

UNIVERSITY OF CALIFORNIA
Santa Barbara

Networked Estimation and Communication with
Minimalist Models

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

by

Sriram Venkateswaran

Committee in Charge:

Professor Upamanyu Madhow, Chair

Professor João P. Hespanha

Professor Michael Liebling

Professor Bangalore S. Manjunath

Professor Kenneth Rose

Professor Subhash Suri

December 2011

The Dissertation of
Sriram Venkateswaran is approved:

Professor João P. Hespanha

Professor Michael Liebling

Professor Bangalore S. Manjunath

Professor Kenneth Rose

Professor Subhash Suri

Professor Upamanyu Madhow, Committee Chairperson

December 2011

Networked Estimation and Communication with Minimalist Models

Copyright © 2011

by

Sriram Venkateswaran

Acknowledgements

Over the course of the last five years, Professor Madhow has changed the way I think and express myself. For this, I am deeply thankful to him. I have learnt from him that it is vital to understand the larger picture before solving specific problems. His insightful feedback on the paper drafts I sent him and the oral presentations I made has taught me the importance of clarity and structure while conveying ideas in any form.

Translating ideas from this dissertation into a demonstration for the source localization and UAV routing project involved a lot of work. Jason and Danny provided great company that made this work enjoyable. I would like to thank Professor Hespanha for his guidance on this project and all my committee members for their feedback. Many ideas in this dissertation were shaped and refined during discussions with my labmates and I thank them for their inputs.

I would like to thank Professor Koilpillai at IIT Madras for the extraordinary lengths he went to help me and his guidance.

Amidst a very busy life, my parents have always done their best for me. They have invested time and effort in making the right choices, taught me numerous things, given constructive advice and been generous with love and encouragement. Thanks for all this and much more. My grandparents, Kamba and Thatha, have

given me unconditional love and unstinting support. My uncle acted as the catalyst in my shift to a healthier lifestyle. Thanks to all of them too.

Each one of my housemates over the years – Karthik, Vivek, Prakash, Sandeep, Jefy, Bala, Ajay and Jalan – has made daily life, far away from family, enjoyable in his own way. Sumit has been a great sounding board and I would like to thank him for this. Thanks also go to many friends who have shared parts of what has been a very interesting journey.

Curriculum Vitæ

Sriram Venkateswaran

Education

December 2011	Doctor of Philosophy, Electrical and Computer Engineering University of California, Santa Barbara
December 2007	Master of Science, Electrical and Computer Engineering University of California, Santa Barbara
August 2006	Bachelor of Technology, Electrical Engineering Indian Institute of Technology Madras

Publications

Journals

- S. Venkateswaran and U. Madhow. “Localizing multiple events using times of arrival”, *IEEE Transactions on Signal Processing* (submitted)
- D.J. Klein, S. Venkateswaran, J.T. Isaacs, J. Burman, T. Pham, J.P. Hespanha, U. Madhow. “Source Localization in a Sparse Acoustic Sensor Network using UAV-based Semantic Data Mules”. *ACM Transactions on Sensor Networks* (submitted)

Conferences

- S. Venkateswaran and U. Madhow. “Collaborative Estimation in Dispersive Environments: A Frequency Domain Approach”. *Asilomar Conference on Signals, Systems and Computers*, November 2011, Asilomar, CA
- S. Venkateswaran and U. Madhow. “Space-time localization using Times of Arrival”. *Allerton Conference on Communication, Control and Computing*, September 2011, Monticello, IL
- S. Venkateswaran and U. Madhow. “Implicit Network Timing Synchronization With Phase-Only Updates”. *Conference on Information Sciences and Systems*, March 2011, Baltimore, MD
- S. Venkateswaran, S. Singh, U. Madhow, R. Mudumbai. “Distributed Synchronization and Medium Access in Wireless Mesh Networks”. *Information Theory and Applications* (invited paper), February 2011, San Diego, CA
- J. Burman, J. Hespanha, U. Madhow, T. Pham, J. Isaacs and S. Venkateswaran. “Bio-inspired UAV Routing, Source Localization and Acoustic Signature Classification for persistent surveillance”. *SPIE Defense, Security and Sensing Symposium*, 2011, Orlando, FL

- J. Burman, J. Hespanha, U. Madhow, D. Klein, J. Isaacs, S. Venkateswaran and T. Pham. “Heterogeneous battlefield sensor networks : a bio-inspired overlay architecture”. *Military Sensing Symposium*, 2010
- J. Burman, J. Hespanha, U. Madhow, D. Klein, T. Pham, J. Isaacs and S. Venkateswaran. “Heterogeneous sensor networks : a bio-inspired overlay architecture”. *SPIE Defense, Security and Sensing Symposium*. 2010, Orlando, FL
- H. Zhang, S. Venkateswaran and U. Madhow. “Channel Modeling and MIMO Capacity For Outdoor Millimeter Wave Links”. *Wireless Communications and Networking Conference*, April 2010, Sydney, Australia
- S. Venkateswaran and U. Madhow. “Distributed Detection With A Minimalistic Signal Model : A Framework For Exploiting Correlated Sensing”. *International Symposium on Information Theory*, July 2008, Toronto, Canada

Abstract

Networked Estimation and Communication with Minimalist Models

Sriram Venkateswaran

We provide three examples to show that we can solve complex problems in sensor networks even with minimalist observation and communication models.

First, we propose a scheme to maintain synchrony in a Time Division Multiplexed network with minimal overhead. Each node estimates the offset in its clock phase with its neighbors based on the differences between the expected and actual times at which it receives communication packets. Using such estimates, the nodes adjust their clock phases every time they receive a packet and also adjust their clock frequencies on a slower timescale. We provide insight by analyzing a simpler “averaged” system and use simulations to demonstrate the efficacy of the algorithm.

Next, we consider the problem of localizing multiple events that are closely spaced in time, based solely on their Times of Arrival (ToAs) at different sensors. The challenge is to identify and group the ToAs belonging to a given event. The naive approach of trying all possible groupings suffers from excessive complexity. We design a three-stage algorithm to sidestep such bottlenecks. The simplification

comes from the first stage, where we discretize the times at which events occur to reduce the set of event candidates considerably. However, some of these candidates are “phantoms” that arise because we do not know the correct groupings. We refine the estimates in a Bayesian manner and solve a matching problem on a graph to reject the phantoms and group the ToAs. We use simulations to illustrate the near-optimal localization performance.

Finally, we consider the problem of estimating an unknown signal recorded at multiple sensors through an unknown dispersive environment. We parallelize the problem by solving it in the frequency domain. We first estimate the signal over small bands efficiently, up to a scale factor. We then estimate the scale factors by choosing the small bands to have significant overlap. We show via experiments and simulations that the algorithm is effective in reconstructing signals with “moderate” bandwidths. For signals with larger bandwidths, we demonstrate fundamental ambiguities in the form of multiple source signals explaining the recorded observations.

Contents

Acknowledgements	iv
Curriculum Vitæ	vi
Abstract	viii
List of Figures	xiii
List of Tables	xvii
1 Introduction	1
1.1 Implicit network timing synchronization	4
1.2 Localizing multiple events from Times of Arrival	7
1.3 Collaborative Estimation in Dispersive Environments	11
1.4 Outline	14
2 Distributed Implicit Network Timing Synchronization	15
2.1 Related Work	18
2.2 System Model	21
2.3 Phase-only Adjustments	28
2.3.1 Averaged System	29
2.3.2 Actual System	32
2.4 Design of Phase-Frequency Adjustments	35
2.4.1 Choosing round sizes	38
2.4.2 Convergence of frequency adjustment algorithm	41
2.5 LLN Arguments	54
2.6 Simulation Results	63
2.6.1 Phase-Only Adjustments	65

2.6.2	Phase & Frequency Adjustments	67
3	Space-time localization using times of arrival	81
3.1	Related Work	84
3.2	System Model	85
3.3	Feasibility of localizing multiple events	87
3.4	Algorithm Overview	97
3.5	Stage 1: Generating Candidate Events	102
3.6	Stage 2: Refining the Estimates	113
3.7	Stage 3: Picking true events from the palette	115
3.8	Simulation Results	124
4	Collaborative Estimation in Dispersive Environments	128
4.1	Related Work	130
4.2	System Model	132
4.3	Signal Estimation Algorithm	137
4.3.1	Frequency domain channel model	138
4.3.2	Stage 1: Estimation within a band	139
4.3.3	Stage 2: L-to-R Stitching Algorithm	144
4.3.4	Reconstructing the source signal	150
4.4	Experimental Results	151
4.5	Simulation Results	159
4.6	Multiple Explanations	163
4.6.1	Distorting the outputs of Stage 1	165
4.6.2	Global Stitching Algorithm	166
4.6.3	Simulation Results	172
5	Conclusions	176
5.1	Implicit Timing Synchronization	176
5.2	Localizing multiple events from ToAs	178
5.3	Collaborative Estimation In Dispersive Environments	179
5.4	Minimalism all the way	180
	Appendices	184
A		185
A.1	An expression for the excess phases $\varphi_{ex}[s]$ in the averaged system	185
A.2	Linear Programming Formulation	187

A.3	Actual System - Phase Only Adjustments	191
A.4	Estimating skews from raw phases	193
A.5	Expression for the skew estimate - Averaged system	194
A.6	Evolution of the excess phases across a round	195
A.7	Recursive bounds on the excess phases	196
A.8	LLN arguments for the actual system	198
B		203
	Bibliography	205

List of Figures

1.1	Three events, that we call “Blue (B)”, “Green (G)” and “Red (R)”, happen close to one another in time and produce ToAs at 8 sensors. The ToAs at each sensor are sorted in ascending order. The red arrows connect the ToAs produced by the red event and so on. Note that the events need not arrive at the sensors in the same order: for example, the order of ToAs at sensor 1 is RGB, whereas it is BRG at sensor 2. We ask: (a) under what conditions can we group the ToAs appropriately – draw the arrows that connect ToAs produced by the same event – and localize the events and (b) how do we do this in a robust fashion with low complexity? Note that we can have outlier ToAs at some sensors, such as the ToA in the orange bubble at sensor 1, which must be discarded. Additionally, some sensors might miss an event and not have a corresponding ToA, but this is not shown in the figure.	9
1.2	A source is recorded at multiple sensors through dispersive channels. We design an estimation algorithm to reconstruct the source from the faded signals.	12
2.1	Nodes make <i>phase jumps</i> each time they receive a packet. However, they change their frequencies (slope of the lines) only at the end of a <i>round</i> consisting of “many” slots.	26
2.2	Splitting the nodes into 10 sets based on their excess frequencies. Nodes in S_1 and S_2 , that are “far” from convergence, are guaranteed to reduce their frequencies. Nodes in S_3 and S_4 , that are “close” to convergence, either reduce their frequencies or do not change it, but never increase their frequencies. Nodes in S_5 , that are “closest” to convergence, will not change their frequencies. Analogous results hold for nodes with negative excess frequencies.	43

2.3	Worst error between neighbors for the actual system and the averaged system with only phase adjustments in a <i>directional network</i>	67
2.4	Worst error between neighbors for the actual system and the averaged system with only phase adjustments in an <i>omnidirectional network</i> . The network operates in the ONLYINTENDED mode.	68
2.5	Frequency deviations of all 36 nodes. Skews are <i>randomly</i> distributed and measurements are <i>noiseless</i>	71
2.6	Frequency deviations of all 36 nodes in an <i>omnidirectional</i> network with a <i>grid</i> topology. Skews are <i>randomly</i> distributed and measurements are <i>noisy</i>	72
2.7	Network wide frequency error in a <i>directional</i> setting with a <i>grid</i> topology. Skews are distributed <i>randomly</i>	73
2.8	Network wide frequency error in a <i>directional</i> setting with a <i>ring</i> topology. Skews are <i>randomly</i> distributed.	73
2.9	Network wide frequency error in an <i>omnidirectional</i> setting with a <i>grid</i> topology. Skews are <i>randomly</i> distributed.	74
2.10	Network wide frequency error in an <i>omnidirectional</i> setting with a <i>ring</i> topology. Skews are <i>randomly</i> distributed.	74
2.11	Worst phase error between neighbors in a <i>directional</i> setting with <i>noisy</i> measurements. Skews are <i>randomly</i> distributed.	76
2.12	Worst error between neighbors in an <i>omnidirectional</i> setting with <i>randomly</i> distributed skews. Measurements are <i>noisy</i> and nodes are in the ONLYINTENDED mode.	76
2.13	Worst error between neighbors in an <i>omnidirectional</i> setting with <i>randomly</i> distributed skews. Measurements are <i>noisy</i> and nodes are in the EAVESDROP mode.	77
2.14	Worst phase error between neighbors in a <i>directional</i> setting. Skews are <i>badly</i> distributed and measurements are <i>noisy</i>	78
2.15	Worst phase error between neighbors in an <i>omnidirectional</i> setting with a <i>grid</i> topology. Skews are <i>badly</i> distributed, measurements are <i>noisy</i>	78
2.16	Worst phase error between neighbors in an <i>omnidirectional</i> setting with a <i>ring</i> topology. Skews are <i>badly</i> distributed, measurements are <i>noisy</i>	79

3.1	Each one of the sensors shown by the pink dots record two ToAs – one from Event 1 and the other from Event 2, whose locations are shown by the black triangles. However, events A and B, shown by the red squares, also produce the same set of ToAs at all the sensors. Therefore, the sensors are unable to decide which of the event sets $\{\mathcal{E}_a, \mathcal{E}_b\}$ and $\{\mathcal{E}_1, \mathcal{E}_2\}$ occurred.	96
3.2	Geometry of the processing in Stage 1. Six sensors $s, s', s_1, s_2, s_3, s_4$ are shown. Sensors s and s' have two ToAs each, denoted by $\{\tau_1(s), \tau_2(s)\}$ and $\{\tau_1(s'), \tau_2(s')\}$. ToAs $\tau_1(s)$ and $\tau_1(s')$ were produced by an event \mathcal{E} that occurred at time $t_e \approx l\epsilon$. Consider a hypothesized event time $u = l\epsilon$ and draw circles C_{1s} and C_{2s} , centered at sensor s , with radii $\tau_s(1) - u$ and $\tau_2(s) - u$ (likewise for $C_{1s'}$ and $C_{2s'}$). C_{1s} and $C_{1s'}$ intersect at a point \hat{e} close to \mathcal{E} 's location. All other points of intersection between C_{is} and $C_{js'} \forall i, j$ (denoted by $p_i, i = 1, \dots, 5$) are phantom estimates.	98
3.3	Modified version of matching problem on a bipartite graph. Events in the palette are shown as blue circles and the observations at sensors are shown as blue stars. Green circles represent events that are picked while red circles denote phantom events. We need to draw edges between the picked events and the observations, subject to constraints, so as to maximize the sum of the values of the edges.	116
3.4	Localization errors with the proposed algorithm and a genie-based scheme with $N = 8$ and $N = 16$ sensors. The errors virtually coincide with one another, demonstrating the efficacy of the proposed algorithm.	127
4.1	The figure shows our choice of overlapping frequency bands. The signal and the channel estimate samples in the crossed squares are used to determine the scale factors z_b in different bands.	144
4.2	Illustrating a “hole” in the signal spectrum. Two bands of high energy flank a band with relatively low energy. The reconstruction procedure works fine in the flanking bands individually. However, the overall reconstruction is poor due to the loss in continuity because of the low energy band in between.	150
4.3	An estimate of the indoor propagation channel	154
4.4	The topmost plot shows the true Chirp200 waveform, with a constant envelope. The following four plots show the recorded waveforms at different sensors. Notice that these waveforms undergo “deep fades” and no longer have a constant envelope. The final plot shows the reconstructed Chirp200 waveform, whose envelope shows lesser variation, illustrating the benefits of the L-to-R algorithm.	157

4.5	Optional caption for list of figures	174
-----	--	-----

List of Tables

4.1	Results of L-to-R processing and single tap approximation of the recorded signals.	158
4.2	Fit between source and estimate in bands of width 50 Hz is very good. Band i spans the frequencies $[1000 + 50(i - 1), 1000 + 50i]$ Hz. .	159
4.3	Delay between the true source and the estimate over bands of width 50 Hz (in samples @ $f_s = 16$ kHz). We see that the estimates in different bands have different delays with respect to the source. Band i spans the frequencies $[1000 + 50(i - 1), 1000 + 50i]$ Hz.	159
4.4	Performance of the L-to-R stitching algorithm and the SVD Estimate with Chirp and “Random” signals of varying bandwidths.	161

Chapter 1

Introduction

Sensors are everywhere. A smartphone that fits in the palm of our hand includes a microphone to record sounds, a camera to take photographs, a screen that responds to touch, a Global Positioning System (GPS) unit to locate the phone on the surface of the Earth, an accelerometer to estimate the phone's orientation and a temperature sensor to prevent the phone from overheating. Sensors have proliferated in this fashion mainly due to advances in miniaturizing them, making convenient device form factors feasible. Over the past decade, these advances, coupled with ever-increasing computing capacity, have stimulated research in the allied field of *sensor networks*. In a sensor network, multiple sensors are used to observe the surrounding environment. The sensors then pool their observations and use their computing capabilities to make “interesting” inferences about the surroundings. For example, sensor networks have now been deployed for a variety of purposes such as military surveillance, monitoring the health of buildings, ob-

serving seabirds, studying volcanoes and localizing woodpeckers from their calls. While these deployments differ greatly in their specifics, the corresponding bare bones versions share three common features:

- The sensors observe the underlying environment and exchange their observations.
- The sensors cooperate to establish a common frame of reference so that the observations at different sensors are “comprehensible” to one another.
- The sensors then use the “known” relationship between the phenomenon of interest and the recorded data (typically, a statistical relationship) to make deductions about the phenomenon.

For a concrete illustration, consider the problem of localizing a source of sound based on the times at which it is heard at different sensors. For the times at which the source is heard at two different sensors to bear any meaningful relationship to one another, the underlying clocks used to measure these times must be well synchronized. Thus, a common notion of time across sensors is the frame of reference that the network needs to agree on. Having done this, the sensors can then use the times at which the source was heard and the geometric propagation constraints to estimate the location of the source.

In this dissertation, we make the case that a *minimalist* approach to the twin

problems of agreeing on a common frame of reference and making inferences from the observations is extremely attractive because it uses network resources efficiently and also guarantees robust performance. We now explain why.

Establishing a common frame of reference is only a prerequisite to a larger goal (such as making deductions about the phenomenon of interest) and not an end by itself. Therefore, network resources that are dedicated for this purpose are considered an overhead. For example, to synchronize the network, nodes could explicitly exchange messages that contain timing information. However, these messages fritter away scarce resources such as the network bandwidth and the power available at the sensors. Therefore, a minimalist approach that avoids such explicit message exchanges is desirable and we design an algorithm to do this.

Minimalism is also useful in making inferences from observations since it guarantees robustness. The process of making inferences typically occurs in two stages: first, we abstract a model of the sensing process from our understanding of the world and then design algorithms for making deductions, assuming that the observations are generated according to the abstracted model. However, when the recorded data deviates from the model, because of unforeseen and unavoidable variations in the sensing process, the algorithm is prone to fail. Therefore, an algorithm that makes fewer assumptions is more likely to work in a wider range of environments.

However, we do pay a price for such increased robustness – straightforward implementations of algorithms with minimal assumptions typically lead to unacceptable computational complexities. The challenge is to dodge such bottlenecks while maintaining robustness. We present two problems – that of localizing multiple events from the times at which they are heard at different sensors and estimating an unknown source waveform recorded through unknown dispersive channels – where *parallelizing* the available evidence is the key to lowering the computational complexity while making inferences with minimal models.

We now provide an overview of the three examples we have mentioned to argue the case for minimalism in sensor networks and summarize our contributions.

1.1 Implicit network timing synchronization

We consider a sensor network that operates in a Time Division Multiplexed (TDM) fashion and ask the following question: can the sensor nodes maintain synchrony by leveraging the timing information present in the existing communication in the network? There are two key steps in network timing synchronization. At startup, the clocks at different nodes have times that are completely random. A coarse level of synchrony can be established using explicit synchronization mes-

sages. The overhead involved with such explicit messaging is tolerable since this is only a one-time procedure. Furthermore, we can use these messages to estimate the propagation delays between nodes. Having established synchrony, we face the more critical problem of *maintaining* it. The clocks at different nodes run at different frequencies due to manufacturing and temperature variations. As a result, the clocks drift apart, tending to destroy the established synchrony. This forces us to periodically compensate for such drifts and drive the network back towards synchrony. The naïve approach for such compensations would be to rerun the coarse synchronization procedure “often”. However, this leads to an unreasonably large message overhead. We solve the problem of maintaining synchrony with minimal overhead as follows: first, we explain how a node in a TDM network receives an *implicit timestamp* every time it receives a packet from its neighbor. Then, we show how the nodes can use the implicit timestamps to adjust their clock phases and frequencies to achieve network-wide synchrony.

Nodes in a TDM network know when to expect packets from their neighbors. Therefore, when a node receives a packet from its neighbor (as part of the existing communication), it can compare the *actual* time of reception with the *expected* time of reception. The difference between these quantities is precisely the phase error in the clocks of these two nodes. The node that receives the packet uses this estimate of the phase error to adjust its phase and frequency at two different

timescales in order to drive the network to synchrony. The rules for phase and frequency adjustment are as follows:

- Each time a node receives a packet, it adjusts its phase so that the error with the transmitter reduces by a factor of $1 - \beta$ ($0 < \beta < 1$).
- All nodes adjust their frequencies only once per *round*, consisting of many slots. A node makes a frequency adjustment based on the average error it observes with its neighbors over the entire round. If the average error seen by the node is positive, we show that its frequency is larger than the mean frequency across nodes. In such cases, the node reduces its frequency by μ to drive it closer to the mean. Similarly, if the average error is negative, the node increases its frequency by μ .

Contributions: We make the following contributions towards solving this problem:

1. We show that a phase-only adjustment algorithm (nodes never adjust their frequencies), which is easier to implement, suffices for small networks. However, for bigger networks, we use a linear programming formulation to show that the algorithm can lead to large errors between neighboring nodes due to uncompensated frequency errors.

2. We show that the phase-frequency adjustment algorithm described above is effective in synchronizing the clocks in both phase and frequency. We do this in two stages: first, we rigorously prove that the algorithm converges for a fictitious *averaged* system, which smooths out the randomness in the communication pattern that is typical of TDM networks. Then, we use the Law of Large Numbers (LLN) to relate the true system to the averaged system. Finally, we show via extensive simulations that the algorithm achieves frequency and phase synchrony for both directional and omnidirectional networks.

We now describe two scenarios that illustrate the value of parallelization in making inferences with minimal models.

1.2 Localizing multiple events from Times of Arrival

We investigate the problem of localizing multiple events from their Times of Arrival (ToAs) at different sensors. This problem is representative of many domains – it could involve localizing animals from their sounds in an environmental

monitoring context or localizing gunfire in a defence/homeland security scenario. Our model is minimalist and hence, very general: each sensor has a list of ToAs and the association between the events and the ToAs is not known *a priori*. In contrast to much of the traditional literature, we allow the events to be closely spaced in time. This leads us to the central problem: when the events occur in quick succession, due to the varying propagation delays, the order in which a sensor hears the events need not be the same as the order in which the events occur. Figure 1.1 shows a simple example of how this can happen. In such cases, the naïve strategy of sorting the ToAs in ascending order at each sensor, associating the i th ToA at each sensor to the i th event and then localizing the events one at a time will fail.

An alternative strategy of considering all possible combinations of ToAs and retaining only those that are “good” incurs a complexity that grows exponentially with the number of sensors (if E denotes the number of events and N , the number of sensors, we need to look through E^N combinations). This approach suffers from two additional complications: (a) there is no guarantee that only E among the E^N combinations will be declared to be “good”. (b) Realistically, each sensor misses events and also observes outlier ToAs, thereby causing the number of ToAs to vary across sensors. Extending the naïve strategy of looking through all combinations

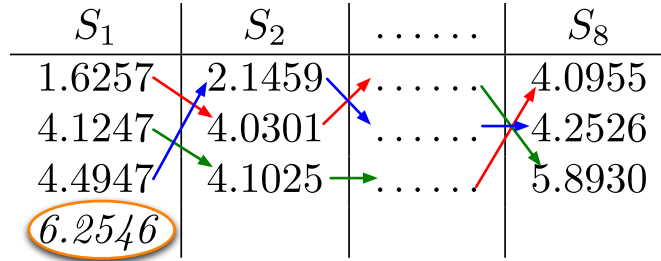


Figure 1.1: Three events, that we call “Blue (B)”, “Green (G)” and “Red (R)”, happen close to one another in time and produce ToAs at 8 sensors. The ToAs at each sensor are sorted in ascending order. The red arrows connect the ToAs produced by the red event and so on. Note that the events need not arrive at the sensors in the same order: for example, the order of ToAs at sensor 1 is RGB, whereas it is BRG at sensor 2. We ask: (a) under what conditions can we group the ToAs appropriately – draw the arrows that connect ToAs produced by the same event – and localize the events and (b) how do we do this in a robust fashion with low complexity? Note that we can have outlier ToAs at some sensors, such as the ToA in the orange bubble at sensor 1, which must be discarded. Additionally, some sensors might miss an event and not have a corresponding ToA, but this is not shown in the figure.

to such a scenario is unclear. In this dissertation, we pose and answer the following questions:

- Under what conditions, can we guarantee that all the events will be localized perfectly? Can we construct an example where the ToAs produced by the different events “interfere” with one another and prevent perfect localization?
- How do we estimate the number of the events, their locations and the times at which they occur from the ToAs at different sensors?

Contributions: We answer these questions as follows:

- We show that, under ideal conditions (no ToA measurement noise, misses or outliers), nine sensors suffice to localize two events, provided that the sensors do not lie on one branch of a hyperbola. We also construct an example where six sensors cannot localize two events perfectly, thereby demonstrating a fundamental ambiguity.
- We propose a robust, low-complexity algorithm to localize events from their ToAs in the presence of misses and outliers. The algorithm has three stages. The key step in this algorithm is to discretize the times at which events occur. We think of each of these times as one among many *parallel hypotheses*. For a hypothesized event time t and an observed ToA τ , the event must lie on a circle of radius $c(\tau - t)$ centered at the sensor (c is the speed of propagation). Locating the events by intersecting circles drawn at different sensors is very easy, since the point of intersection of two circles can be specified in closed-form. However, some of these intersection points are “phantoms” (arising from the intersection of circles corresponding to ToAs for different events) and some are “duplicates” (produced by intersecting circles corresponding to ToAs due to the same event, but by considering different sensor pairs). In the second stage, we refine these estimates using

measurements at all the sensors and use a clustering procedure to merge duplicates. Having narrowed down the space of candidate events considerably, we cast the problem of associating these candidates to the observed ToAs as a modification of the matching problem on a graph. We formulate this as a binary integer program. Finally, we show via simulations that a linear programming relaxation of the integer program is efficient in solving the association problem and localizing the events.

1.3 Collaborative Estimation in Dispersive Environments

We investigate the fundamental problem of making inferences about events, regarding which we have limited prior knowledge. The specific formulation we consider is as follows: a noisy version of an unknown signal is observed at many sensors, after being distorted by a dispersive environment, which is also largely unknown. The only information we are given regarding the dispersive channels is a coarse estimate of their *delay spread* – the temporal span of these channels. We wish to reconstruct the signal by pooling the observations at different sensors, averaging out the noise and undoing the effects of the dispersive channels in the

process. We also wish to understand whether the minimalism in modeling leads to fundamental ambiguities, such as multiple signals explaining the recorded data.

Figure 1.2 explains the problem setup pictorially.

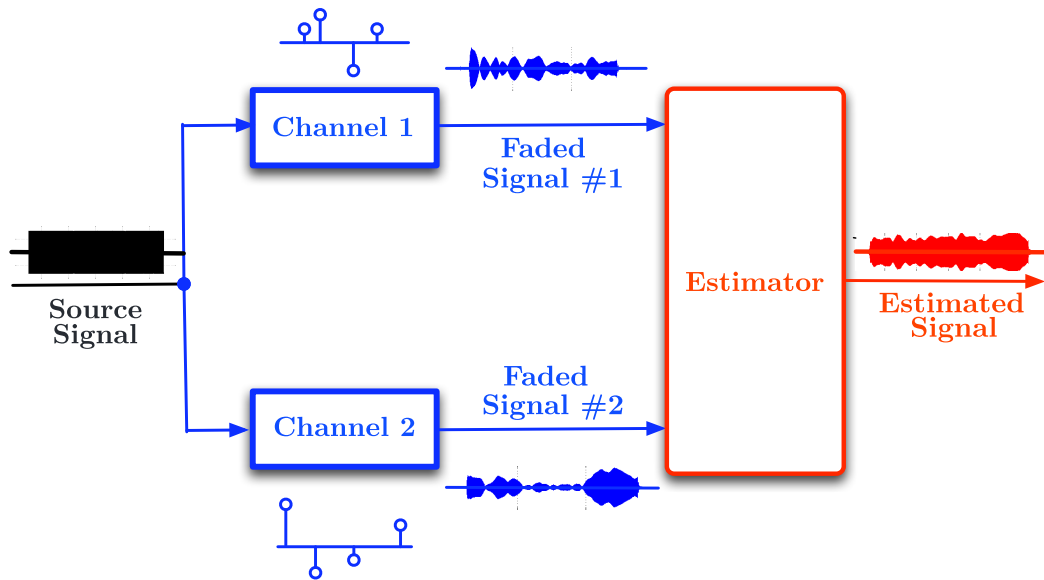


Figure 1.2: A source is recorded at multiple sensors through dispersive channels. We design an estimation algorithm to reconstruct the source from the faded signals.

Traditional time-domain approaches to this problem are useful in establishing feasibility results. However, designing algorithms based on such methods results in unacceptable computational complexity, particularly with long data records. We overcome this hurdle by switching to the frequency domain, thereby parallelizing the problem. The channel seen by each sensor can be approximated by a

constant within a small enough frequency band. The size of such a band is typically called the *coherence bandwidth* of the channel and it is roughly the inverse of the delay spread. Since the channels have a simple structure over a small range of frequencies, the unknown source signal can be estimated easily within each band. Specifically, we use a Singular Value Decomposition (SVD) of the recorded signals *in each band*. However, since the signal as well the channels are unknown, the signal estimate in each band is accurate only up to a complex scale factor. We estimate the scale factor in two steps: first, we refine the channel model and approximate the channels as quadratic functions of frequency within a band. We then propose an iterative procedure to estimate the signal and quadratic channels over small bands, bootstrapping with the SVD estimate of the signal. Next, we choose the bands to have considerable overlap and use the continuity of the channel in the frequency domain to estimate the scale factors. We use the scale factors to “stitch” together the signal contributions from different bands.

Contributions:

1. Using simulations and experiments with an acoustic sensor network, we show that the proposed algorithm is effective in accurately reconstructing signals whose bandwidths are within 10 – 20 times the coherence bandwidth of the channel.
2. For larger signal bandwidths, we find that fundamental ambiguities arise in reconstructing the signal. We propose a modified stitching algorithm that estimates the scale factors using all the available constraints and show that multiple signals, which differ widely, can explain the recorded data.

1.4 Outline

We explain the algorithms for implicit timing synchronization, localizing multiple events from their ToAs and collaborative estimation in dispersive environments in Chapters 2,3 and 4 respectively. We present our conclusions in Chapter 5.

Chapter 2

Distributed Implicit Network Timing Synchronization

In this chapter, we consider the problem of maintaining timing synchrony in a time division multiplexed (TDM) sensor network by exploiting the timing information present in the ongoing communication. The broad ideas behind the algorithm we propose to solve the problem are as follows:

- A node in a TDM network knows when to expect packets from its neighbors.

Therefore, by comparing the time at which it *actually* receives packets to the times at which *expects* to receive them, the node can estimate the error with its neighbors.

- The node adjusts its phase (and possibly, its frequency) so as to reduce the clock error with its neighbor.

- Such adjustments are coupled through the network transmission schedule: the phase and frequency changes made by the node impact the times at which it transmits, and hence the adjustments made by nodes who receive these transmissions.

While we present these ideas in the context of a sensor network (where synchrony is needed for sleep scheduling or localization based on ToAs), they only rely the fact that the transmissions in the network are time division multiplexed. Thus, they can be applied to any wireless network that operates on a TDM schedule. For example, the use of TDM is extremely attractive for emerging millimeter (mm) wave networks [41], since CSMA-based medium access is infeasible due to deafness induced by highly directional links. Similarly, it can also be used in existing WiFi-style communication networks based on TDM. In such scenarios, the proposed algorithm would be effective in establishing time slotting mechanisms with minimal guard periods, on top of which TDM-based medium access control (MAC) protocols can be built.

Our goal in this dissertation is to obtain fundamental insight into the feasibility of implicit timing synchronization. Therefore, we do not model the physical and medium access layer of the underlying network in detail. Our simulations are therefore based on the following abstractions: (a) the set of nodes that can

successfully receive a packet sent by a given node, (b) the sets of *matchings*, or links that can be simultaneously active in a given TDM slot. The TDM schedules we consider for testing our timing synchronization method are chosen randomly (as described in Section 3.8) from a set of maximal matchings.

Map of this chapter: We begin by placing this work in the context of the vast literature on timing synchronization in Section 2.1. Section 2.2 describes the system model and introduces the concept of an averaged system that smooths out the randomness in communication patterns of a TDM network. Section 2.3 describes the phase-only adjustment algorithm, where the nodes adjust only their phases and never adjust their frequencies in response to implicit timestamps they receive from their neighbors. We quantify the worst-case performance of this algorithm in two stages: (a) Using a linear programming formulation, we identify the set of frequencies that maximize the pairwise-error for the averaged system. (b) We then show that the errors in the averaged system are a lower bound to the errors in the actual system. Thus, the same set of frequencies are also “bad” for the actual system and this lets us show that the error between neighbors can grow with the size of the network. In Section 2.4, we solve this problem in the averaged system by adjusting the frequencies in addition to the phases, but with the frequency adjustments occurring on a slower timescale. Over a round, consisting of many slots, nodes only adjust their phases. At the end of the round, nodes

estimate whether their clock frequency is larger or smaller than the network-wide mean and adjust their frequencies accordingly. We rigorously show that the proposed algorithm achieves network-wide synchrony. In Section 2.5, we translate these insights to the actual system and use the Law of Large Numbers (LLN), to propose the following rule to adjust the frequencies: (1) a node averages the phase errors it observes with all its neighbors over a round of slots. (2) If, in spite of the all the phase adjustments it made over the round, the node finds this average error to be positive, it concludes that its frequency is larger than the network-wide average. In response, it reduces its frequency by a “small” amount μ . Simulations in Section 3.8 show that the proposed algorithm achieves phase and frequency synchrony over a wide range of scenarios.

2.1 Related Work

We believe that this is the first work to introduce the concept of *implicit* timestamps based on ongoing communication, to provide fundamental theoretical insight into attainable performance (via the averaged model), and to provide a completely decentralized algorithm for *both phase and frequency* adjustment. The well-known firefly-inspired algorithm [27] can be made to work with implicit timestamps, but it is only designed for phase synchrony, and does not handle either

propagation delays or oscillator skew. The Reachback Firefly Algorithm (RFA) in [50] adapts the firefly algorithm to account for propagation delays in its phase adaptation, but it does not handle oscillator skew and therefore requires periodic resynchronization.

A number of algorithms ([42], [34], [43]) that have been proposed for distributed timing synchronization can be broadly classified as “consensus algorithms” [28]. These algorithms typically proceed in two stages: (1) estimate relative offset and skew with each neighbor and (2) use neighbor’s estimates of offset and skew (typically with respect to a reference node) along with pairwise estimates to arrive at an estimate of one’s offset and skew. However, these algorithms are not amenable to an *implicit* implementation because all of them require *explicit* exchange of additional information to achieve consensus on *both frequency and phase*. The use of this information is critical for decoupling the problems of phase and frequency adjustments, in order to be able to run separate consensus algorithms for each of them. For example, [42] requires each node to broadcast the average error (both in rate and time) that it observes with its neighbors, while [34] and [43] require the nodes to broadcast their clock rates. The second order consensus algorithms proposed by [31] and [30] are not directly applicable because they also involve exchange of more information than timestamps (implicit or explicit); in order to use these algorithms for timing synchronization,

the nodes would also need to exchange their clock rates. Reference [16] analyzes the algorithm proposed in [42] and shows that, for random connected networks, the error variance does not scale up with the size of network, thereby demonstrating the feasibility of using consensus-style algorithms for synchronization in large networks. In contrast, for our implicit model, phase-only adjustments lead to pairwise phase errors between neighbors which can increase with network size. To resolve this without requiring explicit messages, we need to introduce memory in our frequency adjustment rule, and run it on a slower time scale than the phase adjustment rule. While [14] also considers the use of memory, it does so to speed up convergence of *phase* synchronization, and does not consider clock skew.

Reference [3] characterizes the effect of asymmetric communication on consensus algorithms and identifies a scenario where the variance of the estimate can *increase* with more measurements because of lack of bidirectional communication. These are consistent with our own observations that convergence requires a "rich enough" communication pattern. The survey [38] provides a broad perspective on the field of distributed timing synchronization (see also the excellent discussion in [43]), while [12] identifies the fundamental limits of synchronizing "affine" clocks in the presence of delays, which includes but is not restricted to propagation delay. For distributed synchronization specifically applied to WiFi, see [18]. In contrast to the distributed algorithms discussed so far, references [11], [13] and [25] adopt

a more centralized approach to timing synchronization. Reference [13] builds a rooted spanning tree on the network in the Level Discovery Phase and perform hop-by-hop synchronization, but it does not address clock skews, and therefore requires periodic resynchronization. Reference [25] uses linear regression to correct skew, with a dynamically elected root node flooding the network with its timestamp. As pointed out in [43], however, timing synchronization based on information flowing from a “distant” node, as in [25], performs poorly in certain topologies (specifically the ring topology). Reference [11] uses reference broadcasts from a transmitter to synchronize a cluster of receivers; while this is designed for single-hop time synchronization, nodes that are common to two clusters can “convert” between the timestamps of two clusters.

2.2 System Model

We consider a network of N sensor nodes, labeled $\mathcal{N}_1, \mathcal{N}_2, \dots, \mathcal{N}_N$, each equipped with a clock that runs at a *nominal frequency* of f_{nom} Hz. The *actual* frequency of the clock at \mathcal{N}_i , denoted by f_i , differs slightly from the nominal rate of the clock f_{nom} , and is written as $f_i = f_{nom}(1 + \rho_i)$ where ρ_i is called the skew, or drift, of the clock. The magnitude of the skew is assumed to be less than ρ_{max} ,

which is typically on the order of 10-100 parts per million (ppm); for example, $\rho = 20 \times 10^{-6}$ corresponds to a skew of 20 ppm. Differences in the skews across nodes are caused by manufacturing tolerances. The variation of the skew at a given node with respect to time is typically very slow, hence we approximate ρ_i as a constant that takes values in $[-\rho_{max}, \rho_{max}]$.

The network employs a Time Division Multiplexed (TDM) schedule in which transmissions begin at integer multiples of a *slot time*, denoted by T_{slot} , according to the *transmitter's clock*. It is convenient to describe the system from the point of view of an external observer who possesses a clock that runs at exactly the nominal rate f_{nom} . When the time on the external observer's clock is t , let $\varphi_i(t)$ denote the measure of time at \mathcal{N}_i . We refer to $\varphi_i(t)$ as the clock *phase* of \mathcal{N}_i . Suppose that \mathcal{N}_j transmits to its neighbor \mathcal{N}_i in slot s . Therefore, \mathcal{N}_j starts transmitting when $\varphi_j(t) = sT_{slot}$. We assume that nodes have estimates of the propagation delays, including processing times, and subtract them out from the packet reception times. Thus, \mathcal{N}_i begins receiving this packet at $\varphi_i(t)$, and can implicitly conclude that its clock is behind \mathcal{N}_j 's clock by $\varphi_j(t) - \varphi_i(t) = sT_{slot} - \varphi_i(t)$. \mathcal{N}_i can now reduce its phase error with respect to \mathcal{N}_j using the following linear phase adjustment.

Linear Phase Adjustment: Letting t^- and t^+ denote the times on the external observer's clock just before and after the phase jump, we have,

$$\varphi_i(t^+) = \varphi_i(t^-) + \beta[\varphi_j(t^-) - \varphi_i(t^-)] \quad (2.1)$$

where $0 < \beta < 1$ is a design parameter.

Quasi-Synchronous Approximation: To simplify analysis, we assume that the phase and frequency adjustments are made at integer multiples of the slot times based on the external observer's clock, rather than on the receiver's clock (which keeps changing as we make phase and frequency adjustments). This approximation causes a second order error (because of measuring the phase offset between nodes at a time that is slightly offset from the true time) that is negligible compared to the phase offsets themselves. In simulations of the original system, we do not make this approximation, and verify that the quasi-synchronous approximation is indeed valid. We define $\varphi_i[s] = \varphi_i(sT_{slot}^-)$ as the phase of the i th node just before the right edge of the s th slot. These phases are adjusted as in (2.1) at the slot boundaries, and then evolve linearly according to the relative frequency offsets of the nodes across a slot.

Modeling skews: The nodes change their frequencies only when they receive a packet and hence, skews can change explicitly only at slot boundaries. Let $f_i[s]$ be the raw frequency of the i th node over the s th slot. Since we will primarily be

dealing with the clock phases φ_i , it is convenient to normalize them and simplify the notation. To do this, we introduce an intermediate “raw phase” for node i , denoted by $\theta_i(t)$, which evolves across slot s as,

$$\theta_i((s+1)T_{slot}^-) = \theta_i(sT_{slot}^+) + f_i[s] \times T_{slot} \quad (2.2)$$

We define the clock phase $\varphi_i(t)$ (as used above) to be $\varphi_i(t) = \theta_i(t)/(f_{nom}T_{slot})$. Let $F_i[s] = \frac{f_i[s]}{f_{nom}}$ denote the corresponding normalized frequency. Define the *normalized mean frequency*

$$\bar{\psi}[s] = \frac{1}{N} \sum_{i=1}^N F_i[s]$$

This is the common part of the clock phase drift over the s th slot, and hence does not impact phase differences across nodes. The latter are actually driven by the normalized *excess frequencies*

$$\delta_i[s] = \frac{f_i}{f_{nom}} - \bar{\psi}[s], \quad i = 1, \dots, N$$

where $\delta_i[s]$ is the amount by which the phase of the i th node drifts relative to the average.

Discrete time dynamics: When node i receives a packet from its neighbor node j , its phase evolves as follows:

$$\varphi_i[s+1] = \varphi_i[s] + \beta[\varphi_j[s] - \varphi_i[s]] + \bar{\psi}[s] + \delta_i[s] \quad (2.3)$$

According to the quasi-synchronous approximation, these phase updates occur synchronously at all nodes receiving packets, and can be conveniently expressed in vector form. Defining $\boldsymbol{\varphi}[s] = (\varphi_1[s], \varphi_2[s], \dots, \varphi_N[s])$ and $\boldsymbol{\delta}[s] = (\delta_1[s], \delta_2[s], \dots, \delta_N[s])$, we have

$$\boldsymbol{\varphi}[s+1] = G_s \boldsymbol{\varphi}[s] + \boldsymbol{\delta}[s] + \bar{\psi}[s] \mathbf{1} \quad (2.4)$$

where $\mathbf{1}$ denotes the vector with all components equal to 1, and G_s is a matrix defined as follows: (1) If \mathcal{N}_i receives a packet from \mathcal{N}_j in slot s , $G_s(i, j) = \beta$, $G_s(i, i) = 1 - \beta$ and $G_s(i, j') = 0 \forall j' \neq i, j$ and (2) if \mathcal{N}_i does not receive a packet from any node in slot s , $G_s(i, i) = 1$ and $G_s(i, j) = 0 \forall j \neq i$. We refer to G_s as the *system matrix* at slot s . Note that G_s is a stochastic matrix (each row has nonnegative entries summing to one), so that $G_s \mathbf{1} = \mathbf{1}$ (i.e., $\mathbf{1}$ is an eigenvector of G_s with eigenvalue 1).

Frequency Adjustment: We can see from (2.4) that excess frequencies lead to accumulation of phase errors between a node and its neighbors. Our frequency adjustment algorithm relies on this observation, and is loosely stated as follows (details provided later). A node concludes that its frequency is larger than the prevailing average in the network if its phase is ahead of its neighbors' "on average" despite the phase adjustments it makes, where the averages are of phase errors

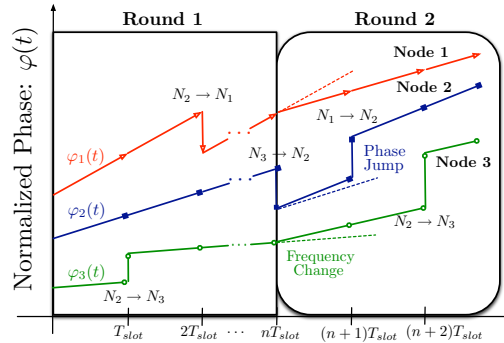


Figure 2.1: Nodes make *phase jumps* each time they receive a packet. However, they change their frequencies (slope of the lines) only at the end of a *round* consisting of “many” slots.

measured over all neighbors over many slots. A representative diagram of the phase and frequency adjustments is sketched in Figure 2.1.

