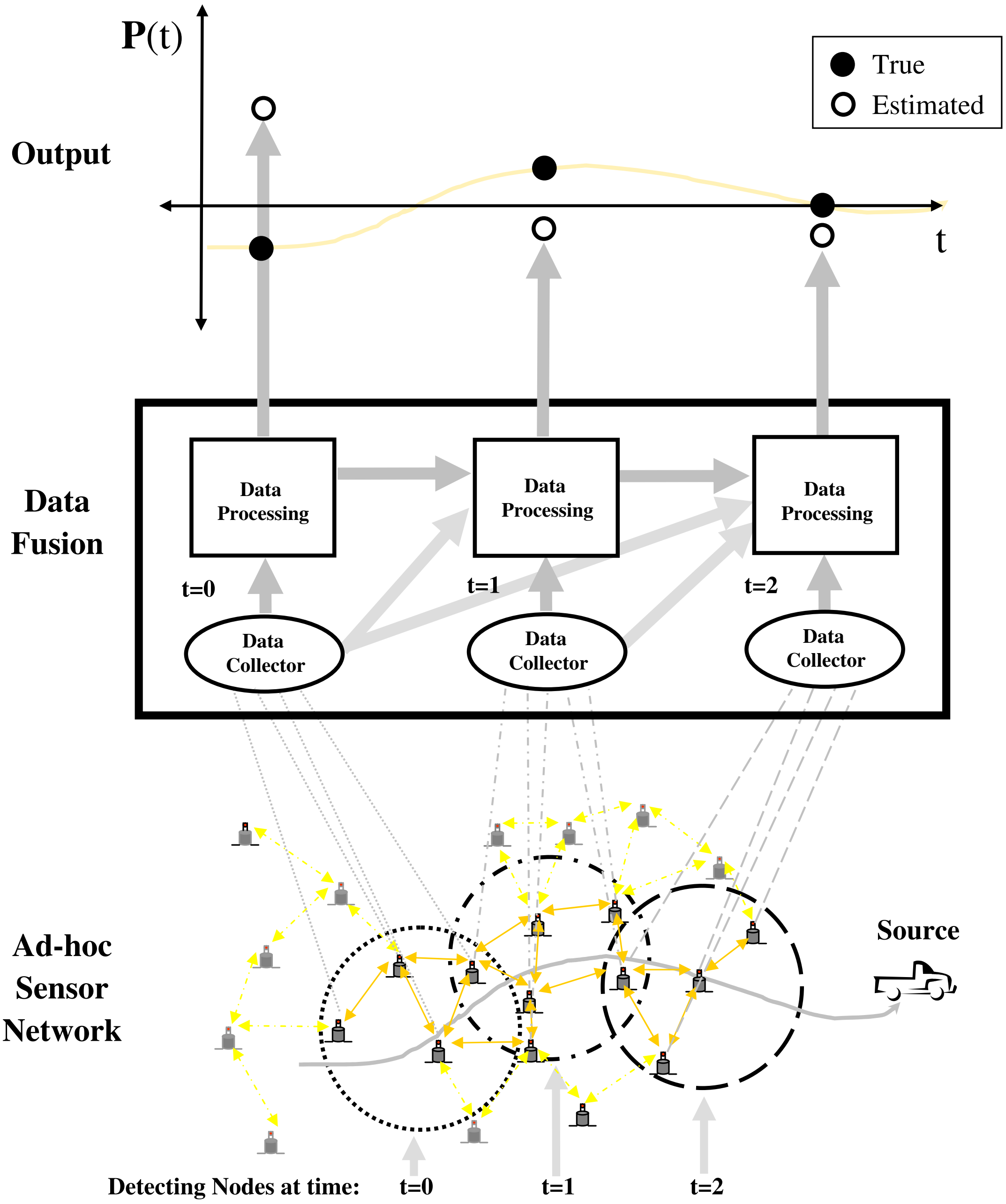


Figure 1.2: Source tracking in a large-scale sensor network involves (i) performing data fusion in space (at a fixed time instant) via a subnetwork formed by a set of detecting nodes in the vicinity of the source, and (ii) performing data fusion in time via a sequence of subnetworks.

adapt to local changes in network topologies. In addition to developing the distributed algorithms for AHSN, we present methods for optimizing and characterizing the performance of the proposed distributed spatial-temporal sensor fusion algorithms as a function of the quality of the acoustic sensor measurements, the source dynamics, the sensor density, and the network topology.

1.0.1 Outline of Thesis

In this thesis, we design and develop a decentralized algorithmic framework for systematically performing spatial-temporal sensor fusion for large-scale sensor networks as shown in Fig. 1.2. To that end, we present the following chapters.

In the first part of Chap. 2, we present the system models for describing the various components for the tracking problem of interest. We first model motion dynamics of the source via a state-space model for the location and velocity of the source in two dimensions. Next, we develop tractable acoustic sensor-source measurement models with acoustic propagation effects that allows us to study and analyze the range-based problem of interest. Next, we present the sensor distribution model which describes the spatial distribution of sensor field. In the second part of Chap. 2, we develop performance bounds for estimating the (centralized) location of source via AHSN and present simulation-based analysis for the developed bounds.

In Chap. 3, in the first part, we present network models with network topologies whereby each node is assumed to establish bidirectional communication and routing with its neighboring nodes. Then, in the second part, we present the basic (scalar) distributed algorithm modules for computations of averages which can be used to develop more complex, multi-dimensional algorithms for distributed space-time sensor fusion in AHSN. We conclude this chapter with briefing simulation-based analysis via a DC signal in noisy observations example.

In Chap. 4, we design and develop sensor fusion algorithms for spatial processing via a distributed network of sensors. In particular, we present resource efficient source localization algorithms that trade-off performance and complexity and can be implemented distributively in decentralized settings. We discuss the key steps in mapping the centralized estimators in the forms of weighted averages so distributed computations can be performed locally at all participating nodes in the network. Then, we present simulation results and discuss performance analysis comparing the distributed source localization algorithms with their centralized counterparts. We conclude this chapter by characterizing the relationships among the network, sensor-source and algorithm parameters to develop methods optimizing algorithm performance.

In Chap. 5, we design and develop sensor fusion algorithms that incorporate spatial fusion estimates for tracking. In particular, we present distributed algorithms that can be implemented decentrally via a state estimation framework; the tracking algorithms we develop exploit the improved distributed computations algorithms for performing spatial-temporal fusion locally at each participating node over changing network topologies. Then we conclude this chapter tracking simulation results and performance analysis.

Finally, in Chap. 6, we summarize our contributions in developing a algorithmic framework for decentralized tracking, improved distributed computation algorithms, distributed spatial-temporal algorithms source localization and tracking. Then, we suggest and discuss potentially interesting and challenging directions for future research in sensor networks related areas.

Chapter 2

System Modeling and Performance Bounds

In order to investigate the tracking of a moving acoustic source in large-scale AHSN as shown in Fig. 1.2, we first need to develop system models to provide a framework for developing the distributed sensor fusion algorithms in space and in time. In particular, we seek to develop models to describe the dynamics of a moving source, the sensor-source measurements and the distribution of sensor nodes. Once the system models are available, we seek to develop performance bounds to analyze and characterize the lower bounds of localization uncertainty.

In this chapter, we first present the system models in Sec. 2.1 and then present performance bounds in Sec. 2.2.

2.1 System Models

In our investigation, we are interested in networks of low-cost, low-power, omnidirectional and passive sensors such as acoustic, seismic, magnetic, infrared and other low-power/low-cost sensing modalities. We choose to focus on acoustic sensing due to (i) the wide availability of acoustic sensor systems and testbeds, (ii) the wide area of sensor coverage for many applications, and (iii) the broad range of civilian and military acoustic sensor network applications. For instance, distributed sensor networks comprised of acoustic (including infrasonic) sensors are being used to monitor volcanic eruptions [9], to recognize and locate specific animal calls [13], to determine the trajectory of a projectile and to localize the position of a sniper [21, 22], and to perform direction-of-arrival (DOA) and track ground targets [3, 5, 23].

In this section, we first present a state-space model describing the motion dynamics of a moving source in Sec. 2.1.1. Next, we present signal in noise measurement models describing the acoustic source of interest in Sec. 2.1.2, and discuss acoustic propagation model and the associated energy-based acoustic model based on the *received signal strength* (RSS) in Sec. 2.1.3 and Sec. 2.1.4 respectively. Then, we briefly present a sensor distribution model in which we generate the location of the source with respect to the lay-out of the sensor field in Sec. 2.1.5.

2.1.1 State-Space Model

The setting of interest involves a single moving source across the large-scale sensor network as shown in Fig. 1.2. We assume that the motion dynamics of the source can be accurately modeled via a state-space model for the location and velocity of the source in two dimensions. For convenience, it is assumed that the source has independent motion components in two dimensions, and the motion in each dimension follows a constant velocity model with a random acceleration [24]. In particular, letting $P_{sn}(t)$, and $V_{sn}(t)$ denote the position and velocity, respectively, of the source in the nth dimension ($n = 1,\, 2$) at time t, the dynamics of the state vector $X_{sn}(t) =$

$[P_{sn}(t)\ V_{sn}(t)]^T$ are described by the following state-transition equation

$$X_{sn}(t+1) = \mathbf{F}\, X_{sn}(t) + \mathbf{G}\, A_n(t),\ t = 0, 1, \ldots \tag{2.1}$$

where $A_n(t)$ denotes the random acceleration, modeled as a zero-mean white Gaussian sequence with power $\mathbf{Q} = \sigma_A^2$,

$$\mathbf{F} = \begin{bmatrix} 1 & T_s \\ 0 & \varrho(T_s) \end{bmatrix}, \mathbf{G} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{2.2}$$

and where T_s denotes the snapshot update interval. The parameter $0 < \varrho(T_s) \leq 1$ is given by $\varrho(T_s) = \varrho_o^{T_s}$ with $\varrho_o = \sqrt{(\sigma_V^2 - \sigma_A^2)/\sigma_V^2}$, where σ_V^2 denotes the variance of $V_{sn}(t)$. Further details and discussions on the parameter ϱ and the two-dimensional state-space model are in App.A.1.

2.1.2 Sensor Source Measurement Models

In this sub-section, we consider and develop tractable acoustic sensor-source measurement models with acoustic propagation effects that allows us to study and analyze the problem of spatial-temporal sensor fusion in AHSN. Then we discuss and characterize the propagation medium via the *spatial transfer function* (STF) and its effects on the sensor-source measurement models.

The setting of interest involves a single acoustic source in a free-field of randomly distributed acoustic sensors. We assume that at a time instant t, $M(t)$ sensors have detected a single radiating acoustic source, *e.g.*, via energy threshold detection and $M(t)$ nodes are assumed to form an AHSN as illustrated in Fig. 1.1. The location of the ith sensor is assumed to be fixed in time and denoted by

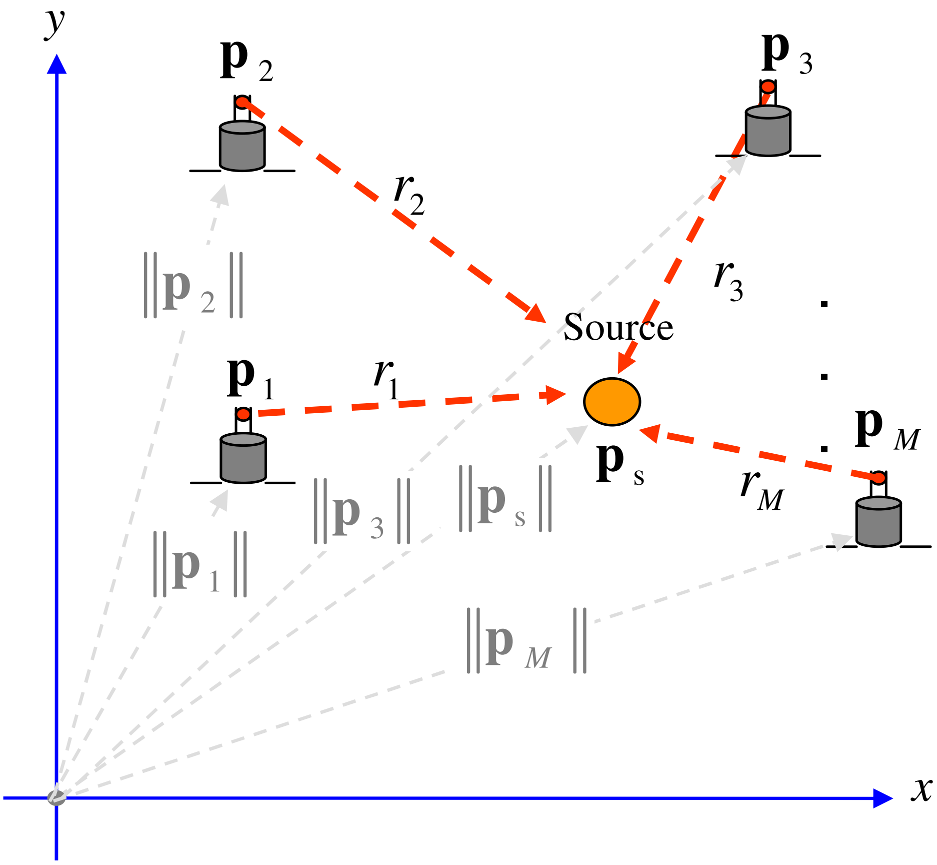


Figure 2.1: AHSN of $M(t)$ detecting sensors. Location of sensor i is $\mathbf{p}_i = \left[x_i \; y_i\right]^T$, $i = 1, 2, \cdots, M(t)$ and $r_i(t)$ denotes the distance between sensor i and the source location $\mathbf{p}_s(t) = \left[x_s(t) \; y_s(t)\right]^T$.

$\mathbf{p}_{in} = \left[P_{i1} \;\; P_{i2}\right]^T$ while $r_i(t)$ denotes the distance between the ith sensor and the source location $\mathbf{p}_{sn}(t) = \left[P_{s1}(t) \;\; P_{s2}(t)\right]^T$. For notational convenience, we also denote $\mathbf{p}_{in} = \mathbf{p}_i = \left[x_i \; y_i\right]^T$ and $\mathbf{p}_{sn}(t) = \mathbf{p}_s(t) = \left[x_s(t) \; y_s(t)\right]^T$ as in "x-y" Cartesian coordinates as in Fig. 2.1. The received signal at the ith sensor is modeled as

$$z_{i,k}(t) = S_k(t) \cdot h_i(\mathbf{p}_s(t), \boldsymbol{\Gamma}(t)) + \eta_{i,k}(t); \; i = 1, \cdots, M, \; k = 1, \cdots, L, \tag{2.3}$$

where $S_k(t)$ is the signal at (data) snapshot k, $\eta_{i,k}(t)$ is the noise at the ith sensor node at snapshot k, $h_i(\mathbf{p}_s(t),\, \mathbf{\Gamma}(t))$ is the STF from the source to sensor i, and L is the number of statistically independent, identically, distributed (i.i.d.) data snapshots. It is convenient to recast (2.3) in the following vector form

$$\mathbf{z}_k = S_k(t) \cdot \mathbf{h}(\mathbf{p}_s(t),\, \mathbf{\Gamma}(t)) + \boldsymbol{\eta}_k(t);\ k = 1, \cdots, L, \tag{2.4}$$

where $\mathbf{z}_k(t) \triangleq \left[z_{1,k}(t)\ z_{2,k}(t)\ \cdots\ z_{M,k}(t)\right]^T$, $\mathbf{h}(\cdot) = \left[h_1(\cdot)\ h_2(\cdot)\ \cdots\ h_M(\cdot)\right]^T$, and $\boldsymbol{\eta}_k(t) \triangleq \left[\eta_{1,k}(t)\ \eta_{2,k}(t)\ \cdots\ \eta_{M,k}(t)\right]^T$.

Stochastic Signal Model

We first consider a stochastic signal in noise measurement model, according to which the sensor noise components are statistically independent and identically distributed in time and space, while the signal components are statistically independent over time and space, and identically distributed over time at each node. From (2.3), $S_k(t) \sim \mathcal{N}(\mu_s,\, \sigma_s^2)$ and $\eta_{i,k}(t) \sim \mathcal{N}(0,\, \sigma^2{}_\eta)$. Without loss of generality, we assume $\mu_s = 0$. The noise and signal are statistically independent and they are zero-mean, Gaussian distributed, random, stationary processes. Given that $\boldsymbol{\eta}_k(t) \sim \mathcal{N}(\mathbf{0},\, \sigma_\eta^2 \mathbf{I})$, $\mathbf{z}_k(t) \sim \mathcal{N}(\mathbf{0},\, \boldsymbol{C}(\mathbf{h}))$, where $\boldsymbol{C}(\mathbf{h})$ is defined as

$$\boldsymbol{C}(\mathbf{h}) = E\left\{\mathbf{z}_k(t) \cdot \mathbf{z}_k(t)^H\right\} = \sigma_s^2 \mathbf{h}(\mathbf{p}_s,\, \mathbf{\Gamma})\, \mathbf{h}(\mathbf{p}_s,\, \mathbf{\Gamma})^H + \sigma_\eta^2 \mathbf{I}. \tag{2.5}$$

Viewing σ_s^2 as unknown in the random signal model described by (2.4), the conditional probability density function (pdf) of $\mathbf{z}(t) = [\mathbf{z}_1(t)^T\ \mathbf{z}_2(t)^T\ \cdots\ \mathbf{z}_L(t)^T]^T$ given the vector of all unknown parameters, $\boldsymbol{\theta} = [\mathbf{h}(\mathbf{p}_s(t),\, \mathbf{\Gamma}(t))^T\ \sigma_s^2]^T$, is given by

$$p_{\mathbf{z}}(\mathbf{z}(t) \mid \boldsymbol{\theta}) = \prod\nolimits_{k=1}^{L} \frac{\exp\{-\frac{1}{2}\mathbf{z}_k^H(t) \boldsymbol{C}^{-1}(\boldsymbol{\theta})\, \mathbf{z}_k(t)\}}{[2\pi \det(\boldsymbol{C}(\boldsymbol{\theta}))]^{\frac{1}{2}}}. \tag{2.6}$$

Deterministic Signal Model

In the absent of priors, we also consider a simple deterministic signal-in-noise measurement model arising via (possibly) L i.i.d. data snapshots. From (2.3), $S_k(t)$ is now deterministic with unknown signal source power $E\left\{S_k^2(t)\right\} = S_k^2(t) = \sigma_s^2$ and $\eta_{i,k}(t) \sim \mathcal{N}(0,\, \sigma^2{}_\eta)$. Similarly, the vector formulation of (2.3) for deterministic signal is $\mathbf{z}_k(t) \sim \mathcal{N}(\boldsymbol{\mu}(\mathbf{h}),\, \sigma_\eta^2\mathbf{I})$, where $\boldsymbol{\mu}(\mathbf{h}) = \sigma_s\mathbf{h}(\mathbf{p}_s,\, \boldsymbol{\Gamma})$. The cdf of $\mathbf{z}(t)$ given $\boldsymbol{\theta}$ is given by

$$p_{\mathbf{z}}(\mathbf{z}(t) \mid \boldsymbol{\theta}) = \prod\nolimits_{k=1}^{L} \frac{\exp\left\{-\frac{1}{2\sigma_s^2}|\mathbf{z}_k(t) - \boldsymbol{\mu}(\boldsymbol{\theta})|^2\right\}}{[2\pi(\sigma_s^2)^M]^{\frac{1}{2}}}. \tag{2.7}$$

2.1.3 Propagation Model

In this section, we describe the general propagation models for acoustic signals in free-field setting. We present a discussion of the acoustic propagation effects that can be characterized by the STF and then discuss how we can model the acoustic signal attenuation via the STF for the AHSN setting of interest. Then, we propose a simplified attenuation model based on the transmission loss.

Propagation Effects

The STF at the ith defined in (2.3) is a very complex function of many environmental parameters. In general acoustic settings, the STF depends on a combination of parameters such as temperature, humidity, wind speed and direction, ground conditions, physical barriers and other environmental factors. Regardless of the settings, $\boldsymbol{\Gamma}(t)$ contains the environmental parameters affecting the STF. The propagation effects in (2.3) also depend on acoustic source, $S(t)$. Acoustic sources of interest for unattended ground sensor (UGS) applications tend to be broadband source with strong sinusoidal components due to rotating machinery (engines) and may include contributions from contact with the ground surface and exhaust noise for ground vehicles [23,25]. In [26,27], the authors discuss the propagation phenomena that affect a sinusoidal signal emitted by a non-moving source as observed by a sensor network. They are: (i) the transmission loss (TL) caused by spreading of the wavefronts, refraction by wind and temperature gradients, ground interactions, and other absorption effects, (ii) the additive noise at the sensors caused by directional interference, wind noise and thermal noise, and (iii) random fluctuations in the amplitude and phase of the signals caused by scattering from random inhomogeneities in the atmosphere such as turbulence.

TL is defined as the attenuation of acoustic energy from a reference value $S_{\mathrm{ref}}(t)$, which is observed in free space at 1 m from the source, to the actual acoustic energy observed at the ith sensor. To a first approximation, the acoustic energy spreads spherically; that is, it diminishes as the inverse of the squared distance from the source. However, TL for sound wave propagating near the ground involves many complex, interacting phenomena, so that the spherical spreading condition is seldom observed except at close range (*e.g.*, less than 100 m from the source) [28]. Several well refined and accurate numerical procedures for calculating TL are presented in [29].

In a realistic environment, the sensor noises in (2.3) may not be independent from sensor to sensor. Interference from an undesired source may produce a common additive noise term that can be correlated (spatially) from node to node and the inference effects can be very difficult to model. Wind noise, for example, near the ground can exhibit spatial correlations over distances of several meters [30]. In contrast, the thermal noise component is well-modeled independent from node to node even when the nodes are closely spaced together within the sensor network.

The scattering of the acoustic signal caused by turbulence can be particularly significant in free-field acoustic setting. The turbulence consists of random atmospheric motions occurring on time scales from seconds to several minutes. Scattering from these motions causes random fluctuations in the acoustic signal at the individual sensor nodes and diminishes the cross coherence of signals between nodes [31]. In [27], a scattering model for a deterministic source is presented. The scattering modifies the signal at the sensor by spreading a portion of the power from the deterministic mean component into zero-mean random process, *i.e.*, (2.3) becomes

$$z_{i,k}(t) = (\sqrt{1-\zeta}\cdot S_k^d(t) + \sqrt{\zeta} S_k^r(t)) \cdot h_i(\mathbf{p}_s(t),\, \mathbf{\Gamma}(t)) + \eta_{i,k}(t);\ i = 1, \cdots, M,\ k = 1, \cdots, L, \tag{2.8}$$

where $S^d(t)$ and $S^r(t)$ are the deterministic and random components of the received signal at the sensor i and $\zeta \in [0,\, 1]$ is the saturation parameter [32]. The scattering may be *weak* ($\zeta \approx 0$) or *strong* ($\zeta \approx 1$), which are analogous to Rician and Rayleigh fading in the radio propagation literature.

Simplified Acoustic Propagation Model

In modeling the free-field acoustic source in (2.3), we consider the two extreme cases in (2.8) and assume either $S(t) = S^r(t)$ for stochastic model or $S(t) = S^d(t)$ for deterministic signal model.[1] We seek to characterize the propagation medium with the most salient parameters to capture the general effects of the medium over the detection range of interest without making the STF overly complicated.

Toward that end, we assume that the acoustic source is point source radiating omni-directionally and the dimension of the source is assumed to be small compared

[1] Initially, we used the stochastic signal model with $L \geq 500$ snapshots, which closely represent many of the acoustic signatures of military interest [23]. However, we later switched to the deterministic model for ease of simulation with $L = 1$ snapshot. We find that both models are tractable and applicable to our research.

to the distance between the sensor and the source. We also assume only additive white Gaussian noise (AWGN) at each sensor node and neglect the effects such as interference and wind noise. To simplify the STF, we assume spherical spreading to be the dominant factor in signal attenuation and approximate TL as

$$\mathrm{TL} \approx (\frac{\beta}{2}) \cdot 10 \log_{10} r(t) \ , \tag{2.9}$$

where β is the TL coefficient for acoustic signal in air [33] and $r(t)$ is the range from the source. Then, the STF at sensor i can be approximated via

$$h_i(\mathbf{p}_s, \mathbf{\Gamma}(t)) = 10^{\frac{-\mathrm{TL}}{10}} \approx r_i(t)^{\frac{-\beta}{2}} = h_i(\mathbf{p}_s(t), \beta) \ , \tag{2.10}$$

where $r_i(t)$ is the range from sensor i to the source.

2.1.4 Energy-based RSS Modeling

Most localization methods depend on physical variables measured by or derived from sensor readings such as time-of-arrival (TOA) and/or time-difference-of-arrival (TDOA) [34, 35], angle-of-arrival (AOA) or direction of arrival (DOA) [23, 36] and energy-based RSS [6, 37, 38]. Next, we present energy-based RSS models based on the simplified STF discussed in Sec. 2.1.3. Given that the propagation effects can be characterized by the STF in (2.10) within the range of interest, the expected power at the ith sensor via is given by

$$\sigma_i^2(t) = \frac{\sigma_s^2}{r_i^\beta(t)} + \sigma_\eta^2(t) \ , \tag{2.11}$$

where $\sigma^2_{\mathrm{RSS}_i}(t) \triangleq \sigma_s^2/r_i^\beta(t)$ is the RSS at the ith sensor. We assume that each sensor within the AHSN can estimate the RSS from a radiating acoustic source within its detection range. The signal-to-noise ratio (SNR) is defined as $\sigma_s^2/\sigma_\eta^2(t)$ at 1 m from the source location.

Estimates of β can be obtained via experimental data. For instant, under benign acoustic conditions (*e.g.*, negligible wind and turbulence), detection range less than 1 km, a loud source such as a military vehicle yields β estimates in the range $1.9 \lessapprox \beta \lessapprox 2.2$. Fig. 2.2 shows that the TL in terms of sound pressure level (SPL) for a large military vehicle compare to $1/r^\beta(t)$ attenuation for $\beta = 2$, and 2.2 as a function of range. As the figure reveals at $r \leq 100$ m, acoustic attenuation is in close agreement with the $1/r^2$ curve, while at distances greater than 100 m, the acoustic attenuation fluctuates between the $1/r^2$ and $1/r^{2.2}$ curves. To simplify further, we assume $\beta = 2$ in (2.9) over the detection range of interest.[2]

In the following, we present RSS models for both stochastic and deterministic signal models.

Simplified Stochastic Signal

We next consider a simplified stochastic signal-in-noise measurement model arising from (possibly) L statistically i.i.d. data samples.[3] We model the source-node range measurements at ith node as an L-vector

$$\mathbf{z}_i(t) = \frac{S(t)}{r_i}\,\mathbf{1}_L + \boldsymbol{\eta}_i(t) \tag{2.12}$$

[2] In our simulations and analysis, we typically perform source localization within the range of 50 to 200 m. So $\beta = 2$ in (2.9) accurately models the propagation effects within the range of interest.

[3] Here, we assume that the L data samples are obtained within a time interval T_{ds} where $T_{ds} < T_s$

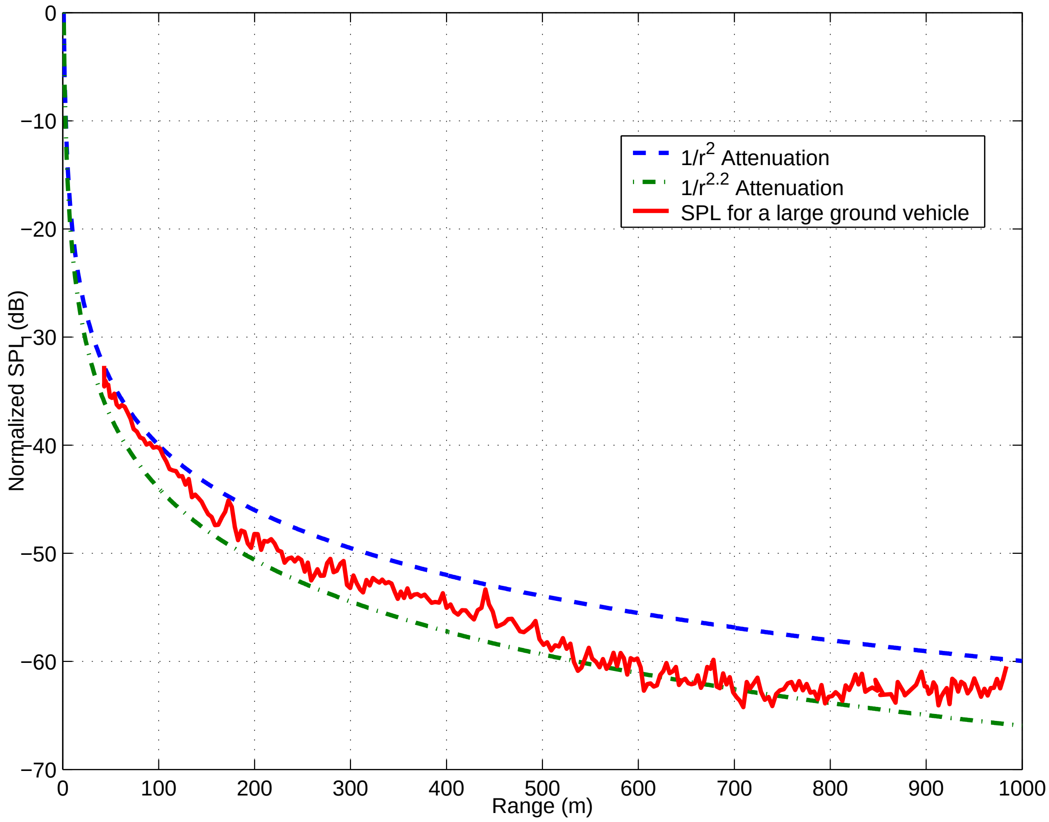


Figure 2.2: Normalized $\frac{1}{r^2}$, and $\frac{1}{r^{2.2}}$ attenuation curves vs. range for acoustic transmission loss, and plot of the acoustic sound pressure level (SPL) vs. range for a large ground vehicle.

for $i = 1, \cdots, M(t)$, where the $\boldsymbol{\eta}_i(t) \sim \mathcal{N}(\mathbf{0}, \sigma_\eta^2 I_L)$. From (2.11), given $\beta = 2$ and $E\left\{S(t)^2\right\} = \sigma_s^2$ the expected power at the ith sensor is given by

$$\sigma_i^2(t) = \frac{\sigma_s^2}{r_i^2(t)} + \sigma_\eta^2 = \sigma_{\mathrm{RSS}_i}^2(t) + \sigma_\eta^2$$

where $\sigma_{\mathrm{RSS}_i}^2(t) \triangleq \sigma_s^2 / r_i^2(t)$ is the RSS at the ith sensor with σ_s^2 and $r_i(t)$ denoting the (unknown) source signal power (received power at nominal distance 1) and the distance between the ith node and the source, and $\mathbf{1}_L$ is a vector of L ones. Assuming that the relative source location does not appreciatively change over successive samples at fixed time instant t, σ_i^2 can be estimated and the RSS at the ith sensor can be estimated as $\widehat{\sigma^2}_{\mathrm{RSS}_i} = \widehat{\sigma^2}_i - \widehat{\sigma^2}_\eta$.

Simplified Deterministic Model

Similarly, we consider a simple deterministic signal-in-noise measurement model arising via (possibly) L statistically i.i.d. data samples. We model source-node range measurements at the ith node as an L-vector

$$\mathbf{z}_i(t) = \mathbf{s}_i(t) + \boldsymbol{\eta}_i(t) = \frac{\sigma_s}{r_i(t)}\mathbf{1}_L + \boldsymbol{\eta}_i(t), \tag{2.13}$$

for $i = 1, \cdots, M(t)$, where the $\boldsymbol{\eta}_i \sim \mathcal{N}(\mathbf{0},\, \sigma_\eta^2 I_L)$, $\sigma^2_{\mathrm{RSS}_i}(t) = \sigma_s^2/r_i^2(t)$, with σ_s^2 and $r_i(t)$ denoting the (unknown) source signal power and the distance between the ith node and the source, and $\mathbf{1}_L$ is a vector of L ones. When considering this signal model, the source-location estimators in Sec. 4.1 exploit the locally available minimum-variance unbiased estimates (MVUEs) of $\sigma^2_{\mathrm{RSS}_i}(t)$, *viz.*,

$$\widehat{\sigma^2}_{\mathrm{RSS}_i}(t)\mathbf{1}_L = \min\left[\mathbf{z}_i^2(t) - \sigma_\eta^2\mathbf{1}_L,\, \mathbf{0}_L\right]. \tag{2.14}$$

Without loss of generality, we assume $L = 1$ from hereon in for deterministic signal model.

We remark that the AHSN for a fixed time t is formed via threshold detection by including in the AHSN only nodes with $\widehat{\sigma^2}_{\mathrm{RSS}_i} > \sigma_T^2$, for some suitably preset threshold $\sigma_T^2 > 0$. We remark that for a given detection threshold, σ_T^2, the probability of detection at the ith node is a function of the source radiating power and the range between the source and the node; in particular, with a slight abuse of notation, the

ith node detection probability is given by

$$
\begin{aligned}
P_i^{(\mathrm{D})}(t) &= \Pr\left[y_i^2(t) > \sigma_T^2\right] = 1 - \Pr\left[y_i^2(t) < \sigma_T^2\right] \\
&= 1 - \Pr\left[-\sigma_T - \frac{\sigma_s}{r_i(t)} < \eta_i(t) < \sigma_T - \frac{\sigma_s}{r_i(i)}\right] \\
&= \mathcal{Q}\left(\frac{\sigma_T}{\sigma_\eta} - \frac{\sigma_s}{\sigma_\eta r_i(t)}\right) + \mathcal{Q}\left(\frac{\sigma_T}{\sigma_\eta} + \frac{\sigma_s}{\sigma_\eta r_i(t)}\right) , \qquad (2.15)
\end{aligned}
$$

where $\mathcal{Q}(x) = (\sqrt{2\pi})^{-1} \int_x^\infty e^{-t^2/2}\, dt$.

2.1.5 Sensor Distribution Model

In this section, we present the distribution of the sensor nodes in a sensor field. If the global statistical information for a random field is available (*i.e.*, geographical terrain), this information can be used to distribute sensors in a resource efficient way to achieve the desired performance [39]. However, in many applications, the statistical background information may not be available prior to sensor deployment but only available after sensor deployment.

We took the approach of having no prior distribution information and we model the sensor field as spatially uniform distributed as shown in Fig. 2.3. In our setting, we assume that a single (acoustic) source is randomly placed within the sensor field Ω of radius R and N sensors with sensor density D defined as $D = N/\pi R^2$. The inner circle (defined as a subspace Ω_o with radius R_o) around the source denotes the detection region. Given that detection has been made, Ω_o is also spatially uniform distributed and the number of detecting nodes, M, in Ω_o with density $D_o = M/\pi R_o^2 \approxeq D$, is Poisson distributed. This Poisson model is an attractive model when we take into account the dynamics of the source as it moves within the sensor field.

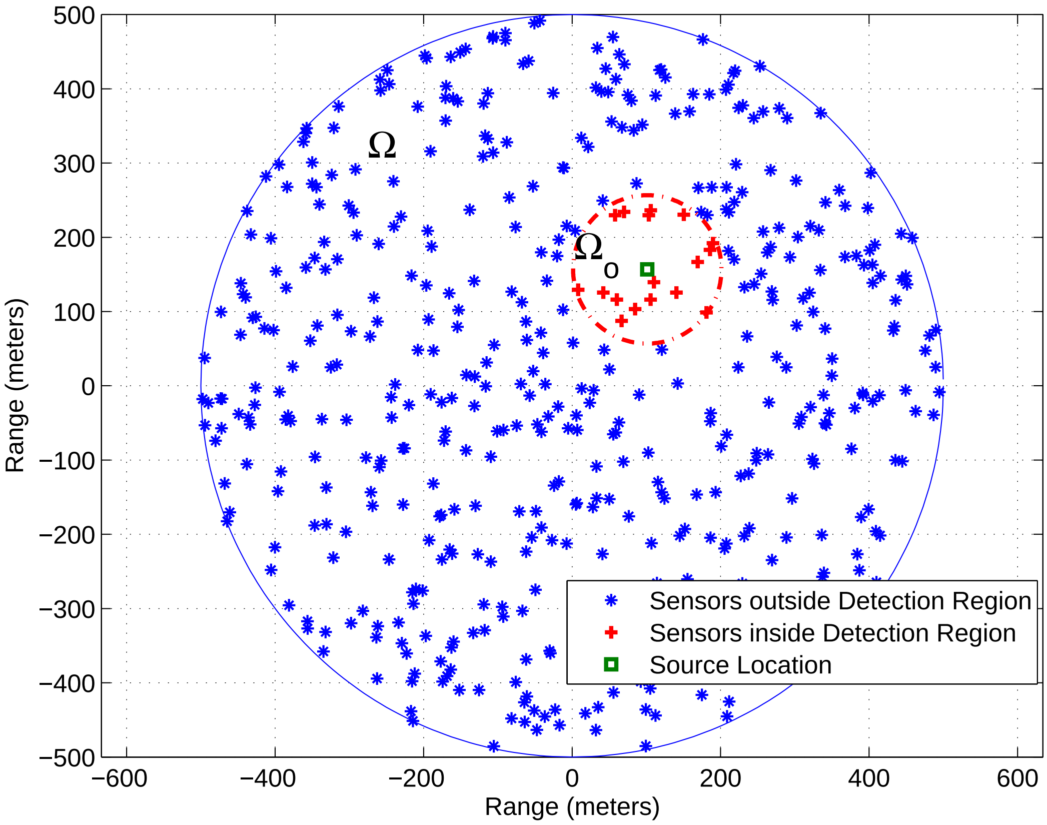


Figure 2.3: Simulated uniform distribution of $N = 500$ sensor nodes within the (blue) circle of 500 m radius. M sensors within the (red) circle of 100 m radius have detected the source, designated by the (green) square.

2.2 Performance Bounds

Sensor networks are attractive for performing spatial sensor fusion via localization due to their spatial diversity over (possibly) large areas. Some examples include sensor nodes self-locating (or self-calibration) within the sensor networks [34, 35], locating earthquake events [9], finding patient's whereabout in case of emergency, locating the position of a shooter in counter-sniper applications [21], locating and tracing odor and chemical plumes [40], and locating the animal calls in habitat-monitoring applications [13]. Thus, localization plays a very important role for many wide-ranging sensor networks applications.

In order to analyze the performance accuracy of source localization for a given model, performance bounds are needed to analyze and characterize the lower bound of localization uncertainty. For example, in [41], the Cramér-Rao Bounds (CRB) for ranging in AHSN are developed for anchored localization (*i.e.*, at least three node locations are known) and for anchor-free localization (*i.e.*, no absolute positions are known); and in [42], a Bayesian method, refer to as the Bayesian bound (BB) is derived instead of the CRB when sensing models are complex but when the uncertainty is Gaussian, the BB equals the CRB.

In this chapter, we develop performance bounds for estimating the (centralized) location of source via sensor networks. Toward this objective, we investigate the CRB for source localization for AHSN as shown in Fig. 1.1, where we assume that each sensor in the AHSN knows its own location. Although the CRB's can be loose bounds, they are most widely used MSE performance evaluation of unbiased (and less often biased) estimators due to its relative ease of computation. The CRB provides means for determining the best (perhaps overly optimistic) localization performance we can hope to achieve. In our AHSN setting, the CRB depends on the cdf's described in (2.6)–(2.7) and the STF in (2.10) but it does not required additional prior information or preliminary estimates like other performance bounds such as the Barankin bound [43]. For convenience, we omit the dependence on t in the CRB computation and analysis.

We begin in Sec. 2.2.1 by developing the CRB for both stochastic and deterministic signal models for the centralized setting. Then, we discuss the minimum mean squared error (MMSE) results for the CRB as function of signal and sensor network parameters in Sec. 2.2.2.

2.2.1 Cramér-Rao Bound

In this section, we compute the CRB's (in the centralized setting) for (2.4) for both signal models to obtain assessments of the theoretical performance limits of source location estimation and to later compare low-complexity estimators to the CRB's. Assuming that the data vector $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{\theta}), \boldsymbol{C}(\boldsymbol{\theta}))$ so that both the mean and the variance depends on $\boldsymbol{\theta}$, the general expression for the CRB with parameter $\boldsymbol{\theta}$ is $CRB(\boldsymbol{\theta}) = \mathbf{I}^{-1}(\boldsymbol{\theta})$, where the $\mathbf{I}(\boldsymbol{\theta})$ is the Fisher information matrix with $(p,\, q)$th element of $\mathbf{I}(\boldsymbol{\theta})$ defined as

$$[\mathbf{I}(\boldsymbol{\theta})]_{pq} = \frac{1}{2}\mathrm{tr}\left[\mathbf{C}^{-1}(\boldsymbol{\theta})\frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial \theta_{\mathrm{p}}}\mathbf{C}^{-1}(\boldsymbol{\theta})\frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial \theta_{\mathrm{q}}}\right] + \left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial \theta_{\mathrm{p}}}\right]\mathbf{C}^{-1}(\boldsymbol{\theta})\left[\frac{\partial \boldsymbol{\mu}(\boldsymbol{\theta})}{\partial \theta_{\mathrm{q}}}\right] \qquad (2.16)$$

for $p, q = 1, \cdots, Q$, where Q is the number of parameters in $\boldsymbol{\theta}$. In the case where L statistically independent snapshots are available, the CRB generalizes to $CRB(\boldsymbol{\theta}) = \frac{1}{L}\mathbf{I}^{-1}(\boldsymbol{\theta})$ [44]. Assuming Gaussian random processes, TL coefficient $\beta = 2$ and L snapshot estimates, the CRB on the MSE for estimating $\boldsymbol{\theta} = \left[\mathbf{p}_s\ \sigma_s^2\right]^T = \left[x_s\ y_s\ \sigma_s^2\right]^T$ in (2.6)–(2.7) based on model (2.4) is $CRB(\boldsymbol{\theta}) = \frac{1}{L}\mathbf{I}^{-1}(\boldsymbol{\theta})$, with $Q = 3$.

