Distributed Speech Enhancement in Wireless Acoustic Sensor Networks

Proefschrift

ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus prof. ir. K. C. A. M. Luyben,
voorzitter van het College voor Promoties,
in het openbaar te verdedigen op donderdag 18 juni 2015 om 15:00 uur

door

Yuan ZENG

Master of Science in Electrical Engineering
Northwestern Polytechnical University

geboren te Jiangxi, China.

Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. R. L. Lagendijk

Toegevoegd promotor:
Dr. ir. R. C. Hendriks

Samenstelling promotiecommissie:

Rector Magnificus	voorzitter
Prof. dr. ir. R. L. Lagendijk	Delft University of Technology, promotor
Dr. ir. R. C. Hendriks	Delft University of Technology, copromotor
Dr. S. Gannot	Bar Ilan University, Israel
Prof. dr. ir. G. Jongbloed	Delft University of Technology
Prof. dr. ir. B. de Vries	Eindhoven University of Technology
Prof. dr. A. J. van der Veen	Delft University of Technology
Prof. dr. K. G. Langendoen	Delft University of Technology

The work described in this thesis was financially supported by China Scholarship Council.

ISBN: 978-94-6186-423-9

Distributed Speech Enhancement in Wireless Acoustic Sensor Networks

Summary

In digital speech communication applications like hands-free mobile telephony, hearing aids and human-to-computer communication systems, the recorded speech signals are typically corrupted by background noise. As a result, their quality and intelligibility can get severely degraded. Traditional noise reduction approaches process signals recorded by microphone arrays using centralized beamforming technologies. Recent advances in micro-electro-mechanical systems and wireless communications enable the development of wireless sensor networks (WSNs), where low-cost, low-power and multi-functional wireless sensing devices are connected via wireless links. Compared with conventional localized and regularly arranged microphone arrays, wireless sensor nodes can be randomly placed in environments and thus cover a larger spatial field and yield more information on the observed signals. This thesis explores some problems on multi-microphone speech enhancement for wireless acoustic sensor networks (WASNs), such as distributed noise reduction processing, clock synchronization and privacy preservation.

First, we develop a distributed delay-and-sum beamformer (DDSB) for speech enhancement in WASNs. Due to limited power of each wireless device, signal processing algorithms with low computational complexity and low communication cost are preferred in WASNs. Distributed signal processing allows that each node only communicates with its neighboring nodes and performs local processing, where communication load and computational complexity are distributed over all nodes in the network. Without central processor and network topology constraint, the DDSB algorithm estimates the desired speech signal via local processing and local communication. The DDSB algorithm is based on an iterative scheme. More specifically, in each iteration, pairs of neighboring nodes update their estimates according to the principle of traditional delay-and-sum (DSB) beamformer. The estimation of the DDSB converges asymptotically to the optimal solution of the centralized beamformer. However, experimental study indicates that the noise reduction performance of the DDSB is at the expense of a higher communication cost, which can be a serious drawback in practical applications.

Therefore, in the second part of this thesis, a clique-based distributed beamformer (CbDB) has been proposed to reduce communication costs of the original DDSB algorithm. In the CbDB, nodes in two neighboring non-overlapping cliques update their estimates simultaneously per iteration. Since each non-overlapping clique consists of multiple nodes, the CbDB allows more nodes to update their estimates and leads to

lower communication costs than the original DDSB algorithm. Furthermore, theoretical and experimental studies have shown that the CbDB converges to the centralized beamformer and is more robust for sensor nodes failures in WASNs.

In the third part of this thesis, we propose a privacy preserving minimum variance distortionless response (MVDR) beamformer for speech enhancement in WASNs. Different wireless devices in WASNs generally belong to different users. We consider a scenario where a user joins the WASN and estimates his desired source via the WASN, but wants to keep his source of interest private. To introduce a distributed MVDR beamformer in such scenario, a distributed approach is first proposed for recursively estimation of the inverse of the correlation matrix in randomly connected WASNs. This distributed approach is based on the fact that using the Sherman-Morrison formula, estimation of the inverse of the correlation matrix can be seen as a consensus problem. By hiding the steering vector, the privacy preserving MVDR beamformer can reach the same noise reduction performance as its centralized version.

In the final part of this thesis, we investigate clock synchronization problems for multi-microphone speech enhancement in WASNs. Each wireless device in WASNs is equipped with an independent clock oscillator, and therefore clock differences are inevitable. However, clock differences between capturing devices will cause signal drift and lead to severe performance degradation of multi-microphone noise reduction algorithms. We provide theoretical analysis of the effect of clock synchronization problems on beamforming technologies and evaluate the use of three different clock synchronization algorithms in the context of multi-microphone noise reduction. Our experimental study shows that the achieved accuracy of the three clock synchronization algorithms enables sufficient accuracy of clock synchronization for the MVDR beamformer in ideal scenarios. However, in practical scenarios with measurement uncertainty or noise, the output of the MVDR beamformer with time-stamp based clock synchronization algorithms gets degraded, while the accuracy of signal based clock synchronization algorithms is still enough for the MVDR beamformer, albeit at a much higher communication cost.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

With the help of recent technology advances in electronic systems and communication devices, speech processing systems have been further developed and play an increasingly important role in our daily life, e.g., to facilitate human-to-human or human-to-computer communication. However, in many speech communication applications, such as haring aids, mobile telephony, telephone conferencing systems and human-to-computer communication systems, the microphones are placed in an environment that contains distortions like background noise and reverberation. In such noisy environments, the speech quality and intelligibility of the recorded speech can get severely degraded. Speech enhancement algorithms can be used to improve speech quality and intelligibility, resulting in pleasant sounding and understandable speech.

In the last few decades, a large number of speech enhancement algorithms have been proposed to reduce or eliminate noise and improve speech quality and intelligibility [1][2][3][4][5]. Speech enhancement algorithms can be divided into two classes: single-microphone and multi-microphone speech enhancement techniques. While single-microphone speech enhancement algorithms estimate the clean speech signal using the observed noisy signal recorded with a single microphone, multi-microphone speech enhancement algorithms use the observed noisy speech signal from multiple microphones or microphone arrays. Although single-microphone noise reduction algorithms can improve the speech quality and intelligibility to some extent [6], improvements are generally modest as they can only utilize the spectral information, see e.g., [1][7][8][4]. Compared with single-microphone speech enhancement algorithms, multi-microphone speech enhancement algorithms have in general a better noise reduction performance, since they can also exploit spatial information and can adapt the amount of amplification with respect to direction. As such, these systems can eliminate interfering signals coming from directions different from those of the target sources, see e.g., [9][10][2][11]. Multi-micrphone noise reduction algorithms rely on the basic concept that signals recorded by microphones at different locations are delayed (and scaled) versions of each other. By adding these different microphone signals, while compensating for their mutual delay in the right way, a direction (location) dependent amplification is applied. This allows to differentiate the amount of sound suppression with respect to the different locations in the environment. Alternatively, multi-microphone algorithms are also often referred to as beamforming techniques, since the direction dependent amplification can be interpreted as a beam that is steered in a certain direction, amplifying the target source, while suppressing sounds from other directions.

Even though microphone arrays are rather conventional, they are often used in the context of multi-microphone speech enhancement. They generally lead to better quality and intelligibility than their single-microphone counterparts, e.g., [12]. Although multi-microphone speech enhancement algorithms can potentially improve speech quality and intelligibility of the recorded signals, the performance of multi-microphone noise reduction algorithms is still limited when using conventional microphone arrays. The performance of multi-microphone speech enhancement algorithms generally improves by increasing the number of microphones, but depends as well on the signal-to-noise ratio (SNR) between the target and disturbance at the individual

microphones. However, conventional microphone arrays usually consider a relatively small number of microphones, which is partly determined by the dimensions of the device. Consider for example a smart phone, tablet, or a hearing aid, which usually contains only at most two or three microphones. Further, the location of a microphone array is generally fixed, and the distances between the microphone array and the target sources can be relative large, resulting in low SNR of the recorded signals.

One promising new direction to overcome the limitations of microphone arrays and further improve the performance of multi-microphone noise reduction algorithms, is to use wireless acoustic sensor networks (WASNs). A WASN is a network where a set of acoustic sensor nodes, each containing a single microphone or a small microphone array and an individual signal processing unit, are connected via wireless links. Recent advances in Micro-Electro-Mechanical Systems (MEMS) enabled the emergence of these small, low-power and low-cost sensor nodes. With WASNs, it is possible to use more microphones at positions that are not limited by just one device, and break the limitations of conventional microphone arrays. Moreover, as the sensor nodes in a WASN are not limited anymore by the physical dimensions of a single device, the nodes can cover a much larger area than conventional microphone arrays. This allows to place sensor nodes at locations out of reach of conventional arrays, e.g., close to target sources, providing a higher SNR.

A possible setup for a WASN is one that contains a fusion center, to which all sensor nodes are able to communicate (directly or indirectly via relay nodes). The observed signals are then transmitted to the fusion center, and processed using conventional multi-microphone noise reduction algorithms, e.g., [13]. Such a fusion center can be one of the devices that is a part of the network. However, due to power limitations, a limited transmission range and privacy considerations, such a fusion center may be undesirable in many applications. Moreover, such a fusion center is not robust, since a breakdown of the fusion center (e.g. in the case that the device is turned off or gets out of reach for other nodes) implies a complete breakdown of the WASN for all users. To realize speech enhancement without a fusion center, it is necessary to employ distributed speech enhancement algorithms, where the nodes process data locally and communicate only with their neighbors. Often, distributed signal processing is better scalable than centralized processing for large WASNs, since local processing can reduce computational complexity and the required communication bandwidth, as multiple signals are locally combined, requiring only transmission of the end result. Moreover, in the case that a node leaves the network, the remaining nodes can in general still perform multi-microphone noise reduction, albeit with a different network topology and one node less. In the case of centralized processing, if the fusion center or other nodes that play a crucial role in transmission towards the fusion center break down or leave the network, this will have dramatic impact on the ability of the complete network to perform its task. Therefore, unlike conventional centralized beamforming technologies where the observation signals of all microphones are gathered and processed in a fusion center, distributed speech enhancement algorithms aim to perform beamforming principles in distributed way (every node gathers observations from its neighboring nodes and then processes speech enhancement algorithms, rather than sending its information to the fusion center and receiving the final output from

the fusion center) and obtain the same noise reduction performance as conventional centralized speech enhancement algorithms.

Recently, there has been an increased interest for distributed multi-microphone noise reduction, leading to various algorithms for speech enhancement in a WASN, e.g., [14][15][16][17]. One of the first algorithm in this category is the distributed multi-channel Wiener filter, which was first proposed in [14] to estimate a single target source with a binaural hearing aid where both hearing aids contain multiple microphones and are connect via a wireless link. With this algorithm, each hearing aid is supposed to work as a data sink, gathering compressed signals from the other neighboring hearing aid, and estimates the optimal spatial filter coefficients in an iterative fashion. Later, several extensions were proposed to generalize this framework to WASNs, e.g., [18][15][19][16]. In general, these distributed noise reduction algorithms are assumed to operate in a WASN with a special network topology, such as a fully connected topology or a tree topology. However, WASNs may be dynamic as nodes may join or leave the network due to a defect or an empty battery, resulting in unpredictable changes in network size and topology. As a consequence, these distributed algorithms cannot always be used reliably in a WASN. Further, WASNs are generally randomly connected due to wireless communication range. To construct specific network topologies, some available links may be pruned, or some extra links between nodes with long distances have to be constructed, which makes those distributed speech enhancement algorithms suboptimal. Therefore, distributed noise reduction algorithms without network topology constraints are important for WASNs.