Averaged System: Each time \mathcal{N}_i receives a packet, it adjusts its clock phase based on the observed error with the transmitter. Since the node that transmits a packet to \mathcal{N}_i changes over slots, it introduces “fine details” into the dynamics of phase offset evolution that must be averaged out when estimating and compensating for skews. We therefore introduce the following fictitious *averaged system*: in each slot of the averaged system, *every* node updates its phase based on a weighted average of the phases of *all* its neighbors. The weights are derived from the communication pattern in the actual system and do not change with time. For example, for a random communication pattern, we model the matrices $\{G_s\}_{s=0}^{\infty}$, and thereby the set of links active concurrently, as being chosen indepen-

dently and identically from a set $\mathcal{S}_{matching} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$ with probabilities $\{p_1, p_2, \dots, p_M\}$ respectively. We denote the averaged system matrix by $\overline{\mathcal{G}}$ and define it to be $\overline{\mathcal{G}} = \sum_{i=1}^M p_i \mathcal{G}_i$. The phase evolution in the averaged system is therefore exactly as in (2.4), except that G_s is replaced by $\overline{\mathcal{G}}$.

$$\varphi[s + 1] = \overline{\mathcal{G}}\varphi[s] + \delta[s] + \overline{\psi}[s]\mathbf{1} \quad (2.5)$$

For simplicity, we assume throughout this paper that the averaged system matrix $\overline{\mathcal{G}}$ is symmetric. This corresponds to the links in the network having the same probability of being active in either direction.

Network Start-up: While our focus here is to understand the limits of implicit synchronization *maintenance*, our simulations do include a gateway-led start-up scheme for coarse initial synchronization (no attempt is made to optimize this). A gateway node broadcasts its time and other nodes in its neighborhood “set” their clocks to this value. Each of these nodes then broadcasts its time enabling nodes that are two hops from the gateway to set their clocks. This broadcast is made at a random instant in an interval $[\tau_{min}, \tau_{max}]$ after its clock was set. If \mathcal{N}_i 's clock is already set when it hears \mathcal{N}_j 's broadcast, it simply adjusts its clock to the mean value. Specifically, if \mathcal{N}_j 's time when it makes the broadcast is φ_j and at this point, \mathcal{N}_i 's time is φ_i , \mathcal{N}_i adjusts its clock to $(\varphi_j + \varphi_i)/2$. Each node makes one such broadcast and the network can be coarsely synchronized in this

fashion in a time period that scales linearly with the number of nodes. We assume throughout the paper that nodes have good estimates of the propagation delay to their neighbors. This can be done by exchanging multiple packets with neighbors at the time of network startup and estimating the propagation delay in a manner similar to [12]. Any residual errors in estimating the propagation delay are considered to be subsumed in the phase jitter in our model.

2.3 Phase-only Adjustments

In this section, we analyze the performance of a system in which the nodes adjust only their clock phases, and never adjust their frequencies. Phase-only updates are of interest in their own right (due to their simplicity), while also being the first step in the design of phase-frequency updates. The uncompensated skews in such a system tend to drive the network towards asynchrony, while the recurrent phase compensations tend to pull the phases closer to each other. Our metric for quantifying which of these two forces is dominant is the worst-case pairwise clock phase error between neighbors, which is precisely the overhead required to maintain a TDM schedule. We first analyze this metric for the averaged system, and then observe that this provides a lower bound for the original system.

2.3.1 Averaged System

Starting with (2.5), note that the mean and excess frequencies, $\bar{\psi}[s]$ and $\boldsymbol{\delta}[s]$, do not change with s when we only adjust phases. We can therefore drop the time index s and denote these quantities by $\bar{\psi}$ and $\boldsymbol{\delta}$, respectively. Iterating, the phases in slot s can be expressed in terms of the initial conditions as

$$\boldsymbol{\varphi}[s] = \bar{\mathcal{G}}^s \boldsymbol{\varphi}[0] + s\bar{\psi}\mathbf{1} + (\mathbb{I} + \bar{\mathcal{G}} + \bar{\mathcal{G}}^2 + \dots + \bar{\mathcal{G}}^{s-1})\boldsymbol{\delta} \quad s \geq 1 \quad (2.6)$$

As with the frequencies, it is useful to decompose the phases into mean and excess phases, as follows:

$$\boldsymbol{\varphi}[s] = \bar{\varphi}[s]\mathbf{1} + \boldsymbol{\varphi}_{ex}[s]$$

where $\bar{\varphi}[s] = \mathbf{1}^T \boldsymbol{\varphi}[s]/N = \frac{1}{N} \sum_{i=1}^N \varphi_i(s)$ is the average phase and $\boldsymbol{\varphi}_{ex}[s]$ represents the excess phase. Since the mean component $\bar{\varphi}[s]\mathbf{1}$ is identical across nodes, we only need to track the evolution of the excess phases $\boldsymbol{\varphi}_{ex}[s]$ to quantify the phase error between neighbors. We use an eigendecomposition of $\bar{\mathcal{G}}$ in (2.6) to derive an expression for $\boldsymbol{\varphi}_{ex}[s]$ in “steady-state” (large values of s). Before doing this, we review results on the eigenstructure of $\bar{\mathcal{G}}$ briefly.

The eigendecomposition of $\bar{\mathcal{G}}$ is given by

$$\bar{\mathcal{G}} = \sum_{l=1}^N \lambda_l \mathbf{v}_l \mathbf{v}_l^T \quad (2.7)$$

where (a) $\bar{\mathcal{G}}\mathbf{v}_l = \lambda_l\mathbf{v}_l$, (b) the eigenvalues are arranged in descending order of magnitude $|\lambda_1| \geq |\lambda_2| \dots \geq |\lambda_N|$, (c) the eigenvectors \mathbf{v}_l have unit norm, or $\mathbf{v}_l^T\mathbf{v}_l = 1$ and (d) eigenvectors corresponding to different eigenvalues are orthogonal, or $\mathbf{v}_i^T\mathbf{v}_j = 0$ $i \neq j$, because $\bar{\mathcal{G}}$ is symmetric. Because $\bar{\mathcal{G}}$ is stochastic, its largest eigenvalue is one, with $\mathbf{v}_1 = \mathbf{1}/\sqrt{N}$ being the corresponding unit norm eigenvector. For convergence of consensus style algorithms, this largest eigenvalue must not be repeated. The condition for this is that the graph corresponding to $\bar{\mathcal{G}}$ must be connected [32] (clearly needed for network-wide convergence). In this case, we can approximate $\bar{\mathcal{G}}^s$ by the dominant term in its spectral decomposition for large values of s as,

$$\bar{\mathcal{G}}^s \approx \lambda_1^s \mathbf{v}_1 \mathbf{v}_1^T = \frac{1}{N} \mathbf{1} \mathbf{1}^T \quad (2.8)$$

for reasonably large s , so that the first term in (2.6) tends to $\mathbf{1}^T \boldsymbol{\varphi}[0]/N$, the mean phase at start-up. The second term in (2.6) also contributes only to the mean phase. For the third term, the eigenmode corresponding to eigenvalue one has no response, since the excess frequency $\boldsymbol{\delta}$ has zero mean. Thus, the evolution of the excess phase (which is what drives our phase-only algorithm) can be obtained by excising the first eigenmode from $\bar{\mathcal{G}}$ to get the matrix

$$\bar{\mathcal{G}}_{ex} = \bar{\mathcal{G}} - \frac{\mathbf{1} \mathbf{1}^T}{N} = \sum_{l=2}^N \lambda_l \mathbf{v}_l \mathbf{v}_l^T \quad (2.9)$$

which contains only eigenmodes with eigenvalues with magnitude strictly smaller than one (the dynamics induced by this matrix are dominated by the magnitude of the second largest eigenvalue, $|\lambda_2|$). For large s , the excess phase therefore exhibits the following behavior:

$$\begin{aligned} \varphi_{ex}[s] &= \left(\mathbb{I} + \overline{\mathcal{G}}_{ex} + \overline{\mathcal{G}}_{ex}^2 + \dots + \overline{\mathcal{G}}_{ex}^{s-1} \right) \boldsymbol{\delta} \\ &\rightarrow \left(\mathbb{I} - \overline{\mathcal{G}}_{ex} \right)^{-1} \boldsymbol{\delta} \quad , \quad s \rightarrow \infty \end{aligned} \tag{2.10}$$

We provide a formal proof of this statement in Appendix A.1. Thus, the excess phases exhibit an irreducible error floor that depends on the skews. To quantify the error floor, we consider the largest pairwise phase error between neighboring nodes and maximize this over the set of possible skews. The maximization procedure consists of two steps:

1. First, we show that the pairwise error between a *fixed* pair of neighbors can be maximized by solving a linear programming (LP) problem, where the constraints are determined by maximum allowable skew ρ_{max} .
2. We maximize over the solutions of the LPs corresponding to every pair of neighbors to obtain the worst-case pairwise phase errors.

We also show that the worst-case phase error between neighboring nodes is produced by (roughly) half the node clocks running at the largest frequency and the other half running at the smallest frequency for *any* network topology. This proof

rests on the fact that the various LPs in Step 2 have the same feasible set for their variables and that the optimizing solution occurs at an extreme point of the feasible set. We provide the details of the formulation as well as a characterization of the extreme points in Appendix A.2.

2.3.2 Actual System

The irreducible error in the averaged system (Equation 2.10) can be used to obtain a lower bound on the errors between neighbors in the actual system. We now sketch the central ideas used to derive this bound and provide the details in Appendix A.3.

Iterating (2.4), we obtain that the phase of the actual system evolves as:

$$\begin{aligned} \boldsymbol{\varphi}[s] = & G_{s-1}G_{s-2}\dots G_0\boldsymbol{\varphi}[0] + s\bar{\boldsymbol{\psi}}\mathbf{1} \\ & + \left(\mathbb{I} + \sum_{k=1}^{s-1} G_{s-1}G_{s-2}\dots G_{s-k}\right)\boldsymbol{\delta} \quad s \geq 2 \end{aligned} \quad (2.11)$$

The second term only contributes to the mean phase. Under a suitable connectedness condition [32] which is needed for consensus, we can show that the same is true for the first term as well as s gets large. The result in [32] states that

$$\lim_{s \rightarrow \infty} G_s G_{s-1} \dots G_0 = \mathbf{1}\boldsymbol{\gamma}^T \quad (2.12)$$

where the elements of $\boldsymbol{\gamma}$ are nonnegative and $\boldsymbol{\gamma}^T \mathbf{1} = 1$. We can therefore throw out the first and second terms in (2.11) when considering the evolution of the

excess phases, and obtain

$$\boldsymbol{\varphi}_{ex}[s] \approx \left(\mathbb{I} + \sum_{k=1}^{s-1} G_{s-1} G_{s-2} \dots G_{s-k} \right) \boldsymbol{\delta} \quad (2.13)$$

Let us now average the evolution over many realizations of the communication schedule. For each schedule, the matrices G_n are chosen in i.i.d. fashion from the set $\mathcal{S}_{matching} = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$ with probabilities $\{p_1, p_2, \dots, p_M\}$ and the average across this choices is denoted by $\overline{\mathcal{G}}$, the matrix for our averaged system (the corresponding excised matrix is $\overline{\mathcal{G}}_{ex}$). Let $\boldsymbol{\varphi}_{ex}^{(i)}[s]$ denote the excess phases in slot s for the i th realization of the communication schedule ($i = 1, 2, \dots, M$). We define the averaged version of the excess phases as

$$\overline{\boldsymbol{\varphi}}_{ex}[s] = \frac{1}{M} \sum_{i=1}^M \boldsymbol{\varphi}_{ex}^{(i)}[s].$$

When the number of schedule realizations M is large, we can use the LLN to show the averaged trajectory of excess phases satisfies

$$\overline{\boldsymbol{\varphi}}_{ex}[s] \approx \mathbb{E} \left[\left(\mathbb{I} + \sum_{k=1}^{s-1} G_{s-1} G_{s-2} \dots G_{s-k} \right) \boldsymbol{\delta} \right] = \left(\mathbb{I} + \sum_{k=1}^{s-1} \overline{\mathcal{G}}^k \right) \boldsymbol{\delta}.$$

As before, we can excise the first eigenmode of $\overline{\mathcal{G}}$ to show that the averaged trajectory $\overline{\boldsymbol{\varphi}}_{ex}[s]$ satisfies

$$\overline{\boldsymbol{\varphi}}_{ex}[s] \approx \left(\mathbb{I} + \sum_{k=1}^{s-1} \overline{\mathcal{G}}_{ex}^k \right) \boldsymbol{\delta} \quad (2.14)$$

This equation reveals the precise relationship between the averaged and the actual systems: the *averaged trajectory* of the excess phases in the actual system (2.14) is

the same as the excess phases in the averaged system (see (2.10)), for large values of the slot index s .

We now use Jensen’s inequality to bound the phase errors in the actual system using the corresponding errors in the averaged system. To do this, we note that the maximum value of the pairwise difference between neighbors’ phases in the averaged system (or equivalently, excess phases) can be expressed as $\|\mathbf{C}\boldsymbol{\varphi}_{ex}[s]\|_\infty$, where matrix multiplication by \mathbf{C} corresponds to all possible pairwise differences across neighbors and $\|\mathbf{x}\|_\infty$ denotes the infinity norm, or maximum element, of a vector \mathbf{x} . The corresponding worst-pairwise error in the actual system over the i th realization is given by $\|\mathbf{C}\boldsymbol{\varphi}_{ex}^{(i)}[s]\|_\infty$. Since the function $f(\mathbf{a}) = \|\mathbf{C}\mathbf{a}\|_\infty$ is convex in \mathbf{a} , we can use Jensen’s inequality, coupled with the relationship in (2.10) and (2.14), to show the following inequality

$$\|\mathbf{C}\boldsymbol{\varphi}_{ex}[s]\|_\infty \leq \frac{1}{M} \sum_{i=1}^M \|\mathbf{C}\boldsymbol{\varphi}_{ex}^{(i)}[s]\|_\infty \quad \text{for large } M$$

Thus, the maximum pairwise phase difference for the averaged system is a lower bound on the average of the maximum pairwise phase difference in the original system, averaged across realizations.

This bound holds true independent of the distribution of skews $\boldsymbol{\delta}$ across nodes. Therefore, any set of skews that are “bad” for the averaged system are guaranteed to be bad for the actual system as well. We use this observation in Section

3.8 to show that the mean value of the worst-case pairwise error between neighbors can increase with the size of the network, so that both frequency and phase adjustments are required for large networks.

2.4 Design of Phase-Frequency Adjustments

We now present an algorithm that achieves frequency and phase synchrony for the averaged system. Equation (2.10) shows that, after sufficiently many phase-only adjustments, the excess phases, and hence the residual phase differences between nodes, are a function of the skews. Conversely, these residual phase differences provide estimates of the skews, which are then used to make frequency adjustments. We note from (2.10) that $\boldsymbol{\delta} \approx (\mathbb{I} - \overline{\mathcal{G}}_{ex}) \boldsymbol{\varphi}_{ex}[s]$ for large enough s . For decentralized adaptation, we clearly do not have access to the excess phases $\boldsymbol{\varphi}_{ex}$. However, using the fact that $\overline{\mathcal{G}}$ is a stochastic matrix, we can also express the skews in terms of the raw phases as follows (see Appendix A.4 for details) :

$$\boldsymbol{\delta} \approx (\mathbb{I} - \overline{\mathcal{G}}) \boldsymbol{\varphi}[s] = \overline{\mathcal{L}} \boldsymbol{\varphi}[s] \tag{2.15}$$

for sufficiently large s , where the matrix $\overline{\mathcal{L}} = \mathbb{I} - \overline{\mathcal{G}}$ is called the Laplacian of the system. For our phase-only updates, it is easy to show (see Appendix A.5) that

the i th component of (2.15) can be written as

$$[\overline{\mathcal{L}}\varphi_\infty]_i = \beta \sum_{j=1}^N q_{j \rightarrow i} (\varphi_{\infty,i} - \varphi_{\infty,j}) = \delta_i \quad (2.16)$$

where $q_{j \rightarrow i}$ denotes the probability that \mathcal{N}_j transmits to \mathcal{N}_i (even though the summation is defined over all the nodes, we note that the only terms that contribute are those with $q_{j \rightarrow i} > 0$ i.e. nodes that are neighbors of \mathcal{N}_i). Thus, the i th node can estimate its excess frequency simply by considering the weighted average of its residual phase differences with its neighbors, with weights depending on the average communication pattern defined by $\overline{\mathcal{G}}$.

Our phase-frequency adjustment algorithm consists of estimating the skews based on the residual phase errors with neighbors after many phase-only adjustments, and adjusting the frequencies accordingly. We follow multiple rounds of this procedure, starting from round 0, with round r consisting of W_r slots. Each node employs the same algorithm, hence we describe it from the point of view of a particular node i :

- *Step 1:* In round r , node i makes phase-only adjustments for W_r slots;
- *Step 2:* At the end of round r , node i makes an estimate of its current skew, denoted by $\hat{\delta}_i$, by averaging the observed errors with its neighbors in the following slot according to (2.16);
- *Step 3:* If the skew is small enough, with $|\hat{\delta}_i| < \epsilon$ falling in a “dead zone,” node

i does not change its frequency. If $|\hat{\delta}_i| > \epsilon$, then node i makes a frequency jump of $\pm\mu$, depending on the sign of the skew. That is, it adds $\Delta F_i = -\mu \text{sign}(\hat{\delta}_i)$ to its frequency. (The dead zone parameter ϵ and the step size parameter μ are discussed below.)

- *Step 4:* Go back to Step 1 for round $r + 1$.

Algorithm parameters: We prove that our frequency adjustment algorithm guarantees that the maximum frequency deviation away from the mean of any node is at most $\epsilon + \mu + \chi$, where ϵ is the dead zone size, μ is the frequency adjustment step size, and χ is the designed error in frequency estimation at the end of each round. The parameters $\epsilon, \mu, \chi > 0$ can be chosen freely, except that our convergence proof requires that $\epsilon > \mu + \chi$ (the dead zone must be large enough to accommodate wrong frequency steps and frequency estimation errors). For example, starting with a maximum frequency offset of 100 ppm, we could eventually converge to a maximum offset of less than 4 ppm by choosing $\epsilon = 2$ ppm, $\mu = 1$ ppm and $\chi = 0.8$ ppm. This would reduce the worst-case phase offset in the network by a factor of 25.

Our theoretical development proceeds as follows. We first identify the number of slots needed in round r , which we denote by W_r , so that the maximum error in estimating the current skew of any node is smaller than χ . We then show that, if $\epsilon > \mu + \chi$, then in each round of frequency adjustment, either (a) the nodes

move towards convergence or (b) all current skews are lesser than $\epsilon + \mu + \chi$ (in magnitude) and will continue to remain so in the future. It is convenient at this point to introduce two-dimensional indexing of slots: slot $[s, r]$ is the s th slot in round r , and is therefore the k th slot in a one-dimensional enumeration of slots, where $k = \sum_{q=0}^{r-1} W_q + s$. In the r^{th} round, the slot index s takes values between 0 and $W_r - 1$. We use both one-dimensional and two-dimensional enumeration of slots, depending on the context.

2.4.1 Choosing round sizes

Since the mean phase does not contribute to the actions of our algorithms (which depend only on phase *differences*), we only need to model what happens to the excess phases. As in Section 2.3.1, this means that we can excise the eigenmode corresponding to eigenvalue one, which determines evolution of the mean phase. We show in Appendix A.6 that the excess phase in slot s of round r evolves as

$$\varphi_{ex}[s, r] = \overline{\mathcal{G}}_{ex}^s \varphi_{ex}[0, r] + D_s \delta[0, r] \quad (2.17)$$

where $D_s = \mathbb{I} + \overline{\mathcal{G}}_{ex} + \overline{\mathcal{G}}_{ex}^2 + \dots + \overline{\mathcal{G}}_{ex}^{s-1}$. Assuming that slot $s = W_r - 1$ is the last one in round r (i.e., the nodes estimate their current skews as $\overline{\mathcal{L}}\varphi[s, r]$), the

current skew estimates at the end of round r are given by

$$\hat{\boldsymbol{\delta}}[s, r] = (\mathbb{I} - \bar{\mathcal{G}}) \boldsymbol{\varphi}[s, r] = (\mathbb{I} - \bar{\mathcal{G}}_{ex}) \boldsymbol{\varphi}_{ex}[s, r] \quad (2.18)$$

Substituting the expression for the excess phases from equation (2.17) into equation (2.18), we get an expression for the errors incurred by the nodes with this estimate. Denoting these errors by $\mathbf{e}_\delta[s, r] = \hat{\boldsymbol{\delta}}[s, r] - \boldsymbol{\delta}[0, r]$, we get

$$\mathbf{e}_\delta[s, r] = (\bar{\mathcal{G}}_{ex}^s - \bar{\mathcal{G}}_{ex}^{s+1}) \boldsymbol{\varphi}_{ex}[0, r] - \bar{\mathcal{G}}_{ex}^s \boldsymbol{\delta}[0, r] \quad (2.19)$$

We approximate $\bar{\mathcal{G}}_{ex}^s$ by its dominant eigenmode $\lambda_2^s \mathbf{v}_2 \mathbf{v}_2^T$ and thereby, the errors as,

$$\mathbf{e}_\delta[s, r] \approx \lambda_2^s \mathbf{v}_2 \mathbf{v}_2^T (1 - \lambda_2) \boldsymbol{\varphi}_{ex}[0, r] - \lambda_2^s \mathbf{v}_2 \mathbf{v}_2^T \boldsymbol{\delta}[0, r] \quad (2.20)$$

We can now “invert” this relationship to show that the estimation error can be made arbitrarily small by waiting “long enough”. Specifically, the maximum error in estimating the excess frequency in round r can be made smaller than χ by choosing the number of slots W_r , to satisfy,

$$W_r \geq \frac{1}{\log(1/\lambda_2)} \times \left(\log \frac{\chi}{\|\mathbf{v}_2 \mathbf{v}_2^T\|_\infty} - \log \{ \|\boldsymbol{\varphi}_{ex}[0, r]\|_\infty + \|\boldsymbol{\delta}[0, r]\|_\infty (1 - \lambda_2) \} \right) \quad (2.21)$$

However, this inequality does immediately not tell us how long we need to wait in round r . This is because the RHS of (2.21) requires bounds on the excess

phases $\|\varphi_{ex}[0, r]\|_\infty$ and the excess frequencies $\|\delta[0, r]\|_\infty$ in round r , which are not readily available. But, we can obtain such bounds recursively: we relate $\|\varphi_{ex}[0, r]\|_\infty$ and $\|\delta[0, r]\|_\infty$ to their corresponding values in round $r - 1$ and ultimately, to bounds on the excess phases and frequencies at the start of our algorithm, which are assumed to be available. We now explain the recursive procedure.

For the first round, the accuracy of the coarse synchronization procedure at startup determines bounds on the excess phases $\|\varphi_{ex}[0, 1]\|_\infty$. Also, the largest excess frequency $\|\delta[0, 1]\|_\infty$ is bounded by the difference between the maximum frequency $(1 + \rho_{max})$ and the minimum frequency $(1 - \rho_{max})$. We use these bounds in equation (2.21) to calculate the number of slots in the first round W_1 . We now show how we can derive bounds on the excess phases and frequencies for Round 2.

First, we derive a general recursive relationship that relates the excess phases in round $r + 1$ to the excess phases and frequencies in round r (see Appendix A.7 for a proof):

$$\|\varphi_{ex}[0, r + 1]\|_\infty \leq \|\bar{\mathcal{G}}_{ex}^{W_r - 1}\|_\infty \|\varphi_{ex}[0, r]\|_\infty + (1 + \|D_{W_r - 1}\|_\infty) \|\delta[0, r]\|_\infty \quad (2.22)$$

We use the available bounds on $\|\varphi_{ex}[0, 1]\|_\infty$ and $\|\delta[0, 1]\|_\infty$ in this equation to bound $\|\varphi_{ex}[0, 2]\|_\infty$. In Section 2.4.2, we show that the excess frequencies in round

two $\delta[0, 2]$ are also bounded by $2\rho_{max}$ in magnitude. The basic idea behind this result is that the maximum (respectively, minimum) frequency in round 2 does not increase (respectively, decrease) from its value in round 1, despite the frequency adjustments made by nodes at the end of round 1. Substituting these bounds for $\|\delta[0, 2]\|_\infty$ and $\|\varphi_{ex}[0, 2]\|_\infty$ in equation (2.21), we can calculate the number of slots needed in round 2. By repeating these arguments, we can recursively compute the number of slots needed in any round r .

2.4.2 Convergence of frequency adjustment algorithm

By making each round long enough, the errors in estimating the excess frequencies can be made smaller than χ where χ can be arbitrarily small. We demonstrate that each frequency adjustment drives the network towards (and never away from) frequency synchrony. For this, we need to choose the width of the “dead zone” ϵ to be larger than the sum of the the estimation error χ and the frequency adjustment size μ . Specifically, we show that, after “many” rounds of frequency adjustments, the excess frequencies are no bigger than $\epsilon + \mu + \chi$. Since the latter quantity can be made as small as we wish, we can get arbitrarily close to perfect frequency synchrony.

Since frequencies change only over rounds, we index the current skews by round number: the current skew of node i in any slot of the r^{th} round is denoted by $\delta_i\{r\}$. We do not need to track phases any more, but rather repeatedly use the fact that the error in estimating the excess frequency is smaller than χ , or $|\delta_i\{r\} - \hat{\delta}_i\{r\}| \leq \chi \forall i, r$.

Consider a given round r . To show that the nodes always adjust their frequencies to drive the network to synchrony, we begin by splitting the nodes into ten sets based on the sign of their excess frequencies and how far they are from convergence. The split is shown in Figure 2.2. The first set $S_1\{r\}$ contains nodes $\delta_i\{r\} > \epsilon + \mu + \chi$ (positive skews, “very far” from convergence). The second set $S_2\{r\}$ contains nodes with $\epsilon + \chi < \delta_i\{r\} \leq \epsilon + \mu + \chi$ (positive skews, “far from convergence”). The next three sets $S_3\{r\}, S_4\{r\}, S_5\{r\}$ contain nodes that are “close”, “closer” and “closest” to convergence: for $S_3\{r\}$, $\delta_i\{r\} \in (\epsilon, \epsilon + \chi]$; for $S_4\{r\}$, $\delta_i\{r\} \in (\epsilon - \chi, \epsilon]$; for $S_5\{r\}$, $\delta_i\{r\} \in (0, \epsilon - \chi]$. The sets S_{-i} are defined analogously, except that the signs are reversed, as shown in the figure. We characterize the actions of nodes in each of these sets in a sequence of propositions and use this characterization to show that the algorithm converges and achieves frequency synchrony (to within $\epsilon + \mu + \chi$). We provide an outline of the proof before stating the results formally:

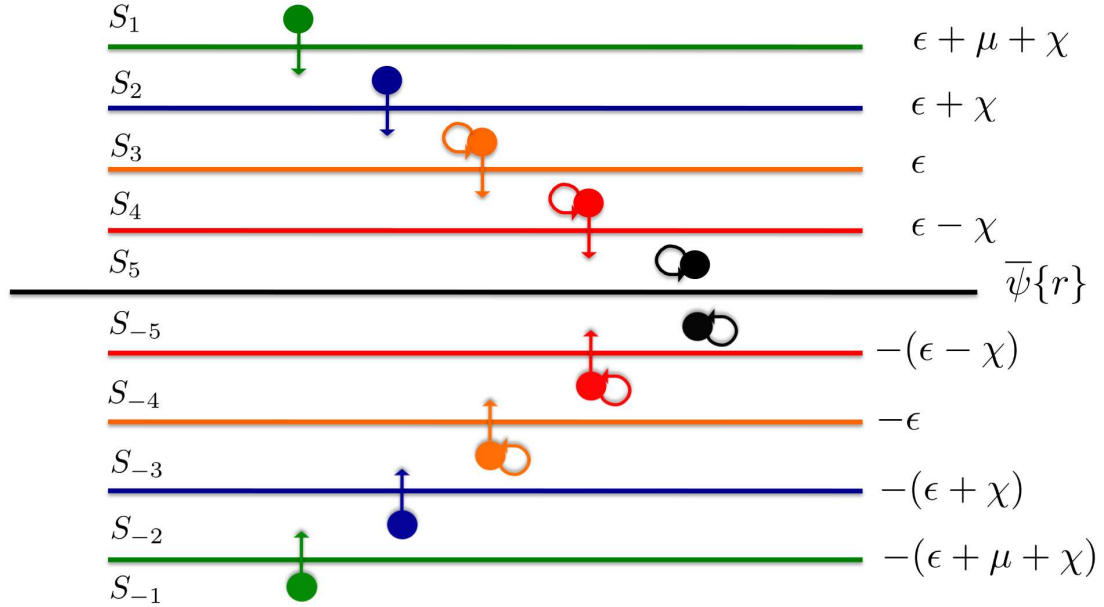


Figure 2.2: Splitting the nodes into 10 sets based on their excess frequencies. Nodes in S_1 and S_2 , that are “far” from convergence, are guaranteed to reduce their frequencies. Nodes in S_3 and S_4 , that are ”close” to convergence, either reduce their frequencies or do not change it, but never increase their frequencies. Nodes in S_5 , that are ”closest” to convergence, will not change their frequencies. Analogous results hold for nodes with negative excess frequencies.

1. First, we show that nodes either make “correct” frequency adjustments or do not change them at all, but never make a “wrong” frequency adjustment. Specifically, we show that a node with a positive current skew in round r either lowers its frequency by μ or does not change it at all. The node never increases its frequency further by μ . Furthermore, we show that nodes which are either far (S_2) or very far (S_1) from convergence are guaranteed to reduce

their frequencies by μ . Analogous results hold for nodes with negative skews in round r as well.

2. Next, we show that the frequency jumps are small (relative to the size of the dead zone), so that nodes do not cross one another while making frequency adjustments. Specifically, a node with a positive excess frequency in round r that makes a downward frequency jump never “crosses” a node with negative excess frequency in round r that is increasing its frequency.
3. The previous two results immediately imply that the maximum frequency across nodes either decreases or remains the same, but never increases (corresponding result holds for the minimum frequency too). Denoting the maximum and minimum frequencies across nodes in round r by $F_{max}\{r\}$ and $F_{min}\{r\}$ respectively, we show that $F_{max}\{r + 1\} \leq F_{max}\{r\}$ and $F_{min}\{r + 1\} \geq F_{min}\{r\}$. We also get an intuitively pleasing conclusion from this proposition: if we achieve frequency synchrony, the common frequency will be between the maximum and minimum frequencies at the start.
4. Next, we consider two disjoint possibilities: (a) the network is not close to convergence so that there is at least one node “very far” from the mean frequency (in either S_1 or S_{-1}) or (b) the network is close to convergence

so that all nodes have excess frequencies within $\epsilon + \mu + \chi$ of the mean (in $S_2 - S_5$ or their negative counterparts). We consider these cases separately.

- a. In this case, we show that the network will move towards convergence and not away from it. We do this in two stages: firstly, supposing that S_1 is non-empty, we use the first and second propositions to show that the maximum frequency will decrease by μ . Conversely, if S_{-1} is non-empty, the minimum frequency will increase by μ . We prove these results in Proposition 4. Then, assuming that one of these sets is non-empty, we identify a non-negative function of the frequencies – which can be used a measure of the network convergence level – that strictly decreases. Specifically, we show that the *difference* between the maximum and minimum frequencies $\xi^*\{r\} = F_{max}\{r\} - F_{min}\{r\}$ is bound to decrease at least by μ at the end of round r . This is proved in Proposition 5.

- b. When the network is close to convergence, we show that it will continue to remain in the same state and not drift away from convergence. Specifically, we show that a node with a “small” excess frequency in round r ($|\delta_i\{r\}| \leq \epsilon + \mu + \chi$) will continue to have a small excess frequency in round $r + 1$ ($|\delta_i\{r + 1\}| \leq \epsilon + \mu + \chi$). In other words, if

$\delta^*\{r\}$ denotes the largest excess frequency in round r (in magnitude) and $\delta^*\{r\} \leq \epsilon + \mu + \chi$, then we have $\delta^*\{r+1\} \leq \epsilon + \mu + \chi$. This is shown in Proposition 6.

5. If the network has converged in round r (case (b) above), then by the above argument it continues to remain converged in further rounds. Thus, the only way the network might not achieve frequency convergence is if case (a) is true in all rounds. However, this is impossible: since $\xi^*\{r\}$ is lower bounded by 0, it can only decrease so many times by μ . Thus, for some large enough round R_0 , case (b) must be true and the network will remain converged from this point onwards. We show this in Theorem 1.

We now state the propositions and the theorem formally.

Proposition 1. *If \mathcal{N}_i 's frequency in round r is larger than the mean, then its frequency in round $r+1$ cannot be larger than its value in round r i.e. if $F_i\{r\} \geq \bar{\psi}\{r\}$, then $F_i\{r+1\} \leq F_i\{r\}$. Furthermore, if \mathcal{N}_i 's frequency is "much larger" than the mean in round r , it will necessarily reduce its frequency: if $\mathcal{N}_i \in S_1\{r\} \cup S_2\{r\}$, then $F_i\{r+1\} = F_i\{r\} - \mu$. Analogously, if $F_i\{r\} \leq \bar{\psi}\{r\}$, then $F_i\{r+1\} \geq F_i\{r\}$, and if $\mathcal{N}_i \in S_{-1}\{r\}/S_{-2}\{r\}$, then $F_i\{r+1\} = F_i\{r\} + \mu$.*

Proof. We only prove the statement for nodes with positive current skews, or $F_i\{r\} \geq \bar{\psi}\{r\}$, since the proof for nodes with $F_i\{r\} \leq \bar{\psi}\{r\}$ is entirely analogous.

Case 1: If node i is in $S_1\{r\} \cup S_2\{r\}$, then $\delta_i\{r\} > \epsilon + \chi$. Since the estimation error is bounded by χ (we have ensured this by making the rounds long enough), the estimated current skew falls outside the dead zone: $\hat{\delta}_i\{r\} \geq \delta_i\{r\} - \chi > (\epsilon + \chi) - \chi = \epsilon$. Thus, node i adjusts its frequency downwards.

Case 2: For nodes in S_3, S_4 or S_5 , the current skew is positive: $0 < \delta_i\{r\} < \epsilon + \chi$. Since the estimation error is at most χ , we have $-\chi < \hat{\delta}_i\{r\} < \epsilon + 2\chi$. Since $\epsilon > \mu + \chi > \chi$, we conclude that $\hat{\delta}_i\{r\} > -\epsilon$, so that node i does not increase its frequency. \square

Proposition 2. *If $F_i\{r\} > \bar{\psi}\{r\}, F_i\{r+1\} > \bar{\psi}\{r\}$. Analogously, if $F_i\{r\} < \bar{\psi}\{r\}, F_i\{r+1\} < \bar{\psi}\{r\}$.*

Proof. Once again, we only prove the statement for nodes with frequencies larger than the mean, since the other case is exactly analogous.

Case 1: Let $\mathcal{N}_i \in S_5\{r\}$. Therefore, $0 \leq \delta_i\{r\} \leq \epsilon - \chi$. Since the estimation error is bounded by χ , we have $-\chi \leq \hat{\delta}_i\{r\} \leq \epsilon$. Since $\epsilon > \chi + \mu$, we have $-\chi > -\epsilon$. Therefore, the estimate $\hat{\delta}_i\{r\}$ falls in the dead zone and \mathcal{N}_i does not change its frequency at the end of round r . Thus, we have $F_i\{r+1\} = F_i\{r\}$ and by definition, $F_i\{r\} > \bar{\psi}\{r\}$, which together imply $F_i\{r+1\} > \bar{\psi}\{r\}$.

Case 2: If \mathcal{N}_i is in $S_1\{r\} \cup S_2\{r\} \cup S_3 \cup \{r\} \cup S_4\{r\}$, we have $F_i\{r\} > \bar{\psi}\{r\} + \epsilon - \chi$.

At the end of round r , \mathcal{N}_i can reduce its frequency at most by μ . Therefore, $F_i\{r+1\} \geq F_i\{r\} - \mu > \bar{\psi}\{r\} + \epsilon - \chi - \mu$. Since $\epsilon > \chi + \mu$, $F_i\{r+1\} > \bar{\psi}\{r\}$. \square

Proposition 3. *The maximum frequency can never increase and the minimum frequency can never decrease i.e. $F_{max}\{r+1\} \leq F_{max}\{r\}$ and $F_{min}\{r+1\} \geq F_{min}\{r\}$.*

Proof. We only prove that F_{max} can never increase, since the proof that F_{min} can never decrease is similar. Let $\Gamma^+\{r\}$ and $\Gamma^-\{r\}$ denote the set of all nodes with positive and negative excess frequencies in round r respectively i.e. $\Gamma^+\{r\} = \cup_{i=1}^5 S_i\{r\}$ and $\Gamma^-\{r\} = \cup_{i=1}^5 S_{-i}\{r\}$. Unless all nodes are at the same frequency (in which case we are done), neither of these sets can be empty. Otherwise, all the nodes would have their frequencies on one side of the mean frequency, which is impossible.

Let u^+ and u^- be any nodes in $\Gamma^+\{r\}$ and $\Gamma^-\{r\}$ respectively. From Proposition 2, we have $F_{u^+}\{r+1\} > \bar{\psi}\{r\}$ and $F_{u^-}\{r+1\} < \bar{\psi}\{r\}$. Therefore, we have $F_{u^+}\{r+1\} > F_{u^-}\{r+1\} \forall u^+, u^-$. Thus, the node with the maximum frequency in round $r+1$, denoted by u_{r+1}^* , must have had a positive excess frequency in round r , or $u_{r+1}^* \in \Gamma^+\{r\}$. But, if $u_{r+1}^* \in \Gamma^+\{r\}$, by Proposition 1, its frequency in round $r+1$ cannot be larger than its frequency in round r i.e. $F_{u_{r+1}^*}\{r+1\} \leq F_{u_{r+1}^*}\{r\}$. Since the maximum frequency across nodes in round r is larger than

u_{r+1}^* 's frequency in the same round, we get $F_{u_{r+1}^*}\{r\} \leq F_{max}\{r\}$. Chaining these inequalities, we get, $F_{max}\{r+1\} = F_{u_{r+1}^*}\{r+1\} \leq F_{u_{r+1}^*}\{r\} \leq F_{max}\{r\}$. Therefore, the maximum frequency can never increase across rounds. \square

Proposition 4. *If $S_1\{r\} \neq \emptyset$, $F_{max}\{r+1\} = F_{max}\{r\} - \mu$. Furthermore, if $\mathcal{N}^\#$ is the node with the maximum frequency in round r , it continues to be the node with the maximum frequency in round $r+1$. Analogously, if $S_{-1}\{r\} \neq \emptyset$, $F_{min}\{r+1\} = F_{min}\{r\} + \mu$. Also, if $\mathcal{N}^\#$ is the node with the minimum frequency in round r , it continues to be the node with the minimum frequency in round $r+1$*

Proof. As usual, we only prove the statement for the case where $S_1\{r\} \neq \emptyset$. Let $\mathcal{N}^\#$ be the node with the maximum frequency in round r . Since $S_1\{r\} \neq \emptyset$, $\mathcal{N}^\# \in S_1\{r\}$. Thus, from Proposition 1, $\mathcal{N}^\#$ reduces its frequency after round r . Therefore, $F_{\mathcal{N}^\#}\{r+1\} = F_{\mathcal{N}^\#}\{r\} - \mu$. We now prove the proposition by considering three cases.

Case 1: Let \mathcal{N}_i be any node other than $\mathcal{N}^\#$ in $S_1\{r\} \cup S_2\{r\}$. From Proposition 1, \mathcal{N}_i also reduces its frequency after round r , or $F_i\{r+1\} = F_i\{r\} - \mu$. Therefore, \mathcal{N}_i continues to have a frequency smaller or equal to that of $\mathcal{N}^\#$ in round $r+1$: $F_{\mathcal{N}^\#}\{r+1\} - F_i\{r+1\} = (F_{\mathcal{N}^\#}\{r\} - \mu) - (F_i\{r\} - \mu) = F_{\mathcal{N}^\#}\{r\} - F_i\{r\}$, which, by definition, is greater than or equal to zero.

Case 2: Let \mathcal{N}_i be a node in $S_3\{r\}$ or $S_4\{r\}$ or $S_5\{r\}$. Thus, \mathcal{N}_i 's frequency in

the r^{th} round is not too large: $F_i\{r\} \leq \bar{\psi}\{r\} + \epsilon + \chi$. Furthermore, from Proposition 1, \mathcal{N}_i does not increase its frequency at the end of round r i.e. $F_i\{r+1\} \leq F_i\{r\}$. Therefore, we have, $F_i\{r+1\} \leq \bar{\psi}\{r\} + \epsilon + \chi$. However, $\mathcal{N}^\#$'s frequency in round $r+1$ is larger than this : $F_{\mathcal{N}^\#}\{r+1\} = F_{\mathcal{N}^\#}\{r\} - \mu > (\bar{\psi}\{r\} + \epsilon + \mu + \chi) - \chi = \bar{\psi}\{r\} + \epsilon + \chi$ (the inequality follows from the fact that $\mathcal{N}^\# \in S_1\{r\}$).

Case 3: Consider \mathcal{N}_i with a negative excess frequency in round r . From Proposition 2, we know that $F_i\{r+1\} < \bar{\psi}\{r\}$. However, we have already shown that $F_{\mathcal{N}^\#}\{r+1\} > \bar{\psi}\{r\} + \epsilon + \chi$. Therefore, $F_{\mathcal{N}^\#}\{r+1\} > \bar{\psi}\{r\} > F_i\{r+1\}$.

Therefore, $F_{\mathcal{N}^\#}\{r+1\} \geq F_i\{r+1\} \forall i$ and $\mathcal{N}^\#$ continues to be the node with the highest frequency in round $r+1$. Since it was also the node with maximum frequency in the r^{th} round and its frequency decreased by μ at the end of round r , we have $F_{\max}\{r+1\} = F_{\max}\{r\} - \mu$. \square

Proposition 5. *If $S_1\{r\}$ or $S_{-1}\{r\} \neq \emptyset$, $\xi^*\{r+1\} \leq \xi^*\{r\} - \mu$. Furthermore, if both $S_1\{r\}$ and $S_{-1}\{r\} \neq \emptyset$, $\xi^*\{r+1\} = \xi^*\{r\} - 2\mu$.*

Proof. Consider the case when $S_1\{r\}$ and $S_{-1}\{r\} \neq \emptyset$. Then, from Proposition 4, $F_{\max}\{r+1\} = F_{\max}\{r\} - \mu$ and $F_{\min}\{r+1\} = F_{\min}\{r\} + \mu$. Therefore, $\xi^*\{r+1\} = F_{\max}\{r+1\} - F_{\min}\{r+1\} = (F_{\max}\{r\} - \mu) - (F_{\min}\{r\} + \mu) = \xi^*\{r\} - 2\mu$.

Now consider the case when $S_1 \neq \emptyset$ but $S_{-1} = \emptyset$ (the reversed case is very similar). From Proposition 4, $F_{\max}\{r+1\} = F_{\max}\{r\} - \mu$. Similarly, from Proposition

3, $F_{min}\{r+1\} \geq F_{min}\{r\}$. Therefore, we can see that ξ^* must decrease at least by μ as follows: $\xi^*\{r+1\} = F_{max}\{r+1\} - F_{min}\{r+1\} = (F_{max}\{r\} - \mu) - F_{min}\{r+1\} \leq (F_{max}\{r\} - \mu) - F_{min}\{r\} = \xi^*\{r\} - \mu$. \square

To tackle the scenario when there is no node very far from convergence, we need a preliminary result which shows that the mean frequency does not change a lot across rounds.

Corollary 1. *The mean frequency $\bar{\psi}\{r+1\}$ is bounded below by $\bar{\psi}\{r\} - \mu$ and above by $\bar{\psi}\{r\} + \mu$.*

Proof. Since \mathcal{N}_i changes its frequency by at most μ , $F_i\{r\} - \mu \leq F_i\{r+1\} \leq F_i\{r\} + \mu$. Adding these inequalities across nodes and dividing by the number of nodes, we get $\bar{\psi}\{r\} - \mu \leq \bar{F}_i\{r+1\} \leq \bar{\psi}\{r\} + \mu$. \square

Proposition 6. *If $\mathcal{N}_i \notin S_1\{r\} \cup S_{-1}\{r\}$ so that $|\delta_i\{r\}| \leq (\epsilon + \mu + \chi)$, then $|\delta_i\{r+1\}| \leq (\epsilon + \mu + \chi)$.*

Proof. We only prove the statement for nodes with positive excess frequencies in the r^{th} round. For notational ease, we denote the change in the mean frequency from round r to round $r+1$ by $\Delta\bar{\psi}\{r\} = \bar{\psi}\{r+1\} - \bar{\psi}\{r\}$. From corollary 1, we have $|\Delta\bar{\psi}| \leq \mu$.