CRB Stochastic Signal Model

For the stochastic signal model with zero-mean, $[\mathbf{I}(\boldsymbol{\theta})]_{p,q}$ is just the first term in (2.16). Here for $i, j = 1, \cdots, M$, with $(i,\, j)$th element of $\boldsymbol{C}(\boldsymbol{\theta})$ is given by

$$[\boldsymbol{C}(\boldsymbol{\theta})]_{i,j} = [\boldsymbol{C}(x_s,\, y_s,\, \sigma_s^2)]_{i,j} = \begin{cases} \frac{\sigma_s^2}{r_i\, r_j} & i \neq j \\ \frac{\sigma_s^2}{r_i^2} + \sigma_\eta^2 & i = j \end{cases}, \qquad (2.17)$$

where the distance from sensor i to the source is $r_i = \sqrt{(x_i - x_s)^2 + (y_i - y_s)^2}$, and

$$\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{i,j}}{\partial \theta_1} = \frac{\sigma_s^2 \left[r_j^2(x_i - x_s) + r_i^2(x_j - x_s)\right]}{(r_i\, r_j)^3}, \tag{2.18a}$$

$$\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{i,j}}{\partial \theta_2} = \frac{\sigma_s^2 \left[r_j^2(y_i - y_s) + r_i^2(y_j - y_s)\right]}{(r_i\, r_j)^3}, \tag{2.18b}$$

$$\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{i,j}}{\partial \theta_3} = \frac{1}{(r_i\, r_j)}. \tag{2.18c}$$

To obtain the CRB on range estimation, it is convenient to convert $\boldsymbol{\theta} = \begin{bmatrix} x_s \; y_s \; \sigma_s^2 \end{bmatrix}^T$ into polar coordinates $\tilde{\boldsymbol{\theta}} = \begin{bmatrix} r_s \; \phi_s \; \sigma_s^2 \end{bmatrix}^T$, where $r_s = \sqrt{x_s^2 + y_s^2}$ and $\phi_s = \arctan(\frac{y_s}{x_s})$. Using the functional CRB form where $\tilde{\boldsymbol{\theta}} = \mathbf{g}(\boldsymbol{\theta})$, we obtain

$$CRB(\tilde{\boldsymbol{\theta}}) = \frac{1}{L}\frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \mathbf{I}^{-1}(\boldsymbol{\theta}) \frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}^T \tag{2.19}$$

where $\frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ is the 3×3 Jacobian matrix given by

$$\frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \frac{x_s}{r_s} & \frac{y_s}{r_s} & 0 \\ \frac{-y_s}{r_s} & \frac{x_s}{r_s} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.20}$$

Further details of the CRB calculation for stochastic signal model can be found in Appendix A.2.

CRB for Deterministic Signal Model

The CRB for the deterministic signal model can be derived straight from the general CRB expression in (2.16) where $[\mathbf{I}(\boldsymbol{\theta})]_{p,q}$ equals the second term in (2.16). Here for

$i = 1, \cdots, M$,

$$\frac{[\partial \boldsymbol{\mu}(\boldsymbol{\theta})]_i}{\partial \theta_1} = \frac{\sigma_s\,(x_i - x_s)}{r_i^3}, \tag{2.21a}$$

$$\frac{[\partial \boldsymbol{\mu}(\boldsymbol{\theta})]_i}{\partial \theta_2} = \frac{\sigma_s\,(y_i - y_s)}{r_i^3}, \tag{2.21b}$$

$$\frac{[\partial \boldsymbol{\mu}(\boldsymbol{\theta})]_i}{\partial \theta_3} = \frac{1}{r_i}. \tag{2.21c}$$

As the result, the CRB for the deterministic signal is defined as

$$CRB(\boldsymbol{\theta}) = \begin{bmatrix} \sum_{i=1}^{M} \frac{\sigma_s^2(x_i-x_s)^2}{\sigma_\eta^2 r_i^6} & \sum_{i=1}^{M} \frac{\sigma_s^2(x_i-x_s)(y_i-y_s)}{\sigma_\eta^2 r_i^6} & \sum_{i=1}^{M} \frac{\sigma_s(x_i-x_s)^2}{\sigma_\eta^2 r_i^4} \\ \sum_{i=1}^{M} \frac{\sigma_s^2(x_i-x_s)(y_i-y_s)}{\sigma_\eta^2 r_i^6} & \sum_{i=1}^{M} \frac{\sigma_s^2(y_i-y_s)^2}{\sigma_\eta^2 r_i^6} & \sum_{i=1}^{M} \frac{\sigma_s(y_i-y_s)^2}{\sigma_\eta^2 r_i^4} \\ \sum_{i=1}^{M} \frac{\sigma_s(x_i-x_s)^2}{\sigma_\eta^2 r_i^4} & \sum_{i=1}^{M} \frac{\sigma_s(y_i-y_s)^2}{\sigma_\eta^2 r_i^4} & \sum_{i=1}^{M} \frac{1}{\sigma_\eta^2 r_i^2} \end{bmatrix}^{-1}. \tag{2.22}$$

2.2.2 CRB Simulations & Analysis

In this section, we analyze the theoretical localization performance via the CRB's for both signal models. Even for the simplified signal models assumed in Sec. 2.1.2, explicit close-form solutions for the CRB are not readily available (especially for the stochastic case), for example see [45]. Therefore, we perform simulations with L data snapshots to analyze the CRB's for the source localization. We simulate a single static source located at the center of a circle with radius R with N sensor nodes uniformly distributed in the circle. We obtain the MSE results (in terms of range squared in dB m^2) by averaging over $MC = 400$ independently drawn sensor lay-outs. We analyze the CRB on the MSE as a function of: (i) the number of nodes N and also the average number of detecting sensor nodes, M_{ave} (which depends on N, σ_s^2 and the detection threshold, σ_T^2), (ii) SNR, and (iii) σ_T^2.

In the first simulation example, we set $L = 1000$ and we vary the sensor density D by fixing $R = 100$ m and varying N. Fig. 2.4 shows the CRB on the MSE for SNR $= \left[40,\, 60,\, 80\right]$ dB versus M_{ave} with $\sigma_T^2 = 0$ dB. As expected, the MSE is lower for higher SNR levels. For the range of SNR values, the knees of the curves suggest using an ad-hoc network of 5–20 sensors can produce acceptable localization performance (*i.e.*, range error of 10 m or less). Not surprisingly, the performance gaps (among the CRB curves for $M_{\text{ave}} \geq 10$ (*i.e.*, the MSE differences in performance) correspond to the differences in the SNR levels and the MSE curves monotonically decrease with increasing sensor density (M_{ave}). The plot suggests a signal processing gain of approximately 6 dB for doubling the number of sensors.

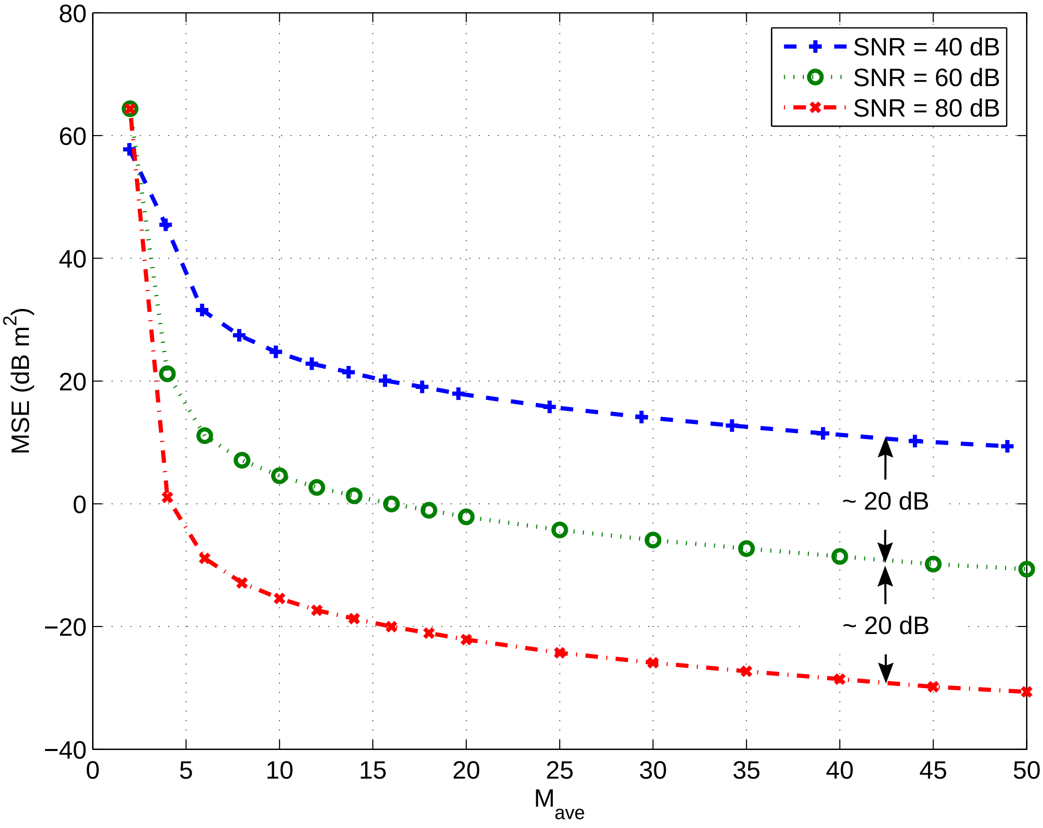


Figure 2.4: CRB analysis: MSE vs. M_{ave} for SNR $= [40,\, 60,\, 80]$ dB.

In the next simulation example, we fix N and vary the SNR levels with same parameter values for L, R and σ_T^2 as above. Fig. 2.5 shows the CRB on the MSE for fixed sensor densities of $N = [10, 20, 40]$ nodes versus SNR. Depending upon the locations of the sensor nodes relative to the location of the source, at low SNR's, $M_{\text{ave}} \lessapprox N$ and at high SNR's, $M_{\text{ave}} \approxeq N$. As expected, the MSE is lower for higher sensor density. In this example, for SNR ≥ 40 dB, the MSE decreases linearly with SNR with slope $\approxeq 1$ and the performance gaps among the density curves are approximately 6 dB's corresponding to the differences in sensor density in dB's.

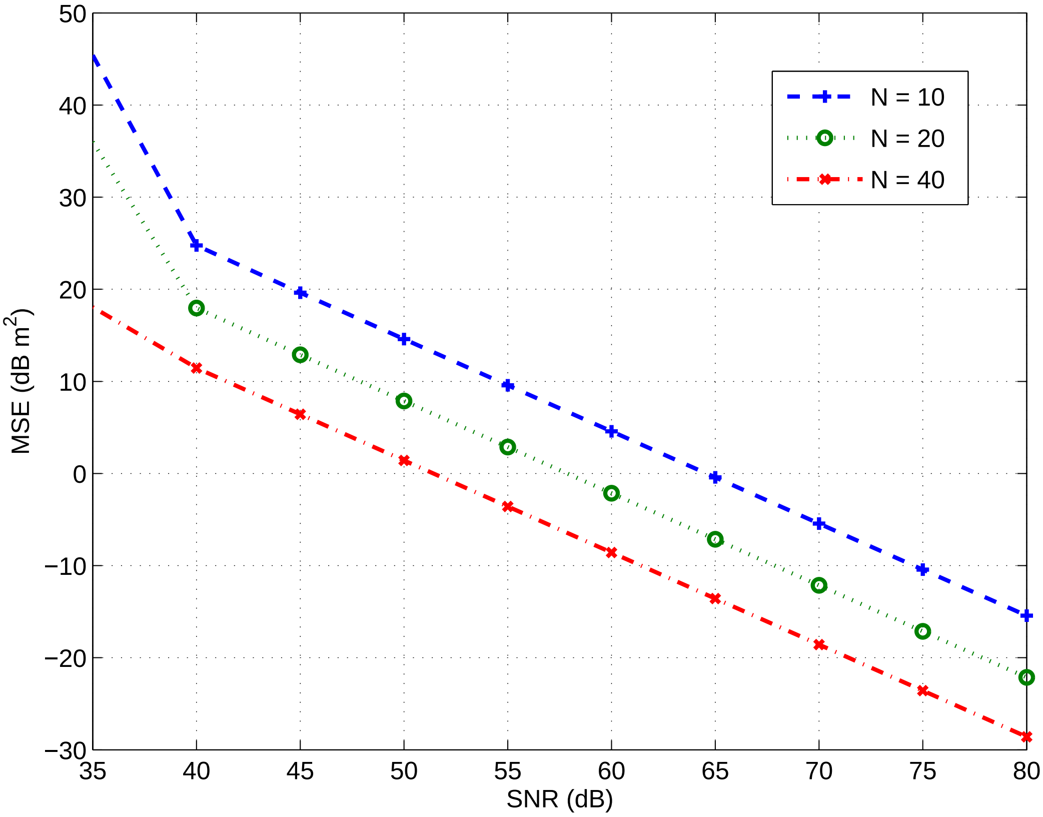


Figure 2.5: CRB analysis: MSE vs. SNR for $N = [10, 20, 40]$ sensor nodes.

We next analyze CRB for the case where σ_s^2 is assumed to be known, σ_{kn}^2, and compare its results to the case where for the σ_s^2 is unknown, σ_{un}^2 with the same parameter values for L, R and σ_T^2 as in the first example.[4] Fig. 2.6 shows the CRB comparisons for σ_{kn}^2 and σ_{un}^2 with respect to SNR and M_{ave}. As the figure reveals, there are slight performance improvements at lower M_{ave}'s and SNR's for σ_{kn}^2 over σ_{un}^2. At higher M_{ave}'s and SNR's, the MSE performances are nearly the same.

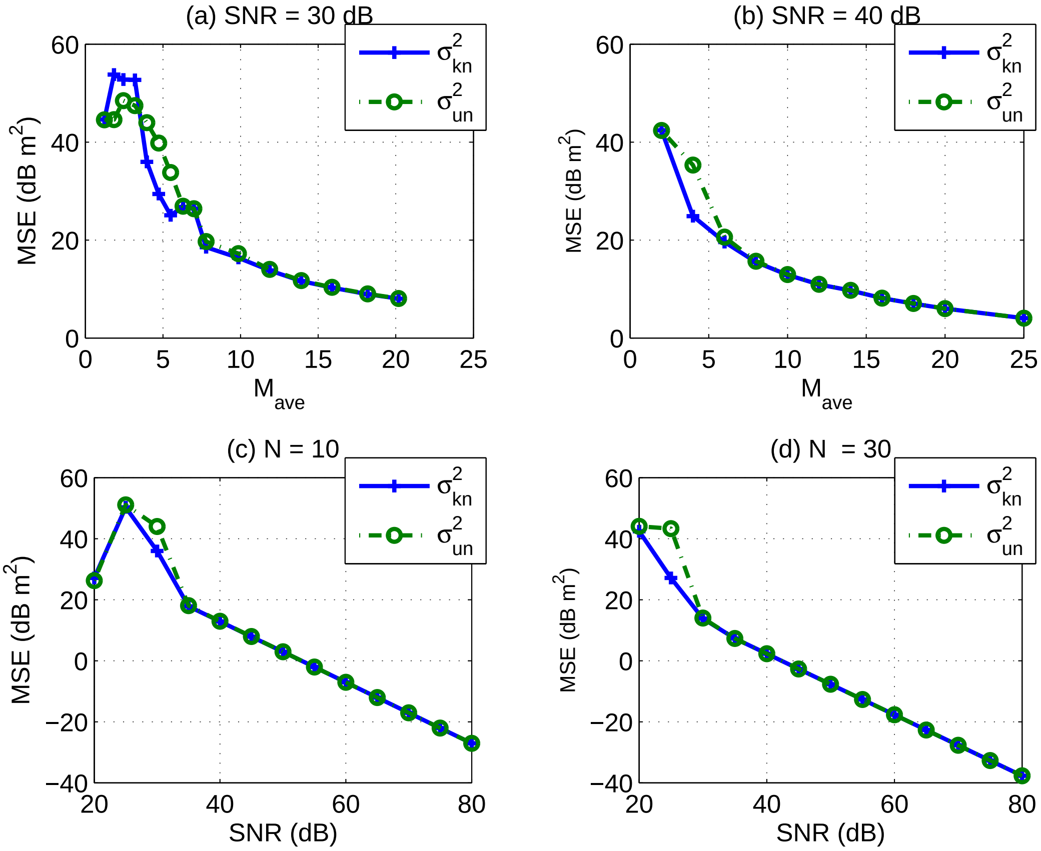


Figure 2.6: CRB comparisons for σ_{kn}^2 and σ_{un}^2: MSE vs. M_{ave} for (a) SNR = 30 dB and (b) SNR = 40 dB; and MSE vs. SNR for (c) $N = 10$ sensors and (d) $N = 30$ sensors.

In the next example, we assume a deterministic signal model with $L = 1$ snapshot, $N = 400$ nodes, $R = 200$ m. Fig. 2.7 shows the CRB on the MSE vs.

[4] Some source localization algorithms require σ_s^2 to be known [6] or to be eliminated from via ratio of measurements from pair of nodes [46].

SNR and σ_T^2. Similar to the stochastic case, the MSE decreases linearly with SNR (for SNR $\geq$ 40 dB) as shown in Fig. 2.7 (a) for $\sigma_T^2 = 10$ dB. In varying σ_T^2 with SNR $= 60$ dB, we see a nonlinear relationship between the MSE and σ_T^2, *i.e.*, the MSE stays nearly constant for $0 \leq \sigma_T^2 \leq 30$ dB and suddenly jumps for $\sigma_T^2 > 30$ dB, as shown in Fig. 2.7 (b). This is due to the fact that a high detection threshold limits the number of participating (detecting) nodes in the AHSN.

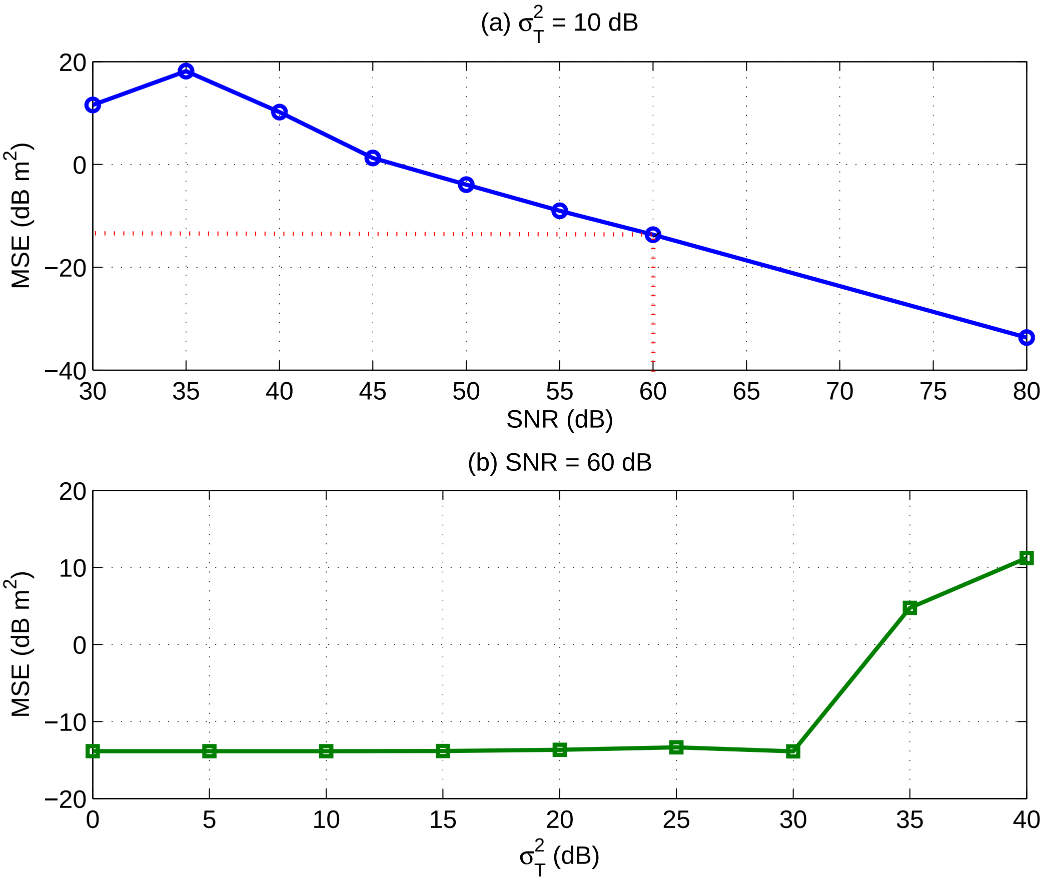


Figure 2.7: CRB analysis: (a) MSE vs. SNR with $\sigma_T^2 = 10$ dB and (b) MSE vs. σ_T^2 with SNR $= 60$ dB for the deterministic acoustic signal model.

Strategies for Reducing Network Complexity

We next explore strategies for keeping network complexity at manageable levels and limiting communication and energy costs by selecting a subset of detecting nodes to perform localization. For example in [47, 48], in the context of target tracking using DOA estimates, the authors proposed simplex methods for selecting subsets of detecting nodes to optimize tracking results while conserving energy.

In this section, we investigate the approach of using only K out of the M (with $K < M$) detecting sensor nodes for source localization in AHSN. The viability of such methods is suggested by Fig. 2.4, as the CRB-based performance gains by including more sensors are limited beyond 10 to 20 sensors. It is also important to select the subset of K participating nodes judiciously and efficiently. Fig. 2.8 shows the CRB on the MSE for several M-choose-K cases for the stochastic signal model ($L = 1000$), $M = N = 15$ sensor nodes, SNR $= 60$ dB, $\sigma_T^2 = 0$ dB , and $R = 100$ m over $MC = 200$ sensor layouts. The "best" and "worst" cases in the figure correspond to choosing the K-sensor configurations out of all possible M-choose-K configurations that give the lowest and highest CRB values, respectively. The "random" case corresponds to randomly choosing a K-sensor configuration from all possible M-choose-K configurations, while the "average" case corresponds to averaging the MSE's of all possible M-choose-K sensor configurations. The "loudest" case corresponds to choosing the K sensors with the highest RSS (*i.e.*, the K closest sensors to the source). As the figure reveals, the curves for the "best" and "worst" cases provide lower and upper bounds on M-choose-K performance. For all the schemes, at low sensor density (*e.g.*, $K < 4$) the spatial distribution of the sensor locations with respect to the source location is key in performance accuracy; however, the ad-hoc network tends to gain spatial diversity with increasing sensor density. In this example for $M = N = 15$ and

$K \geq 5$, the loudest scheme performs well with respect to the best scheme, suggesting the viability of RSS-based techniques for limiting the number of sensors participating in the computation. Alternatively, for $M = 15$ and $K \geq 10$, the "random" case performs within a few dB of the "best" case.

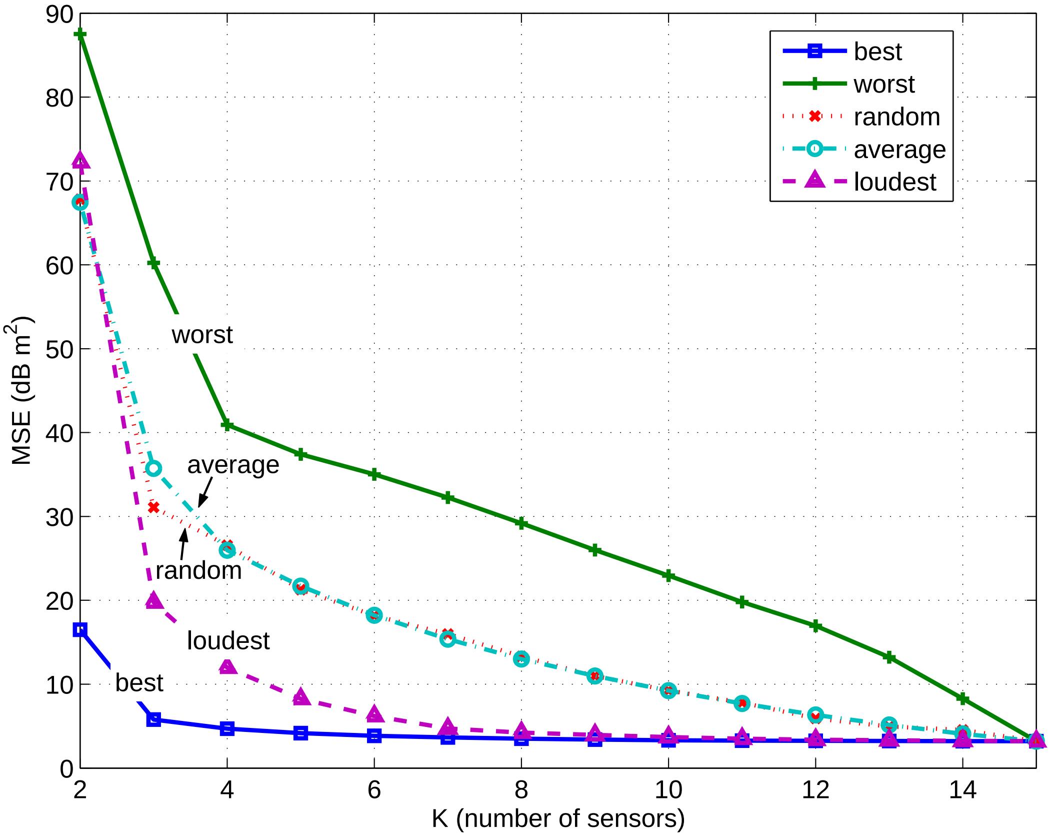


Figure 2.8: CRB analysis: MSE vs. K sensors for various M-choose-K cases: "best", "worst", "random", "average" and "loudest".

2.2.3 CRB Summary

In summary, we developed the CRB's for the stochastic and deterministic acoustic signal models with simplified STF. We demonstrated the theoretical source localization performance as functions of SNR, sensor density D (via M_{ave} and N) and detection threshold, σ_T^2. For medium to high SNR levels (*e.g.*, SNR $\geq$ 30 dB in the simulation examples), the CRB on the MSE decreases linearly with increasing SNR with slope $\approxeq$ 1. Similarly, for sensor density greater than 10–20 nodes, the CRB on the MSE decreases linearly with increasing SNR with signal processing gain of approximately 6 dB for doubling of sensors. Finally, we investigated strategies for limiting the number of sensors participating in source localization via the CRB. We found that using a scheme involving a small number of sensors that are closest to the source performs very well suggesting the viability of RSS-based source localization schemes. Lastly, we found that both signal models yield similar CRB on MSE trends.

Chapter 3

Distributed Computations

As discussed in the introduction, decentralized algorithm frameworks are becoming increasingly more attractive than their centralized counterparts for space-time processing in large-scale ad-hoc networks. A decentralized data fusion system typically consists of sensor nodes with their own processing capabilities, and fusion and data processing occur locally at each node based on local observations and information communicated from neighboring nodes. Thus, a decentralized system is characterized by three constraints: (i) there is no single central fusion center; (ii) there is no common communication center, and only node-to-node, not broadcast, communications; (iii) sensor nodes have only local, not global, knowledge of network topology. However, these imposed constraints provide a number of important characteristics for decentralized data fusion systems: (i) the system is scalable, since nodes only communicate with neighboring nodes; (ii) the system can be made survivable to loss or addition of nodes and to dynamic changes in the network structure, *e.g.*, mobile sensor nodes; and (iii) sensor nodes can be designed, constructed, and programmed in a modular fashion [15, 49–51].

For decentralized ad-hoc networks, it is desirable to develop distributed routing and fusion algorithms that are scalable, fault tolerant, and robust to changing network topology. Indeed, distributed processing has received attention in the early 1980's via Tsitsiklis's work on methods for reaching agreement and consensus [52]. However, recently, interests in distributed processing and computations have grown tremendously due to the omnipresence of sensor networks [17,53] and interests in developing computational models and distributed agents (which are small, autonomous, self-describing programs) based on biological and social networks [54,55]. Many of these approaches have in common a set of basic distributed computational tasks or modules in which more complex tasks or systems can be built upon.

In this chapter, we first present network models with topologies whereby each node in the network is assumed to establish noise-free bidirectional communications and routing with its neighboring nodes. We present models for describing the topologies of the overall network and the sequence of subnetworks of detecting nodes that track the movement of the source in Sec. 3.1. Then, we present the basic distributed computational algorithm modules which then can be used to develop distributed sensor fusion algorithms for AHSN's. More specifically, we present distributed algorithms for computation of averages that can be used for performing source localization [17–19], and then we present improved versions of distributed algorithms that can be used in a broad-class of problems for performing localization and tracking [3,20] in Sec. 3.2. Next, we present a more general version of the distributed computation algorithm for computing weighted averages in Sec. 3.3. We conclude with a brief simulation-based analysis via a fictitious example of a signal estimation in noisy observations by a AHSN in Sec. 3.4 .

3.1 Network Model

In this section, we present and discuss network models that have the following properties: (i) efficient use of transmit power, (ii) each sensor node receives messages sent by connected neighboring nodes, and (iii) each sensor node broadcasts messages to connected nodes. We consider large-scale networks of uniformly distributed sensors as in Sec. 2.1.5. We focus on bidirectional network topologies according to which, each node is assumed to establish noise-free bidirectional communication with a subset of nodes in its proximity as illustrated in Fig. (3.1). Letting N denote the total number of nodes in the network at a fixed time t, the network topology is described by an $N \times N$ matrix Φ, where $\phi_{ij} = [\Phi]_{ij}$ denotes the connection status of the link between nodes $i \neq j$, defined as

$$\phi_{ij} = \phi_{ji} = \begin{cases} 1 & \text{if } i \leftrightarrow j \\ 0 & \text{otherwise} \end{cases} \tag{3.1a}$$

and where $i \leftrightarrow j$ denotes that nodes i and j are bidirectionally connected. We also let for convenience

$$\phi_{ii} = -\sum_{j \neq i}^{N} \phi_{ij} \,. \tag{3.1b}$$

The connection status ϕ_{ij} of any two nodes i and j is modeled as a probabilistic function of d_{ij}, the distance between nodes i and j, and is given by

$$\Pr[\phi_{ij} = 1] = 2^{-(\frac{d_{ij}}{d_o})^m} , \tag{3.2}$$

where d_o denotes the nominal distance at which nodes i and j are connected with probability $\frac{1}{2}$, and where the parameter m determines the rate of decay of probability of connection with distance. The probabilistic connection model described in (3.2)

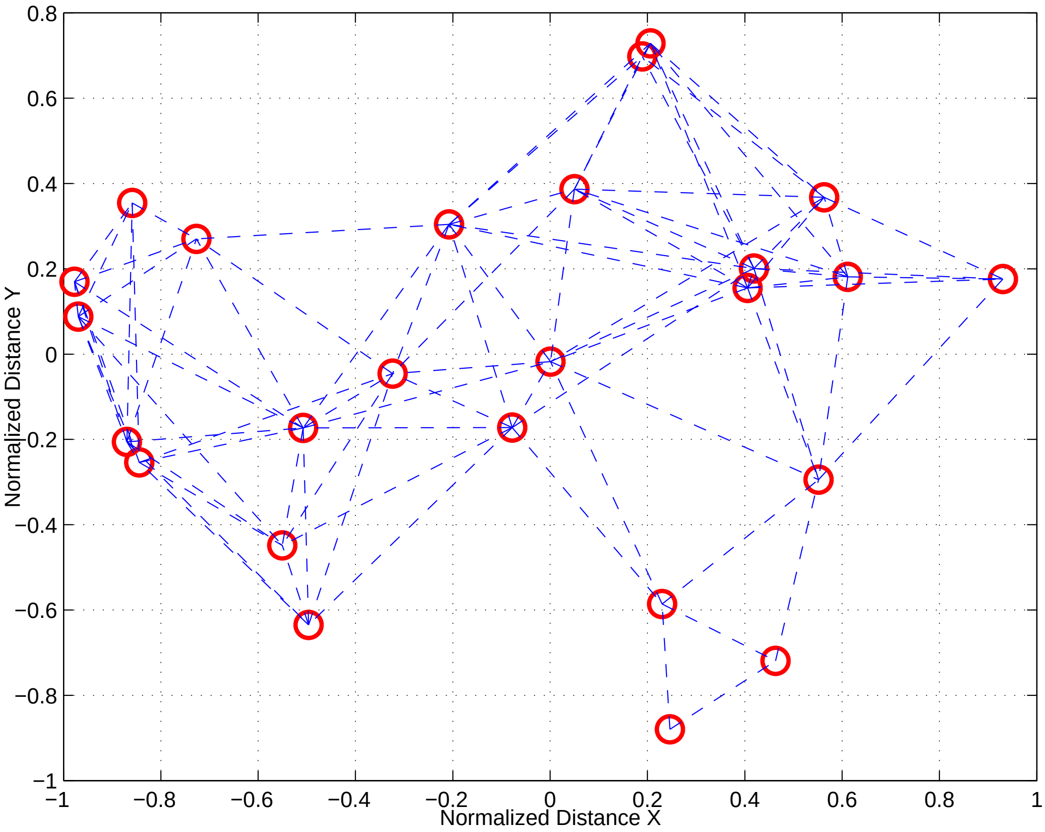


Figure 3.1: An example of network connectivity for network $N = 25$ sensor nodes.

captures the connectivity trends in the context of sensor communication for nodes i and j over a variety of Rayleigh fading channels over which the average power loss is of the form

$$\mathrm{P_R} \propto \mathrm{P_T}\, d_{ij}^{-m} \tag{3.3}$$

where $\mathrm{P_R}$ and $\mathrm{P_T}$ denote the receive and transmit power respectively and m takes on values in the range $2 \leq m \leq 4$ [56]. We remark that the connectivity model in (3.2) has several desirable properties. First, $\Pr[\phi_{ij} = 1]$ is a decreasing function of d_{ij}, *i.e.*, nodes that are close to each other are more likely to be connected. Also, as $m \to \infty$, $\Pr[\phi_{ij} = 1] \to 1$ for $d_{ij} < d_o$ while $\Pr[\phi_{ij} = 1] \to 0$ for $d_{ij} > d_o$.

In the following, we present alternative formulations of network and connectivity models for performing sensor fusion in time in Sec. 3.1.1. Then, we present some key properties of the matrix Φ in Sec. 3.1.2 that help facilitate the design of distributed computation algorithms.

3.1.1 Network Model Formulations for Fusion in Time

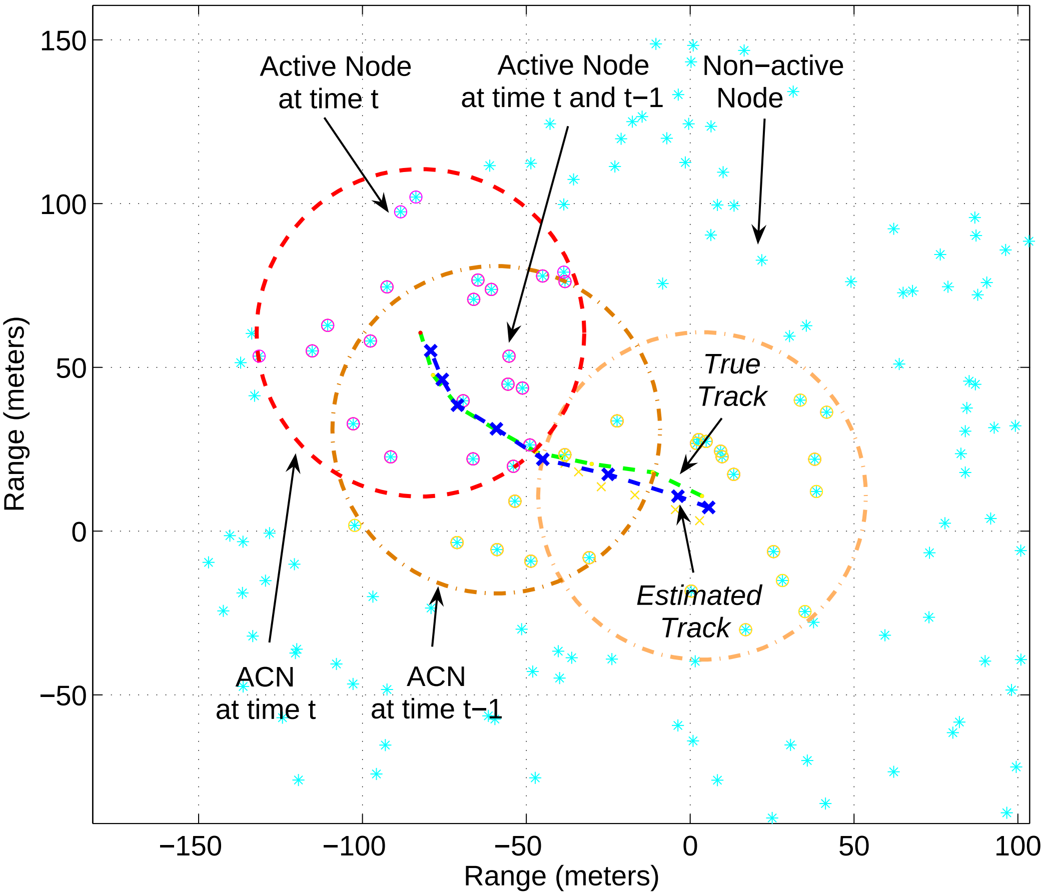


Figure 3.2: Source tracking in a large-scale sensor network via computations over a sequence of subnetworks formed by the detecting nodes in the vicinity of the source.

Although, the network and connectivity models we describe above are in the context of spatial sensor fusion, *e.g.*, source localization, they canbe extended to space-time sensor fusion, *e.g.*, tracking. In particular, we can view tracking as a successive sequence of source localization over suitably chosen sequences of subnetworks as shown in Fig. 1.2. To this end, we let $\mathcal{I}(t)$ denote the set of nodes that comprise the ACN employed at time t, *i.e.*, the subnetwork over which the tth tracking estimate is to be computed. Here we focus on the simplest case where the set of nodes in the computation network at time t coincides with the subset of the nodes that detect the source at time t. In general, however, the computation network may also include additional peripheral (non-detecting) nodes to assist in the routing and computation.

The network topology of the ACN at time t can also be expressed in terms of an $N \times N$ matrix $\tilde{\Phi}(t)$ where the (i, j)th element of $\tilde{\Phi}(t)$, for $i \neq j$, is given by

$$\tilde{\phi}_{ij}(t) = \tilde{\phi}_{ji}(t) = \begin{cases} \phi_{ij} & \text{if}\, i, j \in \mathcal{I}(t) \\ 0 & \text{otherwise} \end{cases} \tag{3.4a}$$

while again, for convenience, we set

$$\tilde{\phi}_{ii}(t) = -\sum_{j \neq i} \tilde{\phi}_{ij}(t)\,. \tag{3.4b}$$

Alternatively, letting $M(t) = |\mathcal{I}(t)|$ denotes the number of nodes that have detected the source at time t and $\{\mathcal{I}(t)\}_i$ denote the ith element in $\mathcal{I}(t)$, the ACN network topology at time t can be alternatively described via $\mathcal{I}(t)$, and an $M(t) \times M(t)$ matrix $\Psi(t)$ where the (i, j)th element of $\Psi(t)$, for $i \neq j$, is defined as

$$\psi_{ij}(t) = \psi_{ji}(t) = \phi_{\{\mathcal{I}(t)\}_i \, \{\mathcal{I}(t)\}_j} \tag{3.5a}$$

while again, for convenience, we set

$$\psi_{ii}(t) = -\sum_{j \neq i} \psi_{ij}(t)\,. \tag{3.5b}$$

We can think of the network described by $\tilde{\Phi}(t)$ in (3.4a) as a large-scale network where the only available connections for performing computations are among the set of nodes in $\mathcal{I}(t)$. Alternatively, we can focus on the ACN formed by the nodes in $\mathcal{I}(t)$, with network topology given by $\Psi(t)$ in (3.5a).

3.1.2 Properties of Network Topology Matrix Φ

In this section, we present some interesting and important properties of the topology matrix $\Phi(t)$ that provide the basis for development of the distributed computation algorithms over connected networks. For convenience, we drop the dependence on t. Recall $\phi_{ij} = [\Phi]_{ij}$ and $\phi_{ij} = \phi_{ji}$; therefore, the matrix Φ is a symmetric matrix. In addition, Φ is negative semi-definite. In particular, let

$$\Phi = V_\Phi\, \Lambda_\Phi\, V_\Phi^T \tag{3.6}$$

denotes the eigen-decomposition of Φ, where

$$V_\Phi = \big[\mathbf{v}_1(\Phi)\, \mathbf{v}_2(\Phi)\, \cdots\, \mathbf{v}_N(\Phi)\big]$$

is a unitary matrix comprised of unitary vectors $\{\mathbf{v}_i(\Phi)\}_{i=1}^N$ and

$$\Lambda_\Phi = \mathrm{diag}\big(\lambda_1(\Phi),\, \lambda_2(\Phi),\, \cdots,\, \lambda_N(\Phi)\big)$$

is a diagonal matrix of eigenvalues $\{\lambda_i(\Phi)\}_{i=1}^N$. By using Gersgorin's theorem ([57], pp. 344–348), we have $\lambda_i(\Phi) \leq 0, \forall i$, where to get equality $\lambda_i(\Phi) = 0$ for some i requires that each element of the associated unitary vector $\mathbf{v}_i(\Phi)$ be either 0, or $\tilde{c} \neq 0$, for some constant $\tilde{c}$, independent of the element index (*e.g.*, $\tilde{c} = 1/N$).[1] Without loss of generality, let $\lambda_1(\Phi) \geq \lambda_2(\Phi) \geq \cdots \geq \lambda_N(\Phi)$. Furthermore, due to (3.1), we have $\Phi \cdot \mathbf{1} = \mathbf{0}$ where $\mathbf{1}$ and $\mathbf{0}$ denote $N \times 1$ vectors of ones and zeros respectively; consequently, we have $\lambda_1(\Phi) = 0$ [17].

In our investigation, we mainly focus on *connected topologies*, *i.e.*, topologies for which there exists a multi-hop communication path between every pair of nodes in the sensor network. It is clear from (3.1) that for connected topologies, $|\phi_{ii}| > 0, \forall i$. We can also show that Φ describes a connected topology if and only if $\lambda_i(\Phi) < 0, \forall i \geq 2$. In particular, if an eigenvalue $\lambda_i(\Phi)$ other than $\lambda_1(\Phi)$ is zero, it would have to be associated with the unitary vector $\mathbf{v}_i(\Phi)$ that has one or more (but not all) elements equal to zero. Letting E_i denote the subset of indices of the non-zero and equal elements of $\mathbf{v}_i(\Phi)$, for every $k \in E_i$, we must have $\sum_{j \in E_i} \phi_{kj} = 0$, implying that the subset E_i of the nodes form a subnetwork that is *disconnected* from the rest of the (overall) network. As a result, Φ is a connected topology if and only if $\lambda_i(\Phi) < 0, \forall i \geq 2$ [17].

Further details on the topology matrix Φ and its properties with respect to distributed implementations can be found in [17, 18].

[1] We use Gersgorin's theorem to prove a convergence theorem for first-order linear LTI rules in Sec. 3.2.2.

3.2 Distributed Computation of Averages

To motivate the construction of distributed algorithms for performing sensor data fusion in space and in time, we present the (scalar) distributed algorithms for computing any such elementary averaging target computation [17,18]. Then, we show how these elementary distributed algorithms can be used to perform more complex (vector) distributed tasks such as source localization and tracking [3, 19]. For convenience, we omit the dependence on t.