Although WASNs offer many advantages for multi-microphone speech enhancement, it also comes with new risks. Among these risks is the fact that privacy of the users is not always guaranteed to be preserved. With conventional microphone arrays, e.g., consider a hearing aid, the only user of the device is the owner. In the distributed setup, the WASN can be formed by devices that are not any longer owned by the user himself, leading to serious privacy issues. One example could be the situation, where a hearing aid user makes use of the WASN to increase the intelligibility of a conversation he is having during a cocktail party. His hearing aid devices are therefore shared with the available WASN. Even though the hearing aid user would like to use the WASN to estimate his signal of interest, he might not want to share to which source or conversation in the environment he is interested in. Information privacy might be a serious problem in distributed signal processing, since the multiple sensors or wireless devices in a WASN can be owned by many different users and private data or information may become public with such distributed signal processing. In the context of speech enhancement in a WASN, such privacy problems have been first considered in [20] and [21] for two scenarios. The scenario in [20] considered the case where a user keeps the exact source of interest private for other users, while [21] considered the scenario where eavesdropping by untrusted third parties is overcome. More specifically, both papers employed homomorphic encryption [22] to realize privacy preservation. However, homomorphic encryption is computationally very complex and requires very high bit rates for data transmission. In the given application of a WASN, it is thus difficult to perform homomorphic encryption, as both power and computational capacity of sensor nodes is limited.

Another important problem in distributed multi-microphone signal processing is the fact that each device in the network has its own individual clock. Multi-microphone noise reduction algorithms heavily depend on timing information, since they usually employ the delay that is experienced when an acoustical signal is observed at different positions. Thus, their performance will heavily degrade when these clocks are not synchronized. Most of the existing distributed multi-microphone noise reduction algorithms are based on the implicit assumption that the internal clocks are synchronized, see e.g., [15][16][23][24]. In a practical WSN, clock differences between nodes are inevitable, since each node is equipped with an independent clock oscillator. This will introduce clock differences between nodes. Such clock differences can cause unwanted time differences between the observed signals at the different nodes, since signals originating from different microphones are sampled at different sampling rates, finally leading to performance degradation of the multi-microphone enhancement algorithm. Although the clock synchronization problem is neglected in most contributions on distributed multi-microphone speech enhancement, several clock synchronization algorithms have been developed, see e.g., [25][26][27][13]. In general, these algorithms are not specifically developed for distributed speech processing, but originate from different contexts. Moreover, different clock synchronization algorithms are generally based on different principles and assumptions, which can affect the accuracy and robustness of clock synchronization. However, currently it is unclear to which extent the accuracy effects the performance of multi-channel signal processing for speech enhancement.

1.2 Research Questions

In this thesis, we address distributed speech enhancement algorithms for WASNs by the following assumptions.

1. WASNs are randomly connected and none of the nodes in the network will act as a fusion center.

2. WASNs are used by multiple users. Different users may be interested in different speakers, and they want to keep the source they are interested in private.

3. Each node in WASNs is equipped with an independent oscillator. Clock differences between different nodes need to be investigated.

Based on these assumptions, we formulate the following research questions and their motivations.

Question 1. How can we develop distributed algorithms that perform in randomly connected WASNs via local processing and improve the speech quality and intelligibility in a similar way as centralized algorithms?

In the previous section, we have mentioned that many existing distributed speech enhancement algorithms can only be performed in WASNs with specialized network

topologies, such as a fully connected or tree connected network. In Chapters 3, 4 and 5, we will not only develop distributed approaches to perform noise reduction in WASNs without network topology constraints, but also measure the mean square error (MSE) between the output of the proposed algorithms and the output of the centralized noise reduction algorithms, which we consider as the optimal solution. We also compare the noise reduction performance of the proposed algorithms with those of existing distributed speech enhancement algorithms.

Question 2. How to effectively reduce communication cost in distributed speech enhancement algorithms?

Communication cost of distributed algorithms is an important measure in WASNs, since it is inversely proportional to service life of WASNs. To improve the usability of the proposed distributed speech enhancement algorithms in WASNs, we have to assess their communication cost and find an efficient way to reduce this. On the other hand, robustness of distributed speech enhancement algorithms is also an important measure in WASNs, since wireless devices (nodes) may leave the network due to a defect or an empty battery. Therefore, a reduction of the communication cost of an algorithm should not come at the expense its robustness. This question will be answered in Chapter 4.

Question 3. How to estimate the inverse correlation matrix in distributed way?

Many multi-microphone noise reduction algorithms depend on the inverse of the noise or noisy correlation matrix, e.g., the MVDR beamformer or the multi-channel Wiener filter. In practical, correlation matrices are usually estimated recursively by exponential smoothing. Existing methods for distributed estimation of correlation matrices require specialized network topologies, such as fully connected networks or tree connected networks. In Chapter 5, we develop a distributed algorithm to estimate correlation matrices by exponential smoothing in randomly connected WASNs. In order to measure the performance of the proposed algorithm, we further introduce a distributed beamforming technology based on the proposed method, and measure the noise reduction performance of the distributed beamformer.

Question 4. How to develop distributed methods for speech enhancement in privacy preserving WASNs?

Privacy preservation is a challenge topic when exploring distributed speech enhancement algorithms in WASNs, since WASNs generally formed and owned by multiple users. To include the concept of privacy preservation in distributed noise reduction algorithms in randomly connected WASNs, in Chapter 5, we present a method where each user can estimate a different signal of interest from a mix of many different signals by means of distributed beamforming technologies without the need to reveal the source of interest to other entities in the network.

Question 5. How does clock synchronization problems affect multi-microphone noise reduction and what effect do the clock synchronization algorithms have on multi-microphone noise reduction?

Although often neglected in the development of distributed multi-microphone noise reduction algorithms, clock synchronization plays an important role in more practical setups. One reason is the fact that clock differences between capturing devices will cause signal drift. Despite general knowledge that the noise reduction performance of multi-microphone noise reduction algorithms will be affected by clock synchronization problems, an analysis about the effect of clock synchronization problems on multi-microphone speech enhancement is still missing. In Chapter 6, we first perform an initial study on the effect of clock synchronization problems on multi-microphone signal processing in a distributed setup, and then we give an overview of three clock synchronization algorithms, that can potentially be used in WASNs. To use those clock synchronization algorithms for multi-microphone noise reduction, the clock synchronization have to be accurate enough. Furthermore, as different clock synchronization algorithms are based on different theoretical frameworks, we discuss their advantages and drawbacks for multi-microphone noise reduction processing.

1.3 Structure of the Thesis

This thesis consists of seven chapters. In the current chapter, we give a brief overview of the state of the art and current trends in distributed multi-microphone signal processing for speech enhancement in a WASN, and briefly introduce the main research topics of this thesis.

Chapter 2 This chapter provides the necessary background in order to read the following chapters. This chapter first introduces the basic notation and the speech enhancement problem statement. Later, a basic introduction on conventional beamforming technologies, wireless acoustic sensor networks and distributed consensus problems are provided. The final section of this chapter presents an overview of some existing state-of-the-art distributed noise reduction algorithms for speech enhancement.

Chapter 3 In this chapter we investigate the use of randomized gossip for distributed speech enhancement and present a distributed delay and sum beamformer (DDSB). The algorithm aims to estimate the desired signal at each node by communicating only with its neighbors in a randomly connected WASN. Based on the communication schemes of the randomized gossip algorithm, we first provide an asynchronous DDSB, where each pair of neighboring nodes updates its data asynchronously. Then, we introduce an improved general distributed synchronous averaging algorithm (IGDSA), which can be used in any connected network, and combine that with the DDSB algorithm where multiple node pairs can update their estimates simultaneously. Convergence analysis and the simulation results show that the DDSB using several different updating schemes can reach the same performance as the centralized beamformer with enough message transmissions, and the proposed IGDSA convergences much faster than the original synchronous communication scheme. Moreover, comparisons are performed with several existing distributed speech enhancement algorithms from literature. In the simulated scenario, the DDSB leads to performance improvement at the expense of a higher communication cost. However, in contrast to other reference

methods, which are constraint to perform in a network with special network topology (e.g., fully connected or tree connected), the DDSB can be applied in any randomly connected network.

Chapter 4 To improve convergence speed and reduce communication cost of the proposed DDSB algorithm, which is presented in Chapter 3, a new clique-based distributed beamformer (CbDB) is proposed. Unlike the DDSB, where estimates are updated across two neighboring nodes, the CbDB updates estimates across two neighboring non-overlapping cliques. Theoretical and experimental analysis shows that the CbDB improves the convergence speed of the DDSB. Moreover, experimental results also show that the CbDB is more robust than a reference algorithm that is based on clusters, since cliques generally have a better connectivity than clusters.

Chapter 5 In this chapter, we consider a privacy preserving scenario where users in the network want to perform distributed target source estimation with a WASN, without revealing the actual source of interest to other entities in the network. Moreover, we consider distributed estimation of the inverse noise or noisy correlation matrix, which is an important aspect for distributed multi-microphone noise reduction in WASNs and in general a challenging problem. To make both privacy preservation as well as distributed multi-microphone noise reduction possible, we make use of the fact that recursive estimation of the inverse correlation matrix can be structured as a consensus problem and can be realized in a distributed manner via the randomized gossip algorithm. This makes it possible to compute the MVDR in a distributed manner without revealing the steering vector to any of the other entities in the network, and providing privacy about the actual source of interest. However, theoretical analysis and numerical simulations show that the convergence error between the gossip-based estimated correlation matrix and the centralized estimated correlation matrix accumulates across time. To eliminate this convergence error, a clique-based algorithm for distributed estimation of the inverse correlation matrix (CbDECM) is proposed. Further, we investigate the performance of the presented CbDECM algorithm in combination with a distributed privacy persevering MVDR beamformer, where information about the actual source of interest is kept private. Theoretical and experimental analysis show that the proposed algorithm converges to the centralized MVDR beamformer.

Chapter 6 In a WASN, the local clocks of different nodes are usually not identical, since each node is equipped with an independent clock oscillator. Such clock difference between nodes may cause signal drift and severe performance degradation of multi-microphone signal processing, such as multi-microphone noise reduction algorithms. In this chapter, we first analyze the effect of clock synchronization problems on the performance of multi-microphone noise reduction. To facilitate a good analysis of the clock synchronization problem, this is done using synthetically generated signals and the delay and sum beamformer. Further, we investigate the use of clock synchronization algorithms for clock skew estimation and compensation, and evaluate the effects of clock synchronization algorithms on the noise reduction performance of the MVDR beamformer in simulated WASNs. This experimental study shows that the

achieved precision of clock synchronization enables sufficient accuracy of clock skew estimation and compensation for the MVDR beamformer either in a scenario with or without noise on the parameters required for clock synchronization.

Chapter 7 This chapter provides a summary and discussion of all the results of this thesis, and some insights for future research.

1.4 List of Papers

The following papers have been published by the author of this thesis during her Ph.D. studies:

Journals

[A] Y. Zeng and R. C. Hendriks. Distributed Estimation of the Inverse of the Correlation Matrix for Privacy Preserving Beamforming, *Elsevier signal processing*, 107:109-122, Feb. 2015.

[B] Y. Zeng and R. C. Hendriks. Distributed Delay and Sum Beamformer for Speech Enhancement via Randomized Gossip, *IEEE Trans. Audio, Speech and Language Processing*, 22(1):260-273, Jan. 2014.