Case 1: Let $\mathcal{N}_i \in S_2\{r\}$. From Proposition 1, we know that \mathcal{N}_i decreases its

frequency at the end of round r , or $F_i\{r+1\} = F_i\{r\} - \mu$. Consequently, its excess frequency in round $r+1$ is, $\delta_i\{r+1\} = F_i\{r+1\} - \bar{\psi}\{r+1\} = F_i\{r\} - \mu - \bar{\psi}\{r+1\}$. Adding and subtracting $\bar{\psi}\{r\}$ to the extreme right hand side and grouping terms, we get $\delta_i\{r+1\} = \delta_i\{r\} - \mu - \Delta\bar{\psi}\{r\}$. Since $\mathcal{N}_i \in S_2\{r\}$, we have bounds on its excess frequency, $\epsilon + \chi < \delta_i\{r\} \leq \epsilon + \chi + \mu$. Using these bounds and the fact that $|\Delta\bar{\psi}\{r\}| \leq \mu$ in the expression for $\delta_i\{r+1\}$, we get, $\epsilon + \chi - 2\mu \leq \delta_i\{r+1\} \leq \epsilon + \chi + \mu$. We also have $\epsilon + \chi - 2\mu = -(\epsilon + \mu + \chi) + (2\epsilon - \mu + 2\chi) > -(\epsilon + \mu + \chi)$ since we have chosen $\epsilon > \mu + \chi$. Therefore, $|\delta_i\{r+1\}| \leq (\epsilon + \chi + \mu)$.

Case 2: Let \mathcal{N}_i be in $S_3\{r\} \cup S_4\{r\} \cup S_5\{r\}$. From Proposition 1, we know that \mathcal{N}_i does not increase its frequency after round r , or $F_i\{r+1\} \leq F_i\{r\}$. From Proposition 2, we also know that it does not decrease its frequency too much, $F_i\{r+1\} > \bar{\psi}\{r\}$. Chaining these inequalities, we get, $\bar{\psi}\{r\} \leq F_i\{r+1\} \leq F_i\{r\}$. We first subtract $\bar{\psi}\{r\}$ throughout and then, add and subtract $\bar{\psi}\{r+1\}$ to the middle term and obtain, $0 \leq \delta_i\{r+1\} + \Delta\bar{\psi}\{r\} \leq \delta_i\{r\}$. Equivalently, we have $-\Delta\bar{\psi}\{r\} \leq \delta_i\{r+1\} \leq \delta_i\{r\} - \Delta\bar{\psi}\{r\}$. Since $\mathcal{N}_i \in S_3\{r\} \cup S_4\{r\} \cup S_5\{r\}$, we have $0 \leq \delta_i\{r\} \leq \epsilon + \chi$. Using this bound and the fact that $|\Delta\bar{\psi}\{r\}| \leq \mu$ in the expression for $\delta_i\{r+1\}$ we get, $-\mu \leq \delta_i\{r+1\} \leq \epsilon + \mu + \chi$. Since $-\mu > -(\epsilon + \mu + \chi)$, we conclude that $|\delta_i\{r+1\}| \leq (\epsilon + \mu + \chi)$. \square

Theorem 1. Let $\xi^*\{r\} = F_{max}\{r\} - F_{min}\{r\}$ and $\delta^*\{r\} = \max_i |\delta_i\{r\}|$. In each round r , atleast one of the following statements is true,

1. $\xi^*\{r + 1\} \leq \xi^*\{r\} - \mu$.
2. $\delta^*\{r\}$ and $\delta^*\{r + 1\} \leq (\epsilon + \mu + \chi)$

Therefore, eventually, the maximum deviation in frequency from the mean is atmost $\epsilon + \mu + \chi$.

Proof. In the r^{th} round of frequency adjustments, $S_{far}\{r\} \triangleq S_1\{r\} \cup S_{-1}\{r\}$ is either empty or it is non-empty. If $S_{far}\{r\}$ is nonempty, then by Proposition 5, $\xi^*\{r + 1\} \leq \xi^*\{r\} - \mu$. On the other hand, if $S_{far}\{r\}$ is empty, all the nodes are in $(\cup_{i=2}^5 S_i\{r\}) \cup (\cup_{i=2}^5 S_{-i}\{r\})$. Thus, the excess frequency of each node is smaller (in magnitude) than $(\epsilon + \mu + \chi)$ and hence, $\delta^*\{r\} \leq (\epsilon + \mu + \chi)$. In this scenario, Proposition 6 guarantees that $|\delta_i\{r + 1\}| \leq (\epsilon + \mu + \chi) \forall i \Rightarrow \delta^*\{r + 1\} \leq (\epsilon + \mu + \chi)$. Using recursion, we can conclude that if $\delta^*\{r\} \leq (\epsilon + \mu + \chi)$, then $\delta^*\{r'\} \leq (\epsilon + \mu + \chi) \forall r' \geq r$.

We have already seen that $\xi^*\{r\} \geq 0 \forall r$. Assume that the node frequencies are bounded by $1 \pm \rho_{max}$ before any frequency adjustments are made. Therefore, $\xi^*\{1\} = F_{max}\{1\} - F_{min}\{1\} \leq 2\rho_{max}$. If condition (1) is true in round r , we know that ξ^* decreases by μ . However, condition 1 cannot be true indefinitely because $\xi^*\{r\}$ is lower bounded by 0 and $\xi^*\{1\}$ is finite. Therefore, condition 2 will have to

be true in some round $r = R_0$ for the first time. By our proof, it will then continue to hold forever. Therefore, we can conclude that $\delta^*\{r'\} \leq \epsilon + \mu + \chi \forall r' \geq R_0$.

□

We can now estimate R_0 - the number of rounds needed for the maximum excess frequency to become smaller than $\epsilon + \mu + \chi$ for the first time. By definition, we have, $\xi^*\{R_0\} \leq 2\delta^*\{R_0\} / \leq 2(\epsilon + \mu + \chi)$. In each round r between 1 and R_0 , we know from Theorem (1) that ξ^* decreases by μ (at least). Therefore, R_0 needs to be no bigger than $\frac{2\rho_{max} - 2(\epsilon + \mu + \chi)}{\mu}$. Note that, while the number of *slots* needed per round of frequency adjustment might (and typically does) increase with network size, the number of *rounds* required for convergence is independent of the number of nodes in the network.

2.5 LLN Arguments

We design a phase-frequency adjustment scheme for the actual system based on the insights from the proposed algorithm for the averaged system. There are two important design guidelines we infer:

- The first lesson is that we must not adjust the frequencies in each slot, but only once in a round consisting of many slots. The intuitive reason for this is as follows. The phase error between a pair of nodes \mathcal{N}_i and \mathcal{N}_j changes over

time for two reasons: (a) the clock frequencies at \mathcal{N}_i and \mathcal{N}_j are different and (b) \mathcal{N}_i and \mathcal{N}_j adjust their phases each time they receive a packet from any of their neighbors (note that this could be a completely different node \mathcal{N}_k). Let e_1 and e_2 be the errors that \mathcal{N}_i observes on *successive* occasions that it receives a packet from \mathcal{N}_j (the successive occasions could be many slots apart). If we could apportion the change in the phase error $e_2 - e_1$ to the two cases, \mathcal{N}_i could estimate the frequency error using the contribution from case (a) and adjust its frequency accordingly. However, the specific technique to split $e_2 - e_1$ into its constituent components is unclear. By waiting for many slots, we allow the contribution to the phase error from frequency differences to build (this contribution grows linearly with time), thereby allowing us to estimate the frequency differences accurately.

- With this understanding, suppose that the nodes only change their frequencies once in a round consisting of S_R slots. The question then becomes: how should a given node \mathcal{N}_i adjust its frequency based on the phase errors that it observes with its neighbors over the round? We guess the rule based on the *interpretation* of the phases in the averaged system, which we describe next. Consider the phase evolution in the actual system (with mean frequency 0):

$$\varphi[s + 1] = G_s \varphi[s] + \delta$$

Taking an expectation with respect to the random communication pattern on both sides of this equation, we get,

$$\mathbb{E}(\varphi[s + 1]) = \mathbb{E}(G_s \varphi[s]) + \delta$$

Since the evolution of the phases until the beginning of slot s is completely independent of the communication in slot s , we have $\mathbb{E}(G_s \varphi[s]) = \mathbb{E}(G_s) \mathbb{E}(\varphi[s])$. Thus, the above equation is identical to the evolution equation in the averaged system:

$$\mathbb{E}(\varphi[s + 1]) = \bar{\mathcal{G}} \mathbb{E}(\varphi[s]) + \delta$$

Therefore, the phases in the averaged system are nothing but the *ensemble average* of the phases in the actual system and nodes in the averaged system adjust their frequencies based on the differences between ensemble averages of their phases. While measuring the ensemble average is impossible in the actual system, we guess that nodes must adjust their frequencies based on the differences in the *empirical averages* of the phases. Indeed, simulations provide evidence to support this guess: nodes whose excess frequencies are positive have phases that are typically (but not always) ahead of their neighbors with smaller excess frequencies. We track the average of the node phases over a round and use the LLN to explain how nodes can estimate their excess frequencies from such average phases.

We describe the frequency adjustment procedure only for the first round of slots, with the understanding that the same procedure is repeated in subsequent rounds.

Let the running average of the phases from the start of the round to a slot s ($0 \leq s \leq S_R - 1$) be denoted by $\varphi_{av}[s] = \sum_{s'=0}^s \varphi[s'] / (s + 1)$. We compute an expression for $\varphi_{av}[s]$ using the system evolution equation and use the LLN to show that the excess component of φ_{av} satisfies,

$$\varphi_{av,ex}[S_R - 1] \approx (\mathbb{I} - \overline{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta}[0] \quad (2.23)$$

when the number of slots in the round S_R is “large”. While the nodes do not have access to $\varphi_{av,ex}$, they can compute their excess frequencies in a simple manner that is motivated by (2.23). The key steps in the development are as follows:

- Using the expression for $\varphi[s]$ from (2.11), we obtain the average phases at the end of the round $\varphi_{av}[S_R - 1]$ to be:

$$\begin{aligned} \varphi_{av}[S_R - 1] = & \frac{[\sum_{s'=2}^{S_R-1} s' \overline{\psi}[0]]}{S_R} \mathbf{1} + \frac{\varphi[0] + \varphi[1] + \sum_{s'=2}^{S_R-1} G_{s'-1} G_{s'-2} \dots G_0 \varphi[0]}{S_R} + \\ & \frac{\sum_{s'=2}^{S_R-1} (\mathbb{I} + \sum_{p=1}^{s'-1} G_{s'-1} G_{s'-2} \dots G_{s'-p})}{S_R} \boldsymbol{\delta}[0] \quad (2.24) \end{aligned}$$

- We simplify the second term in this equation using an approximation that is guided by (2.12). Specifically, we approximate the product of the stochastic matrices $G_s G_{s-1} \dots G_0$ by a rank-one matrix of the form $\mathbf{1} \boldsymbol{\gamma}^T$ when the number of matrices in the product exceeds a critical limit S_w . The critical limit S_w depends

on the set $\{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$ from which the matrices $\{G_t\}$ are chosen and the probabilities $\{p_1, p_2, \dots, p_M\}$ with which they are chosen from this set. With this approximation, we get the averaged phases in slot $S_R - 1$ to be,

$$\begin{aligned} \varphi_{av}[S_R - 1] = & \frac{[\sum_{s'=2}^{S_R-1} s' \bar{\psi}[0] + \sum_{s'=S_w}^{S_R-1} (\gamma^T \varphi[0])]}{S_R} \mathbf{1} \\ & + \frac{\varphi[0] + \varphi[1] + \sum_{s'=2}^{S_w-1} G_{s'-1} G_{s'-2} \dots G_0 \varphi[0]}{S_R} + \\ & \frac{\sum_{s'=2}^{S_R-1} (\mathbb{I} + \sum_{p=1}^{s'-1} G_{s'-1} G_{s'-2} \dots G_{s'-p})}{S_R} \delta[0] \quad (2.25) \end{aligned}$$

- We can discard the first two terms for the purposes of frequency adjustment for different reasons. The frequency adjustment rule is only driven by the excess component of the averaged phases and not by the average component. Therefore, we can discard the first term. The second term is a transient when the number of slots S_R is large. Thus, only the third term contributes to the *excess-averaged-phases* $\varphi_{av,ex}$.

- We switch the order of the summations in the third term and collect all the terms in the summation which are products of the same *number* of system matrices. We then use the LLN and the approximation described above to simplify the summations and prove (2.23) when $S_R \gg S_w$ and $S_R \gg S_{\bar{\mathcal{G}}}$, where $S_{\bar{\mathcal{G}}}$ is large enough for us to set $\lambda_2^{S_{\bar{\mathcal{G}}}} \approx 0$ (and hence, $\bar{\mathcal{G}}_{ex}^{S_{\bar{\mathcal{G}}}} \approx 0$). We provide the details in Appendix A.8.

While the excess frequencies satisfy $\boldsymbol{\delta}[0] \approx (\mathbb{I} - \overline{\mathcal{G}}_{ex})\boldsymbol{\varphi}_{av,ex}$, they cannot be estimated in this fashion, since we do not have access to the excess-averaged-phases $\boldsymbol{\varphi}_{av,ex}$. But, we can use the fact that $\overline{\mathcal{G}}$ is a stochastic matrix and mimic the proof for the averaged system (see Appendix A.4) to show that $(\mathbb{I} - \overline{\mathcal{G}}_{ex})\boldsymbol{\varphi}_{av,ex} = (\mathbb{I} - \overline{\mathcal{G}})\boldsymbol{\varphi}_{av}$. Thus, the nodes can estimate their excess frequencies from the raw phases as $\boldsymbol{\delta}[0] \approx (\mathbb{I} - \overline{\mathcal{G}})\boldsymbol{\varphi}_{av}[S_R - 1]$.

Deriving an explicit frequency adjustment rule: Let us denote the averaged phase at node i in slot $S_R - 1$ by $\varphi_{av,i}[S_R - 1]$. By substituting the expression for $\overline{\mathcal{G}}$ (and mimicking the proof in Appendix A.5), we get the skew estimate of the i th node to be,

$$\hat{\delta}_i = \beta \sum_{j \neq i} q_{j \rightarrow i} (\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1]) \quad (2.26)$$

where $q_{j \rightarrow i}$ is the probability that node j transmits to node i . We cannot implement this directly for two reasons: (1) nodes are not aware of the link activity probabilities $q_{j \rightarrow i}$ and (2) they cannot measure the difference $(\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1])$. To see why the second reason is true, consider the difference

$$\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1] = \sum_{s=0}^{S_R-1} \frac{\varphi_i[s] - \varphi_j[s]}{S_R} \quad (2.27)$$

Therefore, $\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1]$ depends on the phase error between \mathcal{N}_i and \mathcal{N}_j in all slots between 0 and $S_R - 1$. However, \mathcal{N}_i can only measure the errors in those slots when it receives a packet from \mathcal{N}_j . But, \mathcal{N}_i can approximate

$\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1]$ by averaging over the phase errors that it does measure. Specifically, suppose that \mathcal{N}_j that has a nonzero probability of transmitting a packet to \mathcal{N}_i and that it transmits on n_{ij} occasions in the first round of slots to \mathcal{N}_i . We denote these slots by $s_{j \rightarrow i}(1), s_{j \rightarrow i}(2), \dots, s_{j \rightarrow i}(n_{ij})$ respectively. Then, \mathcal{N}_i approximates the average phase error over the first round with \mathcal{N}_j to be,

$$\varphi_{av,i}[S_R - 1] - \varphi_{av,j}[S_R - 1] \approx \sum_{t=1}^{n_{ij}} \frac{\varphi_i[s_{j \rightarrow i}(t)] - \varphi_j[s_{j \rightarrow i}(t)]}{n_{ij}} \quad (2.28)$$

We use the empirical definition of probability to approximate $q_{j \rightarrow i}$ as $q_{j \rightarrow i} \approx n_{ij}/S_R$ and substitute equation (2.28) in equation (2.26) to get,

$$\hat{\delta}_i[S_R - 1] \approx \frac{\beta}{S_R} \sum_{j \neq i} \sum_{t=1}^{n_{ij}} \left(\varphi_i[s_{j \rightarrow i}(t)] - \varphi_j[s_{j \rightarrow i}(t)] \right) \quad (2.29)$$

We use this as the defining equation for \mathcal{N}_i 's estimate of its excess frequency: it simply adds the phase errors it observes when it receives packets from its neighbors over the entire round (and scales it by β/S_R) to decide on the frequency adjustment at the end of the round. This rule is intuitively pleasing: if a node finds that despite all the phase adjustments made, its clock is still “ahead” ($\varphi_i - \varphi_j > 0$) of its neighbors' clocks over a long time, it concludes that its clock is running faster than the average across the network. We now summarize the algorithm from the point of view of node i .

Overall algorithm: Node i estimates its skew over a round according to equation (2.29). If $\hat{\delta}_i$ falls within a dead zone of width ϵ_{approx} (i.e. $|\hat{\delta}_i| \leq \epsilon_{approx}$),

node i does not change its frequency; in other cases, node i changes its frequency by $\pm\mu$ based on the sign of $\hat{\delta}_i$ i.e. it makes a frequency jump $\Delta F_i = -\mu \text{sign}(\hat{\delta}_i)$ at the end of a round of S_R slots. In subsequent rounds, the process is repeated by resetting the time-averaged phases and recomputing them from the start of the round.

An attractive feature of the frequency adjustment rule is the minimal storage required at each node, in spite of the significant amount of memory the rule entails. To adjust its frequency, each node needs to store only one number and recursively update it. Thus, the storage needed is independent of the size of the network and the network topology.

Tackling Measurement Noise: When the implicit timestamps are noisy, we leave the phase adjustment rule unaltered but change the frequency adjustment rule. We now describe these modifications. Let $\xi_{ij}[s_{j \rightarrow i}(t)] = \varphi_i[s_{j \rightarrow i}(t)] - \varphi_j[s_{j \rightarrow i}(t)]$ denote the true phase offset between \mathcal{N}_i and \mathcal{N}_j in slot $s_{j \rightarrow i}(t)$ and $\hat{\xi}_{ij}[s_{j \rightarrow i}(t)]$ denote \mathcal{N}_i 's estimate of this offset. The phase offset estimation error $\hat{\xi}_{ij}[s_{j \rightarrow i}(t)] - \xi_{ij}[s_{j \rightarrow i}(t)]$, induced by the measurement noise is denoted by $\gamma_{ij}[s_{j \rightarrow i}(t)]$. From equation (2.29), this translates to an error of

$$\Delta e_i = \frac{\beta}{S_R} \sum_{j=1}^N \sum_{t=1}^{n_{ij}} \gamma_{ij}[s_{j \rightarrow i}(t)]$$

in \mathcal{N}_i 's estimate of its excess frequency. We model the errors $\gamma_{ij}[s_{j \rightarrow i}(t)]$ as random variables that are independent across slots and node-pairs. Furthermore, we assume that these errors are picked from the same distribution $f_\Gamma(\gamma)$ with zero mean and a standard deviation σ_f . For example, in our simulations, we choose $f_\Gamma(\gamma)$ to be uniform. Let us assume that \mathcal{N}_i receives $P_i = \sum_j n_{ij}$ packets, from all its neighbors combined, in a round of S_R slots. We can rewrite the error in estimating the excess frequencies due to the measurement noise as $\Delta e_i = \frac{\beta P_i}{S_R} \times \frac{1}{P_i} \sum_{j=1}^N \sum_{t=1}^{n_{ij}} \gamma_{ij}[s_{j \rightarrow i}(t)]$. If P_i is reasonably large, we can invoke the Central Limit Theorem to approximate the distribution of $\frac{1}{P_i} \sum_j \sum_t \gamma_{ij}[s_{j \rightarrow i}(t)]$ by a Gaussian with zero mean and variance $\sigma^2 = \sigma_f^2/P_i$. Since a Gaussian random variable is very likely to be contained within three standard deviations on either side of the mean, we approximate the estimation error induced by the noise Δe_i to be bounded in magnitude by $(\beta P_i/S_R) \times (3\sigma) = (\beta P_i/S_R) \times (3\sigma_f/\sqrt{P_i})$. To combat this error in the estimate of its excess frequency, \mathcal{N}_i further increases the size of the dead zone from its value in the noiseless scenario. This increase needs to be no larger than the maximum value that the error can take, or $\frac{\beta P_i}{S_R} 3\sigma_f/\sqrt{P_i}$. However, in our simulations we are more conservative and increase the width of dead zone by $3\sigma_f/\sqrt{P_i}$ (note that $\beta P_i/S_R$ is guaranteed to be smaller than 1). To summarize, the only change in the frequency adjustment rules is an increase in the width of the dead zone; \mathcal{N}_i 's dead zone in the presence of measurement noise

with variance σ_f^2 given by $\epsilon'_i = \epsilon_{approx} + 3\sigma_f/\sqrt{P_i}$ where ϵ_{approx} is the dead-zone width in the noiseless setting and P_i is the number of packets received by \mathcal{N}_i over a round of slots.

2.6 Simulation Results

We now describe the models and parameters used in our simulations, followed by the results.

Topologies: We consider the *ring* topology (each node has two neighbors) and the rectangular *grid* topology (each node has four neighbors) as canonical examples of two-dimensional networks with large and small diameter, respectively.

Interference Model: We use the node exclusive interference model (two links can be active if they do not have a node in common) as an abstraction for “highly directional” networks (such as mm-wave networks [41]). For omnidirectional networks, we use the 2-hop interference model [36]: two links can be active simultaneously as long as they do not have a node in common, and the nodes involved in the two links are not neighbors.

Communication Patterns: We consider TDM schedules in which a randomly chosen maximal matching is active in each slot; a *matching* is a set of links that can be simultaneously active without interfering with one another, and a *maximal*

matching is a matching to which no link can be added without interfering with one of the active links. For large networks (e.g., a 64 node grid with directional links), it is infeasible to enumerate all maximal matchings. In this case, for each link, we randomly choose 120 maximal matchings in which it participates. The overall set of maximal matchings $\mathcal{S}_{matching}$ is obtained by taking the union of the sets generated for each link.

Opportunistic listening: In omnidirectional networks, we consider two modes for timing adjustment. In the ONLYINTENDED mode, a node adjusts its clock only when it receives a packet explicitly addressed to it. In the EAVESDROP mode, a node adjusts its clock whenever it can receive a collision-free packet, even if it is not the intended recipient.

Simulation Parameters: We fix $\beta = 0.5$ in all our simulations. Our choice of timescales in directional networks are motivated by Gigabit rate mm-wave networks: we choose the slot time $T_{slot} = 10\mu s$ and for the coarse synchronization procedure at startup, we choose $\tau_{min} = 10\mu s$ and $\tau_{max} = 300\mu s$ (see Section 2.2). The timescales in omnidirectional settings are motivated by WiFi-style networks: we choose $T_{slot} = 10ms$ and $\tau_{min} = 10ms$ and $\tau_{max} = 300ms$. We set the maximum value of the skew ρ_{max} to 50 ppm.

Performance Metric: Since TDM slotting overhead depends on the phase error

among communicating nodes, the key performance metric is the *worst neighbor timing error*, which is the largest magnitude of phase error *between neighbors*.

2.6.1 Phase-Only Adjustments

We consider directional networks and omnidirectional networks in the ONLY-INTENDED mode consisting of 9-64 nodes set in ring and grid topologies. For each topology, we generate a set of “bad” skews by solving linear programs that optimize the phase error on each link of the averaged system. We choose these “bad” skews to be the distribution of skews across nodes in the actual system. We then average the worst error between nodes in the actual system at the end of 3000 slots over 2000 realizations of communication patterns. The results for the directional and omnidirectional settings are shown in Figures 2.3 and 2.4 respectively. The figures confirm that the errors for the averaged system are indeed a lower bound for those in the actual system, and that the worst neighbor error increases with network size. The error grows faster with N for a ring topology. For directional ring networks (Figure 2.3(a)), the error grows as N in the averaged system and as $N^{1.33}$ (approximately) in the actual system. For directional grid networks (Figure 2.3(b)), the error grows only as $N^{0.66}$ in the averaged system and as $N^{0.875}$ (approximately) in the actual system.

Typical numbers for directional networks: From Figure 2.3, we see that the error between neighbors in a 64 node network set in a ring topology can be as large as 220 ns. This is comparable in magnitude to the guard interval needed to handle propagation delays. For example, the envisioned Gigabit rate outdoor mesh networks with link ranges on the order of 200 m require guard intervals of $200m/(3 \times 10^8m/s) \approx 666ns$ to handle propagation delays. Therefore, an additional synchronization error of 220 ns represents a 33% increase in the overhead. On the other hand, for small directional networks in a ring topology (say, 16 nodes) or reasonably large networks in a grid topology (36 nodes), we see that the largest phase error between neighbors will only be 40 ns. Therefore, the maximum increase in the overhead is only 6 %. To summarize, phase-only adjustments with implicit timestamps may suffice in small directional networks with linear topologies or moderately sized networks in grid topologies, but frequency adjustments are necessary for large networks, especially in linear topologies.

Recommendations for omnidirectional networks: Since the slot duration T_{slot} is longer in omnidirectional networks and the phase error scales in proportion to T_{slot} , we see from Figure 2.4 that the errors with phase-only adjustments are also correspondingly larger (~ 10 's of μs even for small networks). However, the guard interval needed to handle propagation delays is still $\approx 1\mu s$ since the link ranges are on the order of hundreds of meters. Therefore, phase-only adjustments

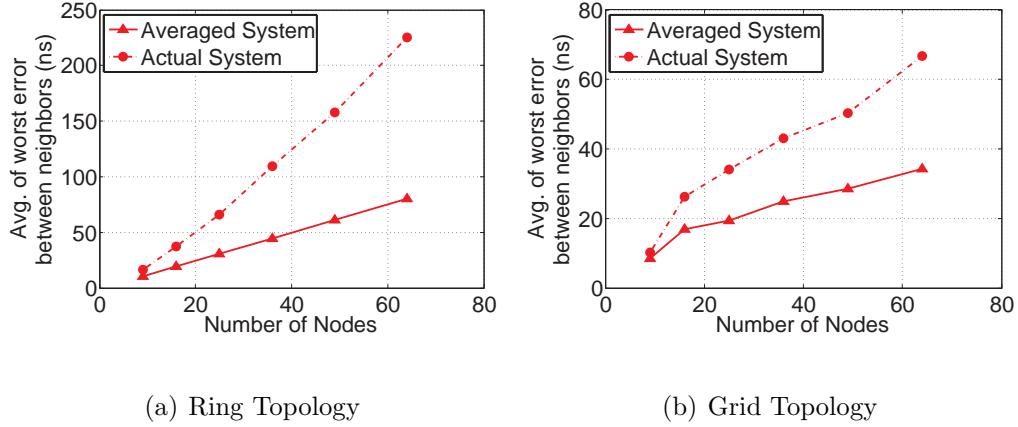


Figure 2.3: Worst error between neighbors for the actual system and the averaged system with only phase adjustments in a *directional network*.

are not sufficient to guarantee phase synchrony on the order of the guard interval in omnidirectional networks. However, a more practical definition of the tolerable overhead is to define it as a fraction of the payload (slot size) rather than the guard interval for propagation delay. Phase-only adjustments might then suffice since the phase errors grow in proportion to T_{slot} and their *ratio* is simply a constant. Thus, here too, phase-only adjustments would suffice for small-to-moderate sized networks.

2.6.2 Phase & Frequency Adjustments

We begin by introducing parameters and performance metrics specific to the phase-frequency adjustment algorithm and then describe the results.

Simulation Parameters: We consider two types of skew distribution across

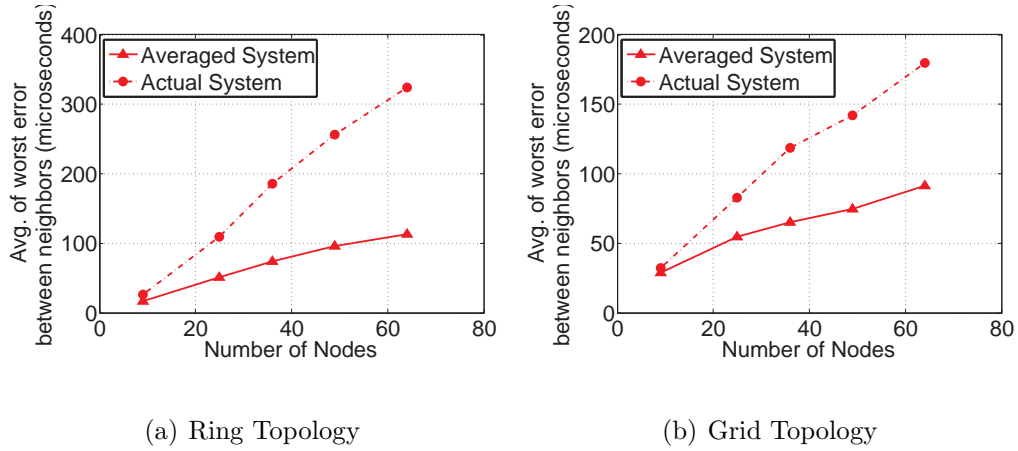


Figure 2.4: Worst error between neighbors for the actual system and the averaged system with only phase adjustments in an *omnidirectional network*. The network operates in the ONLYINTENDED mode.

nodes: (a) “random skews” chosen independently and uniformly in $[-50ppm, 50ppm]$ (b) “bad skews” chosen by maximizing the worst neighbor error for the averaged system. Nodes adjust their frequencies by $\mu = \pm 1ppm$ once a round, which consists of $S_R = 200$ slots. We investigate the performance with noiseless as well as noisy implicit timestamps. We model the noise as being uniformly distributed in $[-5ns, 5ns]$ for directional networks and in $[-5\mu s, 5\mu s]$ for omnidirectional networks. We use 30,000 TDM slots for each simulation run (and average over 50 runs). We choose the width of the dead zone in the noiseless setting ϵ_{approx} to be $3\mu = 3 ppm$.

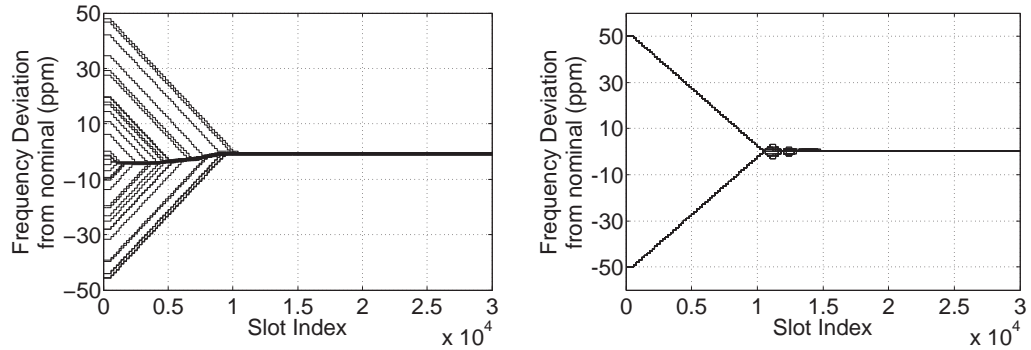
Performance Metrics: In addition to the worst phase error between neighbors, we also investigate the *network wide frequency error* as a measure of fre-

quency synchrony. The network wide frequency error is defined to be the maximum difference in frequency between any two nodes in the network. We now describe the typical evolution of node frequencies in a single trial and then provide the results for the phase and frequency errors averaged over 50 trials.

Evolution of node frequencies: We describe three scenarios that serve as exemplars for the evolution of frequencies at different nodes in a single simulation run. *Example 1:* We consider a 36 node omnidirectional network arranged in a grid topology. The skews at different nodes are distributed randomly and the implicit timestamps are noiseless. From Figure (2.5(a)), we see that nodes begin with varying frequencies and move in concert towards a common frequency. Every node's frequency is eventually confined to the band $f_{nom} \times [1 - 1.25 \text{ ppm}, 1 - 0.34 \text{ ppm}]$ where f_{nom} is the nominal frequency. Note that f_{nom} itself is not within this band. This shows that the nodes simply converge to a common frequency and not to the nominal frequency. Even in the noiseless scenario, the eventual frequency alignment is not perfect for two reasons: the adjustments have a resolution of ± 1 ppm and we have enforced a dead zone within which nodes do not change their frequencies. *Example 2:* A bad skew distribution best illustrates the fact that the nodes adjust their frequencies in concert. We consider such a skew distribution in a 36 node directional network with a ring topology. The implicit timestamps are noiseless in this example. We plot the frequencies of *all* nodes as a function

of time in Figure (2.5(b)). However, we see only two traces until about 10000 slots. This is because the nodes start off at identical frequencies of ± 50 ppm and make identical frequency adjustments of ∓ 1 ppm in each round. Consequently, at any time, the frequencies of all the nodes are confined to one of two values, $f_s(1 \pm k)$, $k \in \{1, 2, \dots, 50\}$ ppm. We also see that some nodes make unnecessary adjustments when the frequencies have almost converged (around 10000 slots). If the dead zone is not large enough, such unnecessary adjustments by a few nodes could drag the frequencies of the other nodes along. This might result in the nodes eventually having a common frequency that is outside the band where they began i.e $f_s \times [1 - \rho_{max}, 1 + \rho_{max}]$. This is not desirable in practice. In our simulations, we find that a dead zone of width 3μ suffices in the noiseless setting to prevent this from happening. A precise characterization dead zone width required to guarantee that the frequencies remain bounded within the original range, similar to the one provided for the averaged system, is left as an open issue.

Example 3: We now consider the same setting as in Example 1, except that the measurements are noisy. From Figure 2.6, we see that the steady march of the nodes towards convergence is eventually stalled by the measurement noise. The node frequencies fall within the dead zone and the nodes are unable to make a reliable decision on whether their frequencies are greater or lesser than the average. Thus, we have an eventual network-wide frequency error of 11.7 ppm in this



(a) Omnidirectional network, grid topology (b) Directional network, ring topology

Figure 2.5: Frequency deviations of all 36 nodes. Skews are *randomly* distributed and measurements are *noiseless*.

case. The size of the dead zone could potentially be reduced by waiting longer between frequency adjustments, thereby allowing the nodes to average the noise in the timestamps to a greater degree. However, this approach does not provide substantial gains because the phase error between nodes, in this regime, is dominated by erroneous phase adjustments rather than mildly disparate frequencies.

While we have limited our discussion here to three representative examples, we have carried out extensive simulations that show that our observations apply qualitatively to diverse combinations of node topologies, network directionality and measurement noise strength.

Mean Network Wide Frequency Error: We plot the network wide frequency error, averaged over 50 runs, as a function of time in Figures 2.7 - 2.10. We consider networks of 16/36/64 nodes arranged in a ring/grid topology with

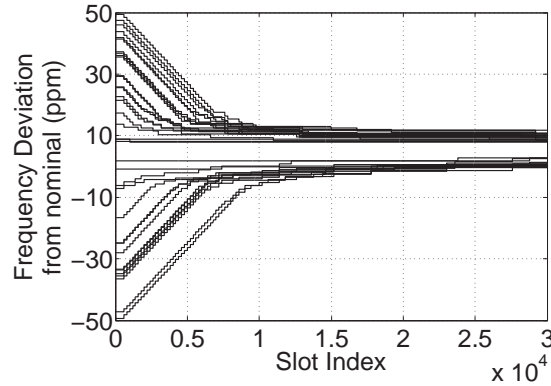


Figure 2.6: Frequency deviations of all 36 nodes in an *omnidirectional* network with a *grid* topology. Skews are *randomly* distributed and measurements are *noisy*.

noiseless and noisy implicit timestamps. We examine each scenario for omnidirectional as well as directional networks. Two things are common to these plots: (1) For about the first 10000 slots, the network wide frequency error drops by $2\mu = 2$ ppm each time a frequency adjustment is made. This drop is virtually independent of different parameters such as network size, topology, directionality and measurement noise. The steady fall occurs because there are nodes with frequencies far above and below the mean at the beginning; these nodes make the right adjustment every time (see Proposition 5). (2) We observe that the number of rounds of frequency adjustment that contribute to a decrease in the network wide frequency error is also virtually independent of parameters such as network size, topology, directionality and measurement noise. It depends only on

the network wide frequency error at the start and the adjustment step size. Since the network wide error drops by 2 ppm in each round, the number of rounds of frequency adjustment required is roughly $100 \text{ ppm} / 2 \text{ ppm} = 50$.

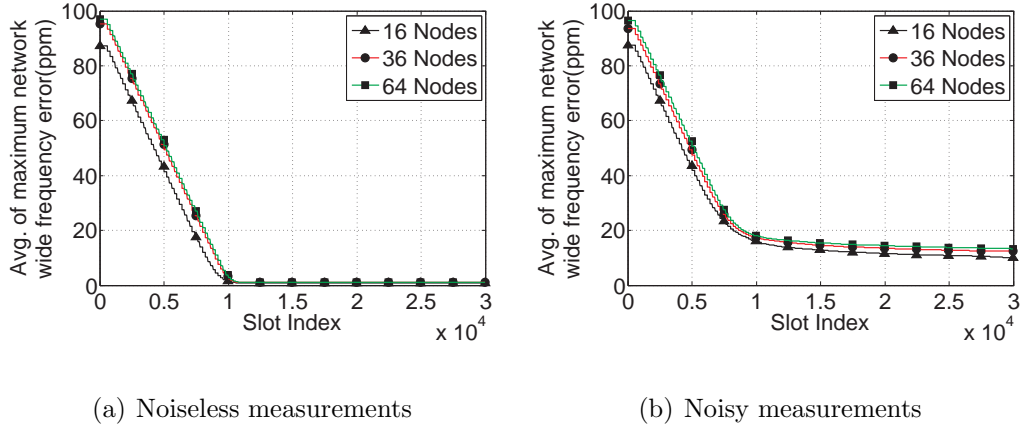


Figure 2.7: Network wide frequency error in a *directional* setting with a *grid* topology. Skews are distributed *randomly*.

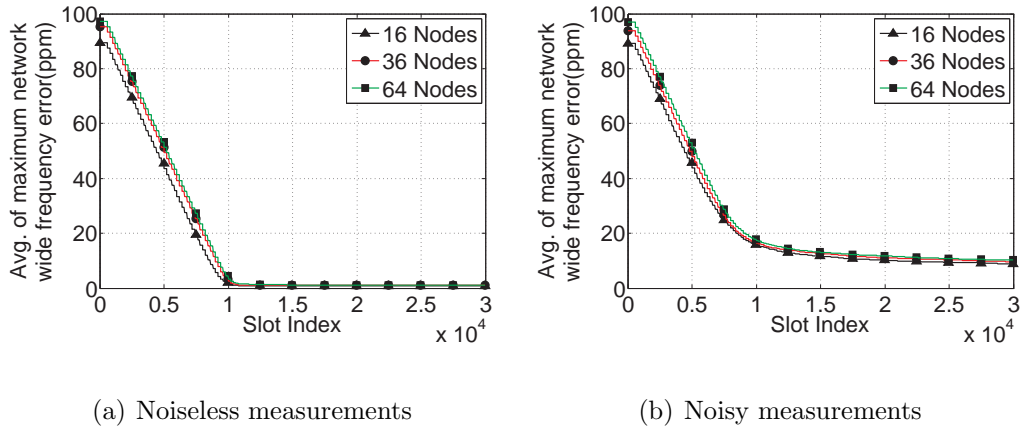


Figure 2.8: Network wide frequency error in a *directional* setting with a *ring* topology. Skews are *randomly* distributed.

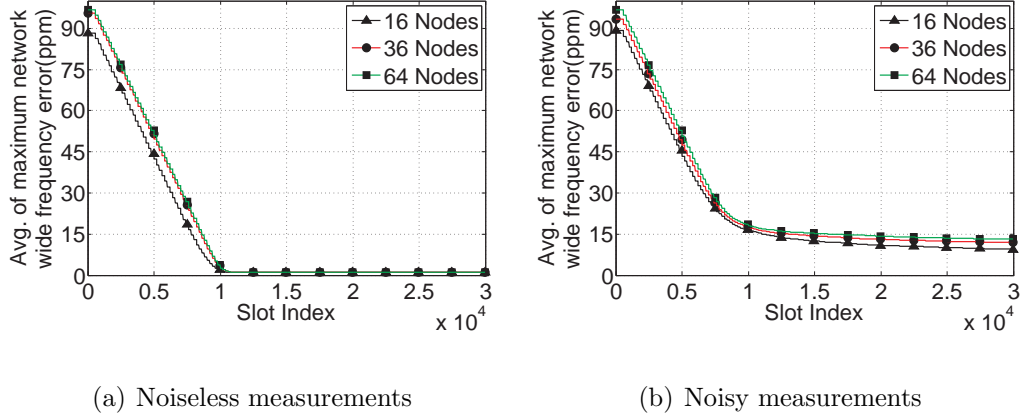


Figure 2.9: Network wide frequency error in an *omnidirectional* setting with a *grid* topology. Skews are *randomly* distributed.

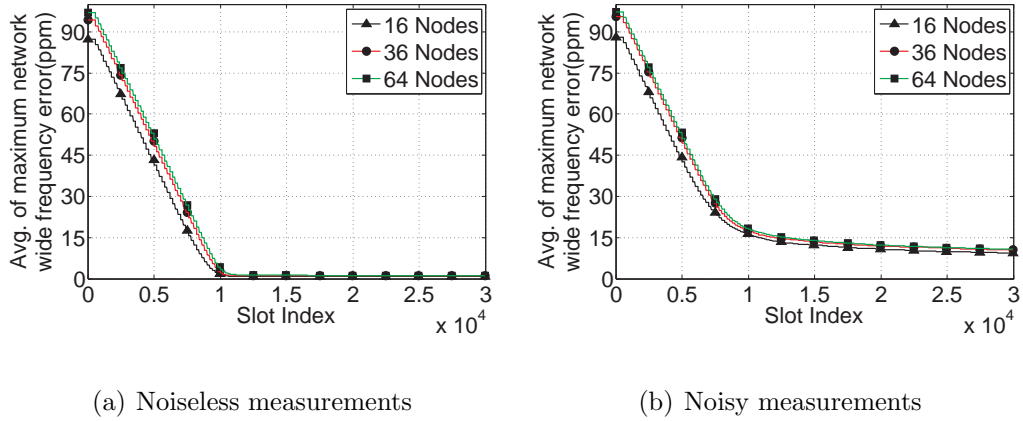


Figure 2.10: Network wide frequency error in an *omnidirectional* setting with a *ring* topology. Skews are *randomly* distributed.

An important metric is the eventual network wide frequency error. For the noiseless setting (Figures 2.7(a), 2.8(a), 2.9(a), 2.10(a)) the network wide frequency error settles to a value on the order of the adjustment step-size $\mu = 1$ ppm. This value is almost independent of the network topology and the directionality; for

all combinations of the topology and directionality, the error is confined to 0.9-1.2 ppm. When the implicit timestamps are noisy, the eventual network wide frequency error depends mildly on the size of the network and the directionality (Figures 2.7(b),2.8(b),2.9(b),2.10(b)). For example, in directional networks with a ring topology, the eventual network wide frequency error is about 8.9 ppm with 16 nodes and it increases to 10.2 ppm with 64 nodes; the corresponding numbers for an omnidirectional setting are 9.4 ppm and 10.9 ppm respectively. The eventual frequency synchronization error is also a little higher with the grid topology: for example, in a directional setting, the error is about 10.1 ppm with 16 nodes and 13.3 ppm with 64 nodes. In all these cases, we see that there is a ten-fold decrease in the network wide frequency error - from 100 ppm to about 10 ppm (roughly).

Worst Phase Error Between Neighbors: We now investigate the largest error in clock phase between any two nodes that are neighbors. We split our discussion into two cases: in the first case, the skews are distributed randomly and in the second case, they are distributed badly.

Case A - Random Distribution of Skews: We choose the node skews randomly and plot the worst phase error for directional networks in Figure 2.11. We also plot the worst phase error for omnidirectional networks operating in the ONLYINTENDED mode and EAVESDROP mode in Figures 2.12 and 2.13 respectively. Note that all the plots have noisy implicit timestamp measurements. From these plots,

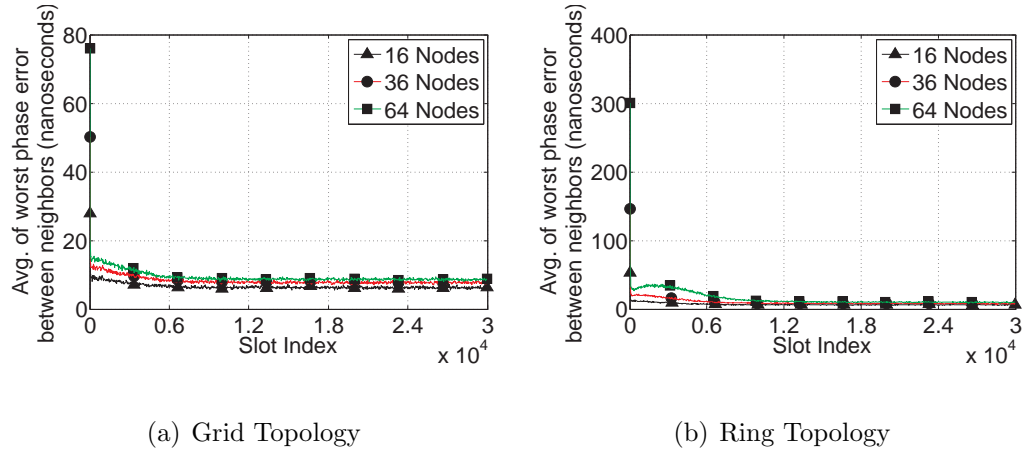


Figure 2.11: Worst phase error between neighbors in a *directional* setting with *noisy* measurements. Skews are *randomly* distributed.