To this end, let the target scalar computation for an AHSN of N sensor nodes be given by

$$G(\mathbf{f}(\mathbf{x})) = \frac{1}{N}\sum_{i=1}^{N} f_i(x_i) \tag{3.7}$$

where $\mathbf{x} = \begin{bmatrix} x_1 \; x_2 \; \cdots \; x_N \end{bmatrix}^T$, with x_i denoting the scalar data observation at the ith node and $\mathbf{f} = \begin{bmatrix} f_1(\cdot) \; f_2(\cdot) \; \cdots \; f_N(\cdot) \end{bmatrix}^T$, with $f_i(\cdot)$ denoting an arbitrary local scalar-valued function at the ith node. We are interested in fusion rules that are iterative, locally-constructed rules that generate at each node i a sequence of state approximations $f_i(x_i[k])$ to the desired (global) computation $G(\mathbf{f}(\mathbf{x}))$ by exploiting states broadcasted by nodes in direct bidirectional communication with the ith node. For convenience, we simplify the notation by letting $G(\mathbf{f}) = G(\mathbf{f}(\mathbf{x}))$ and (3.7) becomes

$$G(\mathbf{f}) = \frac{1}{N}\sum_{i=1}^{N} f_i \tag{3.8}$$

where $f_i = f_i(x_i)$ and $f_i[k] = f_i(x_i[k])$ are *observation* and sequence of state approximations respectively at the ith node.

We next present several classes of linear time-invariant (LTI) rules implemented over a given connected topology Φ.

3.2.1 Class of Admissible, LTI and Asymptotically Converging Rules

In this section, we present a class of fusion rules that are implemented over a given topology described by $N \times N$ topology matrix Φ that generate at the ith node a sequence of state approximation the desired *scalar* computation $G(\mathbf{f})$. We are interested in distributed fusions that can be described by the following definition [17, 18]:

Definition 1: *Let $\mathcal{U}_i$ denote the set of nodes that have a direct bidirectional communications link with the ith node, i.e.*

$$\mathcal{U}_i \triangleq \big\{j \in \{1, 2, \cdots, N\};\ \phi_{ij} \neq 0\big\}. \tag{3.9}$$

Then, a set of rules $\big\{F_i^{(k)}\big\}_{i=1}^{N}$, $k > 0$, will be referred to as an ***admissible*** *distributed rule with respect to a given topology Φ if*

$$f_i[k+1] = F_i^{(k)}\left(f_i,\ \{f_j[l];\ l \leq k,\ j \in \mathcal{U}_i(\Phi) \cup \{i\}\}\right). \tag{3.10}$$

In particular, we focus on admissible LTI rules of the form

$$\mathbf{f}[k] = \sum_{l \geq 1} W[l]\,\mathbf{f}[k-l], \quad k > 0, \tag{3.11}$$

where $\mathbf{f}[k] = \big[f_1[k]\ f_2[k]\ \cdots\ f_N[k]\big]^T$, and $W[l]$ is an $N \times N$ admissible matrix kernel.[2] Lets define the (i, j)th element of matrix sequence $W[l]$ be $W_{ij}[l] \triangleq \{W[l]\}_{ij}$. Then,

[2] This admissible LTI rule is a subclass of admissible linear rules of the form $\mathbf{f}[k] = \sum_{l \geq 1} W[k;l]\,\mathbf{f}[k-l]$, $k > 0$, where $W[k;l]$ is an $N \times N$ admissible matrix kernel. For LTI, $W[k;l] = W[1;l] \triangleq W[l]$.

admissibility of the rule defined in (3.11) in the sense of definition in (3.10) requires that $W_{ij}[l] = 0$, for $l \leq 0$, or if $\phi_{ij} = 0$.

In addition to admissibility and LTI, we are interested in a class of admissible LTI rules that asymptotically compute desired functions such as $G(\mathbf{f})$.

Definition 2: *An admissible rule over a given topology Φ is **asymptotically converging** (AC) to the desired scalar function $G(\mathbf{f})$ if the sequence $\mathbf{f}[k]$ satisfies*

$$\lim_{k \to \infty} \|\mathbf{f}[k] - \mathbf{1} \cdot G(\mathbf{f})\| = 0, \tag{3.12}$$

where $\|\cdot\|$ is the Euclidean norm, and $\mathbf{1}$ is an $N \times 1$ vector of 1's.

Furthermore, we are interested in a simpler but very useful subclass of LTI rules that are admissible and AC, namely first-order LTI rules which will be described next.

3.2.2 First-Order LTI Rules

In this section, we consider first-order admissible LTI rules, the admissible matrix kernel $W[l] = W\,\delta[l-1]$, where W is an $N \times N$ admissible matrix, *i.e.*, satisfying $W_{ij} = 0$ for $\phi_{ij} = 0$, and (3.11) reduces to

$$\mathbf{f}[k] = W\,\mathbf{f}[k-1] \;\text{ for }\; k > 0\,. \tag{3.13a}$$

We consider the initialization of the recursion rule (3.13) via

$$\mathbf{f}[k] = \mathbf{f} \quad \text{for }\; k \leq 0\,, \tag{3.13b}$$

where $\mathbf{f} = \left[f_1\, f_2\, \cdots\, f_N\right]^T$. This initialization is admissible according to (3.10) since it is just setting $f_i[k] = f_i$, $\forall k < 0$, for $i = 1, 2, \cdots, N$.

In the following theorem and associated proof, we demonstrate the convergence for the class of first-order LTI rules of interest with some basic network conditions.

Theorem 1. *Given an $N \times N$ network topology Φ, consider the following rule*

$$f_i[k] = \sum_{j=1}^{N} \rho_{ij}\, f_j[k-1]\, , k > 0 \tag{3.14}$$

initialized with $f_i[0] = f_i$, for $i = 1, 2 \cdots, N$. Assume the rule is admissible on Φ, that is, for any $i \neq j$ such that $\phi_{ij} = 0$, we have $\rho_{ij} = 0$. Assume also the following:

$$\Phi \;\; \textit{is a bidirectional topology} \tag{3.15a}$$

$$\Phi \;\; \textit{is a connected topology} \tag{3.15b}$$

$$\rho_{ij} = \rho_{ji},\ \forall(i,j) \;\; \textit{(reciprocity)} \tag{3.15c}$$

$$\sum_{j=1}^{N} \rho_{ij} = 1,\ \forall i \;\; \textit{(balancing)} \tag{3.15d}$$

$$\rho_{ij} > 0,\ \forall(i,j) \;\; \textit{such that,} \;\; i = j, \;\; \textit{or} \;\; \phi_{ij} = 1 \tag{3.15e}$$

$$\sum_{j=1, j\neq i}^{N} \rho_{ij} < 1. \tag{3.15f}$$

Then

$$\lim_{k\to\infty} f_i[k] = \frac{1}{N}\sum_{j=1}^{N} f_j,\ \forall i. \tag{3.16}$$

We remark that although condition (3.15f) is implied by conditions (3.15d)–(3.15e), it is included for convenience. We also remark the use of the following notations for proving Thm. 1. Given any $N \times N$ real symmetric matrix $D = D^T$ we denote its eigen-decomposition as follows:

$$D = V_D \, \Lambda_D V_D^T \tag{3.17}$$

where $V_D = \left[\mathbf{v}_1(D)\ \mathbf{v}_2(D)\ \cdots\ \mathbf{v}_N(D)\right]$ is the (unitary) eigenvector matrix, and Λ_D is a diagonal eigenvalue matrix with real diagonal entries $\{\lambda_i(D)\}$. Unless stated otherwise, we will assume that the eigenvalues of D are in decreasing order, *i.e.*,

$$\lambda_1(D) \geq \lambda_2(D) \geq \cdots \geq \lambda_N(D) \ . \tag{3.18}$$

Proof of Theorem 1: We recall a few of the properties of Φ derived in Sec. 3.1.2 that are relevant to the following proof of the theorem. First, due to condition (3.15a), Φ has a decomposition of the form (3.6) with $\lambda_i(\Phi) \leq 0$ for all i. In addition, assuming that the eigenvalues of W are ordered as in (3.18), we have: $\lambda_1(\Phi) = 0$ with $\mathbf{v}_1(\Phi) = \mathbf{1}/\sqrt{N}$. Furthermore, due to condition (3.15b), $\lambda_i(\Phi) < 0$ for all $i \geq 2$.

Given $\mathbf{f}[k] = \left[f_1[k]\ f_2[k]\ \cdots\ f_N[k]\right]^T$, we can write

$$\mathbf{f}[k] = W \, \mathbf{f}[k-1]$$

where $[W]_{ij} = \rho_{ij}$. Due to condition (3.15c), $W = W^T$, implying that W has a eigen-decomposition of the form (3.17) with real eigenvalues. We also note that, due to condition (3.15d), $W \cdot \mathbf{1} = \mathbf{1}$ (where $\mathbf{1}$ denotes an $N \times 1$ vector of one's) implying that

$$\mathbf{v}_1(W) = \frac{\mathbf{1}}{\sqrt{N}}, \text{ and } \lambda_1(W) = 1 \ . \tag{3.19}$$

Proving the theorem amounts to showing that

$$|\lambda_i(W)| < 1,\ \forall i \geq 2 \tag{3.20}$$

since, together with (3.19), it implies that $f_i[k] \rightarrow \sum_j f_j/N$ as $k \rightarrow \infty$, for all i.

Let $U = W - I$. This matrix has an eigen-decomposition of the form (3.17), where

$$V_U = V_W \text{ and } \lambda_i(U) = \lambda_i(W) - 1. \tag{3.21}$$

As a result, proving (3.20) is equivalent to proving

$$-2 < \lambda_i(U) < 0, \text{ for all } i \geq 2\ . \tag{3.22}$$

Let $R_i^U = \sum_{j,\, j \neq i} u_{ij}$. Then $R_i^U = \sum_{j,\, j \neq i} \rho_{ij} < 1$ by condition (3.15f) and where we also used the fact that $u_{ij} = w_{ij} = \rho_{ij}$ for all $j \neq i$. Furthermore, using condition (3.15d)

$$u_{ii} = \rho_{ii} - 1 = -R_i^U\ .$$

Applying Gersgorin's Theorem (Thm 6.1.1, pp. 344–345, in [57]) on U, reveals that all the eigenvalues of U must be in the union of the following N disks $|\lambda(U) - u_{ii}| \leq R_i^U$, or, equivalently, $0 \geq \lambda(U) \geq -2R_i^U$, which due to condition (3.15f) implies that

$$-2 < \lambda_i(U) \leq 0, \text{ for all } i\ . \tag{3.23}$$

From (3.21), we have $\lambda_1(U) = \lambda_1(W) - 1 = 0$. To complete the proof of the validity (3.22) using (3.23) we simply need to show that $\lambda_i(U) < 0$ for all $i \geq 2$. To this end, let $\mathcal{A} = \{(i,j); i \neq j, \phi_{ij} = 1\}$, and define

$$\rho_{\min} = \min_{(i,j)\in\mathcal{A}} \rho_{ij} \ .$$

From condition (3.15e) we have $\rho_{\min} > 0$. Define $A = \rho_{\min}\Phi$, and $B = U - A$. Then, $A = A^T$ and $B = B^T$, and thus both A and B have decompositions of the form (3.17). Furthermore, since $\lambda_i(A) = \rho_{\min}\lambda_i(\Phi)$, we have $\lambda_1(A) = 0$, $\mathbf{v}_1(A) = \mathbf{v}_1(\Phi) = \mathbf{1}/\sqrt{N}$, and $\lambda_i(A) < 0$ for all $i \geq 2$. Considering the matrix B, let $R_i^B = \sum_{j\neq i} b_{ij}$. We next show that for all $j \neq i$, $0 \leq b_{ij} \leq u_{ij}$, which implies

$$0 \leq R_i^B \leq R_i^U < 1 \ . \tag{3.24}$$

In particular, for any $j \neq i$, $\phi_{ij} \geq 0$, which together with $\rho_{\min} > 0$, shows that

$$b_{ij} = u_{ij} - a_{ij} = u_{ij} - \rho_{\min}\phi_{ij} \leq u_{ij} \ .$$

Similarly, we can show that for $j \neq i$, $b_{ij} \geq 0$, by separately considering $(i,\, j) \in \mathcal{A}$ and $(i,\, j) \notin \mathcal{A}$. In particular, for $(i,\, j) \notin \mathcal{A}$, $\phi_{ij} = 0$, and since W is an admissible rule, $\rho_{ij} = 0$. Thus $b_{ij} = u_{ij} - a_{ij} = \rho_{ij} - \rho_{\min}\phi_{ij} = 0$. For $(i,\, j) \in \mathcal{A}$, $\phi_{ij} = 1$, so $a_{ij} = \rho_{\min}$ and since $u_{ij} = \rho_{ij} \geq \rho_{\min}$, we have $b_{ij} = u_{ij} - a_{ij} \geq 0$. Finally, regarding the diagonal elements of B

$$\begin{aligned} b_{ii} &= u_{ii} - a_{ii} = -\sum_{j,\, j\neq i} \rho_{ij} - \rho_{\min}\phi_{ii} = -\sum_{j,\, (i,j)\in\mathcal{A}} \rho_{ij} - \rho_{\min} \\ &= \sum_{j,\, (i,j)\in\mathcal{A}} [\rho_{ij} - \rho_{\min}] = -\sum_{j,\, j\neq i} b_{ij} \\ &= -R_i^B \ . \end{aligned} \tag{3.25}$$

Thus, an application of Gersgorin's theorem on B, along the same lines for the one used for U, gives

$$-2 < -2R_i^B \leq \lambda_i(B) \leq 0, \text{ for all } i \tag{3.26}$$

Furthermore, since $B \cdot \mathbf{1} = \mathbf{0}$ we have $\lambda_1(B) = 0$ with $\mathbf{v}_1(B) = \mathbf{1}/\sqrt{N}$. In addition, assuming that the eigenvalues of A and B are listed in decreasing order according to (3.18), using the fact that $\lambda_1(B) = \lambda_1(A)$ and $\mathbf{v}_1(B) = \mathbf{v}_1(A)$, and applying Weyl's Theorem on $U = A + B$ (Thm. 4.3.1, pp. 181–182, in [57]) on the subspace of R^N that is orthogonal to the span of $\mathbf{v}_1(A) = \mathbf{v}_1(B) = \mathbf{1}/\sqrt{N}$, we get

$$\lambda_i(U) \leq \lambda_i(B) + \lambda_2(A), \ \forall i \geq 2 \tag{3.27}$$

Finally, using (3.27) and the fact that $\lambda_2(A) = \rho_{\min}\lambda_2(\Phi) < 0$ and $\lambda_i(B) \leq 0$, we get $\lambda_i(U) < 0$ for all $i \geq 2$ which completes the proof of (3.22) and the theorem [20].

To summarize, the key conditions that we used in the proof are *reciprocity* and *balancing*. Reciprocity corresponds to each pair of connected nodes using the same fraction of each other's state in their computation, while balancing corresponds to ensuring that the algebraic sum of all fractions used in adjusting the state of any particular node is zero. For the matrix W, the conditions corresponds to:

$$\textit{Reciprocity}: \quad W_{ij} = W_{ji}, \tag{3.28a}$$

with the condition $W_{ij} = W_{ji} = 0$ if $\phi_{ij} = 0$ and

$$\textit{Balancing}: \quad W_{ii} = 1 - \sum_{j \neq i} W_{ij}, \ \forall i. \tag{3.28b}$$

We next present a class of non-uniform diffusion rules (NUD) that is amenable to distributed implementations over Φ that leads to developing space-time fusion algorithms that exploit locally available information at the nodes in Sec. 3.2.3. Then, we present a more specific class of uniform diffusion (UD) rules that can either exploit locally available information at the nodes or the macroscopic information about network topology in Sec. 3.2.4.

3.2.3 First-Order Non-Uniform Diffusion LTI Rules

The first-order NUD LTI rules are described by (3.14) and (3.15) with NUD parameters ρ_{ij}'s.

We next present a locally negotiated algorithm that can adaptively choose the NUD parameters ρ_{ij}'s. The algorithm exploits the conditions in (3.15) and the condition $|\rho_{ii}| < 1$ that guarantee convergence.

Local Negotiation Algorithm

Choices for the ρ_{ij}'s that satisfy (3.15) can be made via local negotiations. In particular, the following iterative *local negotiation* (LN) algorithm yields sets of ρ_{ij}'s reported in [3, 20]. The algorithm yields sequences of improving sets of ρ_{ij}'s each of which satisfies (3.15). The algorithm achieves $|\rho_{ii}| < 1$ by guaranteeing that ρ_{ii} is at most $(1 - \epsilon)$ for some small $\epsilon > 0$. At the outset, the algorithm is initialized with $\rho_{ij}[0] = 0$, $\phi_i[0] = |\phi_{ii}|$, and $\Delta_i[0] = 1 - \epsilon$. Given an arbitrarily small $\epsilon > 0$, the kth step of the algorithm at the ith node, for any $k \geq 1$, takes the following form:

(i) set and broadcast:

$$\delta_i[k] = \begin{cases} \Delta_i[k-1]/\phi_i[k-1] & \text{if } \phi_i[k-1] > 0 \\ 0 & \text{otherwise;} \end{cases}$$

(ii) for all j, set

$$\rho_{ij}[k] = \begin{cases} \rho_{ij}[k-1] + \min\{\delta_i[k], \delta_j[k]\} & \text{if } i \leftrightarrow j \\ 0 & \text{otherwise;} \end{cases} \tag{3.29}$$

(preserving reciprocity)

(iii) set

$$\begin{aligned} \rho_{ii}[k] &= 1 - \sum_{j \neq i} \rho_{ij}[k], \text{ (balancing)} \\ \Delta_i[k] &= 1 - \epsilon - \sum_{j \neq i} \rho_{ij}[k] \\ \phi_i[k] &= \left|\{j;\, \phi_{ij} = 1,\, \delta_j[k] \neq 0\}\right|. \end{aligned}$$

It is straightforward to verify that this algorithm satisfies the conditions in (3.15) at every step k, and terminates after a finite number of iterations for any finite-size network, *i.e.*, $\delta_i[k] = 0, \forall i$ and $k > k_o$ for some finite k_o.

3.2.4 First-Order Uniform Diffusion LTI Rules

In this section, we present a class of UD rules that are used to develop locally-constructed algorithms.[3] For UD rules, the reciprocity and balancing conditions at

[3] These rules have strong connections to networks of coupled non-linear oscillators used to describe global synchronization phenomenology found in biological species [58].

the ith node become

$$\rho_{ij} = \rho_{ji} = \rho = \begin{cases} 1 & \text{if } \phi_{ij} = 1 \\ \\ 0 & \text{if } \phi_{ij} = 0 \end{cases} \qquad \text{for } \; i \neq j \tag{3.30a}$$

$$\rho_{ii} = 1 - \sum_{i \neq j} \rho_{ij} = 1 - \rho \sum_{i \neq j} \phi_{ij} = 1 + \rho\, \phi_{ii}\,, \tag{3.30b}$$

where ρ is the UD parameter that needs to be carefully chosen to yield admissible AC rules. From (3.13) and (3.30), the distributed algorithm that generates at the ith node a sequence of approximations, $f_i[k]$, for $k \geq 0$, has the implementation form

$$\begin{aligned} f_i[k] &= \rho_{ii}\, f_i[k-1] + \sum_{j \neq i} \rho_{ij}\, f_j[k-1] \\ &= (1 - \sum_{j \neq i} \rho_{ij})\, f_i[k-1] + \sum_{j \neq i} \rho_{ij}\, f_j[k-1] \\ &= f_i[k-1] + \sum_{j \neq i} \rho_{ij}\, (f_j[k-1] - f_i[k-1]) \\ &= f_i[k-1] + \rho \sum_{j \neq i} \phi_{ij}\, (f_j[k-1] - f_i[k-1])\,, \end{aligned} \tag{3.31}$$

with locally initialized condition $f_i[0] = f_i$ at the ith node. In vector representation, the admissible matrix W for the UD rules can be expressed as

$$W = W(\Phi,\, \rho) = I + \rho\, \Phi\,, \tag{3.32a}$$

and (3.13a) becomes

$$\mathbf{f}[k] = \mathbf{f}[k-1] + \rho\, \Phi\, \mathbf{f}[k-1]\,. \tag{3.32b}$$

We next analyze the eigenvalues of W for the UD rules to find the associated sufficient conditions for convergence as in (3.34). First, substituting (3.6) in (3.32a), we obtain the eigendecomposition of W,

$$W = V_\Phi \, \Lambda_W \, V_\Phi^T \, , \tag{3.33}$$

with $\Lambda_W = I + \rho \, \Lambda_\Phi = \text{diag}(\lambda_1(W), \, \lambda_2(W), \, \cdots, \, \lambda_N(W))$, where $\lambda_i(W) = 1 + \rho \, \lambda_i(\Phi)$ and $\lambda_1(W) \geq \lambda_2(W) \geq \cdots \lambda_N(W)$. The sufficient conditions for convergence, via Thm. 1, are the following: If

$$0 < \rho < (\phi_{\max})^{-1} \, , \tag{3.34a}$$

where $\phi_{\max} = \max_i |\phi_{ii}|$, then

$$\lambda_1(W) = 1, \;\; \text{and} \;\; |\lambda_i(W)| < 1 \;\; \text{for} \;\; i \geq 2 \tag{3.34b}$$

implying that in the limit, $f_i[k] \rightarrow G(\mathbf{f})$, $\forall i$.

The rate of convergence to $G(\mathbf{f})$ depends on the the selection of the UD parameter ρ in (3.32). We next present several choices ρ for first-order UD rules in Sec. 3.2.4.

Choices for ρ

In this section, we reference several choices of ρ from [17], including an optimal and two sub-optimal (but "good") choices that guarantee asymptotic convergence, and discuss their advantages and disadvantages.

(1) Optimal choice of ρ involves minimizing $\lambda(W)_{max} = \max_{2 \leq i \leq N} |\lambda(W)_i|$ and is given by

$$\rho_\infty = \frac{1}{\lambda(\Phi)_2 + \lambda(\Phi)_N} \, . \tag{3.35}$$

However, to obtain the optimal convergence rate for UD rules via (3.35), macroscopic information is needed, *i.e.*, global knowledge of Φ, or eigenvalues of Φ.

(2) One good choice of ρ that can be obtained via local processing and leads to convergence is given as the inverse of the number of connections of the maximally connected node, *i.e.*,

$$\rho_{\max} = \frac{1}{\phi_{\max} + \epsilon}, \tag{3.36}$$

where $\phi_{\max} = \max_i |\phi_{ii}|$ can be obtained by local computations and converges for any $\epsilon > 0$ arbitrarily small.

(3) Another good choice ρ that typically provide better convergence than $\rho_{\max}$ and leads to convergence is given by

$$\rho_1 = \frac{-\sum_i \lambda(\Phi)_i}{-\sum_i \lambda(\Phi)_i + \sum_i \lambda(\Phi)_i^2} = \frac{\frac{1}{N}\sum_i |\phi_{ii}|}{\frac{1}{N}\sum_i |\phi_{ii}| + \frac{1}{N}\sum_i |\phi_{ii}|^2}, \tag{3.37}$$

where $\langle \phi_{ii} \rangle$ and $\langle \phi_{ii} \rangle^2$ can be obtained via averaging computations, or estimated macroscopically.

3.2.5 Higher-Order LTI Rules

The use of slightly more complex local fusion rules can lead to improved local-estimation convergence rates. As shown in [17, 19], the convergence modes of the distributed computation algorithms are determined by the network topology and the choice the diffusion parameters. The large magnitude modes dominate the resulting system convergence rate. Extensions of distributed algorithms presented in Sec. 3.2.3–Sec. 3.2.4 can be formed that exploit a strictly causal filter, $H(z)$, at every node in order to reshape the convergence-mode magnitudes. As shown in Fig. 3.3, a block diagram for this extended class of local fusion rules at node i reduces to no filtering when $H(z) = 1$. A simple yet effective class of second order mode-shaping (higher

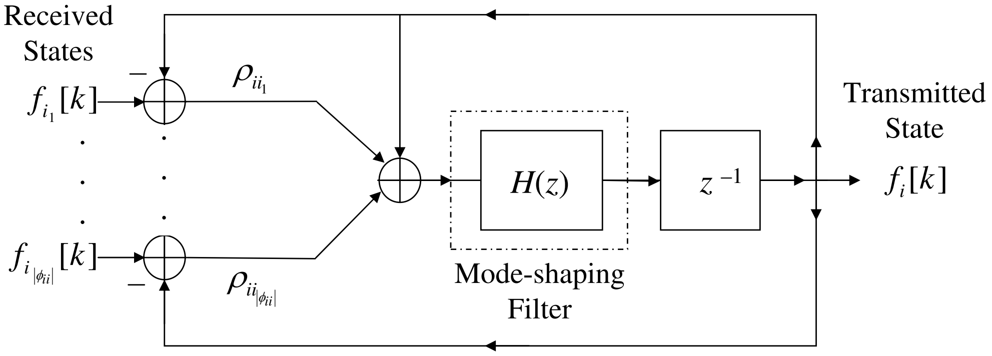


Figure 3.3: Block diagram for a mode-shaping fusion rule at the ith node.

than first-order) filters is given by

$$H(z) = \frac{1+c}{1+cz^{-2}} \tag{3.38}$$

for some $0 \leq c < 1$. Any such filter with $c > 0$ increases the magnitudes of all (unfiltered) modes with magnitude less than $\sqrt{c}$ at the benefit of decreasing the magnitudes of all modes with magnitude greater than $\sqrt{c}$. The specific convergence rate is determined by the set of resulting modes, and for large k is dominated by the maximum magnitude mode (see App. B.1.1). Thus (3.38) provides improved convergence rates when the choice of c is well matched to the set of unfiltered modes. We remark that although all $c \in [0, 1)$ yield convergence, as shown in App. B.1.1 and [17, 18], proper choice of c can greatly expedite the convergence rate of the distributed computations as we demonstrate via simulations in Sec. 3.4.

3.2.6 Improved Distributed Computation Algorithms

In this section we present improved versions of the distributed computation algorithms developed in [17,19] based on ρ_{ij}'s and mode-shaping filter $H(z)$ [3,20]. Given the topology Φ and we first consider a class of iterative algorithms that generate a sequence of approximations $f_i[k]$'s to $G(\mathbf{f})$ at the ith node via

$$f_i[k] \; = \begin{cases} f_i[0] & \text{if } k=0 \\ f_i[0] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\, f_j[0] & \text{if } k=1 \\ (1+c)\Big\{f_i[1] + \sum_{j\in\mathcal{U}_i}\, \rho_{ij}\, f_j[1]\Big\} & \\ \quad -c\Big\{f_i[0] + \sum_{j\in\mathcal{U}_i}\, \rho_{ij}\, f_j[0]\Big\} & \text{if } k=2 \\ (1+c)\Big\{f_i[k-1] + \sum_{j\in\mathcal{U}_i}\, \rho_{ij}\, f_j[k-1]\Big\} & \\ \quad -cf_i[k-2] & \text{if } k>2 \end{cases} \tag{3.39}$$

where recall $\mathcal{U}_i \triangleq \Big\{j \in \{1,\, 2,\, \cdots,\, N\};\ \phi_{ij} \neq 0\Big\}$, $c \in [0,\, 1)$ is to be macroscopically selected, and where the ρ_{ij}'s are to be selected via local negotiations over Φ.

3.3 Distributed Computations of Weighted Averages

In this section, we present a more general distributed version of the algorithm in (3.39), referred to as the *one-pass* algorithm, that can compute weighted averages of the form

$$\tilde{G}(\mathbf{f}) = \frac{1}{N}\sum_{i=1}^{N} \alpha_i\, f_i \tag{3.40}$$

where

$$\alpha_i = w_i/(\sum_{j=1}^{N} w_j) \tag{3.41}$$

w_i's are the locally available scalar weights (*e.g.*, functions of RSS estimates for source localization). At each node, the algorithm computes a sequence of estimates $\alpha_i[k]$'s of α_i and $f_i[k]$'s of $\tilde{G}(\mathbf{x})$, and for each "pass" (*i.e.*, each kth iteration), it computes both $\alpha_i[k]$ and $f_i[k]$, hence the name "one-pass".

Given the locally available data, f_i's and w_i's, and the one-pass algorithm (with mode-shaping filter $H(z)$) distributively computes a sequence of weighted estimates at the ith node via the following implementation steps:

1. *Step* $k = 0$ (*Initialization*)$:$ Let $s_i = \ w_i\, f_i$,

$$\alpha_i[0] = w_i, \tag{3.42a}$$

$$f_i[0] = \frac{s_i}{\alpha_i[0]}. \tag{3.42b}$$

2. *Step* $k = 1$:

$$\alpha_i[1] = \alpha_i[0] + \sum_{j \in \mathcal{U}_i} \rho_{ij} \alpha_j[0], \tag{3.43a}$$

$$f_i[1] = f_i[0] + \sum_{j \in \mathcal{U}_i} \rho_{ij} f_j[0] + (\frac{s_i}{\alpha_i[1]} - \frac{s_i}{\alpha_i[0]}). \tag{3.43b}$$

3. *Step* $k = 2$:

$$\alpha_i[2] = (1+c)\Big\{\alpha_i[1] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ \alpha_j[1]\Big\} - c\Big\{\alpha_i[0] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ \alpha_j[0]\Big\}, \tag{3.44a}$$

$$\begin{aligned} f_i[2] = (1+c)\Big\{f_i[1] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ f_j[1]\Big\} - c\Big\{f_i[0] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ f_j[0]\Big\} \\ + (\frac{s_i}{\alpha_i[2]} - \frac{s_i}{\alpha_i[1]}) - c(\frac{s_i}{\alpha_i[1]} - \frac{s_i}{\alpha_i[0]}). \end{aligned} \tag{3.44b}$$

4. *Step* $k > 2$:

$$\alpha_i[k] =(1+c)\Big\{\alpha_i[k-1] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ \alpha_j[k-1]\Big\} - c\alpha_i[k-2], \tag{3.45a}$$

$$\begin{aligned} f_i[k] =&(1+c)\Big\{f_i[k-1] + \sum_{j\in\mathcal{U}_i} \rho_{ij}\ f_j[k-1]\Big\} - cf_i[k-2] \\ &+(\frac{s_i}{\alpha_i[k]} - \frac{s_i}{\alpha_i[k-1]}) - c(\frac{s_i}{\alpha_i[k-1]} - \frac{s_i}{\alpha_i[k-2]}). \end{aligned} \tag{3.45b}$$

As $k \to \infty$, $\alpha_i[k] \to \alpha_i$ and the estimate at each node i, $f_i[k]$, converges to the global $\tilde{G}(\mathbf{f})$ estimate.

3.4 Distributed Computations Simulations & Analysis

In this section, we characterize the performance of the admissible, AC, LTI rules of Sec. 3.2 via a fictitious example of a signal estimation in noisy observations by a AHSN of N nodes. Given the observations

$$x_i = S + \eta_i\,, \quad i = 1,\, 2,\, \ldots,\, N \tag{3.46}$$

where $S \in \mathbb{R}$ is the parameter to be estimated and $\eta_i \sim \mathcal{N}(0, \sigma^2)$. Then, $x_i \sim \mathcal{N}(S, \sigma^2)$ and the minimum variance unbiased estimator (MVUE) of scalar-valued S based on the observations $\mathbf{x} = \begin{bmatrix} x_1 \; x_2 \; \cdots \; x_N \end{bmatrix}^T$, is given by

$$\hat{S} = \frac{1}{N} \sum_{j=1}^{N} x_j = \frac{1}{N} \mathbf{1}^T \mathbf{x} \,, \tag{3.47}$$

with mean-square-error (MSE) given by $\sigma^2(\hat{S}) = \mathrm{var}\,(\hat{S}) = \sigma^2/N$. We remark that (3.47) is of the form (3.8) with $\mathbf{f}(\mathbf{x}) = \mathbf{x}$. This is analogous to an unbiased estimator for a DC-level in white Gaussian noise (WGN), where x_i's are the independent snapshot observations and S is the unknown DC-level to be estimated [44].

The distributed implementation steps of the estimator $\hat{S}$ via distributed algorithms in Sec. 3.2 at the ith node are as follows:

(1) Initialize via locally available information, namely, the data x_i:

$$\hat{S}_i[0] = x_i \,, \tag{3.48}$$

(2) Apply (3.39) for k iterations to obtain the distributed estimate,

$$\hat{S}_i[k] = x_i[k] \,. \tag{3.49}$$

As $k \to \infty$, the distributed estimate at each node i converges to the global estimate, *i.e.*,

$$\hat{S}_i = \lim_{k\to\infty} \hat{S}_i[k] = \lim_{k\to\infty} x_i[k] = \frac{1}{N} \sum_{j=1}^{N} x_j = \hat{S}. \tag{3.50}$$

In the following examples, we use $N = 200$ nodes uniformly distributed in a circle of radius R with nominal distance d_o where $d_o/R = 1/4$. In the first example, we compare the performance of the first-order distributed algorithms for the various choices of the diffusion parameters using the relative MSE (rMSE) performance metric:

$$\text{rMSE} \;=\; \frac{\text{Additional MSE incurred by distributed computations}}{\text{MVUE MSE}}.$$

As the Fig. 3.4 shows, the first-order rule with NUD parameters, ρ_{ij}'s via local

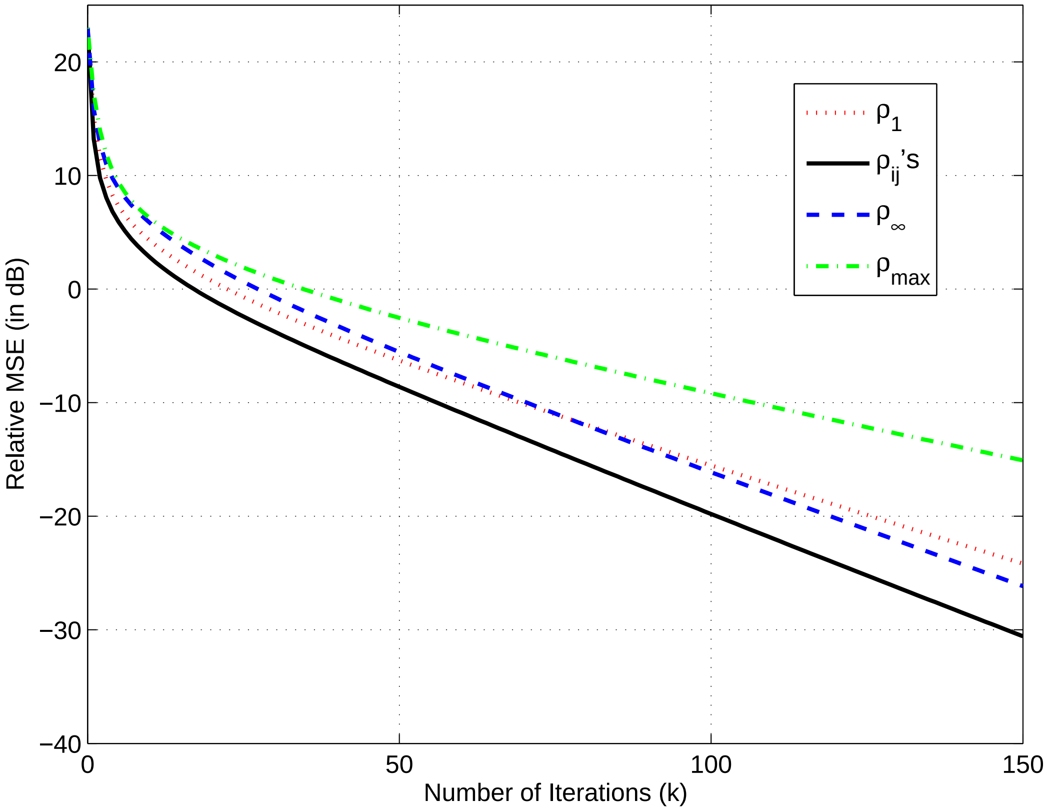


Figure 3.4: Performance comparison of distributed algorithms with UD parameters ρ_∞, $\rho_{\max}$, ρ_1 and NUD set of parameters ρ_{ij}'s via LN algorithm.

negotiation algorithm, outperforms the first-order rules with UD parameters, $\rho =$

$\rho_{\max}$, $\rho = \rho_\infty$ and $\rho = \rho_1$ respectively. As noted in [17], the rules with parameters ρ_∞ and ρ_1 have similar rMSE performance and they outperform rules with $\rho_{\max}$. This suggests that for UD rules, we should use rules with ρ_1 over rules with ρ_∞ since the former can be implemented without the need for macroscopic information (which may or may not be available).

In the second example, we examine higher-order LTI rules via the mode-shaping filter (3.38). In particular, we compare rules that employ ρ_{ij}'s via the LN algorithm and $\rho_{\max}$ for the filter parameters $c = \big[0,\, 0.3,\, 0.6\big]$, with $c = 0$ referring to the no-filtering case as in the first example. As Fig. 3.5 shows, the higher-order

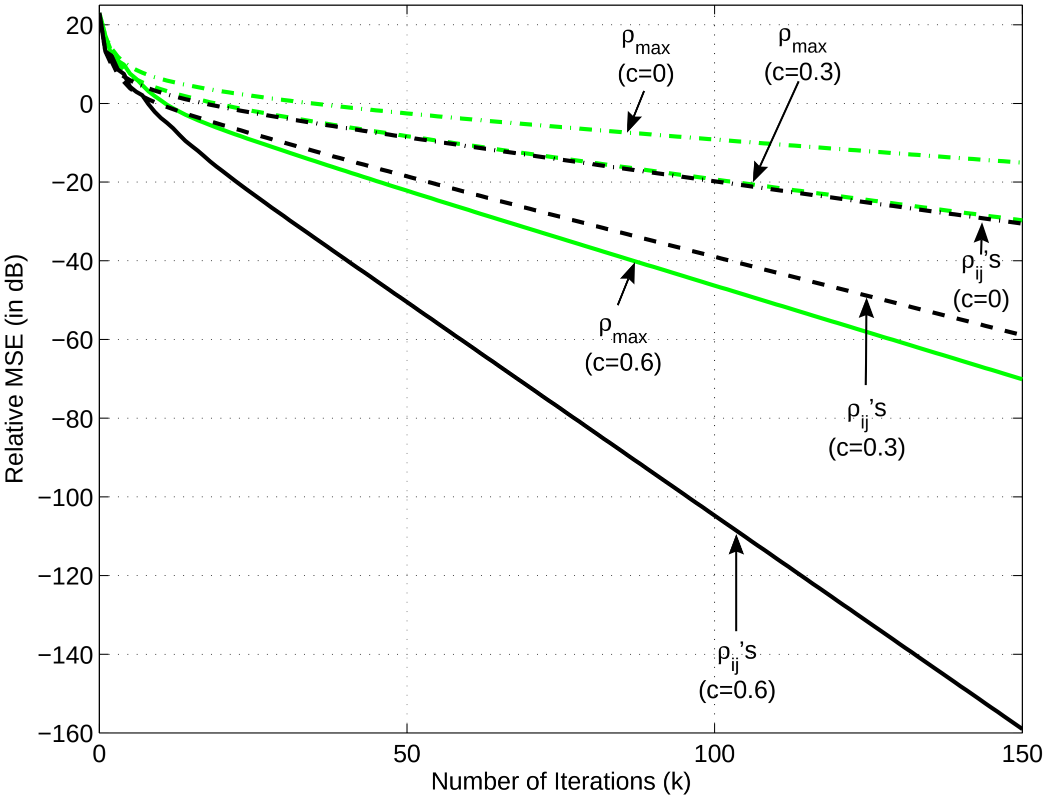


Figure 3.5: Performance comparison of distributed algorithms with mode-shaping filter $H(z)$ with $c = \big[0,\, 0.3,\, 0.6\big]$ for $\rho_{\max}$ and ρ_{ij}'s (via LN algorithm).

LTI rules via $H(z)$ with both $c = 0.3$ and $c = 0.6$ outperform first-order LTI rules with $c = 0$. The performance improvement between rules with $c = 0.6$ versus $c = 0$ is quite dramatic, as much as 50 dB rMSE between the $\rho_{\max}$ cases and 85 dB rMSE between the ρ_{ij}'s cases at $k = 100$ respectively.

3.5 Distributed Computations Summary

In summary, we presented a broad class of admissible, LTI and AC rules for distributed computations and discuss the key conditions needed to guarantee convergence. In addition, we discussed several classes of first-order LTI rules and the advantages/disavantages of these algorithms with regards to distributed implementation in AHSN's. We developed new and improved versions of the distributed algorithms reported in [17, 18] that (i) use NUD parameters ρ_{ij}'s via the LN algorithm with improved convergence rates and (ii) can implement distributively a broader class of problems based on weighted averages. The brief simulations substantiate the improved performance of these new algorithms.

Chapter 4

Fusion in Space

In this chapter, we design and develop sensor fusion algorithms for spatial processing via a distributed network of sensors. As discussed in Sec. 2.2, the use and deployment of sensor networks for performing source localization has gained wide interest in many civilian and military applications due to the large spatial-coverage capabilities. Numerous localization methods, such as TOA, TDOA, DOA and RSS based methods, involving sensor networks have been proposed recently in literature and the methods vary widely depending on the applications. In addition, as discussed previously, localization methods for AHSN's also vary in terms of processing architecture such as centralized [13,35], hierarchical of sub-networks or clusters [37,48,59] including mobile rovers [40] and decentralized localization [15, 54, 60] depending upon the availability of sensor network's resources such as battery power and communication bandwidth.

In our investigation, we focus on resource efficient source localization algorithms for AHSN's that trade-off performance and complexity. In addition to low-complexity, we are interested in algorithms that can be implemented distributively in decentralized settings. In the process, we develop signal processing strategies that keep power consumption and communication bandwidth within the sensor network to acceptable levels without significantly sacrificing localization accuracy.

In Sec. 4.1, we examine several classes of low-complexity estimators and develop the corresponding localization algorithms for a single (static) source within a sensor network. Then, we perform simulations to compare and contrast the MSE performance of the derived algorithms. In Sec. 4.2, we demonstrate how these algorithms can be implemented distributively in AHSN's. We present the key steps to mapping the (global) centralized estimators to the forms of averages of Sec. 3.2 so distributed computations can be performed locally at all participating nodes in the network. Then, we present simulation-based analysis and discuss performance results comparing the distributed source localization algorithms with their centralized counterparts in Sec. 4.3. We remark that for convenience, we drop the dependence on t in the presentation of this chapter.

4.1 Resource-Efficient Source Localization Algorithms

We examine low-complexity estimators that can be implemented distributively based on centroid and least-squares (LS) estimations to determine the unknown source location $\mathbf{p}_s$. In our algorithm development, we assume that given M detecting sensor nodes in the AHSN, each node knows its own position location (*e.g.*, via a GPS sensor available on-board or via one of the proposed autonomous methods for sensor network self-localization or self-calibration such as using beacons or moving targets of opportunity in the AHSN [34, 61–64]). We further assume that each participating node has the capability to estimate the RSS as discussed in Sec. 2.1.4.