Conferences

[a] Y. Zeng and R. C. Hendriks and N. D. Gaubitch. On clock synchronization for multi-microphone speech processing in wireless acoustic sensor networks, *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Brisbane, Australia, April 2015.

[b] Y. Zeng, R. C. Hendriks and R. Heusdens. Clique-based Distributed Beamforming for Speech Enhancement in Wireless Sensor Networks, *Proc. European Signal Processing Conference*, Marrakesh, Morocco, September 2013.

[c] Y. Zeng and R. C. Hendriks. Distributed Delay and Sum Beamformer in Regular Networks Based on Synchronous Randomized Gossip, In *Proc. Int. Workshop on Acoustic Echo and Noise Control*, Aachen, Germany, September 2012.

[d] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng and W. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing, In *Proc. Int. Workshop on Acoustic Echo and Noise Control*, Aachen, Germany, September 2012.

[e] Y. Zeng and R. C. Hendriks. Distributed Delay and Sum Beamformer for Speech Enhancement in Wireless Sensor Networks via Randomized Gossip, In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 4037-4040, Kyoto, Japan, May 2012.

References

[1] Philipos C. Loizou. *Speech Enhancement - Theory and Practice*. CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, 2007.

[2] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[3] J. Benesty, S. Makino, and J. Chen. *Speech enhancement*. Springer, 2005.

[4] R. C. Hendriks, T. Gerkmann, and J. Jensen. *DFT-Domain Based Single-Microphone Noise Reduction for Speech Enhancement: A Survey of the State of the Art*. Morgan & Claypool, 2013.

[5] S. Doclo. *Multi-microphone noise reduction and dereverberation techniques for speech applications*. PhD thesis, Katholieke Universiteit Leuven, 2003.

[6] J. Jensen and R. C. Hendriks. Spectral magnitude minimum mean-square error estimation using binary and continuous gain functions. *IEEE Trans. Audio, Speech, Lang. Process.*, 20(1):92–102, Jan. 2012.

[7] Y. Ephraim and D. Malah. Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Trans. Audio, Speech, Lang. Process.*, 32(6):1109–1121, Dec. 1984.

[8] T. Lotter and P. Vary. Speech enhancement by MAP spectral amplitude estimation using a super-Gaussian speech model. *EURASIP Journal on Applied Signal Processing*, 2005(7):1110–1126, Jan. 2005.

[9] S. Doclo and M. Moonen. GSVD-based optimal filtering for single and multimicrophone speech enhancement. *IEEE Trans. Signal Process.*, 50(9):2230–2244, September 2002.

[10] S. Gannot, D. Burshtein, and E. Weinstein. Signal enhancement using beamforming and nonstationarity with applications to speech. *IEEE Trans. Signal Process.*, 49(8):1614–1626, Aug. 2001.

[11] Harry L. Van Trees. *Detection, estimation, and modulation theory. Part IV., Optimum array processing*. Wiley-Interscience, New York, 2002.

[12] K. Eneman et al. Evaluation of signal enhancement algorithms for hearing instruments. In *EURASIP Europ. Signal Process. Conf. (EUSIPCO)*, Lausanne, Switzerland, August 2008.

[13] S. Markovich-Golan, S. Gannot, and I. Cohen. Blind sampling rate offset estimation and compensation in wireless acoustic sensor networks with application to beamforming. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[14] S. Doclo, M. Moonen, T. Van den Bogaert, and J. Wouters. Reduced-bandwidth and distributed MWF-based noise reduction algorithms for binaural hearing aids. *IEEE Trans. Audio, Speech, Lang. Process.*, 17(1):38–51, Jan. 2009.

[15] A. Bertrand and M. Moonen. Distributed node-specific LCMV beamforming in wireless sensor networks. *IEEE Trans. Signal Process.*, 60(1):233–246, Jan. 2012.

[16] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[17] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer for speech enhancement via randomized gossip. *IEEE Trans. Audio, Speech, Lang. Process.*, 22:260 – 273, Jan. 2014.

[18] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[19] S. Markovich-Golan, S. Gannot, and I. Cohen. A reduced bandwidth binaural MVDR beamformer. In *Int. Workshop on Acoustic Echo and Noise Control*, Israel, Aug. 2010.

[20] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 7005–7009, Canada, May 2013.

[21] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy preserving distributed beamforming based on homomorphic encryption. In *Proc. European Signal Proc. Conf. Eusipco*, pages 7005–7009, Marrakesh, Morocco, 2013.

[22] C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security.*, 2007:1–10, Jan. 2007.

[23] Y. Zeng, R. C. Hendriks, and R. Heusdens. Clique-based distributed beamforming for speech enhancement in wireless sensor networks. In *Proc. European Signal Proc. Conf. (EUSIPCO)*, Marrakesh, Morocco, 2013.

[24] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng, and W. B. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[25] Y. C. Wu, Q. Chaudhari, and E. Serpedin. Clock synchronization of wireless sensor networks. *IEEE Signal Processing Magazine*, 21:260 – 273, Nov. 2013.

[26] L. Schenato and F. Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, 2011.

[27] R. T. Rajan and A. Veen. Joint ranging and clock synchronization for a wireless network. In *IEEE Int. Workshop on Computational Advances in Multi-Sensor Adaptibe Processing*, pages 297–300, 2011.

Chapter 2

Background

This chapter provides some background information in order to read this thesis. First we introduce the problem statement and notation in Section 2.1. This problem statement will serve as the target applications for many of the noise reduction algorithms that are described in this thesis. Since many distributed noise reduction algorithms are based on conventional beamforming technologies, we describe in Section 2.2 conventional beamforming for speech enhancement. In Section 2.3, we briefly provide the basic framework on which some existing distributed beamformers rely. In Section 2.4, we explain the concept and the major characteristics of WASNs. The distributed noise reduction algorithms in this thesis are mainly based on distributed consensus algorithms. Section 2.5 therefore describes the basics of average consensus problems in WSNs and state-of-the-art distributed techniques to solve them.

2.1 Problem Statement and Notation

Let us consider the scenario in Fig. 2.1, where a realization of the desired speech signal is recorded by a microphone array consisting of M microphones. We assume that the data model of the mth microphone signal consists of a target source degraded by additive noise, which is given by

$$y_m(n) = x_m(n) + v_m(n), m \in \{1, \dots, M\}, \tag{2.1}$$

where $y_m(n)$ is the observed signal at time sampling index n, and $x_m(n)$ and $v_m(n)$ denote the clean speech signal and noise signal, respectively, at the location of microphone m. Notice that, although $x_m(n)$ is referred to as the target speech component, $v_m(n)$ is not necessarily non-speech (e.g., competing speakers can be included in $v_m(n)$). In our notation, we use upper case letters to denote random variables and the corresponding lower case letters to denote their realizations. Further, we use non-bold symbols to represent scalars, while vectors and matrices are denoted by bold symbols. The noise component $v_m(n)$ and the speech $x_m(n)$ are the realization of the random variables $V_m(n)$ and $X_m(n)$, respectively, and $V_m(n)$ and $X_m(n)$ are assumed to be zero-mean and mutually uncorrelated. The observed signals can be transformed to the frequency domain using the short-time discrete Fourier transform (DFT). Applying the short-time DFT to the random time process, we obtain

$$Y_m(f,k) = X_m(f,k) + V_m(f,k), m \in \{1, \dots, M\}, \tag{2.2}$$

where $Y_m(f,k)$, $X_m(f,k)$ and $V_m(f,k)$ denote the noisy speech, target speech and noise DFT coefficient, respectively, at frequency-bin index f and time-frame index k. Let $[\cdot]^T$ denote the transposition of a vector or a matrix. We then define

$$\mathbf{Y}(f,k) = [Y_1(f,k), \cdots, Y_M(f,k)]^T$$

as the M-channel signal where the DFT coefficients $Y_m(f,k), \forall\, m$ are stacked. Similarly, $\mathbf{X}(f,k)$ and $\mathbf{V}(f,k)$ are defined in the same way as $\mathbf{Y}(f,k)$. Let $\mathbf{d}(f,k) = [d_1(f,k), \cdots, d_M(f,k)]^T$ denote the acoustic transfer function from the speech source to all microphones. In general, $\mathbf{d}(f,k)$ models all reflections of a source to a certain

location, i.e., the direct path and the reverberation, as also sketched in Fig. 2.1. However, notice that in this thesis we generally neglect reverberation in order to constrain our problem. Given $\mathbf{d}(f,k)$, the speech DFT vector $\mathbf{X}(f,k)$ for all microphones can therefore be written as

$$\mathbf{X}(f,k) = \mathbf{d}(f,k)S(f,k), \tag{2.3}$$

where $S(f,k)$ denotes the fth DFT coefficient at time-frame k for the clean speech at the target location.

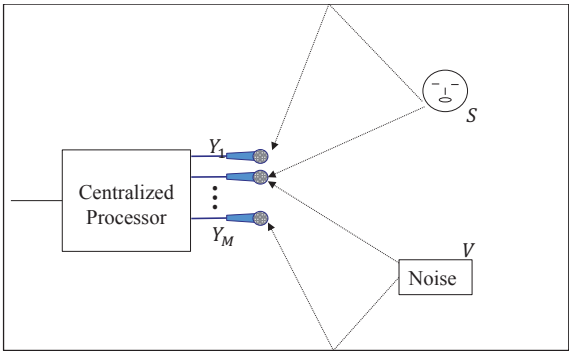


Figure 2.1: Typical conventional acoustic scenario with a microphone array.

The goal of noise reduction algorithms is to estimate the desired speech $S(f,k)$ from the observation $\mathbf{Y}(f,k) = \mathbf{X}(f,k) + \mathbf{V}(f,k) = \mathbf{d}(f,k)S(f,k) + \mathbf{V}(f,k)$. Let $\hat{S}(f,k)$ denote the estimator of $S(f,k)$. To obtain $\hat{S}(f,k)$, noise reduction algorithms first filter the microphone signals, and then sum the M filter outputs. Let $\mathbf{w}(f,k)$ denote a vector with filter coefficients, then the estimate $\hat{S}(f,k)$ is given by

$$\hat{S}(f,k) = \mathbf{w}^H(f,k)\mathbf{Y}(f,k) = \mathbf{w}^H(f,k)\mathbf{d}(f,k)S(f,k) + \mathbf{w}^H(f,k)\mathbf{V}(f,k), \tag{2.4}$$

where $(\cdot)^H$ denotes the Hermetian transposition of a matrix. Since the target and noise DFT coefficients are often assumed to be independent across time and frequency, we omit the time and frequency indices for notational convenience.