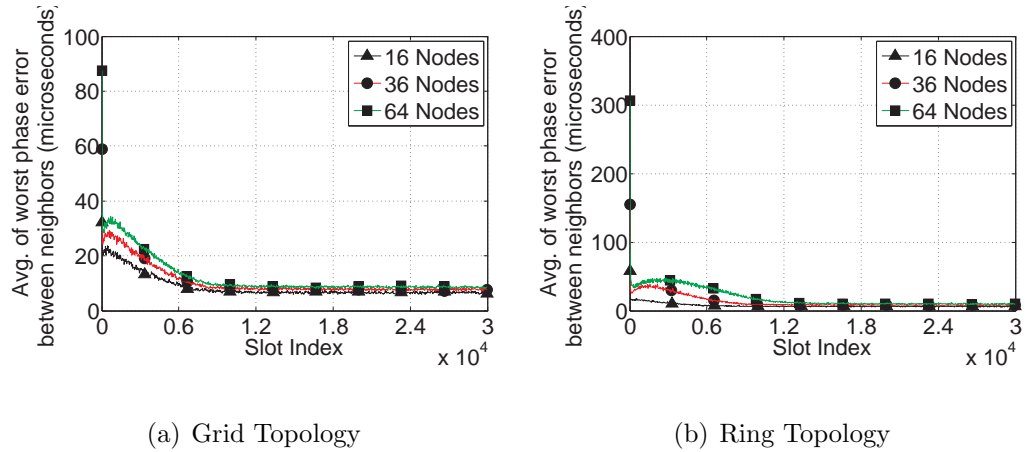


Figure 2.12: Worst error between neighbors in an *omnidirectional* setting with *randomly* distributed skews. Measurements are *noisy* and nodes are in the ONLYINTENDED mode.

we see that the phase errors fall “smoothly” from their values at startup - after the coarse synchronization procedure - to their steady-state values. The eventual phase error increases mildly with network size and changes very little with the

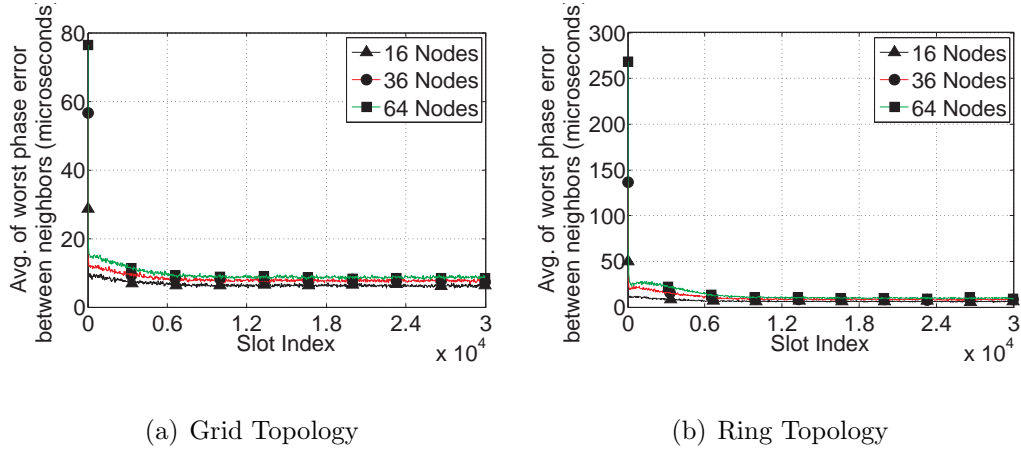


Figure 2.13: Worst error between neighbors in an *omnidirectional* setting with *randomly* distributed skews. Measurements are *noisy* and nodes are in the EAVES-DROP mode.

topology. For example, in directional networks with a ring topology, the eventual phase error only increases from 6.25 ns to 10.7 ns when the network size increases from 16 nodes to 64 nodes. The corresponding numbers for the grid topology are very similar: 6.49 ns (16 nodes) and 9.05 ns (64 nodes). Similarly, the errors in omnidirectional networks with ring and grid topologies are close; they increase from 6.5 μ s for 16 node networks to about 10 μ s for 64 node networks.

Case B - Badly distributed skews: We plot the largest phase error with badly distributed skews for directional networks in Figure 2.14 and omnidirectional networks in Figures 2.15 and 2.16 respectively.

These curves have a conspicuous “hump” at the start - particularly the ones corresponding to the ring topology, Figures 2.14(b), 2.16(a), 2.16(b)) - where the

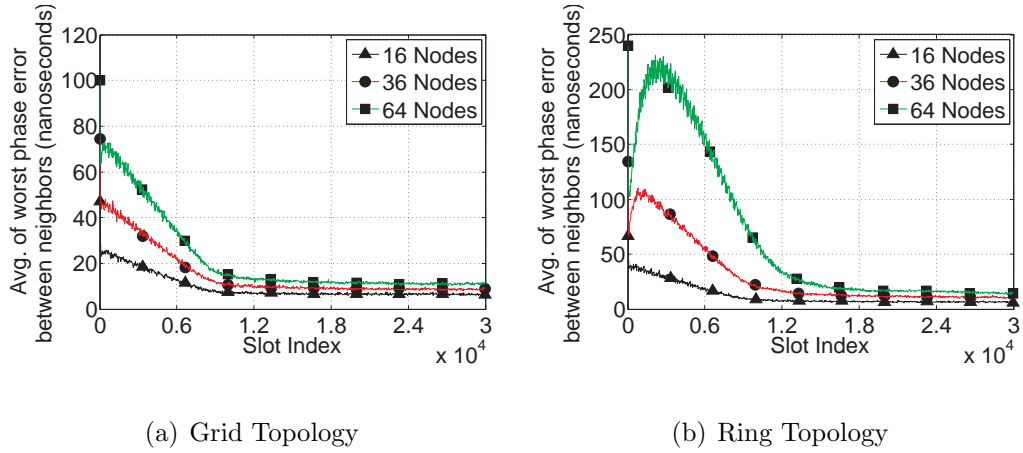


Figure 2.14: Worst phase error between neighbors in a *directional* setting. Skews are *badly* distributed and measurements are *noisy*.

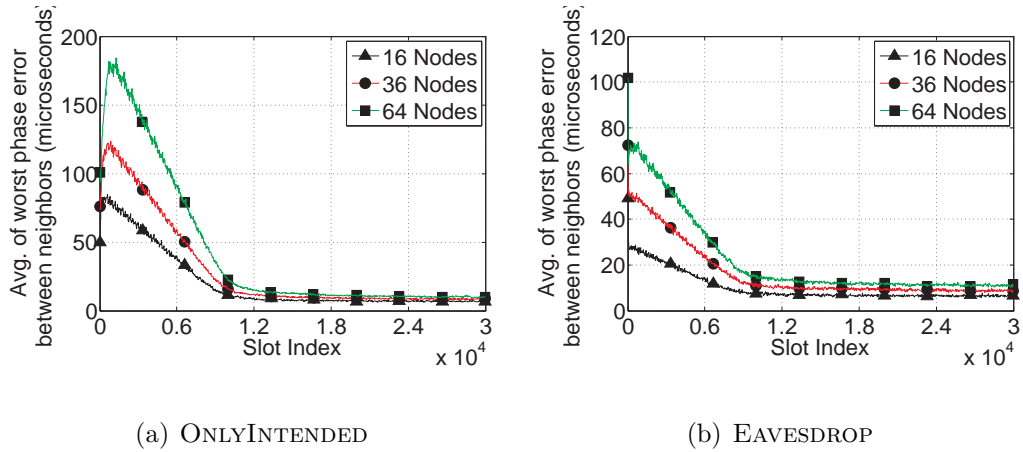


Figure 2.15: Worst phase error between neighbors in an *omnidirectional* setting with a *grid* topology. Skews are *badly* distributed, measurements are *noisy*.

phase error rises before falling. This occurs because the round sizes (and hence spacing between frequency updates) required are large, which gives phase errors a chance to grow before they can be reduced with frequency updates. We now compare the size of the hump - which is the peak value of the phase error between

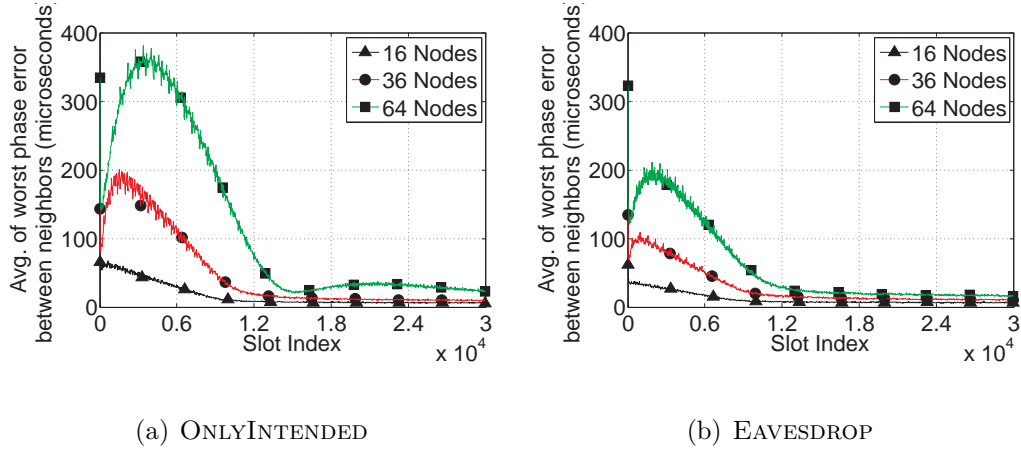


Figure 2.16: Worst phase error between neighbors in an *omnidirectional* setting with a *ring* topology. Skews are *badly* distributed, measurements are *noisy*.

neighbors - for differing network sizes, topologies and modes of operation.

Firstly, by comparing the plots for grid and ring topologies (for example, Figure 2.14(a) vs. 2.14(b) [or] Figure 2.15(a) vs. 2.16(a)), we observe that the hump is much larger with the ring topology. For example, in the ONLYINTENDED mode in omnidirectional networks with 64 nodes, the size of the hump is $360\mu s$ in the ring topology as compared to $175\mu s$ in the grid topology. This is consistent with earlier results from phase-only adjustments, where we found that the errors scale much faster with the number of nodes for a ring topology as compared to the grid topology. In all these figures, we see that the size of the hump increases with the number of nodes in the network. We can summarize both these facts by concluding that the size of the hump increases with the network diameter. Next, we compare omnidirectional networks operating in the EAVESDROP and

ONLYINTENDED modes. Comparing Figures 2.16(a) and 2.16(b), we see that the size of the hump in an omnidirectional setting with a ring topology is almost twice as large in the ONLYINTENDED mode as compared to the EAVESDROP mode. For example, with a 64 node ring network, the size of the hump is about $360\mu s$ in the ONLYINTENDED mode; but, it is only about $200\mu s$ in the EAVESDROP mode. This is a clear illustration of the benefits of eavesdropping.

Chapter 3

Space-time localization using times of arrival

We investigate the problem of localizing multiple events that occur in quick succession based on their Times of Arrival (ToAs) at different sensors. We start with a quick recap of the main features of the problem:

- The order in which the events are heard at any given sensor need not be the same as the order in which they occur.
- The naïve strategy of sorting the ToAs in ascending order and using the i th ToA at each sensor to localize the i th event will fail. Considering all combinations of ToAs and retaining only the “good” combinations is excessively complex and does not guarantee correct localization either. Furthermore, these methods are fragile and break easily when the sensing process is non-ideal (sensors miss events and record outlier ToAs).

In this chapter, we present feasibility results, an example of ambiguity that arises due to the presence of multiple events and a robust, low-complexity algorithm to localize the events assuming that we have deployed “enough” sensors.

Map of this chapter: We begin by describing the related literature on source localization in Section 3.1 and then explain the system model in Section 3.2. In Section 3.3, we explore the fundamental limits of localizing multiple events from their ToAs when the sensing process is ideal – there are no misses or outliers at any of the sensors and the ToA measurements are noiseless. First, we show that when the temporal separation between the events is larger than the propagation delay corresponding to the diameter of the deployment region, the ordering of the events is preserved in the observed ToAs at each sensor. In such cases, the naïve strategy of sorting ToAs will work. Then, we show that we can localize two events perfectly if we deploy enough sensors and that there might be fundamental ambiguities in localization if we do not deploy enough sensors. Specifically, we show that nine sensors are sufficient to localize two events (as long as they do not lie on a branch of a hyperbola) and we construct an example to show that six sensors do not suffice to guarantee perfect localization. After establishing these fundamental limits, we propose and describe a three-stage algorithm to localize multiple events from their ToAs in the presence of outliers and misses. We present the key ideas behind the algorithm in Section 3.4. The next three

sections present the mathematical details behind each of the stages. In Section 3.5, we thin down the set of candidate events considerably by discretizing the times at which events occur. This lets us generate hypothesized event locations by intersecting circles drawn at pairs of sensors. However, some of these events are “phantoms” (corresponding to events that did not occur, but are on the list of candidates) and others are “duplicates” (since every pair of sensors we consider generates a candidate close to the location where an event actually occurred). We present a clustering algorithm to merge these duplicates and at the end of this stage, we are left with a palette of events, consisting of true and phantom candidates. In Section 3.6, we use measurements at all the sensors to refine these candidates in an iterative fashion, bootstrapping with the estimates from Section 3.5. At the end of the refining process, the palette has two properties: (1) it contains much fewer candidates than what a naïve discretization of space and time would produce and (2) the estimates of the actual events are very close to their true locations because of the refinements in Section 3.6. Thus, we can now solve the problem of rejecting the phantoms, picking the events that occurred and associating the events to the observed ToAs with relatively low-complexity. We do this in two steps: (a) first, we show that the association problem is a variant of the matching problem on a graph and that it can be formulated as an integer program. (b) We then relax the integer program to solve a linear program

and further reduce the complexity. Finally, we show via extensive simulations in Section 3.8 that the proposed algorithm is effective in estimating the number of events as well as their locations and times.

3.1 Related Work

There is a vast literature on the general problem of source localization, including algorithms using ToAs [49], AoAs [15], Time Differences of Arrival (TDoAs) [7], [4],[52], hybrid versions of these (hybrid TDoA-AoA) [5] and wideband processing of recorded signals [8]. The survey paper [24] provides a more exhaustive set of references for each of these techniques. Most such prior work considers one event at a time, and hence ignores the association problem central to this paper. A notable exception is [40], which does consider localization of multiple events using ToA sensors. The basic idea is to discretize the space-time grid, count how many sensors “agree” with a given grid point, and estimate event locations and times as local maxima of this count. The complexity of this approach grows with the size of the deployment region and the desired granularity of the space-time estimates. In contrast, our algorithm only discretizes the times at which events can occur, so that its complexity is independent of the size of the deployment region. Further, the algorithm in [40] does not explicitly account for the constraints associated

with the problem (for example, every ToA must either be associated with a single event or declared to be an outlier), unlike the linear program based matching in our algorithm. Note that [40] considers three-dimensional localization, while we restrict attention to two spatial dimensions for simplicity. However, our approach generalizes in a straightforward manner to three spatial dimensions by intersecting spheres instead of circles, using triplets rather than pairs of sensors in the first stage. The second and third stages would remain unchanged.

In terms of feasibility of localization, the number of sensors and conditions on the sensor configuration for perfect event localization from TDoA measurements are characterized in [53]. However, these results are for a single event, and do not address the space-time localization problem considered here.

It is interesting to note that the inspiration for our algorithm comes from the center-surround neural response characteristic of mammalian vision [19]. The complex scenes that we perceive are obtained by intersecting such responses and using feedback from higher layers, which is similar in spirit to our algorithm.

3.2 System Model

We consider N sensors deployed at locations $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N$ within a two-dimensional region \mathcal{D} that we wish to monitor. We observe the system over the

time window $[0, T]$. An unknown number of events, E , occur during this period. The e th event is described by the triplet $(\boldsymbol{\alpha}_e, t_e)$ where $\boldsymbol{\alpha}_e \in \mathbb{R}^2$ is the spatial location and t_e the time of occurrence of the event. For any sensor s , the event e is missed with probability p_{miss} , and, with probability $1 - p_{miss}$, produces the following noisy ToA reading:

$$\tau(e \leftrightarrow s) = t_e + \frac{\|\boldsymbol{\alpha}_e - \boldsymbol{\theta}_s\|}{c} + n \quad (3.1)$$

where c denotes the speed of propagation, $\|\cdot\|$ denotes the two-norm of a vector and n is the measurement noise, assumed to be distributed as $N(0, \sigma^2)$. Misses and measurement noise are assumed to be independent across events and sensors. We simplify notation by choosing our units to set $c = 1$, so that the measurement model in (3.1) simplifies to

$$\tau(e \leftrightarrow s) = t_e + \|\boldsymbol{\alpha}_e - \boldsymbol{\theta}_s\| + n. \quad (3.2)$$

For our statistical processing, we model event occurrence as a space-time Poisson process, with events occurring at rate λ_{LS} per unit time, with locations uniformly distributed over \mathcal{D} .

Outliers: Outliers typically result from “small-scale” events, typically heard at only one sensor (e.g., a nearby slamming car door may trigger an acoustic sensor deployed for detecting far-away explosions), which we are therefore unable to, and not interested in, localizing. We do not model the locations of such events and

model their ToAs as arising from a Poisson process with a rate λ_O (per unit time) at each sensor. These processes are assumed to be independent across sensors.

Sensor Observations: Suppose that sensor s records M_s ToAs due to events and outliers in the time window $[0, T]$. We denote the i th ToA at sensor s by $\tau_s(i)$, where the ToAs are sorted in ascending order at each sensor. Therefore, the set of observations at sensor s is given by $\Omega_s = \{\tau_s(1), \tau_s(2), \dots, \tau_s(M_s)\}$, with $\tau_s(i) \leq \tau_s(j)$ whenever $i \leq j$. The number of ToAs can vary across sensors because misses and outliers occur independently at each sensor.

3.3 Feasibility of localizing multiple events

In this section, we investigate the feasibility of localizing multiple events under the most ideal of conditions: no misses, no outliers, no noise. We first ask when a standard single event localization algorithm can be used as a building block for multiple event localization using a naïve approach that solves the association problem using sorting: the ToAs at each sensor are arranged in ascending order, the i th largest ToA at each sensor is associated with the i th event, followed by the use of a standard single event localization algorithm to localize each event. Clearly, this approach works when the order of ToAs at each sensor is the same as

the order of occurrence of the events, which happens if the events are separated “enough” in time. The following theorem provides a precise characterization of this observation.

Theorem 2. *The naïve approach will localize all events perfectly if the time interval between any two events is larger than the diameter of the deployment region D . (Note: Units of space and time are chosen so that the propagation speed $c = 1$. For arbitrary units, the temporal separation must be D/c .)*

Proof. The naïve algorithm will localize all events correctly if the order in which the events arrive at all sensors is the same as the order in which they occur. The i th grouping will then consist of the ToAs produced by the i th earliest event and, since the measurements are noiseless, the conventional algorithm will localize the events perfectly. We now show that the ordering of events is preserved in the ToAs they produce, if every pair of events is temporally separated by at least the diameter of the deployment region D .

Suppose that two events $\mathcal{E}_a = (\boldsymbol{\alpha}_a, t_a)$ and $\mathcal{E}_b = (\boldsymbol{\alpha}_b, t_b)$ produce ToAs τ_a and τ_b at a sensor located at $\boldsymbol{\theta}$. Without loss of generality, suppose that \mathcal{E}_a occurs after \mathcal{E}_b ($t_a \geq t_b$). Using the defining equation for the ToAs (3.2) and the fact that the measurements are noiseless, we get

$$\tau_a - \tau_b = t_a - t_b + (||\boldsymbol{\alpha}_a - \boldsymbol{\theta}|| - ||\boldsymbol{\alpha}_b - \boldsymbol{\theta}||). \quad (3.3)$$

Since the sensors and the events are located within the deployment region, we have $0 \leq \|\alpha_a - \theta\| \leq D$ and $0 \leq \|\alpha_b - \theta\| \leq D$ where D is the diameter of the deployment region. Therefore, the term within brackets in (3.3) can be bounded as

$$-D \leq \|\alpha_a - \theta\| - \|\alpha_b - \theta\| \leq D. \quad (3.4)$$

Since the events are well separated in time, we also have $t_a \geq t_b + D$. Using this fact and the inequality from (3.4) in (3.3), we get

$$\tau_a - \tau_b \geq D - D = 0. \quad (3.5)$$

Thus, $\tau_a \geq \tau_b$ and the order in which events arrive at any sensor is the same as the order in which they occur. \square

Example: Consider a network of acoustic sensors deployed over a circular region of radius 1 km. Since the speed of sound is 340 m/s, the naïve approach suffices to localize events that are separated by roughly 6 seconds. However, if the time between events is smaller than 6 seconds, we need more sophisticated approaches to the association problem.

Even when events are closely spaced in time, it is intuitively plausible that we can localize these events correctly if we deploy “enough” sensors. We now show that 9 sensors suffice to localize two events (with arbitrarily small separation in space and time, for noiseless observations), if the placement of the sensors is not

degenerate. Specifically, all sensors should not lie on one branch of a hyperbola. We briefly review the terminology associated with hyperbolas, introduce and define the term *half-hyperbola* and then prove the required result.

A point $\boldsymbol{\theta} \in \mathbb{R}^2$ lies on a hyperbola with foci $\mathbf{f}_1, \mathbf{f}_2 \in \mathbb{R}^2$ and major axis of length $|a| (a \in \mathbb{R})$ if

$$\left| \|\boldsymbol{\theta} - \mathbf{f}_1\| - \|\boldsymbol{\theta} - \mathbf{f}_2\| \right| = |a|. \quad (3.6)$$

A hyperbola consists of two branches – the first branch contains the points that lie on $\|\boldsymbol{\theta} - \mathbf{f}_1\| - \|\boldsymbol{\theta} - \mathbf{f}_2\| = |a|$ and the other branch contains the points that lie on $\|\boldsymbol{\theta} - \mathbf{f}_1\| - \|\boldsymbol{\theta} - \mathbf{f}_2\| = -|a|$. We use the term *half-hyperbola* to refer to a curve which is one of the branches of a hyperbola, defined as follows:

Definition A set of points Θ in the two-dimensional plane are said to lie on a half-hyperbola if there exist $\mathbf{f}_1, \mathbf{f}_2 \in \mathbb{R}^2$ and $a \in \mathbb{R}$, so that,

$$\|\boldsymbol{\theta} - \mathbf{f}_1\| - \|\boldsymbol{\theta} - \mathbf{f}_2\| = a \quad \forall \boldsymbol{\theta} \in \Theta. \quad (3.7)$$

We are now ready to state our feasibility result.

Theorem 3. *Suppose that two events $\mathcal{E}_1 = (\boldsymbol{\alpha}_1, t_1)$ and $\mathcal{E}_2 = (\boldsymbol{\alpha}_2, t_2)$ produce ToAs at each one of N sensors located at $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \dots, \boldsymbol{\theta}_N$. Then, we can guarantee perfect localization of the events, if the number of sensors $N \geq 9$ and the sensors do not lie on a half-hyperbola.*

Proof. We prove the result by contradiction. First, we suppose that an alternate set of events that explains the ToAs at all the sensors exists, preventing us from localizing the events correctly. We then show that the existence of such an alternate explanation violates one of the conditions in the theorem, thereby proving the required result.

For our ideal observation model (no misses, no outliers, exactly two ToAs at each sensor), we can have an imperfect reconstruction only if there is an alternate set of events $\mathcal{E}_a = (\alpha_a, t_a)$ and $\mathcal{E}_b = (\alpha_b, t_b)$ which also explain the recorded ToAs. For the explanation set $\{\mathcal{E}_a, \mathcal{E}_b\}$ to be different from the set of events that produced the ToAs $\{\mathcal{E}_1, \mathcal{E}_2\}$, these sets must differ in at least one event. This can happen in one of two ways:

Case a: Neither of the events $\{\mathcal{E}_a, \mathcal{E}_b\}$ are the same as either of the events $\{\mathcal{E}_1, \mathcal{E}_2\}$.

Case b: One of the events is the same but the other event is different. For example, we might have $\mathcal{E}_a = \mathcal{E}_1$ (meaning, $\alpha_a = \alpha_1$ and $t_a = t_1$), but $\mathcal{E}_b \neq \mathcal{E}_2$ (meaning $\alpha_b \neq \alpha_2$, $t_b \neq t_2$, or both).

We analyze these cases separately.

Case a: All events are different – Since the explanation $\{\mathcal{E}_a, \mathcal{E}_b\}$ produces the same *set* of ToAs as $\{\mathcal{E}_1, \mathcal{E}_2\}$, *exactly* one of the following conditions must be true at each sensor:

- The ToA corresponding to \mathcal{E}_a is equal to the one produced by \mathcal{E}_1 and the ToA corresponding to \mathcal{E}_b is equal to that produced by \mathcal{E}_2 .
- The ToA corresponding to \mathcal{E}_a is equal to the one produced by \mathcal{E}_2 and the ToA corresponding to \mathcal{E}_b is equal to that produced by \mathcal{E}_1 .

Let \mathcal{W}_1 denote the subset of sensors which satisfy the first condition and \mathcal{W}_2 denote the subset that satisfy the second condition. Then, the total number of sensors $N = |\mathcal{W}_1| + |\mathcal{W}_2|$ where $|\mathcal{A}|$ denotes the cardinality of the set \mathcal{A} . We now show that $|\mathcal{W}_1| \leq 4$.

Let $\boldsymbol{\theta}$ denote the location of any sensor in \mathcal{W}_1 . From the first condition, we see that the location $\boldsymbol{\theta}$ must satisfy both of the following equations:

$$t_a + \|\boldsymbol{\alpha}_a - \boldsymbol{\theta}\| = t_1 + \|\boldsymbol{\alpha}_1 - \boldsymbol{\theta}\| \quad \Rightarrow \quad \|\boldsymbol{\theta} - \boldsymbol{\alpha}_a\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_1\| = t_1 - t_a \quad (3.8)$$

$$t_b + \|\boldsymbol{\alpha}_b - \boldsymbol{\theta}\| = t_2 + \|\boldsymbol{\alpha}_2 - \boldsymbol{\theta}\| \quad \Rightarrow \quad \|\boldsymbol{\theta} - \boldsymbol{\alpha}_b\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_2\| = t_2 - t_b. \quad (3.9)$$

Equations (3.8) and (3.9) each describe a half-hyperbola. Furthermore, since the events are all different, neither of these equations are trivial. Therefore, any sensor in \mathcal{W}_1 must be located at the intersection of two half-hyperbolas. The number of points of intersection of two half-hyperbolas is upper bounded by the number of points in which their “parent” hyperbolas (obtained by including the other branch of each half-hyperbola) intersect. By Bezout’s Theorem [53], two

hyperbolas intersect in at most 4 points. Thus, we have $|\mathcal{W}_1| \leq 4$.

By a similar argument, we can show that $|\mathcal{W}_2| \leq 4$. Thus, to construct two events $\{\mathcal{E}_a, \mathcal{E}_b\}$ so that: (i) they explain the ToAs produced by \mathcal{E}_1 and \mathcal{E}_2 at all the sensors and (ii) neither \mathcal{E}_a nor \mathcal{E}_b is the same as either of the events \mathcal{E}_1 or \mathcal{E}_2 , we need the number of sensors N to satisfy,

$$N = |\mathcal{W}_1| + |\mathcal{W}_2| \leq 4 + 4 = 8 \quad (3.10)$$

Case b: One of the events is same – Suppose now that $\mathcal{E}_a = \mathcal{E}_1$ but $\mathcal{E}_b \neq \mathcal{E}_2$.

Then, (3.8) is trivially true and all we can say about sensors in \mathcal{W}_1 (\mathcal{W}_1 and \mathcal{W}_2 have the same definition as in Case (a)) is that their location $\boldsymbol{\theta}$ must satisfy (3.9).

We now show that the sensors in \mathcal{W}_2 will also lie on this half-hyperbola.

Let $\boldsymbol{\theta}$ denote the location of any sensor in \mathcal{W}_2 . Using the fact that $\mathcal{E}_a = \mathcal{E}_1$ and writing out the conditions that a sensor in \mathcal{W}_2 must satisfy, we have

$$t_1 + \|\boldsymbol{\alpha}_1 - \boldsymbol{\theta}\| = t_2 + \|\boldsymbol{\alpha}_2 - \boldsymbol{\theta}\| \quad \Rightarrow \quad \|\boldsymbol{\theta} - \boldsymbol{\alpha}_2\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_1\| = t_1 - t_2 \quad (3.11)$$

$$t_b + \|\boldsymbol{\alpha}_b - \boldsymbol{\theta}\| = t_1 + \|\boldsymbol{\alpha}_1 - \boldsymbol{\theta}\| \quad \Rightarrow \quad \|\boldsymbol{\theta} - \boldsymbol{\alpha}_b\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_1\| = t_1 - t_b. \quad (3.12)$$

Subtracting (3.11) from (3.12), we see that the location $\boldsymbol{\theta}$ of a sensor in \mathcal{W}_2 must satisfy (3.9). Thus, if we construct an alternate explanation for the recorded data which differs in only one of the two events, then all sensors must lie on a half-hyperbola.

Therefore, if we can find two sets of events $\{\mathcal{E}_1, \mathcal{E}_2\}$ and $\{\mathcal{E}_a, \mathcal{E}_b\}$ that explain the recorded ToAs at N sensors, then either (i) $N \leq 8$ or (ii) all the sensors lie on a half-hyperbola. The true events $\{\mathcal{E}_1, \mathcal{E}_2\}$ obviously explain the recorded data. Since we are given that $N \geq 9$ and we cannot draw a half-hyperbola through all the sensors, an alternate explanation of the data such as $\{\mathcal{E}_a, \mathcal{E}_b\}$ cannot exist (since such an explanation would violate either (i) or (ii)). Consequently, the only explanation for the ToAs are the events $\{\mathcal{E}_1, \mathcal{E}_2\}$ themselves, thereby guaranteeing perfect localization.

□

Remark: If all the sensors were to lie on a half-hyperbola, then we cannot guarantee localizing even a single event (an event happening at one focus of the half-hyperbola could produce the same ToAs as an event happening at the other focus provided their times are chosen appropriately [53]). Thus, the only *additional* requirement in localizing two events is that the number of sensors $N \geq 9$. Note that degenerate placement of sensors on a half-hyperbola is a zero probability event for random sensor deployment, and is not possible for regular grid-like deployments.

The immediate question that arises is: can we give an example where two distinct event sets $\{\mathcal{E}_1, \mathcal{E}_2\}$ and $\{\mathcal{E}_a, \mathcal{E}_b\}$ produce the same ToAs at $N = 8$ sensors

(which do not lie on a half-hyperbola), so that these sets cannot be disambiguated?

While we are not able to answer this question conclusively, we use insights from the analysis in Case (a) to construct an example where 6 sensors are unable to distinguish between the event sets $\{\mathcal{E}_1, \mathcal{E}_2\}$ and $\{\mathcal{E}_a, \mathcal{E}_b\}$.

Example: We use two key ideas in constructing this example:

- We choose the event locations $\alpha_1, \alpha_2, \alpha_a$ and α_b and the parameters $t_1 - t_a$ and $t_2 - t_b$ so that the half-hyperbolas in (3.8) and (3.9) intersect in four points. This gives us four sensors in \mathcal{W}_1 which cannot disambiguate between the event sets $\{\mathcal{E}_1, \mathcal{E}_2\}$ and $\{\mathcal{E}_a, \mathcal{E}_b\}$.
- Having made these choices, we show that there is only one free parameter that fixes the two defining half-hyperbolas for any sensor in \mathcal{W}_2 . We choose this parameter to ensure that these half-hyperbolas intersect in two points.

The example is shown in Figure 3.1 and we now provide the details of the construction.

We denote the half-hyperbolas that define the set \mathcal{W}_1 by HH_1 (3.8) and HH_2 (3.9) respectively. First, we choose $\alpha_1 = (1, 0)$, $\alpha_a = (-1, 0)$ and $t_1 - t_a = 1.3$ so that HH_1 opens out towards the right and its axis coincides with the x-axis. This is the blue curve marked HH_1 in Figure 3.1. We now choose α_2 and α_b so that HH_2 's axis points *towards* the axis of HH_1 , thereby tending to increase

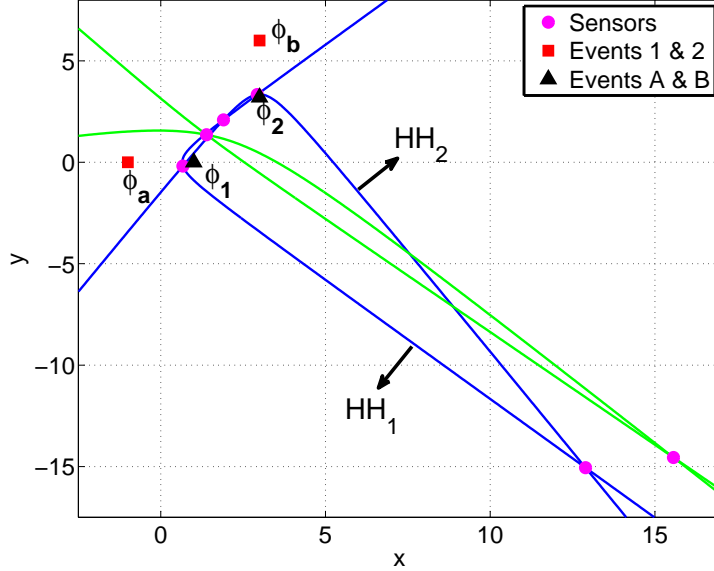


Figure 3.1: Each one of the sensors shown by the pink dots record two ToAs – one from Event 1 and the other from Event 2, whose locations are shown by the black triangles. However, events A and B, shown by the red squares, also produce the same set of ToAs at all the sensors. Therefore, the sensors are unable to decide which of the event sets $\{\mathcal{E}_a, \mathcal{E}_b\}$ and $\{\mathcal{E}_1, \mathcal{E}_2\}$ occurred.

the number of points in which HH_1 and HH_2 intersect. Specifically, we choose $\alpha_2 = (3, 3.2)$ and $\alpha_b = (3, 6)$ so that axes are at right-angles to one another.

Next, we pick $t_2 - t_b = 2.5$ so that HH_2 opens downwards and intersects HH_1 in four points. The four sensors located at these points of intersection cannot disambiguate between the event sets $\{\mathcal{E}_1, \mathcal{E}_2\}$ and $\{\mathcal{E}_a, \mathcal{E}_b\}$. We now identify possible sensor locations in the set \mathcal{W}_2 . By the definition of the set \mathcal{W}_2 , the location θ of

any sensor in \mathcal{W}_2 must satisfy both the equations

$$\|\boldsymbol{\theta} - \boldsymbol{\alpha}_2\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_a\| = t_a - t_2 \quad (3.13)$$

$$\|\boldsymbol{\theta} - \boldsymbol{\alpha}_1\| - \|\boldsymbol{\theta} - \boldsymbol{\alpha}_b\| = t_b - t_1. \quad (3.14)$$

Since we have already chosen $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_a$ and $\boldsymbol{\alpha}_b$, we only need to specify $t_b - t_1$ and $t_a - t_2$ in these equations to define the set \mathcal{W}_2 . However, we cannot choose $t_b - t_1$ and $t_a - t_2$ independently because we have already set $t_1 - t_a = 1.3$ and $t_2 - t_b = 2.5$ and these four quantities add up to zero. Therefore, we choose $t_a - t_2$, and set $t_b - t_1$ appropriately, so that the half-hyperbolas in (3.13) and (3.14) intersect in two points. For the example shown in Figure 3.1, we used $t_a - t_2 = -0.3$. These equations can be solved numerically to obtain the sensor locations. The six sensors shown in Figure 3.1 cannot distinguish between the event pairs:

$$\text{Event A: } \boldsymbol{\alpha}_a = (-1, 0), t_a = 0 \quad \text{Event B: } \boldsymbol{\alpha}_b = (3, 6), t_b = -2.2 \quad (3.15)$$

$$\text{Event 1: } \boldsymbol{\alpha}_1 = (1, 0), t_1 = 1.3 \quad \text{Event 2: } \boldsymbol{\alpha}_2 = (3, 3.2), t_2 = 0.3 \quad (3.16)$$

3.4 Algorithm Overview

We now turn our attention to the problem of designing a low-complexity algorithm to localize multiple events which are closely spaced in time, in a manner

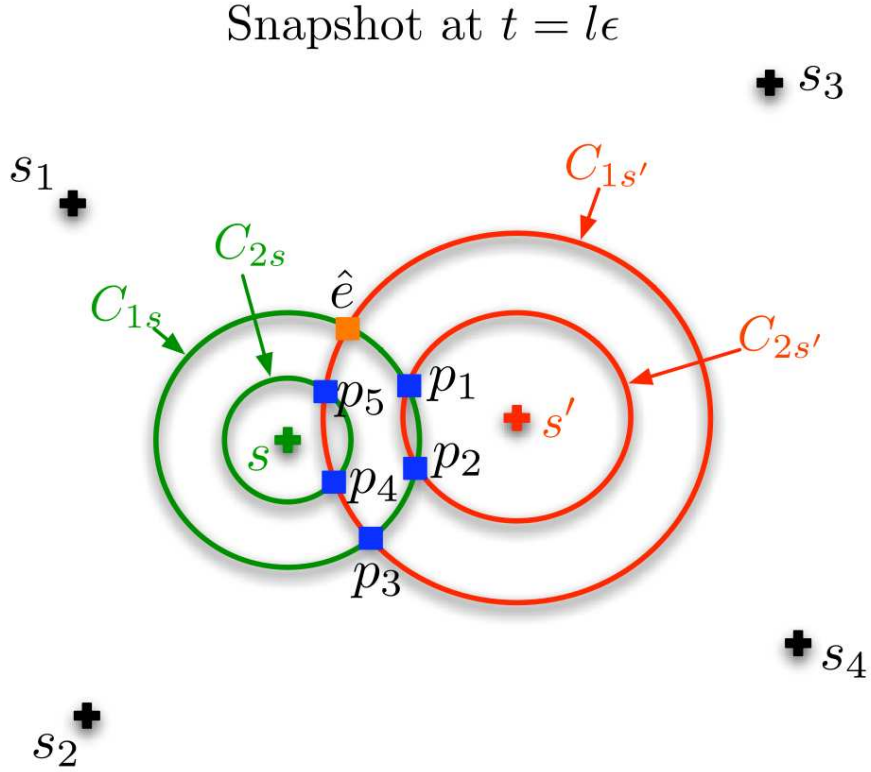


Figure 3.2: Geometry of the processing in Stage 1. Six sensors $s, s', s_1, s_2, s_3, s_4$ are shown. Sensors s and s' have two ToAs each, denoted by $\{\tau_1(s), \tau_2(s)\}$ and $\{\tau_1(s'), \tau_2(s')\}$. ToAs $\tau_1(s)$ and $\tau_1(s')$ were produced by an event \mathcal{E} that occurred at time $t_e \approx l\epsilon$. Consider a hypothesized event time $u = l\epsilon$ and draw circles C_{1s} and C_{2s} , centered at sensor s , with radii $\tau_1(s) - u$ and $\tau_2(s) - u$ (likewise for $C_{1s'}$ and $C_{2s'}$). C_{1s} and $C_{1s'}$ intersect at a point \hat{e} close to \mathcal{E} 's location. All other points of intersection between C_{is} and $C_{js'} \forall i, j$ (denoted by $p_i, i = 1, \dots, 5$) are phantom estimates.

that is robust to misses and outliers. We provide an overview of our three-stage algorithm here, followed by a detailed description of each stage in the following sections.

Stage 1: The goal of the first stage is to quickly generate a number of candidate events that are “reasonably good”. To do this, we first discretize the times at

which large-scale events might have occurred and suppose that they only take the values $(\dots, -2\epsilon, -\epsilon, 0, \epsilon, 2\epsilon, \dots)$. Next, we hypothesize that an event at $t = l\epsilon$ produced the i th ToA at sensor s and j th ToA at sensor s' . Under these hypotheses, it is easy to show that the event must be located at the intersection of two circles – the circles are centered at the sensor locations $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_{s'}$ and their radii are given by $\tau_s(i) - l\epsilon$ and $\tau_{s'}(j) - l\epsilon$ respectively. Since the intersection of two circles can be specified in closed-form, we can generate the event location easily. By repeating this process for other values of (l, s, i, s', j) , we generate a number of candidate event locations.

What does this list of candidates look like? Suppose that an event $(\boldsymbol{\alpha}_e, t_e)$ produced the ToAs $\tau_s(i)$ and $\tau_{s'}(j)$ at sensors s and s' *in reality*. Consider a hypothesized event at a time u that is close to t_e , say $u = \epsilon \lfloor t_e / \epsilon \rfloor$. The event location produced by intersecting circles centered at $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_{s'}$ with radii $\tau_s(i) - u$ and $\tau_{s'}(j) - u$ will be close to the true event location $\boldsymbol{\alpha}_e$. Thus, estimates close to true event locations will be a part of the list of candidates. On the other hand, suppose that one of hypotheses is false – either the ToAs $\tau_s(i)$ and $\tau_{s'}(j)$ are *not* produced by the same large scale event or even if they are, the event is not at the hypothesized time $l\epsilon$. In such cases, intersecting circles centered at $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_{s'}$ with radii $\tau_s(i) - l\epsilon$ and $\tau_{s'}(j) - l\epsilon$ produces an estimated event location; but, such an event clearly did not occur in reality. We call such estimates “phantoms”.

Figure 3.2 shows an example of true and phantom estimates being generated by intersecting circles drawn at sensors s and s' .

To discard phantom estimates from the list, we compute a “goodness” metric for each of these events. This metric uses the measurements at all the sensors to capture the likelihood of the event having happened. We discard candidate estimates whose goodness falls below a threshold from the list. Since many phantoms have a low value of goodness, the list is pruned considerably. However, we choose the threshold conservatively to ensure that no event estimate close to a true event is discarded (in spite of non-idealities such as noise, misses at some sensors etc.). As a result, some phantom estimates survive the pruning process and remain on the list of candidate events. Therefore, the output of Stage 1 is a list of candidate events, containing both true and phantom estimates.

Stage 2: An event estimate from Stage 1 produced by intersecting circles centered at a pair of sensors is bound to be noisy, since it does not take measurements at other sensors into account. In the second stage, we use measurements at all the sensors to refine this noisy estimate. We do this in an iterative fashion. In the first iteration of Stage 2, we use the event estimates from Stage 1 as a starting point and linearize the constraints imposed by the ToA measurements about this point at all the sensors. This allows us to refine the estimate with low-complexity. In subsequent iterations of Stage 2, we use the estimates from the previous iteration

as the starting point to further refine them. The output of this stage, therefore, is a *palette* of events, that contains refined versions of both true and phantom event estimates.

Stage 3: The goal of Stage 3 is to pick the true event estimates from the over-complete palette of candidate events, containing both true and phantom event estimates. To do this, we pick the *subset* of events from the palette that fit the observations at all the sensors in the “best” possible fashion. In principle, this can be done by brute-force: pick a subset of events from the palette, hypothesize that this subset contains all the true events that occurred and no phantom events, evaluate the likelihood of the observations under this hypothesis and pick the subset with the largest likelihood as the estimate of events that occurred. However, we show that it can be solved far more efficiently by posing it as a variant of the matching problem on a graph. The events in the palette form the first set of nodes in the graph and the observations at different sensors form the other set. The objective is to pick events from the palette and then add edges to the graph, pairing the “picked events” with the observations at different sensors. Each edge comes with a value, which captures the likelihood that the event at one end of the edge generated the observation at the other end. Furthermore, we need to satisfy some constraints while adding the edges. Specifically, the two sets of constraints are: (i) at each sensor, an event must either produce a ToA or declared to be

missed and (ii) every ToA must either be associated with an event or declared to be an outlier. We capture such constraints in a binary integer programming problem. Finally, we relax the integer program and solve a linear problem to pick the “most likely” subset of events from the palette.

We now provide a detailed description of each stage.