We first discuss centroid-based methods in Sec. 4.1.1 and range-difference LS-based methods Sec. 4.1.2. Then, we perform Monte Carlo simulations to compare and contrast the MSE performance of the derived algorithms against the associated CRB's and among the algorithms while assessing the overall complexity in Sec. 4.1.3.

4.1.1 Centroid-based Estimators

The simplest class of estimators that we considered is centroid-based estimators. Centroid-based estimators are inherently low-complexity and simple to implement, and they can provide increasingly more reliable estimates with increasing sensor density. Given M detecting sensor nodes in AHSN, the (centralized) *centroid* (CEN) *estimator*, $\hat{\mathbf{p}}_{\mathrm{c}}$, is given by

$$\hat{\mathbf{p}}_{\mathrm{c}} = \begin{bmatrix} \hat{x}_{\mathrm{c}} & \hat{y}_{\mathrm{c}} \end{bmatrix}^T = \begin{bmatrix} \frac{1}{M}\sum_{i=1}^{M} x_i & \frac{1}{M}\sum_{i=1}^{M} y_i \end{bmatrix}^T, \tag{4.1}$$

where $\mathbf{p}_i = \begin{bmatrix} x_i & y_i \end{bmatrix}^T$ for $i = 1, \ldots, M$, is the ith sensor location assumed to be known. As the estimator in (4.1) does not use RSS information, its performance is high SNR-limited. The RSS estimates, $\widehat{\sigma^2}_{\mathrm{RSS}_i}$'s, can be exploited to obtain improved estimators at slightly higher complexity. The (centralized) *weighted-centroid* (WCEN) *estimator*, $\hat{\mathbf{p}}_{\mathrm{wc}}$, has the form

$$\hat{\mathbf{p}}_{\mathrm{wc}} = \begin{bmatrix} \hat{x}_{\mathrm{wc}} & \hat{y}_{\mathrm{wc}} \end{bmatrix}^T = \begin{bmatrix} \frac{\sum_{i=1}^{M} \widehat{\sigma^2}_{\mathrm{RSS}_i} x_i}{\sum_{j=1}^{M} \widehat{\sigma^2}_{\mathrm{RSS}_j}} & \frac{\sum_{i=1}^{M} \widehat{\sigma^2}_{\mathrm{RSS}_i} y_i}{\sum_{j=1}^{M} \widehat{\sigma^2}_{\mathrm{RSS}_j}} \end{bmatrix}^T. \tag{4.2}$$

The WCEN estimator produces better estimates than the CEN estimator because it gives more weight to the sensor nodes with higher RSS's (*i.e.*, nodes that are closer to the source) and less weights to sensor nodes with lower RSS's (*i.e.*, nodes that are farther from the source). Both estimators can be computed distributively; by exploiting locally available information, distributed algorithms can be constructed that asymptotically obtain (4.1)– (4.2) over any arbitrary, connected, ad-hoc network [3,19]. However, the distributed implementation of the WCEN estimator can be much more computationally complex than the CEN estimator unless the one-pass algorithm is used.

4.1.2 LS-based Estimators

By "processing" the M-dimensional estimation problem corresponding to (2.4), we may obtain a reduced $(M-1)$-dimensional range difference problem to which linear LS estimates can be readily constructed [3,6,19,65,66]. For convenience, we refer to the resulting estimators as LS estimators.

We next consider a (centralized) *linear LS* (LLS) *estimator* that is a generalized version of [19]. It is assumed that the ith node in the network knows its location, the ith node possesses source-node range measurements of the form

$$g_i = \sigma_{\mathrm{RSS}_i} + \omega_i \tag{4.3}$$

where the ω_i's are zero-mean σ_ω^2-power i.i.d. Gaussian sequences, and $\sigma^2_{\mathrm{RSS}_i} = \sigma_s^2/r_i^2$, with $r_i = \|\mathbf{p}_i - \mathbf{p}_\mathrm{s}\|$. The source-location estimators we considered exploit the locally available minimum-variance unbiased estimates (MVUEs) of $\sigma^2_{\mathrm{RSS}_i}$, *viz.*,

$$\widehat{\sigma^2}_{\mathrm{RSS}_i} = \min\left[g_i^2 - \sigma_\omega^2,\, 0\right] . \tag{4.4}$$

We remark that the AHSN for a fixed time t is formed via threshold detection by including in the AHSN only nodes with $\widehat{\sigma^2}_{\mathrm{RSS}_i} > \sigma_T^2$, for some suitably preset threshold $\sigma_T^2 > 0$.

Assuming M detecting nodes in AHSN, the LLS estimator of interest is based on $(M-1)$ range-squared difference equations, formed by viewing the Mth sensor (arbitrarily chosen) as a reference. Given that

$$r_i^2 = \|\mathbf{p}_i - \mathbf{p}_s\|^2 = \|\mathbf{p}_i\|^2 + \|\mathbf{p}_s\|^2 - 2\,\mathbf{p}_i^T\,\mathbf{p}_s\,, \tag{4.5}$$

the LLS estimator exploits the following:

$$\begin{aligned} r_i^2 - r_M^2 &= \sigma_s^2\,(\widehat{\sigma^{-2}}_{\mathrm{RSS}_i} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M}) \\ &= \|\mathbf{p}_i\|^2 - \|\mathbf{p}_M\|^2 - 2\,(\mathbf{p}_i - \mathbf{p}_M)^T\,\mathbf{p}_s\,. \end{aligned} \tag{4.6}$$

Expanding (4.6) into vector form and rearranging the terms, we obtain

$$\begin{aligned} \begin{bmatrix} \|\mathbf{p}_i\|^2 - \|\mathbf{p}_M\|^2 \\ \vdots \\ \|\mathbf{p}_{M-1}\|^2 - \|\mathbf{p}_M\|^2 \end{bmatrix} &= \begin{bmatrix} 2(\mathbf{p}_i - \mathbf{p}_M)^T \\ \vdots \\ 2(\mathbf{p}_{M-1} - \mathbf{p}_M)^T \end{bmatrix} \mathbf{p}_s + \begin{bmatrix} \widehat{\sigma^{-2}}_{\mathrm{RSS}_i} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M} \\ \vdots \\ \widehat{\sigma^{-2}}_{\mathrm{RSS}_{M-1}} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M} \end{bmatrix} \sigma_s^2 \\ &= \begin{bmatrix} 2(\mathbf{p}_i - \mathbf{p}_M)^T & (\widehat{\sigma^{-2}}_{\mathrm{RSS}_i} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M}) \\ \vdots & \vdots \\ 2(\mathbf{p}_{M-1} - \mathbf{p}_M)^T & (\widehat{\sigma^{-2}}_{\mathrm{RSS}_{M-1}} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M}) \end{bmatrix} \begin{bmatrix} \mathbf{p}_s \\ \sigma_s^2 \end{bmatrix} \end{aligned} \tag{4.7}$$

Let $\tilde{\mathbf{r}}$ be the $(M-1) \times 1$ vector on the LHS of (4.7), and $\boldsymbol{\Delta}\mathbf{V}$ be the $(M-1) \times 3$ matrix on the RHS and $\mathbf{x} \triangleq \begin{bmatrix} \mathbf{p}_s^T & \sigma_s^2 \end{bmatrix}^T = \begin{bmatrix} x_s & y_s & \sigma_s^2 \end{bmatrix}^T$ be the 3×1 the vector of unknowns on the RHS of (4.7). Then the LLS estimator $\hat{\mathbf{x}}_{\mathrm{lls}}$ of the unknown $\mathbf{x}$ is

$$\hat{\mathbf{x}}_{\mathrm{lls}} = \begin{bmatrix} \hat{x}_{\mathrm{lls}} & \hat{y}_{\mathrm{lls}} & \widehat{\sigma^2}_{\mathrm{lls}} \end{bmatrix}^T = (\boldsymbol{\Delta}\mathbf{V}^T \boldsymbol{\Delta}\mathbf{V})^{-1} \boldsymbol{\Delta}\mathbf{V}^T \tilde{\mathbf{r}}\,. \tag{4.8}$$

Similarly, a *weighted-LS* (WLS) *estimator* can be formed as follows:

$$\hat{\mathbf{x}}_{\text{wls}} = \left[\hat{x}_{\text{wls}} \ \ \hat{y}_{\text{wls}} \ \ \widehat{\sigma^2}_{\text{wls}}\right]^T = (\mathbf{\Delta V}^T\mathbf{W}\mathbf{\Delta V})^{-1}\mathbf{\Delta V}^T\mathbf{W}\tilde{\mathbf{r}}\,, \tag{4.9}$$

where the $\mathbf{W}$ is an $(M-1)\times(M-1)$ matrix function of the RSS estimates. For weighted LS estimation with uncorrelated noise assumptions, setting $\mathbf{W} = \mathbf{C}^{-1}(\mathbf{x})$ will produce estimates with the smallest variances [44]. However, in general, the covariance matrix $\mathbf{C}(\mathbf{x})$ is not amenable to distributed implementation.

The Weight Matrix

In [6, 19], a simple and straight forward use of the locally available RSS estimates as weights is to define $\mathbf{W}$ as

$$\hat{\mathbf{W}} = \begin{bmatrix} \widehat{\sigma^2}_{\text{RSS}_1} & 0 & \cdots & 0 \\ 0 & \widehat{\sigma^2}_{\text{RSS}_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \widehat{\sigma^2}_{\text{RSS}_{(M-1)}} \end{bmatrix}. \tag{4.10}$$

However, upon performing further simulation analysis, we find that the weights are not functions of $\widehat{\sigma^2}_{\text{RSS}}$ but instead functions of $\widehat{\sigma^6}_{\text{RSS}}$.

In particular, the estimators exploit the relative node-source range-squared estimates via the estimated RSS's of the form

$$\widehat{\sigma^{-2}}_{\text{RSS}_i} = \left[\widehat{\sigma^2}_{\text{RSS}_i}\right]^{-1} = \frac{r_i^2}{\sigma_s^2} + \epsilon_i \tag{4.11}$$

and where the estimation-errors ϵ_i are independent in i. In simulation-based evaluation of the associated ith MSE, $\sigma^2_{\epsilon_i} \triangleq E\left\{\epsilon_i^2\right\}$, reveals it is proportional to r_i^6 as shown

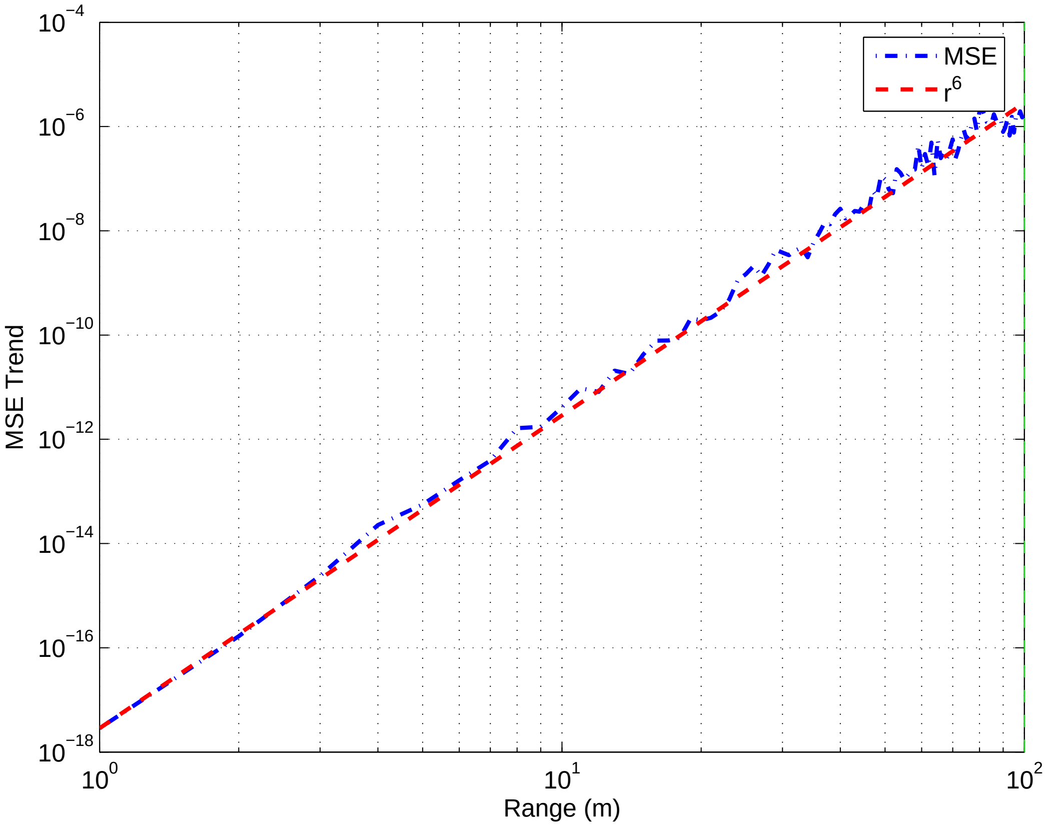


Figure 4.1: MSE trend of $\sigma^2_{\epsilon_i}$ shows that $\sigma^2_{\epsilon_i} \propto \sigma^{-6}_{\mathrm{RSS}_i} = r^6_i/\sigma^2_s$, for the AHSN example of a $R = 100$ m, $M = 20$ sensors and SNR = 60 dB.

in Fig. 4.1, *i.e.*,

$$\sigma^2_{\epsilon_i} \propto \sigma^{-6}_{\mathrm{RSS}_i} \ , \tag{4.12}$$

for $\widehat{\sigma^2}_{\mathrm{RSS}_i} > \sigma^2_T$ and for any detection threshold $\sigma^2_T > 0$. Further details can be found in App. C.1.1. Analytically, it is shown in App. C.1 that $\mathbf{W}$ is the following $(M-1) \times (M-1)$ MSE-weight matrix

$$\mathbf{W} = \left[\mathrm{diag}(\sigma^2_{\epsilon_1}, \sigma^2_{\epsilon_2}, \cdots, \sigma^2_{\epsilon_{M-1}}) + \sigma^2_{\epsilon_M}\, \mathbf{1}\mathbf{1}^T \right]^{-1} , \tag{4.13}$$

where $\mathbf{1}$ denotes an $(M-1) \times 1$ vector of one's. Due to (4.12) and (4.13), $\hat{\mathbf{x}}_{\mathrm{wls}}$

from (4.9) is not a valid estimate as it depends on the unknown source location and power. A valid WLS-type estimate of the form (4.9) that is amenable to distributed implementation can be obtained by employing in (4.9) the following expression for $\mathbf{W}$ in place of (4.13)

$$\hat{\mathbf{W}} = \mathrm{diag}(\widehat{\sigma^6}_{\mathrm{RSS}_1}, \widehat{\sigma^6}_{\mathrm{RSS}_2}, \cdots, \widehat{\sigma^6}_{\mathrm{RSS}_{M-1}}) \tag{4.14}$$

where $\widehat{\sigma^6}_{\mathrm{RSS}_i}$ is the MVUE of $\sigma^6_{\mathrm{RSS}_i}$ and is given by

$$\widehat{\sigma^6}_{\mathrm{RSS}_i} = g_i^6 - 15\,\sigma_\omega^2\,g_i^4 + 45\,\sigma_\omega^4\,g_i^2 - 15\,\sigma_\omega^6. \tag{4.15}$$

Further details can be found in App. C.1.2.

4.1.3 Localization Simulations & Analysis

We next present the average MSE (range squared in m^2) performance evaluation of the the centroid-based estimators in (4.1)– (4.2) and LS estimators in (4.8)–(4.9) based on Monte-Carlo simulations. We compare and analyze the localization performance for the different estimators with each other and also with the associated CRB, and obtain the MSE results in dB m^2.

The setting of interest involves a single (static) acoustic source located at the center of a circle of radius R with sensor nodes uniformly distributed in this circle. We assume a stochastic signal model with $L = 1000$ independent snapshots[1] over a circle with $R = 100$ m and detection threshold, $\sigma_T^2 = 0$ dB. We further assume

[1] The number of snapshots depends on the data sampling rate, f_s. In general, acoustic sensing for military applications [23, 38], f_s is typically from 512 Hz to 4096 Hz. For example, in [23], acoustic signals are sampled at $f_s = 1024$ Hz and $L = 512$ snapshots, and in [38], acoustic signals are sampled at $f_s = 4096$ Hz and $L = 3730$ snapshots are used.

that the reference sensor, the Mth sensor, in (4.8)–(4.9) is chosen to be the "loudest" or the closest sensor to the source. For this analysis, the weight matrix used for the WLS estimator is a diagonal matrix of RSS estimates as in (4.10). The resulting MSE curves are based on averaging over $MC = 250$ independently drawn sensor lay-outs.

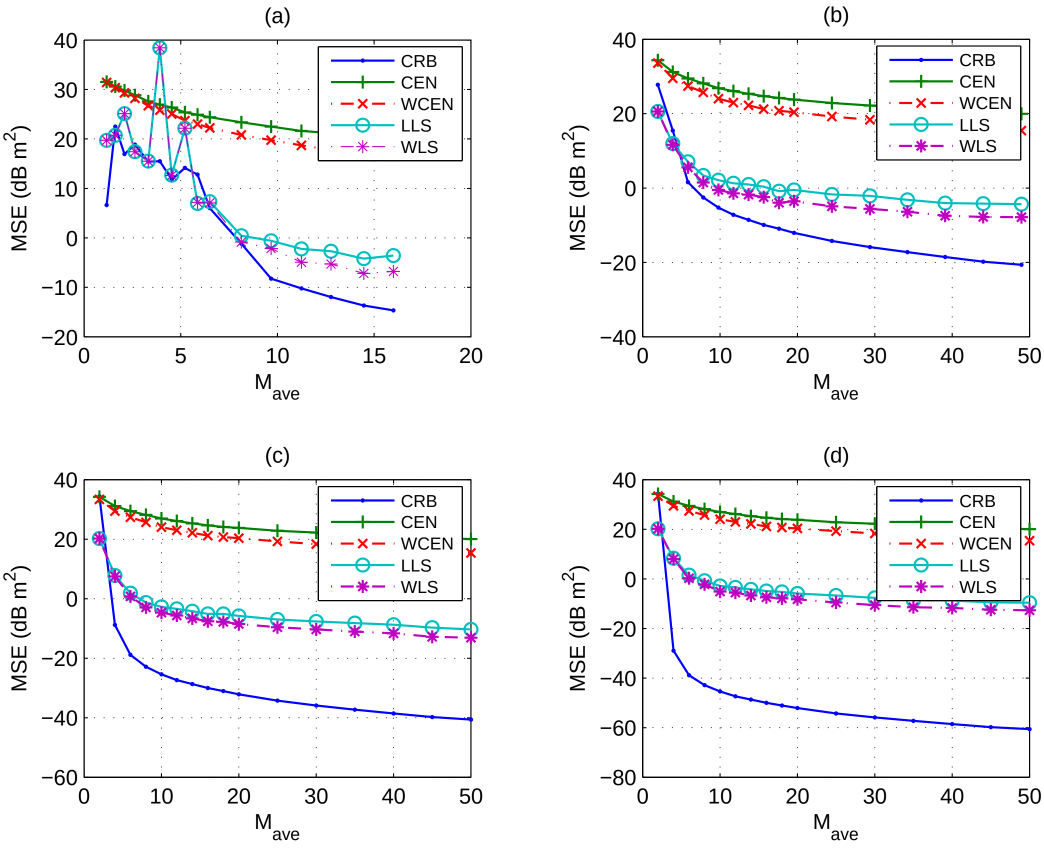


Figure 4.2: Performance comparison of CEN, WCEN, LLS, and WLS estimators and the associated CRB: MSE vs. M_{ave} sensors for (a) SNR = 30 dB, (b) SNR = 40 dB, (c) SNR = 60 dB and (d) SNR = 80 dB.

Fig. 4.2 shows the simulated estimator MSE and the associated CRB vs. M_{ave} for SNR $= \left[30,\ 40,\ 60,\ 80\right]$ dB. The plots show the MSE performance trade-offs as a function of M_{ave}. First, we note that for $M_{\text{ave}} \geq 10$ sensors, the MSE performance of the four estimators (CEN, WCEN, LLS and WLS) improves with increasing M_{ave}. In particular, the performance improves approximately by 3 dB when M_{ave} is doubled. Also, we note that the range estimation errors for the four estimators are within 10's of meters; MSE $\leq 1\,\text{m}^2$ and MSE $\leq 100\,\text{m}^2$ for LS-based and centroid-based estimators, respectively. Finally, the LS-based estimators perform well with respect to the CRB, especially at lower SNR's.

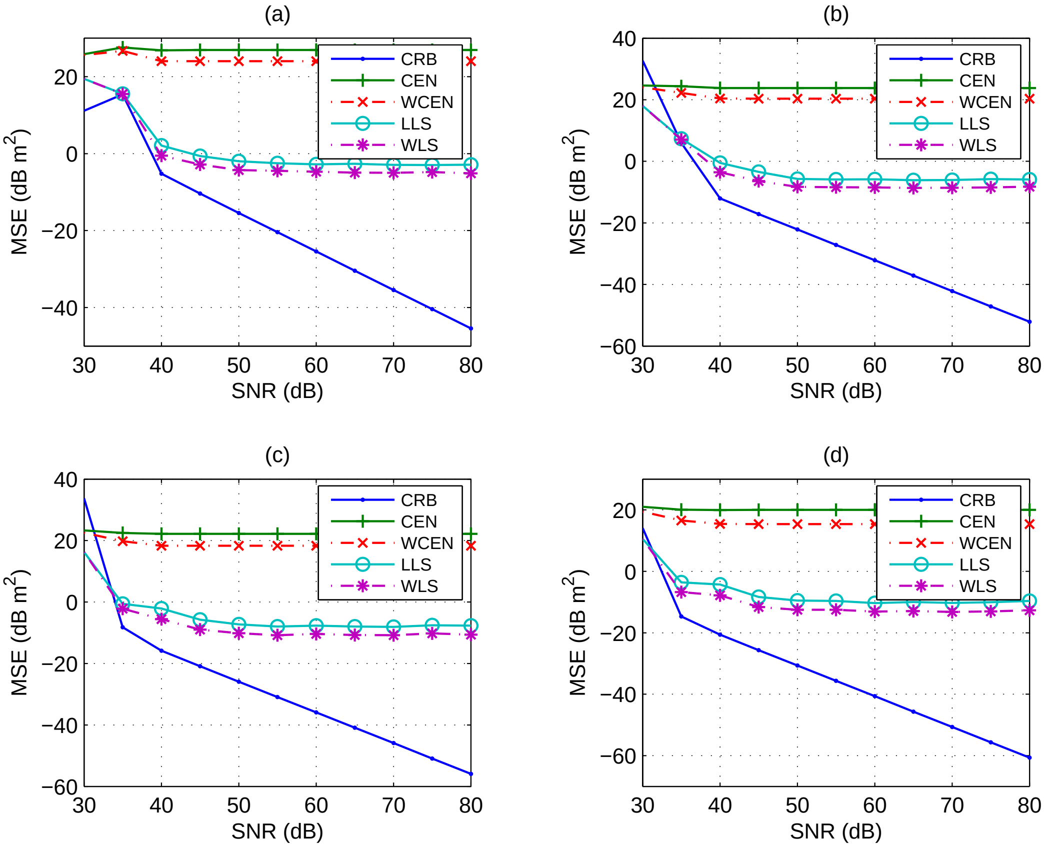


Figure 4.3: Performance comparison of CEN, WCEN, LLS, and WLS estimators and the associated CRB: MSE vs. SNR for (a) $N = 10$, (b) $N = 20$, (c) $N = 30$ and (d) $N = 50$ sensors.

Fig. 4.3 shows the MSE vs. SNR comparison of the four estimators and the associated CRB for $N = \left[10,\ 20,\ 30,\ 50\right]$ sensors. As the figure reveals, weighted estimators, whether centroid-based or LS-based, outperform their non-weighted counterparts by several dB. The performance gain of weighted estimators over non-weighted estimators, however, does not improve with increasing SNR (*i.e.*, SNR-limited). In addition, LS-based estimators considerably outperform their centroid-based counterparts; for SNR $\geq$ 40 dB, LS-based estimators produce less than 1 m range error. Furthermore, as all four estimators are biased, their MSE curves level off at higher SNR's, as opposed to the associated CRB. Although, in principle, sub-meter range errors could be possible as indicated by the CRB, range estimation errors of several meters are well within acceptable performance for many such ad-hoc networks of acoustic sensors [67].

4.1.4 Reference Sensor Sensitivity Analysis

In this section, we study the effect on MSE performance of the range difference LLS and WLS estimators with respect to choosing a reference sensor. Again, we assume the stochastic signal model in (2.12) with $L = 1000$ and we assume $R =$ 100 m. From the previous discussion, we choose the "loudest" or the closest sensor to the source as the centralized (global) reference sensor. However, in decentralized settings, choosing the loudest might not be possible due to communication and/or computational complexity constraints. Therefore, we perform simulation analysis for LS-based estimators comparing a randomly chosen (global) reference versus the loudest (global) reference. Fig. 4.4 shows the MSE vs. M_{ave} comparison for SNR = $\left[40,\ 60\right]$ dB and the MSE vs. SNR comparison for $N = \left[10,\ 30\right]$ sensors. As the figure reveals, for $M_{\text{ave}} \geq 10$ (Fig. 4.4 (a) and (b)) and SNR $\geq$ 40 dB (Fig. 4.4 (c) and (d)), both LLS and WLS with random reference estimators are sensor density limited and SNR limited respectively at $\approx$ 0 dB MSE, while both the LLS and WLS with loudest reference estimators are only SNR limited as discussed previously for SNR $\geq$ 50

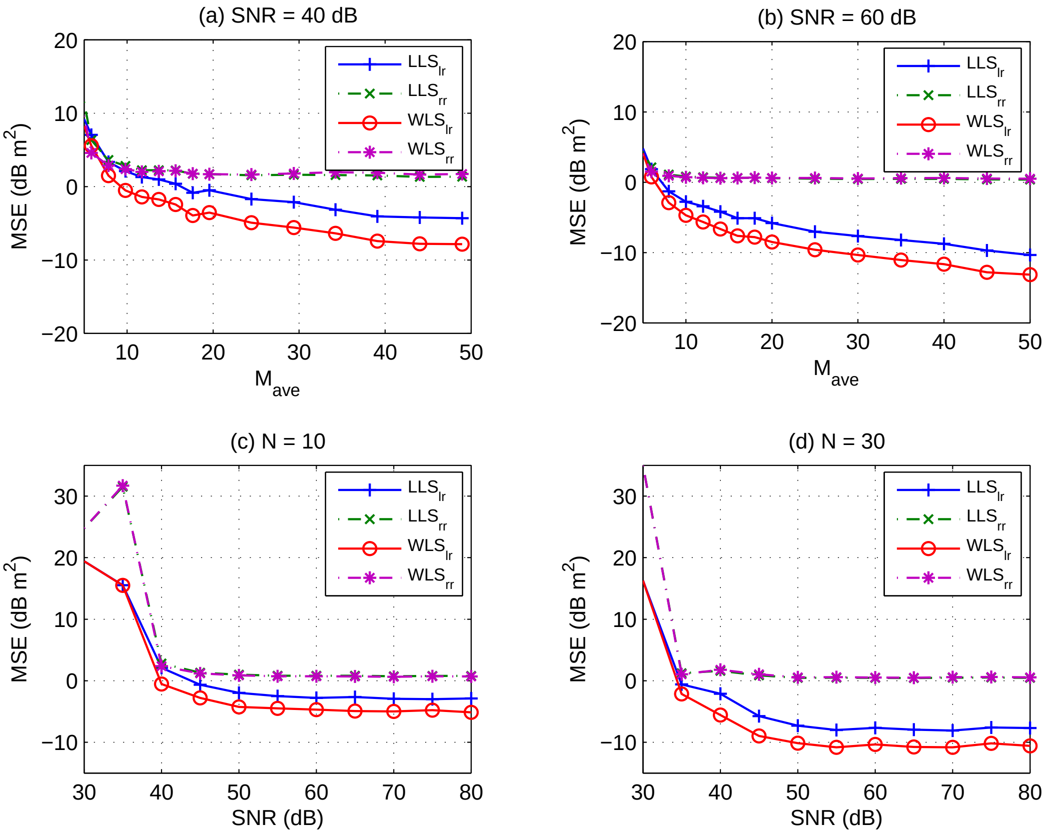


Figure 4.4: Performance comparison of LLS and WLS with the "loudest" sensor as the reference sensor (denoted as LLS$_{\mathrm{lr}}$ and WLS$_{\mathrm{lr}}$) and a randomly chosen sensor as the reference sensor (denoted as LLS$_{\mathrm{rr}}$ and WLS$_{\mathrm{rr}}$): MSE vs. M_{ave} for (a) SNR = 40 dB and (b) SNR = 60 dB; and MSE vs. SNR for (c) $N = 10$ sensors and (d) $N = 30$ sensors.

dB. Also note that there is no performance gain using WLS with random reference sensor over the non-weighted version. In summary, the degradation in performance is attributed to the fact that a randomly (globally) selected sensor reference has a lower RSS compared to that of the "loudest" sensor, and this leads to higher sensitivity to measurement errors in (4.7).

4.2 Distributed Implementation

We next present the essential steps of mapping centralized localization algorithms described in Sec. 4.1 to computations of the form (3.8). We first dicuss the steps for the distributed implementation of centroid-based estimators and then discuss the steps for distributed implementation the LS-based estimators in Sec. 4.2.1 and Sec. 4.2.2 respectively. Next, we discuss the performance metrics used in performance evaluation and comparison in Sec. 4.2.3. Then, we present simulation results and performance analysis comparing the distributed source localization algorithms with their centralized counterparts in Sec. 4.2.4.

4.2.1 Distributed Centroid-based Estimation

Distributed CEN

We first consider the centroid algorithm for source localization as it can be readily mapped to a locally-constructed distributed computation. Given M sensors, with each sensor knowing its own location, the (centralized) centroid location estimator, $\hat{\mathbf{p}}_{\mathrm{c}} = \begin{bmatrix}\hat{x}_{\mathrm{c}} & \hat{y}_{\mathrm{c}}\end{bmatrix}^T$, is given by (4.1). Each entry of $\hat{\mathbf{p}}_{\mathrm{c}}$ is already in the form (3.8). The distributed implementation of $\hat{\mathbf{p}}_{\mathrm{c}}$ involves two parallel-distributed computations corresponding to the x and y components of $\hat{\mathbf{p}}_{\mathrm{c}}$. Let n_p denote the number of parallel operations and $\mathbf{f}_i = \left[f_i^{(1)}\, f_i^{(2)}\, \cdots\, f_i^{(n_p)}\right]$ (in this case $n_p = 2$), then the implementation steps at the ith node are as follows:

(1) Initialize via locally available information, namely, the coordinates of node i:

$$\mathbf{f}_i[0] = \left[f_i^{(1)}[0] \;\; f_i^{(2)}[0]\right]^T = [x_i \;\; y_i]^T; \tag{4.16}$$

(2) Apply distributed computation algorithms of Sec. 3.2 for k iterations to obtain the distributed centroid (dCEN) estimate,

$$\hat{\mathbf{p}}_{i_{\text{dc}}}[k] = \left[f_i^{(1)}[k] \;\; f_i^{(2)}[k]\right]^T = \left[\hat{x}_{i_{\text{dc}}}[k] \;\; \hat{y}_{i_{\text{dc}}}[k]\right]^T. \tag{4.17}$$

As $k \rightarrow \infty$, the dCEN estimate at each node i converges to the global CEN estimate, *i.e.*,

$$\begin{aligned}\hat{\mathbf{p}}_{i_{\text{dc}}} &= \lim_{k\rightarrow\infty}\left\{\left[\hat{x}_{i_{\text{dc}}}[k] \;\; \hat{y}_{i_{\text{dc}}}[k]\right]^T\right\} = \lim_{k\rightarrow\infty}\left\{\left[f_i^{(1)}[k] \;\; f_i^{(2)}[k]\right]^T\right\} \\ &= \left[\frac{1}{M}\sum_{i=1}^{M} x_i \;\; \frac{1}{M}\sum_{i=1}^{M} y_i\right]^T = \left[\hat{x}_{\text{c}} \;\; \hat{y}_{\text{c}}\right]^T = \hat{\mathbf{p}}_{\text{c}}.\end{aligned} \tag{4.18}$$

Distributed WCEN

The weighted centroid algorithm can also be mapped to a locally-constructed distributed computation of averages but it requires a few more steps than the non-weighted CEN estimator. Here, we assumes the weights are the RSS estimates. First, we notice that we can rewrite (4.2) as

$$\hat{\mathbf{p}}_{\text{wc}} = \left[\hat{x}_{\text{wc}} \;\; \hat{y}_{\text{wc}}\right]^T = \left[\frac{1}{M}\sum_{i=1}^{M}\left\{\frac{\widehat{\sigma^2}_{\text{RSS}_i}}{\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\text{RSS}_j}}\right\} x_i \;\; \frac{1}{M}\sum_{i=1}^{M}\left\{\frac{\widehat{\sigma^2}_{\text{RSS}_i}}{\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\text{RSS}_j}}\right\} y_i\right]^T. \tag{4.19}$$

Each term in (4.19) is a weighted sum of weighted sums, which requires a two-stage cascade implementation. In the first stage, we need to distributively compute $\hat{\alpha}$, an estimate for the average weights $\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\mathrm{RSS}_j}$, which is already of the form in (3.8). In the second stage, we substitute the $\hat{\alpha}$ into (4.19) to map each term of (4.19) into the form in (3.8). Then, we can apply the distributed algorithm to compute $\hat{\mathbf{p}}_{\mathrm{wc}}$. The distributed implementation of $\hat{\mathbf{p}}_{\mathrm{wc}}$ using (3.31) or (3.39) involves two parallel cascade-distributed computation of averages corresponding to the x and y components of $\hat{\mathbf{p}}_{\mathrm{wc}}$. At node i, the implementation steps are as follows:

(1) Initialize the average weight via locally available information, namely the RSS estimate at node i: $\mathbf{f}_i^{\mathrm{stage1}}[0] = f_i^{\mathrm{stage1}}[0] = \widehat{\sigma^2}_{\mathrm{RSS}_i}$.

(2) Apply distributed computation algorithms of Sec. 3.2 for k_1 iterations to obtain the distributed average weight estimate, $\hat{\alpha}_i[k_1] = f_i^{\mathrm{stage1}}[k_1]$.

(3) For each term in (4.19), initialize with $\hat{\alpha}_i[k_1]$ and locally available information, namely the RSS estimates and the coordinates of node i:

$$\begin{aligned}\mathbf{f}_i^{\mathrm{stage2}}[k_1,0] =& \left[f_i^{\mathrm{stage2(1)}}[k_1,0] \;\; f_i^{\mathrm{stage2(2)}}[k_1,0]\right]^T \\ =& \left[\frac{\widehat{\sigma^2}_{\mathrm{RSS}_i}}{\hat{\alpha}_i[k_1]}\, x_i \;\; \frac{\widehat{\sigma^2}_{\mathrm{RSS}_i}}{\hat{\alpha}_i[k_1]}\, y_i\right]^T. \end{aligned} \tag{4.20}$$

(4) Apply distributed computation algorithms of Sec. 3.2 for k_2 iterations to obtain the distributed weighted centroid (dWCEN) estimate,

$$\begin{aligned}\hat{\mathbf{p}}_{i_{\mathrm{dwc}}}[k_1,k_2] &= \left[f_i^{\mathrm{stage2(1)}}[k_1,k_2] \quad f_i^{\mathrm{stage2(2)}}[k_1,k_2]\right]^T \\ &= \left[\hat{x}_{i_{\mathrm{dwc}}}[k_1,k_2] \quad \hat{y}_{i_{\mathrm{dwc}}}[k_1,k_2]\right]^T. \end{aligned} \tag{4.21}$$

The convergence of dWCEN to WCEN, is dependent on the convergence of $\hat{\alpha}_i[k_1]$. As $k_2 \rightarrow \infty$ and $k_1 \rightarrow \infty$, the dWCEN estimate at each node i converges to the global WCEN estimate, *i.e.*,

$$\begin{aligned}
\hat{\mathbf{p}}_{i_{\mathrm{dwc}}} &= \lim_{k_2\rightarrow\infty}\left\{\lim_{k_1\rightarrow\infty}\left\{\left[\hat{x}_{i_{\mathrm{wdc}}}[k_1,k_2] \quad \hat{y}_{i_{\mathrm{dc}}}[k_1,k_2]\right]^T\right\}\right\} \\
&= \left[\frac{1}{M}\sum_{i=1}^{M}\left\{\frac{\widehat{\sigma^2}_{\mathrm{RSS}_i}}{\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\mathrm{RSS}_j}}\right\}x_i \quad \frac{1}{M}\sum_{i=1}^{M}\left\{\frac{\widehat{\sigma^2}_{\mathrm{RSS}_i}}{\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\mathrm{RSS}_j}}\right\}y_i\right]^T \\
&= [\hat{x}_{\mathrm{wc}} \;\; \hat{y}_{\mathrm{wc}}]^T = \hat{\mathbf{p}}_{\mathrm{wc}}\,. \qquad (4.22)
\end{aligned}$$

Also, we remark that unlike dCEN, the MSE performance of dWCEN based on distributed implementation of (4.19) is dependent on the coordinate reference system (*e.g.*, absolute (global) reference versus local reference) due to the cascade implementation. To eliminate this dependency, we choose one of the sensors in the AHSN (*e.g.*, the Mth) as reference sensor and reformulate the weighted average problem as follows:

$$\sum_{i=1}^{(M-1)}\kappa_i\mathbf{p}_i = \sum_{i=1}^{(M-1)}\kappa_i(\mathbf{p}_i - \mathbf{p}_M) + \mathbf{p}_M\,, \qquad (4.23)$$

where $0 \leq \kappa_i < 1$ and $\sum_{i=1}^{(M-1)}\kappa_i = 1$. Further details can be found in Appendix C.2.

We remark that using the one-pass algorithm of Sec. 3.3, the distributed implementation of WCEN reduces to three parallel distributed computations rather than the two parallel-cascade computations, *i.e.*, three parallel distributed computations at the ith node corresponding to the x, y components of $\hat{\mathbf{p}}_{\mathrm{wc}}$ and the ith weight component $\tilde{\alpha}_i = \frac{\widehat{\sigma^2}_{\mathrm{RSS}_i}}{\frac{1}{M}\sum_{j=1}^{M}\widehat{\sigma^2}_{\mathrm{RSS}_j}}$.

4.2.2 Distributed LS-based Estimation

Distributed LS

The same principles apply to constructing distributed LS-type (dLLS) estimators based on weighted sums. By exploiting

$$\frac{\mathbf{\Delta V}^T\mathbf{\Delta V}}{(M-1)} = \frac{1}{(M-1)}\sum_{j=1}^{M-1}(\mathbf{v}_j - \mathbf{v}_M)(\mathbf{v}_j - \mathbf{v}_M)^T \tag{4.24a}$$

$$\frac{\mathbf{\Delta V}^T\tilde{\mathbf{r}}}{(M-1)} = \frac{1}{(M-1)}\sum_{j=1}^{M-1}(\mathbf{v}_j - \mathbf{v}_M)\,\tilde{r}_j\ , \tag{4.24b}$$

where $\mathbf{v}_i = \Big[2(\mathbf{p}_i - \mathbf{p}_M)^T \ \ (\widehat{\sigma^{-2}}_{\mathrm{RSS}_i} - \widehat{\sigma^{-2}}_{\mathrm{RSS}_M})\Big]^T$ and $\tilde{r}_i = [\,\tilde{\mathbf{r}}\,]_i$ we may rewrite (4.8) into a form involving computations of averages (3.8). Distributed implementation of $\hat{\mathbf{x}}_{\mathrm{lls}}$ via (4.24) involves six parallel approximations for (4.24a) and three parallel approximations for (4.24b). At node i, the implementation steps are as follows:

(1) Initialize via locally available information, *i.e.*, sensor positions and RSS estimates: $\mathbf{f}_i[0] = \Big[f_i^{(1)}[0] \ \ \cdots \ \ f_i^{(9)}[0]\Big]^T$, where

$$\begin{bmatrix} f_i^{(1)}[0] & f_i^{(2)}[0] & f_i^{(3)}[0] \\ f_i^{(2)}[0] & f_i^{(4)}[0] & f_i^{(5)}[0] \\ f_i^{(3)}[0] & f_i^{(5)}[0] & f_i^{(6)}[0] \end{bmatrix} = (\mathbf{v}_i - \mathbf{v}_M)(\mathbf{v}_i - \mathbf{v}_M)^T \tag{4.25a}$$

$$\begin{bmatrix} f_i^{(7)}[0] \\ f_i^{(8)}[0] \\ f_i^{(9)}[0] \end{bmatrix} = (\mathbf{v}_i - \mathbf{v}_M)\,\tilde{r}_i\,. \tag{4.25b}$$

(2) Apply distributed computation algorithms of Sec. 3.2 for k iterations to obtain

the distributed LS estimate,

$$\hat{\mathbf{x}}_{i_{\text{dlls}}}[k] = \mathbf{f}_i[k] = \Big[\hat{x}_{i_{\text{dlls}}}[k] \;\; \hat{y}_{i_{\text{dlls}}}[k] \;\; \widehat{\sigma^2_s}_{i_{\text{dlls}}}\Big]^T . \tag{4.26}$$

As $k \to \infty$, the dLLS estimate at each node i converges to the global LS estimate, *i.e.*,

$$\hat{\mathbf{x}}_{i_{\text{dlls}}} = \lim_{n\to\infty} \mathbf{f}_i[k] = \Big[(\boldsymbol{\Delta}\mathbf{V}^T\boldsymbol{\Delta}\mathbf{P})^{-1}\boldsymbol{\Delta}\mathbf{V}^T\tilde{\mathbf{r}}_M\Big] = \Big[\hat{x}_{\text{lls}} \;\; \hat{y}_{\text{lls}} \;\; \widehat{\sigma^2_s}_{\text{lls}}\Big]^T = \hat{\mathbf{x}}_{\text{lls}} . \tag{4.27}$$

We remark that the distributed algorithm in (3.31), (3.39) or the one-pass algorithm (3.3) can be used to compute the nine terms in parallel.