The power spectral density (PSD) of the output of a noise reduction algorithm is then given by

$$\mathbf{R}_{\hat{S}\hat{S}} = E\left[\hat{S}\hat{S}^H\right] = \mathbf{w}^H\mathbf{R}_{\mathbf{YY}}\mathbf{w}. \tag{2.5}$$

where $E[\cdot]$ denotes mathematical expectation and

$$\mathbf{R}_{\mathbf{YY}} = \mathbf{R}_{\mathbf{XX}} + \mathbf{R}_{\mathbf{VV}} + E\left[\mathbf{X}\mathbf{V}^H\right] + E\left[\mathbf{V}\mathbf{X}^H\right]. \tag{2.6}$$

Making use of the assumption that target speech and noise are statistically uncorrelated and zero-mean, i.e., $E\left[\mathbf{X}\mathbf{V}^H\right] = E\left[\mathbf{V}\mathbf{X}^H\right] = 0$, the noisy spectral covariance matrix $\mathbf{R}_{\mathbf{YY}}$ can then be written as

$$\mathbf{R}_{\mathbf{YY}} = \mathbf{R}_{\mathbf{XX}} + \mathbf{R}_{\mathbf{VV}}. \tag{2.7}$$

Using (2.7), (2.5) can be rewritten as

$$\mathbf{R}_{\hat{S}\hat{S}} = \mathbf{w}^H \mathbf{R}_{\mathbf{XX}} \mathbf{w} + \mathbf{w}^H \mathbf{R}_{\mathbf{VV}} \mathbf{w}, \tag{2.8}$$

where $\mathbf{R}_{\mathbf{XX}} = E\left[\mathbf{X}\mathbf{X}^H\right]$ is the PSD matrix of the speech signal and $\mathbf{R}_{\mathbf{VV}}$ is the PSD matrix of the noise field. Further, in coherent noise field where the noise signals on different microphones are strongly correlated, $\mathbf{R}_{\mathbf{VV}}$ is not a full-rank matrix, and in incoherent noise fields where the noise measured at any given spatial location is uncorrelated with the noise measured at all other locations, $\mathbf{R}_{\mathbf{VV}}$ is a full-rank matrix.

2.2 Conventional Beamforming Technologies

In a multi-microphone noise reduction algorithm, several microphone signals are combined in order to estimate the signal of interest. Multi-microphone noise reduction algorithms can be written as the concatenation of a beamformer and a single-microphone noise reduction algorithm, see e.g., [1] [2] [3]. The latter performs only temporal filtering, while the former performs spatial filtering. With a beamformer (see e.g., [4]) it is thus possible to amplify signals coming from certain directions. There are many types of beamforming techniques for speech enhancement. For a survey on beamforming technologies, see e.g., [5] [4].

Conventional beamforming methods can be classified in fixed and adaptive beamforming. Fixed beamformers aim to estimate the speech signal from certain direction, and suppress the background noise not coming from the same direction as the speech source using fixed filters. Specifically, the filter coefficients in a fixed beamformer are chosen to present a specified response for all signals and interference scenarios. In adaptive beamformers, filter coefficients are solutions to optimization problems, and updated based on microphone signals. Thus, adaptive beamformers enable to adapt to changing acoustic scenarios and generally have better performance than fixed beamformers. In this thesis, we will mainly focus on the minimum variance distortionless response (MVDR) beamformer [5] [6], which is a classic adaptive beamformer. This beamforming method forms the backbone of the distributed delay-and-sum beamformer (DDSB) and distributed MVDR beamformer, which will be introduced in Chapters 3 and 5, respectively.

2.2.1 MVDR Beamformer

The MVDR beamformer is often used in the field of microphone array signal processing, e.g., [5] [6], and is a special case of the the linearly constrained minimum variance (LCMV) beamformer [7] [6]. The LCMV beamformer was proposed by Frost in [7] and can be obtained by minimizing the beamformer output power subject to multiple constraint of maintaining constant response in directions of interest. The MVDR beamformer is a special case as it uses a single constraint on a single target source. To reduce noise without speech distortion, the filter coefficient $\mathbf{w}$ of the MVDR beamformer can be obtained by minimizing the beamformer output power and subject to no speech cancellation or distortion, that is

$$\min_{\mathbf{w}} \mathbf{w}^H \mathbf{R}_{\mathbf{YY}} \mathbf{w}, \tag{2.9}$$

subject to

$$\mathbf{w}^H\mathbf{d} = 1. \tag{2.10}$$

The method of Lagrange multiplier (e.g., [8]), can be used to solve the optimization problem in (2.9). With the Lagrange multiplier λ, the cost function J is given by

$$J(\mathbf{w}) = \mathbf{w}^H\mathbf{R}_{\mathbf{YY}}\mathbf{w} + \lambda\left(\mathbf{w}^H\mathbf{d} - 1\right). \tag{2.11}$$

The complex derivative (see e.g., [8]) of $J(\mathbf{w})$ with respect to filter coefficients $\mathbf{w}$ is given by

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}^H} = 2\mathbf{R}_{\mathbf{YY}}\mathbf{w} + \mathbf{d}\lambda. \tag{2.12}$$

Setting (2.12) equal to zero, yields the solution

$$\mathbf{w} = -\frac{1}{2}\mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{d}\lambda. \tag{2.13}$$

Using the constraint (2.10), the solution of Lagrange multiplier λ is

$$\lambda = -\frac{1}{\mathbf{d}^H\mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{d}}. \tag{2.14}$$

Substituting λ in (2.13), we get the solution of the desired filter $\mathbf{w}$, that is

$$\mathbf{w}_{\text{MVDR}_1} = \frac{\mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{d}}{\mathbf{d}^H\mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{d}}. \tag{2.15}$$

Let $P_S = E\left[SS^*\right]$ denote the PSD of the clean speech signal at the target location with $(\cdot)^*$ the convolution operator. The PSD matrix $\mathbf{R}_{\mathbf{XX}}$ can then be written as

$$\mathbf{R}_{\mathbf{XX}} = P_S\mathbf{d}\mathbf{d}^H. \tag{2.16}$$

Using the assumption that the clean speech and noise are uncorrelated, (2.7) can be written as

$$\mathbf{R}_{\mathbf{YY}} = P_S\mathbf{d}\mathbf{d}^H + \mathbf{R}_{\mathbf{VV}}. \tag{2.17}$$

Using the matrix inversion lemma [6], $\mathbf{R}_{\mathbf{YY}}^{-1}$ can be written as

$$\mathbf{R}_{\mathbf{YY}}^{-1} = \mathbf{R}_{\mathbf{VV}}^{-1} - \frac{P_S\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}\mathbf{d}^H\mathbf{R}_{\mathbf{VV}}^{-1}}{1 + P_S\mathbf{d}^H\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}}. \tag{2.18}$$

Right multiplying with $\mathbf{d}$ on both sides of (2.18), and substituting $\mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{d}$ into (2.15), the MVDR filter coefficients can be written as

$$\mathbf{w}_{\text{MVDR}_2} = \frac{\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}}{\mathbf{d}^H\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}}. \tag{2.19}$$

The noise correlation matrix $\mathbf{R}_{\mathbf{VV}}$ can be estimated during noise-only periods with a voice activity detection (VAD) (see e.g., [9] [10]), while $\mathbf{R}_{\mathbf{YY}}$ can be estimated during speech+noise periods. Using the adaptive filters in (2.15) and (2.19), background noise in an acoustic scenario can be suppressed. By properly updating $\mathbf{R}_{\mathbf{VV}}$ or $\mathbf{R}_{\mathbf{YY}}$, the filter coefficients are adapted towards the changing noise environment.

2.2.2 Delay-and-Sum Beamformer

The delay-and-sum-beamformer (DSB) is a special case of the MVDR beamformer. Assuming that the noise across microphones is spatially uncorrelated, the off-diagonal elements in $\mathbf{R_{VV}}$ are equal to zero. The noise correlation matrix can then be expressed as

$$\mathbf{R_{VV}} = \text{diag}\left\{\sigma^2_{V_1}, \cdots, \sigma^2_{V_M}\right\}, \tag{2.20}$$

where $\sigma^2_{V_m}$ is the PSD of the noise process at microphone m. Notice that this assumption is valid in an incoherent noise field. This assumption is, even though not always fully true, often made for simplicity. Combining the MVDR filter from (2.19) with (2.20), the optimal solution of the DSB can be written as

$$\hat{S} = \frac{\sum_{m=1}^{M} d_m^* \sigma_{V_m}^{-2} Y_m}{\sum_{m=1}^{M} d_m^* \sigma_{V_m}^{-2} d_m}. \tag{2.21}$$

It should be noted that the beamformer in (2.21) allows for different noise PSDs per microphone, while many definitions of the DSB assume the same noise PSD for all microphones (see, e.g., [5]). Thus, compared to the standard DSB, the beamformer in (2.21) is more general. The DSB is appropriate for incoherent noise fields, since the noise DFT coefficient V_m between different microphones in incoherent noise fields are uncorrelated. In diffuse noise fields and/or that the distance between microphones is sufficiently large, the noise DFT coefficient V_m, $\forall m$ can be argued to be approximately spatially uncorrelated. In those noise fields, the DSB can reach the same noise reduction performance as the MVDR beamformer. However, in coherent noise fields where the noise DFT coefficients between the different microphones are correlated, the noise reduction performance of the MVDR beamformer is better than that of the DSB, albeit at a higher computational complexity, due to calculation of the inverse of the correlation matrix in the MVDR.

2.2.3 Multi-Channel Wiener Filter

The multichannel Wiener filter (MWF) is an optimal filter and is designed to minimize the estimation error e between the estimate $\hat{S}$ and the desired speech signal S (i.e., $e = S - \hat{S}$) in a statistical way. Considering the data model in (2.2), the MWF filter is obtained by minimizing the mean-square error cost function of the error e, that is

$$J(\mathbf{w}) = E\left[\left|S - \mathbf{w}^H \mathbf{Y}\right|^2\right]. \tag{2.22}$$

Taking the derivative of $J(\mathbf{w})$ and setting it to zero, the MWF filter is given by

$$\mathbf{w}_{\text{MWF}} = \mathbf{R}_{\mathbf{YY}}^{-1} \mathbf{R}_{\mathbf{Y}S}, \tag{2.23}$$

with $\mathbf{R}_{\mathbf{Y}S} = E\left[\mathbf{Y}S^H\right]$.

In practical applications, the MWF filter is generally used to estimate the clean speech component of a reference microphone, e.g., [11] [12]. Assuming that the first microphone is the reference microphone, the MWF filter is then used to estimate the

clean speech component of the first microphone, which is denoted as X_1, and can be obtained as $\mathbf{w}_{\text{MWF}} = \mathbf{R}_{\mathbf{YY}}^{-1}\mathbf{R}_{\mathbf{Y}X_1}$. Using the assumption that the noise signal and the desired speech signal are uncorrelated, the correlation matrix $\mathbf{R}_{\mathbf{Y}X_1}$ can be obtained as

$$\mathbf{R}_{\mathbf{Y}X_1} = \mathbf{R}_{\mathbf{XX}}\mathbf{e}_1, \tag{2.24}$$

where $\mathbf{e}_1 = [1, 0, \cdots, 0]^T$ is an M-dimensional vector with the first entry set to 1, and all other entries set to 0. Equation (2.23) shows that the acoustic transfer function is not explicitly required in the formulation of the MWF. To estimate $\mathbf{R}_{\mathbf{XX}}$, one can estimate $\mathbf{R}_{\mathbf{YY}}$ and $\mathbf{R}_{\mathbf{VV}}$ (e.g., during noise-only using a voice activity detector) and use the fact that $\mathbf{R}_{\mathbf{XX}} = \mathbf{R}_{\mathbf{YY}} - \mathbf{R}_{\mathbf{VV}}$.

2.3 Basic Framework of Some Existing Distributed Beamformers

In this section, we explain the basic framework of a class of existing distributed beamformers, e.g., [13] [14] [15], which was first proposed in a distributed MWF-based noise reduction algorithm for binaural hearing aids [13].

Consider a WASN where N sensor nodes are connected via wireless links and each node consists of a microphone array with M_i microphones. The data model of the observed signal of each microphone is given by (2.2). Let $\mathbf{Y}_i = [Y_{i,1}, \cdots, Y_{i,M_i}]^T$ denote the M_i-channel noisy DFT coefficients at node i. Similar as $\mathbf{Y}_i$, $\mathbf{X}_i$ and $\mathbf{V}_i$ denote the M_i-channel speech DFT coefficients and noise DFT coefficients, respectively, at node i.