3.5 Stage 1: Generating Candidate Events

Suppose that $\tau_s(i)$, the i th ToA at sensor s , and $\tau_{s'}(j)$, the j th ToA at sensor s' , are produced by the same event that occurred at time $u = l\epsilon$. If the measurements are noiseless, we see from (3.2), the location of the hypothesized event, denoted by \mathbf{r} , must satisfy

$$u + \|\mathbf{r} - \boldsymbol{\theta}_s\| = \tau_s(i) \Rightarrow \|\mathbf{r} - \boldsymbol{\theta}_s\| = \tau_s(i) - u \quad (3.17)$$

$$u + \|\mathbf{r} - \boldsymbol{\theta}_{s'}\| = \tau_{s'}(j) \Rightarrow \|\mathbf{r} - \boldsymbol{\theta}_{s'}\| = \tau_{s'}(j) - u. \quad (3.18)$$

Thus, the hypothesized event must be located at the intersection of two circles centered at $\boldsymbol{\theta}_s$ and $\boldsymbol{\theta}_{s'}$ with radii $\tau_s(i) - u$ and $\tau_{s'}(j) - u$ respectively. Denoting the points of intersection by \mathbf{r}_+ and \mathbf{r}_- , we have

$$\mathbf{r}_{\pm} = \frac{\boldsymbol{\theta}_s + \boldsymbol{\theta}_{s'}}{2} - \frac{R_{s'}^2 - R_s^2}{2d_{ss'}^2}(\boldsymbol{\theta}_{s'} - \boldsymbol{\theta}_s) \pm \frac{b}{2d_{ss'}^2}(\boldsymbol{\theta}_{s'}^{\perp} - \boldsymbol{\theta}_s^{\perp}), \quad (3.19)$$

where $R_s = \tau_s(i) - u$ and $R_{s'} = \tau_{s'}(j) - u$ are the radii of the circles, $d_{ss'} = \|\boldsymbol{\theta}_s - \boldsymbol{\theta}_{s'}\|$ is the distance between the sensors, $b = \sqrt{[(R_s + R_{s'})^2 - d_{ss'}^2][d_{ss'}^2 - (R_s - R_{s'})^2]}$ and $\boldsymbol{\theta}_{s'}^\perp - \boldsymbol{\theta}_s^\perp = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} (\boldsymbol{\theta}_{s'} - \boldsymbol{\theta}_s)$ is a vector perpendicular to the line joining the sensors $\boldsymbol{\theta}_{s'} - \boldsymbol{\theta}_s$.

If an event at $(\boldsymbol{\alpha}_e, t_e)$ indeed produced *both* the ToAs $\tau_s(i)$ and $\tau_{s'}(j)$, and the hypothesized event time $u = l\epsilon \approx t_e$, then one of the location estimates \mathbf{r}_+ or \mathbf{r}_- will be close to $\boldsymbol{\alpha}_e$ (but perturbed by noise) and the other will be a phantom estimate. On the other hand, if $\tau_s(i)$ and $\tau_{s'}(j)$ are not produced by the same event or the hypothesized event time is not close to that of any event, then both (\mathbf{r}_+, u) and (\mathbf{r}_-, u) are phantom estimates that we should discard ultimately.

Accuracy of Estimates: We now calculate the covariance of the spatial location estimates \mathbf{r}_\pm when the ToA measurements are noisy and use it to compute the “goodness” of the estimated events (\mathbf{r}_\pm, u) . Suppose that the ToAs $\tau_s(i)$ and $\tau_{s'}(j)$ are corrupted by measurement noises $n_s(i)$ and $n_{s'}(j)$ respectively, leading to corresponding errors \mathbf{e}_\pm in the spatial event estimates (3.19). Assuming that the measurement noises are small and using a Taylor series expansion of (3.19),

we get $\mathbf{e}_\pm = \mathbf{K}_\pm \begin{bmatrix} n_s(i) \\ n_{s'}(j) \end{bmatrix}$ where,

$$\mathbf{K}_\pm = \mathbf{M} \begin{bmatrix} \frac{R_s}{d_{ss'}} & \frac{-R_{s'}}{d_{ss'}} \\ \frac{R_s(d_{ss'} - \beta)}{d_{ss'}\sqrt{R_s^2 - \beta^2}}q_\pm & \frac{\beta R_{s'}}{d_{ss'}\sqrt{R_s^2 - \beta^2}}q_\pm \end{bmatrix} \quad (3.20)$$

with $\mathbf{M} = \begin{bmatrix} \frac{\boldsymbol{\theta}_{s'} - \boldsymbol{\theta}_s}{d_{ss'}} & \frac{\boldsymbol{\theta}_{s'}^\perp - \boldsymbol{\theta}_s^\perp}{d_{ss'}} \end{bmatrix}$, $\beta = (R_s^2 - R_{s'}^2 + d_{ss'}^2)/2d_{ss'}$ and $q_\pm = \text{sign}((\mathbf{r}_\pm - \boldsymbol{\theta}_s)^T (\boldsymbol{\theta}_{s'}^\perp - \boldsymbol{\theta}_s^\perp))$. Using the fact that $n_s(i)$ and $n_{s'}(j)$ are independent Gaussian random variables, we obtain the covariance matrices of the estimates to be $\mathbf{C}_\pm = \mathbb{E}(\mathbf{e}_\pm \mathbf{e}_\pm^T) = \sigma^2 \mathbf{K}_\pm \mathbf{K}_\pm^T$.

We repeat this process for other choices of ToA pairs and hypothesized event times, resulting in a list of candidate events $\{(\mathbf{r}_n, u_n)\}$ with covariance matrices $\{\mathbf{C}_n = \sigma^2 \mathbf{K}_n \mathbf{K}_n^T\}$, $n = 1, 2, \dots$. We now compute a goodness metric for each of these candidates to quickly discard those that are “obviously” phantom estimates from the list.

Goodness Metric: Consider a candidate event $\mathcal{E} = (\mathbf{r}, u)$ with a covariance matrix $\mathbf{C} = \sigma^2 \mathbf{K} \mathbf{K}^T$. The goodness metric for \mathcal{E} is designed to capture the likelihood of the observations at all the sensors, assuming that \mathcal{E} happened. We compute the goodness in two steps: first, we calculate individual goodnesses for \mathcal{E} at each sensor and then, multiply them out to obtain an overall goodness. The basic

steps in computing the goodness for the event \mathcal{E} at a generic sensor s are as follows:

1. Assuming that \mathcal{E} happened, we predict the expected ToA that sensor s must have seen, given by $\eta_s = u + \|\mathbf{r} - \boldsymbol{\theta}_s\|$.

2. We compare the ToAs observed at sensor s , given by $\Omega_s = \{\tau_s(1), \dots, \tau_s(M_s)\}$, with the predicted ToA η_s and pick the one that is closest to η_s . We denote this ToA by $\tau_s(\mathcal{E})$, i.e.,

$$\tau_s(\mathcal{E}) = \arg \min_i |\tau_s(i) - \eta_s| \quad (3.21)$$

Loosely speaking, $\tau_s(\mathcal{E})$ is the best evidence that the sensor s has to offer for the event \mathcal{E} having taken place.

3. The goodness at sensor s depends solely on the difference between the predicted ToA η_s and the observed ToA $\tau_s(\mathcal{E})$, given by $z_s = \tau_s(\mathcal{E}) - \eta_s$. We denote the goodness for \mathcal{E} at sensor s by $L(z_s)$. Intuitively, we expect $L(z_s)$ to be large if the mismatch z_s is small and decrease monotonically as z_s increases.

Before getting into the details of computing $L(z_s)$, we make a few remarks on what we would expect z_s to typically look like. Suppose that the event \mathcal{E} actually happened and is not a phantom estimate. Assume further that it was heard at sensor s . Then, the only sources of mismatch z_s are the uncertainty in estimated event location (captured by the covariance matrix \mathbf{C}) and the measurement noise,

both of which are expected to be “small”. Thus, in this case, we would expect z_s also to be small. On the other hand, the event \mathcal{E} could have happened, but might have been missed at sensor s . In this case, z_s is obtained by comparing the predicted ToA η_s with the ToA produced by *some other event* ($\tau_s(\mathcal{E})$ has to be produced by a different event, since \mathcal{E} was missed at sensor s). In such cases, we would expect z_s to be large (compared to the typical values taken by the measurement noise). Therefore, if z_s is large at a “few” sensors, then it is possible that the event \mathcal{E} was simply missed at these sensors. On the other hand, if z_s is large at too many sensors, then it is more likely that \mathcal{E} is a phantom estimate.

Based on this intuition, we compute $L(z_s)$ in two steps: we first condition on the event \mathcal{E} being heard or missed at sensor s and obtain the conditional likelihoods, $L(z_s|\mathcal{E} \text{ heard})$ and $L(z_s|\mathcal{E} \text{ missed})$. The overall goodness is a weighted average of the conditional likelihoods, with the weights depending on the probability of miss as

$$L(z_s) = (1 - p_{miss})L(z_s|\mathcal{E} \text{ heard}) + p_{miss}L(z_s|\mathcal{E} \text{ missed}). \quad (3.22)$$

We now provide the details of computing $L(z_s|\mathcal{E} \text{ heard})$ and $L(z_s|\mathcal{E} \text{ missed})$.

Computing $L(z_s|\mathcal{E} \text{ heard})$: Suppose that event \mathcal{E} was heard at sensor s and $\tau_s(\mathcal{E})$ was the corresponding ToA. Since the spatial location of \mathcal{E} has an uncertainty captured by a covariance matrix \mathbf{C} , we can model it as the sum of two terms $\mathbf{r} + \mathbf{e}$,

where \mathbf{e} is a random variable with $\mathbb{E}(\mathbf{e}) = \mathbf{0}$ and $\mathbb{E}(\mathbf{e}\mathbf{e}^T) = \mathbf{C}$. Assuming now that the event occurred at $(\mathbf{r} + \mathbf{e}, u)$, $\tau_s(\mathcal{E})$ must satisfy the measurement model (3.2):

$$\tau_s(\mathcal{E}) = u + \|\mathbf{r} + \mathbf{e} - \boldsymbol{\theta}_s\| + n, \quad (3.23)$$

where the $n \sim N(0, \sigma^2)$ is the measurement noise. If the error in the estimated event location $\|\mathbf{e}\|$ is much smaller than the distance between the event and sensor s $\|\mathbf{r} - \boldsymbol{\theta}_s\|$, we can expand $\|\mathbf{r} + \mathbf{e} - \boldsymbol{\theta}_s\|$ as a Taylor series in \mathbf{e} and retain only the linear term, to approximate it as

$$\|\mathbf{r} + \mathbf{e} - \boldsymbol{\theta}_s\| \approx \|\mathbf{r} - \boldsymbol{\theta}_s\| + \left\langle \mathbf{e}, \frac{\mathbf{r} - \boldsymbol{\theta}_s}{\|\mathbf{r} - \boldsymbol{\theta}_s\|} \right\rangle, \quad (3.24)$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product. Using this approximation in (3.23), we get

$$\tau_s(\mathcal{E}) \approx u + \|\mathbf{r} - \boldsymbol{\theta}_s\| + \left\langle \mathbf{e}, \frac{\mathbf{r} - \boldsymbol{\theta}_s}{\|\mathbf{r} - \boldsymbol{\theta}_s\|} \right\rangle + n. \quad (3.25)$$

Recognizing $u + \|\mathbf{r} - \boldsymbol{\theta}_s\|$ to be the predicted ToA η_s and using the fact that $\tau_s(\mathcal{E}) - \eta_s = z_s$, we get

$$z_s = \left\langle \mathbf{e}, \frac{\mathbf{r} - \boldsymbol{\theta}_s}{\|\mathbf{r} - \boldsymbol{\theta}_s\|} \right\rangle + n. \quad (3.26)$$

This equation is intuitively pleasing: the mismatch z_s arises because of the uncertainty in the event location estimate \mathbf{e} and measurement noise n . From (3.20),

we see that the error in the location estimate \mathbf{e} can be expressed as $\mathbf{K} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}$

where n_1 and n_2 are independent $N(0, \sigma^2)$ random variables. Since n_1, n_2 and n are all Gaussian random variables with zero mean, z_s is also a zero mean Gaussian random variable. Neglecting any correlation between n_1, n_2 and n , we compute the variance of z_s , denoted by σ_s^2 , to be

$$\sigma_s^2 = \sigma^2 \left(1 + \frac{\|(\mathbf{r} - \boldsymbol{\theta}_s)^T \mathbf{K}\|^2}{\|\mathbf{r} - \boldsymbol{\theta}_s\|^2} \right). \quad (3.27)$$

Finally, since $z_s \sim N(0, \sigma_s^2)$, we obtain the conditional likelihood $L(z_s | \mathcal{E} \text{ heard})$ to be

$$L(z_s | \mathcal{E} \text{ heard}) = \frac{1}{\sqrt{2\pi\sigma_s^2}} \exp\left(-\frac{z_s^2}{2\sigma_s^2}\right). \quad (3.28)$$

Computing $L(z_s | \mathcal{E} \text{ missed})$: When the event \mathcal{E} is missed at sensor s , the observation z_s is obtained by taking the difference between the predicted ToA for \mathcal{E} , denoted by η_s , and an observed ToA $\tau_s(\mathcal{E})$ produced by a *completely different event*. Furthermore, we note that $\tau_s(\mathcal{E})$ is the closest among all ToAs recorded at sensor s to η_s (see (3.21)). Two conditions must be satisfied for this to have happened: (a) there must be no ToAs at sensor s that are closer to η_s than $\tau_s(\mathcal{E})$. In other words, there must be no ToAs in the interval $[\eta_s - |z_s|, \eta_s + |z_s|]$ where $z_s = \tau_s(\mathcal{E}) - \eta_s$. (b) We must observe a ToA close to $\eta_s + z_s$ i.e. there must be a ToA in the infinitesimal interval $[\eta_s + z_s, \eta_s + z_s + dz]$. We note that each sensor observes ToAs being generated at a rate $\lambda = \lambda_{LS}(1 - p_{miss}) + \lambda_O$. We make the further assumption that these ToAs arrive according to a Poisson process with

rate λ . With this assumption, the probability of (a) happening is $\exp(-2\lambda|z_s|)$ and the likelihood of (b) is λdz . Since the time intervals considered in (a) and (b) do not overlap, the events are independent and we obtain

$$L(z_s|\mathcal{E} \text{ missed})dz = \exp(-2\lambda|z_s|)\lambda dz. \quad (3.29)$$

Therefore, $L(z_s|\mathcal{E} \text{ missed}) = \lambda \exp(-2\lambda|z_s|) \quad \forall z_s$.

Putting (3.22),(3.28) and (3.29) together, we get the goodness at sensor s to be

$$L(z_s) = \frac{(1 - p_{miss})}{\sqrt{2\pi\sigma_s^2}} \exp\left(-\frac{z_s^2}{2\sigma_s^2}\right) + p_{miss}\lambda \exp(-2\lambda|z_s|). \quad (3.30)$$

Note that the exponential in the second term decays much slower than the first – it goes down only as $\exp(-|z_s|)$ unlike the first which decays as $\exp(-|z_s|^2)$. Therefore, the net effect of this term is to ensure that $L(z_s)$ does not become too small even when z_s is fairly large. Furthermore, typical values of $1/\lambda$ (for example, a value of 5/3 corresponding to 3 events in 5 seconds) are much larger than σ (typically 0.01 s), further slowing down the decay of the second term relative to the first. Therefore, for all practical purposes, we can neglect the decay in the second term and simply approximate $L(z_s)$ as

$$L(z_s) \approx \frac{(1 - p_{miss})}{\sqrt{2\pi\sigma_s^2}} \exp\left(-\frac{z_s^2}{2\sigma_s^2}\right) + p_{miss}\lambda. \quad (3.31)$$

The overall goodness for the event \mathcal{E} , denoted by g , is

$$g = \sum_{s=1}^N \log(L(z_s)) \quad (3.32)$$

If the goodness g falls below a threshold κ , we declare the event to be a phantom and discard it from the candidate list.

Choosing the threshold κ : We choose the threshold κ conservatively, thereby ensuring that we retain estimates close to true event locations and only discard candidates that are “obviously” phantoms. Specifically, we choose κ so that a true event is discarded with a probability smaller than a predetermined value δ_{throw} .

We now provide the details of computing κ .

Consider the e th event $\mathcal{E}_e = (\boldsymbol{\alpha}_e, t_e)$ that occurred. Let \mathcal{X}_0 and \mathcal{X}_1 denote the set of sensors that missed and heard \mathcal{E}_e respectively. As before, we let z_s denote the difference between the predicted time and the “best evidence” (3.21) for \mathcal{E}_e at sensor s . Consider a sensor s that heard the event ($s \in \mathcal{X}_1$). At such a sensor, the first term in (3.31) dominates the second (for the parameter choices we are interested in) and we can approximate $\log L(z_s)$ as

$$\log L(z_s) \approx \left[\log \frac{1 - p_{miss}}{\sqrt{2\pi\sigma_s^2}} \right] - \frac{z_s^2}{2\sigma_s^2} \quad s \in \mathcal{X}_1. \quad (3.33)$$

where $z_s \sim N(0, \sigma_s^2)$. To further simplify the analysis, we neglect variations in the prediction error variance σ_s^2 across sensors *in the first term* of the above expression.

We set $\sigma_s^2 \approx \sigma^2$ in the first term of (3.33) and get

$$\log L(z_s) \approx \left[\log \frac{1 - p_{miss}}{\sqrt{2\pi\sigma^2}} \right] - \frac{z_s^2}{2\sigma_s^2} \quad s \in \mathcal{X}_1. \quad (3.34)$$

Now consider a sensor s' that missed \mathcal{E}_e ($s' \in \mathcal{X}_0$). At such a sensor, the first term in (3.31) dies down rapidly and we can approximate $\log L(z_{s'})$ as

$$\log L(z_{s'}) \approx \log p_{miss} \lambda \quad s' \in \mathcal{X}_0. \quad (3.35)$$

Therefore, if N_1 sensors hear the event and $N - N_1$ sensors miss the event, we can use (3.34) and (3.35) to obtain an approximate expression for the goodness of \mathcal{E}_e :

$$g = \left[\log \frac{1 - p_{miss}}{\sqrt{2\pi\sigma^2}} \right] N_1 + [\log(p_{miss}\lambda)](N - N_1) - \frac{1}{2} \sum_{s \in \mathcal{X}_1} \frac{z_s^2}{\sigma_s^2}. \quad (3.36)$$

Since the prediction error $z_s \sim N(0, \sigma_s^2)$, the term $\sum_{s \in \mathcal{X}_1} \frac{z_s^2}{\sigma_s^2}$ in (3.36) is a chi-squared distributed random variable with N_1 degrees of freedom. Thus, conditioned on N_1 sensors hearing \mathcal{E}_e , the goodness has a chi-squared distribution with N_1 degrees of freedom, whose mean is shifted by the first two terms in (3.36). Let us denote this distribution by $f_G(g|N_1)$. By the law of total probability, the unconditional distribution of the goodness $f_G(g)$ is given by

$$f_G(g) = \sum_{N_1=0}^N f_G(g|N_1) p_{miss}^{N-N_1} (1 - p_{miss})^{N_1}. \quad (3.37)$$

We set the threshold κ so that the chance that the goodness for \mathcal{E}_e is lower than this threshold is equal to δ_{throw} :

$$P(G < \kappa) = \int_{-\infty}^{\kappa} f_G(g) dg = \delta_{throw} \quad (3.38)$$

We run Monte Carlo simulations to generate samples of G according to the distribution in (3.37) and then pick the threshold to satisfy (3.38).

Clustering via ToA groupings: While we have discarded events that are obviously phantoms by thresholding the goodness, the list can be pruned further by eliminating events that are essentially “poorer duplicates” of other events on the list. To understand the origins of duplication in the list of candidate events, consider the following example. Suppose that for a hypothesized event time u , we intersect circles of radii $\tau_s(i) - u$ and $\tau_{s'}(j) - u$ centered at sensors s and s' to produce an event estimate \mathbf{r}_a . Consider the “next” hypothesized event time $u' = u + \epsilon$. If ϵ is “small”, intersecting circles of radii $\tau_s(i) - u'$ and $\tau_{s'}(j) - u'$ centered at sensors s and s' will result in an event estimate \mathbf{r}_b that is very close to \mathbf{r}_a . It is clear that (\mathbf{r}_a, u) and (\mathbf{r}_b, u') are estimates of the same event and it is sufficient to store the “better” estimate among the two. We could do this by using a standard clustering algorithm which groups estimates that are close in space and time and stores only the “best” representative from each group. However, we exploit the structure of the problem to cluster these events in a principled fashion by introducing the concept of a *grouping*.

Consider an event $\mathcal{E} = (\mathbf{r}, u)$ whose goodness is above the threshold. As before, we denote the predicted ToA for \mathcal{E} at sensor s by $\eta_s = u + \|\mathbf{r} - \boldsymbol{\theta}_s\|$ and the corresponding “best fit” evidence by $\tau_s(\mathcal{E})$ (the observed ToA at sensor s that is closest to η_s). The grouping \mathbf{p} associated with an event \mathcal{E} is a set of N quantities $\{p_1, p_2, \dots, p_N\}$ where p_s stores the evidence $\tau_s(\mathcal{E})$ if it is “compelling”; otherwise,

it records the fact that sensor s has missed the event. Specifically, if the difference $|\tau_s(\mathcal{E}) - \eta_s|$ is smaller than a threshold γ , we set $p_s = \tau_s(\mathcal{E})$; otherwise, we store the string *miss* in p_s . Duplicated events, such as \mathbf{r}_a and \mathbf{r}_b in the example described above, are likely to have the same grouping – since the events are close to one another in space and time, evidence that is “compelling” for one is also likely to be compelling for the other. This observation provides us with a simple rule to cluster events and pick a representative: if two events (\mathbf{r}_1, u_1) and (\mathbf{r}_2, u_2) have groupings \mathbf{p}_1 and \mathbf{p}_2 that are identical, then we only retain the event with the greater goodness (as defined in (3.32)).

3.6 Stage 2: Refining the Estimates

The event location estimates in Stage 1 are produced by intersecting circles whose radii are derived from the observed ToAs at a pair of sensors s, s' . Since the measurements are noisy and we do not account for the measurements at other sensors, the estimates can have significant errors. In this section, we use the measurements at all sensors to refine such noisy estimates.

Let $\mathcal{E} = (\mathbf{r}, u)$ be a generic event in the candidate list at the end of stage 1. We only use the sensors that have “compelling” evidence for \mathcal{E} in the refinement process. Specifically, letting $\eta_s = u + \|\mathbf{r} - \boldsymbol{\theta}_s\|$ denote the predicted ToA for \mathcal{E}

at sensor s and $\tau_s(\mathcal{E})$ be the corresponding “best fit” ToA, we use sensor s in the refinement process only if $|\tau_s(\mathcal{E}) - \eta_s| \leq \gamma$.

Refinement Procedure: Suppose that the sensors s_1, s_2, \dots, s_Q have ToAs that are within γ of the predicted ToA for $\mathcal{E} = (\mathbf{r}, u)$ at these sensors. We denote the best evidence for \mathcal{E} at these sensors by $\tau_{s_1}(\mathcal{E}), \tau_{s_2}(\mathcal{E}), \dots, \tau_{s_Q}(\mathcal{E})$ respectively. Since the refined estimate, denoted by $(\mathbf{r} + \Delta\mathbf{r}, u + \Delta u)$, must fit the measurement model, we have

$$\tau_{s_j}(\mathcal{E}) = u + \Delta u + \|\mathbf{r} + \Delta\mathbf{r} - \boldsymbol{\theta}_{s_j}\| + n_{s_j}, \quad (3.39)$$

where $n_{s_j} \sim N(0, \sigma^2)$ and $j = 1, 2, \dots, Q$. If the spatial refinement $\|\Delta\mathbf{r}\|$ is much smaller than the distance $\|\mathbf{r} - \boldsymbol{\theta}_{s_j}\|$ between the event and sensor s_j , we can expand $\|\mathbf{r} + \Delta\mathbf{r} - \boldsymbol{\theta}_{s_j}\|$ as a Taylor series in $\Delta\mathbf{r}$ and retain only the linear term to approximate it as

$$\|\mathbf{r} + \Delta\mathbf{r} - \boldsymbol{\theta}_{s_j}\| \approx \|\mathbf{r} - \boldsymbol{\theta}_{s_j}\| + \left\langle \Delta\mathbf{r}, \frac{\mathbf{r} - \boldsymbol{\theta}_{s_j}}{\|\mathbf{r} - \boldsymbol{\theta}_{s_j}\|} \right\rangle, \quad (3.40)$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product. With this approximation, (3.39) can be rewritten as

$$\tau_{s_j}(\mathcal{E}) - u - \|\mathbf{r} - \boldsymbol{\theta}_{s_j}\| = \begin{bmatrix} \frac{\mathbf{r}^T - \boldsymbol{\theta}_{s_j}^T}{\|\mathbf{r} - \boldsymbol{\theta}_{s_j}\|} & 1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{r} \\ \Delta u \end{bmatrix} + n_{s_j}. \quad (3.41)$$

for $j = 1, 2, \dots, Q$. Let \mathbf{y} denote the Q dimensional vector whose j th component is $\tau_{s_j}(\mathcal{E}) - u - \|\mathbf{r} - \boldsymbol{\theta}_{s_j}\|$ and H denote the $Q \times 3$ matrix whose j th row is

$\begin{bmatrix} \frac{\mathbf{r}^T - \boldsymbol{\theta}_{s_j}^T}{\|\mathbf{r} - \boldsymbol{\theta}_{s_j}\|} & 1 \end{bmatrix}$. Then, the least-squares estimate of $[\Delta \mathbf{r} \ \Delta u]$ is given by

$$\begin{bmatrix} \Delta \hat{\mathbf{r}} \\ \Delta \hat{u} \end{bmatrix} = (H^T H)^{-1} H^T \mathbf{y}. \quad (3.42)$$

We update the event location and time estimates and set $\mathbf{r} \leftarrow \mathbf{r} + \Delta \hat{\mathbf{r}}$ and $u \leftarrow u + \Delta \hat{u}$. We now use the refined estimate $(\mathbf{r} + \Delta \hat{\mathbf{r}}, u + \Delta \hat{u})$ as a starting point and repeat the process – this includes computing the grouping for $(\mathbf{r} + \Delta \hat{\mathbf{r}}, u + \Delta \hat{u})$, using the grouping to identify sensors that heard it and then refining the estimate further using the ToAs at these sensors. We typically perform 10 such rounds of refinement for each candidate point from stage 1. The threshold parameter γ – used to decide if a sensor heard/missed \mathcal{E} – must be chosen appropriately and from our simulations, we find that choosing $\gamma = 6\sigma$ works well.

To conclude, the output of stage 2 is a palette of candidate events that are refined versions of the estimates from stage 1. Note that the palette contains both true and phantom event estimates.

3.7 Stage 3: Picking true events from the palette

In this stage, we pose and solve a problem on a graph to discard phantom events from the overcomplete palette and only retain the ones that occurred. The events in the palette, denoted by $\mathcal{E}_1 = (\mathbf{r}_1, u_1), \dots, \mathcal{E}_P = (\mathbf{r}_P, u_P)$, form one set

of nodes of the graph and the observations at the sensors form the other set of nodes. Figure 3.3 shows an example of such a graph. We represent the events in

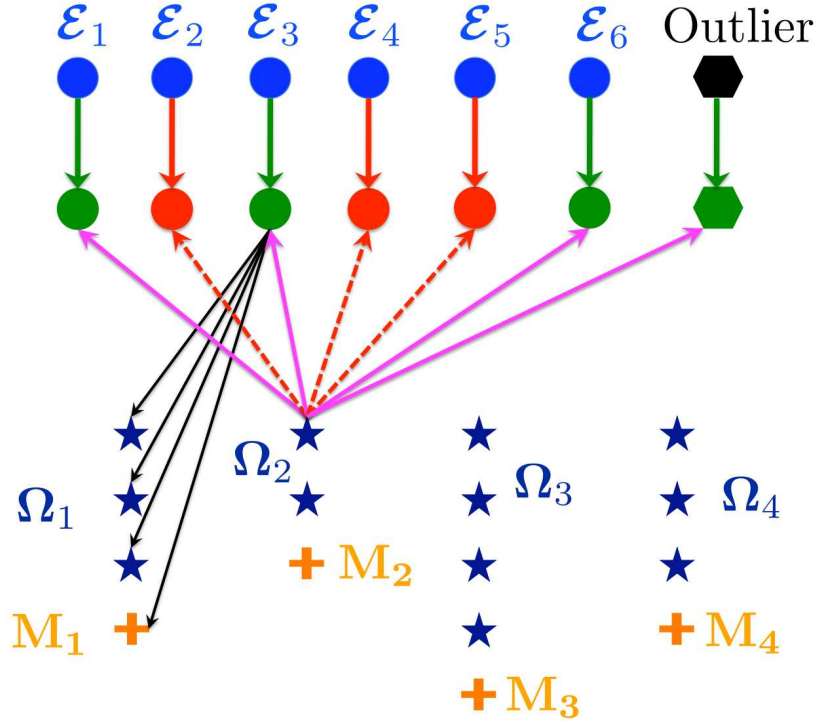


Figure 3.3: Modified version of matching problem on a bipartite graph. Events in the palette are shown as blue circles and the observations at sensors are shown as blue stars. Green circles represent events that are picked while red circles denote phantom events. We need to draw edges between the picked events and the observations, subject to constraints, so as to maximize the sum of the values of the edges.

the palette by the blue circles in the uppermost row of the graph. Each observed ToA is denoted by a blue star and the ToAs observed at a sensor are arranged

in a column. In the example shown in Figure 3.3, there are 4 sensors and they have $\{3, 2, 4, 3\}$ observations respectively. The nodes marked M_1, M_2, M_3, M_4 are “miss” nodes, that serve as proxies for any observations that might have been missed at the sensors. Similarly, the node marked Outlier acts as a representative for small-scale events that generate outlier observations at different sensors. The goal of Stage 3 is to pick a subset of the events $\mathcal{E}_1, \mathcal{E}_2, \dots, \mathcal{E}_P$ from the palette and associate them with the observations at the different sensors, thereby establishing a correspondence between the two “halves” of the bipartite graph. The problem of picking the “most reasonable” correspondence can be phrased as a constrained binary integer program and we provide the details now.

The first set of decisions we must make are: for each $e = 1, 2, \dots, P$, did event \mathcal{E}_e happen or is it a phantom? We store the decision for the e th event in a binary variable δ_e and refer to those events that we declared to have happened ($\delta_e = 1$) as “picked events”. The second row of the graph in Figure 3.3 shows an example of such decisions – picked events are shown in green ($\mathcal{E}_1, \mathcal{E}_3$ and \mathcal{E}_6) while the ones declared to be phantoms are shown in red ($\mathcal{E}_2, \mathcal{E}_4$ and \mathcal{E}_5).

The next task is to establish a correspondence between the picked events and the observations at various sensors. First, we identify the constraints that a “valid” correspondence must satisfy. Consider a picked event \mathcal{E}_e (such as \mathcal{E}_3 in Fig. 3.3). For each sensor s , we must decide on two things with regard to \mathcal{E}_e :

(a) Was \mathcal{E}_e heard/missed at sensor s and (b) if it was heard, which observation did it generate? These decisions can be neatly summarized in the graph of Figure 3.3. Suppose that we draw edges between the event \mathcal{E}_e and all the observation nodes at sensor s (this includes blue stars representing the ToAs in Ω_s and the miss node M_s , denoted by an orange cross). The four black lines in Figure 3.3 connecting \mathcal{E}_3 to $\tau_1(1), \tau_1(2), \tau_1(3)$ and M_1 are examples of such edges. We can now provide the answers to (a) and (b) by “activating” exactly one of these edges and specifying which one it is; for example, if decide that \mathcal{E}_e was missed at sensor s , we activate the edge joining \mathcal{E}_e and M_s (\mathcal{E}_3 and M_1 in the example of Fig. 3.3). On the other hand, if we decide that \mathcal{E}_e was heard at sensor s , we activate the edge between \mathcal{E}_e and the appropriate observation in Ω_s (one of the edges between \mathcal{E}_3 and $\tau_1(1), \tau_1(2)$ and $\tau_1(3)$ in this example). Note that we need to specify such a correspondence only for the picked events (green circles) and not for phantom estimates (red circles). Next, we consider the point of view of an observed ToA at one of the sensors (for example, the first observation at sensor 2, denoted by $\tau_2(1)$, in Figure 3.3). This ToA must either be generated by a picked event or it must be an outlier. As before, we draw edges between the observation node and all the events in the palette (shown by pink and red lines in Fig. 3.3). If we declare this ToA to be an outlier, we activate the edge that joins it to the node marked “Outlier”; on the other hand, if we decide that it was produced by a picked

event, we activate the appropriate pink edge. Note that we *cannot* activate any of the dotted red edges, because they associate this ToA to events that we have already declared to be phantoms. We now state these correspondence constraints formally.

Let i be a running index for all the observation nodes (including the “miss” nodes) as shown in the figure. Let $\Omega_s^\# = \Omega_s \cup M_s$ – the set of all observed ToAs at sensor s and the miss node M_s – denote the *observation nodes* at sensor s . Let $w_{ie} = 1$ if we activate the edge between observation i and event \mathcal{E}_e ; otherwise, we set $w_{ie} = 0$.

Event Node Constraints: Consider a picked event \mathcal{E}_e (one with $\delta_e = 1$). Since it must be associated with exactly one observation node at each sensor s , we have

$$\begin{aligned} \sum_{i \in \Omega_s^\#} w_{ie} &= 1 \quad \forall s, \{\forall e : \delta_e = 1\} \\ w_{ie} &\in \{0, 1\}. \end{aligned} \tag{3.43}$$

To convert this into a constraint that is valid for all events – and not just for those that are picked – we can rewrite (3.43) as

$$\delta_e \left[\left(\sum_{i \in \Omega_s^\#} w_{ie} \right) - 1 \right] = 0 \quad \forall s, e. \tag{3.44}$$

When $\delta_e = 1$, this reduces to the constraint in (3.43) and when $\delta_e = 0$, the constraint is trivially satisfied. Therefore, we need to choose our decisions w_{ie} and

δ_e so that they satisfy the constraints

$$\sum_{i \in \Omega_s^\#} w_{ie} \delta_e = \delta_e \quad \forall s, e. \quad (3.45)$$

Observation Node Constraints: Next, we consider the point of view of a node i that is an observed ToA at one of the sensors (*not* one of the miss nodes). Let $\mu_{iO} = 1$ if we declare observation i to be an outlier; otherwise, we set $\mu_{iO} = 0$. On the other hand, if we wish to associate it with a picked event (say, event e), then we must have $w_{ie} \delta_e = 1$. Since the observation node i must either be paired with a picked event or declared as an outlier, we have

$$\begin{aligned} \sum_{e=1}^P w_{ie} \delta_e + \mu_{iO} &= 1 \quad \forall i \in \Omega_1 \cup \Omega_2 \dots \cup \Omega_N \\ w_{ie}, \mu_{iO}, \delta_e &\in \{0, 1\} \quad \forall i, e. \end{aligned} \quad (3.46)$$

Cost function: We pick events from the palette and choose their correspondence with the observed ToAs to maximize the likelihood of the observations, given these decisions. Computing the likelihood is simplified by using the following fact: given the decisions (picking events and choosing the correspondence), the observations at two sensors s and s' are independent. This allows us to phrase the problem of picking the “most likely” decisions in a simple manner, using the graphical model of Figure 3.3: (1) We assign a value to each edge that connects an event node and an observation node. This value captures the likelihood – in fact, it is equal to

the logarithm of the likelihood – that the observation was produced by the event.

(2) Our goal is to pick events and activate edges, subject to the aforementioned constraints, so that sum of the values of the *activated* edges is maximized. We now specify the values of different edges.

Consider an edge between an event node \mathcal{E}_e and the miss node M_s at sensor s . The value of this edge is $\log(p_{miss})$, since the chance that an event is missed at any sensor is p_{miss} . Next, consider the edge between event $\mathcal{E}_e = (\mathbf{r}_e, u_e)$ and the j th observation at sensor s , $\tau_s(j)$. For \mathcal{E}_e to produce this observation, two things must happen – (a) \mathcal{E}_e must be heard at sensor s and (b) given that it was heard at sensor s , it must produce the observation $\tau_s(j)$. The chance that (a) happens is $1 - p_{miss}$ and the likelihood of (b) happening is exactly equal to the probability that the measurement noise accounts for the difference between the observed ToA $\tau_s(j)$ and the predicted ToA $u_e + \|\mathbf{r}_e - \boldsymbol{\theta}_s\|$. Putting them together, we get the value of the edge between \mathcal{E}_e and $\tau_s(j)$ to be

$$\text{val}(\mathcal{E}_e, \tau_s(j)) = \left(\log \frac{1 - p_{miss}}{\sqrt{2\pi\sigma^2}} \right) - \frac{(\tau_s(j) - u_e - \|\mathbf{r}_e - \boldsymbol{\theta}_s\|)^2}{2\sigma^2}. \quad (3.47)$$

The value of the edge joining an observation at sensor s to the outlier node is trickier to compute. Since the outliers are generated by a Poisson process of rate λ_O , the chance that there are k outliers at sensor s over an observation window of length T is $e^{-\lambda_O T} (\lambda_O T)^k / k!$. The logarithm of this quantity, denoted by L , is

given by (ignoring constants),

$$L = k \log(\lambda_O T) - \log(k!). \quad (3.48)$$

To see the problem, let us pretend that the term $\log(k!)$ is absent. Then, this equation has a very simple interpretation: declaring an observation to be an outlier has the value $\log(\lambda_O T)$ and the overall value of declaring k observations at a sensor to be outliers is $k \log(\lambda_O T)$. However, the presence of the term $\log(k!)$ implies that the value of declaring an observation to be an outlier cannot be a constant quantity, say $\log \alpha$; rather, it also depends on the number of *other* observations we declare to be outliers. To circumvent this problem, we approximate the distribution of the number of outliers to be geometric with parameter q (as opposed to the true $\text{Poisson}(\lambda_O T)$ distribution). With this approximation, the log-likelihood of observing k outliers at sensor is $k \log q$. Thus, we can set the value of an edge that joins an observation at sensor s to the outlier node to $\log q$. We choose q to ensure that the probabilities assigned by the Poisson and geometric distributions are close to one another. Specifically, we choose q so that the mean-square error between the sequences $\{(1 - q)q^n, n = 0, 1, \dots\}$ and $\{e^{-\lambda_O T} (\lambda_O T)^n / n!, n = 0, 1, \dots\}$ is minimized. The overall value of the decisions

$\{\delta_e, w_{ie}, \mu_{iO}\}$ is given by

$$J = \sum_s \sum_{i \in \Omega_s^\#} \sum_{e=1}^P c_{ie} w_{ie} \delta_e + \sum_s \sum_{i \in \Omega_s} c_{iO} \mu_{iO}, \quad (3.49)$$

where c_{ie} is the value of activating the edge between the i th observation node (note that this could be a miss node too) and the e th event and $c_{iO} = \log q$ is the value of declaring the i th observation node to be an outlier (note that this summation is only over observed ToAs and does not include miss nodes).

From (3.44), (3.46) and (3.49), we see that the decision variables w_{ie} are always multiplied by δ_e and never occur by themselves. This is because we can only activate edges that connect the observation nodes to *events that we pick from the palette*. We use this fact to define new decision variables $\mu_{ie} \triangleq w_{ie} \delta_e$, which are also binary-valued. We can pose the problem of maximizing the cost function in (3.49) subject to the constraints in (3.44) and (3.46) as the following binary integer program (all variables either take the value 0 or 1):

$$\begin{aligned} \max J &= \sum_s \sum_{i \in \Omega_s^\#} \sum_{e=1}^P c_{ie} \mu_{ie} + \sum_s \sum_{i \in \Omega_s} c_{iO} \mu_{iO} \\ \sum_{i \in \Omega_s^\#} \mu_{ie} &= \delta_e \quad \forall s, e \\ \sum_{e=1}^P \mu_{ie} + \mu_{iO} &= 1 \quad \forall i \in \Omega_1 \cup \Omega_2 \dots \cup \Omega_N \\ \delta_e, \mu_{ie}, \mu_{iO} &\in \{0, 1\} \quad \forall i, e. \end{aligned} \quad (3.50)$$

A technical point: The variables μ_{ie} and δ_e are also linked through the relationship $\mu_{ie} = w_{ie}\delta_e$, where w_{ie} is binary valued. It might appear that we have omitted these constraints from the above formulation. We show in the appendix that these constraints can indeed be excluded without any change to the solution.

We relax the integer program and allow the variables to take any value between 0 and 1. This allows us to solve the problem as a linear program (LP), which is much faster. In all our simulations, when the number of sensors is “large enough” (typically, we simulate $N = 8$ or $N = 16$ sensors and $E = 3$ events), we find that, the decision variables that optimize the LP only take the values 0 or 1 and never take any value in between. This is analogous to the efficacy of LP decoding for turbo-like codes and it is of interest to investigate whether the literature in this area [33] can shed light on the performance of our algorithm. Finally, we declare the events “picked” by the LP (those with $\delta_e = 1$) to have taken place.

3.8 Simulation Results

Sensor Deployment Model: We run two sets of simulations – one with a “moderate” density of sensors and the other with a larger deployment density. For the moderate density scenario, we place $N = 8$ sensors at random in a circular region of radius $R = 1020$ meters. The denser deployment consists of $N = 16$ sensors

placed at random over an identical region.

Event Generation Model: We generate $E = 3$ large-scale events, with their times chosen at random from the interval 0-5 seconds. We choose the event locations so that they are “inside” the convex hull of the sensors. Specifically, we pick them randomly from a region that is a scaled-down version of the convex hull of the sensors, with the scale-factor being 90%. We generate outliers at a rate $\lambda_O = 3/100$ events/sec at each sensor.

Measurement Model: We set the speed of sound c to 340 m/s. Guided by our experimental results in [20], we choose the standard deviation of the measurement noise σ to be 0.02 s. We choose the probability with which the sensors miss an event to be $p_{miss} = 5\%$.

Algorithm Choices: We set the granularity of the hypothesized event times at $\epsilon = 0.04$ s. We assume that every sensor observes ToAs arriving at a rate $\lambda = 63/100 (= 3/5 + 3/100)$ events/sec and use it to compute the goodness. We choose the probability with which an actual event is discarded at the end of Stage 1 to be $\delta_{throw} = 4 \times 10^{-5}$ and compute the threshold κ accordingly (see (3.38)). Finally, we choose q – the parameter of the geometric distribution that approximates the $\text{Poisson}(\lambda_O T)$ distribution – to be 0.14.

We run 100 trials of the algorithm with the above parameters. The localization errors obtained using the proposed algorithm are plotted in Figures 3.4(a)

and 3.4(b) for the $N = 8$ and $N = 16$ sensor scenarios respectively. To benchmark the performance of our algorithm, we use the following “genie”. For each large-scale event $(\boldsymbol{\alpha}_e, t_e)$, we pick the ToAs produced by this event at various sensors. We then form the *Time Difference of Arrivals* (TDoAs) by taking the difference between the ToAs observed at different sensors and the one seen at the “first” sensor. These TDoAs are solely a function of the event location and we use them to localize the source by a brute force search over “reasonably good” candidates. Specifically, we discretize a $70\text{m} \times 70\text{m}$ region around the true source location $\boldsymbol{\alpha}_e$ into a 2000×2000 grid of points. We choose the point in the grid that best fits the TDoAs at different sensors in the least-squares sense as the estimate of the source (this is also the ML estimate if the measurement noise is Gaussian). The solid red line in Figure 3.8 shows the localization errors observed with this genie based approach. We make the following observations:

1. In all the trials, our algorithm correctly estimates the number of events to be three.
2. From Figures 3.4(a) and 3.4(b), we see that the estimation errors produced by the proposed algorithm and the genie virtually coincide with one another, demonstrating the efficacy of the proposed scheme.

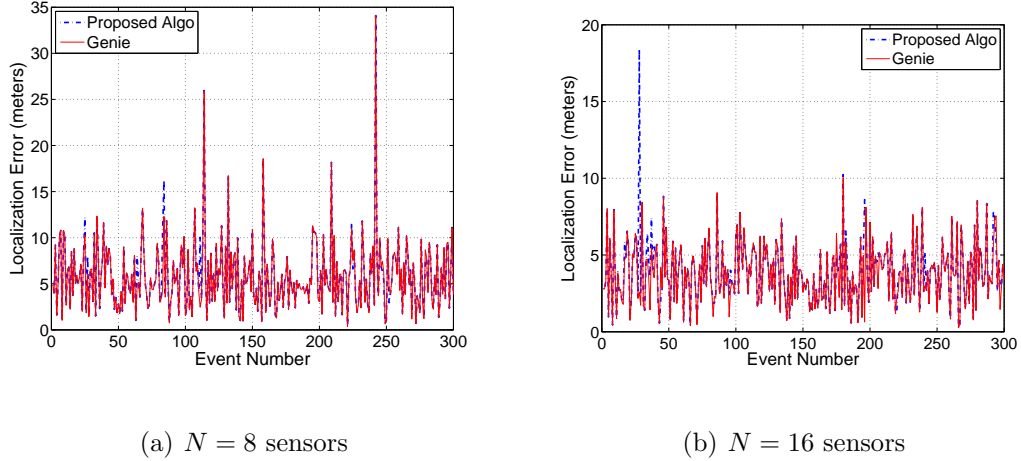


Figure 3.4: Localization errors with the proposed algorithm and a genie-based scheme with $N = 8$ and $N = 16$ sensors. The errors virtually coincide with one another, demonstrating the efficacy of the proposed algorithm.

3. Specifically, for the $N = 8$ sensor case, the average localization error obtained with proposed scheme is 5.87 m, which is only 0.07 m larger than that obtained with the genie (average error = 5.8 m). Similarly, for the larger deployment density scenario, the average localization error with the proposed scheme is 3.92 m and the corresponding metric with the genie based scheme is only marginally lower at 3.77 m.