Distributed WLS

Constructing distributed WLS-type (dWLS) estimators based on weighted sums is a straightforward extension of dLLS if the weight matrix is diagonal as described in (4.10) or approximated by $\hat{\mathbf{W}}$ in (4.13). Assuming $\hat{\mathbf{W}} = \text{diag}\Big\{\hat{w}_1, \cdots, \hat{w}_{M-1}\Big\}$ By exploiting

$$\frac{\boldsymbol{\Delta}\mathbf{V}^T\hat{\mathbf{W}}\boldsymbol{\Delta}\mathbf{V}}{(M-1)} = \frac{1}{(M-1)}\sum_{j=1}^{M-1}(\mathbf{v}_j - \mathbf{v}_M)^T(\mathbf{v}_j - \mathbf{v}_M)\,\hat{w}_j \tag{4.28a}$$

$$\frac{\boldsymbol{\Delta}\mathbf{V}^{\text{T}}\hat{\mathbf{W}}\,\tilde{\mathbf{r}}_M}{(M-1)} = \frac{1}{(M-1)}\sum_{j=1}^{M-1}(\mathbf{v}_j - \mathbf{v}_M)^{\text{T}}\,\tilde{r}_j\,\hat{w}_j \;, \tag{4.28b}$$

we may rewrite (4.9) into a form involving computations of averages in (3.8). Distributed implementation of $\hat{\mathbf{x}}_{\text{wls}}$ via (4.28) involves six parallel approximations for (4.28a) as in (4.25a) and three parallel approximations for (4.28b) as in (4.25b) with the additional constant multiplier $\hat{w}_j$ on the RHS of both (4.25a) and (4.25b).

Extensions of Distributed LS

The distributed implementation of LS-based estimators in Sec. 4.2.2 so far deal with the use of fixed (global) reference sensors (arbitrarily chosen as the Mth sensor) for range difference location estimation. In the AHSN setting, the use of a global reference sensor requires additional information that might not be readily available and/or additional computations might be costly as discussed in Sec. 4.1.4.

Distributed implementations of appropriate extensions of (4.9) can be readily developed, whereby each sensor employs one or more of its connected neighboring sensors as reference sensor(s). Assuming for example that each sensor uses only one randomly chosen reference sensor from its connected neighbors and letting $\ell(i)$ denote the index of the reference used by node i, an LS-type localization estimator amenable to distributed implementation similar to (4.24) is readily given by (4.8), by replacing $\mathbf{\Delta V}$ defined in (4.7) with

$$\mathbf{\Delta V} = \begin{bmatrix} 2(\mathbf{p}_i - \mathbf{p}_{\ell(1)})^T & (\widehat{\sigma^{-2}}_{\text{RSS}_i} - \widehat{\sigma^{-2}}_{\text{RSS}_{\ell(1)}}) \\ \vdots & \vdots \\ 2(\mathbf{p}_M - \mathbf{p}_{\ell(M)})^T & (\widehat{\sigma^{-2}}_{\text{RSS}_M} - \widehat{\sigma^{-2}}_{\text{RSS}_{\ell(M)}}) \end{bmatrix} \tag{4.29}$$

and $\tilde{\mathbf{r}}$ in defined in (4.7) with $\tilde{r}_i = \|\mathbf{p}_i\|^2 - \|\mathbf{p}_{\ell(i)}\|^2$. We remark that for these distributed extensions of LS algorithms can be M-dimensional (or higher) instead of $(M-1)$-dimensional for the fixed (global) reference algorithm.

4.2.3 Performance Metrics

In the following simulations, we employ as our figure of merit the sample-mean MSE (in dB m^2) performance of the associated localization (and later tracking) algorithms based on MC independent realizations. In addition to the MSE, we use the *MSE* *difference* (dMSE) and the *relative additional MSE* (rMSE) to compare distributed against centralized localization estimators and they are defined as follows:[2]

$$\mathrm{dMSE}[k] \triangleq 10\log_{10}\left(\frac{\mathrm{mse}_{\mathrm{dist}}[k]}{\mathrm{mse}_{\mathrm{cent}}}\right) \tag{4.30a}$$

$$\mathrm{rMSE}[k] \triangleq 10\log_{10}\left(\frac{\mathrm{mse}_{\mathrm{dist}}[k]-\mathrm{mse}_{\mathrm{cent}}}{\mathrm{mse}_{\mathrm{cent}}}\right), \tag{4.30b}$$

where $\mathrm{mse}_{\mathrm{dist}}[k]$ and $\mathrm{mse}_{\mathrm{cent}}$ denote the sample-mean MSE of the local sensor estimates after k iterations, and the MSE of the associated centralized algorithm respectively.

4.2.4 Distributed Source Localization Simulations & Analysis

We next (i) investigate the performance of the distributed localization estimators of Sec. 4.2 as a function of the sensor density D (via N or M_{ave}), the communication and fusion steps or number of iterations, k, and the network connectivity parameter d_o, and (ii) compare the performance of distributed implementations to that of their centralized counterparts. We model the acoustic source with spherical spreading (*i.e.*, $\beta = 2$), and assume that the source is located at the origin with N uniformly distributed detecting sensor nodes within a disk of radius R. We analyze the average source localization performance obtained via Monte Carlo simulations averaged over MC independently drawn sensor layouts.

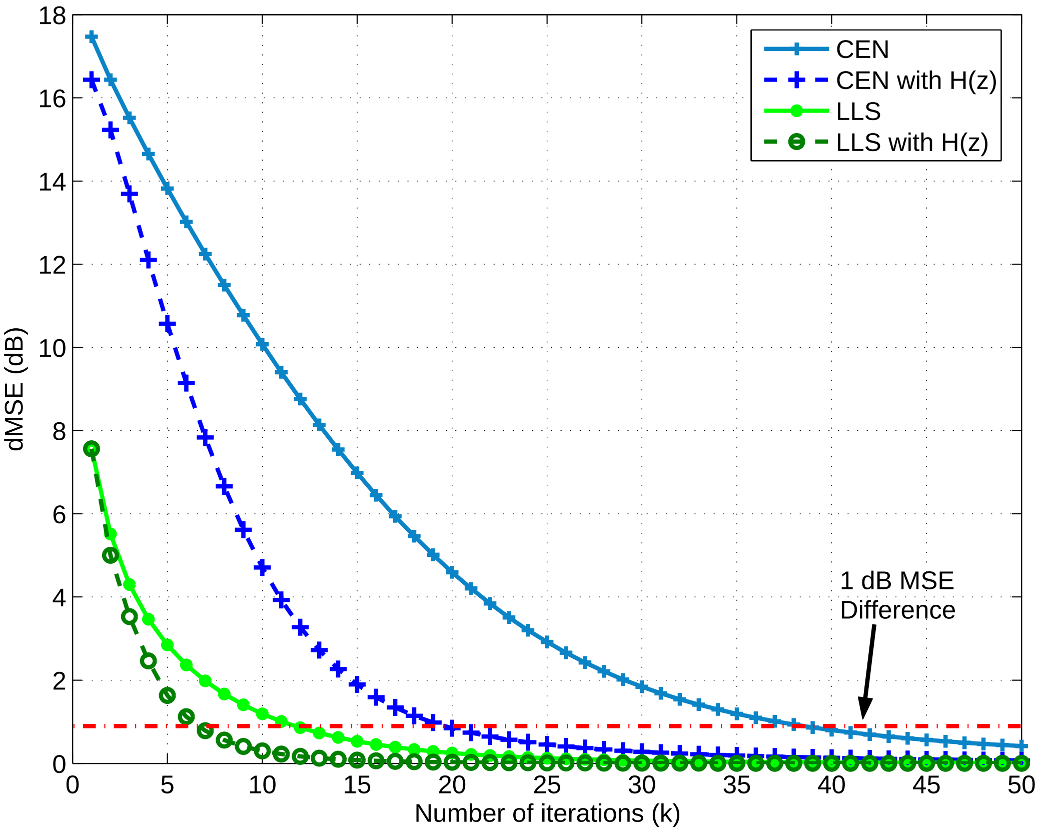


Figure 4.5: MSE difference between the distributed and the corresponding centralized centroid and LLS estimators vs. k with and without mode-shaping filter $H(z)$ with $c = 0.3$.

Centralized vs. Distributed

In the first three examples, we assume the stochastic signal model in (2.12) and distributed computation of average algorithm via first-order UD LTI rule in (3.31) with $\rho = \rho_{\max}$, $R = 100$, and $MC = 200$. The following parameter values are used for the first simulation example: $N = M = 50$, $d_o/R = 0.6$. Fig. 4.5 shows the rate of MSE convergence of the centroid and LS distributed estimators to the MSE of the associated centralized estimators in terms of their dMSE as a function k.

[2] The rMSE performance metric is initially defined in Sec. 3.4 but redefined here for convenience.

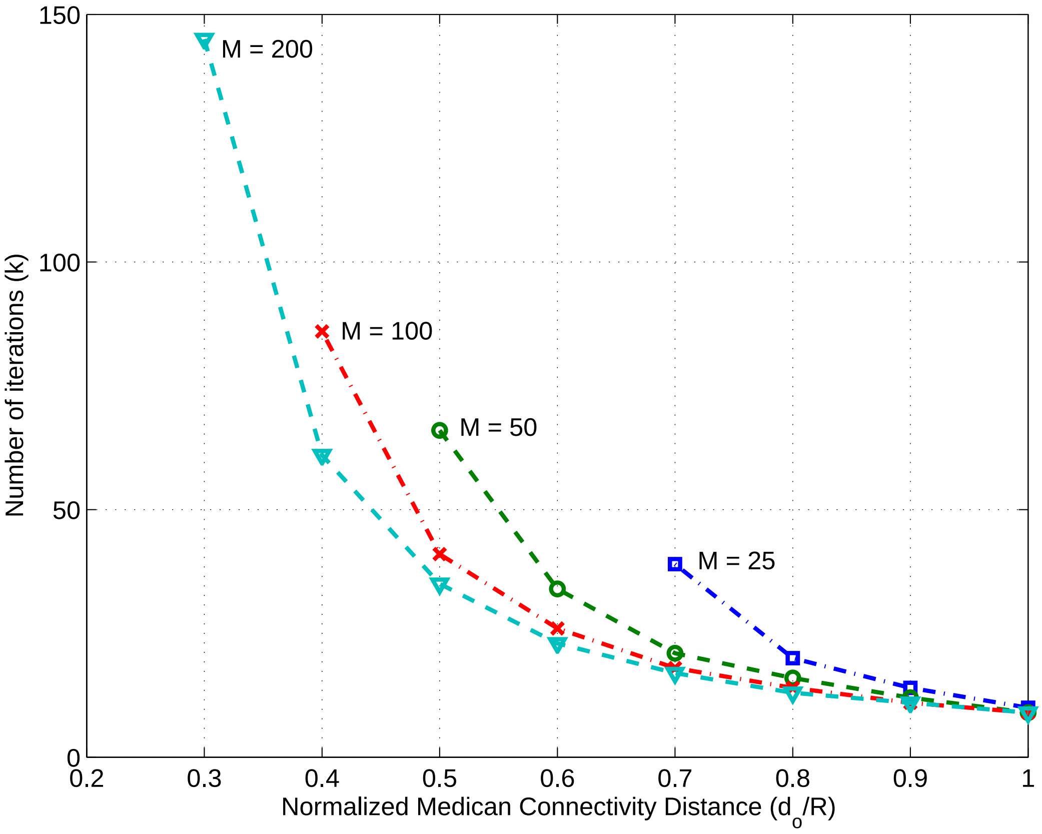


Figure 4.6: Number of communication and fusion steps (k_o) required vs. $(\frac{d_o}{R})$ for $M = \left[25, 50, 100, 200\right]$ sensors with -20 dB rMSE.

The convergence rates are shown for cases with and without a mode-shaping filter. As the figure reveals, both distributed algorithms converge relatively quick to their centralized counterparts, with the distributed LS algorithm converging much quicker ($e.g.$, $k \approx 12$ for 1 dB dMSE) than the distributed centroid algorithm ($e.g.$, $k \approx 38$ for 1 dB dMSE). The use of a mode-shaping filter $H(z)$ with $c = 0.3$, yields faster convergence for both types of estimators as indicated by the corresponding dashed-line curves in Fig. 4.5 (as predicted previously in Sec. 3.4). As the figure reveals, the number of steps required, k_o, for convergence to 1 dB dMSE is approximately reduced by a factor of two when $H(z)$ with $c = 0.3$ is applied.

In the second example, we compare the performance of centralized and distributed LLS estimators (with $H(z) = 1$) as a function of network and algorithm parameters, namely k_o, d_o (normalized by R) and M. Fig. 4.6 shows simulation results for k_o vs. $(\frac{d_o}{R})$ with $M = [25, 50, 100, 200]$ sensors for -20 dB rMSE (*i.e.*, 1% increase due to distributed implementation). As shown in the previous example, the number of steps needed for convergence to -20 dB can be significantly reduced by using a mode-shaping filter $H(z)$. Fig. 4.6 reveals several interesting trends about the properties of these distributed implementations:

1. As $(\frac{d_o}{R}) \rightarrow 1$, k_o is nearly the same ($k_o \approx 10$) for all four sensor densities. This is due to the fact that for $(\frac{d_o}{R}) \approx 1$, the network is nearly fully connected. In other words, all the data is essentially communicated to each node in a small number of iteration steps.

2. As $(\frac{d_o}{R}) \rightarrow 0$, $k_o \rightarrow \infty$ for all four sensor densities. This is due to the fact as $(\frac{d_o}{R})$ decreases, the network becomes increasingly sparsely connected, and eventually disconnected.

3. For intermediate range of $(\frac{d_o}{R})$ values (*e.g.*, $0.3 \leq (\frac{d_o}{R}) \leq 0.8$), the figure suggests various design trade-offs among the parameters M, k_o and d_o. For example, we may achieve higher localization performance with a fixed communication and power cost per node (*i.e.*, keeping k_o and d_o fixed) by increasing the sensor density. Alternatively, we can achieve the same MSE performance by increasing the transmit power per node (*i.e.*, increasing d_o) and employing a fewer number of iterations.

In the third example, we examine the dWCEN estimator as a function of the k_1 and k_2 parameters. Fig. 4.7 shows the MSE performance of the dWCEN estimator

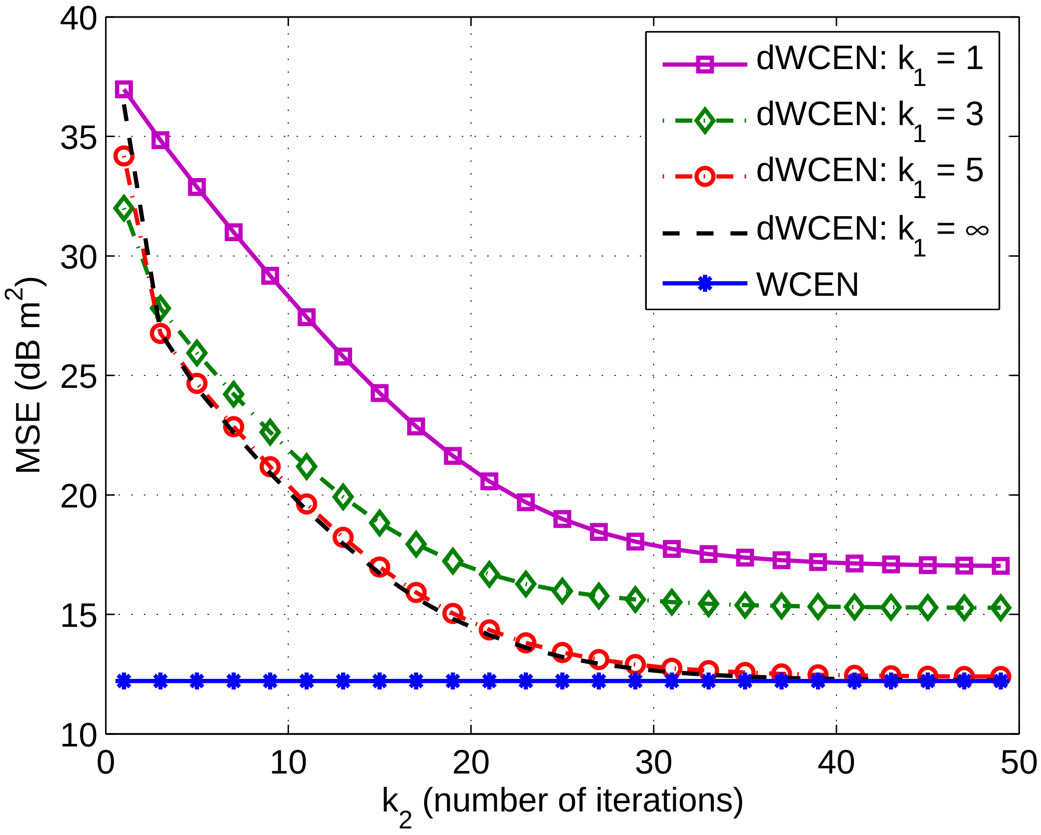


Figure 4.7: MSE vs. k_2 for WCEN estimator for $k_1 = 1, 3, 5, \infty$.

with respect to k_2 iterations (in the second cascade stage) for different k_1 iterations (in the first cascade stage). In this example, dWCEN with $k_1 = 5$ performs just as well as dWCEN with $k_1 = \infty$ with respect to the WCEN estimator. For smaller k_1 values (*e.g.*, $k_1 = 1$ and $k_1 = 3$), the dWCEN performs poorly with respect to WCEN; even as $k_2 \to \infty$, dWCEN does not asymptotically approach WCEN.

Extensions to Distributed LS

In the next three examples, we compare the performance of distributed LS algorithms with reference sensors other than the (global) "loudest" sensor node. The first two examples are based the stochastic signal models with $L = 1000$ snapshots and the third example is based on the deterministic signal model with $L = 1$ snapshot.

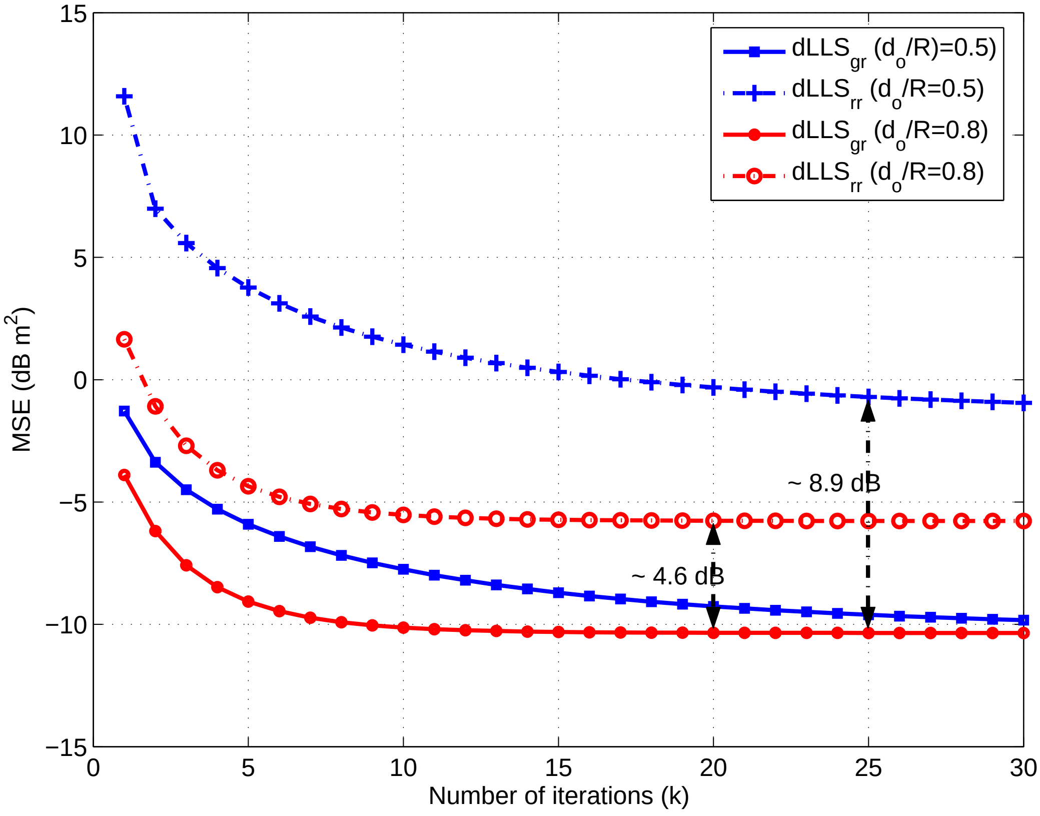


Figure 4.8: MSE comparison of distributed LLS estimators with fixed (global) reference (dLLS_{gr}) versus random reference as a (dLLS_{rr}) function of k for $(\frac{d_o}{R}) = 0.5$ and 0.8.

In the first example, we consider a stochastic signal model $R = 100$ m and $N = M = 50$ sensor nodes. We analyze and compare the MSE performance of dLLS that utilizes the loudest sensor as a global reference (denoted as dLLS_{gr}), and a dLLS estimator that employs locally and randomly selected reference sensors (denoted as dLLS_{rr}). Fig. 4.8 contrasts the MSE performance of dLLS_{gr} estimators to that of dLLS_{rr} estimators where the ith sensor randomly chooses one out of its $|\phi_{ii}|$ locally connected sensors as its reference. As the figure reveals, there is a performance gap between dLLS$_{\text{gr}}$ and dLLS$_{\text{rr}}$, which becomes wider at smaller d_o's (*e.g.*, 8.9 dB and 4.6 dB for $(\frac{d_o}{R}) = 0.5$ and 0.8, respectively). For a network of M sensors, the use of randomly selected local reference sensors can lead to a reduced number ($< M - 1$) of independent measurements. In addition, the inherent proximity of the sensor-reference pairs and the use of references with SNR inferior to that of the global reference sensor leads to higher sensitivity to measurement errors in $\widehat{\sigma^2}_{\text{RSS}_i}$'s. This performance gap is present in both centralized and distributed LS implementations, as revealed by the effectively constant MSE gap between distributed dLLS$_{\text{rr}}$ and dLLS$_{\text{gr}}$ estimators as a function of k.

In the second example, $R = 100$ m and $N = M = 50$ sensor nodes. We consider the MSE performance of dLLS$_{\text{rr}}$ estimators as a function of the average number of local references used by each sensor node in the network. In particular, we consider the case where each sensor node uses on-average γ randomly selected reference sensors where $1 \leq \gamma \leq 2$ (*i.e.*, the ith sensor randomly selects from the set of its connected neighboring sensors one reference sensor with probability $(1 - p_2)$ and two references sensors with probability p_2, and where, $p_2 = \gamma - 1$). Fig. 4.9 shows the simulated dMSE (MSE with random references versus MSE with a fixed reference) vs. $(\frac{d_o}{R})$, for $\gamma = \big[1,\ 1.5,\ 2\big]$. As the figure reveals, for a fixed $(\frac{d_o}{R})$, the performance improves with increasing γ, *i.e.*, the number of independent range difference measurements increases with increasing γ. Also, as the sensor communication range (*i.e.*, d_o) increases, the number of reference loops decreases and the sensor-reference pairs are on average farther apart, and, as a result, the dLLS$_{\text{rr}}$ MSE performance improves.

In the third example, we consider another extension to (4.8) that uses locally determined "loudest" sensors as references. In this approach, each sensor node de-

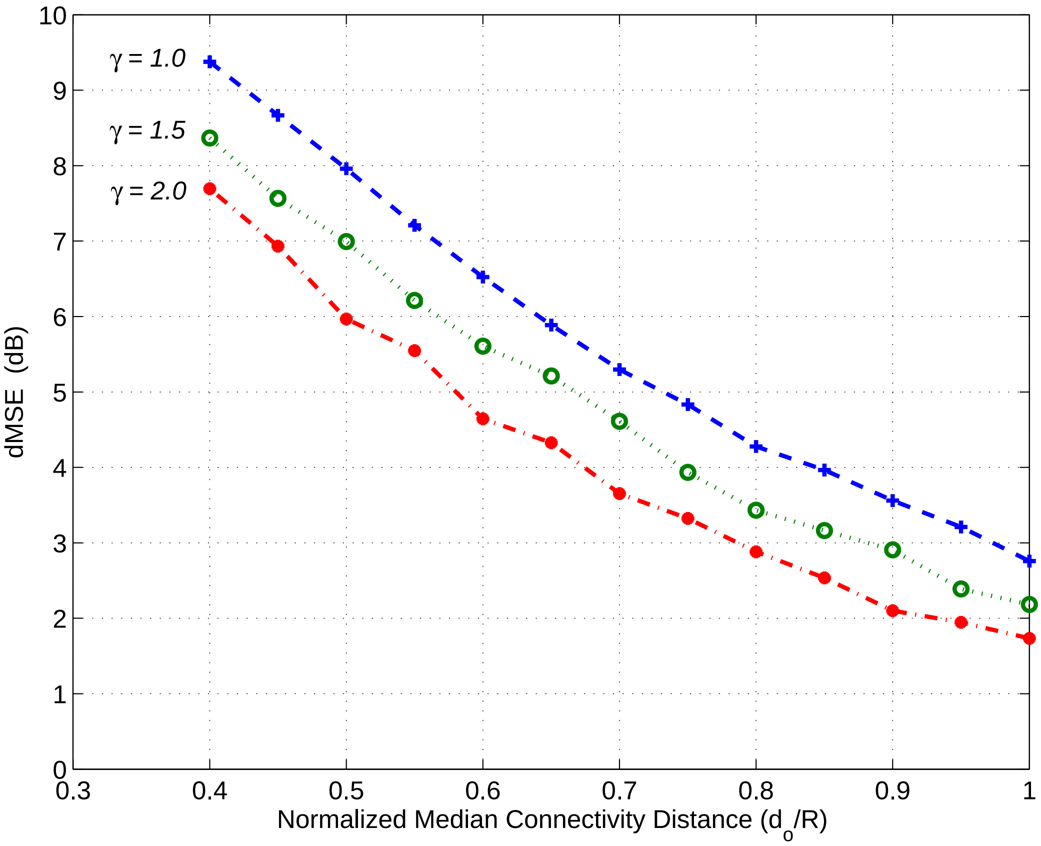


Figure 4.9: MSE difference vs.$(\frac{d_o}{R})$ for centralized LLS with random references (LLS$_{\text{rr}}$) with respect to the centralized LLS with the loudest sensor as a global reference (LLS$_{\text{gr}}$).

termines which of its connected neighbors is the "loudest" and use that node as its reference. Specifically, $\forall j \longleftrightarrow i$ and $j \neq i$, the local reference node for the ith node, $\ell(i) = \arg\max_j \{\sigma^2_{\text{RSS}_j}\}$. We denote the WLS estimator that uses locally determined "loudest" sensors as local references as WLS$_{\text{lr}}$. Fig. 4.10 shows the MSE performance for WLS$_{\text{lr}}$ and WLS$_{\text{gr}}$ as a function of the nominal connectivity distance, d_o, normalized by R, for the case of SNR $=$ 60 dB, σ^2_T $=$ 29 dB, N $=$ 200 nodes and R $=$ 100 m (D $=$ 0.0064 node/m^2) over MC $=$ 200 independent realizations. As the figure reviews, for $(\frac{d_o}{R}) \lessapprox 0.55$, WLS$_{\text{gr}}$ performs slightly better than WLS$_{\text{lr}}$. For

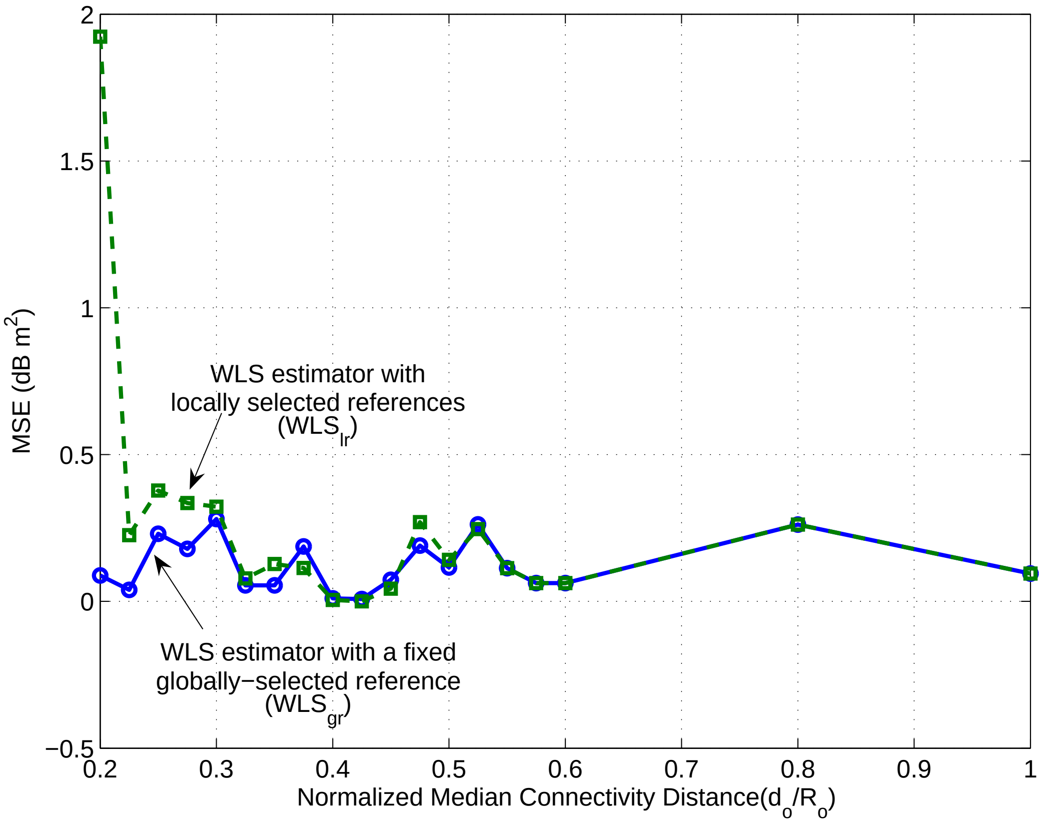


Figure 4.10: MSE performance for WLS estimator with locally selected "loudest" references ($\mathrm{WLS_{lr}}$) and LS-estimator with a fixed globally selected "loudest" reference ($\mathrm{WLS_{gr}}$) vs. $(\frac{d_o}{R})$.

$(\frac{d_o}{R}) \gtrapprox 0.55$, the two estimators have the similar MSE performance. This is due to the fact that at high $(\frac{d_o}{R})$, the nodes in the ACN are nearly fully or fully connected and therefore, the locally determined "loudest" references are essentially the same as the global "loudest" reference.

The use of locally selected "loudest" nodes as references require slightly more computation but it alleviates the need for a fixed global reference and is amenable to distributed implementation. In this example, the neighboring nodes are one-communication step away, however, the algorithm can be generalized to neighboring nodes that are k-communication steps away. We remark that the extensions of the distributed LLS are based on M-dimensional instead of $(M-1)$-dimensional range difference equations

4.3 Performance Analysis & Algorithm Design

Results from previous sections show the interdependence of sensor network parameters (*e.g.*, density), signal parameters (*e.g.*, SNR) and localization algorithm parameters (*e.g.*, detection threshold) on estimation performance. In this section, we investigate the relationship among the various network, signal and algorithm parameters with the goal of developing design rules for predicting the MSE performance.

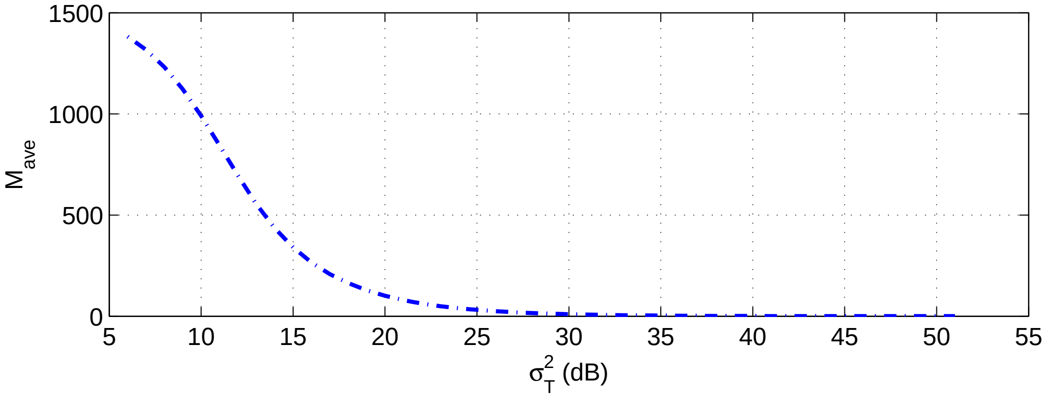
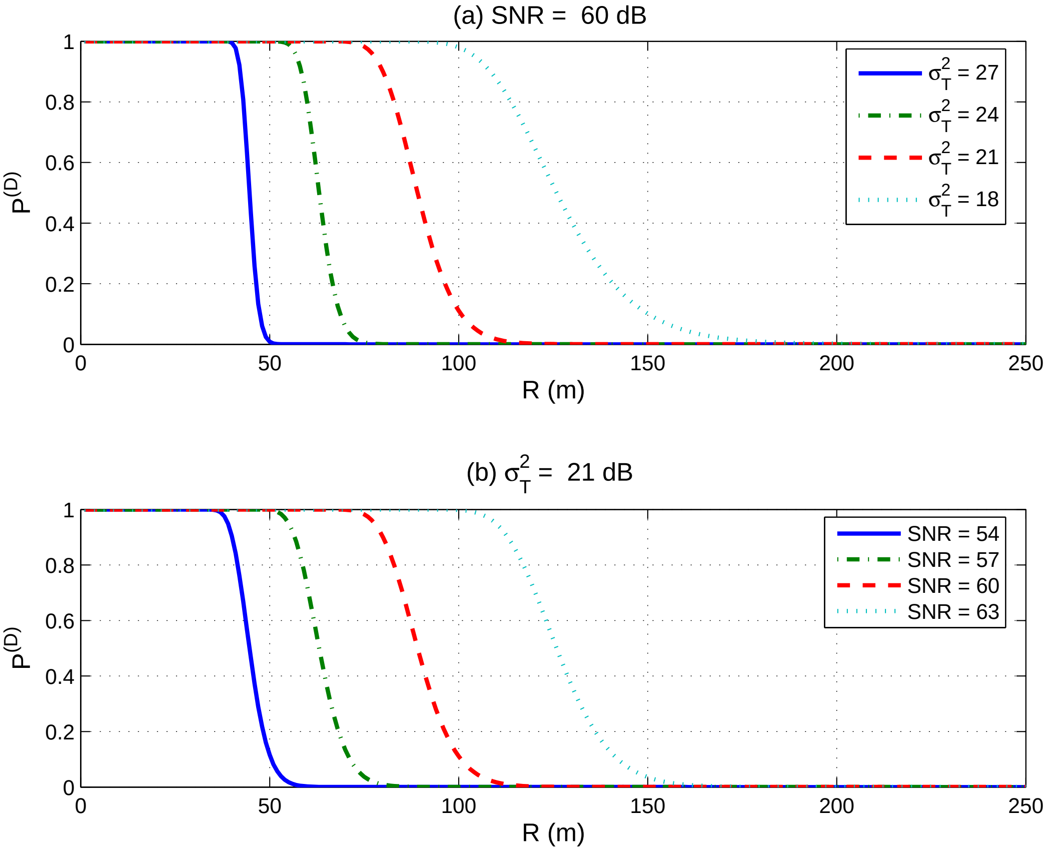


Figure 4.11: M_{ave} versus σ_T^2 for SNR = 60 dB.

To this end, we investigate the performance of the (centralized) WLS source localization algorithm with respect to SNR, σ_T^2, and D based on the signal model described by (2.13). For convenience, we set $\sigma_\eta^2 = 1$ so SNR $= \sigma_{\mathrm{s}}^2$ and we drop the dependence of the estimators on t. We simulate a baseline case for $D = 0.0032$, first with $N = 1600$ nodes uniformly distributed with in a circle of radius $R = 400$ m,



(a) SNR = 60 dB

(b) σ_T^2 = 21 dB

Figure 4.12: (a) $P^{(\mathrm{D})}$ vs. R at $\sigma_T^2 = \left[18,\ 21,\ 24,\ 27\right]$ dB for fixed SNR $= 60$ dB, and (b) $P^{(\mathrm{D})}$ vs. R at SNR $= \left[54,\ 57,\ 60,\ 63\right]$ dB for a fixed $\sigma_T^2 = 21$ dB.

while varying SNR and σ_T^2 over $MC = 500$ independent realizations. For a given SNR level, the average number of detecting nodes, M_{ave}, in the sensor network depends on σ_T^2. As σ_T^2 increases (decreases), M_{ave} decreases (increases) exponentially as shown in Fig. 4.11. This is due to the fact the probability of detection, $P^{(\mathrm{D})}$, for each node defined in (2.15) depends on σ_T^2 and its relative distance from the source. For example, at a lower detection threshold, the probability of detection is higher for all the nodes resulting in higher M_{ave} values.

Fig. 4.12 illustrate the relationship between $P^{(\mathrm{D})}$ versus the range or radius

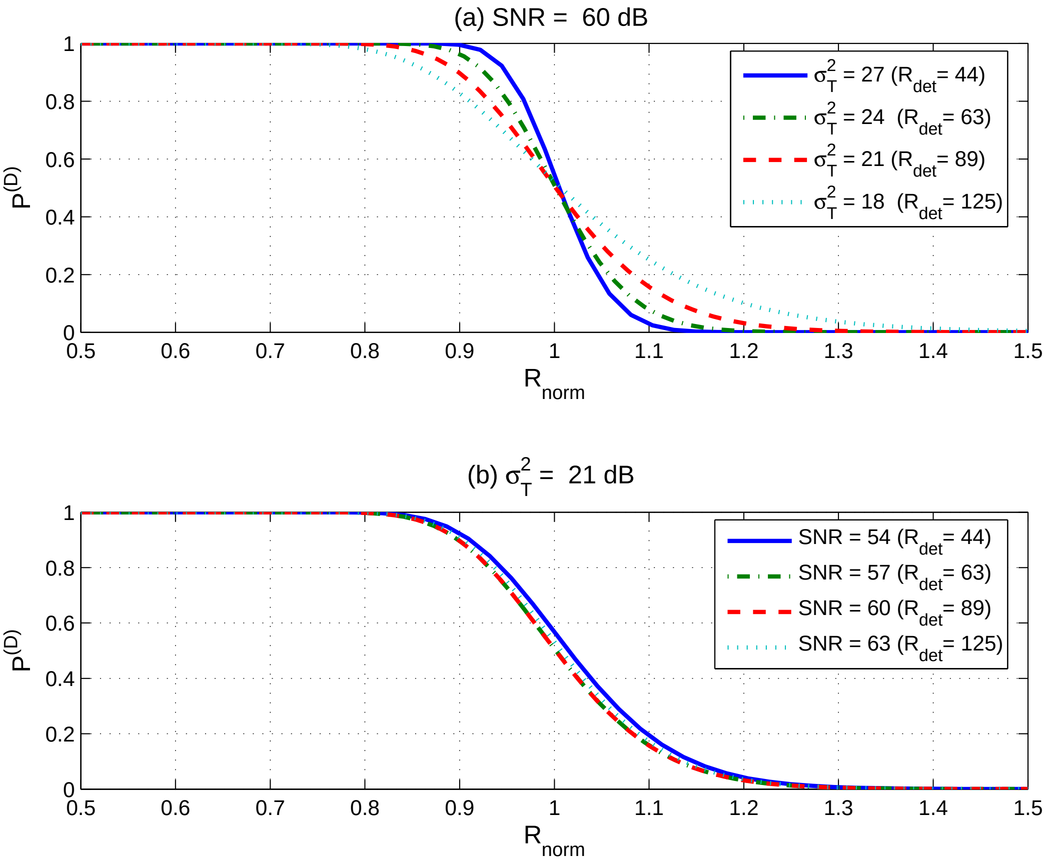


Figure 4.13: (a) $P^{(\mathrm{D})}$ vs. normalized R at $\sigma_T^2 = \left[18,\ 21,\ 24,\ 27\right]$ dB for fixed SNR $= 60$ dB, and (b) $P^{(\mathrm{D})}$ vs. normalized R at SNR $= \left[54,\ 57,\ 60,\ 63\right]$ dB for a fixed $\sigma_T^2 = 21$ dB.

R: (a) for varied detection thresholds, $\sigma_T^2 = \left[18,\ 21,\ 24,\ 27\right]$ dB at a fixed SNR $= 60$ dB and (b) for varied SNR $= \left[54,\ 57,\ 60,\ 63\right]$ dB at a fixed $\sigma_T^2 = 21$ dB. For ease of comparison, we normalized range so that the all the plots intersect at $P^{(\mathrm{D})} = 0.5$ and the normalized range $R_{\mathrm{norm}} = 1$ as shown in Fig. 4.13. The figures reveal two important trends as σ_T^2 is decreased: (i) the radius of detection, denoted by R_{det}, is increased as shown in Fig. 4.12(a); and (ii) the transition region between detection and non-detection regions (*i.e.*, from $P^{(\mathrm{D})} \approx 1$ to $P^{(\mathrm{D})} \approx 0$) is decreased as shown in Fig. 4.13(a). Similarly, figures reveal similar trends as SNR is increased (i) the

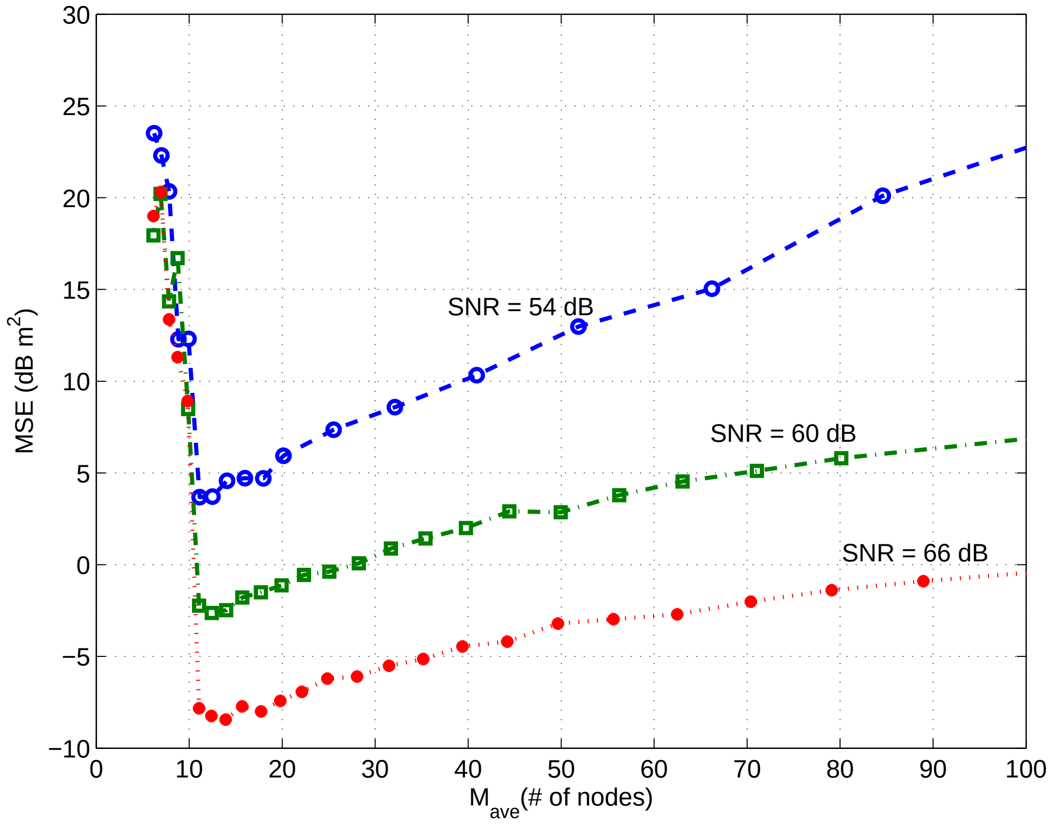


Figure 4.14: MSE performance for WLS versus M_{ave} for SNR $= \left[54,\, 60,\, 66\right]$ dB.

radius of detection is increased as shown in Fig. 4.12(b); and (ii) the transition region between detection and non-detection regions ($i.e.$, from $P^{(\text{D})} \approx 1$ to $P^{(\text{D})} \approx 0$) however stays the same as shown in Fig. 4.13(b).