In conventional centralized MWF algorithms, each node i has to send its M_i-channel microphone signals to the center processor, since conventional MWF algorithms require to access all M-channel microphone signals to estimate the clean speech component of the reference microphone. Unlike conventional noise reduction algorithms, the distributed algorithms as proposed in [13] [14] [15] estimate the clean speech component of the reference microphone at each node i in an iterative way without the need for each node to send its M_i-channel microphone signals to the center processor. Notice that each node i in distributed scheme will locally estimate the clean speech component of its reference microphone, which generally is its first microphone and is denoted by $X_{i,1}$ (the clean speech DFT coefficient of the first microphone at ith node). At each iteration t, each node i estimates $X_{i,1}$ using its own microphone signals $\mathbf{Y}_i$ and the signals received from its neighbors. A basic scheme of the distributed noise reduction algorithms with two nodes is depicted in Fig. 2.2.

Since we consider a single desired speech source in the WASN, each node i first estimates the desired speech signal by applying a compression filter $\mathbf{w}_{ii}$ to its microphone signals $\mathbf{Y}_i$, that is

$$Z_{ii}(t) = \mathbf{w}_{ii}^H(t)\mathbf{Y}_i, \tag{2.25}$$

where $\mathbf{w}_{ii}$ is the local filter coefficient of node i, and $Z_{ii}(t)$ denotes the local estimates of the desired signal at iteration t. Then, each node i updates its estimates by filtering and summing its microphone signals $\mathbf{Y}_i$ and the transmitted signal $\mathbf{Z}_i(t)$ from its

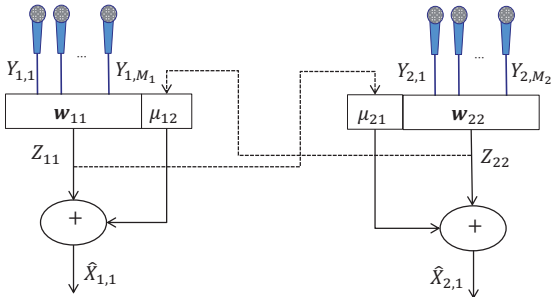


Figure 2.2: Basic diagram of distributed noise reduction algorithms in a WASN after [13] [14] [15] with two nodes.

neighboring nodes with $\mathbf{Z}_i(t) = \{Z_{jj}(t)|j \in \mathcal{N}_i\}$, that is

$$\hat{X}_{i,1}(t) = \mathbf{w}_{ii}^H(t+1)\mathbf{Y}_i + \mu_i(t+1)\mathbf{Z}_i(t), \tag{2.26}$$

where $\mu_i(t+1)$ is a $|\mathcal{N}_i|$-dimensional vector that is applied to the signal $\mathbf{Z}_i(t)$, and $\hat{X}_{i,1}(t)$ is the estimates of the desired signal at node i. In particular, the basic framework of the distributed noise reduction consists of the following steps:

1. Initialize the iteration index $t = 0$, and initialize the filters $\mathbf{w}_{ii}(0)$ and $\mu_i(0)$, $\forall i$ with non-zero random vectors, respectively.

2. Each node i computes $Z_{ii}(t)$ as given in (2.25), and transmit this signal to its neighboring nodes.

3. A node i is selected to update its filters $\mathbf{w}_{ii}(t+1)$ and $\mu_i(t+1)$ based on its local microphone signals $\mathbf{Y}_i$ and the transmitted signals $\mathbf{Z}_i(t)$. This can be done by using principles of conventional beamforming technologies [5] [6]. For example, $\mathbf{w}_{ii}(t+1)$ and $\mu_i(t+1)$ can be updated using the principle of the MWF, which is minimizing the mean-square error between the output signal and the desired signal, i.e.

$$\begin{bmatrix} \mathbf{w}_{ii}(t+1) \\ \mu_i(t+1) \end{bmatrix} = \underset{\mathbf{w}_{ii},\mu_i}{\operatorname{argmin}} E\left\{ \left| X_{i,1} - \begin{bmatrix} \mathbf{w}_{ii} & \mu_i \end{bmatrix} \begin{bmatrix} \mathbf{Y}_i \\ \mathbf{Z}_i(t) \end{bmatrix} \right|^2 \right\} \tag{2.27}$$

   where $X_{i,1}$ is the speech component of the first/reference microphone of the node i.

4. $t = t + 1$ and change the updating node i.

5. Return to step 2).

For the above distributed scheme it has been shown that the amount of data transmitted by each node is reduced compared to centralized processing, since each node i

transmits M_i-channel signals in centralized processing and only transmits one-channel signal in the distributed scheme. Each node in this distributed scheme only needs to transmit a single-channel signal to its neighboring nodes and receives a $|\mathcal{N}_i|$-channel signals from its neighboring nodes. For comparison, each node in the centralized scheme has to transmit M_i-channel microphone signals to its neighboring nodes.

In [13] it is shown that this distributed procedure converges to the centralized solution in the case of a single desired speech source and two sensor nodes, and can reach the same noise reduction performance as the centralized processing. For more than two sensor nodes and multiple desired speech sources, the convergence analysis is given in [14], for which it turns out that this distributed processing converges to the centralized processing when a block of data $Z_{ii}, \forall i$ is iteratively re-estimated based on the observed signal $\mathbf{Y}$. This may require many transmissions and much computational power when the required number of iterations is large. Although the different iterations of this distributed processing can be spread out over different data blocks in a time-recursive implementation, such that at each iteration $Z_{ii}, \forall i$ is estimated using different observations of $\mathbf{Y}$ (e.g., the observed signal in different time-frames), this distributed scheme cannot reach the same noise reduction performance as the centralized processing [16]. Further, the noise reduction performance and convergence speed of this distributed processing depends on the updating order of the nodes in the network [17]. Moreover, this distributed processing is required to perform in specific network topologies, such as a fully connected topology and a tree topology, since it is not guaranteed to converge in a randomly connected network [18].

2.4 Wireless Acoustic Sensor Networks

A wireless acoustic sensor network (WASN) is a network where hundreds or even thousands of small sensor nodes can be connected through wireless links. As an example, consider Fig. 2.3. Each sensor node in a WASN is equipped with an acoustic sensor, such as a single microphone or a microphone array. Further, besides the capability of harvesting information with sensors, each node also consists of a processing unit and transmission unit, which can perform simple processing on the extracted data and transmission of the output of the processor to neighboring nodes. Since these nodes can communicate among each other via wireless communication links, a large number of such nodes can be placed to sense larger areas and positions that are otherwise hard to reach. Compared to conventional microphone arrays, WASNs can employ more microphones to cover a larger spatial field, and are not limited by the dimensions of a single device. The main characteristics of a WASN include

- **Multiple nodes:** Each node in a WASN is a device with multiple functions, such as sensing, data processing and wireless communication.
- **Dynamic network topology:** Sensor nodes can construct different network topologies via wireless communication. The specific topology depends on the position of the nodes as well as on their communication range. A few common network topologies are shown in Fig. 2.4. Recently, many researchers have addressed distributed noise reduction problems in WASNs with specific network

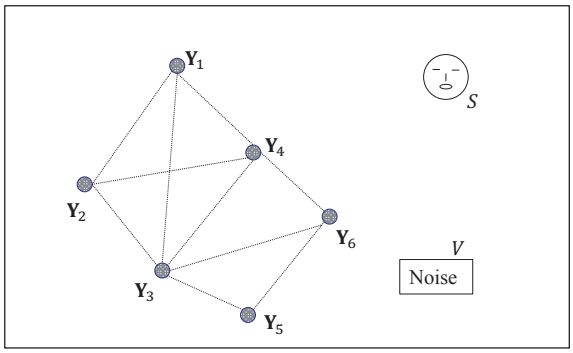
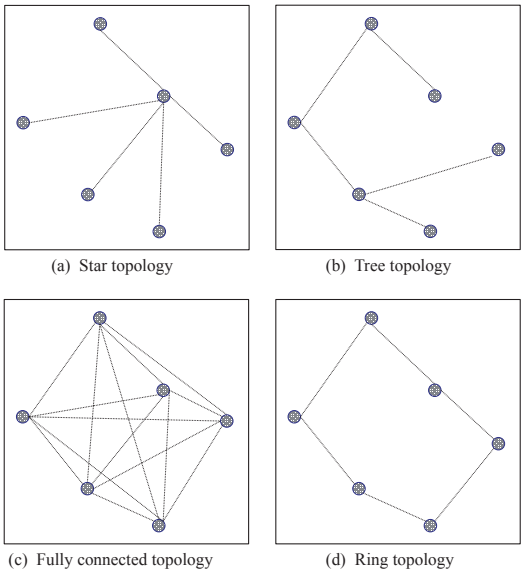


Figure 2.3: A randomly connected WASN with 6 sensor nodes.

topologies, such as a fully connected topology [14] [15] or a tree topology [18]. However, setting up a specific network topology usually requires an extra upper-layer protocol and communication. Such algorithms may be not robust against moving nodes, node failure or new nodes joining the network, which will lead to changes in network topology. Further, such algorithms are usually suboptimal, since some available links between nodes may be pruned, or some extra links between nodes with long distances have to be constructed to meet the demands of network topology. Algorithms without network topology constraints can be advantageous for WASNs.

- **Power consumption constraints:** Power consumption of individual sensor nodes is proportional to lifetime of WASNs. However, the nodes in WASNs are usually powered by batteries. A low-power consumption of the nodes is therefore important in practical applications. Computation and wireless communication (which includes data transmission and information processing), are the two main energy-consuming operations of a node. Therefore, lowering the computational complexity of data processing and minimizing the number of transmissions are important aspects for WASNs. The transmission power heavily depends on the distance between the different nodes and the network topology. Some specific network topologies, such as a star topology, usually consume more energy than a randomly connected network, since nodes in a randomly connected network only share data with nodes that are close by, and nodes in star topologies require more transmission power to transmit data to a certain master node. An effective way to reduce wireless communication and thus the amount of energy consumption, is to reduce the communication bandwidth. In distributed signal processing, communication bandwidth generally can be reduced by processing and compression of signals locally.



(a) Star topology (b) Tree topology

(c) Fully connected topology (d) Ring topology

Figure 2.4: Schematic diagram of network topologies.

2.5 Distributed Average Consensus Problems

In Chapters 3, 4 and 5, consensus techniques by means of randomized gossip methods are used to formulate distributed beamformers. Distributed average consensus problems were first studied for distributed computation in [19]. With the advances in WSNs, the distributed average consensus problems have attracted much attention for possible applications to sensor networks, see [20] [21] [22]. In this section, we introduce the average consensus problem statement and give a brief overview of the randomized gossip algorithm [23], which is the main backbone of distributed beamforming technologies in this thesis.