Chapter 4

Collaborative Estimation in Dispersive Environments

In this chapter, we study the problem of reconstructing an unknown signal recorded at multiple sensors through unknown dispersive channels. The only information we are provided regarding the channels is a coarse estimate of their length in time, also called the *delay spread*. We parallelize the problem by switching to the frequency domain and solve it as follows:

- Over small enough frequency bands, we approximate the channels seen by each sensor as a quadratic function of frequency. This allows us to estimate the signal within each band efficiently, bootstrapping with an estimate obtained by approximating the channels to be constant within the band.
- However, since the signal and the channels are both unknown, the signals can only be estimated up to a scale factor.

- We overlap the bands and use the continuity of the channels in the overlapping region to estimate the scale factors. Finally, we reconstruct the signal by combining the estimates from different bands with the appropriate scale factors.

We find that the algorithm's performance depends heavily on the bandwidth of the signal and deteriorates as the signal bandwidth increases. We show that the degradation at large signal bandwidths is because of a fundamental ambiguity where multiple sources can explain the recorded observations. To do this, we perturb the signal estimate in each frequency band by delaying it slightly. The delays in different bands are not the same, but vary in a smooth manner. Thus, even though the signal estimate in each frequency band approximates the true signal very well, the overall reconstruction is very different. Simultaneously, we ensure that the channel estimates satisfy the delay spread constraint, thereby providing multiple explanations for the recorded observations.

Map of this chapter: We begin by describing the related work in Section 4.1 and the system model in Section 4.2. In Section 4.3, we show that the channels can be approximated as piecewise quadratic functions of frequency. We then explain the estimation procedure in each band and the algorithm to estimate the scale factors from overlapping bands. Sections 4.4 and 4.5 provide experimental and simulation

results that quantify the performance of the proposed algorithm. We describe the procedure to construct multiple explanations for the observations when the signal bandwidth is large in Section 4.6.

4.1 Related Work

Recently, acoustic sensor networks have been used to localize a wide variety of sources; for example, sniper detection is investigated in [39], while applications in field biology, such as localizing woodpeckers and marmots, are investigated in [48] and [1] respectively. Acoustic sensing platforms that enable rapid deployability have been developed in [17] and [2].

While our specific interest is in devising practical algorithms for collaborative sensing, we note that the problem posed here falls within a broader framework of blind multichannel identification and equalization, on which there is a large body of literature in the signal processing community (e.g., see [44], [23] for excellent reviews). A large subset of this work is not applicable to our setting, since it assumes prior knowledge of source statistics, such as oversampled second order statistics [46][21], or higher order moments [35]. However, there is a body of work on “deterministic subspace methods” [51] that is indeed directly applicable, since it treats the input as an unknown, deterministic sequence and requires no

assumptions on its statistics. Much of this work focuses on the mathematical structure of the problem, and the key result is that, provided that the channels do not have common zeroes, it is possible to reconstruct the signal. However, signal reconstruction based on extracting common zeros of the recorded signals is infeasible in practice because of its numerical instability and extreme sensitivity to perturbations. Attempts to robustify this algorithm include [45], but this requires statistical assumptions on the input. Reference [51] uses the idea of a “cross-ratio” to formulate a least-squares problem in the time domain which can then be solved using subspace methods. However, this time domain algorithm does not scale to large datasets or large delay spreads, and, as the authors acknowledge, is sensitive to knowledge of channel length. In contrast, our approach achieves scalability by breaking the complex time domain problem into simple subproblems in the frequency domain, while requiring only coarse assumptions on channel length (to estimate the coherence bandwidth). Such parallelism in the frequency domain makes it possible to devise implementable algorithms for sensing applications involving blind deconvolution over dispersive channels; this is analogous to the way in which OFDM simplified communication transceivers by eliminating the need for complex time-domain based channel equalization schemes. Indeed, to the best of our knowledge, the experimental results from our acoustic testbed presented here are the first to demonstrate blind deconvolution in a practical setting involv-

ing long data records and highly dispersive channels.

Source localization and implicit signal estimation is investigated in [9], but these results are for a LOS channel model.

There is a significant literature on correlated sensing for source coding (e.g., [29]) which is not relevant to this work, since we are using correlations to improve signal estimation rather than to save transmission capacity. There is also a large body of work on distributed detection [6, 47], but this is typically based on the assumption that a model for the signal of interest is available, with the focus being to reduce communication costs in fusing sensor observations.

4.2 System Model

We consider a system of S sensors recording distorted and noisy versions of a common source signal $x(t)$. For example, in our indoor acoustic testbed, an acoustic signal is recorded at multiple microphones after multiple bounces off walls and other reflectors. We model the distortion seen by the i th channel as a linear time-invariant channel with impulse response $h_i(t)$, so that the signal recorded by sensor i ($1 \leq i \leq S$) is given by

$$y_i(t) = (x \star h_i)(t) + n_i(t) \quad t \in [0, T]$$

where \star denotes the convolution operation, $n_i(t)$ is noise, and T is the duration of the recording window. We do not have a prior model for the signal $x(t)$ that we wish to extract using our algorithms, hence the only distinction between signal and noise is that we assume that the noise waveforms $n_i(t)$ are uncorrelated across sensors. Thus, if there are undesired signals (which is, after all, what we think of as “noise”) that are correlated across sensors, they would appear at the output of our algorithm. Further, we model the noise waveforms $n_i(t)$ as white and Gaussian. This assumption is not central to our algorithms (which are based on least squares style methods), but if it holds, then maximum likelihood interpretations can be given for certain estimates produced by our algorithms.

We work with discrete time samples of the recorded signals at rate f_s , leading to a received vector of length $N = f_s T$ samples at each sensor. We denote the samples at sensor i by $y_i[n] = y_i(n/f_s)$, $n = 0, 1, \dots, N - 1$. In our acoustic testbed, we typically record over $T \approx 4$ seconds at $f_s = 16000$ Hz, resulting in $N \approx 64000$ samples.

Channel Model: Since the impulse response $h_i(t)$ typically decays rapidly with time, we model it as being time-limited to τ_{max} : $h_i(t) = 0, t \geq \tau_{max}, \forall i$. The equivalent discrete time channel $h_i[n]$ is therefore timelimited to $P = f_s \tau_{max}$ taps. For example, in our testbed, we find that the acoustic channels can be well modeled as timelimited to $\tau_{max} = 100$ milliseconds, corresponding to a length of

$P = 1600$. The discrete time received signal at sensor i is therefore given by

$$y_i[n] = (x \star h_i)[n] + n_i[n], \quad n = 0, 1, 2, \dots, N - 1 \quad (4.1)$$

where \star denotes the *linear* convolution operation between the sequences $\{x[n], n = 0, 1, \dots, N - 1\}$ and $\{h_i[n], n = 0, 1, \dots, P - 1\}$. The noise samples $n_i[n]$ are modeled as Gaussian random variables with variance σ^2 and are assumed to be independent across sensors and time.

Frequency Domain Definitions: We employ frequency domain processing in order to break the “high-dimensional” problem of estimating the signal in the time domain into a number of “low-dimensional” sub-problems that can be solved in parallel. To this end, we take the N -point Discrete Fourier Transform (DFT) of the received samples $y_i[n]$:

$$Y_i[k] = \frac{1}{f_s} \sum_{n=0}^{N-1} y_i[n] \exp(-j2\pi \frac{kn}{N}), \quad k = 0, 1, \dots, N - 1 \quad (4.2)$$

Similarly, we denote the N -point DFT’s of $x[n]$ and $h_i[n]$ by $X[k]$ and $H_i[k]$ ($k = 0, 1, \dots, N - 1$), respectively. Approximating the linear convolution in (4.1) by a circular convolution and taking the DFT of (4.1), we obtain

$$Y_i[k] = X[k]H_i[k] + N_i[k], \quad k = 0, 1, 2, \dots, N - 1 \quad (4.3)$$

where $N_i[k]$ are independent, complex Gaussian random variables with variance σ^2 .

Note on relation between continuous and discrete frequencies: Defining the Fourier transform of the underlying continuous time signal $y_i(t)$ to be $Y_i(f) = \int y_i(t) \exp(-j2\pi ft) dt$, we can interpret the summation in (4.2) as an approximation to the Riemann integral defining $Y_i(f)$. Thus, the discrete frequency index k corresponds to a “physical” frequency of k/T

$$Y_i[k] \approx Y_i(f) \Big|_{f=k/T}. \quad (4.4)$$

For example, assuming a recording window of $T = 4s$, the term $Y_i[200]$ is approximately equal to the contribution from a sinusoid of frequency $f = 200/4s = 50$ Hz to the continuous time signal $y_i(t)$.

Block Processing: The channel responses $H_i(f)$ can be approximated by a constant over “small” frequency bands whose width is termed the channel *coherence bandwidth*, denoted by B_{coh} (the coherence bandwidth is inversely related to the channel delay spread τ_{max}). Thus, the discrete frequency coefficients $H_i[k]$ are approximately constant over $L = B_{coh}T$ contiguous samples. Accordingly, we process the frequency domain samples of the recorded signal $Y_i[k]$ in blocks of L contiguous samples. For example, with a 4 second recording and a coherence bandwidth of 5 Hz, the channels may be approximated by a constant over $L = 5 \text{ Hz} \times 4 \text{ s} = 20$ samples. We can now see the benefits of processing in the frequency domain as opposed to the time domain: instead of processing “large”

blocks (on the order of 64000 samples), we can process many small blocks (of ~ 20 samples) and then “stitch” the small blocks together. Note that the number of samples in a block depends only on the channel coherence bandwidth B_{coh} and the observation interval T , but *not* on the sampling frequency f_s .

Notation: Since our algorithms work on portions of the recorded sequence at a time, it is worth establishing notation (to be followed throughout the rest of the paper) for picking “subsets” of vectors. We use boldface to denote vectors (for eg., \mathbf{u}). We collect the samples of the source sequence $x[n]$, channel to sensor i $h_i[n]$, and the recorded samples $y_i[n]$ in N -dimensional real-valued vectors and denote them by \mathbf{x} , \mathbf{h}_i and \mathbf{y}_i respectively. Similarly, we collect the N -point DFT’s of these quantities in N -dimensional complex-valued vectors denoted by \mathbf{X} , \mathbf{H}_i and \mathbf{Y}_i respectively. We denote the l^{th} element of a vector \mathbf{U} by $\mathbf{U}[l]$. We process the recorded samples in frequency bands consisting of L contiguous indices. We index the frequency bands by b . Suppose that frequency band b corresponds to the indices $\{i_1, i_1 + 1, \dots, i_1 + L - 1\}$. Then, we use $\mathbf{U}[I_b]$ to denote the vector obtained by picking the samples at locations $i_1, i_1 + 1, \dots, i_1 + L - 1$ from \mathbf{U} i.e. $\mathbf{U}[I_b] = (\mathbf{U}[i_1] \ \mathbf{U}[i_1 + 1] \ \dots \ \mathbf{U}[i_1 + L - 1])$.

Example: To illustrate the notation, suppose that the frequency band b corresponds to the physical frequencies 1000 Hz-1005 Hz, with the recording interval being $T = 4$ s. Then, from (4.4), the discrete indices corresponding to band b are

$\{4000, 4001, 4002, \dots, 4019\}$. Therefore, we have $\mathbf{U}[I_b] = (\mathbf{U}[4000], \mathbf{U}[4001], \dots, \mathbf{U}[4019])$. Finally, we use $\mathbf{U}[I_b, l], 0 \leq l \leq L - 1$ to denote the l^{th} element of $\mathbf{U}[I_b]$. For example, with the above definition of band b , $\mathbf{U}[I_b, 2]$ picks out the second element of $\mathbf{U}[I_b]$, which is $\mathbf{U}[4001]$.

4.3 Signal Estimation Algorithm

We have already noted that even a complicated time domain channel exhibits a simple structure over a small enough frequency band. Traditionally (e.g., in wireless transceiver design), it is approximated as constant over its coherence bandwidth. However, for our purpose, a more sophisticated approximation is required: we approximate the channel as a quadratic function of frequency over each frequency block. Additionally, we note that the channel response varies continuously with frequency. We reconstruct the source from the recorded signals in two stages by exploiting these observations, as summarized below.

- *Stage 1:* We split the entire frequency range into small bands over which the channel response may be approximated by a quadratic. We bootstrap by approximating the channel as constant over band b , which allows us to employ an SVD to estimate the source in band b efficiently. This source estimate provides the starting point for an alternating optimization procedure that refines the estimates

of the quadratically varying channels and the source signal.

The alternating optimization consists of multiple iterations of two basic steps:

1. Given an estimate of the source signal in band b , find the best estimates of channel responses to each sensor that are quadratic functions of frequency.
2. Given channel estimates to each sensor in band b that are quadratic functions of frequency, find the best estimate of the source within this band.

- *Stage 2:* Since the signal as well as channels are unknown, there is a scaling ambiguity in the signal estimate produced in stage 1. Thus, in order to reconstruct a signal with bandwidth larger than the channel coherence bandwidth, the estimates from different bands must be scaled consistently. This is accomplished by using overlap between successive bands and exploiting the continuity of the channel. Finally, we use the estimated scale factors along with the signal estimates from Stage 1 to reconstruct the source signal. We now provide the details.

4.3.1 Frequency domain channel model

We motivate our frequency domain channel modeling using a tapped delay line time domain channel model. The impulse response is $h(t) = \sum_k \mu_k \delta(t - \tau_k)$ where $\delta(t)$ is the Dirac delta function, and $0 \leq \tau_k \leq \tau_{max}$. The frequency response $H(f) = \sum_k \mu_k \exp(-j2\pi f \tau_k)$. Consider the channel response over a frequency

band $[f_0 - B_{coh}/2, f_0 + B_{coh}/2]$ where f_0 is the center frequency and B_{coh} is “small”. Expressing a frequency f within band b relative to the center frequency as $f = f_0 + \delta$ ($|\delta| \leq B_{coh}/2$), the channel response at f can be expressed as,

$$H(f_0 + \delta) = \sum_k \mu_k \exp(-j2\pi f_0 \tau_k) \exp(-j2\pi \delta \tau_k) \quad (4.5)$$

$$= \sum_k \mu_k \exp(-j2\pi f_0 \tau_k) [1 - (j2\pi \tau_k) \delta - (2\pi^2 \tau_k^2) \delta^2 + \dots] \quad (4.6)$$

Assuming $|\delta \tau_k| \ll 1$, cubic and higher powers of δ are negligible in the power series expansion of $\exp(-j2\pi \delta \tau_k)$. Since $|\delta| \leq B_{coh}/2$ and $0 \leq \tau_k \leq \tau_{max}$, the condition $|\delta \tau_k| \ll 1$ is guaranteed by choosing $B_{coh} \tau_{max} \ll 1$. Thus, a channel that is time limited to τ_{max} can be approximated as a quadratic function (with complex coefficients) of frequency over any band of width B_{coh} , where $B_{coh} \ll 1/\tau_{max}$. Correspondingly, the discrete frequency samples $H[k]$ are approximated as quadratic functions of k over $L = B_{coh}T$ consecutive samples.

Continuity: For two successive bands overlapping on a set of common frequencies F_{common} , we have

$$H[k] = A_1 + B_1 k + C_1 k^2 \approx A_2 + B_2 k + C_2 k^2 \quad \forall k \in F_{common} \quad (4.7)$$

4.3.2 Stage 1: Estimation within a band

Consider a band b spanning the continuous frequencies $[f_1, f_1 + B_{coh}]$, and corresponding discrete frequencies $I_b = \{f_1 T, f_1 T + 1, f_1 T + 2, \dots, (f_1 + B_{coh})T\}$.

The frequency domain samples at sensor i in this band are given by

$$Y_i[k] = H_i[k]X[k] + N_i[k], \quad k \in I_b.$$

The algorithm begins with a bootstrapped estimate based on approximating the channels as constant, followed by alternating optimization of signal and quadratically fitted channels.

Initial Guess

Approximating the channel gains as constant over the band, we obtain

$$Y_i[k] \approx \bar{H}_i X[k] + N_i[k], \quad k \in I_b \quad (4.8)$$

Following the notation in Section 4.2, we collect the samples within the band in a vector and rewrite (4.8) concisely as

$$\mathbf{Y}_i[I_b] \approx \bar{H}_i \mathbf{X}[I_b] + \mathbf{N}[I_b] \quad (4.9)$$

where $\mathbf{N}[I_b] \sim CN(0, \sigma^2 \mathbb{I})$.

Since the processing in each band is identical, we drop the identity of the band from our notation in the following. Denoting the Maximum Likelihood (ML) estimates of the signal by $\hat{\mathbf{S}}$ and the channel for sensor i by \hat{G}_i , respectively, we have,

$$(\hat{\mathbf{S}}, \hat{G}_i) = \arg \min_{\mathbf{S}, G_i} \sum_{i=1}^S \|\mathbf{Y}_i[I_b] - G_i \mathbf{S}\|^2 \quad (4.10)$$

where S is the number of sensors. An explicit procedure to obtain the estimates becomes apparent with a small change in notation. We store the received signals as the columns of a matrix $\mathbb{Y}[I_b] = [\mathbf{Y}_1[I_b] | \mathbf{Y}_2[I_b] | \dots | \mathbf{Y}_S[I_b]]$ and collect the complex conjugate of the channel gains in an $S \times 1$ vector $\mathbf{G} = (G_1^*, G_2^*, \dots, G_S^*)$.

Then, the cost function in (4.10) can be rewritten as

$$\sum_{i=1}^S \|\mathbf{Y}_i[I_b] - G_i \mathbf{S}\|^2 = \|\mathbb{Y}[I_b] - \mathbf{S} \mathbf{G}^H\|_F^2 \quad (4.11)$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Thus, the channel and the signal estimates are simply the row and column spaces of a rank-one matrix that approximates the recorded data \mathbb{Y} in the best possible manner. We can compute the best rank-one approximation easily by taking a singular value decomposition of \mathbb{Y} and retaining only contribution from the left and right singular vectors corresponding to the largest singular value.

Formally, the ML estimate of the signal $\hat{\mathbf{S}}$ is the eigenvector with the largest eigenvalue of the nonnegative definite matrix

$$\mathbf{M} = \sum_{i=1}^S \mathbf{Y}_i[I_b] (\mathbf{Y}_i[I_b])^H$$

and the corresponding channel estimates are given by $\hat{G}_i = \hat{\mathbf{S}}^H \mathbf{Y}_i[I_b] / \|\hat{\mathbf{S}}\|^2$. However, we note that these ML estimates are unique only up to scaling: $z\hat{\mathbf{S}}$ and $(1/z)\hat{G}_i$ give exactly the same value for the cost function in (4.10) for any complex scalar z .

Alternating Optimization

We now use the preceding signal estimate to refine the channel estimates by capturing their quadratic variation over the bin. These refined channel estimates, in turn, yield an updated signal estimate. This alternating optimization procedure is described in detail below.

Given signal, estimate channels: Based on our quadratic approximation, the i th channel, denoted by $\mathbf{G}_i[l]$, must be expressible in the form,

$$\mathbf{G}_i[l] = A_i + B_i l + C_i l^2 \quad l = 0, 1, \dots, L - 1 \quad (4.12)$$

Given a signal estimate $\hat{\mathbf{S}}$, the channel to the i th sensor, over the band, must satisfy

$$\mathbf{Y}_i[I_b, l] \approx \hat{\mathbf{S}}[l] \mathbf{G}_i[l] + \mathbf{N}_i[I_b, l], \quad l = 0, 1, \dots, L - 1 \quad (4.13)$$

The ML estimate of the i th channel is now obtained by estimating the parameters (A_i, B_i, C_i) as follows:

$$(\hat{A}_i, \hat{B}_i, \hat{C}_i) = \arg \min_{A_i, B_i, C_i} \sum_{l=0}^{L-1} \left| \mathbf{Y}_i[I_b, l] - \hat{\mathbf{S}}[l] (A_i + B_i l + C_i l^2) \right|^2 \quad (4.14)$$

and substituting them into (4.12). This is a linear least squares problem that is solved using standard regression techniques for each sensor.

Given channels, estimate signal: Now, suppose we have channel estimates

$\hat{\mathbf{G}}_i[l] = A_i + B_i l + C_i l^2$. Then the ML signal estimate satisfies

$$\hat{\mathbf{S}}[l] = \arg \min_{\mathbf{S}[l]} \sum_{i=1}^S |\mathbf{Y}_i[I_b, l] - \hat{\mathbf{G}}_i[l] \mathbf{S}[l]|^2 \quad (4.15)$$

The solution is a “maximal ratio combining” rule (often used to generate decision statistics for multipath channels):

$$\hat{\mathbf{S}}[l] = \frac{\sum_{i=1}^S \hat{\mathbf{G}}_i^*[l] \mathbf{Y}_i[I_b, l]}{\sum_{i=1}^S |\hat{\mathbf{G}}_i[l]|^2} \quad (4.16)$$

Output of Stage 1

After a few rounds of alternating optimization, we have an estimate of the signal in each band b , denoted by $\hat{\mathbf{S}}_b$, and channels to all the sensors, denoted by $\hat{\mathbf{G}}_{i,b}$. Due to the scale factor ambiguity, the estimates $\hat{\mathbf{S}}_b$ and $\hat{\mathbf{G}}_{i,b}$ are related to the true signal $\mathbf{X}[I_b]$ and the channels $\mathbf{H}_i[I_b]$ in band b by an arbitrary complex number z_b , so that,

$$\begin{aligned} z_b \hat{\mathbf{S}}_b &\approx \mathbf{X}[I_b] \\ 1/z_b \hat{\mathbf{G}}_{i,b} &\approx \mathbf{H}_i[I_b] \end{aligned} \quad (4.17)$$

In Stage 2, we use the continuity of the channel’s frequency response across bands to estimate the scale factors z_b and “stitch” together the signal estimates from different bands.

4.3.3 Stage 2: L-to-R Stitching Algorithm

We estimate the weights z_b by choosing adjacent bands to have significant overlap with one another, and enforcing consistency in the overlapped region.

Reconciling multiple estimates of the same quantity: Consider adjacent bands $b-1$ and b , as shown in Figure 4.1. Denote the frequencies common to bands

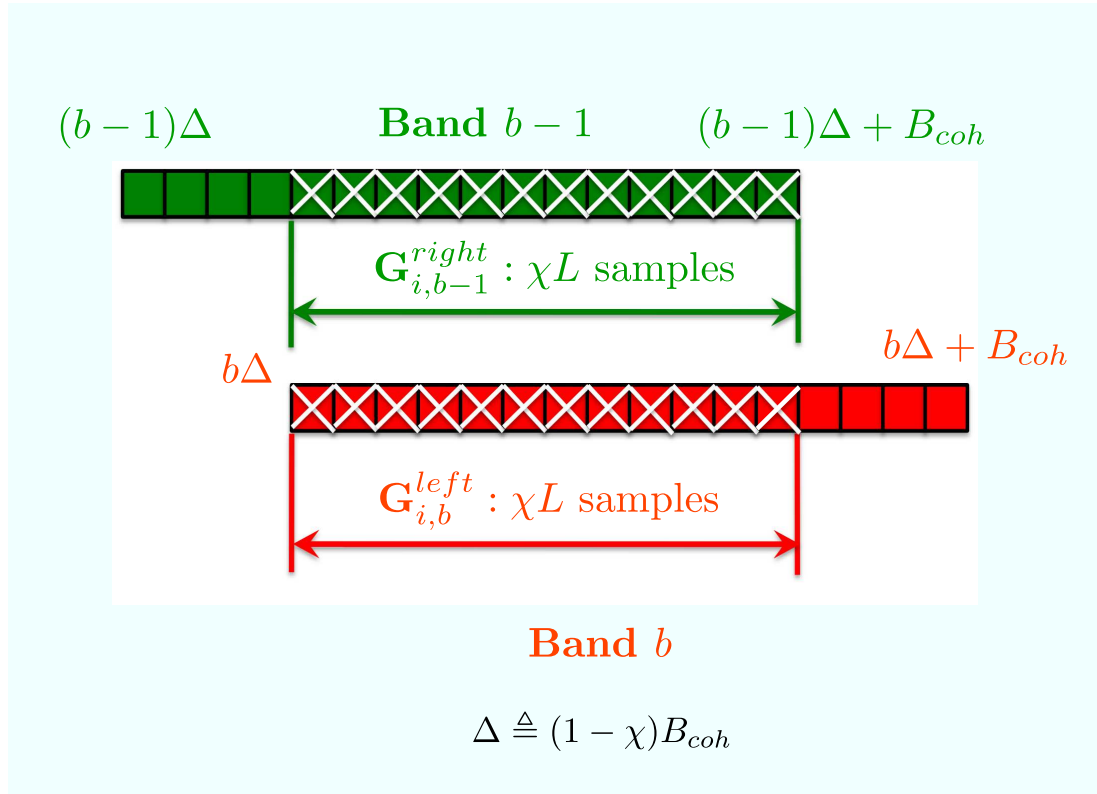


Figure 4.1: The figure shows our choice of overlapping frequency bands. The signal and the channel estimate samples in the crossed squares are used to determine the scale factors z_b in different bands.

$b-1$ and b by F_{common} i.e. $F_{common} = [b\Delta, (b-1)\Delta + B_{coh}]$ with $\Delta = (1 - \chi)B_{coh}$.

The parameter $\chi \in (0, 1)$ controls the amount of overlap between adjacent bands.

We obtain *two* estimates of the channel to sensor i (upto a scale factor) within F_{common} from Stage 1. The first estimate is obtained from band $b - 1$ by picking the *rightmost* χL entries of $\hat{\mathbf{G}}_{i,b-1}$. The chosen entries are shown by crossed squares in Figure 4.1 and we denote this estimate by $\hat{\mathbf{G}}_{i,b-1}^{right}$. The second estimate is obtained from band b by picking the *leftmost* χL entries of $\hat{\mathbf{G}}_{i,b}$ and is denoted by $\hat{\mathbf{G}}_{i,b}^{left}$. Denoting the true channel to sensor i within F_{common} by $\mathbf{H}_{i,common}$, we have,

$$(1/z_{b-1})\hat{\mathbf{G}}_{i,b-1}^{right} \approx \mathbf{H}_{i,common} \quad (4.18)$$

$$(1/z_b)\hat{\mathbf{G}}_{i,b}^{left} \approx \mathbf{H}_{i,common} \quad (4.19)$$

Equating the left hand sides of (4.18) and (4.19), we get a constraint on the channel estimates from stage 1 to each sensor:

$$z_b \hat{\mathbf{G}}_{i,b-1}^{right} \approx z_{b-1} \hat{\mathbf{G}}_{i,b}^{left} \quad \forall i = 1, 2, \dots, S \quad (4.20)$$

In an exactly analogous fashion, the signal estimates from stage 1 in bands $b - 1$ and b also provide consistency conditions. Denote the *leftmost* χL entries of $\hat{\mathbf{S}}_b$ by $\hat{\mathbf{S}}_b^{left}$ and the *rightmost* χL entries of $\hat{\mathbf{S}}_{b-1}$ by $\hat{\mathbf{S}}_{b-1}^{right}$. From (4.17), we get,

$$z_{b-1} \hat{\mathbf{S}}_{b-1}^{right} \approx z_b \hat{\mathbf{S}}_b^{left} \quad (4.21)$$

We combine the constraints in (4.20) and (4.21) to obtain a single constraint of the form,

$$z_b \mathbf{u}_{b,b-1} \approx z_{b-1} \mathbf{u}_{b-1,b} \quad (4.22)$$

where the vectors $\mathbf{u}_{b,b-1}$ and $\mathbf{u}_{b-1,b}$ are given by $\mathbf{u}_{b,b-1} = (\hat{\mathbf{S}}_b, \hat{\mathbf{G}}_{1,b-1}^{right}, \dots, \hat{\mathbf{G}}_{S,b-1}^{right})$ and $\mathbf{u}_{b-1,b} = (\hat{\mathbf{S}}_{b-1}, \hat{\mathbf{G}}_{1,b}^{left}, \dots, \hat{\mathbf{G}}_{S,b}^{left})$.

We estimate the scale factors $\{z_b\}$ in two steps. First, we use (4.22) to calculate the “relative” scale factor $\gamma_{b,b-1} \triangleq z_b/z_{b-1}$ between bands b and $b-1$. Then, we recursively obtain the weights $\{z_b\}$ from $\{\gamma_{b,b-1}\}$.

Step 1: Assuming that the errors $z_b \mathbf{u}_{b,b-1} - z_{b-1} \mathbf{u}_{b-1,b}$ are Gaussian distributed, the ML estimate of $\gamma_{b,b-1} = z_b/z_{b-1}$ is given by,

$$\gamma_{b,b-1} = \frac{\mathbf{u}_{b,b-1}^H \mathbf{u}_{b-1,b}}{\|\mathbf{u}_{b,b-1}\|^2} \quad (4.23)$$

Step 2: We can only hope to recover the signal upto a scale factor. For example, even in the noiseless case, a signal estimate $\hat{\mathbf{s}} = \alpha \mathbf{x}$ and channel estimates $\hat{\mathbf{h}}_i = \frac{1}{\alpha} \mathbf{h}_i$ would also explain the received data, for any $\alpha \neq 0$. Thus, without loss of generality, we can set the weight of one band, say the first band z_0 , to 1. This sets the “scale” of the reconstructed signal. Then, we recursively estimate the scale factors in other bands as

$$z_b = z_{b-1} \gamma_{b,b-1} \quad b \geq 1$$

We call this the *L-to-R stitching algorithm* since the weights are estimated in a recursive fashion, starting from low frequencies (left end of the spectrum) and proceeding on to high frequencies (right end of the spectrum).

Handling low energy bands

So far, we have assumed that the source signal has sufficient energy in all the bands, guaranteeing trustworthy estimates of the channels and the signal in each band. However, if the signal energy in a given frequency band is too low, neither the signal nor the channel estimates will be reliable, which can severely disrupt the stitching procedure. Hence, as described in the following, we identify bands with low signal energy and omit them from our processing, giving them a weight of zero when reconstructing the signal.

To estimate the received signal energy in band b , we approximate the channel to sensor i within band b by a constant i.e. $\mathbf{H}_i[I_b, l] \approx \bar{H}_i \forall l = 0, 1, \dots, L - 1$. Thus, the received signals in band b can be approximated as

$$\mathbf{Y}_i[I_b] \approx \bar{H}_i \mathbf{X}[I_b] + \mathbf{N}_i[I_b], \quad \forall i = 1, 2, \dots, S \quad (4.24)$$

with $\mathbf{N}_i[I_b] \sim CN(0, \sigma^2 \mathbb{I})$. With this model, the total signal energy received at all the sensors in band b is given by,

$$E_{sig,b} = \left(\sum_i |\bar{H}_i|^2 \right) \|\mathbf{X}[I_b]\|^2 \quad (4.25)$$

To estimate the received energy from the recorded signals, we compute the eigenvalues $\lambda_{1,b} \geq \lambda_{2,b} \geq \dots \lambda_{S,b}$ of

$$\mathbf{M} = \sum_{i=1}^S \mathbf{Y}_i[I_b] (\mathbf{Y}_i[I_b])^H$$

We have

$$\lambda_{1,b} \approx E_{sig,b} + \sigma^2 \tag{4.26}$$

$$\lambda_{2,b} \approx \lambda_{3,b} \approx \dots \lambda_{S,b} \approx \sigma^2 \tag{4.27}$$

We therefore estimate the signal energy in band b to be

$$\hat{E}_{sig,b} = \lambda_{1,b} - \frac{\lambda_{2,b} + \lambda_{3,b} + \dots + \lambda_{S,b}}{S - 1} \tag{4.28}$$

Good and Bad Bands: Let $\hat{E}_{sig,max} = \max_b \hat{E}_{sig,b}$ be the maximum value of the estimated signal energy across bands. We declare band b to be “good” if the signal energy in this band exceeds a fraction η ($0 < \eta < 1$) of the maximum $\hat{E}_{sig,max}$ i.e. if $\hat{E}_{sig,b} > \eta \hat{E}_{sig,max}$. Otherwise, we declare band b to be “bad” and the signal estimate from band b in Stage 1 is given no weight ($z_b = 0$) while reconstructing the source signal.

Overall algorithm

We now summarize the L-to-R Stitching algorithm to obtain the scale factors z_b .

1. For each band b , determine whether it is good or bad. Let Λ_{good} denote the set of good bands.
2. *Initialize:* Let $b_0 = \min \Lambda_{good}$ be the “first” good band. Set $z_{b_0} = 1$. For every subsequent band, we do the following:
3. *Bad bands:* If band b is bad, set $z_b = 0$, so that it does not contribute to the reconstructed signal.
4. *Good bands:* If band b is good, there are two possibilities:
 - If band $b - 1$ is also good, we have consistency conditions that allow us to “stitch” $\hat{\mathbf{S}}_b$, the estimate from band b , to the estimates from previous bands $0, 1, \dots, b - 1$. To do this, we compute $\gamma_{b,b-1} = \frac{\mathbf{u}_{b,b-1}^H \mathbf{u}_{b-1,b}}{\|\mathbf{u}_{b,b-1}\|^2}$ and set $z_b = \gamma_{b-1,b} z_{b-1}$.
 - *Re-initialize:* If band $b - 1$ is bad, so that $z_{b-1} = 0$, we do not have trustworthy, “local” consistency conditions to stitch $\hat{\mathbf{S}}_b$ to the estimates from previous bands. Therefore, we are forced to “re-initialize” the weight computations, set $z_b = 1$ and go back to step 2.

Impact of “holes” in source spectrum: The need for reinitialization when the signal energy falls below a threshold means that holes in the source spectrum create ambiguities in reconstruction. Figure 4.2 depicts a low energy band b_{low} flanked by bands with significant energy on both sides. For each of these flanking

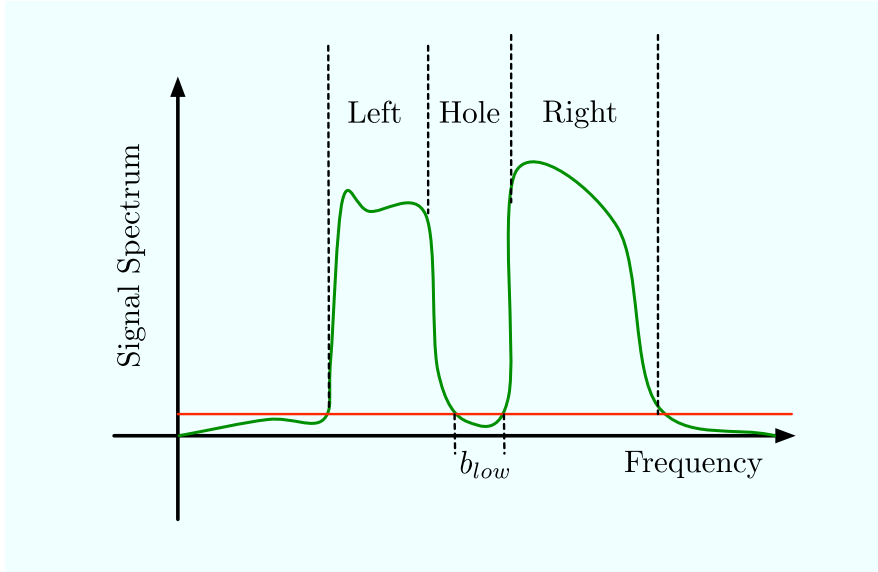


Figure 4.2: Illustrating a “hole” in the signal spectrum. Two bands of high energy flank a band with relatively low energy. The reconstruction procedure works fine in the flanking bands individually. However, the overall reconstruction is poor due to the loss in continuity because of the low energy band in between.

bands, our stitching procedure works fine. However, we cannot stitch these two bands together because of the scaling ambiguity left by the hole. Thus, we expect our source signal to be reproduced well within each band, but not when the bands are combined together.

4.3.4 Reconstructing the source signal

We can now reconstruct the source signal over the entire frequency range by (1) taking the signal estimates from different bands in Stage 1, (2) weighing the estimate in band b by z_b calculated in Section 4.3.3 and (3) adding the resulting

signals “in situ” (i.e. in the appropriate frequency bands). The exact reconstruction procedure is as follows,

Initialize: $\hat{\mathbf{X}} \leftarrow \mathbf{0}$

Iterate: **for** ($b = 0; b \leq B - 1; b++$)

Update Band b : $\hat{\mathbf{X}}[I_b] \leftarrow \hat{\mathbf{X}}[I_b] + z_b \hat{\mathbf{S}}_b;$

end

We obtain an estimate of the source in the time domain, denoted by $\hat{\mathbf{x}}$, by applying an inverse Fourier transform to $\hat{\mathbf{X}}$.

4.4 Experimental Results

We use a Logitech Z5 omnidirectional speaker as the acoustic source and record data using three Samson C03U USB microphones set omnidirectional receive patterns. We play each source signal from four different locations to emulate recordings with a larger number of microphones. Thus, we have twelve recordings (4 source locations \times 3 microphones) of each source, emulating the effect of 12 sensors with correlated observations.

We use four source signals to test the performance of the L-to-R algorithm:

(i) *chirp* signals with bandwidths 50 Hz and 200 Hz, denoted by Chirp50 and

Chirp200; and (ii) sinusoids, spaced 2 Hz apart, over bands of width 50 Hz and 200 Hz, denoted by Sines50 and Sines200. We now describe each of these sources in detail. A chirp signal, whose instantaneous frequency increases linearly from f_0 Hz to f_1 Hz over a time window of T seconds, is given by,

$$x_{chirp}(t) = \cos \left(2\pi \left(f_0 + \frac{f_1 - f_0}{T} t \right) t \right) \quad 0 \leq t \leq T \quad (4.29)$$

Both Chirp50 and Chirp200 last for $T = 4$ seconds and begin at $f_0 = 1000$ Hz. While Chirp200 goes up to a frequency $f_1 = 1200$ Hz, Chirp50 only goes up to $f_1 = 1050$ Hz. Sines50 and Sines200 are the sum of sinusoids spaced 2 Hz apart in the bands 1000 Hz-1050 Hz and 1000 Hz-1200 Hz respectively. The individual sinusoids have the same amplitude; their phases rise linearly from 0 at the start of the band to $\pi/2$ in the middle of the band and then fall back to 0 at the end of the band. We use a sampling frequency $f_s = 16000$ Hz to record the data.

Speaker-microphone response: We first conducted an *outdoor* experiment to characterize the distortions introduced (if any) by the speaker and the microphone, and separate them from the effects of the indoor propagation channel. We placed the speaker on a table in an open lawn, to avoid reflections, and positioned the microphone right in front of the speaker. In this Line-of-Sight (LoS) environment, the normalized correlations between Chirp200 and its recording on three occasions were 0.9940, 0.9967 and 0.9928 [where a value of 1 indicates a perfect

match]. This suggests that the speaker and microphone are nearly ideal in the band of interest and can be ignored in subsequent analyses. Furthermore, any deviations in the recorded signal from the ground truth can only be due to the propagation environment.

Indoor acoustic channel: The autocorrelation function of Chirp200 is reasonably “spiky”: it takes sizable values only when the Chirp200 waveform is relatively well aligned with itself. Thus, we can obtain a coarse estimate of the delay spread of the indoor propagation channel (which is all that our algorithm needs) by simply crosscorrelating the recording at one of the sensors with time-shifted versions of the true Chirp200 waveform. We plot the crosscorrelation in Figure 4.3 and observe that it takes relatively large values over a time window of 100 ms. This suggests that the delay spread of the channel is on the order of 100 ms with a corresponding coherence bandwidth of about 5-10 Hz.

Preprocessing: The low frequency components in the recorded signals contain substantial energy from background hum. Since the hum is registered at all the sensors, it counts as “signal” rather than spatially uncorrelated “noise.” In order to investigate the performance of our algorithm, we would like to control the source signal being sensed, hence we filter out the received signal energy in the bands from 0-950 Hz to eliminate the hum. We then coarsely synchronize the recorded waveforms in a data-driven fashion: we pick a reference sensor at ran-

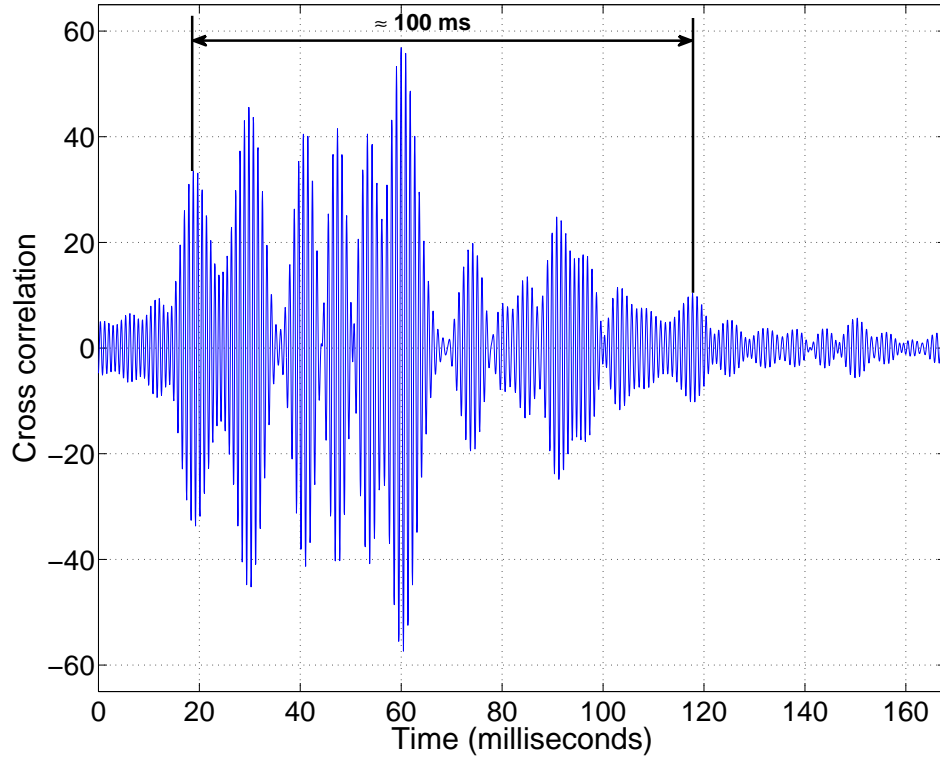


Figure 4.3: An estimate of the indoor propagation channel

dom, matched-filter the recorded waveforms at other sensors against the reference waveform, and shift them in time so that they are “aligned” with the reference waveform. Finally, the recordings at one of the twelve “sensors” - corresponding to a particular source-microphone arrangement - had a very good correlation with the true signal, indicating that it had a near Line-of-Sight channel to the source. Since our goal is to understand the limits of recovering signals in the face of significant multipath, we exclude this sensor from further processing.

Parameters & Results: Guided by our coarse characterization of the indoor channel delay spread, we set $B_{coh} = 5$ Hz. We choose the overlap between adjacent bands χ to be 50%. We declare a band to be good if its energy is greater than $(1/10)^{th}$ of the band with maximum energy i.e. $\eta = 0.1$. Note that there is always a “global” delay ambiguity in the estimate: we can delay the signal by τ and correspondingly *advance* all the channels by τ to explain the received data. Therefore, we quantify the performance of the L-to-R Stitching algorithm by the normalized crosscorrelation, denoted by ρ_{L-to-R} , between the source \mathbf{x} and its estimate $\hat{\mathbf{x}}$, maximized over all possible time-shifts τ of the estimate:

$$\rho_{L-to-R} = \max_{\tau} \frac{(\mathcal{D}^{\tau} \hat{\mathbf{x}})^T \mathbf{x}}{\|\mathbf{x}\| \|\hat{\mathbf{x}}\|} \quad (4.30)$$

where \mathcal{D} is the delay operator that shifts the vector \mathbf{x} by one sample. We note that $0 \leq \rho_{L-to-R} \leq 1$ with $\rho_{L-to-R} = 1$ indicating a perfect match between \mathbf{x} and $\hat{\mathbf{x}}$. We compare the L-to-R output against a solution that ignores multipath and approximates the channel to consist only of a single tap. Since the optimal estimate with this approximation is given by an SVD of the received signals (this can be shown in an identical fashion to the proof in Section 4.3.2), we refer to this estimate as the “SVD estimate” and denote its correlation with the truth by ρ_{SVD} . We compare these estimates in Table 4.1 and make the following observations:

- The L-to-R algorithm consistently performs better than the SVD estimate.

Therefore, accounting for the multipath channel and piecing together estimates from different bands yields a better reconstruction even when the signal bandwidth is relatively small (~ 50 Hz).