We next analyze the MSE performance of the WLS estimators as a function of M_{ave} for a range of SNR levels. As Fig. 4.14 reveals, higher SNR yields better MSE performance. In addition, there is an "optimal" M_{ave} (and the associated optimal σ_T^2), denoted as M_{ave}^* (and σ_T^{2*}), that corresponds to the lowest MSE at each SNR level ($e.g.$, $11 \leq M_{\text{ave}}^* \leq 14$). We remark that at very low threshold levels, the set of detecting nodes is larger than at high threshold levels as shown in Fig. 4.11. However, these

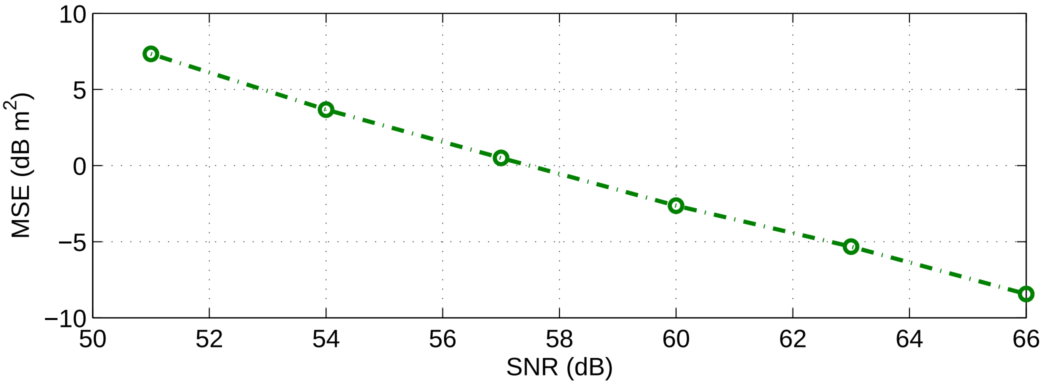


Figure 4.15: MSE performance at $M_{\text{ave}} = M^*_{\text{ave}}$ versus SNR for $\hat{\mathbf{p}}_{\text{WLS}}$.

detecting nodes are spread out over a larger area with the distant nodes providing poorer measurements, resulting in location estimates with higher MSE's. Fig. 4.15 shows the linear relationship between MSE performance at $M_{\text{ave}} = M^*_{\text{ave}}$ and SNR. As a result, there is an inherent network scaling relationship for this source location estimator.

The linear dependence of the MSE performance on SNR (and σ_s^2) for the WLS estimator can be exploited to predict the source localization performance as the sensor network scales in terms of network density D (*i.e.*, $N/\pi R^2$). Recall that the signal power is $\sigma_s^2 = \text{SNR}/\sigma_\eta^2$ and the RSS at the ith sensor is $\sigma^2_{\text{RSS}_i} = \sigma_s^2/r_i^2$. So for example, if the sensor density is doubled to $\tilde{D} = 2D$ via either doubling the number of sensor node to $\tilde{N} = 2N$ or reducing the radius to $\tilde{R} = R/\sqrt{2}$ (resulting in $\tilde{r}_i = r_i/\sqrt{2}, \forall i$), then the MSE performance is expected to improve by 6 dB. Fig. 4.16 illustrates the scaling relationship via simulations for three different SNR and network settings resulting in similar MSE performances at $M_{\text{ave}} = M^*_{\text{ave}}$.

Figs. 4.14 – 4.16 collectively suggest a method for predicting the source-localization performance based on the MSE performance over a reference network.

In particular, letting $\mathrm{MSE}_{\mathrm{ref}}$, $\mathrm{S}_{\mathrm{ref}}$, and D_{ref} denote the reference MSE, SNR and density quantities respectively. Then, the predicted MSE performance for a sensor network with density D is given by

$$\mathrm{MSE} \approx \mathrm{MSE}_{\mathrm{ref}} + (\mathrm{SNR}_{\mathrm{ref}} - \mathrm{SNR}) + 20\log_{10}(\frac{D_{\mathrm{ref}}}{D}), \tag{4.31}$$

where $\mathrm{MSE}_{\mathrm{ref}}$, $\mathrm{SNR}_{\mathrm{ref}}$, and D_{ref} are the reference MSE, SNR and density quantities respectively.

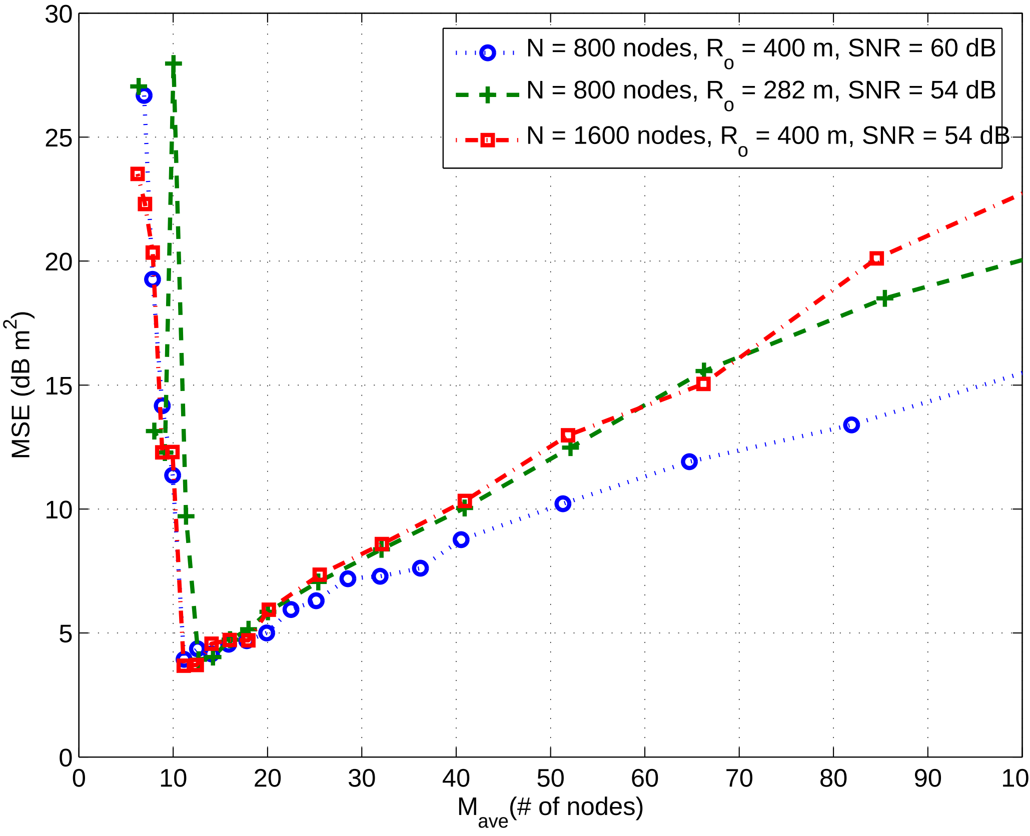


Figure 4.16: MSE vs. M_{ave} simulation example for the WLS source location estimator illustrating the scaling relationship among sensor network parameters.

Next, we investigate the performance of the distributed source location estimators as a function k and the network connectivity density, defined as $\nu = N \cdot d_o^2/R^2$, and compare the performance of distributed implementations to that of their centralized counterparts over $MC = 200$ independent realizations. In each realization, a network is first generated with $N = 200$ nodes uniformly distributed within a circle of $R = 200$ m ($D = 0.0016$ node/m^2). A network topology is generated according to (3.2) with $m = 2$. Initially (*i.e.*, prior to performing source localization), a set of base ρ_{ij}'s are computed for the entire network by applying the LN algorithm of Sec. 3.2 for $k_{in} = 20$ iterations. For convenience, the source is placed at the origin of the network.

In the following simulation example, we use the deterministic signal model and set $\nu = \left[10,\ 20\right]$ and SNR = 57 dB with the associated σ_T^{2*} (M_{ave}^*) for best localization performance.

Fig. 4.17 shows the rate of MSE convergence of the LS distributed estimators to the MSE of the associated centralized estimators as a function of k. As the figure reveals, the distributed WLS algorithm converge quickly to its centralized counterpart. For example, the number of steps required, k_o, for convergence to 1 dB dMSE are $k_o = 8$ and 5 for $\hat{\mathbf{p}}_{d\text{lls}} \rightarrow \hat{\mathbf{p}}_{\text{lls}}$ respectively for $\nu = 10$ and 20.[3]

4.4 Sensor Fusion in Space Summary

In summary, we developed low-complexity centroid and LS algorithms for source localization. We compared and contrasted localization performance of the estimators against each other and against the associated CRB. We found that the our estimators

[3] Rate of convergence result reported in [19] for the stochastic signal model shows that distributed LLS converging faster to its centralized counterpart than CEN. However, the LLS used in [19] assumes that σ_s^2 is known.

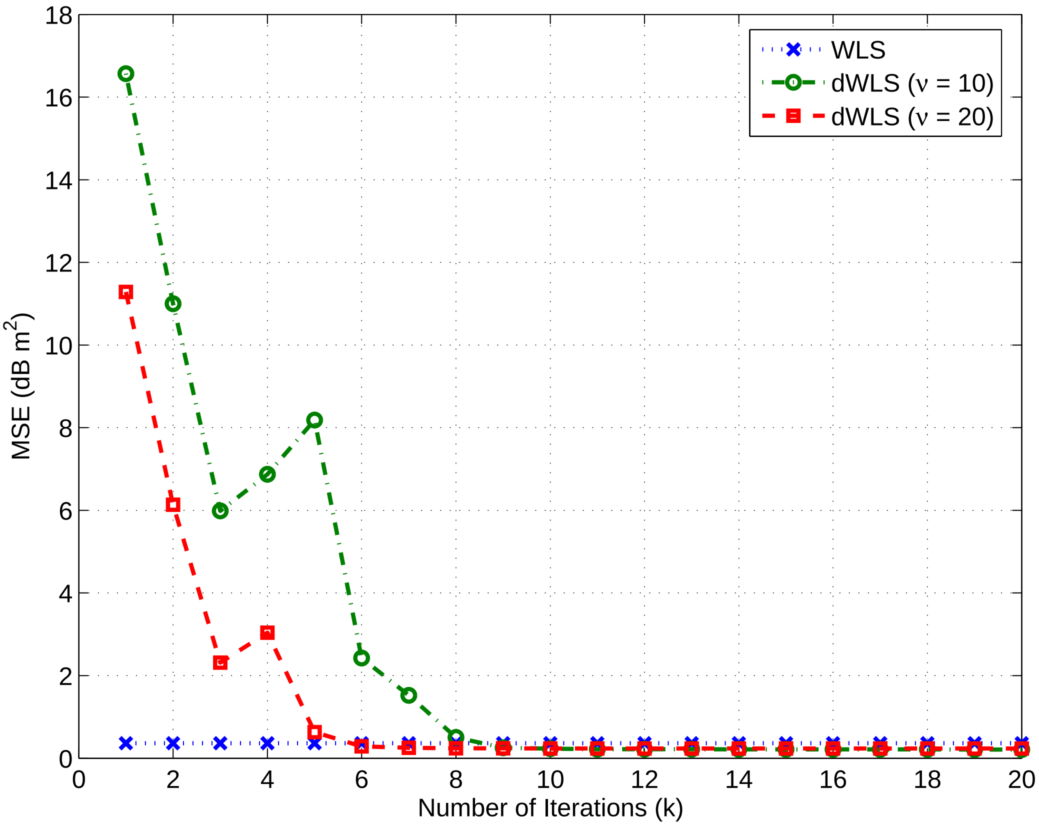


Figure 4.17: MSE difference between the distributed and the corresponding centralized WLS estimator vs. k.

perform well over a wide range of SNR's and sensor densities. The MSE performance improves with increasing sensor density, approximately by 3 dB when M_{ave} is doubled; however, they are SNR-limited due the fact that the estimators are biased. We found that LS-based algorithms outperform centroid-based algorithms with $>$ 20 dB m^2 MSE improvement. For LS-based algorithms, we also analyzed performance of the algorithms with the reference sensor being the closest sensor ("loudest") to the source or a randomly chosen reference sensor. We found that there is no difference for the LLS estimators while there is a difference of a few dB for WLS estimators.

Chapter 5

Fusion in Time

In this chapter, we design and develop sensor fusion algorithms for space-time processing via a distributed network of sensors. In particular, we are interested distributed signal processing algorithms for decentralized tracking of a moving source within a large scale sensor network. As with localization, deployment of distributed sensors use for tracking has received wide interests due to the large spatial-coverage capabilities. Some examples of civilian applications include vehicular traffic monitoring on roads and highways, and detection and tracking of people movement in secure areas; and examples of military applications include remote border surveillance and battlefield reconnaissance. Recent distributed tracking methods of interest include particle filtering [60,68] and graphical-based methods [62,69]. However, the predominant distributed approaches proposed in literature are based on Kalman filtering [3,70–72]. This is due to the fact that the Kalman filter (KF) based approaches are relatively easier to implement in comparison to the other methods and due to the pioneering work of Durrant-Whyte and others in the development of (fully) decentralized KF (DKF) architectures for multi-sensor data fusion [14,15,49,50]. The principal approach that Durrant-Whyte and others took is to reformulate the KF state estimation problem into an information form [50].

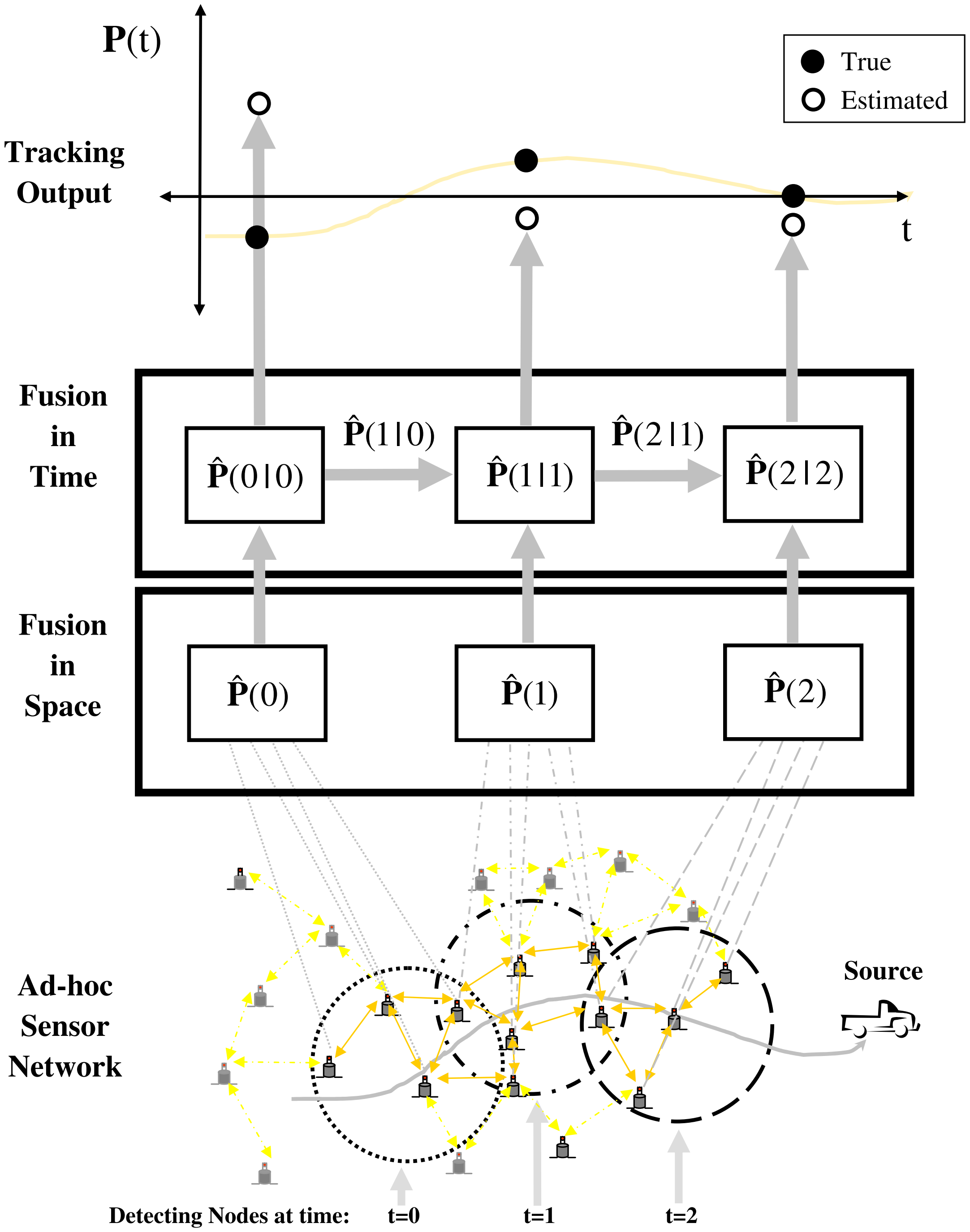


Figure 5.1: One-dimensional distributed tracking concept in (large-scale) decentralized networks via a sequence of subnetworks (*e.g.*, ACN's): (i) fusion in space via distributed source localization, and then (ii) fusion in time via distributed tracking.

In a centralized framework, a central fusion center has all the information it needs to perform (centralized) tracking, *e.g.*, the observation data, tracking parameters and previous state estimates. In a decentralized framework where the set of detecting nodes and the associated ACN vary in time with respect to the moving source, it is not clear how the data can be communicated, routed and fused. Conceptually, we first need to perform distributed data fusion in space to obtain source location estimates, and then uses the location estimates and prior prediction estimates to perform distributed data fusion in time to obtain tracking estimates as illustrated in Fig. 5.1. In our investigation, we present a systematic framework for performing distributed tracking in a large-scale sensor network. In particular, our approaches use a general KF framework for state estimation that exploits the improved distributed computations algorithms developed in Chap. 3 for performing both data fusion in space and fusion in time locally at each participating node.

We first discuss two approaches to modeling and implementing distributed tracking in large-scale sensor network in Sec. 5.1. We next present and discuss the distributed tracking algorithm based on KF in Sec. 5.2. Then, we present tracking simulation results and performance analysis in Sec. 5.3.

5.1 Sensor Fusion in Time Modeling

5.1.1 Tracking via Entire Sensor Network

As the acoustic source moves within the sensor network, at any time instant t there is a set of active nodes or *active contributors* (*e.g.*, nodes in the network that have detected the source) and a set of non-active nodes or *non-contributors* (*e.g.*, nodes in the network that have not detected the source). All active contributors and non-contributors are connected together and their connection topology is described by $N \times N$ matrix Φ in(3.1), however, only the active contributors at time t are the ones that distributively compute the location and prediction estimates at time t. The roles of the non-contributors in distributed tracking can varied depends on their interactions with the neighboring active contributors. For instance, they can participate in routing, fusing of data from neighboring active contributors and/or both.

In this setup, distributed source localization and tracking can be performed via the one-pass algorithm (described in Sec. 3.3). First, let $\mathcal{U}_{ac}$ denotes the set of active contributors, $\mathcal{U}_{nc}$ denotes the set of non-contributors and $\mathcal{U}_{en} = \mathcal{U}_{ac} \cup \mathcal{U}_{nc}$ denotes the set of all nodes in the entire network with $|\mathcal{U}_{en}| = N$. Next, let a_i denotes the "weight" of sensor i at a fixed time instant t defined by the following:

$$a_i = \begin{cases} 1 & \text{if } i \in \mathcal{U}_{ac} \text{ at time } t \\ \\ 0 & \text{if } i \in \mathcal{U}_{nc} \text{ at time } t \,. \end{cases} \tag{5.1}$$

Then, we can model the problem as

$$G(\mathbf{f}(\mathbf{z}_{ac})) = \frac{1}{|\mathcal{U}_{ac}|} \sum_{i=1}^{|\mathcal{U}_{ac}|} f_i(z_i) = \sum_{i=1}^{|\mathcal{U}_{en}|} \left\{ \frac{a_i}{\sum_{j=1}^{|\mathcal{U}_{en}|} a_j} \right\} f_i(z_i) \tag{5.2}$$

(note that $\sum_{j=1}^{|\mathcal{U}_{en}|} a_j = |\mathcal{U}_{ac}|$), and view the entire network as if it is an "active" network where each node in the network has "detected" the source and not just the subnetwork of active contributors as considered previously. From (5.2), we can directly apply the one-pass algorithm for distributed computations of weighted averages. We remark that if RSS-weighted estimators such as as WCEN are employed for source localization, the weights used in the one-pass algorithm are the compounded weights *i.e.*, $\tilde{\alpha}_i = a_i \, \alpha_i$, $\forall i$.

The key advantage to modeling in (5.2) is the ease of tracking implementation. In this setup, each node is essentially "on", active and communicates bi-directionally with its neighbors without distinguishing whether the neighboring nodes as active contributors and non-contributors. Since the network is essentially "fixed", we can consider network parameters (*e.g.*, UD ρ) based on macroscopic information for the entire network in addition to local network parameters (*e.g.*, NUD ρ_{ij}'s) based on the varying active subnetworks. Furthermore, by considering one large network with connected topology, we don't have to be concerned if the subnetwork of active contributors at each time instant has connected topology or not. The main disadvantage to modeling in (5.2) is the additional (potentially inefficient) use of power and communication resources that are incurred for utilizing the entire network. The total resource consumed is proportional to N, and N can be much larger than $M(t)$.

5.1.2 Tracking via Sequence of ACN's

As discussed Sec. 3.1, we can view the large-scale sensor network of uniformly distributed sensors with network topology described by $\tilde{\Phi}(t)$ in (3.4a) in which available connections for performing computations are among the set of nodes in $\mathcal{I}(t)$. The size and number of *active* nodes in the ACN vary with time and are dependent on the source's trajectory and signal power via the RSS's. The RSS at each node in the AHSN is inversely proportional to a (nonlinear) function of range. As a result, nodes that are outside of the ACN, referred to as *nonactive* nodes, have low RSS's (low quality data) and provide negligible contribution to the source location estimation. However, the nonactive nodes in the vicinity of the ACN can receive broadcasted (estimated) source location information from the neighboring active nodes in the ACN. As shown in Fig. 5.2, as the source moves (at reasonable speeds) from point-A to point-B at time t, some of active nodes in the ACN at time $t-1$ remain active in the new ACN at time t. This is indicated by the intersecting region of the two successive ACN's. The nodes in the intersecting region can broadcast source location information at time $t-1$ to the other remaining active nodes in the ACN at time t. As the result, all of the nodes in the ACN have sufficient prior and current information to perform tracking individually at each node.

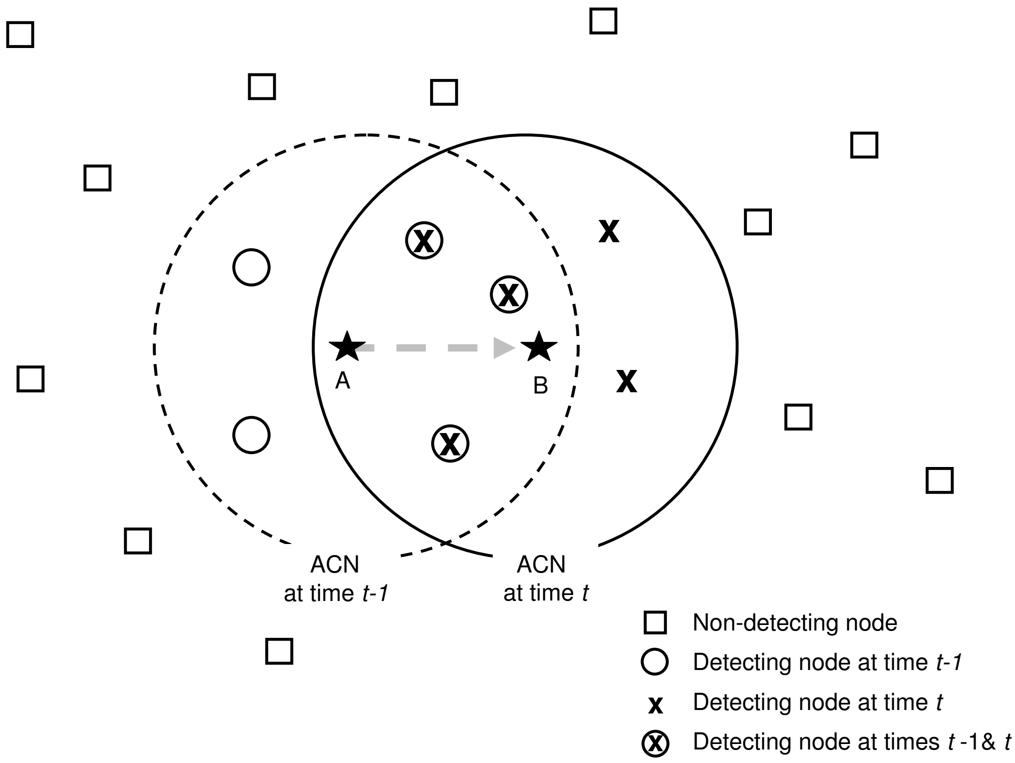

A
B
ACN
at time t-1
ACN
at time t
□ Non-detecting node
○ Detecting node at time t-1
x Detecting node at time t
ⓧ Detecting node at times t -1& t

Figure 5.2: A source moves from location A, where it was detected by nodes in ACN at time $t-1$, to location B, where it is detected by nodes in ACN at time t. The intersecting region contains nodes that have detected the source at both locations and times.

Alternatively, we can focus on the subnetwork, ACN, formed by the nodes in $\mathcal{I}(t)$ described by network topology $\Psi(t)$ in (3.5a) with the following assumptions: (i) at the outset, the N nodes in the entire network self-organize with the set of NUD parameters ρ_{ij}'s pre-computed for the entire network,[1] and then (ii) at each time t, the ρ_{ij}'s are further refined for the ACN. The distributed tracking is performed over a sequence of ACN's, where KF is employed at all of the nodes in $\mathcal{I}(t)$. In subsequent discussion, we use the ACN model characterized by $\{\mathcal{I}(t), \Psi(t)\}$.

5.2 Distributed Tracking via KF

Our approach relies on obtaining $Z_n(t)$, a sequence of snapshot estimates of $P_{sn}(t)$, the sequence of source locations, based on measurements collected by nodes in the sensor network, and viewing the estimate $Z_n(t)$ at time t as a *single* (location) measurement equation, *i.e.*, for $n = 1$, 2

$$Z_n(t) = P_{sn}(t) + R_n(t) \tag{5.3}$$

where the observation noise $R_n(t)$ denotes the snapshot estimation error, assumed to be a zero-mean white sequence, independent of $A_n(t)$, and with power equal to the field-averaged mean-square error of the estimator and denoted by $\mathbf{R} = \sigma_R^2$. We next discuss the distributed KF filter-based algorithms that fuse temporal information over a sequence of ACN's to provide tracking estimates. The model that we employ for developing tracking algorithms exploits the fact that, at any given time t, sensors in the sensor field obtain measurements, based on which they can form distributed source localization estimates of the source as demonstrated in Sec. 4.2. In particular, the tracking algorithms we develop view the resulting source position estimates, $Z_n(t)$, nth dimension ($n = 1$, 2) at time t, as observations in a *single* measurement equation. For each $n \in \{1, 2\}$, we remark that the scalar source-location estimate error sequences

[1] The entire network can also self-organize with UD rules with parameter ρ.

defined as

$$e_n(t) = Z_n(t) - P_{sn}(t)\,, \tag{5.4}$$

is in general, correlated in time (t). To account for this temporal correlation, we present a more expanded state-measurement model than the one described in (2.1)–(5.3). We model $e_n(t)$ (for n = 1, 2) as a pth order autoregressive (AR(p)) process. It is assumed that the AR parameters of the process $e_n(t)$, *i.e.*, its order p, the $p \times 1$ vector $\mathbf{a}_p = \begin{bmatrix} a_p(1) & a_p(2) & \cdots & a_p(p) \end{bmatrix}^T$, and the associated innovation process power, σ_U^2, are first estimated during a training mode.

Letting $E_n(t) = \begin{bmatrix} e_n(t) & e_n(t-1) & \cdots & e_n(t-(p-1)) \end{bmatrix}^T$, the dynamics of the state is now $X_n(t) = \begin{bmatrix} X_{sn}^T(t) & E_n^T(t) \end{bmatrix}^T$ for n = 1, 2, are described by the following equation

$$X_n(t+1) = \mathbf{F}\, X_n(t) + \mathbf{G}\, B_n(t),\ t = 0, 1, \ldots \tag{5.5a}$$

where $\mathbf{F}$ and $\mathbf{G}$ are now

$$\mathbf{F} = \begin{bmatrix} 1 & T_s & \mathbf{0}_p^T \\ 0 & \varrho(T_s) & \mathbf{0}_p^T \\ \mathbf{0}_p & \mathbf{0}_p & \mathbf{A}_p \end{bmatrix}, \ \mathbf{G} = \begin{bmatrix} 0 & 0 \\ T_s & 0 \\ \mathbf{0}_p & \mathbf{u}_p \end{bmatrix}, \tag{5.5b}$$

$\mathbf{0}_k$ is $k \times 1$-vector of 0's, $\mathbf{u}_p = [1\ \mathbf{0}_{p-1}^T]^T$,

$$\mathbf{A}_p = \begin{bmatrix} \mathbf{a}_p^T \\ I_{(p-1)} \quad \mathbf{0}_{p-1} \end{bmatrix}, \tag{5.5c}$$

I_k is the $k \times k$ identity matrix, $B_n(t) = \begin{bmatrix} A_n(t) & U_n(t) \end{bmatrix}^T$, and where $A_n(t)$ and the snapshot-estimate error innovation process $U_n(t)$ are uncorrelated zero-mean white sequences, *i.e.*, $E\left\{B_n(t)\, B_n^T(\tau)\right\} = \mathbf{Q}\delta[t-\tau]$, and where $\mathbf{Q} = \mathrm{diag}(\sigma_A^2, \sigma_U^2)$.

For each $n \in \{1, 2\}$, the associated (scalar) measurement equations are now given by

$$Z_n(t) = \mathbf{H}\, X_n(t) = P_{sn}(t) + e_n(t) \tag{5.6}$$

where $\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & \mathbf{0}_{p-1}^T \end{bmatrix}$. We remark that the case where the snapshot error sequences $e_n(t)$ can be accurately modeled as white (*i.e.*, $p = 0$) is also captured by the model (5.5)–(5.6) by setting $p = 1$, $a_1(1) = 0$, and $\sigma_U^2 = E\left\{e_n^2(t)\right\}$ which is also equivalent to the model described in (2.1)–(5.3) with $\sigma_U^2 = \sigma_R^2$.

5.2.1 KF Tracking Model

We first consider the problem of tracking the location of the source with motion dynamics described by (5.5) at a fictitious node that has available at time t all the snapshot estimates of the source location up to time t, *i.e.*, $\{Z_n(\tau)\}_{\tau \le t}$ given by (5.6). We assume that the parameters of the AR(p) process $e_n(t)$ used in the model (5.5) have been estimated via training. In particular, a sequence of autocorrelation sequence estimates $\hat{r}_e(i)$ for $0 \le i \le p$ is first obtained (based on a sufficiently large set of training sample paths) and, subsequently, estimates of $\mathbf{a}_p$ and σ_U^2 are obtained by exploiting the normal equations and the energy matching property, respectively, for AR modeling (see [40]). Given that we also assume independent motion in each dimension, the 2-D source localization problem decouples into two independent 1-D source localization problems [73]. For any t and s, we let

$$\hat{X}_n(t|s) = E\left\{X_n(t) | \{Z_n(\tau)\}_{\tau \le s}\right\} \tag{5.7}$$

denote the linear mean-square-error (LMSE) estimate of the state $X_n(t)$ at time t based on all observations up to time step s (*i.e.*, $\{Z_n(\tau)\}_{\tau\leq s}$) and

$$\hat{\mathbf{\Sigma}}_n(t|s) = E\left\{\|X_n(t) - \hat{X}_n(t|s)\|^2 | Z_n(\tau)\}_{\tau\leq s}\right\} \tag{5.8}$$

denote the covariance matrix of the associated estimate. Evidently, the $(1, 1)$ entry of $\hat{\mathbf{\Sigma}}_n(t|s)$ provides the mean-square-error (MSE) of the associated position estimate.

The Kalman filter provides a recursive algorithm for obtaining the LMSE estimate $\hat{X}_n(t|t)$ of the state $X_n(t)$ based on all snapshot estimates up to time t, in terms of $\hat{X}_n(t-1|t-1)$ and the new observation $Z_n(t)$. The KF algorithm takes the following form [73]:

$$\begin{aligned}
\hat{X}_n(t|t-1) &= \mathbf{F}\,\hat{X}_n(t-1|\,t-1) && \text{(5.9a)}\\
\hat{\mathbf{\Sigma}}_n(t|t-1) &= \mathbf{F}\,\hat{\mathbf{\Sigma}}_n(t-1|\,t-1)\,\mathbf{F}^T + \mathbf{G}\,\mathbf{Q}\,\mathbf{G}^T && \text{(5.9b)}\\
\hat{X}_n(t|t) &= \hat{X}_n(t|t-1) + \mathbf{K}_n(t)[Z_n(t) - \mathbf{H}\hat{X}_n(t|t-1)] && \text{(5.9c)}\\
\hat{\mathbf{\Sigma}}_n(t|t) &= \hat{\mathbf{\Sigma}}_n(t|t-1) - \mathbf{K}_n(t)\,\mathbf{H}\,\hat{\mathbf{\Sigma}}_n(t|t-1) && \text{(5.9d)}\\
\mathbf{K}_n(t) &= \hat{\mathbf{\Sigma}}_n(t|t-1)\mathbf{H}^T[\mathbf{H}\hat{\mathbf{\Sigma}}_n(t|t-1)\mathbf{H}^T]^{-1}\,. && \text{(5.9e)}
\end{aligned}$$

The algorithm is initialized with

$$\hat{X}_n(0|0) = \begin{bmatrix} Z_n(0) \\ 0 \\ \mathbf{0}_p \end{bmatrix}, \tag{5.10a}$$

$$\hat{\mathbf{\Sigma}}_n(0|0) = \begin{bmatrix} \hat{r}(0) & 0 & -\hat{\mathbf{r}}_p^T \\ 0 & \sigma_v^2 & \mathbf{0}_p^T \\ -\hat{\mathbf{r}}_p & \mathbf{0}_p & \hat{\mathbf{R}}_p \end{bmatrix}, \tag{5.10b}$$

where $\hat{\mathbf{r}}_p = [\hat{r}_e(0)\ \hat{r}_e(1)\ \ldots\ \hat{r}_e(p-1)]^T$, and $\hat{\mathbf{R}}_p$ is the $p \times p$ Toeplitz matrix of $\hat{\mathbf{r}}_p$. We remark that our model formulation is an *extended* KF that involve the AR estimation error $e(t)$. However, for convenience we refer to the resulting tracking model as KF.

The KF algorithm (5.9) serves as a basis for developing distributed tracking algorithms, according to which the tth step of (5.9) is performed at *each* node within the ACN, $\{\mathcal{I}(t), \Psi(t)\}$. Observation of (5.9) reveals that for any node in $\mathcal{I}(t)$ to be able to perform the tth step of the algorithm and obtain $\hat{X}_n(t|t)$, the node must have available: (i) the gain $\mathbf{K}_n(t)$, (ii) the snapshot estimate $Z_n(t)$, and (iii) the previous tracking estimate $\hat{X}_n(t-1|t-1)$.

Distributed Computation of $\mathbf{K}_n(t)$

We assume at the outset that the KF initialization parameters in (5.10), *i.e.*, $\hat{\mathbf{\Sigma}}_n(0|0)$, are available at all of the sensor nodes in the network. Then, as the gain sequence $\mathbf{K}_n(t)$ can be pre-computed locally at each node, propagation of the index t as the source moves through the network suffices for allowing any node in $\mathcal{I}(t)$ to compute $\mathbf{K}_n(t)$.

Distributed Computation of $Z_n(t)$

As discussed in Sec. 4.2, $Z_n(t)$ is obtained via distributed source localization algorithms over the ACN at time t such as the ones based on centroid and LS. The choice of centroid-based versus LS-based location estimates for $Z_n(t)$ can depend on several factors including accuracy, computational complexity and the update rate, $f_s = 1/T_s$.

Distributed Computation of $\hat{X}_n(t-1|t-1)$.

We next focus on how $\hat{X}_n(t-1|t-1)$ can be approximated via distributed computations over the computation network $\{\mathcal{I}(t-1), \Psi(t-1)\}$. $\hat{X}_n(t-1|t-1)$ is directly available only to the nodes in $\mathcal{I}(t)$ that were also part of the active network at time $t-1$, *i.e.*, only to the nodes in $\mathcal{I}(t) \cap \mathcal{I}(t-1)$ as illustrated in Fig. 5.2. Given that the "measurements" employed at time $t-1$ are approximations to $Z_n(t-1)$, each of the resulting $\hat{X}_n(t-1|t-1)$ via (5.9) at the nodes in $\mathcal{I}(t) \cap \mathcal{I}(t-1)$ are in general distinct approximations. For this reason, a distributed computation of average algorithm (such as the ones described in Sec. 3.2) is employed on the network topology induced by the restriction of $\Phi(t)$ on $\mathcal{I}(t) \cap \mathcal{I}(t-1)$, to provide to all the nodes in $\mathcal{I}(t) \cap \mathcal{I}(t-1)$ an approximation to the average of the available $\hat{X}_n(t-1|t-1)$ estimates. In parallel, a broadcasting algorithm is employed to provide $\hat{X}_n(t-1|t-1)$ estimates to the remaining nodes in $\mathcal{I}(t)$, according to which at each cycle, each node in the subset broadcasts its $\hat{X}_n(t-1|t-1)$ estimate (if one is available) to its neighbors, and iteratively computes a new estimate as the average of the available estimates.

Due to the averaging of the available $\hat{X}_n(t-1|t-1)$ estimates, the algorithm (5.9) with $\mathbf{Q}$ replaced by $\mathbf{Q}'$ provides a conservative KF algorithm, in the sense that the true covariance matrix of the resulting $\hat{X}_n(t-1|t-1)$ at each node in $\mathcal{I}(t)$ is upper-bounded by $\hat{\mathbf{\Sigma}}_n(t-1|t-1)$ in (5.9), *i.e.*, the difference of the two covariance matrices is negative semi-definite.

5.3 Distributed Tracking Simulations & Analysis

In the following simulations, we employ as our figure of merit the sample-mean MSE (in dB) performance of the associated localization and tracking algorithms based on $MC = 1000$ independent realizations. In each realization, a network is first generated with $N = 400$ nodes uniformly distributed in a circle of radius $R = 200$ m ($D = 0.0032$ node/m^2). A network topology is then generated according to (3.2) with $m = 2$ and $d_o = 55$ m. At the outset (*i.e.*, prior to tracking) a set of base ρ_{ij}'s are computed for the entire network by applying the LN algorithm of Sec. 3.2 for $k_{\text{in}} = 10$ iterations. Next, a source-motion sample path is generated by placing the source at the origin of the network and using model (5.5) to generate a source trajectory. In the simulation examples below, we employ a deterministic signal model with SNR $= \sigma_s^2/\sigma_\eta^2 = 55.6$ dB, and $\sigma_T^2 = 20$ dB. This threshold yields ACN's with (detecting) nodes within approximately a 60 m radius from the source location. We refer to the "detection" radius as R_{det}. Prior to running the distributed algorithm (3.31) on any given set of snapshot data, the base ρ_{ij}'s (computed at the outset on the whole network) are refined for $k_{\text{re}} = 5$ iterations by applying the LN algorithm on the current ACN. The distributed algorithm (3.31) is then applied with $c = 0.6$ for $k_o = 20$ iterations to approximate $Z_n(t)$ in (5.6) at each node in the ACN. Finally, $e(t)$ in (5.5) is modeled throughout as an AR(3) process.