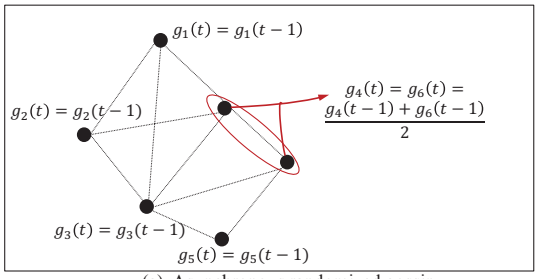
Consider a connected network $\mathcal{G}$ with N nodes. Let $g_i(t)$ denote the value of node i at the end of the tth tick of the global clock and let $g_i(0)$ denote an initial value of node i. The average of the initial values at each node i is then given by

$$g_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N} g_i(0). \tag{2.28}$$

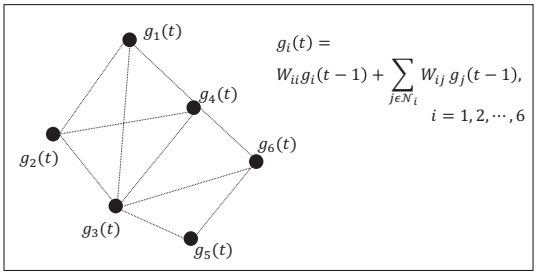
The objective of distributed average consensus algorithms is to find the average value g_{ave} at all nodes in the network by using local information and local communication.

Recently, gossip algorithms have been studied to solve averaging consensus problems without any requirement of specialized routing or network topology, e.g., [23]

[24] [25]. Gossip algorithms can be categorized into two classes: randomized and deterministic. In randomized gossip algorithms, each pair of neighboring nodes is chosen randomly based on a probabilistic model to update information (Fig. 2.5(a)), while neighboring nodes in deterministic gossip algorithms are chosen in a deterministic way (e.g., by using knowledge on the network topology) to update information (Fig. 2.5(b)).



(a) Asynchronous randomized gossip



(b) Deterministic gossip

Figure 2.5: Gossip processing of all nodes in a WASN at time slot t.

In deterministic gossip algorithms, at each iteration t, node i, $\forall i$ updates its estimate $g_i(t)$ with a convex combination of its own value value and the values from all of its neighbors (i.e., a linear combination of points with non-negative weights that sum up to one), e.g., [25], that is

$$g_i(t) = W_{ii}g_i(t-1) + \sum_{j\in\mathcal{N}_i} W_{ij}g_j(t-1), \tag{2.29}$$

where W_{ij} is the weight on $g_j(t-1)$ to calculate the updated value for node i with $W_{ij} = 0$ when $j \notin \mathcal{N}_i$. To guarantee all nodes in the network to converge to the average value g_{ave} for any initial value, the necessary condition for the weight matrix $\mathbf{W}$ is [25]

$$\lim_{t\to\infty} \mathbf{W}^t = \frac{\mathbf{1}\mathbf{1}^T}{N}, \tag{2.30}$$

with $\mathbf{1}$ denoting a vector of all ones.

A classical randomized gossip algorithm is pairwise randomized gossip [23], where random pairs of connected nodes iteratively and locally average their values until convergence to the average value g_{ave}. More specifically, at each iteration t, multiple node pairs, e.g., the pair (i, j), are randomly selected to communicate with each other, and update their estimates as

$$g_i(t) = g_j(t) = \frac{g_i(t-1) + g_j(t-1)}{2}. \tag{2.31}$$

Depending on the exact protocol, the randomized gossip algorithm can be performed asynchronously or synchronously. In asynchronous setups only one pair of neighboring nodes updates its estimates per iteration, while in synchronous communication schemes multiple pairs of neighboring nodes update their estimates simultaneously per iteration in the synchronous communication scheme. When each pair of neighboring nodes in $\mathcal{G}$ gossips frequently enough, the estimates of each node are guaranteed to converge to the average value g_{ave} [23].

Many contributions on distributed averaging have been proposed which improve convergence speed and reduce the number of required transmissions to reach consensus, e.g., [26] [27] [28] [29]. The algorithm in [26] is based on broadcast nature of the WSNs, where one node in each time-slot is randomly selected to broadcast its data to all neighboring nodes, and each neighbor then updates its estimates with the received information. Although the algorithm in [26] increases the convergence rate of the randomized gossip algorithms, it is not guaranteed to converge to the average value, since the networks global sum is not preserved. To guarantee that the broadcast gossip algorithm converges to the average value, a algorithm which incorporate the weighted gossip into the broadcast gossip algorithm was presented in [29]. In [27], the convergence speed of randomized gossip algorithms is improved by forming overlapping clusters of nodes, and subsequently averaging per cluster instead of per node-pair. A further improvement is obtained by averaging across two neighboring non-overlapping clusters as proposed in [28].

References

[1] K. U. Simmer, J. Bitzer, and C. Marro. Post-filtering techniques. In M. S. Brandstein and C. Ward, editors, *Microphone Arrays: Signal Processing Techniques and Applications*, pages 39–60. Springer, Berlin, 2001.

[2] R. C. Hendriks, R. Heusdens, U. Kjems, and J. Jensen. On optimal multi-channel mean-squared error estimators for speech enhancement. *IEEE Signal Process. Lett.*, 16(10):885–888, October 2009.

[3] R. Balan and J. Rosca. Microphone array speech enhancement by Bayesian estimation of spectral amplitude and phase. In *Proc. of IEEE Sensor Array and Multichannel Signal Processing Workshop*, 2002.

[4] B. D. Van Veen and K. M. Buckley. Beamforming: A versatile approach to spatial filtering. *IEEE ASSP Magazine*, 5(2):4–24, 1988.

[5] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[6] H. L. Van Trees. *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing*. John Wiley and Sons, New York, 2002.

[7] O. L. Frost III. An algorithm for linearly constrained adaptive array processing. *Proc. IEEE*, 60(8):926 – 933, August 1972.

[8] D. Brandwood. A complex gradient operator and its application in adaptive array theory. *Proc. IEEE*, 130(1):11 – 16, Feb. 1983.

[9] S. G. Tanyer and H. Ozer. Voice activity detection in nonstationary noise. *IEEE Trans. Speech Audio Processing*, 8(4):478–482, 2000.

[10] P. K. Ghosh, A. Tsiartas, and S. S. Narayanan. Robust voice activity detection using long-term signal variability. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(3):600–613, 2011.

[11] S. Doclo, A. Spriet, J. Wouters, and M. Moonen. Frequency-domain criterion for the speech distortion weighted multichannel wiener filter for robust noise reduction. *Speech Communication*, 49(7-8):636–656, 2007.

[12] B. Cornelis, M. Moonen, and J. Wouters. Performance analysis of multichannel wiener filter-based noise reduction in hearing aids under second order statistics estimation errors. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(5):1368–1381, 2011.

[13] S. Doclo, M. Moonen, T. Van den Bogaert, and J. Wouters. Reduced-bandwidth and distributed MWF-based noise reduction algorithms for binaural hearing aids. *IEEE Trans. Audio, Speech, Lang. Process.*, 17(1):38–51, Jan. 2009.

[14] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[15] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[16] A. Bertrand and M. Moonen. Robust distributed noise reduction in hearing aids with external acoustic sensor nodes. *EURASIP J. Adv. Sig. Proc.*, 2009, 2009.

[17] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part II: Simultaneous and asynchronous node updating. *IEEE Trans. Signal Process.*, 58(10):5292 –5306, Oct. 2010.

[18] A. Bertrand and M. Moonen. Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology. *IEEE Trans. Signal Process.*, 59(5):2196–2210, May. 2011.

[19] J. Tsitsiklis. *Problems in decentralized decision making and computation*. PhD thesis, MIT.

[20] L. Schenato and F. Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, 2011.

[21] D. S. Scherber and H. C. Papadopoulos. Locally constructed algorithms for distributed computations in ad-hoc networks. In Kannan Ramchandran, Janos Sztipanovits, Jennifer C. Hou, and Thrasyvoulos N. Pappas, editors, *IPSN*, pages 11–19. ACM, 2004.

[22] C. C. Moallemi and B. V. Roy. Consensus propagation. *IEEE Trans. Inf. Theory*, 52(11):4753–4766, 2006.

[23] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[24] F. Bnzit, V. D. Blondel, P. Thiran, J. N. Tsitsiklis, and M. Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *ISIT*, pages 1753–1757. IEEE, 2010.

[25] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Syst. Control Lett.*, 53(1):65–78, 2004.

[26] N. Wang, D. Li, and Z. Yin. Broadcast gossip algorithm with quantization. In *FSKD*, pages 2143–2147. IEEE, 2012.

[27] M. Zheng, M. Goldenbaum, S. Stanczak, and H. Yu. Fast average consensus in clustered wireless sensor networks by superposition gossiping. In *IEEE Wireless communications and networking conference*, pages 1982–1987, Jun. 2012.

[28] W. Li and H. Dai. Cluster-based distributed consensus. *IEEE Trans. Wireless Communications*, 8(1):28 –31, Jan. 2009.

[29] F. Iutzeler, P. Ciblat, W. Hachem, and J. Jakubowicz. A new broadcast based distributed averaging algorithm over wireless sensor networks. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 3117–3120, Kyoto, 2012.

Chapter 3

Distributed Delay and Sum Beamformer for Speech Enhancement via Randomized Gossip

---

This chapter is published as "Distributed Delay and Sum Beamformer for Speech Enhancement via Randomized Gossip", by Y. Zeng and R. C. Hendriks in the *IEEE Trans. Speech, Audio and Language Processing*, vol. 22, no. 1, pages 260-273, Jan. 2014.

3.1 Introduction

In many speech processing applications, such as mobile telephony, hearing aids, and human-machine communication systems, speech quality and intelligibility get severely degraded in noisy environments. In the last few decades, a large number of speech enhancement algorithms have been developed to improve the quality and intelligibility of noisy speech and to reduce or eliminate the acoustical noise in speech communication systems. Speech enhancement algorithms can be categorized into two classes: single-channel and multi-channel techniques. Although single-channel algorithms can improve quality and have been shown to be able to improve speech intelligibility to some extent [1], improvements are generally modest as they can utilize only the spectral information [2][3][4]. Multi-channel speech enhancement algorithms have in theory the potential to improve the speech quality and intelligibility by using both spectral and spatial information about the speech and the noise sources [5], [6]. However, this also requires additional information such as the sensor and source locations or the steering vectors, which are not always easy to estimate in practice. The performance of multi-channel speech enhancement algorithms generally increases with the number of microphones. However, conventional microphone arrays usually consider a relatively small number of microphones with fixed locations. Recently, advances in micro electro-mechanical systems (MEMS) enabled the emergence of small, low-cost and low-power smart acoustic sensors with multiple functions such as sensing, data processing and communication. Such smart sensors enable distributed sensing and extend the sensing range, and therefore the sensors can be placed closer to the desired sources and provide a higher signal-to-noise ratio (SNR). In addition, such acoustic sensors can construct networks via wireless links, often referred to as wireless acoustic sensor networks (WASNs) [7][8][9].

In principle, the observed signals of wireless microphones can be transmitted to a fusion center where all signals are processed. This enables the use of conventional centralized multi-channel noise reduction algorithms. However, due to privacy considerations, transmission range and battery limitations, such a fusion center may be undesirable in many applications. An alternative solution is to use distributed noise reduction algorithms, e.g., [10][11][12][13][14], where each node can process data locally and communicate with its neighbors, rather than with a fusion center.

In [10], a distributed multi-channel Wiener filter (DB-MWF) was proposed for the minimum mean squared error (MMSE) estimation of a single desired source in a binaural hearing aid where both hearing aids contain multiple microphones. Markovich-Golan [15] considered a special case of the DB-MWF algorithm and proposed a distributed minimum variance distortionless response (MVDR) beamformer for a similar binaural hearing aid system.