- We illustrate the gains provided by the L-to-R stitching algorithm visually, by plotting the true Chirp200 waveform, the corresponding recorded signals at four sensors and the estimated signal in Figure 4.4. The true chirp waveform, in the topmost plot, has a constant envelope. In the four subsequent plots, we show the recorded signals at four sensors (chosen at random out of the 11 sensors used in the processing). It is clear that the recorded signals are significantly different from the source signal and do not have a constant envelope. Next, we note that the chirp signal can be considered as a sinusoid with time-varying frequency $f(t)$ and constant envelope. Thus, the amplitude of the recorded signal at time t approximately equals the channel gain at frequency $f(t)$, and the time domain envelope of the recorded signal gives a coarse estimate of the channel frequency response in the signal band. From the variations in the envelope, we see that all the channels experience “deep fades”, where the channel response at some frequencies is close to zero. The reconstructed waveform is shown in the last plot and we see that it exhibits significantly smaller variations in its envelope and resembles the signal to a greater degree, reflected by an increased correlation coefficient with the true source (0.82 for the reconstruction vs a maximum of 0.68 at any individual

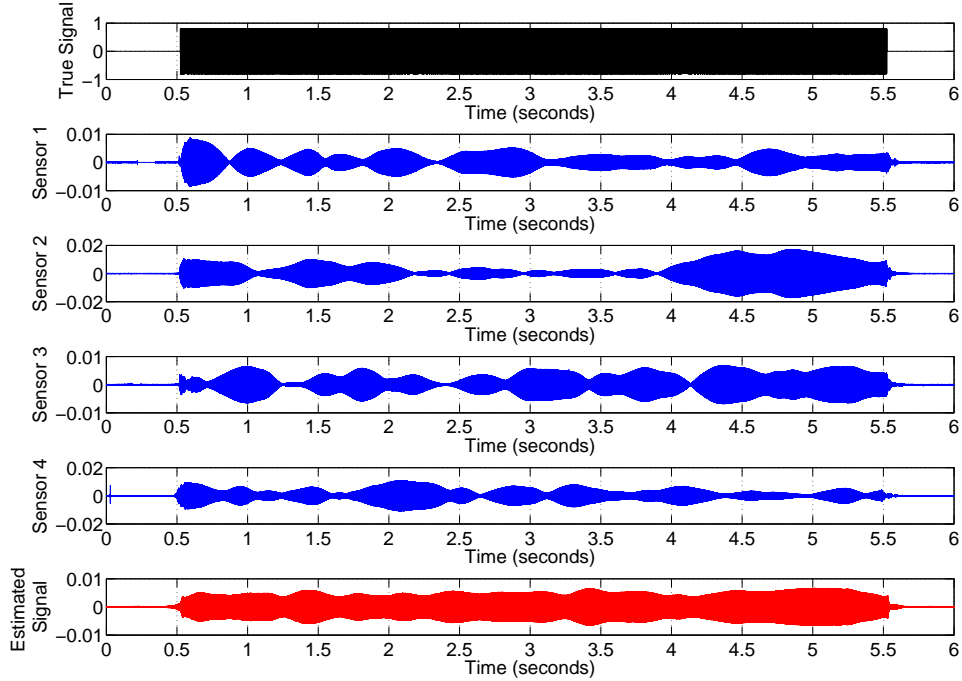


Figure 4.4: The topmost plot shows the true Chirp200 waveform, with a constant envelope. The following four plots show the recorded waveforms at different sensors. Notice that these waveforms undergo “deep fades” and no longer have a constant envelope. The final plot shows the reconstructed Chirp200 waveform, whose envelope shows lesser variation, illustrating the benefits of the L-to-R algorithm.

sensor).

- From Table 4.1, we observe that the reconstruction is very good when the signal bandwidth is small and worsens as the bandwidth increases. For example, the reconstruction is nearly perfect when the source is Chirp50, with $\rho_{L-to-R} = 0.97$. In contrast, when the bandwidth of the chirp signal is increased to 200 Hz,

Signal	ρ_{L-to-R}	ρ_{SVD}
Chirp50	0.97	0.84
Chirp200	0.82	0.7
Sines50	0.87	0.72
Sines200	0.68	0.56

Table 4.1: Results of L-to-R processing and single tap approximation of the recorded signals.

ρ_{L-to-R} drops to 0.82. Similarly, ρ_{L-to-R} is 0.87 for the Sines50 source and drops to 0.68 when the bandwidth is increased to 200 Hz.

- To understand why reconstruction is poorer as signal bandwidth increases, we investigate how well a source of bandwidth 200 Hz is reconstructed over smaller bands of width 50 Hz. To do this, we filter the source \mathbf{x} and the estimate $\hat{\mathbf{x}}$, so that they possess energy only in a 50 Hz band, say 1000 Hz-1050 Hz. We then use the correlation metric in (4.30) to quantify the fit between the *filtered versions* of \mathbf{x} and $\hat{\mathbf{x}}$. From the results in Table 4.2, we see that (a) the fit over bands of width 50 Hz is very good (the chirp signal is reconstructed nearly perfectly over each 50 Hz-band with $\rho_{L-to-R} \approx 0.98$) and (b) for both signals, the quality of the fit, measured by ρ_{L-to-R} , is remarkably consistent across bands.

Digging deeper, we find that the delay between the reconstructed signal (over 50 Hz-bands) and the true signal in these bands varies across bands (see Table 4.3). We conjecture, therefore, that these delay variations across 50 Hz-bands cause the signal contributions from these bands to combine “incoherently”, thereby affecting

Signal	Band 1	Band 2	Band 3	Band 4
Chirp200	0.982	0.952	0.983	0.987
Sines200	0.882	0.859	0.869	0.847

Table 4.2: Fit between source and estimate in bands of width 50 Hz is very good. Band i spans the frequencies $[1000 + 50(i - 1), 1000 + 50i]$ Hz.

the quality of the reconstruction over a larger band. In Section 4.6, we show that delay variations are indeed the cause of fundamental ambiguities, and can be used to systematically construct multiple explanations of the recorded data. In

	Band 1	Band 2	Band 3	Band 4
Chirp200	287	203	123	172
Sines200	320	214	156	164

Table 4.3: Delay between the true source and the estimate over bands of width 50 Hz (in samples @ $f_s = 16$ kHz). We see that the estimates in different bands have different delays with respect to the source. Band i spans the frequencies $[1000 + 50(i - 1), 1000 + 50i]$ Hz.

addition to our indoor experiments, we simulated the L-to-R algorithm extensively to characterize its performance statistically and investigate its robustness with channel variations. We report these results in the next section.

4.5 Simulation Results

We simulated the performance of the L-to-R algorithm with two *classes* of signals, which we call chirp and “random”. We consider three signals within each class with different bandwidths. All the chirp signals have the general form spec-

ified in (4.29), beginning at $f_0 = 1000$ Hz and lasting for $T = 4$ seconds. The bandwidths considered are 50 Hz, 200 Hz and 300 Hz, and we denote these signals by Chirp50, Chirp200 and Chirp300 respectively. The “random” signals lie in the band $[1000, 1000 + W]$ where the bandwidth of the signal W , once again, takes the values 50 Hz, 200 Hz and 300 Hz. We denote these signals by Random50, Random200 and Random300 respectively. We describe the construction of Random200; the other Random signals are generated by a similar procedure. Random200 lies in the frequency band from 1000 Hz to 1200 Hz and lasts for $T = 4$ seconds. The signal amplitudes and phases are chosen randomly at each of the discrete frequency indices $\{4000, 4001, 4002, \dots, 4799\}$ corresponding to this band: the amplitude is a Gaussian random variable with unit variance and the phase is uniform in $[0, 2\pi]$. We simulate a setting with $S = 6$ sensors using a sampling frequency of $f_s = 16$ kHz. The channel to each of these sensors is generated at random as follows: it consists of fifteen taps located uniformly within a delay spread τ_{max} of 20 ms. The tap amplitudes are chosen uniformly between -2 and 2. We add independent Gaussian noise to each recording so that the signal-to-noise ratio (SNR) is 5 dB. We choose a coherence bandwidth $B_{coh} = 5$ Hz for the processing in Stage 1 with 70% overlap between the bins ($\chi = 0.7$). We run 100 trials of the algorithm with the channel realizations varying randomly across trials. We compare the performance of the L-to-R algorithm and the SVD Esti-

Signal	$\rho_{av,L-R}$	$\rho_{av,SVD}$	$\rho_{min,L-R}$	$\rho_{min,SVD}$
Random50	0.978	0.926	0.932	0.69
Chirp50	0.977	0.925	0.939	0.722
Random200	0.946	0.738	0.876	0.539
Chirp200	0.942	0.713	0.827	0.535
Random300	0.914	0.670	0.744	0.490
Chirp300	0.925	0.664	0.807	0.462

Table 4.4: Performance of the L-to-R stitching algorithm and the SVD Estimate with Chirp and “Random” signals of varying bandwidths.

mate by the *average* of the correlation coefficients ρ_{L-to-R} and ρ_{SVD} across trials. We also quantify the robustness of these estimates by the *minimum* values of the corresponding correlation coefficients over the 100 trials. We present the results in Table 4.4 and make the following observations:

- **Small signal bandwidth:** Consider the signals of bandwidth 50 Hz. As in our experiments, we see that the L-to-R stitching algorithm produces nearly perfect reconstructions on the average when the bandwidth of the signal is small. For example, with the Chirp50 signal, the average correlation $\rho_{av,L-to-R}$ is 0.977. Furthermore, the L-to-R algorithm is robust, with the minimum correlation over 100 trials being large; for example, with the Random50 signal, $\rho_{min,L-to-R} = 0.932$. The SVD estimate also performs fairly well on the average with $\rho_{av,SVD} = 0.925$ for the Chirp50 signal. This is reasonable: since the bandwidth of the signal is small, the effects of frequency selective fading are not very pronounced. However, the SVD estimate is not robust, with $\rho_{min,SVD} = 0.69$ for the Random50 signal.

This indicates that ignoring the dispersive nature of the channels can occasionally lead to poor reconstruction, even over small bands.

- **Moderate signal bandwidth:** When the signal bandwidth increases to 200 Hz, the L-to-R algorithm continues to perform well on the average and is reasonably robust. The average correlation $\rho_{av,L-to-R}$ is still large (≈ 0.94) for both the random and chirp signals. The minimum correlation $\rho_{min,L-to-R}$ is lower than in the small bandwidth case, but is still fairly high ($\approx 0.83-0.87$) for both signals. However, the SVD estimator performs very poorly over such bandwidths; even the average correlation $\rho_{av,SVD}$ is only on the order of 0.72. This demonstrates the need to account for multipath propagation as the signal bandwidth increases.

- **Large signal bandwidth:** For signals with large bandwidths (300 Hz), the L-to-R estimator continues to perform well on the average with $\rho_{av,L-to-R} \approx 0.91$ for both Chirp300 and Random300 signals. However, we observe occasional glitches in the performance with the L-to-R estimator : $\rho_{min,L-to-R}$ drops to 0.74 with the Random300 signal.

We draw two conclusions from the simulation results: (1) The L-to-R stitching algorithm is robust to channel variations when the bandwidth of the signal is a small ($\sim 10x$) multiple of the coherence bandwidth. (2) The statistical results of the simulations are in excellent agreement with the experimental findings: both

predict excellent performance for the L-to-R algorithm over small bandwidths and deterioration in the performance over larger bandwidths.

4.6 Multiple Explanations

In this section, we show that the deterioration in the normalized correlation between the reconstructed signal and the true signal that we discovered from our experiments and simulations is actually because of a fundamental ambiguity caused by delay variations across bands. In particular, we show that multiple source estimates which are considerably different from the true source can explain the received data, while respecting constraints on the channel delay spreads, when the bandwidth of the signal becomes “large.” Since our L-to-R stitching algorithm does not account for time domain constraints explicitly (it reproduces the signal well, but need not produce channel estimates that fall within the hypothesized delay spreads), we propose a “global stitching” algorithm which respects these time domain constraints, and show that multiple explanations can still be constructed by making small perturbations of the delay across adjacent bands as we stitch them together, as follows:

- *Step (a)*: As before, we estimate the signals and channels in each coherence band with the alternating optimization algorithm from Stage 1 (see Section 4.3.2),

and denote the signal estimate in band b by $\hat{\mathbf{S}}_b$ and the channel to sensor i by $\hat{\mathbf{G}}_{i,b}$. Suppose that these estimates explain the data in band b accurately. We perturb the solution in band b by delaying the signal estimate $\hat{\mathbf{S}}_b$ by τ_b samples. The channel estimates $\hat{\mathbf{G}}_{i,b}$ are correspondingly advanced by τ_b samples, ensuring that the recorded data in each band continues to be explained well. By choosing the delays τ_b to vary with the band index b , we can ensure that the reconstructed source does not combine “coherently” across bands with the true source, thereby leading to a poor reconstruction.

- *Step (b)*: The challenge now is to use the *perturbed versions* of the signal and channel estimates from different bands to produce an estimate of the source and channels over the *entire* band that satisfies all constraints. We propose an algorithm to choose the scaling coefficients z_b , to be assigned to band b , so that the two available constraints are met: (i) the reconstructed channels must be restricted to the given delay spread $P = f_s \tau_{max}$ and (ii) the recorded data must be matched as closely as possible by the source and channel estimates. The proposed algorithm is more complex than the L-to-R stitching algorithm, since it is forced to exploit constraints that are “global”; however, its utility lies in the fact that it provides alternate explanations that satisfy all the constraints (the L-to-R algorithm does not provide such guarantees) and thereby, demonstrates fundamental

ambiguities. We refer to this algorithm as the *global stitching algorithm*. We now explain each step of the processing involved in obtaining alternate solutions.

4.6.1 Distorting the outputs of Stage 1

Denote the signal estimate in band b at the output of Stage 1 by $\hat{\mathbf{S}}_b = (\hat{\mathbf{S}}_b[0], \hat{\mathbf{S}}_b[1], \dots, \hat{\mathbf{S}}_b[L-1])$ and the corresponding channel estimate to sensor i by $\hat{\mathbf{G}}_{i,b} = (\hat{\mathbf{G}}_{i,b}[0], \hat{\mathbf{G}}_{i,b}[1], \dots, \hat{\mathbf{G}}_{i,b}[L-1])$. Suppose that we wish to delay the signal estimate from band b by τ_b samples (in the time domain). In the frequency domain, this corresponds to a “distorted” signal estimate $\hat{\mathbf{S}}_b^\#$, whose l^{th} sample is given by,

$$\hat{\mathbf{S}}_b^\#[l] = \hat{\mathbf{S}}_b[l] \exp\left(-j2\pi \frac{l\tau_b}{N}\right), \quad l = 0, 1, \dots, L-1 \quad (4.31)$$

where $N = f_s T$ is the total number of received samples. We advance all the channel estimates by τ_b to ensure that the recorded signals in band b continue to be explained well and obtain “distorted” channel estimates $\hat{\mathbf{G}}_{i,b}^\#$ satisfying,

$$\hat{\mathbf{G}}_{i,b}^\#[l] = \hat{\mathbf{G}}_{i,b}[l] \exp\left(j2\pi \frac{l\tau_b}{N}\right) \quad l = 0, 1, \dots, L-1 \quad (4.32)$$

We choose the delays τ_b to satisfy the following properties:

- **Not too large:** The distorted channels $\hat{\mathbf{G}}_{i,b}^\#[l]$ must also be quadratic functions of the frequency index l , otherwise they cannot be explained by the global stitching algorithm in Step (b) with the allowed delay spread. Therefore, the

terms $\exp(-j2\pi\frac{l\tau_b}{N})$, $l = 0, 1, \dots, L - 1$ must not vary significantly with l . This can be achieved by choosing $L\tau^*/N \ll 1$, where $\tau^* = \max_b |\tau_b|$ is the maximum injected delay across bands. In our simulations, we choose $L\tau^*/N \approx 0.05$.

- **Substantial variation:** To produce an “alternate explanation” that differs substantially from the true source, we need “substantial” variation in τ_b across bands. In our simulations, we choose the variation $\max_b \tau_b - \min_b \tau_b$ to be 300 samples (at 16 kHz, with a signal of bandwidth 200 Hz). A sinusoidal variation (smooth yet tracing out the entire allowable “perturbation budget”) is found to be effective.

We now describe the global stitching algorithm which combines the distorted channel and signal estimates from different bands, to produce corresponding estimates over the entire frequency range. In this process, we ensure that all available constraints are satisfied.

4.6.2 Global Stitching Algorithm

Consider the distorted channel estimate $\hat{\mathbf{G}}_{i,b}^\#$ from a “good” band b . Since the perturbations in each band are not too large, the distorted channel estimate $\hat{\mathbf{G}}_{i,b}^\#$ can still be approximated by a scaled version of the true channel $\mathbf{H}_i[I_b]$, so that,

$$\hat{\mathbf{G}}_{i,b}^\# \approx \frac{1}{z_b} \mathbf{H}_i[I_b], \quad b \in \Lambda_{good} \quad (4.33)$$

Equivalently, given a hypothetical weight z_b , the reconstructed channel in band b is given by $z_b \hat{\mathbf{G}}_{i,b}^\#$. These weights must be chosen so that the corresponding channel in the time domain \mathbf{g}_i satisfies three properties: (1) It is real-valued. (2) The channel length is restricted to $P = f_s \tau_{max}$ samples where τ_{max} is the delay spread of the channel (i.e. \mathbf{g}_i is a P -dimensional vector). (3) In a “good” band b , the Fourier transform of the channels \mathbf{g}_i must roughly be equal to the hypothesized channel $\hat{\mathbf{G}}_{i,b}^\# z_b$.

Denote the Fourier matrix by $F \in \mathbb{C}^{N \times P}$ with $F(m, p) = \exp(-j2\pi \frac{mp}{N})$ and $0 \leq p \leq P - 1$ and $0 \leq m \leq N - 1$. Let I_b denote the indices corresponding to band b . Then, F_b denotes the submatrix obtained by picking the rows of F contained in I_b . With these definitions, we see that the time domain constraints on the channel can be phrased as: Choose weights $\{z_b, b \in \Lambda_{good}\}$ so that

$$F_b \mathbf{g}_i \approx \hat{\mathbf{G}}_{i,b}^\# z_b, \quad b \in \Lambda_{good}, i = 1, 2, \dots, S \quad (4.34)$$

and \mathbf{g}_i is a real-valued vector with P dimensions. Since there are no constraints on the signal, (4.34) captures all the available constraints, and any set of weights $\{z_b, b \in \Lambda_{good}\}$ that satisfying these constraints will lead to a valid reconstruction.

To obtain $\{z_b, b \in \Lambda_{good}\}$ satisfying (4.34), we minimize the least-squares cost function

$$J(\{\mathbf{g}_i\}, \{z_b\}) = \sum_{i=1}^S \sum_{b \in \Lambda_{good}} \|F_b \mathbf{g}_i - \hat{\mathbf{G}}_{i,b}^\# z_b\|^2 \quad (4.35)$$

where \mathbf{g}_i is a P -dimensional real-valued vector. Let $\Re(\mathbb{A})$ and $\Im(\mathbb{A})$ denote the element-by-element real and imaginary parts of the matrix \mathbb{A} respectively. Expressing $F_b, \hat{\mathbf{G}}_{i,b}^\#$ and z_b in terms of their real and imaginary parts, we can rewrite the cost function $J(\{\mathbf{g}_i\}, \{z_b\})$ in terms of the purely real variables \mathbf{g}_i and $\tilde{\mathbf{z}}_b$ as,

$$\tilde{J}(\{\mathbf{g}_i\}, \{\tilde{\mathbf{z}}_b\}) = \sum_{i=1}^S \sum_{b \in \Lambda_{good}} \|\tilde{F}_b \mathbf{g}_i - \hat{\mathbf{G}}_{i,b} \tilde{\mathbf{z}}_b\|^2 \quad (4.36)$$

where $\tilde{F}_b = \begin{bmatrix} \Re(F_b) \\ \Im(F_b) \end{bmatrix}$, $\hat{\mathbf{G}}_{i,b} = \begin{bmatrix} \Re(\hat{\mathbf{G}}_{i,b}^\#) & -\Im(\hat{\mathbf{G}}_{i,b}^\#) \\ \Im(\hat{\mathbf{G}}_{i,b}^\#) & \Re(\hat{\mathbf{G}}_{i,b}^\#) \end{bmatrix}$ and $\tilde{\mathbf{z}}_b = \begin{bmatrix} \Re(z_b) \\ \Im(z_b) \end{bmatrix}$.

We minimize the cost function \tilde{J} by a three-step process:

1. For any given set of values $\{\tilde{\mathbf{z}}_b, b \in \Lambda_{good}\}$, we compute the channel estimates in the time domain \mathbf{g}_i that minimize $\tilde{J}(\{\mathbf{g}_i\}, \{\tilde{\mathbf{z}}_b\})$.
2. We substitute these channel estimates into \tilde{J} to obtain a “reduced” cost function \tilde{J}_{red} that depends only on $\tilde{\mathbf{z}}_b$.
3. We optimize the reduced cost function \tilde{J}_{red} over $\tilde{\mathbf{z}}_b$ to obtain the global optimum of $\tilde{J}(\{\mathbf{g}_i\}, \{\tilde{\mathbf{z}}_b\})$.

Before explaining these steps in detail, we massage the objective function in (4.36) into a form where the solution becomes apparent. Let the “good” bands, which are elements of Λ_{good} , be denoted by b_1, b_2, \dots, b_M . By grouping terms in (4.36)

across bands, we can show that,

$$\tilde{J}(\{\mathbf{g}_i\}, \{\tilde{\mathbf{z}}\}) = \sum_{i=1}^S \|\tilde{F}\mathbf{g}_i - \hat{\mathbb{G}}_i\tilde{\mathbf{z}}\|^2 \quad (4.37)$$

where $\tilde{F} = [\tilde{F}_{b_1}^T \tilde{F}_{b_2}^T \dots \tilde{F}_{b_M}^T]^T$, $\tilde{\mathbf{z}} = [\tilde{\mathbf{z}}_{b_1}^T \tilde{\mathbf{z}}_{b_2}^T \dots \tilde{\mathbf{z}}_{b_M}^T]^T$ and $\hat{\mathbb{G}}_i = \begin{pmatrix} \hat{\mathbb{G}}_{i,b_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \hat{\mathbb{G}}_{i,b_M} \end{pmatrix}$

Step 1 - Find \mathbf{g}_i given $\tilde{\mathbf{z}}$: Suppose that we are given a set of hypothetical weights $\tilde{\mathbf{z}}$ and our goal is to estimate the channels $\{\mathbf{g}_i\}$ that minimize \tilde{J} . From (4.37), we see that the best estimate of the channel to sensor i can be obtained as:

$$\hat{\mathbf{g}}_i = \arg \min_{\mathbf{g}_i} \|\tilde{F}\mathbf{g}_i - \hat{\mathbb{G}}_i\tilde{\mathbf{z}}\|^2 \quad (4.38)$$

This is a standard least-squares problem with solution

$$\hat{\mathbf{g}}_i = \tilde{F}^\dagger \hat{\mathbb{G}}_i \tilde{\mathbf{z}} \quad (4.39)$$

where \tilde{F}^\dagger denotes the pseudoinverse of \tilde{F} .

Step 2 - The reduced cost function $\tilde{J}_{red}(\tilde{\mathbf{z}})$: Given a set of hypothetical weights $\tilde{\mathbf{z}}$, the lowest achievable cost over all possible channels is obtained by substituting the estimates from (4.39) into (4.37). Thus, we obtain the reduced

cost function $\tilde{J}_{red}(\tilde{\mathbf{z}})$,

$$\begin{aligned}\tilde{J}_{red}(\tilde{\mathbf{z}}) &= \sum_{i=1}^S \|\tilde{F}\tilde{F}^\dagger\hat{\mathbb{G}}_i\tilde{\mathbf{z}} - \hat{\mathbb{G}}_i\tilde{\mathbf{z}}\|^2 \\ &= \sum_{i=1}^S \|(\mathbb{I} - \tilde{F}\tilde{F}^\dagger)\hat{\mathbb{G}}_i\tilde{\mathbf{z}}\|^2\end{aligned}\tag{4.40}$$

The reduced cost function has an intuitive interpretation: given a set of arbitrary weights $\tilde{\mathbf{z}}$, the hypothesized channel at sensor i in the frequency domain is given by $\hat{\mathbb{G}}_i\tilde{\mathbf{z}}$. However, the postulated channel does not necessarily lie in the class of allowed channels: the space of Fourier transforms of real-valued channels restricted to a length of P samples. We denote this class of allowed channels by Ω . The matrix $(\mathbb{I} - \tilde{F}\tilde{F}^\dagger)$ projects the postulated channel $\hat{\mathbb{G}}_i\tilde{\mathbf{z}}$ onto the orthogonal complement of Ω . Therefore, \tilde{J}_{red} measures the energy in the portion of the hypothesized channels that cannot be explained by the class of allowed channels and tries to minimize it.

Step 3: Obtaining the weights $\tilde{\mathbf{z}}$: Using Pythagoras' theorem, we interpret the reduced cost function $\tilde{J}_{red}(\tilde{\mathbf{z}})$ as $\|\Gamma\tilde{\mathbf{z}}\|^2$ where the matrix Γ is defined

as $\Gamma \triangleq \begin{pmatrix} (\mathbb{I} - \tilde{F}\tilde{F}^\dagger)\hat{\mathbb{G}}_1 \\ \vdots \\ (\mathbb{I} - \tilde{F}\tilde{F}^\dagger)\hat{\mathbb{G}}_S \end{pmatrix}$. Therefore, the optimal weights can be obtained by

minimizing $\tilde{J}_{red}(\tilde{\mathbf{z}}) = \|\Gamma\tilde{\mathbf{z}}\|^2$. To eliminate the trivial solution $\tilde{\mathbf{z}} = \mathbf{0}$, we impose the condition $\|\tilde{\mathbf{z}}\| = 1$. With this constraint, the optimal weights $\tilde{\mathbf{z}}_{opt}$ are given

by choosing the right singular vector of Γ corresponding to its smallest singular value.

Estimating the signal and channels given $\tilde{\mathbf{z}}$: We substitute the optimal weights obtained from Step 3 into (4.39) to obtain time domain estimates of the channels $\hat{\mathbf{g}}_i$. We then estimate the signal in the frequency domain by Maximal Ratio Combining. To do this, we first form a frequency domain estimate of the channels $\hat{\mathbf{G}}_i = F\hat{\mathbf{g}}_i$. Let the signal estimate in the frequency domain be denoted by $\hat{\mathbf{X}}$. We set $\hat{\mathbf{X}} = \mathbf{0}$ within the “bad” bands. Within the good bands, we estimate the signal via Maximal Ratio Combining as,

$$\hat{\mathbf{X}}[l] = \frac{\sum_{i=1}^S \hat{\mathbf{G}}_i^*[l] \mathbf{Y}_i[l]}{\sum_{i=1}^S |\hat{\mathbf{G}}_i[l]|^2} \quad (4.41)$$

where $\mathbf{Y}_i[l]$ is the recorded sample at sensor i and frequency index l .

Complexity: The number of rows in the matrices \tilde{F} and Γ grow in proportion to the number of “good” bands - equivalently, the bandwidth of the signal - and the recording window. Additionally, the number of columns in \tilde{F} grows with the channel length P . Therefore, computing the pseudoinverse of \tilde{F} and the SVD of Γ (to obtain $\tilde{\mathbf{z}}$) can be expensive for signals with large time-bandwidth products (say, bandwidth of 600 Hz, recorded for $T = 4$ s).

4.6.3 Simulation Results

We now present simulation results that demonstrate fundamental ambiguities in estimating signals with “large” bandwidths. We consider a “random” signal (defined in Section 4.5) that lies in the frequency band from 0 to 200 Hz lasting for $T = 4$ seconds. We simulate a setting with $S = 6$ sensors using a sampling frequency of $f_s = 16$ kHz. The channel to each of these sensors has a delay spread $\tau_{max} = 20$ ms and is generated in the same fashion as in Section 4.5. Since we wish to establish fundamental results on ambiguity, we simulate an ideal scenario with no noise added to the recorded signals. We choose a coherence bandwidth $B_{coh} = 5$ Hz for the processing in Stage 1 with no overlap between the bins (it is not needed since we are processing Stage 1 estimates “globally”). Therefore, only the first 40 bands in the received signals contain energy. We delay the signal/channel estimates at the output of Stage 1 in these 40 bands, as explained in (4.31) and (4.32), by choosing $\tau_b = (150 \text{ samples}) \times \sin(2\pi b/40)$, $b = 0, 1, \dots, 39$ and then reconstruct the signal and channels using the global stitching algorithm. To ensure that the reconstructed channels are “well-conditioned”, we fix the condition number (ratio of largest to smallest singular value) of \tilde{F} to 100 before computing its pseudoinverse. We run 50 trials with the channel realizations varying randomly across trials and focus our attention on two questions:

- **How well is the source reconstructed?:** We quantify the quality of the reconstruction by the correlation ρ_{source} between the source signal \mathbf{x} and the estimate $\hat{\mathbf{x}}$, defined as

$$\rho_{source} = \max_{\tau} \frac{\hat{\mathbf{x}}^T (\mathcal{D}^{\tau} \mathbf{x})}{\|\mathbf{x}\| \|\hat{\mathbf{x}}\|} \quad (4.42)$$

where \mathcal{D} the unit delay operator.

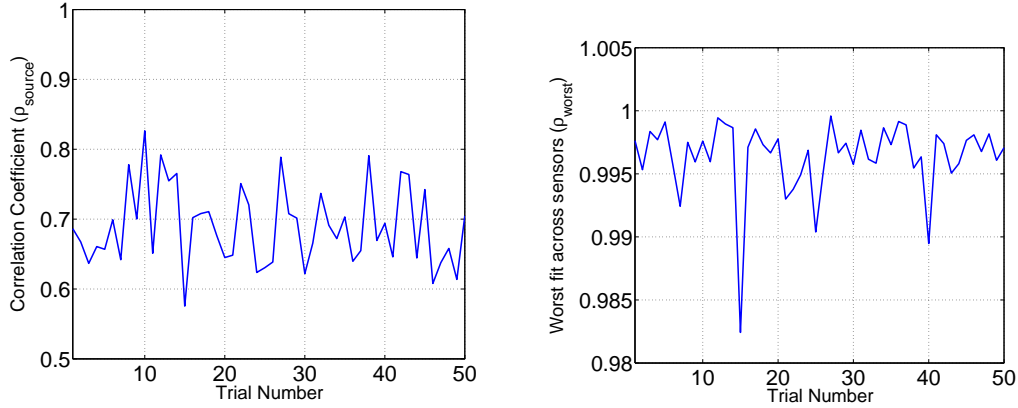
- **How well is the received data explained?:** We form an estimate of the received signal at sensor i , denoted by $\hat{\mathbf{y}}_i$, based on the *estimated* channel $\hat{\mathbf{g}}_i$ and the *estimated* signal $\hat{\mathbf{x}}$. We estimate its fit with the recorded data \mathbf{y}_i using the correlation coefficient $\rho_{fit,i}$, given by

$$\rho_{fit,i} = \frac{\hat{\mathbf{y}}_i^T \mathbf{y}_i}{\|\mathbf{y}_i\| \|\hat{\mathbf{y}}_i\|}$$

Finally, we quantify the performance of the algorithm in terms of explaining the received data by the *worst* fit across sensors, given by $\rho_{fit,worst} = \min_i \rho_{fit,i}$. The results are shown in Figures 4.5(a) and 4.5(b) and we make the following observations:

- **Quality of Fit:** From Figure 4.5(b), we see that the signal and channels estimated by global stitching algorithm explain the received data nearly perfectly: the worst fit across sensors $\rho_{fit,worst}$ is typically greater than 0.99; even the lowest value taken by the worst fit across iterations is 0.982.

- **Quality of Reconstruction:** From Figure 4.5(a), we see that the quality



(a) Quality of Source Estimate

(b) Quality of explanation of received data

Figure 4.5: Plots of ρ_{source} and $\rho_{worst,fit}$ for the simulated scenarios. We see that the quality of the source reconstruction ρ_{source} fairly low and fluctuates a lot, even though the data is consistently explained well ($\rho_{worst,fit} > 0.98$ always).

of source reconstruction, ρ_{source} fluctuates a lot with varying channel realizations and it can be as low as 0.58.

- **Multiple Explanations:** To take a particular example, in the 46th trial, the correlation between the estimated source and the true source is 0.61, indicating that the estimate is significantly different from the truth. However, the estimates fit the recorded data nearly perfectly : the correlations $\rho_{fit,i}$ are $\{0.9984, 0.9981, 0.9981, 0.9984, 0.9983, 0.9985\}$. This shows that a source estimate which is significantly different from the original source can explain the recorded data nearly perfectly, while respecting constraints on the channel delay spreads. Furthermore, we can generate as many additional explanations as we wish by varying the profile

of the injected delays τ_b , subject to the broad rules specified in Section 4.6.1.

- **Impact of injecting additional delays:** To understand the influence of the injected delays, we compare these results with the case where we do not delay the estimates from different bands in Stage 1 before stitching them together. When we do not inject any delay, the *average* value of ρ_{source} over 50 channel realizations is 0.90. On the other hand, when we introduce delays into the outputs of Stage 1, the average value of ρ_{source} drops to 0.69, demonstrating the adverse impact of the injected delays on signal reconstruction.

Chapter 5

Conclusions

The examples provided in this dissertation illustrate that it is possible to solve complex problems even under minimalist assumptions on observation and communication models. We now summarize these results, outline some directions for future work and conclude with a broader discussion on the promise held by minimalist approaches to other problems in sensor networks.

5.1 Implicit Timing Synchronization

We established the basic feasibility of implicit timing synchronization in TDM networks, leveraging the timing information present in the existing communication. We investigated phase-only and phase-frequency adjustment algorithms, both of which were guided by the insights derived from a synchronous averaged system. We concluded that phase-only adjustments suffice for small networks, but

frequency adjustments, occurring on a slower timescale, are necessary for larger networks. Simulations show that with such phase-frequency adjustments, we can achieve synchrony upto tens of microseconds in omnidirectional networks and tens of nanoseconds in directional networks.

Future Work: A significant amount of effort is required in translating our ideas to practice, starting with more detailed simulations that explicitly model traffic patterns, medium access control, and initial establishment of coarse timing synchronization (including estimation of propagation delays). The proposed algorithm can be implemented as follows: a timing synchronization application, layered on top of the network protocol stack, generates packets only at startup, but not for synchronization maintenance. During the maintenance phase, the implicit timestamps could be sent to this application from the physical layer (if extremely accurate timing synchronization is needed) or from the MAC layer (if lower synchronization accuracy suffices) to adjust the clock. Substantial effort is needed to translate these ideas into an implementation on a sensor mote or open source platforms such as the WARP radios.

5.2 Localizing multiple events from ToAs

We investigated the feasibility of localizing multiple events that occur in quick succession from their ToAs and proposed a robust, low-complexity algorithm to do this. For the case of two events, we showed that if we have “enough” sensors (nine of them, not lying on a branch of a hyperbola), we can guarantee perfect localization. On the other hand, if we have too “few” sensors (an example with six sensors), there could be ambiguities in localizing the events. We proposed a three-stage algorithm that relies on cutting down the set of candidate solutions by discretizing the times at which events occur and then using a linear program to solve the ToA-to-event association problem. Simulations show excellent performance in estimating the number of events as well as their locations and times.

Future Work: A detailed investigation is needed to extend the fundamental feasibility results, assuming ideal sensing conditions, to scenarios where more than two events occur. We also need to understand the limits of localizing multiple events when the measurements are noisy. Finally, we believe that general approach of parallelizing the evidence is also applicable to the problem of localizing multiple events with other types of sensors such as Angle of Arrival (AoA) and binary proximity sensors [37].

5.3 Collaborative Estimation In Dispersive Environments

We proposed an algorithm to reconstruct an unknown signal recorded at multiple sensors through an unknown dispersive channel. We solved the problem with manageable complexity by switching to the frequency domain. This allowed us to solve a number of easier subproblems and then combine their results appropriately. The proposed technique works well when the bandwidth of the signal is within 10-20 times the coherence bandwidth of the channel, but we demonstrated fundamental ambiguities over larger bands. Such ambiguities arise by the introduction of small delay differences across bands, which result in multiple combinations of signal and channels that can explain a given set of observations.

Future Work: It remains an open issue as to whether and how additional knowledge about the signal and/or channels can be best leveraged to alleviate these ambiguities. For example, when the signal has sufficient frequency diversity (precisely defined in [51]), multiple explanations for the recorded data can exist only if the channels seen by the different sensors have common zeros. The direction for future investigations depends on whether or not typical acoustic multipath channels have common zeros. If they do, we cannot hope to reconstruct the signal perfectly and it would be of interest to predict the degradation in the estimate due

to the common zeros. If they do not, we need to modify the proposed frequency domain algorithm to incorporate this fact, while continuing to reconstruct the source in a robust manner with low-complexity.

5.4 Minimalism all the way

These problems constitute only a small subset of the scenarios where a minimalist design is attractive in sensor networks. Indeed, we need to solve a chain of problems before we can deploy a sensor network and we now give examples to argue that adhering to minimalism is useful in designing every link of this chain.

Typically, we are confronted with the following sequence of issues while designing a sensor network: choose sensors to observe the environment \rightarrow design algorithms to detect interesting activity at each sensor \rightarrow reduce high dimensional observations to low-dimensional “feature vectors” that capture the essence of the activity \rightarrow design protocols to forward these feature vectors to a fusion center \rightarrow use the relationship between the phenomenon of interest and the recorded observations to make inferences. As we will see in the following examples, designing each of these stages minimally leads to a variety of benefits: it reduces the computational load, allows the system to be adaptive, increases the network efficiency and makes the system robust to variations in the surroundings.

Neuromorphic cameras [22] are a good example of reducing the sensing process to a bare minimum. In contrast to conventional cameras that produce a steady stream of data, sensors in a neuromorphic camera produce an output only when the light that is incident on them changes “sufficiently”. Since scenes are relatively static, the volume of data produced by a neuromorphic camera is considerably smaller, thereby reducing the computational load. However, image processing algorithms will have to be designed afresh, accounting for the nature of the data produced by the new hardware.

To design sensor networks that “learn” in an automated manner, we need a minimalist approach to detect interesting patterns of activity without detailed models for the signal. In such networks, each sensor could estimate the typical background activity over time, identify deviations from the background and use the observations at other sensors to cross-check these deviations. If enough sensors agree, a deviation could be declared interesting, thereby allowing the sensor network to detect new sources and adapt to uncertain environments.

In large networks, we require minimalist networking protocols to aggregate data from many sensors at a fusion center with little overhead. Consider the following thought experiment: we deploy temperature sensors over a large region to monitor climate change. In the normal mode of operation, a subset of sensors forward their data (perhaps subsampled in time) to a fusion center via a multihop

network. When the fusion center observes that the readings in some regions are “out of the ordinary”, it requests the sensors in these areas to forward their data with the highest possible fidelity. We now need to solve two problems:

- Since many sensors that are close to each other need to transmit simultaneously, they have to cooperate in accessing the shared wireless medium so that their transmissions do not collide. Thus, we need decentralized algorithms which ensure that the overhead involved in such cooperation is negligible, when compared to the volume of data being transported.
- Since the sensors of interest are roughly in the same “direction” from the fusion center, using a greedy routing protocol to direct the data along the shortest path can lead to congestion. We need a simple algorithm that routes the data with little overhead and avoids hotspots, while provisioning for multiple regions of interesting activity.

Choosing a minimalist model while making inferences provides insurance against unpredictable variations in the sensing process. The concept of binary proximity sensors provides the perfect illustration for this idea. A binary proximity sensor – it could be acoustic, Passive Infrared or seismic, to give three examples – simply outputs a 1 if something interesting has occurred in its vicinity and a 0 otherwise. Even an inexpensive sensor in a poor sensing environment is extremely likely to

provide this level of information. This guarantees – on paper and in practice – a base level of performance. Next, since the sensors can be inexpensive, we can deploy many of them to improve the performance to a desired level. To give another example, consider an algorithm that tracks objects from the frames of a video using “fine” features such as color and texture. Such an algorithm will work very well in a favorable environment, but is susceptible to changes in the ambient lighting (for example). On the other hand, an algorithm that works with coarser features – like the processing in the retina that only uses local differences in brightness to ultimately build complex scenes – is better equipped to handle unforeseen variations in the ambience.

Appendices

Appendix A

In this appendix, we provide detailed derivations for some results in Chapter 2.

A.1 An expression for the excess phases $\varphi_{ex}[s]$ in the averaged system

We begin by obtaining an approximate expression for the phases in slot s (where s is large) by substituting equation (2.8) into equation (2.6),

$$\varphi[s] \approx (\bar{\varphi}[0] + s\bar{\psi})\mathbf{1} + (\mathbb{I} + \sum_{k=1}^{s-1} \bar{\mathcal{G}}^k)\boldsymbol{\delta} \quad (\text{A.1})$$

We now simplify this expression by showing that $\bar{\mathcal{G}}^k \boldsymbol{\delta} = \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta}$. Since $\mathbf{1}^T \boldsymbol{\delta} = 0$, we have

$$\bar{\mathcal{G}}^k \boldsymbol{\delta} = \bar{\mathcal{G}}^k \boldsymbol{\delta} - \frac{\mathbf{1}\mathbf{1}^T}{N} \boldsymbol{\delta} = \left(\bar{\mathcal{G}}^k - \frac{\mathbf{1}\mathbf{1}^T}{N} \right) \boldsymbol{\delta} \quad (\text{A.2})$$

Using the spectral decomposition of $\bar{\mathcal{G}}_{ex}$, we see that, $\bar{\mathcal{G}}_{ex}^k = \sum_{l=2}^N \lambda_l^k \mathbf{v}_l \mathbf{v}_l^T = \bar{\mathcal{G}}^k - \mathbf{1}\mathbf{1}^T/N$. Substituting this into equation (A.2), we get $\bar{\mathcal{G}} \boldsymbol{\delta} = \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta}$. We now substitute this result back into equation (A.1) to obtain,

$$\boldsymbol{\varphi}[s] \approx (\bar{\varphi}[0] + s\bar{\psi})\mathbf{1} + \sum_{k=0}^{s-1} \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta} \quad (\text{A.3})$$

Next, we compute the mean phase across nodes in slot s . We have,

$$\bar{\varphi}[s] = \frac{\mathbf{1}^T \boldsymbol{\varphi}[s]}{N} = (\bar{\varphi}[0] + s\bar{\psi}) \frac{\mathbf{1}^T \mathbf{1}}{N} + \sum_{k=0}^{s-1} \mathbf{1}^T \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta} \quad (\text{A.4})$$

The term $\mathbf{1}^T \mathbf{1}/N$ is simply 1. We now show that $\mathbf{1}^T \bar{\mathcal{G}}_{ex}^k = \mathbf{0}^T$. Since $\bar{\mathcal{G}}_{ex}^k = \bar{\mathcal{G}}^k - \mathbf{1}\mathbf{1}^T/N$, we get

$$\mathbf{1}^T \bar{\mathcal{G}}_{ex}^k = \mathbf{1}^T \bar{\mathcal{G}}^k - \frac{\mathbf{1}^T \mathbf{1}}{N} \mathbf{1}^T = \mathbf{1}^T \bar{\mathcal{G}}^k - \mathbf{1}^T \quad (\text{A.5})$$

Since $\bar{\mathcal{G}}$ is symmetric and stochastic, all its powers are also symmetric and stochastic. Thus, we have $\mathbf{1}^T \bar{\mathcal{G}}^k = ((\bar{\mathcal{G}}^k)^T \mathbf{1})^T = (\bar{\mathcal{G}}^k \mathbf{1}^T)^T = \mathbf{1}^T$ where the second equality follows from the symmetry and the third from the stochasticity of $\bar{\mathcal{G}}$. Substituting back into equation (A.5), we get $\mathbf{1}^T \bar{\mathcal{G}}_{ex}^k = \mathbf{0}^T$. Therefore, the expression for the mean phases in equation (A.4) reduces to $\bar{\varphi}[s] = \bar{\varphi}[0] + s\bar{\psi}$. Since $\boldsymbol{\varphi}_{ex}[s] = \boldsymbol{\varphi}[s] - \bar{\varphi}[s]\mathbf{1}$, from equation (A.3), we get

$$\boldsymbol{\varphi}_{ex}[s] = \left(\sum_{k=0}^{s-1} \bar{\mathcal{G}}_{ex}^k \right) \boldsymbol{\delta}$$

Since the eigenvalues of $\bar{\mathcal{G}}_{ex}$, namely $\lambda_2, \lambda_3, \dots, \lambda_N$, are less than 1 in magnitude, the Neumann sum $\sum_{k=0}^{\infty} \bar{\mathcal{G}}_{ex}^k$ converges to $(\mathbb{I} - \bar{\mathcal{G}}_{ex})^{-1}$ [26]. Therefore, we have the desired result, $\boldsymbol{\varphi}_{ex}[s] \rightarrow (\mathbb{I} - \bar{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta}$ as $s \rightarrow \infty$.