Fig. 5.3 depicts the simulated MSE performance of centralized and decentralized tracking algorithms in the case that $\sigma_V = 2$ m/s, $\sigma_A = 0.1$ m/s^2, and $T_s = 1$ s. As the figure suggests, the distributed snapshot source localization and tracking algorithms provide effectively the same MSE performance as their centralized counterparts, using only a small number of iterations for initialization (*i.e.*, $k_{\text{in}} + k_{\text{re}} = 15$) and distributed computation (*i.e.*, $k_o = 20$). Furthermore, distributed tracking yields

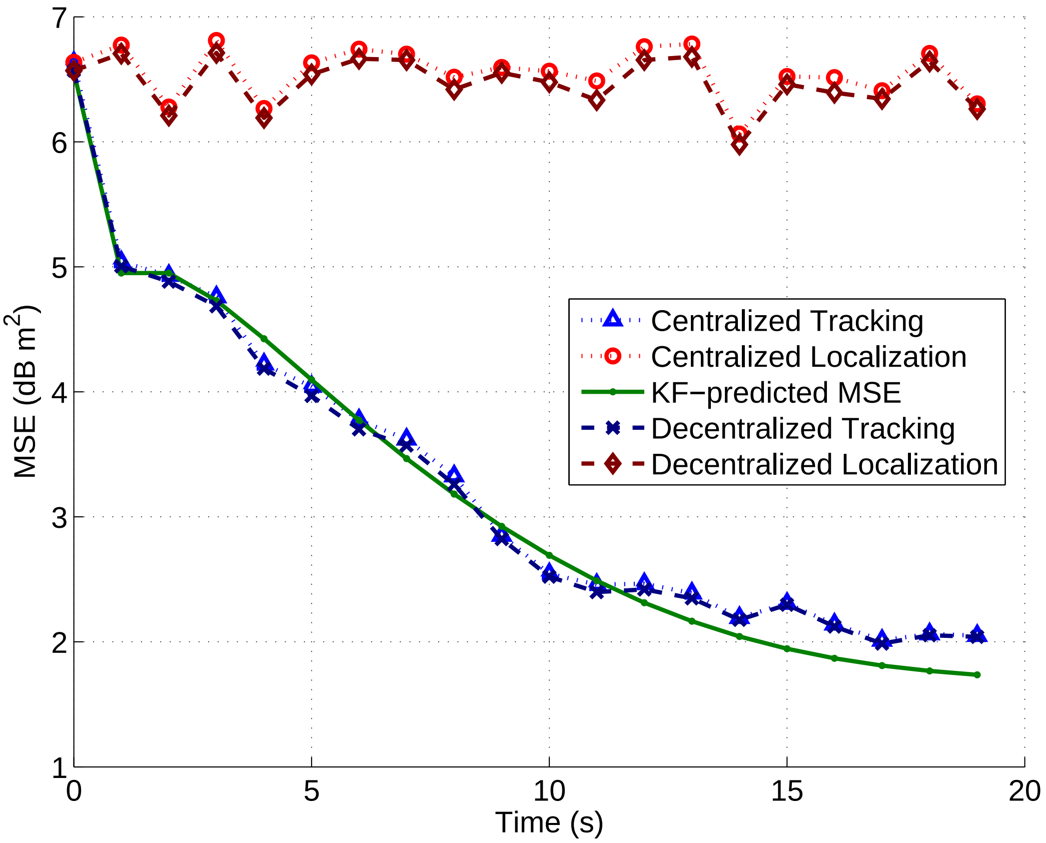


Figure 5.3: MSE comparison of centralized and decentralized localization and tracking algorithms for $\sigma_v = 2$ m/s, $\sigma_a = 0.1$ m/s^2, and $T_s = 1$ s.

a gain of approximately 4 dB with respect to the associated distributed snapshot source localization algorithm. Finally, we note that the tracking algorithm MSE performance is in close agreement with the MSE performance predicted by the (1, 1) entry (solid line) of the LHS of (5.9d).

Fig. 5.4 shows the MSE performance of the proposed algorithms as a function of the snapshot rate, when $\sigma_v = 2$ m/s and $\sigma_a = 0.1$ m/s^2. In particular, the successively lower curves show the simulated MSE of the distributed tracking algorithms for $T_s = 0.5, 1,$ and 2 s, respectively. As the figure suggests, in this example, increasing the snapshot rate by a factor of 2 reduces the MSE by approx. 1.5 dB.

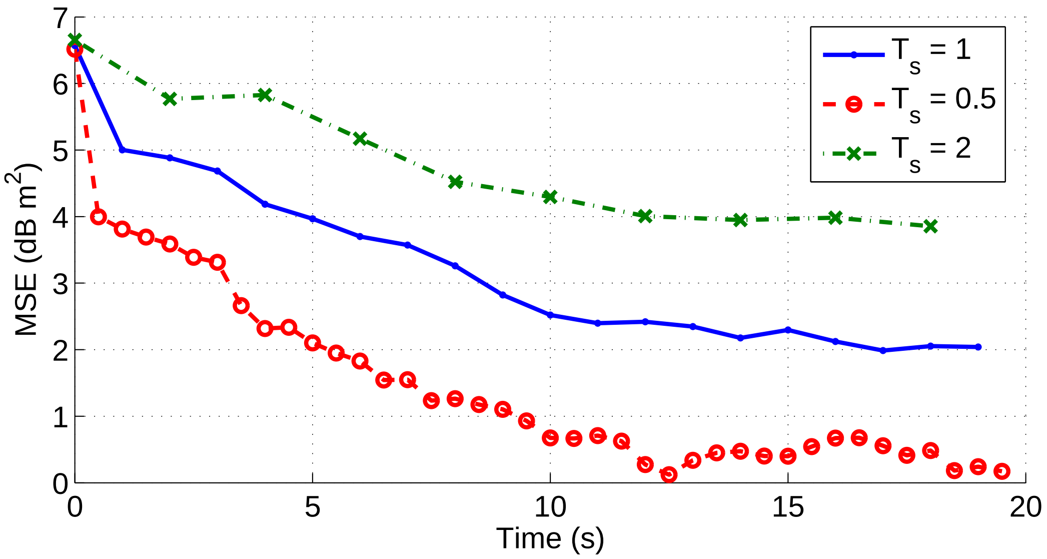


Figure 5.4: MSE performance of distributed tracking algorithms for various values of T_s, for $\sigma_v = 2$ m/s and $\sigma_a = 0.1$ m/s^2.

Fig. 5.5 shows the MSE performance of the distributed tracking algorithms as the source speed and acceleration parameters are varied, with snapshots taken at rate of 1 measurement/s per sensor. As the figure reveals, increasing σ_a by a factor of 2 while keeping σ_v unchanged results in increasing the steady-state MSE by 0.5 dB, while increasing σ_v by a factor of 2 while keeping σ_a unchanged does not appreciably affect the steady-state MSE performance. We remark that by increasing σ_a, more randomness is introduced in the state-space model (5.5) and therefore, more error is incurred.

5.4 Sensor Fusion in Time Summary

In summary, we developed a systematic and viable decentralized framework based on Kalman filtering for performing distributed space-time processing. We presented

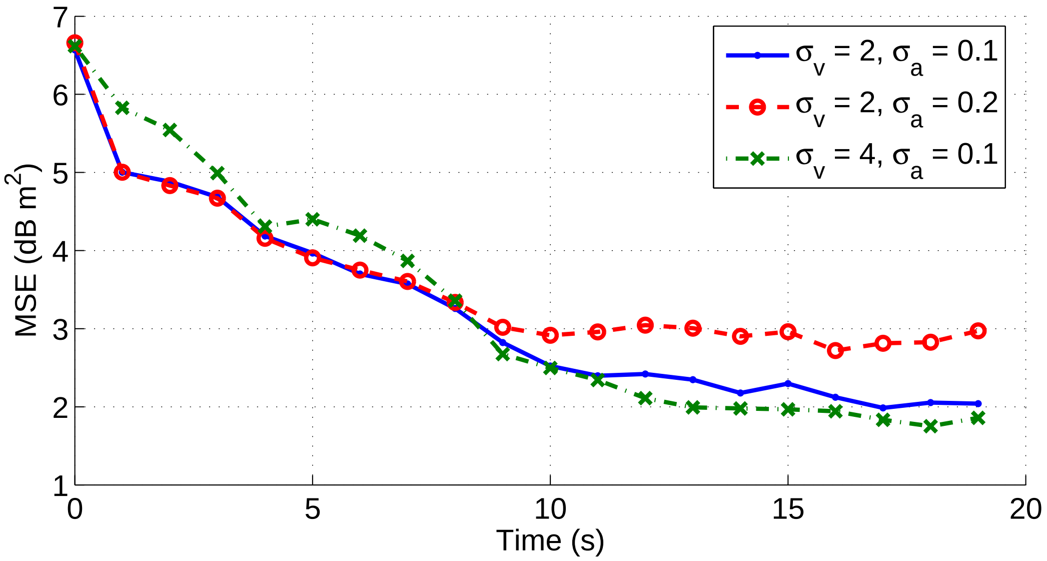


Figure 5.5: MSE performance of distributed tracking algorithms as σ_v and σ_a are varied, while keeping $T_s = 1$ s.

distributed tracking methods whereby each participating node in the detecting subnetwork locally obtains the gain, snapshot estimate, and previous tracking estimate information and performs local KF tracking. We found via simulations that distributed tracking (i) yields consistent processing gains with respect to the associated distributed (snapshot) source localization algorithm, and (ii) is in close agreement with the centralized counterpart and the associated KF predicted MSE performance. We also found that, the distributed tracking performance varies and potential scales with snapshot update rate and the source dynamics such as velocity.

Chapter 6

Contributions and Future Directions

The final chapter summarizes the contributions of this dissertation and highlights some of the areas for further research in distributed signal processing for sensor networks.

6.1 Contributions

Research and development interests in sensor networks for many civilian and military applications have proliferated in recent years due to a confluence of a multitude of technologies/disciplines such as MEMS sensor designs, software and hardware, signal and information processing, communications and networks. As a result, the problems associated with the design and implementation of sensor networks are vast. The promising idea of "instrumenting the world" [2] with very large-scale wireless sensor networks with thousands nodes to provide pervasive monitoring and surveillance is a difficult challenge due to (battery) power and communication bandwidth constraints. In our investigation, we have chosen to focus on developing potentially more realizable approaches involving detection, source localization and tracking via ad-hoc networks of (low-power) sensors. By focusing on more simplified settings, we were able to identify interesting and tractable problems and develop innovative solutions that are adaptable to changing network conditions and can be easily scaled to larger sensor networks.

In this dissertation, we developed a decentralized algorithmic framework for systematic tracking of moving acoustic sources in large-scale ad-hoc networks. In the process, we identified the key challenges and offered a systematic approach to distributively solving a class of source localization and tracking problems. The algorithms we developed performed iterative, space-time processing with sensor fusion in space first follow by sensor fusion in time. Distributed fusion in space is performed by fusing current high-quality sensed data from the subnetwork of detecting sensor nodes to produce the current source location estimate. The location estimates are computed over a sequences of subnetworks and subsequently fused in time to produce tracking estimates. We developed fusion in space and fusion in time algorithms that are amenable to distributed implementation over the underlying network topology. The algorithms we developed are based on the distributed computation of averages algorithms and are locally-constructed at each participating sensor node exploiting only locally available observations and local available network connectivity information. In addition, these algorithms are also resource efficient, scalable, fault-tolerant and can readily adapt to local changes in network topologies.

In the context of distributed algorithms implemented over the underlying AHSN topologies, we developed improved versions of the distributed algorithms of computation of averages reported in [17, 18] and demonstrated how two classes of source localization algorithms can be mapped into the forms of (weighted) averages for parallel distributed implementations. Specifically, the improvements in the distributed computation algorithms are: (i) no macroscopic information of the overall global network (*e.g.*, ρ_{ij}'s not ρ), (ii) more robust to changes in network topologies, (iii) faster convergence rates, and (iv) more amenable to distributed space-time fusion implementations. We also demonstrated the feasibility of distributed implementations as these distributed algorithms, *e.g.*, only requiring approximately 5–20 iterations to converge to within 1 dB MSE with respect to their centralized counterparts.

In the context of sensor fusion in space, we focused our investigation in the design and development of distributed source localization algorithms that trade-off performance and complexity. Specifically, we developed centroid-based and range-difference LS-based estimators that can be implemented distributively in AHSN and yield good location estimates (especially LS-based) under a variety of sensor measurement, sensor network density and topological conditions. We also developed algorithmic and design strategies that keep power consumption and communication within the AHSN to acceptable levels without significantly sacrificing localization accuracy.

In the context of space-time fusion and tracking, we developed a decentralized framework whereby tracking is performed via Kalman filtering at all of the participating nodes in each of the successive computational subnetworks (*e.g.*, ACN's). We developed distributed KF algorithms that employed at any give node exploit the availability of snapshot data (via source distributed source localization algorithms) and previous state estimates both of which are computed/fused via locally-constructed algorithms over ad-hoc networks.

Finally, our investigation into the interdependence of sensor network, signal and localization algorithm parameters on estimation performance (via simulations) yielded design rules for predicting the MSE performance of LS-based estimators as the networks scale in node density and coverage area.

6.2 Future Directions

There are a number of interesting and perhaps valuable directions for further research that arise as extensions of this dissertation. In the following, we summarize a representative collections of important directions for future work, including some the issues that have been alluded to in the earlier chapters.

While our research focused on the distributed source localization and tracking of a single source in a decentralized sensor network setting, important extensions can be pursued in order to generalize the research to an even broader range of sensor network conditions and scenarios. One important and challenging extension is localization and tracking of multiple moving sources in AHSN. In this extension, we can consider several possible scenarios: (i) multiple sources are well-separated spatially with non-overlapping detection regions in time and space, (ii) multiple sources are relatively close together and moving in a convoy, and (iii) multiple sources moving in different directions and are not well-separated with overlapping detection regions. In the first scenario, our current formulation can be easily extended with minimal additional overhead information and calculations since only regionalized (*i.e.*, locally available) information is exploited. In the second scenario, we can consider the problem that arises in networked sensors surveillance applications of determining the number of sources in the convoy and the overall (convoy) direction. Again, our current formulation can be extended to address this problem with additional detection/estimation information obtained by the sensor nodes that are in close proximity to the convoy. For example, the close proximity nodes can share closet point of approach (CPA) data use to determine the number of sources with more distant nodes [5]. The third scenario can very challenging as it may require more sophisticated signal modeling (*i.e.*, in addition to just RSS), complex sensor management, and/or signal processing strategies for fully-decentralized tracking.

Another interesting research and very challenging direction is distributed sensor and data fusion of disparate information (*i.e.*, RSS, DOA, TOA, range, detection-only, and others) from possibly disparate sensor nodes (*e.g.*, acoustic, seismic, passive infrared (IR), electro-optic (EO) and others) for source localization and tracking. In this context, we can consider a multitude of combinations of possible scenarios and useful applications.[1] One specific and more realizable extension is to consider a network "low-end" sensor nodes with coarse information estimates (*e.g.*, RSS's) intermixed with "high-end" sensor nodes with higher resolution information estimates (*e.g.*, TOA's and DOA's) [75]. Even in this setting, designing distributed processing algorithms for a fully-decentralized tracking architecture can be challenging.

As we have discussed earlier, the problems associated with the design and implementation of sensor networks are vast due to the confluence of technological factors. Much of the current research focus on designing efficient communication protocols and/or signal processing algorithms for wireless sensor networks with bandwidth and power constraints. However, the research contributions we have made in the area of distributed signal processing for sensor networks are valuable in that others can build on further research for other related and non-related sensor network areas.

[1] Distributed sensor and information fusion of a network of heterogenous and disparate sensors is an important area of research and is being consider by the US Army Research Laboratory and the UK Ministry of Defense under the joint International Technology Alliance (ITA) program [74].

Appendix A

A.1 State-Measurement Models

We consider the following time-invariant state-space model for tracking

$$X(t+1) = \mathbf{F}\, X(t) + \mathbf{G}\, U(t),\ t = 0, 1, \ldots \tag{A.1a}$$

$$Z(t+1) = \mathbf{H}\, X(t) + N(t),\ t = 0, 1, \ldots \tag{A.1b}$$

where $X(t)$ is the L_X-dimensional state vector, $U(t)$ is the L_U-dimensional input process vector "acting" on $X(t)$, $Y(t)$ and $N(t)$ are L_Z-dimensional (L_Z can be different from L_X and L_U) measurement process and noise process vectors respectively. Hence the matrices $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{H}$ have dimensions of $L_X \times L_X$, $L_X \times L_U$ and $L_Z \times L_X$ respectively. Moreover, the processes $U(t)$ and $N(t)$ are zero-mean and white with known covariances $\mathbf{Q} = \sigma_U^2\, \mathbf{I}_{U\times U} = (T_s^2 \sigma_A^2)\, \mathbf{I}_{U\times U}$ and $\mathbf{R} = \sigma_N^2\, \mathbf{I}_{Z\times Z}$ respectively. The initial condition X_0 has zero-mean and covariance $\mathbf{\Sigma}$.

A.1.1 1-D Motion Model

We assume that the two-dimensional (2-D) motion parameters and the measurements for the acoustic source traveling within the sensor networks in x-y coordinates are independent from one dimension to another and we derive motion model for the acoustic source based on a one-dimensional (1-D) motion model. We assume the 1-D source motion is subject to a random acceleration $A(t)$ for $t \geq 0$, where the position, $P(t)$ and the velocity, $V(t)$ of the source at each time t satisfy $V(t) = \partial P(t)/\partial t$ and $A(t) = \partial V(t)/\partial t$. This is a second order model sometimes refer to as a *white noise acceleration* or *constant velocity* model [24]. Assuming we observe the motion of the source every T_s seconds where T_s is small, we can describe the 1-D motion from one observation time to the next via Taylor series approximation as

$$P_{(t+1)T_s} \cong P_{tT_s} + T_s\, V_{tT_s} \tag{A.2a}$$

$$V_{(t+1)T_s} \cong \varrho(T_s)\, V_{tT_s} + T_s\, A_{tT_s} \tag{A.2b}$$

where $0 < \varrho(T_s) \leq 1$. From (A.2), only two states, namely position and velocity, are needed to describe the 1-D source motion. If we set $X(t) = \Big[X_1(t)\ X_2(t)\Big]^T = \Big[P(t)\ V(t)\Big]^T$ and $U(t) = A(t)T_s$, the motion can be described approximately by the state and measurement equations respectively as

$$X(t+1) = \mathbf{F}\, X(t) + \mathbf{G}\, U(t),\ t = 0, 1, \ldots \tag{A.3a}$$

$$Z(t+1) = \mathbf{H}\, X(t) + N(t),\ t = 0, 1, \ldots \tag{A.3b}$$

where $\mathbf{F}$ is the 2×2 matrix, $\mathbf{G}$ is the 2×1 matrix and $\mathbf{H}$ is the 1×2 matrix are defined as

$$\mathbf{F} = \begin{bmatrix} 1 & T_s \\ 0 & \varrho(T_s) \end{bmatrix},\ \mathbf{G} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ \mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{A.4}$$

respectively [24, 73, 76].

A.1.2 2-D Motion Model

Given that the acceleration components and measurement components are assumed to be independent from one another, we can easily extend the 1-D motion model described in (A.3) and (A.4) into a 2-D motion model. Tracking in 2-D would require a four-state, $X(t) = \begin{bmatrix} P_x(t) & V_x(t) & P_y(t) & V_y(t) \end{bmatrix}^T$ with $U(t) = \begin{bmatrix} A_x(t)T_s & A_y(t)T_s \end{bmatrix}^T$, two-measurement model with the corresponding 4×4 matrix $\mathbf{F}$, 4×2 matrix $\mathbf{G}$ and 2×4 matrix $\mathbf{H}$ described in (A.3) by

$$\begin{bmatrix} P_x(t+1) \\ V_x(t+1) \\ P_y(t+1) \\ V_y(t+1) \end{bmatrix} = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & \varrho(T_s) & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & \varrho(T_s) \end{bmatrix} \begin{bmatrix} P_x(t) \\ V_x(t) \\ P_y(t) \\ V_y(t) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ T_s & 0 \\ 0 & 0 \\ 0 & T_s \end{bmatrix} \begin{bmatrix} A_x(t) \\ A_y(t) \end{bmatrix} \tag{A.5a}$$

$$\begin{bmatrix} Z_x(t) \\ Z_y(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_x(t) \\ V_x(t) \\ P_y(t) \\ V_y(t) \end{bmatrix} + \begin{bmatrix} N_x(t) \\ N_y(t) \end{bmatrix}. \tag{A.5b}$$

Similarly, we can extend it further to 3-D motion [24, 73].

A.1.3 The Parameter $\varrho(\mathbf{T_s})$

The parameter $\varrho(T_s)$ plays a key role in the tracking performance. We first show that for realistic situation $\varrho(T_s)$ is strictly less than 1 and then examine $\varrho(T_s)$ in detailed as to how it affects convergence and performance.

Suppose $\varrho(T_s)$ in (A.4) is unity, so that (A.3a) implies

$$V(t+1) = V(t) + U(t)\,. \tag{A.6}$$

Then all the change in $V(t)$ from t to $t+1$ is associated with the noise (acceleration) component $U(t)$ where $E\{U(t)\} = 0$ and $E\{U^2(t)\} = \sigma_U^2$. If we set $V(0) = 0$, then after t_n updates, we get $E\{V^2(t_n)\} = t_n\sigma_U^2$, which implies that the mean square speed is unbounded. Clearly, this is unrealistic in any physical situation. It would be reasonable to have $E\{V^2(t_n)\} = \sigma_{V_{max}}^2$ where $\sigma_{V_{max}}^2$ is a constant independent of t but dependent on the speed capabilities of the source. In this case, we can solve for $\varrho(T_s)$ with $0 < \varrho(T_s) < 1$ based on $\sigma_{V_{max}}^2$ and $\sigma_U^2 = T_s^2\sigma_A^2$ via

$$E\{V^2(t)\} = \varrho(T_s)^2 E\{V^2(t)\} + E\{U^2(t)\} \tag{A.7a}$$

$$\sigma_{V_{max}}^2 = \varrho(T_s)^2\sigma_{V_{max}}^2 + T_s^2\sigma_A^2\,, \tag{A.7b}$$

which gives

$$\varrho(T_s) = \sqrt{\frac{\sigma_{V_{max}}^2 - T_s^2\sigma_A^2}{\sigma_{V_{max}}^2}}\,. \tag{A.8}$$

From (A.8), we can see clearly that $\varrho(T_s)$ is a function of the update time or rate, T_s [73]. Suppose for a nominal update rate T_o we have the corresponding $\varrho(T_s)$-value $\varrho(T_s)_o$. Then for any value T, we have

$$\varrho(T) = \varrho_o^{T/T_o}. \tag{A.9}$$

A.2 CRB for Source Localization

The mean and the covariance for the stochastic signal $\mathbf{z}_k$, $k = 1, \cdots, L$ defined in (2.4) with $\beta = 2$ in (2.10), are

$$\boldsymbol{\mu}(\boldsymbol{\theta}) = E\{\mathbf{z}_k\} = E\{S_k\}\mathbf{h}(\mathbf{p}_s, \beta) + E\{\boldsymbol{\eta}_k\} \tag{A.10a}$$

$$
\begin{aligned}
\boldsymbol{C}(\boldsymbol{\theta}) &= E\{\mathbf{z}_k\, \mathbf{z}_k^H\} = E\{S_k\, S_k^*\}\, [\mathbf{h}(\mathbf{p}_s,\, \beta)\, \mathbf{h}^H(\mathbf{p}_s,\, \beta)] + E\{\boldsymbol{\eta}_k\, \boldsymbol{\eta}_k^H\} \\
&= \sigma_s^2 \begin{bmatrix} \frac{1}{r_1^2} & \frac{1}{r_1 r_2} & \cdots & \frac{1}{r_1 r_M} \\ \frac{1}{r_1 r_2} & \frac{1}{r_2^2} & \cdots & \frac{1}{r_2 r_M} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{r_M r_1} & \frac{1}{r_M r_2} & \cdots & \frac{1}{r_M^2} \end{bmatrix} + \sigma_n^2 \mathbf{I}.
\end{aligned}
\tag{A.10b}
$$

For stochastic signal model, the calculation of the CRB (in Cartesian coordinates) via FIM in (2.16) involves only $\mathbf{C}(\boldsymbol{\theta})$ and grammian $\frac{\partial \mathbf{C}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$, since $\boldsymbol{\mu}(\boldsymbol{\theta}) = \mathbf{0}$. From (A.10), it's straightforward to obtain (2.17). By taking partial derivatives of (2.17), the grammian terms in (2.18) are as follows:

$$
\begin{aligned}
\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{p,q}}{\partial \theta_1} &= \sigma_s^2 \frac{\partial}{\partial x_s} \left[[((x_p - x_s)^2 + (y_p - y_s)^2)((x_q - x_s)^2 + (y_q - y_s)^2)] \right]^{-\frac{1}{2}} \\
&= -\frac{\sigma_s^2}{2} (r_p^2\, r_q^2)^{-\frac{3}{2}} [-2(x_p - x_s) r_q^2 - 2(x_q - x_s) r_p^2] \\
&= \frac{\sigma_s^2 [(x_p - x_s) r_q^2 + (x_q - x_s) r_p^2]}{(r_p\, r_q)^3},
\end{aligned}
\tag{A.11a}
$$

$$
\begin{aligned}
\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{p,q}}{\partial \theta_2} &= \sigma_s^2 \frac{\partial}{\partial y_s} \left[[((y_p - y_s)^2 + (y_p - y_s)^2)((y_q - y_s)^2 + (y_q - y_s)^2)] \right]^{-\frac{1}{2}} \\
&= -\frac{\sigma_s^2}{2} (r_p^2\, r_q^2)^{-\frac{3}{2}} [-2(y_p - y_s) r_q^2 - 2(y_q - y_s) r_p^2] \\
&= \frac{\sigma_s^2 [(y_p - y_s) r_q^2 + (y_q - y_s) r_p^2]}{(r_p\, r_q)^3},
\end{aligned}
\tag{A.11b}
$$

$$
\frac{[\partial \mathbf{C}(\boldsymbol{\theta})]_{p,q}}{\partial \theta_3} = \frac{\partial}{\partial \sigma_s^2} [\frac{\sigma_s^2}{(r_p\, r_q)}] = \frac{1}{(r_p\, r_q)},
\tag{A.11c}
$$

where the distance from sensor i to the source is $r_i = \sqrt{(x_i - x_s)^2 + (y_i - y_s)^2}$.

The calculation of the CRB (in polar coordinates) in (2.19) for range estimation involves the FIM in (2.16) and the Jacobian matrix. By taking the partial derivative of $\tilde{\boldsymbol{\theta}} = \mathbf{g}(\boldsymbol{\theta}) = [(r_s,\, \phi_s), \sigma_s^2]$, where $r_s = \sqrt{x_s^2 + y_s^2}$ and $\phi_s = \arctan(\frac{y_s}{x_s})$, with respect to $[x_s, y_s, \sigma_s^2]$, the Jacobian terms in (2.20) are as follows:

$$
\begin{aligned}
\frac{\partial \mathbf{g}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} &= \begin{bmatrix} \frac{\partial[(x_s^2+y_s^2)^{-\frac{1}{2}}]}{\partial x_s} & \frac{\partial[(x_s^2+y_s^2)^{-\frac{1}{2}}]}{\partial y_s} & \frac{[\partial(x_s^2+y_s^2)^{-\frac{1}{2}}]}{\partial(\sigma_s^2)} \\ \frac{\partial[\arctan(\frac{y_s}{x_s})]}{\partial x_s} & \frac{\partial[\arctan(\frac{y_s}{x_s})]}{\partial y_s} & \frac{\partial[\arctan(\frac{y_s}{x_s})]}{\partial(\sigma_s^2)} \\ \frac{\partial(\sigma_s^2)}{\partial x_s} & \frac{\partial(\sigma_s^2)}{\partial y_s} & \frac{\partial(\sigma_s^2)}{\partial(\sigma_s^2)} \end{bmatrix} \\
&= \begin{bmatrix} \frac{x_s}{(x_s^2+y_s^2)^{-\frac{1}{2}}} & \frac{y_s}{(x_s^2+y_s^2)^{-\frac{1}{2}}} & 0 \\ -\frac{y_s}{(x_s^2+y_s^2)^{-\frac{1}{2}}} & \frac{x_s}{(x_s^2+y_s^2)^{-\frac{1}{2}}} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{x_s}{r_s} & \frac{y_s}{r_s} & 0 \\ -\frac{y_s}{r_s} & \frac{x_s}{r_s} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \qquad \text{(A.12)}
\end{aligned}
$$

Appendix B

B.1 Distributed Computations

.

B.1.1 Mode Shaping Filters

A simple yet effective class of second order mode-shaping filters that yield asymptotic converging LTI rules is given by

$$H(z) = \frac{1+c}{1+cz^{-2}}\,; \quad 0 < 1. \tag{B.1}$$

For $c = 1$, $H(z) = 1$ is the no-filtering case. For $c > 0$, the filter $H(z)$ induced the reshaping the eigenvalues of W, *i.e.*, $\{\lambda_i\}_{i=1}^{N}$, and their magnitudes. This is illustrated via a simulation plot in Fig. B.1 of the spectral radius resulting from filter $H(z)$ given by

$$\beta(\lambda; c) = \begin{cases} \sqrt{c} & \text{if } |\lambda| \leq |\lambda_o| \\ \frac{|\lambda|(1+c)+\sqrt{\lambda^2(1+c)^2-4c}}{2} & \text{if } |\lambda_o| \leq |\lambda_o| \leq 1 \end{cases}, \tag{B.2}$$

where $\lambda_o = 2\sqrt{c}/(c+1)$. Fig. B.1 shows that these mode shaping filters increase the spectral radius of λ_i's for which $|\lambda| \leq \sqrt{c}$, at the benefit decreasing the of the large

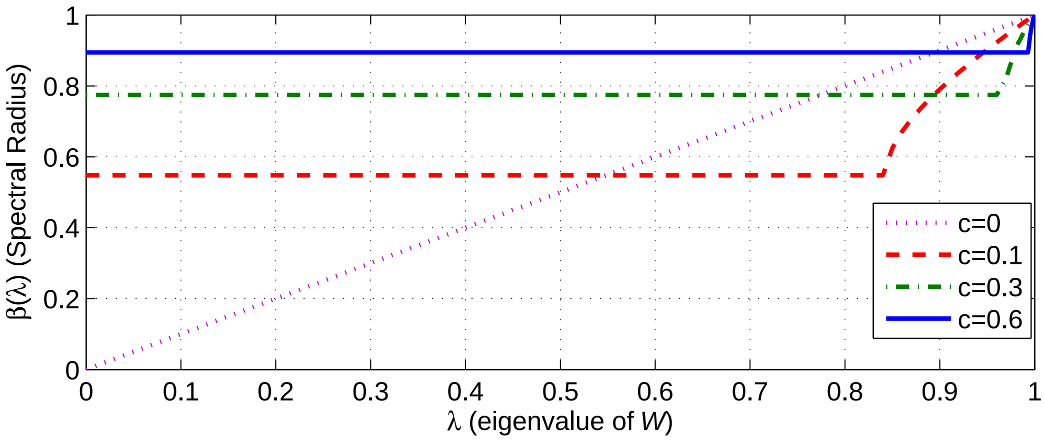


Figure B.1: Eigenvalue shaping by filter $H(z)$ for $c = \big[0,\ 0.1,\ 0.3,\ 0.6\big]$.

magnitudes λ_i's for which $|\lambda| > \sqrt{c}$ [17].

Fig. B.2 shows the eigenvalue magnitude distribution of for the cases of $c = \big[0,\ 0.3,\ 0.6\big]$ applied to a network of $N = 200$ nodes for the case of UD parameter ρ_∞. The histograms on the right show the enlarged portion of the associated histograms for the magnitudes around 1. The figure shows that applying $H(z)$ at each node reduces the large-magnitude modes at the expense of the low-magnitude modes in the network system [17].

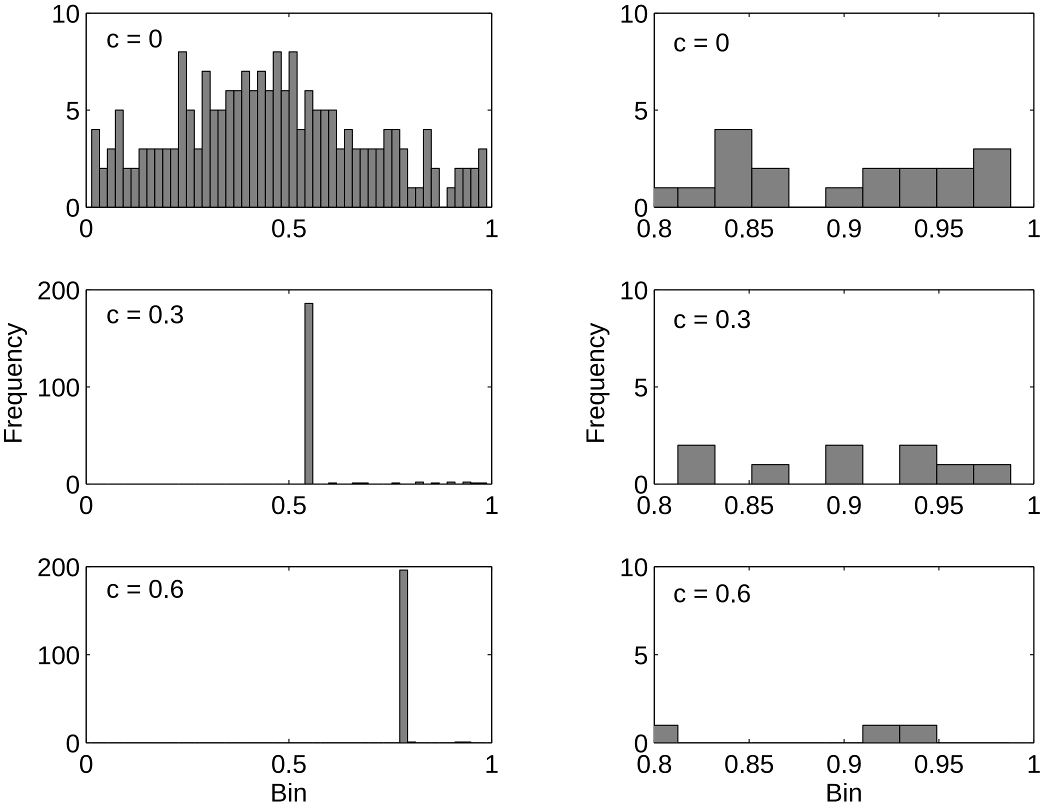


Figure B.2: Eigenvalue-magnitude histogram for a network of $N = 200$ nodes for filter parameters $c = \left[0,\, 0.3,\, 0.6\right]$ respectively. The histograms on the right show enlarged portions of the associated histograms on the left.

Appendix C

C.1 Fusion on Space

In this analysis of LS estimation, we assume a deterministic signal model and $L = 1$. Now lets consider $z_i = \widehat{\sigma^{-2}}_{\mathrm{RSS}_i} = \frac{r_i^2}{\sigma_s^2} + \epsilon_i$ in (4.11) with $\epsilon_i \sim \mathcal{N}(0,\, c\, r_i^6)$ for some constant c. Using the Mth sensor as the reference and $\tilde{\epsilon}_i = \epsilon_i - \epsilon_M$, then

$$\begin{aligned}
\tilde{z}_i = z_i - z_M &= \frac{1}{\sigma_s^2}(r_i^2 - r_M^2) + \tilde{\epsilon}_i \\
&= \frac{1}{\sigma_s^2}\Big[\big[(x_i^2 + y_i^2) + (x_M^2 + y_M^2)\big] + 2\Big[(x_M - x_i)\ \ (y_M - y_i)\Big]\begin{bmatrix} x_s \\ y_s \end{bmatrix}\Big] + \tilde{\epsilon}_i \\
&= \frac{1}{\sigma_s^2}\Big[b_i - \mathbf{w}_i^T \begin{bmatrix} x_s \\ y_s \end{bmatrix}\Big] + \tilde{\epsilon}_i\,,
\end{aligned} \tag{C.1}$$

where $b_i = \big[(x_i^2 + y_i^2) + (x_M^2 + y_M^2)\big]$ and $\mathbf{w} = -2\Big[(x_M - x_i)\ \ (y_M - y_i)\Big]^T$. Let $\tilde{\mathbf{z}} = [\tilde{z}_1 \cdots \tilde{z}_{M-1}]^T, \mathbf{w} = [\mathbf{w}_1 \cdots \mathbf{w}_{M-1}]^T, \mathbf{b} = [b_1 \cdots b_{M-1}]^T$ and $\tilde{\boldsymbol{\epsilon}} = [\tilde{\epsilon}_1 \cdots \tilde{\epsilon}_{M-1}]^T =$

$\boldsymbol{\epsilon} - \epsilon_M \mathbf{1}$ with $\tilde{\boldsymbol{\epsilon}} \sim \mathcal{N}(\mathbf{0},\, \Lambda_{\tilde{\epsilon}})$, we have the following vector equation:

$$\frac{1}{\sigma_s^2}\,\tilde{\mathbf{z}} + \mathbf{w}\begin{bmatrix} x_s \\ y_s \end{bmatrix} - \mathbf{b} = \tilde{\boldsymbol{\epsilon}}$$

$$\begin{bmatrix}\mathbf{w}\;\tilde{\mathbf{z}}\end{bmatrix}\begin{bmatrix} x_s \\ y_s \\ \frac{1}{\sigma_s^2} \end{bmatrix} - \mathbf{b} = \tilde{\boldsymbol{\epsilon}} \tag{C.2}$$

We then have $\mathbf{A}\,\mathbf{x} - \mathbf{b} = \;\tilde{\boldsymbol{\epsilon}}$ where $\mathbf{A} = [\mathbf{w}\;\tilde{\mathbf{z}}]$ and $\mathbf{x} = [x_s\; y_s\; \frac{1}{\sigma_s^2}]^T$. Assuming that $\Lambda_{\tilde{\epsilon}}$ is positive definite and $\Lambda_{\tilde{\epsilon}}^{-1} = V\,\Lambda^{-1}V^T = (V\,\Lambda^{-\frac{1}{2}})\,(\Lambda^{-\frac{1}{2}}\,V^T)$, we use a whitening approach to find an explicit form for $\Lambda_{\tilde{\epsilon}}$, *i.e.*,

$$\Lambda^{-\frac{1}{2}}\,V^T\Big[\mathbf{A}\,\mathbf{x} - \mathbf{b}\Big] = \Lambda^{-\frac{1}{2}}\,V^T\tilde{\boldsymbol{\epsilon}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{C.3}$$

Let $\mathbf{B} = \Lambda^{-\frac{1}{2}}\,V^T\,\mathbf{A}$, solving for $\mathbf{x}$ using least-squares, we have

$$\begin{aligned} \mathbf{B}^T\,\mathbf{B}\,\mathbf{x} &= \Lambda^{-\frac{1}{2}}\,V^T\,\mathbf{b} \\ \mathbf{x} &= (\mathbf{B}^T\,\mathbf{B})^{-1}\,\mathbf{B}^T\,\Lambda^{-\frac{1}{2}}\,V^T\,\mathbf{b} \\ \mathbf{x} &= (\mathbf{A}^T\,\Lambda_{\tilde{\epsilon}}^{-1}\,\mathbf{A})^{-1}\,\mathbf{A}^T\,\Lambda_{\tilde{\epsilon}}^{-1}\,\mathbf{b}\,. \end{aligned} \tag{C.4}$$

Since $\tilde{\boldsymbol{\epsilon}} = \boldsymbol{\epsilon} - \epsilon_M\,\mathbf{1},\; \Lambda_{\tilde{\epsilon}} = \Lambda_{\epsilon} + \Lambda_{\epsilon M}\,\mathbf{1}\mathbf{1}^T$. So the MSE-weighted matrix $\mathbf{W}$ is defined as

$$\mathbf{W} = \Lambda_{\tilde{\epsilon}}^{-1} = c\,\Big[\mathrm{diag}(\mathrm{r}_1^6,\, \mathrm{r}_2^6,\, \cdots,\, \mathrm{r}_{\mathrm{M}-1}^6) + \mathrm{r}_{\mathrm{M}}^6\,\mathbf{1}\mathbf{1}^{\mathrm{T}}\Big]^{-1}. \tag{C.5}$$

Using the matrix inversion lemma, we can solve for $\mathbf{W}$ explicitly.

C.1.1 MSE trends in range-squared estimates

In this appendix we examine the dependence of $\sigma^2_{\epsilon_i}$ on r_i as $r_i \to 0$ and $r_i \to \infty$, in the case that $\sigma^2_T > \sigma^2_\eta$. For notational simplicity we drop the dependence of $\widehat{\sigma^{-2}}_{\mathrm{RSS}_i}$ and r_i on i, and consider without loss of generality the case $\sigma_s = 1$, $\sigma_\eta = 1$, and $L = 1$. Assuming the generic node of interest is a detecting node, (2.13) and (2.14) specialize to

$$y = r^{-1} + v, \tag{C.6}$$

$$\widehat{\sigma^2}_{\mathrm{RSS}} = y^2 - 1 \tag{C.7}$$

respectively. We also let v denote a random variable whose pdf equals the conditional pdf of η conditioned on the event that the node is a detecting node. Using (C.6), (C.7) and (4.11) and the fact that the node is a detecting node if $\widehat{\sigma^2}_{\mathrm{RSS}} > \sigma^2_T$, we obtain the following expression for the pdf of v

$$p_v(v;\, r) = \begin{cases} \frac{1}{\sqrt{2\pi}\, B(r,\sigma^2_T)} e^{-v^2/2} & \text{if } v \in A(r,\sigma^2_T) \\ 0 & \text{otherwise} \end{cases}, \tag{C.8}$$

and where

$$A(r,\sigma^2_T) = (-\infty, -\sqrt{1+\sigma^2_T} - r^{-1}) \bigcup (\sqrt{1+\sigma^2_T} - r^{-1}, \infty)$$

and

$$B(r,\sigma^2_T) = \mathcal{Q}\left(\sqrt{1+\sigma^2_T} - r^{-1}\right) + \mathcal{Q}\left(\sqrt{1+\sigma^2_T} + r^{-1}\right) .$$

First we consider the case $r \to \infty$. Via the triangle inequality we have

$$r^4 - E\left\{\widehat{\sigma^{-2}}^2_{\mathrm{RSS}}\right\} \le \sigma^2_\epsilon \le r^4 + E\left\{\widehat{\sigma^{-2}}^2_{\mathrm{RSS}}\right\} . \tag{C.9}$$

Using (C.6)–(C.8) and (4.11), and taking the limit $r \to \infty$ yields

$$\lim_{r\to\infty} E\Big\{\widehat{\sigma^{-2}}^{2}_{\text{RSS}}\Big\} = \frac{\sqrt{2}}{\sqrt{\pi}B(\infty, \sigma_T^2)} \int_{\sqrt{1+\sigma_T^2}}^{\infty} \frac{e^{-v^2/2}}{\left(v^2-1\right)^2} dv$$

which, for any fixed $\sigma_T^2 > 1$, is a positive finite constant independent of r. The above limit together with (C.9) implies that $\sigma_\epsilon^2 \propto r^4$ for large r.

The case of practical interest, however, corresponds to $\sigma_s^2 \gg \sigma_\eta^2$, and $\sigma_T^2 \gg \sigma_\eta^2$. In this case the values of r with non-negligible probability of detection are relatively small. We have

$$\begin{aligned}\sigma_\epsilon^2 &= E\Big\{\Big(\widehat{\sigma^{-2}}_{\text{RSS}} - r^2\Big)^2\Big\} \\ &= r^6 E\Bigg\{\bigg(\frac{2v + r(v^2-1)}{1 + 2vr + r^2(v^2-1)}\bigg)^2\Bigg\}\end{aligned}$$

Thus,

$$\lim_{r\to 0} \frac{\sigma_\epsilon^2}{r^6} = 4E\big\{v^2; r = 0\big\} = 4E\left\{\eta^2\right\} = 4$$

which verifies the validity of (4.12) for small enough r and is in agreement with the simulations presented in Fig. 4.1.