A more general case was presented in [16], [17], where multiple desired sources and $N (N \geq 2)$ nodes are considered in a so-called distributed adaptive node-specific signal estimation (DANSE) algorithm. The DANSE algorithm considers each node in the network as a data sink, gathering compressed signals from its neighbors, and estimates the optimal spatial filter coefficients in an iterative fashion. The DANSE algorithm was proposed for a fully connected network [16] and a network with a tree topology [17]. Later, a distributed LCMV beamformer was proposed in [11] by com-

bining the framework for the DANSE algorithm with the LCMV beamformer. Related to this, a time-recursive distributed generalized sidelobe cancellation (GSC) for a fully connected WASN was presented in [14]. A different strategy was constructed in [13], where a distributed MVDR beamformer for WASNs was proposed based on a message passing algorithm [18].

These distributed speech enhancement algorithms are assumed to operate in networks with a special topology. For example, the algorithms in [16][17][11][14] are confined to operate in fully connected networks or networks with a tree topology. The algorithm in [13] requires the network topology to be consistent with the noise correlation matrix, where two nodes are neighbors if their noise cross correlation is not equal to zero. However, WASNs may be dynamic as nodes may join or leave the network due to a defect or an empty battery, resulting in unpredictable changes in network size and topology, a distributed beamformer which is robust to changes in network topology and unreliable communication environments is important and valuable in particular for large WASNs.

In this work, we investigate the use of randomized gossip [19] in distributed beamforming for speech enhancement in a randomly connected network. Without any specialized network routing constraint, the randomized gossip algorithm [19] is an attractive algorithm to solve consensus problems, such as computing the average, the minimum or the maximum in a distributed manner. The consensus problems are solved by performing only local information exchanges, and thus providing robust solutions for large scale WASNs with dynamic topology. The randomized gossip algorithm is an iterative processing scheme and uses simple computations. The original randomized gossip algorithm in [19] was presented in two communication schemes: asynchronous and synchronous. In the asynchronous randomized gossip, at every iteration, one randomly selected node wakes up, after which it communicates with one of its neighbors chosen at random. In this scheme, only one pair of neighboring nodes in each timeslot can update its estimates. The distributed synchronous communication schemes were presented for a bounded degree network and an unbounded regular network, respectively. In the distributed synchronous gossip algorithm, multiple communicating node pairs can estimate the signal statistics simultaneously. Thus, it can potentially increase the convergence rate in both the bounded degree and the unbounded regular networks. As a regular network is a special case of a bounded degree network, it could be expected that the communication scheme for a regular network is a special case of the communication scheme of a bounded degree network. However, this is not the case. Therefore, besides the distributed beamformer, we present a generalization and an improvement of the original distributed synchronous averaging (ODSA) algorithm from [19]. We first introduce a generalized and improved synchronous communication framework for any randomly connected network with faster convergence speed, and refer to this algorithm as the improved general synchronous averaging (IGDSA) algorithm. Then, we present a beamformer for distributed estimation of a certain target signal in noise using the IGDSA algorithm. We will show how the theory can be used to compute a distributed delay and sum beamformer (DDSB), i.e., a beamformer where the noise is assumed to be spatially uncorrelated across microphones. These assumptions are validated for diffuse noise fields and/or when the distance between

microphones is sufficiently large. In order to take into account the correlation of the noise across microphones, the presented theory can be combined with the method in [13] to compute the inverse of a noise correlation matrix in a distributed fashion. This would allow to compute a full MVDR beamformer in distributed fashion. However, as we like to focus on investigating the use of randomized gossip for distributed beamforming for speech enhancement, we will mainly focus on the DDSB, but show some results to demonstrate that the presented theory can also be used to compute a distributed MVDR.

Furthermore, in order to focus on the theory and analysis of the distributed beamformer algorithm, we assume here that the steering vector from the speech source to each of the microphones is known. The steering vector in the distributed setup can be obtained by estimating the location of the target source and the microphones. For an overview on sensor network self-localization and source localization algorithms see [20] and [21], respectively. In contrast to the traditional centralized delay and sum beamformer (CDSB), the DDSB algorithm operates in a randomly connected network and aims to estimate the desired signal in a distributed way via gossip processing. The proposed DDSB algorithm is based on an iterative scheme and asymptotically converges to the optimal estimation of the CDSB. At every iteration, each node in the DDSB algorithm estimates the desired signal by using only local information and by performing only local processing. In addition, since the DDSB algorithm needs only local communication and local computing, there are no requirements for a special network topology and there is no risk of having a single point of failure making the DDSB effective for unreliable communication environments.

Some earlier initial results on the work in this article were described in [12] and [22]. In [12], we briefly introduced the asynchronous DDSB (ADDSB) algorithm for speech enhancement via the asynchronous gossip and derived a bound for the averaging time in the case of the worst connected network. The current article provides more details on the convergence analysis and bounds for the averaging time in the best and worst connected networks. In [22] we presented a synchronous version of the DDSB based on an improved version of the ODSA algorithm for regular networks. However, since a regular graph is a strong limitation for the application of the DDSB algorithm, we now provide an improved synchronous DDSB (ISDDSB) algorithm for speech enhancement based on the proposed IGDSA algorithm which can operate in a randomly connected network. In addition, we provide a comparison of the convergence rate of the DDSB under the various presented communication schemes in terms of an analytic convergence analysis as well as using simulation experiments. The simulation results validate the theoretical results, which show that the IGDSA algorithm converges faster than the asynchronous gossip algorithm and the ODSA algorithm in a randomly connected network.

The remainder of this article is organized as follows. The problem formulation and notation are given in Section 3.2. In Section 3.4, we briefly review the asynchronous gossip algorithm and we propose the IGDSA algorithm based on the distributed synchronous gossip algorithm. Then in Section 3.5, we describe the proposed DDSB algorithm in detail. Section 3.6 discusses the conditions for the DDSB algorithm using the different communication schemes to converge to the optimal CDSB solution and

introduce the convergence rate analysis of the DDSB algorithm in the asynchronous and synchronous communication schemes. In Section 3.7, the performance of the DDSB algorithm and the convergence results are illustrated with simulations. Finally, in Section 3.8, conclusions are drawn.

3.2 Problem Formulation

Let us consider a WASN consisting of N (wirelessly) connected nodes. We assume that neighboring nodes can exchange information through a wireless link. Each node is assumed to consist of a microphone and processor. Each node i captures a noisy speech signal $y_i(n)$, which is assumed to consist of a target source degraded by additive noise, given by $y_i(n) = x_i(n) + v_i(n)$, where $x_i(n)$ and $v_i(n)$ denote the speech and noise signals, respectively, of node i at the time-sampling index n. We further assume that the speech $x_i(n)$ and noise $v_i(n)$ are statistically independent. These signals are windowed and transformed into the frequency domain by applying the short-time discrete Fourier transform (DFT) leading to

$$Y_i(k,m) = X_i(k,m) + V_i(k,m), \tag{3.1}$$

where $Y_i(k,m)$, $X_i(k,m)$ and $V_i(k,m)$ denote the noisy speech, target speech and noise DFT coefficient, respectively, at frequency-bin index k, time-frame index m and microphone i. We assume the DFT coefficients to be independent in time and frequency, which allows us omit the time and frequency indices for brevity. We define $\mathbf{Y} = [Y_1, \cdots, Y_N]^T$ as the N-channel signal in which all Y_i are stacked, and where $(\cdot)^T$ indicates a matrix transposition. Similarly, we define $\mathbf{X}$ and $\mathbf{V}$ as the vectors containing the speech and noise DFT coefficients of the N nodes, respectively. We consider a single target speech source in the network. The acoustic path from the desired source to the N nodes is modeled by the steering vector $\mathbf{d}$ with $\mathbf{d} = [d_1, \cdots, d_N]^T$. We can thus formulate the WASN signal model for all nodes as

$$\mathbf{Y} = \mathbf{d}S + \mathbf{V}, \tag{3.2}$$

where S denotes the clean speech DFT coefficient of the target speaker. The objective is then to estimate the desired speech signal S.

3.3 Centralized Beamforming

Although it is of interest to realize the above objective using distributed processing, we will in this subsection briefly recapitulate the conventional solution of a centralized beamformer. In a centralized beamformer, each node i in the network broadcasts its noisy DFT coefficients Y_i to a central processing unit. Then, the clean speech DFT coefficient S can be estimated by applying a complex weight to the vector $\mathbf{Y}$ with noisy DFT coefficients. That is,

$$Z = \mathbf{w}^H \mathbf{Y}, \tag{3.3}$$

where Z is an estimated clean speech DFT coefficient, and $\mathbf{w}$ is a vector with filter coefficients and $(\cdot)^H$ denotes the Hermetian transposition of a matrix. As beamforming is a well-established research topic, there are many types of beamformers that can be used for this purpose. An often used beamformer for speech enhancement is the minimum variance distortionless response (MVDR) beamformer [21]. The corresponding weight vector $\mathbf{w}$ is the solution to the following optimization problem

$$\min_{\mathbf{w}} \mathbf{w}^H \mathbf{R}_{\mathbf{YY}} \mathbf{w}, \text{ subject to } \mathbf{w}^H \mathbf{d} = 1, \tag{3.4}$$

where $\mathbf{R}_{\mathbf{YY}} = E\left[\mathbf{Y}\mathbf{Y}^H\right]$ is the spectral covariance matrix of the noisy signal with the statistical expectation operator $E[\cdot]$. Assuming that the speech signal is uncorrelated with the noise, i.e., $E\left[\mathbf{X}\mathbf{V}^H\right] = E\left[\mathbf{V}\mathbf{X}^H\right] = 0$, the noisy spectral covariance matrix $\mathbf{R}_{\mathbf{YY}}$ can be written as $\mathbf{R}_{\mathbf{YY}} = \mathbf{R}_{\mathbf{XX}} + \mathbf{R}_{\mathbf{VV}}$. Then, solving the optimization problem (3.4) using the Lagrange multiplier approach [23] and the matrix inversion lemma [24], yields the solution for the MVDR weights, given by

$$\mathbf{w} = \frac{\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}}{\mathbf{d}^H \mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}}. \tag{3.5}$$

In this work, we assume that the WASN is in a diffuse noise field and/or that the distance between nodes is sufficiently large. With this assumption, the noise coefficient V_i, $\forall i$ can be argued to be approximately spatially uncorrelated with power spectral density (PSD) $\sigma_{V_i}^2$. The noise correlation matrix PSD can then be expressed as

$$\mathbf{R}_{\mathbf{VV}} = \text{diag}\left\{\sigma_{V_1}^2, \cdots, \sigma_{V_N}^2\right\}. \tag{3.6}$$

However, the work presented in this article can also be applied in the situation where this assumption is not made and $\mathbf{R}_{\mathbf{VV}}$ is not diagonal, e.g., by combining the proposed algorithm with a message passing algorithm as in [13]. To demonstrate this, we will present some additional experimental results in Section 3.7, where we show the potential of the algorithm in combination with the method in [13] for distributed matrix inversion in order to compute a distributed MVDR beamformer.

Combining the MVDR filter from (3.5) with (3.6), the optimal solution, in (3.3) can be written as

$$Z = \frac{\sum_{i=1}^{N} d_i^* \sigma_{V_i}^{-2} Y_i}{\sum_{i=1}^{N} d_i^* \sigma_{V_i}^{-2} d_i}. \tag{3.7}$$

It should be noted that this beamformer allows for different noise PSDs per microphone, while the generally used delay and sum beamformer requires the same noise PSD for all microphones. Thus compared to the standard delay and sum beamformer, the beamformer in (3.7) is more general. To compute the optimal solution of (3.7) in a centralized fashion, each node i needs to transmit its noisy DFT coefficients Y_i and steering vector d_i to the central processing unit. As an alternative we investigate in this work the use of randomized gossip [19] in order to compute the beamformer in a distributed way.