A.2 Linear Programming Formulation

We begin by defining the directed phase error between nodes i and j to be $e_{ij} = \phi_i[s] - \phi_j[s] = \mathbf{c}_{ij}^T \boldsymbol{\varphi}[s]$ where \mathbf{c}_{ij} is defined as: $\mathbf{c}_{ij}[i] = 1, \mathbf{c}_{ij}[j] = -1, \mathbf{c}_{ij}[l] = 0 \forall l \neq i, j$. Since the mean component of $\boldsymbol{\varphi}[s]$ does not contribute to the error, we have, $\mathbf{c}_{ij}^T \boldsymbol{\varphi}[s] = \mathbf{c}_{ij}^T \boldsymbol{\varphi}_{ex}[s] = \mathbf{c}_{ij}^T (\mathbb{I} - \bar{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta}$. We note that the phase error between nodes i and j is a linear function of the skews $\boldsymbol{\delta}$. To obtain an estimate of the overhead required to maintain a TDM schedule, we maximize this quantity over all allowed excess frequencies $\boldsymbol{\delta}$. The first constraint that the excess frequencies must satisfy is $\mathbf{1}^T \boldsymbol{\delta} = 0$. Next, we note that the excess frequencies must be bounded since the node frequencies $F_l = f_l / f_{nom}$ are bounded by $1 \pm \rho_{max}$. We see this as follows: denoting the average frequency across nodes by $\bar{\psi}$, we have, $F_i = \bar{\psi} + \delta_i$. Therefore, $\bar{\psi}$ and δ_i must satisfy $(1 - \rho_{max}) \leq \bar{\psi} + \delta_i \leq (1 + \rho_{max}) \forall i$. We denote these constraints compactly as $(1 - \rho_{max}) \mathbf{1} \leq \bar{\psi} \mathbf{1} + \boldsymbol{\delta} \leq (1 + \rho_{max}) \mathbf{1}$. Finally, since $\bar{\psi}$ is the mean frequency, it is also bounded by $1 \pm \rho_{max}$. Therefore, the problem

of maximizing the directed error between nodes i and j can be stated as,

$$\begin{aligned} \text{Maximize}_{\boldsymbol{\delta}, \bar{\psi}} \quad J_{ij} &= \mathbf{c}_{ij}^T (\mathbb{I} - \bar{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta} \\ \text{subject to} \quad \boldsymbol{\delta}, \bar{\psi} &\in \Lambda \end{aligned}$$

where Λ is the set of allowed excess and mean frequencies given by $\Lambda = \{\bar{\psi}, \boldsymbol{\delta} : \mathbf{1}^T \boldsymbol{\delta} = 0, (1 - \rho_{max}) \mathbf{1} \leq \bar{\psi} \mathbf{1} + \boldsymbol{\delta} \leq (1 + \rho_{max}) \mathbf{1}, 1 - \rho_{max} \leq \bar{\psi} \leq 1 + \rho_{max}\}$. We call this formulation problem \mathcal{P}_{ij} . This problem is a linear program since the constraints and the objective function are linear in the decision variables $\bar{\psi}$ and $\boldsymbol{\delta}$. To find the worst error across all neighboring nodes, we solve such a linear program for each pair of neighbors i and j and pick the largest among resulting errors.

We consider the special case of $\bar{\psi} = 1$ to provide some insight into solutions of these linear programs. In this case, the problem of maximizing the directed phase error between nodes i and j can be simplified as,

$$\begin{aligned} \text{Maximize}_{\boldsymbol{\delta}} \quad J_{ij} &= \mathbf{c}_{ij}^T (\mathbb{I} - \bar{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta} \\ \text{subject to} \quad \boldsymbol{\delta} &\in \tilde{\Lambda} \end{aligned}$$

where $\tilde{\Lambda} = \{\boldsymbol{\delta} : \|\boldsymbol{\delta}\|_{\infty} \leq \rho_{max}, \mathbf{1}^T \boldsymbol{\delta} = 0\}$. We call this reduced problem $\tilde{\mathcal{P}}_{ij}$. It is known [10] that one of the optimizers of any linear program occurs at an “extreme point” of the feasible set (in this case, $\tilde{\Lambda}$). Therefore, we characterize the extreme

points of $\tilde{\Lambda}$ to understand the structure of the solution to problem $\tilde{\mathcal{P}}_{ij}$. We begin with the definition of an extreme point: $\boldsymbol{\delta}_{ex}$ is an extreme point of $\tilde{\Lambda}$, if it cannot be expressed as $(\boldsymbol{\delta}_1 + \boldsymbol{\delta}_2)/2$ for any two points $\boldsymbol{\delta}_1, \boldsymbol{\delta}_2 \in \tilde{\Lambda}$ such that $\boldsymbol{\delta}_1 \neq \boldsymbol{\delta}_2$. We use this definition to make two observations that characterize the extreme points of $\tilde{\Lambda}$.

Observation 1: Let $\boldsymbol{\delta}_{ex} = (\delta_{ex,1}, \delta_{ex,2}, \dots, \delta_{ex,N})$ be an extreme point of $\tilde{\Lambda}$.

Then, at most one of its components can take the value 0.

Proof: We prove the observation by contradiction. First, we note that $\|\boldsymbol{\delta}_{ex}\|_\infty \leq \rho_{max}$ and $\mathbf{1}^T \boldsymbol{\delta}_{ex} = 0$ since $\boldsymbol{\delta}_{ex} \in \tilde{\Lambda}$. Assume that there are two positions m and m' such that $\delta_{ex,m} = \delta_{ex,m'} = 0$. We construct two vectors $\boldsymbol{\delta}_+ = \boldsymbol{\delta}_{ex} + \boldsymbol{\xi}$, $\boldsymbol{\delta}_- = \boldsymbol{\delta}_{ex} - \boldsymbol{\xi}$ where $\boldsymbol{\xi} = (\xi_1, \xi_2, \dots, \xi_N)$ is defined as follows: $\xi_m = \rho_{max}/2$, $\xi_{m'} = -\rho_{max}/2$, $\xi_l = 0 \forall l \neq m, m'$. We see that $\|\boldsymbol{\delta}_+\|_\infty \leq \max(\|\boldsymbol{\delta}_{ex}\|_\infty, \rho_{max}/2) \leq \rho_{max}$ since, $\|\boldsymbol{\delta}_{ex}\|_\infty \leq \rho_{max}$. Also, $\mathbf{1}^T \boldsymbol{\delta}_+ = \mathbf{1}^T \boldsymbol{\delta}_{ex} + \mathbf{1}^T \boldsymbol{\xi} = 0 + 0 = 0$. Therefore, $\boldsymbol{\delta}_+$ is also contained in the feasible set $\tilde{\Lambda}$. Similarly, we can show that $\boldsymbol{\delta}_- \in \tilde{\Lambda}$. We now arrive at a contradiction : we have just expressed the extreme point $\boldsymbol{\delta}_{ex}$ as the average of $\boldsymbol{\delta}_+$ and $\boldsymbol{\delta}_-$, i.e. $\boldsymbol{\delta}_{ex} = (\boldsymbol{\delta}_+ + \boldsymbol{\delta}_-)/2$. Therefore, we conclude that our original assumption is wrong and that at most one component of $\boldsymbol{\delta}_{ex}$ can take the value 0.

Observation 2: Let $\boldsymbol{\delta}_{ex} = (\delta_{ex,1}, \delta_{ex,2}, \dots, \delta_{ex,N})$ be an extreme point of $\tilde{\Lambda}$.

Then, each component $\delta_{ex,i}$ takes one of the three values $\pm\rho_{max}, 0$.

Appendix A.

Proof: We prove the observation by contradiction. First, we write $\delta_{ex,i} = \alpha_{ex,i}\rho_{max}$ with $|\alpha_{ex,i}| \leq 1$ since $\|\boldsymbol{\delta}_{ex}\|_\infty \leq \rho_{max}$. Assume that for some index m , $\delta_{ex,m}$ takes a value other than $\pm\rho_{max}, 0$. Therefore, we can write $\delta_{ex,m} = \nu\rho_{max}$ with $-1 < \nu < 1$ and $\nu \neq 0$. Since $\sum_{i=1}^N \delta_{ex,i} = 0$, we get $\nu = -\sum_{i \neq m} \alpha_{ex,i}$. Since ν is not an integer, we see that there must exist another index $m' \neq m$ so that $-1 < \alpha_{ex,m'} < 1$ and $\alpha_{ex,m'} \neq 0$. We now construct two vectors $\boldsymbol{\delta}_+ = \boldsymbol{\delta}_{ex} + \boldsymbol{\xi}$, $\boldsymbol{\delta}_- = \boldsymbol{\delta}_{ex} - \boldsymbol{\xi}$ where $\boldsymbol{\xi} = (\xi_1, \xi_2, \dots, \xi_N)$ is defined as follows: $\xi_m = \pi_0$, $\xi_{m'} = -\pi_0$, $\xi_l = 0 \forall l \neq m, m'$. We choose π_0 small enough so that $\|\boldsymbol{\delta}_+\|_\infty$ and $\|\boldsymbol{\delta}_-\|_\infty$ are both less than ρ_{max} . We can show, as before, that $\boldsymbol{\delta}_+$ and $\boldsymbol{\delta}_-$ sum to zero and thereby, conclude that $\boldsymbol{\delta}_+, \boldsymbol{\delta}_- \in \tilde{\Lambda}$. We now arrive at the same contradiction as before, and conclude that our original assumption was wrong. This shows that every entry of $\boldsymbol{\delta}_{ex}$ takes one of the three values $\pm\rho_{max}, 0$.

We now put these observations together to arrive at the desired result. Since the elements of $\boldsymbol{\delta}_{ex}$ add up to zero, the number of entries in $\boldsymbol{\delta}_{ex}$ that take the value ρ_{max} and $-\rho_{max}$ must be equal. Furthermore, at most one entry in $\boldsymbol{\delta}_{ex}$ can be zero. From these facts, it is easy to see that: (1) If the number of nodes N is even, then exactly $N/2$ entries of $\boldsymbol{\delta}_{ex}$ equal ρ_{max} and the other $N/2$ entries equal $-\rho_{max}$ and (2) If the number of nodes N is odd, then $(N-1)/2$ entries of $\boldsymbol{\delta}_{ex}$ equal ρ_{max} , another $(N-1)/2$ entries equal $-\rho_{max}$ and there is one entry that is equal to zero. Since one of the optimizers of any linear program occurs at an

extreme point, we conclude that the largest directed phase error between nodes i and j occurs with roughly half the nodes running at the maximum frequency and the other half running at the minimum frequency for *any* network topology. Furthermore, since the feasible sets for all the problems $\tilde{\mathcal{P}}_{ij}$ are the same, this conclusion applies to the directed phase error between any neighboring nodes i and j and therefore, we get the desired result.

A.3 Actual System - Phase Only Adjustments

We begin with the expression for the excess phases from equation (2.13),

$$\varphi_{ex}[s] \approx \left(\mathbb{I} + \sum_{k=1}^{s-1} G_{s-1} G_{s-2} \dots G_{s-k} \right) \boldsymbol{\delta} \quad (\text{A.6})$$

We denote the average of $\varphi_{ex}[s]$ over realizations of $\{G_t\}_{t=0}^{s-1}$, with a fixed set of skews $\boldsymbol{\delta}$, by $\bar{\varphi}_{ex}[s] = \mathbb{E}(\varphi_{ex}[s])$. Using the linearity of expectation, we get,

$$\bar{\varphi}_{ex}[s] = \left(\mathbb{I} + \sum_{k=1}^{s-1} \mathbb{E}(G_{s-1} G_{s-2} \dots G_{s-k}) \right) \boldsymbol{\delta} \quad (\text{A.7})$$

Since $G_{s-1}, G_{s-2}, \dots, G_{s-k}$ are picked independent of one another, we have

$$\mathbb{E}(G_{s-1} G_{s-2} \dots G_{s-k}) = \mathbb{E}(G_{s-1}) \mathbb{E}(G_{s-2}) \dots \mathbb{E}(G_{s-k})$$

While this result is well known for scalar valued random variables, we can easily prove it for matrices as well. Since G_t is picked from $\{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$ with

probabilities $\{p_1, p_2, \dots, p_M\}$, we have

$$\mathbb{E}(G_t) = \sum_{i=1}^M p_i \mathcal{G}_i = \bar{\mathcal{G}} \quad \forall t$$

where $\bar{\mathcal{G}}$ is the system matrix for the averaged system. Substituting in equation (A.7), we have,

$$\bar{\varphi}_{ex}[s] = \left(\sum_{k=0}^{s-1} \bar{\mathcal{G}}^k \right) \delta$$

We can now excise the first eigenmode of $\bar{\mathcal{G}}$, as we did in Section A.1, and obtain

$$\bar{\varphi}_{ex}[s] = \left(\sum_{k=0}^{s-1} \bar{\mathcal{G}}_{ex}^k \right) \delta$$

The phase errors between neighbors are simply the pairwise differences between the corresponding entries of $\varphi_{ex}[s]$. We can obtain all such pairwise differences by operating on the excess phases $\varphi_{ex}[s]$ with a matrix \mathbf{C} . We denote the maximum error between any pair of nodes by $\|\mathbf{C}\varphi_{ex}[s]\|_\infty$. We can easily show that $\|\mathbf{v}\|_\infty$ is a convex function of its argument \mathbf{v} . Using this fact, we apply Jensen's inequality to $\|\mathbf{C}\varphi_{ex}[s]\|_\infty$ and obtain,

$$\begin{aligned} \mathbb{E}(\|\mathbf{C}\varphi_{ex}[s]\|_\infty) &\geq \|\mathbb{E}(\mathbf{C}\varphi_{ex}[s])\|_\infty \\ &= \|\mathbf{C}\mathbb{E}(\varphi_{ex}[s])\|_\infty = \|\mathbf{C}\bar{\varphi}_{ex}[s]\|_\infty \end{aligned} \quad (\text{A.8})$$

For a given set of skews δ , we recognize that $\|\mathbf{C}\bar{\varphi}_{ex}[s]\|_\infty$ is the maximum pairwise phase difference for the averaged system and $\mathbb{E}(\|\mathbf{C}\varphi_{ex}[s]\|_\infty)$ is the average of the

maximum pairwise phase difference in the original system. Thus, we conclude that the largest error from the averaged system provides a lower bound to the worst error in the actual system, averaged over many realizations of communication patterns.

A.4 Estimating skews from raw phases

We show here that $(\mathbb{I} - \bar{\mathcal{G}}) \boldsymbol{\varphi}[s] = (\mathbb{I} - \bar{\mathcal{G}}_{ex}) \boldsymbol{\varphi}_{ex}[s]$. First, since $\mathbf{1}^T \boldsymbol{\varphi}_{ex} = 0$, we have,

$$\begin{aligned} (\mathbb{I} - \bar{\mathcal{G}}_{ex}) \boldsymbol{\varphi}_{ex}[s] &= \left(\mathbb{I} - \bar{\mathcal{G}}_{ex} - \frac{\mathbf{1}\mathbf{1}^T}{N} \right) \boldsymbol{\varphi}_{ex}[s] \\ &= (\mathbb{I} - \bar{\mathcal{G}}) \boldsymbol{\varphi}_{ex}[s] \end{aligned} \tag{A.9}$$

where the second equality follows from the fact that $\bar{\mathcal{G}} = \bar{\mathcal{G}}_{ex} + \mathbf{1}\mathbf{1}^T/N$. Next, since $\bar{\mathcal{G}}$ is stochastic, we have, $(\mathbb{I} - \bar{\mathcal{G}}) \mathbf{1} = \mathbf{1} - \mathbf{1} = \mathbf{0}$. Therefore, we can add $(\mathbb{I} - \bar{\mathcal{G}}) \bar{\varphi}[s] \mathbf{1}$ to the right hand side of equation A.9 and obtain,

$$\begin{aligned} (\mathbb{I} - \bar{\mathcal{G}}_{ex}) \boldsymbol{\varphi}_{ex}[s] &= (\mathbb{I} - \bar{\mathcal{G}}) (\boldsymbol{\varphi}_{ex}[s] + \bar{\varphi}[s] \mathbf{1}) \\ &= (\mathbb{I} - \bar{\mathcal{G}}) \boldsymbol{\varphi}[s] \end{aligned} \tag{A.10}$$

where the second equality follows from the the fact that $\boldsymbol{\varphi}[s] = \bar{\varphi}[s] \mathbf{1} + \boldsymbol{\varphi}_{ex}[s]$.

Thus, we have the desired result.

A.5 Expression for the skew estimate - Averaged system

We first obtain an expression for the entries of $\bar{\mathcal{G}}$. By definition, the $(i, j)^{th}$ element of $\bar{\mathcal{G}}$ is $\bar{\mathcal{G}}(i, j) = \sum_{m=1}^M p_m \mathcal{G}_m(i, j)$. We consider the case $i \neq j$ first. If \mathcal{N}_j transmits a packet to \mathcal{N}_i when the m^{th} matching matrix \mathcal{G}_m is in operation, we have $\mathcal{G}_m(i, j) = \beta$; otherwise, $\mathcal{G}_m(i, j) = 0$. Therefore, the only matrices \mathcal{G}_m that contribute to $\bar{\mathcal{G}}(i, j)$ are those such that $\mathcal{G}_m(i, j) = \beta$. Summing over the probabilities of picking such matrices, we can obtain the probability that \mathcal{N}_j transmits to \mathcal{N}_i . We denote this probability by $q_{j \rightarrow i}$. With these definitions, we see that $\bar{\mathcal{G}}(i, j)$ is exactly $q_{j \rightarrow i} \beta$. Since $\bar{\mathcal{G}}$ is stochastic, we have $\bar{\mathcal{G}}(i, i) = 1 - \beta \sum_{j \neq i} q_{j \rightarrow i}$. We denote the i^{th} component of φ_∞ by $\varphi_{\infty, i}$ and that of $\bar{\mathcal{L}}\varphi_\infty$ by $[\bar{\mathcal{L}}\varphi_\infty]_i$. We now expand the matrix vector product $\bar{\mathcal{L}} = (\mathbb{I} - \bar{\mathcal{G}})\varphi_\infty$ and equate its i^{th} component to δ_i . By this process, we get,

$$[\bar{\mathcal{L}}\varphi_\infty]_i = \beta \sum_{j=1}^N q_{j \rightarrow i} (\varphi_{\infty, i} - \varphi_{\infty, j}) = \delta_i \quad (\text{A.11})$$

A.6 Evolution of the excess phases across a round

Applying equation (2.5) to phase evolution within a round, from slot $(s-1, r)$ to (s, r) , we have

$$\boldsymbol{\varphi}[s, r] = \overline{\mathcal{G}}\boldsymbol{\varphi}[s-1, r] + \boldsymbol{\delta}[s, r] + \overline{\boldsymbol{\psi}}[s, r]\mathbf{1} \quad (\text{A.12})$$

Since the frequencies are unchanged from the beginning of round r , the mean and excess frequencies are also unchanged from their values at the start of round r i.e. $\overline{\boldsymbol{\psi}}[s, r] = \overline{\boldsymbol{\psi}}[0, r]$ and $\boldsymbol{\delta}[s, r] = \boldsymbol{\delta}[0, r]$. Using this observation, and stepping back in time repeatedly using equation (2.5), we obtain

$$\boldsymbol{\varphi}[s, r] = s\overline{\boldsymbol{\psi}}[0, r]\mathbf{1} + \overline{\mathcal{G}}^s\boldsymbol{\varphi}[0, r] + \left(\sum_{k=0}^{s-1}\overline{\mathcal{G}}^k\right)\boldsymbol{\delta}[0, r] \quad (\text{A.13})$$

We now compute the mean phase across nodes in slot s of round r , denoted by $\overline{\boldsymbol{\varphi}}[s, r]$. Since $\overline{\boldsymbol{\varphi}}[s, r] = \mathbf{1}^T\boldsymbol{\varphi}[s, r]/N$, we have,

$$\overline{\boldsymbol{\varphi}}[s, r] = s\overline{\boldsymbol{\psi}}[0, r]\frac{\mathbf{1}^T\mathbf{1}}{N} + \mathbf{1}^T\overline{\mathcal{G}}^s\boldsymbol{\varphi}[0, r] + \mathbf{1}^T\left(\sum_{k=0}^{s-1}\overline{\mathcal{G}}^k\right)\boldsymbol{\delta}[0, r] \quad (\text{A.14})$$

Consider the terms separately. The term $\mathbf{1}^T\mathbf{1}/N$ is simply 1. We see that $\mathbf{1}^T\overline{\mathcal{G}}^k = \mathbf{1}^T \forall k$: $\mathbf{1}^T\overline{\mathcal{G}}^k = ((\overline{\mathcal{G}}^k)^T\mathbf{1})^T = (\overline{\mathcal{G}}^k\mathbf{1})^T = \mathbf{1}^T$ where the second equality follows from the symmetry and the third from the stochasticity of $\overline{\mathcal{G}}$. Thus, the second term in equation (A.14) simplifies to $\mathbf{1}^T\boldsymbol{\varphi}[0, r] = \overline{\boldsymbol{\varphi}}[0, r]$. We also use the fact that $\mathbf{1}^T\overline{\mathcal{G}}^k = \mathbf{1}^T$ to show that the third term $\sum_{k=0}^{s-1}(\mathbf{1}^T\overline{\mathcal{G}}^k)\boldsymbol{\delta}$ is annihilated: $\sum_{k=0}^{s-1}(\mathbf{1}^T\overline{\mathcal{G}}^k)\boldsymbol{\delta} =$

$\sum_{k=0}^{s-1} \mathbf{1}^T \boldsymbol{\delta} = 0$ where the second equality follows from the fact that $\mathbf{1}^T \boldsymbol{\delta} = 0$.

Putting these terms together, the mean phase in slot of s of round r is given by $\bar{\varphi}[s, r] = s\bar{\psi}[0, r] + \bar{\varphi}[0, r]$. Since the excess phases $\varphi_{ex}[s, r]$ are given by $\varphi[s, r] - \bar{\varphi}[s, r]\mathbf{1}$, we have, from equation (A.13),

$$\varphi_{ex}[s, r] = \bar{\mathcal{G}}^s \varphi[0, r] - \bar{\varphi}[0, r]\mathbf{1} + \sum_{k=0}^{s-1} \bar{\mathcal{G}}^k \boldsymbol{\delta} \quad (\text{A.15})$$

Since $\bar{\mathcal{G}}^s$ is also stochastic, we have $\bar{\mathcal{G}}^s \mathbf{1} = \mathbf{1}$, giving us,

$$\varphi_{ex}[s, r] = \bar{\mathcal{G}}^s (\varphi[0, r] - \bar{\varphi}[0, r]\mathbf{1}) + \sum_{k=0}^{s-1} \bar{\mathcal{G}}^k \boldsymbol{\delta} \quad (\text{A.16})$$

$$= \bar{\mathcal{G}}^s \varphi_{ex}[0, r] + \sum_{k=0}^{s-1} \bar{\mathcal{G}}^k \boldsymbol{\delta} \quad (\text{A.17})$$

where the second equality follows from the definition of the excess phases. Since $\bar{\mathcal{G}}^k \boldsymbol{\delta} = \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta}$ (see Section A.1), we obtain the desired result:

$$\varphi_{ex}[s, r] = \bar{\mathcal{G}}^s \varphi_{ex}[0, r] + \sum_{k=0}^{s-1} \bar{\mathcal{G}}_{ex}^k \boldsymbol{\delta}$$

A.7 Recursive bounds on the excess phases

The phases in the last slot of round r and the first slot of round $r+1$ are related by the phase evolution equation: $\varphi[0, r+1] = \varphi[W_r - 1, r] + \bar{\psi}[0, r]\mathbf{1} + \boldsymbol{\delta}[0, r]$.

We can once again calculate and subtract out the mean from both sides of this equation to infer that $\varphi_{ex}[0, r+1] = \varphi_{ex}[W_r - 1, r] + \boldsymbol{\delta}[0, r]$. Taking infinity norms

on both sides and applying the triangle inequality, we get

$$\|\varphi_{ex}[0, r + 1]\|_\infty \leq \|\varphi_{ex}[W_{r-1}, r]\|_\infty + \|\delta[0, r]\|_\infty \quad (\text{A.18})$$

However, we know from equation (2.17) that,

$$\varphi_{ex}[W_r - 1, r] = \overline{\mathcal{G}}_{ex}^{W_r-1} \varphi_{ex}[0, r] + D_{W_r-1} \delta[0, r] \quad (\text{A.19})$$

Taking infinity norms on both sides of this equation and applying the triangle inequality, we get,

$$\begin{aligned} \|\varphi_{ex}[W_r - 1, r]\|_\infty &\leq \|\overline{\mathcal{G}}_{ex}^{W_r-1}\|_\infty \|\varphi_{ex}[0, r]\|_\infty \\ &\quad + \|D_{W_r-1}\|_\infty \|\delta[0, r]\|_\infty \end{aligned} \quad (\text{A.20})$$

Substituting this in equation (A.18), we get,

$$\begin{aligned} \|\varphi_{ex}[0, r + 1]\|_\infty &\leq \|\overline{\mathcal{G}}_{ex}^{W_r-1}\|_\infty \|\varphi_{ex}[0, r]\|_\infty \\ &\quad + (1 + \|D_{W_r-1}\|_\infty) \|\delta[0, r]\|_\infty \end{aligned} \quad (\text{A.21})$$

which proves the desired result.

A.8 LLN arguments for the actual system

Since the third term in equation (2.25) is sufficient to obtain the excess-averaged-phase we denote it by $\varphi_{av,suf}$. We set $\varphi_{av,suf} = H\delta[0]$ where

$$\begin{aligned} H &\triangleq \frac{1}{S_R} \sum_{s'=2}^{S_R-1} \left[\mathbb{I} + \sum_{p=1}^{s'-1} G_{s'-1} G_{s'-2} \dots G_{s'-p} \right] \\ &= \frac{S_R-2}{S_R} \mathbb{I} + \frac{1}{S_R} \sum_{s'=2}^{S_R-1} \sum_{p=1}^{s'-1} G_{s'-1} G_{s'-2} \dots G_{s'-p} \end{aligned} \quad (\text{A.22})$$

We now express the matrix H as the sum of matrices H_p , $0 \leq p \leq S_R-2$, with the matrix H_p collecting all the terms in H which are products of p system matrices.

To do this, we interchange the order of the sums in equation (A.22) and get,

$$H = \frac{S_R-2}{S_R} \mathbb{I} + \frac{1}{S_R} \sum_{p=1}^{S_R-2} \sum_{s'=p+1}^{S_R-1} G_{s'-1} G_{s'-2} \dots G_{s'-p}$$

We can set $H_0 = (S_R-2)/(S_R)\mathbb{I}$ and $H_p = \frac{1}{S_R} \sum_{s'=p+1}^{S_R-1} G_{s'-1} G_{s'-2} \dots G_{s'-p}$, $1 \leq p \leq S_R-2$ and see that each term in the sum for H_p is a product of p system matrices. First, we show that $H_p \approx \bar{\mathcal{G}}^p$ using the LLN, when p is small. In such cases, H_p is virtually independent of the precise schedule in operation during this round of slots.

An expression for H_p for small values of p : We begin with the simplest of these matrices H_0 . For large values of S_R , we can approximate $(S_R-2)/(S_R)$

by 1 and consequently $H_0 \approx \mathbb{I}$ when $S_R \gg 2$. Next, we consider the simplest “nontrivial” matrix H_1 . By definition, we have,

$$H_1 = \frac{1}{S_R} \sum_{s'=2}^{S_R-1} G_{s'-1} = \frac{G_1 + G_2 + \dots + G_{S_R-2}}{S_R} \quad (\text{A.23})$$

Since the matrices $\{G_t\}_{t=0}^\infty$ are chosen independently from the set $\mathcal{S}_m = \{\mathcal{G}_1, \mathcal{G}_2, \dots, \mathcal{G}_M\}$ with probabilities $\{p_1, p_2, \dots, p_M\}$ respectively, we can use the law of large numbers when S_R is large and obtain,

$$H_1 = \frac{S_R - 2}{S_R} \frac{G_1 + G_2 + \dots + G_{S_R-2}}{S_R - 2} \approx \frac{S_R - 2}{S_R} \bar{\mathcal{G}} \quad (\text{A.24})$$

where $\bar{\mathcal{G}}$, as before, denotes the average system matrix $\sum_{i=1}^M p_i \mathcal{G}_i$. When $S_R \gg 2$, we can approximate $(S_R - 2)/S_R$ by 1 and therefore, $H_1 \approx \bar{\mathcal{G}}$.

We now consider the matrix H_2 . By definition, $H_2 = \frac{1}{S_R} \sum_{s'=3}^{S_R-1} G_{s'-1} G_{s'-2}$.

Expanding the sum, we get,

$$H_2 = \frac{G_2 G_1 + G_3 G_2 + G_4 G_3 + \dots + G_{S_R-2} G_{S_R-3}}{S_R} \quad (\text{A.25})$$

We cannot apply the law of large numbers to equation (A.25) directly because the terms in the sum are not independent. For example, the first two terms have G_2 in common and therefore, cannot be independent. However, the correlation between the terms in the sum is fairly weak - only adjacent terms have any dependence at all. For example, the first term $G_2 G_1$ is completely independent of the third term $G_4 G_3$ and any future terms. Therefore, we can break the dependence among the

terms by splitting the sum into two parts: one part consisting of the odd terms and the other part containing the even terms. After such a split, we get,

$$H_2 = \frac{G_2G_1 + G_4G_3 + \dots G_{S_R-3}G_{S_R-4}}{S_R} + \frac{G_3G_2 + G_5G_4 + \dots G_{S_R-2}G_{S_R-3}}{S_R} \quad (\text{A.26})$$

where we have assumed, for concreteness, that S_R is odd. Since the constituent elements of the doublet matrices, such as G_2G_1, G_4G_3 are independent, each doublet has an average value of $\overline{\mathcal{G}}^2$. Now, consider either sum in equation (A.26): it consists of $(S_R - 3)/2$ terms, which are all independent of one another. Applying the law of large numbers separately to each sum, we see that they both approach $\frac{S_R-3}{2S_R}\overline{\mathcal{G}}^2$. Note that we need when $\frac{S_R-3}{2}$ to be “large enough” to apply the LLN to each interleaved sum. Therefore, $H_2 \approx 2 \times \frac{S_R-3}{2S_R}\overline{\mathcal{G}}^2 \approx \overline{\mathcal{G}}^2$ for large values of S_R . In a completely analogous fashion, we can split the expression for H_p into p “interleaved” sums and use the LLN on each sum to conclude that $H_p \approx \frac{S_R-(p+1)}{S_R}\overline{\mathcal{G}}^p \approx \overline{\mathcal{G}}^p$ as long as $\frac{S_R-(p+1)}{p}$ is large enough to apply the LLN. Next, we show that when p becomes large, H_p tends to a rank-one matrix of the form $\mathbf{1}\alpha_p^T$ for some vector α_p .

An expression for H_p for large values of p : Each term in the sum for $H_p = \frac{1}{S_R} \sum_{s'=p+1}^{S_R-1} G_{s'-1}G_{s'-2} \dots G_{s'-p}$ is a product of p stochastic matrices. Thus, if p is larger than the critical limit S_w , a generic term of the form $G_{s'-1}G_{s'-2} \dots G_{s'-p}$,

is equal to a rank-one matrix $\mathbf{1}\alpha_{s',p}^T$. Substituting into the expression for H_p , we get

$$H_p = \mathbf{1}\alpha_p^T \quad (\text{A.27})$$

Let $S_{\bar{\mathcal{G}}}$ be large enough for us to set $\lambda_2^{S_{\bar{\mathcal{G}}}} \approx 0$ (and hence, $\bar{\mathcal{G}}_{ex}^{S_{\bar{\mathcal{G}}}} \approx 0$). We choose the number of slots in a round $S_R \gg S^* = \max(S_{\bar{\mathcal{G}}}, S_w)$ so that we can apply the LLN and approximate H^p by $\bar{\mathcal{G}}^p$ for $p \leq S^*$. We summarize the discussion in the form of three conclusions:

1. Since S^* is much smaller than S_R , $H_p \approx \bar{\mathcal{G}}^p$, $p < S^*$
2. Since S^* is larger than the critical limit S_w , $H_p \approx \mathbf{1}\alpha_p^T \forall p \geq S^*$ for some vector α_p
3. Since S^* is larger than $S_{\bar{\mathcal{G}}}$, $\lambda_2^{S^*}$ is negligible

Adding these terms to form H and substituting into the expression for $\varphi_{av,suf}$, we get,

$$\varphi_{av,suf} = H\delta[0] \approx \kappa\mathbf{1} + \sum_{p=0}^{S^*-1} \bar{\mathcal{G}}^p \delta[0] \quad (\text{A.28})$$

where $\kappa = \sum_{p=S^*}^{S_R-2} (\alpha_p^T \delta[0])$. Since the first term only contributes to the mean of the averaged phases, it can be thrown out. Thus, we get the *excess-averaged-phases* $\varphi_{av,ex}$ to be

$$\varphi_{av,ex} \approx \sum_{p=0}^{S^*-1} \bar{\mathcal{G}}^p \delta[0]$$

Appendix A.

Since $\overline{\mathcal{G}}^p \boldsymbol{\delta}[0] = \overline{\mathcal{G}}_{ex}^p \boldsymbol{\delta}[0]$ (see Section A.1), we have,

$$\boldsymbol{\varphi}_{av,ex} \approx \sum_{p=0}^{S^*-1} \overline{\mathcal{G}}_{ex}^p \boldsymbol{\delta}[0] \quad (\text{A.29})$$

We can now write $\sum_{p=0}^{S^*-1} \overline{\mathcal{G}}_{ex}^p = \sum_{p=0}^{\infty} \overline{\mathcal{G}}_{ex}^p - \sum_{p=S^*}^{\infty} \overline{\mathcal{G}}_{ex}^p$. Since all eigenvalues of $\overline{\mathcal{G}}_{ex}$ are less than 1, the Neumann sum $\sum_{p=0}^{\infty} \overline{\mathcal{G}}_{ex}^p$ converges to $(\mathbb{I} - \overline{\mathcal{G}}_{ex})^{-1}$.

Approximating $\overline{\mathcal{G}}_{ex}^s$ by the dominant term in its spectral decomposition, $\lambda_2^s \mathbf{v}_2 \mathbf{v}_2^T$,

we get,

$$\sum_{p=S^*}^{\infty} \overline{\mathcal{G}}_{ex}^p = \frac{\lambda_2^{S^*}}{1 - \lambda_2} \mathbf{v}_2 \mathbf{v}_2^T \approx 0$$

where the second equality follows from the fact that $\lambda_2^{S^*}$ is negligible. Using these conclusions in equation (A.29), we get the desired result,

$$\boldsymbol{\varphi}_{av,ex} \approx (\mathbb{I} - \overline{\mathcal{G}}_{ex})^{-1} \boldsymbol{\delta}[0]$$

Appendix B

We show that omitting the constraints $\mu_{ie} = w_{ie}\delta_e$ from the integer program in (3.50) does not change the optimal solution.

Technically, the complete integer program formulation must be:

$$\begin{aligned}
 \max J &= \sum_s \sum_{i \in \Omega_s^\#} \sum_{e=1}^P c_{ie} \mu_{ie} + \sum_s \sum_{i \in \Omega_s} c_{iO} \mu_{iO} \\
 \sum_{i \in \Omega_s^\#} \mu_{ie} &= \delta_e \quad \forall s, e \\
 \sum_{e=1}^P \mu_{ie} + \mu_{iO} &= 1 \quad \forall i \in \Omega_1 \cup \Omega_2 \dots \cup \Omega_N \\
 \mu_{ie} &= w_{ie} \delta_e \quad \forall e, i \\
 \delta_e, \mu_{ie}, \mu_{iO}, w_{ie} &\in \{0, 1\} \quad \forall i, e
 \end{aligned} \tag{B.1}$$

This formulation cannot be converted to a linear program directly, since the constraints we just added involve products of variables. On the other hand, the formulation in (3.50) can be converted to linear program via relaxation, and therefore, is desirable. We now show that the formulations in (3.50) and (B.1) equivalent.

We begin by observing that the variables w_{ie} do not appear in the objective function. Therefore, any value that they take, while respecting the constraints, has the same cost. Thus, one approach to solve the complete formulation in (B.1) could be the following: first, we solve the “partial” formulation in (3.50) and then pick $w_{ie} \in \{0, 1\}$ to satisfy $\mu_{ie} = w_{ie}\delta_e$. However, there is a problem with this approach – the solution to the partial formulation could potentially produce μ_{ie}, δ_e such that the equation $\mu_{ie} = w_{ie}\delta_e$ has no solution, when $w_{ie} \in \{0, 1\}$. We now show that this cannot happen.

The equation $\mu_{ie} = w_{ie}\delta_e$ (all variables are binary valued) has a solution for w_{ie} if $\mu_{ie} \leq \delta_e$. The first set of constraints already ensures that this will be the case – since every term in the LHS of $\sum_{i \in \Omega_s^\#} \mu_{ie} = \delta_e$ is non-negative, each one of them is no larger than the RHS. Thus, any solution to the formulation in (3.50) will satisfy $\mu_{ie} \leq \delta_e$, ensuring that the equations $\mu_{ie} = w_{ie}\delta_e$ can be solved. Therefore, the formulations in (3.50) and (B.1) are equivalent.

Bibliography

- [1] A. Ali, S. Asgari, T. Collier, M. Allen, L. Girod, R. Hudson, K. Yao, C. Taylor, and D. Blumstein. An empirical study of collaborative acoustic source localization. *Journal of Signal Processing Systems*, 2009.
- [2] M. Allen, L. Girod, R. Newton, S. Madden, D. Blumstein, and D. Estrin. Voxnet: An interactive, rapidly-deployable acoustic monitoring platform. In *Proc. IPSN 2008*.
- [3] P. Barooah, J. Hespanha, and A. Swami. On the effect of asymmetric communication on distributed time synchronization. In *Proc. 46th IEEE CDC*, pages 5465–5471, 2007.
- [4] A. Beck, P. Stoica, and J. Li. Exact and approximate solutions of source localization problems. *Signal Processing, IEEE Transactions on*, 56(5):1770–1778, 2008.
- [5] A. Bishop, B. Fidan, K. Dogancay, B. Anderson, and P. Pathirana. Exploiting geometry for improved hybrid aoa/tdoa-based localization. *Signal Processing*, 88(7):1775–1791, 2008.
- [6] R. Blum, S. Kassam, and H. Poor. Distributed detection with multiple sensors i. advanced topics. *Proceedings of the IEEE*, Jan 1997.
- [7] Y. Chan and K. Ho. A simple and efficient estimator for hyperbolic location. *Signal Processing, IEEE Trans.*, 42(8), 1994.
- [8] J. Chen, R. Hudson, and K. Yao. Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field. *Signal Processing, IEEE Transactions on*, 50(8):1843–1854, 2002.
- [9] J. Chen, R. Hudson, and K. Yao. Maximum-likelihood source localization and unknown sensor location estimation for wideband signals in the near-field. *Signal Processing, IEEE Transactions on*, Aug 2002.

- [10] G. Dantzig and M. Thapa. *Linear Programming: Theory and extensions*. Springer Verlag, 2003.
- [11] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *ACM SIGOPS Operating Systems Review*, 36:147–163, 2002.
- [12] N. Freris and P. Kumar. Fundamental limits on synchronization of affine clocks in networks. In *Invited paper in the 46th IEEE CDC*, 2007.
- [13] S. Ganeriwal, R. Kumar, and M. Srivastava. Timing-sync protocol for sensor networks. In *Proceedings ACM SenSys 2003*, pages 138–149. ACM New York, NY, USA, 2003.
- [14] X. Gang and K. Shalinee. Discrete-Time Second-Order Distributed Consensus Time Synchronization Algorithm for Wireless Sensor Networks. *EURASIP Journal on Wireless Communications and Networking*, 2009, 2008.
- [15] M. Gavish and A. Weiss. Performance analysis of bearing-only target location algorithms. *Aerospace and Electronic Systems, IEEE Transactions on*, 28(3):817–828, 1992.
- [16] A. Giridhar and P. Kumar. Distributed time synchronization in wireless networks: Algorithms and analysis (I). In *Proc. IEEE CDC 2006*.
- [17] L. Girod, M. Lukac, V. Trifa, and D. Estrin. The design and implementation of a self-calibrating distributed acoustic sensing platform. In *Proc. Sensys 2006*.
- [18] L. Huang and T. Lai. On the scalability of IEEE 802.11 ad hoc networks. In *Proc. 3rd ACM MOBIHOC*, 2002.
- [19] D. Hubel and T. Wiesel. Brain mechanisms of vision. *Scientific American*, 241(3):150, 1979.
- [20] D. Klein, S. Venkateswaran, J.T.Isaacs, J. Burman, T. Pham, J. Hespanha, and U.Madhow. Localization with sparse acoustic sensor networks using uavs as information seeking data mules. In *Submitted to ACM Transactions on Sensor Networks*.
- [21] Y. Li and Z. Ding. Blind channel identification based on second order cyclostationary statistics. In *Proc. ICASSP 1993*.

- [22] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128 x 128 120db 30mw asynchronous vision sensor that responds to relative intensity change. In *Solid-State Circuits Conference, 2006. ISSCC 2006. Digest of Technical Papers. IEEE International*, pages 2060–2069. IEEE, 2006.
- [23] H. Liu, G. Xu, L. Tong, and T. Kailath. Recent developments in blind channel equalization: From cyclostationarity to subspaces. *Signal Processing*, 1996.
- [24] G. Mao, B. Fidan, and B. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10), 2007.
- [25] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi. The flooding time synchronization protocol. In *Proc. ACM SenSys 2004*, pages 39–49, 2004.
- [26] C. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, 2000.
- [27] R. Mirollo and S. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM J. on Appl. Math.*, pages 1645–1662, 1990.
- [28] R. Olfati-Saber and R. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. on Automatic Control*, 49(9):1520–1533, 2004.
- [29] S. Pradhan and K. Ramchandran. Distributed source coding using syndromes (DISCUS): design and construction. *Information Theory, IEEE Transactions on*, 2003.
- [30] W. Ren. Second-order consensus algorithm with extensions to switching topologies and reference models. In *American Control Conference, 2007. ACC'07*, pages 1431–1436, 2007.
- [31] W. Ren and E. Atkins. Distributed multi-vehicle coordinated control via local information exchange. *International J. of Robust and Nonlinear Control*, 17(10-11):1002–1033, 2007.
- [32] W. Ren and R. Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5):655, 2005.
- [33] S. Sanghavi. Equivalence of lp relaxation and max-product for weighted matching in general graphs. In *Information Theory Workshop, 2007. ITW'07. IEEE*, pages 242–247. IEEE.

- [34] L. Schenato and G. Gamba. A distributed consensus protocol for clock synchronization in wireless sensor network. In *Proc. IEEE CDC 2007*.
- [35] O. Shalvi and E. Weinstein. New criteria for blind deconvolution of nonminimum phase systems (channels). *Information Theory, IEEE Transactions on*, 2002.
- [36] G. Sharma, R. Mazumdar, and N. Shroff. On the complexity of scheduling in wireless networks. In *Proc. ACM MobiCom 2006*.
- [37] N. Shrivastava, R. Mudumbai, U. Madhow, and S. Suri. Target tracking with binary proximity sensors. *ACM Transactions on Sensor Networks (TOSN)*, 5(4):30, 2009.
- [38] O. Simeone, U. Spagnolini, Y. BarNess, and S. Strogatz. Distributed synchronization in wireless networks. *IEEE Signal Processing Magazine*, 25(5):81–97, 2008.
- [39] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. In *Proc. Sensys 2004*.
- [40] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 1–12. ACM, 2004.
- [41] S. Singh, R. Mudumbai, and U. Madhow. Distributed coordination with deaf neighbors: efficient medium access for 60 GHz mesh networks. *IEEE INFOCOM 2010*, 2010.
- [42] R. Solis, V. Borkar, and P. Kumar. A new distributed time synchronization protocol for multihop wireless networks. In *Proc. of the 45th IEEE CDC*, 2006.
- [43] P. Sommer and R. Wattenhofer. Gradient Clock Synchronization in Wireless Sensor Networks. In *Proc. ACM/IEEE IPSN*, 2009.
- [44] L. Tong and S. Perreau. Multichannel blind identification : from subspace to maximum likelihood methods. *Proceedings of the IEEE*, 1998.

- [45] L. Tong, G. Xu, B. Hassibi, and T. Kailath. Blind channel identification based on second-order statistics: A frequency-domain approach. *Information Theory, IEEE Transactions on*, 2002.
- [46] L. Tong, G. Xu, and T. Kailath. Blind identification and equalization based on second-order statistics: A time domain approach. *Information Theory, IEEE Transactions on*, 2002.
- [47] R. Viswanathan and P. Varshney. Distributed detection with multiple sensors i. fundamentals. *Proceedings of the IEEE*, 1997.
- [48] H. Wang, C. Chen, A. Ali, S. Asgari, R. Hudson, K. Yao, D. Estrin, and C. Taylor. Acoustic sensor networks for woodpecker localization. 2005.
- [49] X. Wang, Z. Wang, and B. O’Dea. A toa-based location algorithm reducing the errors due to non-line-of-sight (nlos) propagation. *Vehicular Technology, IEEE Transactions on*, 52(1):112–116, 2003.
- [50] G. Werner-Allen, G. Tewari, A. Patel, M. Welsh, and R. Nagpal. Firefly-inspired sensor network synchronicity with realistic radio effects. In *Proc. ACM SenSys 2005*, pages 142–153, 2005.
- [51] G. Xu, H. Liu, L. Tong, and T. Kailath. A least-squares approach to blind channel identification. *Signal Processing, IEEE Transactions on*, 1995.
- [52] X. Xu, N. Rao, and S. Sahni. A computational geometry method for localization using differences of distances. *ACM Transactions on Sensor Networks (TOSN)*, 6(2):10, 2010.
- [53] X. Xu, S. Sahni, and N. Rao. On basic properties of localization using distance-difference measurements. In *Information Fusion, 2008 11th International Conference on*, pages 1–8. IEEE, 2008.