C.1.2 Estimate of the Weight Matrix

For ease of notation, we drop the dependence on i in (4.3) so $g = \sigma_{\text{RSS}} + \omega$ with $\omega \sim \mathcal{N}(0,\, \sigma_\omega^2)$. Recall that the nth moment of g, defined as $l_n \triangleq E\left\{g^n\right\}$, for $n = [2,\, 4,\, 6]$, are

$$l_2 = {\sigma^2}_{\text{RSS}} + \sigma_\omega^2\,, \tag{C.10a}$$

$$l_4 = 4 + 6{\sigma^2}_{\text{RSS}} s^2 + 3{\sigma^4}_{\text{RSS}}\,, \tag{C.10b}$$

$$l_6 = {\sigma^6}_{\mathrm{RSS}} + 15\sigma_\omega^2 {\sigma^4}_{\mathrm{RSS}} + 45\sigma_\omega^4 {\sigma^2}_{\mathrm{RSS}} + 15\sigma_\omega^6 \tag{C.10c}$$

The goal is to find the unbiased estimator of ${\sigma^6}_{\mathrm{RSS}}$ designated as $\widehat{\sigma^6}_{\mathrm{RSS}}$ *i.e.*, $E\{\widehat{\sigma^6}_{\mathrm{RSS}}\} = {\sigma^6}_{\mathrm{RSS}}$. So

$$\begin{aligned}
{\sigma^6}_{\mathrm{RSS}} &= \alpha_6\, l_6 + \alpha_4\, l_4 + \alpha_2\, l_2 + \alpha_0 \\
&= \alpha_6\, {\sigma^6}_{\mathrm{RSS}} + (15\,\alpha_6\,\sigma_\omega^2 + \alpha_4){\sigma^4}_{\mathrm{RSS}} + (45\,\alpha_6\,\sigma_\omega^4 + 6\,\alpha_4\,\sigma_\omega^2 + \alpha_2){\sigma^2}_{\mathrm{RSS}} \\
&\quad + (15\,\alpha_6\,\sigma_\omega^6 + 3\,\alpha_4\,\sigma_\omega^4 + \alpha_2\,\sigma_\omega^2 + \alpha_0) \qquad \text{(C.11)}
\end{aligned}$$

Collecting the terms, we have $\alpha_6 = 1$, $\alpha_4 = -15\,\sigma_\omega^2$, $\alpha_2 = 45\,\sigma_\omega^4$, $\alpha_0 = -15\,\sigma_\omega^6$, which yields

$$\widehat{\sigma^6}_{\mathrm{RSS}} = g^6 - 15\,\sigma_\omega^2\, g^4 + 45\,\sigma_\omega^4\, g^2 - 15\,\sigma_\omega^6. \tag{C.12}$$

C.2 Modified Distributed WCEN

The MSE performance of dWCEN as formulated in (4.19) can degrade significantly if an absolute (global) reference coordinate system is used instead of a local reference coordinate system. To alleviate this problem, we can use one of the detecting sensors in the AHSN (*e.g.*, the Mth one) as the (global) reference sensor (*i.e.*, using a relative coordinate system). Now using the Mth sensor as the reference and the fact that the normalized weights sum to 1, *i.e.*,

$$\frac{1}{(M-1)} \sum_{i=1}^{(M-1)} \frac{\hat{\sigma}^2_{i_{\mathrm{RSS}}}}{\frac{1}{(M-1)} \sum_{j=1}^{(M-1)} \hat{\sigma}^2_{j_{\mathrm{RSS}}}} = 1 \tag{C.13}$$

we can reformulate (4.19) as

$$\hat{\mathbf{p}}_{\text{rwc}} = \left[\frac{1}{(M-1)} \sum_{i=1}^{(M-1)} \left\{ \frac{\hat{\sigma}_{i\text{RSS}}^2 \left(x_i - x_M\right)}{\frac{1}{(M-1)} \sum_{j=1}^{(M-1)} \hat{\sigma}_{j\text{RSS}}^2} + x_M \right\}, \right.$$
$$\left. \frac{1}{(M-1)} \sum_{i=1}^{(M-1)} \left\{ \frac{\hat{\sigma}_{i\text{RSS}}^2 \left(y_i - y_M\right)}{\frac{1}{(M-1)} \sum_{j=1}^{(M-1)} \hat{\sigma}_{j\text{RSS}}^2} + y_M \right\} \right]. \qquad \text{(C.14)}$$

Then, the distributed implementation of $\hat{\mathbf{p}}_{\text{rwc}}$ in (C.14) follows the same two parallel cascade-distributed computations for (C.14) described in Sec. 4.2.1. In our AHSN

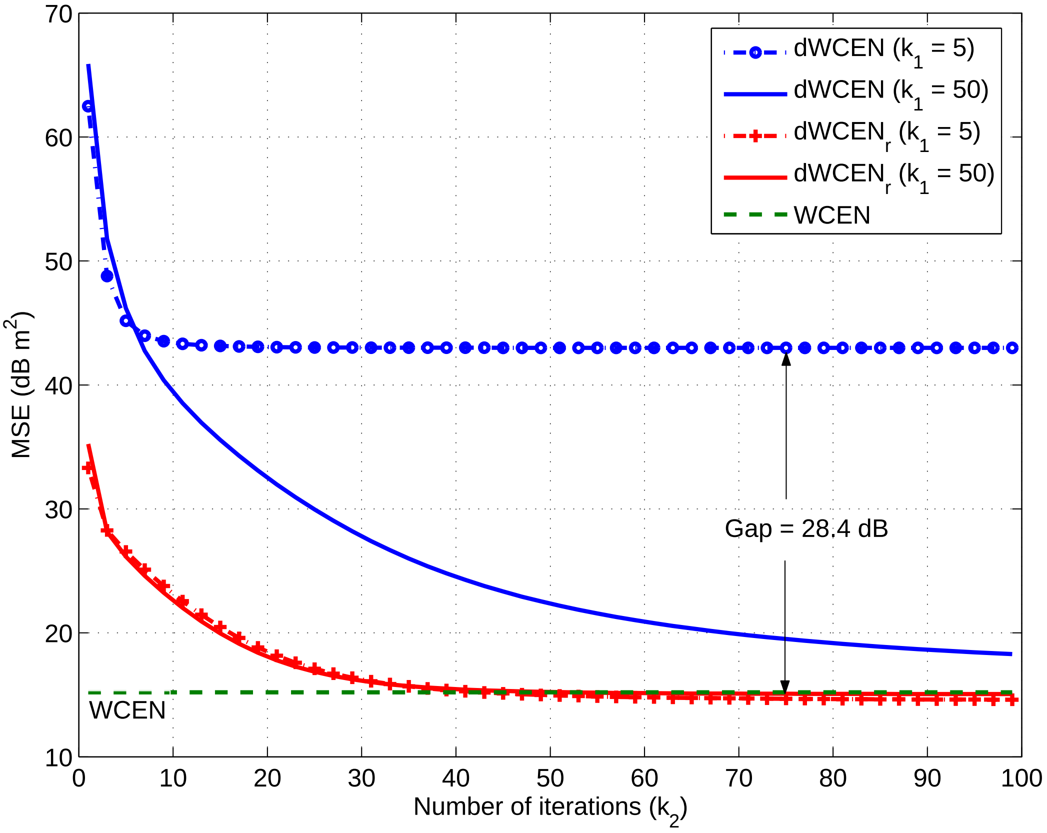


Figure C.1: MSE vs. k_2 for dWCEN and dWCEN$_\text{r}$ for $k_1 = 5$ and 50 for a source located at $\mathbf{p}_s = \left[500\ 500\right]^T$.

simulations, we use a coordinate system in which we assume the source is located at the origin, *i.e.*, $\mathbf{p}_{\mathrm{s}} = [0,\, 0]$. To illustrate the performance difference for distributed implementation of dWCEN in (4.19) versus implementation of dWCEN in (C.14), referred to as dWCEN$_{\mathrm{r}}$, we perform Monte Carlo simulations where the source is located away from the origin, say for example, $\mathbf{p}_{\mathrm{s}} = \left[500\ 500\right]^T$. Fig. C.1 shows the MSE versus k_2 for dWCEN and dWCEN$_{\mathrm{r}}$ for $k_1 = 5$ and 50 iterations and Fig. C.1 reveals several interesting differences between dWCEN and dWCEN$_{\mathrm{r}}$:

1. The additional MSE incurred can be significant, especially at low k_1 values. For example, for $k_1 = 5$ and $k_2 = 75$ the performance gap is as much as 28.4 dB.

2. The dWCEN$_{\mathrm{r}}$ estimator converges quickly to the weighted centroid estimator even at small k_1 values.

3. The dWCEN$_{\mathrm{r}}$ estimator with $k_1 = 5$ actually performs slightly better than both the dWCEN$_{\mathrm{r}}$ estimator with $k_1 = 50$ and the WCEN estimator for $k_2 > 40$. This is due to the fact we are comparing bias estimators.

Appendix D

D.1 Kalman Filtering Algorithm

We start with the time-invariant state-space model in (A.3). In its simplest form, the KF algorithm provides a sequential implementation for obtaining $\hat{X}(t|t)$, the estimate of $X(t)$ given all observations up to and including time t, for arbitrary $t \leq 0$. In the KF algorithm that follows, we denote the error covariance of $X(t|s)$ (the linear least-squares (LLS) estimation error of the LLS estimate of $X(t)$ given all observations up to and including time s) by $\mathbf{\Sigma}(t|s)$. The steps to the algorithms are:

(1) *Initialization*: Initialize the prediction and its associated error variance according to

$$\hat{X}(0|-1) \;=\; \mathbf{0} \tag{D.1a}$$

$$\mathbf{\Sigma}(0|-1) \;=\; \mathbf{\Sigma} \tag{D.1b}$$

(2) *Filtering*: Compute the Kalman gain matrix, $\mathbf{K}$,

$$\mathbf{K}(t) = \mathbf{\Sigma}(t|t-1)\,\mathbf{H}^T(\mathbf{H}\,\mathbf{\Sigma}(t|t-1)\,\mathbf{H}^T + \mathbf{R})^{-1}, \quad t = 0, 1, \ldots \tag{D.2}$$

and generate the filtered (updated) estimate and its associated error covariance

$$\hat{X}(t|t) = \hat{X}(t|t-1) + \mathbf{K}(t)\,(Y(t) - \mathbf{F}\,\hat{X}(t|t-1)), \quad t = 0, 1, \ldots \tag{D.3a}$$

$$\mathbf{\Sigma}(t|t) = \mathbf{\Sigma}(t|t-1) - \mathbf{K}(t)\,\mathbf{H}\,\mathbf{\Sigma}(t|t-1), \quad t = 0, 1, \ldots \tag{D.3b}$$

(3) *Prediction*: Generate the prediction estimate and its associated error covariance

$$\hat{X}(t+1|t) = \mathbf{F}\,\hat{X}(t|t), \quad t = 0, 1, \ldots \tag{D.4a}$$

$$\mathbf{\Sigma}(t+1|t) = \mathbf{F}\,\mathbf{\Sigma}(t|t)\,\mathbf{F}^T + \mathbf{G}\,\mathbf{Q}\,\mathbf{G}^T, \quad t = 0, 1, \ldots \tag{D.4b}$$

(4) *Incrementing*: Increment t and go to step 2 and repeat.

Note that by substituting the gain matrix in (D.2) into (D.3b) and then substituting the updated covariance matrix into (D.4b), we obtain a recursion for predicted covariance matrix

$$\begin{aligned}\mathbf{\Sigma}(t+1|t) = &\mathbf{F}\Big\{\mathbf{\Sigma}(t|t-1) - \mathbf{\Sigma}(t|t-1)\,\mathbf{H}^T\,\big[\mathbf{H}\,\mathbf{\Sigma}(t|t-1)\,\mathbf{H} + \mathbf{R}\big]^{-1}\\ &\mathbf{H}\,\mathbf{\Sigma}(t|t-1)\Big\}\,\mathbf{F}^T + \mathbf{G}\,\mathbf{Q}\,\mathbf{G}^T\,,\ t = 0, 1, \ldots\end{aligned} \tag{D.5}$$

This is called the (discrete-time) *Riccati equation*.

D.2 KF Tracking Model at the ith Sensor

With the state-measurement model in Sec. 2.1.1 and KF algorithm in Sec. D.1, we are ready to define a KF tracking model at each sensor in the AHSN. We assume a constant velocity model where the accelerations are iid from update to update and are Gaussian. We also assume that the acceleration in the x-direction and y-direction are independent from each other. Although the source's motion is 2-D, it is simpler to discuss the tracking problem in 1-D and tracking in 2-D can be easily extended from 1-D tracking as shown in Sec. A.1.2. The assumptions above and time-invariant state-measurement model lead to the tracker equations at sensor i

$$\begin{bmatrix} \hat{P}_i(t+1|t) \\ \hat{V}_i(t+1|t) \end{bmatrix} = \begin{bmatrix} \hat{P}_i(t|t) + T_s\hat{V}_i(t|t) \\ \varrho\hat{V}_i(t|t) \end{bmatrix} \tag{D.6}$$

and

$$\begin{bmatrix} \hat{P}_i(t|t) \\ \hat{V}_i(t|t) \end{bmatrix} = \begin{bmatrix} \hat{P}_i(t|t-1) \\ \varrho\hat{V}_i(t|t-1) \end{bmatrix} + \begin{bmatrix} K_{i,1}(t) \\ K_{i,2}(t) \end{bmatrix} \left[Y_i(t) - \hat{P}_i(t|t-1)\right], \tag{D.7}$$

where the data input $Y_i(t)$ is the distributed location estimate (i.e., either the x-component or y-component location estimate based on a distributed algorithm in Sec. 4.2). The the gain matrix $\mathbf{K}_i(t)$ is a 2×1 vector and from (5.9e) it is given by

$$\begin{bmatrix} K_{i,1}(t) \\ K_{i,2}(t) \end{bmatrix} = \begin{bmatrix} \Sigma_{i,(1,1)}(t|t-1)/(\Sigma_{i,(1,1)}(t|t-1) + \sigma_N^2) \\ \Sigma_{i,(1,2)}(t|t-1)/(\Sigma_{i,(1,1)}(t|t-1) + \sigma_N^2) \end{bmatrix} \tag{D.8}$$

where $\Sigma_{i,(j,k))}(t|t-1)$ is the $(j-k)^{\text{th}}$ component of the matrix $\mathbf{\Sigma}_i(t|t-1)$. The matrix $\mathbf{\Sigma}_i(t|t-1)$ is computed recursively via (5.9d) and (5.9b) [76].

D.3 Steady-state Filters

To reduce the computational burden of the tracker in practical systems, the time-varying filter in (5.9) is sometimes replaced with time-invariant steady-state filter

$$\begin{bmatrix} \hat{P}_i(t|t) \\ \hat{V}_i(t|t) \end{bmatrix} = \begin{bmatrix} \hat{P}_i(t|t-1) \\ \varrho(T_s)\hat{V}_i(t|t-1) \end{bmatrix} + \begin{bmatrix} \alpha \\ \beta/T_s \end{bmatrix} \Big[Y_i(t) - \hat{P}_i(t|t-1) \Big] \,, \tag{D.9}$$

where α and β are constants. The steady-state tracker for the model above is sometimes referred to as an *alpha-beta* tracker. Even for a second-order model , obtaining an explicit expression of the steady-state covariance and filter gain is difficult except in special cases such as for $\varrho(T_s) = 1$ [24]. In our analysis, we derive the steady-state solutions based on the roots of a polynomial in which the steady-state covariance and filter gain can be solved from numerically. [1]

For ease of notation, we suppress the dependence on i, the ith sensor, denote T_s as T and $\varrho(T_s) = \varrho$. We next follow the derivation steps outlined in [24] for $\varrho = 1$ with ours for $0 < \varrho < 1$. The steady-state values of the components of the state estimation covariance matrix are denoted as

$$\lim_{t\to\infty} \mathbf{\Sigma}(t|t) = \begin{bmatrix} p_{11} & p_{21} \\ p_{21} & p_{22} \end{bmatrix} = \mathbf{P} \,, \tag{D.10}$$

[1] The solution of the Riccati equation for the time-invariant system converges to a steady-state covariance if: (1) The pair $\{\mathbf{F}, \mathbf{H}\}$ is *completely observable* and (2) the pair $\{\mathbf{F}, \mathbf{C}\}$, where $\mathbf{Q} \triangleq \mathbf{C}\mathbf{C}^T$ is *completely controllable* then the steady-state covariance matrix is a unique positive definite matrix, independent of the initial conditions. For the model described in (A.3) and (A.4), it satisfies the observability and controllability conditions [24, 73].

the components of the one-step prediction covariance matrix are denoted as

$$\lim_{t\to\infty} \boldsymbol{\Sigma}(t+1|t) = \begin{bmatrix} m_{11} & m_{21} \\ m_{21} & m_{22} \end{bmatrix} = \mathbf{M}\,, \tag{D.11}$$

while the components of the alpha-beta filter gain are denoted as

$$\lim_{t\to\infty} \mathbf{K}_t = \begin{bmatrix} k_1 \\ k_2 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta/T \end{bmatrix} = \mathbf{K}\,. \tag{D.12}$$

Note that, as defined, α and β are dimensionless quantities. The steady-state updated covariance matrix can be expressed as

$$\mathbf{P} = \mathbf{M} - \mathbf{K}\,\mathbf{H}\,\mathbf{M} = \begin{bmatrix} (1-k_1)m_{11} & (1-k_1)m_{12} \\ (1-k_1)m_{12} & m_{22} - k_2 m_{12} \end{bmatrix}, \tag{D.13}$$

where the steady-state Kalman gain is

$$\mathbf{K} = \mathbf{M}\,\mathbf{H}^T \big[\mathbf{H}\,\mathbf{M}, \mathbf{H}^T + \mathbf{R}\big]^{-1} = \begin{bmatrix} \frac{m_{11}}{m_{11}+\sigma_N^2} \\ \frac{m_{12}}{m_{11}+\sigma_N^2} \end{bmatrix}. \tag{D.14}$$

Since $\mathbf{F}$ is invertible with

$$\mathbf{F}^{-1} = \begin{bmatrix} 1 & -\frac{T}{\varrho} \\ 0 & \frac{1}{\varrho} \end{bmatrix}, \tag{D.15}$$

the steady-state predicted covariance matrix can be rewritten as

$$\begin{aligned}\mathbf{P} &= \mathbf{F}^{-1}\,(\mathbf{M} - \mathbf{G}\,\mathbf{Q}\,\mathbf{G}^T)(\mathbf{F}^{-1})^T \\ &= \begin{bmatrix} m_{11} - 2\frac{T}{\varrho}m_{12} + \frac{T^2}{\varrho^2}(m_{22} - \sigma_A^2 T^2) & \frac{T}{\varrho}m_{12} - \frac{T}{\varrho^2}(m_{22} - \sigma_A^2 T^2) \\ \frac{T}{\varrho}m_{12} - \frac{T}{\varrho^2}(m_{22} - \sigma_A^2 T^2) & \frac{1}{\varrho^2}(m_{22} - \sigma_A^2 T^2) \end{bmatrix}.\end{aligned} \tag{D.16}$$

Matching term-by-term from (D.13) to (D.16) and from (D.14), we obtain the following five equations that we can solve for the five unknowns, $m_{11}, m_{12}, m_{22}, k_1$ and k_2:

$$\begin{aligned} m_{11} &= m_{11} - \frac{2T}{\varrho}m_{12} + \frac{T^2}{\varrho^2}m_{22} - \frac{T^4}{\varrho^2}\sigma_A^2 \\ &\Rightarrow\; k_1 m_{11} = \frac{2T}{\varrho}m_{12} - \frac{T^2}{\varrho^2}m_{22} + \frac{T^4}{\varrho^2}\sigma_A^2\,, \end{aligned} \tag{D.17}$$

$$\begin{aligned} (1 - k_1)\,m_{12} &= \frac{1}{\varrho}m_{12} - \frac{T^2}{\varrho^2}m_{22} + \frac{T^3}{\varrho^2}\sigma_A^2 \\ &\Rightarrow\; (k_1 + \frac{1}{\varrho} - 1)\,m_{12} = \frac{T}{\varrho^2}m_{22} - \frac{T^3}{\varrho^2}\sigma_A^2\,, \end{aligned} \tag{D.18}$$

$$\begin{aligned} m_{22} - k_2 m_{12} &= \frac{1}{\varrho^2}(m_{22} - \sigma_A^2 T^2) \\ &\Rightarrow\; k_2 m_{12} = (1 - \frac{1}{\varrho^2})\,m_{22} + \frac{T^2}{\varrho^2}\sigma_A^2\,, \end{aligned} \tag{D.19}$$

$$k_1 = \frac{m_{11}}{m_{11} + \sigma_N^2} \;\Leftrightarrow\; m_{11} = \frac{k_1}{1 - k_1}\sigma_N^2\,, \tag{D.20}$$

$$k_2 = \frac{m_{13}}{m_{11} + \sigma_N^2} \;\Leftrightarrow\; m_{12} = \frac{k_2}{1 - k_1}\sigma_N^2\,. \tag{D.21}$$

The steady-state solutions can be obtained in the following calculations. From (D.18), we have

$$m_{22} = \frac{\varrho^2}{T}\,(k_1 + \frac{1}{\varrho} - 1)\,m_{12} + T^2\sigma_A^2 \tag{D.22}$$

and from (D.19), we have

$$T^2\sigma_A^2 = \varrho^2 k_2 m_{12} + (1-\varrho^2)\,m_{22} \tag{D.23}$$

then (D.22) and (D.23) imply

$$m_{22} = [\frac{1}{T}(k_1 + \frac{1}{\varrho} - 1) + k_2]\,m_{12}\,. \tag{D.24}$$

Now, substituting (D.20), (D.21) and (D.24) into (D.17), we obtain

$$\frac{k_1^2}{(1-k_1)}\sigma_N^2 = \frac{2T}{\varrho}\frac{k_2}{(1-k_1)}\sigma_N^2 - \frac{T^2}{\varrho^2}\,[\frac{1}{T}(k_1 + \frac{1}{\varrho} - 1) + k_2]\,\frac{k_2}{(1-k_1)}\sigma_N^2 + \frac{T^4}{\varrho^2}\sigma_A^2\,, \tag{D.25}$$

and substituting(D.21) and (D.24) into (D.19), we obtain

$$\frac{T^2}{\varrho^2}\sigma_A^2 = \frac{k_2^2}{(1-k_1)}\sigma_N^2 + (\frac{1}{\varrho^2} - 1)[\frac{1}{T}(k_1 + \frac{1}{\varrho} - 1) + k_2]\,\frac{k_2}{(1-k_1)}\sigma_N^2\,. \tag{D.26}$$

Then substituting (D.26) into (D.25) and canceling like terms, we obtain

$$\begin{aligned} k_1^2 &= (\frac{1}{\varrho} + 1)\,Tk_2 - k_1k_2T \qquad \text{(D.27)}\\ &\Leftrightarrow\; k_1^2 + k_1k_2T - (\frac{1}{\varrho} + 1)\,k_2T = 0\,. \end{aligned}$$

Substituting the dimensionless quantities $\alpha = k_1$ and $\beta = k_2T$ and defining $\hat{\rho} \triangleq (\frac{1}{\varrho} + 1)$, we obtain a quadratic equation in terms of the two Kalman gain parameters of interest,

$$\alpha^2 + \alpha\beta - \hat{\rho}\beta = 0\,. \tag{D.28}$$

Solving for α, we get

$$\alpha = [\frac{1}{4}\beta^2 + \hat{\rho}\beta]^{\frac{1}{2}} - \frac{1}{2}\beta\,. \tag{D.29}$$

Since the gain values are positive, α is the positive root of the second-order polynomial in (D.28). To solve for α and β (therefore, $\mathbf{K}$, $\mathbf{M}$ and $\mathbf{P}$), we need to obtain another equation relating α and β. We continue the calculations by substituting (D.21) and (D.24) into (D.19) to obtain

$$T^2k_2^2 = (1 - \frac{1}{\varrho^2})\,\varrho^2 k_1 k_2 T + (1 - \frac{1}{\varrho^2})\varrho^2(\frac{1}{\varrho} - 1)\,k_2 T + \frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2}(1 - k_1)\,. \tag{D.30}$$

If we define $\tilde{\rho}_1 \triangleq (1 - \frac{1}{\varrho^2})\,\varrho^2 = (\varrho^2 - 1)$ and $\tilde{\rho}_2 \triangleq (1 - \frac{1}{\varrho^2})\,\varrho^2(\frac{1}{\varrho} - 1) = \tilde{\rho}_1(\frac{1}{\varrho} - 1)$ which are functions of ϱ, and $\gamma = \frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2}$, and substitute k_1 and k_2T with α and β, (D.30) becomes

$$\beta^2 = \tilde{\rho}_1\alpha\beta + \tilde{\rho}_2\beta + \gamma(1 - \alpha)\,. \tag{D.31}$$

Solving for α, we get

$$\alpha = [\tilde{\rho}_1\beta - \gamma]^{-1}\,[\beta^2 - \tilde{\rho}_2\beta - \gamma]. \tag{D.32}$$

Equating α from (D.29) and (D.32), *i.e.*,

$$[\frac{1}{4}\beta^2 + \hat{\rho}\beta]^{\frac{1}{2}} - \frac{1}{2}\beta = [\tilde{\rho}_1\beta - \gamma]^{-1}\,[\beta^2 - \tilde{\rho}_2\beta - \gamma]\,, \tag{D.33}$$

we obtain an equation in terms of β only. Simplifying further, we get

$$c_4\,\beta^4 + c_3\,\beta^3 + c_2, \beta^2 + c_1\,\beta + c_0 = 0 \tag{D.34a}$$

where

$$
\begin{aligned}
c_4 &= 1 + \tilde{\rho}_1 = \varrho^2 \\
c_3 &= -\,2\tilde{\rho}_2 - \tilde{\rho}_1\tilde{\rho}_2 - \tilde{\rho}_1^2\hat{\rho} - \gamma \\
&= -\,2(\varrho^2-1)(\frac{1}{\varrho}-1) - \frac{2}{\varrho}(\varrho^2-1)^2 - \frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2} \\
c_2 &= \tilde{\rho}_2^2 - (\tilde{\rho}_2 - \tilde{\rho}_1)\,\gamma + 2\tilde{\rho}_1\hat{\rho}\gamma - 2\gamma \\
&= (\varrho^2-1)^2(\varrho-1)^2 + \frac{3}{\varrho}(\varrho^2-1)(\frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2}) - 2(\frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2}) \qquad \text{(D.34b)} \\
c_1 &= 2\tilde{\rho}_2\gamma + (1-\hat{\rho})\gamma^2 \\
&= 2(\varrho^2-1)^2(\frac{1}{\varrho}-1)^2(\frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2}) - \frac{1}{\varrho}(\frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2})^2 \\
c_0 &= \gamma^2 = (\frac{T^4}{\varrho^2}\frac{\sigma_A^2}{\sigma_N^2})^2.
\end{aligned}
$$

Given T, σ_A^2 and σ_N^2, we can numerically solve for β based on the roots of the of fourth-order equation in (D.34a). Of the four roots, there will be only one positive root, one negative root and two complex conjugate roots. As we have discussed previously in Sec. D.3, k_2 is positive and therefore β equals the positive root of (D.34a).

Bibliography

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," in *IEEE Commun. Mag.*, Aug. 2002, pp. 102–114.

[2] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. Int. Conf. Acoust. Speech, Signal Processing*, Salt Lakes City, UT, May 2001.

[3] T. Pham and H. Papadopoulos, "Distributed tracking in ad-hoc sensor networks," in *Proc. 2005 IEEE Workshop on Stat. Signal Proc. (SSP)*, Bordeaux, France, July 2005.

[4] R. Brooks, P. Ramanathan, and A. Sayeed, "Distributed target classification and tracking in sensor networks," in *Proc. IEEE*, vol. 91, Aug. 2003, pp. 1163 – 1171.

[5] "Collaborative signal and information processing in microsensor networks (special issue)," in *IEEE Sig. Proc. Mag.*, Mar. 2002.

[6] T. Pham, B. Sadler, and H. Papadopoulos, "Energy based source localization via ad-hoc acoustic sensor network," in *Proc. 2003 IEEE Workshop on Stat. Signal Proc. (SSP)*, St. Louis, MO, Sept. 2003, pp. 387 – 390.

[7] M. Honikel, "Improvement of insar dem accuracy using data and sensor fusion," in *Proc. IEEE Int. Symp. Geoscience and Remote Sensing)*, vol. 5, July 1998, pp. 2348 – 2350.

[8] N. Bertoldo, S. Hunter, R. Fertig, G. Laguna, and D. MacQueen, "Development of a real-time radiological area monitoring network for emergency response at lawrence livermore national laboratory," *IEEE Journal on Sensors*, vol. 5, pp. 565 – 573, Aug. 2005.

[9] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh, "Monitoring volcanic eruptions with a wireless sensor network," in *Proc. Second European Workshop on Wireless Sensor Networks*, Jan. 2005, pp. 108 – 120.

[10] J. Kumagai and S. Cherry, "Sensors and sensibility," in *IEEE Spectrum*, vol. 2, no. 7, July 2004, pp. 22–28.

[11] J. Ng, "Ubiquitous healthcare localisation schemes," in *Proc. of 7th International Workshop on Enterprise networking and Computing in Healthcare Industry (HEALTHCOM 2005)*, June 2005, pp. 156 – 161.

[12] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet," in *Proc. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Oct. 2002, pp. 96–197.

[13] H. Wang, J. Elson, L. Girod, D. Estrin, and K. Yao, "Target classification and localization in habitat monitoring," in *ICASSP*, vol. 4, Apr. 2003, pp. 844–847.

[14] H. Durrant-Whyte, B. Rao, and H. Hu, "Data fusion in decentralised sensing networks," in *Proc. of IEEE Inter. Conf. on Robotics and Automation,*, vol. 2, May 1990, pp. 1331 – 1336.

[15] B. S. Rao and H. F. Durrant-Whyte, "Fully decentralized algorithm for multisensor kalman fltering," in *IEE Proceedings-D*, vol. 138, no. 5, Sept. 1991, pp. 413–420.

[16] D. Nicholson, C. Lloyd, S. Julier, and J. Uhlmann, "Scalable distributed data fusion," in *Proc. Fifth Int. Conf. on Infor. Fusion*, vol. 1, no. 5, July 2002, pp. 630–635.

[17] D. S. Scherber and H. C. Papadopoulos, "Distributed computation of averages over ad hoc networks," *IEEE J. Selected Areas Commun.*, vol. 23, no. 4, pp. 776–787, Apr. 2005.

[18] D. Scherber and H. Papadopoulos, "Locally constructed algorithms for distributed computations in ad-hoc networks," in *Proc. Third Inter. Symp. Inform. Procesing. for Sensor Networks (ISPN)*, Berkeley, CA.

[19] T. Pham, D. Scherber, and H. Papadopoulos, "Distributed source localization algorithms for acoustic ad-hoc sensor networks," in *Proc. 2004 IEEE Workshop on Sensor Array and Multichannel Sig. Proc.(SAM)*, Barcelona, Spain, July 2004, pp. 613–617.

[20] T. Pham and H. Papadopoulos, "Distributed source localization and tracking in ad-hoc sensor networks," *submitted to IEEE Trans. Signal Processing*, Mar. 2006.

[21] A. Ledeczi, P. Volgyesi, M. Maroti, G. Simon, G. Balogh, A. Nadas, B. Kusy, S. Dora, and G. Pap, "Multiple simultaneous acoustic source localization in urban terrain," in *Proc. Fourth Inter. Symp. Inform. Procesing. for Sensor Networks*, Berkeley, CA, Apr. 2005, pp. 491 – 496.

[22] G. Simon, G. Balogh, G. Pap, M. Maróti, B. Kusy, J. Sallai, A. Lédeczi, A. Nádas, and K. Frampton, "Sensor network-based countersniper system," in *Proc. ACM SenSys'04*, Baltimore, MD.

[23] T. Pham and B. Salder, "Adaptive wideband aeroacoustic array processing," in *Proc. 1996 IEEE Workshop on Stat. Signal and Array Proc. (SSAP)*, Corfu, Greece, June 1996, pp. 295–298.

[24] Y. Bar-Shalom, X. Li, and T. Kirubargjan, *Estimation with Applications to Tracking and Navigation.* New York, NY: John Wiley & Sons, Inc, 2001.

[25] G. Jacyna, C. Christou, B. George, and B. Necioglu, "Netted sensors-based vehicle acoustic classification at tier 1 nodes," in *Proc. SPIE: Unattended Ground Sensor Technologies and Applications VII*, vol. 5796, no. 5, 2005, pp. 404–415.

[26] R. Kozick and B. Sadler, "Performance of doppler estimation for acoustic sources with atmospheric scattering," in *Proc. Int. Conf. Acoust. Speech, Signal Processing*, Montreal, Canada, May 2004.

[27] R. Kozick, B. Sadler, and D. Wilson, *Distributed Sensor Networks: Signal Processing and Propagation for Aeroacoustic Sensor Networking*, S. Iyengar and R. Brooks, Eds. New York: CRC Press, 2004.

[28] T. Embleton, "Tutorial on sound propagation outdoors," *J. Acoust. Soc. Am.*, vol. 100, pp. 31–38, Apr. 1996.

[29] E. Salomons, *Computational Atmospheric Acoustics*, 1st ed. Dordrecht: Kluwer Academic Publishers, Aug. 2001.

[30] G. Daigle, J. Piercy, and T. Embleton, "Line-of-sight propagation through atmospheric turbulence near the ground," *J. Acoust. Soc. Am.*, vol. 74, pp. 1505–1513, 1983.

[31] D. Wilson, "Performance bounds for acoustic direction-of-arrival arrays operating in atmospheric turbulence," *J. Acoust. Soc. Am.*, vol. 103, pp. 1306–1319, Mar. 1998.

[32] D. Wilson, B. M. Sadler, and T. Pham, "Simulation of detection and beamforming with acoustical ground sensors," in *Proc. of SPIE*, Orlando, FL, Apr. 2002, pp. 50–61.

[33] M. Crocker, *Handbook of Acoustics*, 2nd ed. New York, NY: John Wiley & Son, Inc., 1998.

[34] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the node," in *IEEE Signal Processing Magazine*, July 2005, pp. 54–68.

[35] K. Frampton, “Acoustic self-localization in a distributed sensor network,” *IEEE Journal on Sensors*, vol. 6, pp. 166 – 172, Feb. 2006.

[36] J. Chen, K. Yao, and R. Hudson, “Source localization and beamforming,” in *IEEE Sig. Proc. Mag.*, vol. 19, Mar. 2002, pp. 30–39.

[37] W. Chen, J. Hou, and S. L. Sha, “Dynamic clustering for acoustic target tracking in wireless sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 3, pp. 258 – 271, July 2002.

[38] X. Sheng and Y. Hu, “Energy based acoustic source localization,” in *Proc. Second Inter. Symp. Inform. Procesing. for Sensor Networks (IPSN)*, Palo Alto, CA, Apr. 2003.

[39] X. Zhang and S. Wicker, “How to distribute sensors in a random filed?” in *Proc. Third Inter. Symp. Inform. Procesing. for Sensor Networks (IPSN)*, Berkeley, CA, Apr. 2004, pp. 243–250.

[40] A. Hayes, A. Martinoli, and R. Goodman, “Distributed odor source localization,” *IEEE Journal on Sensors*, vol. 2, pp. 260 –271, June 2002.

[41] C. Chang and A. Sahai, “Estimation bounds for localization,” in *Proc. IEEE Communications Society Conf. on Sensor and Ad Hoc Communications and Networks*, Oct. 2004, pp. 415 – 424.

[42] H. Wang, L. Yip, K. Yao, and D. Estrin, “Lower bounds of localization uncertainty in sensor networks,” in *ICASSP*, vol. 3, May 2004, pp. 917–920.

[43] R. McAulay and E. Hofstetter, “Barankin bounds on parameter estimation,” *IEEE Trans. Inform. Theory*, vol. 17, no. 11, pp. 669–676.

[44] S. Kay, *Fundamentals of Statistical Signal Processing Estimation Theory.* Upper Saddle River, NJ: Prentice Hall, 1993.

[45] P. Stoica and A. Nehorai, “Performance study of conditional and unconditional direction-of-arrival estimation,” *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, no. 10, pp. 1783–1795, Oct. 1990.

[46] D. Li, K. Wong, Y. H. Hu, and A. Sayeed, “Detection, classification, and tracking of targets,” in *IEEE Sig. Proc. Mag.*, vol. 19, Mar. 2002, pp. 17–29.

[47] L. Kaplan, “Node selection for target tracking using bearing measurements from unattended ground sensors,” in *Proc. IEEE 2003 Aerospace Conference*, vol. 5, Mar. 2003, pp. 2137– 2152.

[48] Q. Le, L. Kaplan, and J. McClellan, "Multiple-mode kalman filtering with node selection using bearings-only measurements," in *Proc. the Thirty-Sixth Southeastern Symposium on System Theory*, Berkeley, CA, 2004, pp. 185 – 189.

[49] H. Durrant-Whyte, M. Stevens, and E. Nettleton, "Data fusion in decentralised sensing networks," in *Proc. of Fusion*, Montreal, Canada, July 2001, pp. 302–307.

[50] J. Manyika and H. Durrant-Whyte, *Data Fusion and Sensor Management: A Decentralized Information-Theoretic Approach*, 1st ed. Uppler Saddler, NJ: Prentice Hall, 1995.

[51] B. S. Rao and H. F. Durrant-Whyte, "A decentralized bayesian algorithm for identification," in *Proc. of the 29th IEEE Conf. on Decision and Control,*, vol. 2, Dec. 1990, pp. 827 – 828.

[52] J. Tsitsiklis and M. Athans, "On the complexity of decentralized decision making and detection problems," *IEEE Trans. Automatic Control*, vol. AC-30, no. 5, pp. 440 – 446, May 1985.

[53] V. Blondel, J. Hendricks, A. Olshevsky, and J. Tsitsiklis, "Convergence in multiagent coordination, consensus, and flocking," in *Proc. of Joint 44th IEEE Conf. on Dec. and Cont and Euro. Control. Conf. (CDC-ECC'05)*, Seville, Spain, Dec. 2005.

[54] N. Minar, M. Gray, O. Poop, R. Krikorian, and P. Maes, "Hive: distributed agents for networking things," *IEEE Parallel Distributed Technology*, vol. 8, pp. 24 – 33, 2000.

[55] S. Staab, P. Domingos, P. Mike, J. Golbeck, L. Ding, T. Finin, A. Joshi, A. Nowak, and R. Vallacher, "Collaborative signal and information processing in microsensor networks (special issue)," *IEEE Intelligent Systems*, vol. 20, pp. 80–93, 2005.

[56] J. Proakis, *Digital Communications*, 4th ed. New York, NY: McGraw-Hill, 2001.

[57] R. Horn and C. Johnson, *Matrix Analysis*, 1st ed. Cambridge University Press, 1979.

[58] B. Ermentrout, "An adaptive model for synchrony in the firefly *pteroptyx malaccae*," *J. of Math. Biology*, vol. 29, no. 6, pp. 571–585, June 1991.

[59] M. Patil, U. Shaha, U. Desai, and S. Merchant, "Localization in wireless sensor networks using three masters," in *Proc. IEEE Int. Conf. on Personal Wireless Comm. (ICPWC 2005)*, pp. 384 –388.

[60] M. Coates, "Distributed particle filters for sensor networks," in *Proc. Third Inter. Symp. Inform. Procesing. for Sensor Networks*, Berkeley, CA, Apr. 2004, pp. 99–107.

[61] W. Garber and R. Moses, "Self-localization of three dimension sensor networks," in *Proc. 2003 IEEE Workshop on Stat. Signal Processing (SSP)*, St. Louis, MO, Sept. 2003.

[62] A. Ihler, J. Fisher, R. Moses, and A. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Selected Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

[63] D. Lazos, R. Poovendran, and S. Capkun, "ROPE: Robust position estimation in wireless sensor networks," in *Proc. Fourth Inter. Symp. Inform. Procesing. for Sensor Networks*, Berkeley, CA, Apr. 2005, pp. 324–331.

[64] A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattern, "Distributed online localization in sensor networks using a moving target," in *Proc. Fourth Inter. Symp. Inform. Procesing. for Sensor Networks*, Apr. 2004, pp. 61–70.

[65] B. Friedlander, "A passive localization algorithm and its accuracy and analysis," *IEEE J. of Oceanic Eng*, vol. 12, no. 1, pp. 234–245, Dec. 1987.

[66] J. Smith and J. Abel, "Closed-forms least-squares source location estimation from range-difference measurements," *Proc. Int. Conf. Acoust. Speech, Signal Processing*, vol. ASSP-35, no. 12, pp. 1661–1669, Dec. 1987.

[67] N. Srour and J. Robertson, "Final report: Remote netted acoustic detection system," *US Army Research Laboratory*, vol. ARL-TR-706, May 1994.

[68] X. Sheng and Y. Hu, "Distributed particle filters for wireless sensor network target tracking," in *Proc. Int. Conf. Acoust. Speech, Signal Processing*, vol. 4, Palo Alto, CA, Mar. 2005, pp. 845–848.

[69] S. Oh and S. Sastry, "Tracking on a graph," in *Proc. Fourth Inter. Symp. Inform. Procesing. for Sensor Networks*, Los Angeles, CA, Apr. 2005, pp. 195 – 202.

[70] D. Spanos, R. Olfti-Saber, and R. Murray, "Approximate distributed kalman filtering in sensor networks with quantifiable performance," in *Proc. Fourth Inter. Symp. Inform. Procesing. for Sensor Networks*, Los Angeles, CA, Apr. 2005, pp. 133 – 139.

[71] S. Balasubramanian, S. Jayaweera, and K. Namuduri, "Energy-aware, collaborative tracking with ad-hoc wireles sensor netowrks," in *Proc. IEEE Wireless Communications and Networking Conf. (WCNC)*, vol. 3, Mar. 2005, pp. 1878 – 1883.

[72] C. Chong, F. Zhao, S. Mori, and S. Kumar, "Distributed tracking in wireless ad hoc sensor networks," *IEEE Commun. Mag.*, pp. 102–114, Aug. 2002.

[73] B. Anderson and J. Moore, *Optimal Filtering*, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1979.

[74] http://www.usukita.com/.

[75] S. Balasubramanian, I. Elangovan, S. Jayaweera, and K. Namuduri, "Distributed and collaborative tracking for energy-constrained ad-hoc wireless sensor networks," in *Proc. IEEE Wireless Communications and Networking Conf. (WCNC)*, Atlanta, GA, Mar. 2004, pp. 1732–1737.

[76] V. Poor, *An Introduction to Signal Detection and Estimation*, 2nd ed. New York, NY: Springer-Verlag, 1994.