3.4 Randomized Gossip Algorithm

The randomized gossip algorithm [19] is a simple iterative algorithm for solving average consensus problems in a distributed way. Consider a randomly connected network, where the connectivity is represented with an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The vertex set $\mathcal{V} = \{1, 2, \cdots, N\}$ consists of the N nodes, and the edge set $\mathcal{E}$ denotes the communication links between every set of two nodes. $\mathcal{N}_i = \{j | \{i, j\} \in \mathcal{E}\}$ denotes the set of neighbors of node i. In the connected network $\mathcal{G}$, we assume that each node i has an initial value $g_i(0)$. The randomized gossip algorithm aims to find the average value $g_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N} g_i(0)$ of the initial values at each node i by using only local information and local processing. The active communicating node pairs in the randomized gossip algorithm are constrained to be disjoint. This constraint was referred to as the gossip constraint in [19] and guarantees that each node i only communicates with one neighboring node at each iteration. In [19], the randomized gossip algorithm was considered in an asynchronous communication scheme and a synchronous communication scheme. In the asynchronous communication scheme, only one pair of neighboring nodes can update its data per iteration. The synchronous averaging algorithms were proposed in order to obtain multiple communicating node pairs at the same time, assuming that this increases the convergence rate. The synchronous communication schemes were considered for an unbounded degree regular graph and a bounded degree graph. A regular graph, is a graph where each node has an equal number of neighbors. Obviously, this is a strong limitation for the application of distributed speech enhancement. The scheme for a bounded degree graph is more general but still has a relatively low convergence speed as it depends on the node with highest degree, i.e., the node with the maximum number of neighbors. Furthermore, the averaging procedure between two nodes is canceled when more than one active node contacts a non-active node simultaneously. This slows down the convergence speed of the algorithm unnecessarily. While the regular graph is a special case of a bounded degree graph, the synchronous communication scheme for regular graphs is not a special case of the synchronous communication scheme for bounded degree graphs. To obtain a more general framework and faster convergence rate, we present in this section a distributed synchronous averaging algorithm for a randomly connected network meant for distributed speech enhancement, based on the original distributed synchronous averaging (ODSA) algorithms [19] and we refer to this algorithm as the improved general distributed synchronous averaging (IGDSA) algorithm.

3.4.1 Asynchronous Communication

In the asynchronous gossip algorithm [19], a pair of nodes is randomly selected based on the asynchronous time model. Each node i runs a Poisson process of rate 1 independently, which is equivalent to a global clock of rate N and uniform selection of the active node. Here we denote t as the instant of the tth tick of the global clock and $g_i(t)$ as the value of node i at the end of time-slot t. In each time-slot t, when node i's clock ticks, it randomly selects one neighboring node j with probability p_{ij}. All probability elements p_{ij} are stacked in a $N \times N$ dimensional probability matrix $\mathbf{p}$, where $p_{ij} > 0$ if node i and node j are neighbors, otherwise $p_{ij} = 0$. At each iteration

t, with probability $G_{ij}^{A} = \frac{1}{N} p_{ij}$, a pair of neighboring nodes i and j in the network is randomly selected to exchange and update their current estimates as

$$g_i(t) = g_j(t) = \frac{g_i(t-1) + g_j(t-1)}{2}. \tag{3.8}$$

Except for the two active nodes, all other nodes in the network keep the estimates from the previous time-slot $t-1$. When each pair of neighboring nodes in the connected network gossips frequently enough, the estimates of each node are guaranteed to converge to the average value g_{ave}. We discuss the convergence conditions in the analysis in Section 3.6.

3.4.2 Improved Synchronous Communication

In the asynchronous communication scheme, only one pair of neighboring nodes i and j performs an update per iteration, while the other nodes keep their estimates. Therefore, the asynchronous communication scheme may converge slow in time. This problem becomes worse when the network is a large sparse network; then the optimally estimated signal $Z_i, \forall i$ can only be obtained at the cost of a large number of iterations. A reasonable solution is to increase the number of simultaneously communicating node pairs, i.e., multiple node pairs may update their estimates at each iteration, as also suggested in [19] for a regular and bounded degree network with the aforementioned limitations. As an alternative, we present the IGDSA algorithm in order to obtain a general framework for synchronized communication. The algorithm is inspired by the ODSA algorithms in [19], but is generalized to randomly connected networks and is improved for faster convergence rate. In contrast to the ODSA algorithms, the IGDSA algorithm for a regular graph is a special case of the IGDSA algorithm for a bounded degree graph. An additional drawback of the ODSA algorithms is that an inactive node j fails to have contact with any other node when more than one node, say r nodes, contact node j during the same iteration. This means that node j has a decreased probability of contacting its neighboring active node i. An improvement that overcomes this drawback is to allow the inactive node j to select randomly one of the r requesting neighboring nodes with probability $\frac{1}{r}$ if contacted by r active nodes.

Given a randomly connected network of N nodes, each node i at each iteration t is active with probability $\frac{1}{2}$ independently. An active node i randomly contacts one neighboring inactive node j with probability p_{ij} and ignores all requests from other active nodes. The corresponding probability matrix has the same definition as the probability matrix $\mathbf{p}$ in the asynchronous communication scheme. An inactive node j randomly selects a node i with probability $\frac{1}{r}$ from the r active nodes that contact it. After that, nodes i and j update their estimates according to (3.8). The probability that node pair (i, j) is selected, is given by

$$G_{ij}^{I} = \frac{1}{2} \sum_{r=1}^{b_j} \frac{1}{r} \sum_{f=1}^{\binom{b_j-1}{r-1}} \prod_{l \in u_f^r} \frac{1}{2} p_{lj} \left(\prod_{l \in \mathcal{N}_j \setminus u_f^r} (1 - \frac{1}{2} p_{lj}) \right)^{I}, \tag{3.9}$$

where b_j is the size of $\mathcal{N}_j$, i.e., the number of neighbors of node j, and u_f^r is the set of r active nodes that contact node j. Set u_f^r depends on the specific combination f taken

from the $\binom{b_j-1}{r-1}$ possible combinations when there are r active nodes contacting node j; I is an indicator function which is $I = 0$ when the set $\mathcal{N}_j \setminus u_f^r$ is empty and $I = 1$ otherwise. For a regular network, (a graph where each node has exactly b neighbors), G_{ij}^I can be simplified using the Binomial Theorem resulting in

$$G_{ij}^I = \frac{1}{2b}\left(1 - \left(1 - \frac{1}{2b}\right)^b\right). \tag{3.10}$$

At each iteration t, the probability G_{ij}^I can be computed as follows: node j is inactive with probability $\frac{1}{2}$; $r, r \in \{1, \cdots, b_j\}$, neighboring nodes of a node j become active and contact the inactive node j with probability $\prod_{l \in u_f^r} \frac{1}{2} p_{lj}$. The active node i is randomly selected by the inactive node j with probability $\frac{1}{r}$ while the $b_j - r$ remaining nodes do not contact node j with probability $\prod_{l \in \mathcal{N}_j \setminus u_f^r}(1 - \frac{1}{2}p_{lj})$. Note that besides node i, the inactive node j has $b_j - 1$ other neighbors and thus, $\binom{b_j-1}{r-1}$ is the combination of selecting $r-1$ active nodes out of $b_j - 1$ remaining neighboring nodes of node j. The IGDSA is guaranteed to converge to the average value g_{ave} if a sufficient number of iterations is used. We will give a detailed convergence rate analysis in Section 3.6.

3.5 Distributed Delay and Sum Beamformer

The algorithm proposed in this work is referred to as the distributed delay-and-sum beamformer (DDSB), since its objective is to estimate the centralized beamformer from (7) in a distributed way. Unlike the centralized beamformer where the information from all nodes is gathered at a central processing unit, the DDSB allows each node in a randomly connected network to broadcast its data to only one of its neighbors with the aim to obtain the same optimally estimated signal as in (3.7) at each node by using only local information and local processing.

In a randomly connected WASN, we assume that each node i has two initial values for a given time frame $\tilde{Y}_i(0) = d_i^* \sigma_{V_i}^{-2} Y_i$ and $\tilde{d}_i(0) = d_i^* \sigma_{V_i}^{-2} d_i$, where the noisy signal Y_i is obtained from the observation of the microphone at node i; the steering vector d_i and the noise PSD $\sigma_{V_i}^2$ have to be estimated. In order to keep the focus on the theory and analysis of the distributed beamformer algorithm, we assume here that the steering vectors are known. An estimate of d_i in the distributed setup can be obtained by estimating the location of the target source and the microphones. For an overview on sensor network self-localization and source localization algorithms see [20] and [21], respectively. To estimate the noise PSD $\sigma_{V_i}^2$, we make use of the noise PSD estimator presented in [25]. Based on the two initial values $\tilde{Y}_i(0)$ and $\tilde{d}_i(0)$, the optimal centralized beamformer from (3.7) can be obtained as

$$Z = \frac{\frac{1}{N}\sum_{i=1}^N \tilde{Y}_i(0)}{\frac{1}{N}\sum_{i=1}^N \tilde{d}_i(0)}. \tag{3.11}$$

Equation (3.11) shows that the distributed beamformer can be written as a ratio of two averages, and thus, it can be seen as an averaging consensus problem.

Let $\tilde{Y}_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N}\tilde{Y}_i(0)$ and $\tilde{d}_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N}\tilde{d}_i(0)$ denote the averages of all nodes' initial values $\tilde{Y}_i(0)$ and $\tilde{d}_i(0)$, respectively. The objective of the DDSB algorithm is then to find the average value $\tilde{Y}_{\text{ave}}$ and $\tilde{d}_{\text{ave}}$ in a distributed manner. The DDSB considered here is based on the randomized gossip algorithm and is an iterative and randomized scheme, since each pair of communicating neighboring nodes is randomly selected at each iteration. In addition, we classify the DDSB as asynchronous DDSB (ADDSB), original synchronous DDSB (OSDDSB) and improved synchronous DDSB (ISDDSB) depending on the different communication schemes of the randomized gossip algorithm. Although we focus on the DDSB, the same reasoning can be used to compute an MVDR beamformer in distributed manner. In that case, $\mathbf{h} = \mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}$ can be computed, for example, the message passing algorithm presented in [13]. Subsequently, $Z = \frac{\mathbf{h}^H\mathbf{Y}}{\mathbf{h}^H\mathbf{d}}$ can be computed in a distributed fashion using randomized gossip, similar to the DDSB.

Before describing the DDSB, we introduce some additional notation. Let $\tilde{\mathbf{Y}}(0) = [\tilde{Y}_1(0), \cdots, \tilde{Y}_N(0)]^T$ denote a stacked N-dimensional vector consisting of initial values $\tilde{Y}_i(0)$ for all nodes i, and let the N-dimensional vector $\tilde{\mathbf{d}}(0)$ denote a stacked vector of all initial values $\tilde{d}_i(0)$. Similarly, we use the stacked vector notation $\tilde{\mathbf{Y}}(t)$ and $\tilde{\mathbf{d}}(t)$ denoting vectors $\tilde{\mathbf{Y}}$ and $\tilde{\mathbf{d}}$ at iteration t, respectively. Then a general vector form of the DDSB which describes the estimate at iteration t is given by

$$\widetilde{\mathbf{Y}}(t) = \mathbf{U}(t)\widetilde{\mathbf{Y}}(t-1), \tag{3.12}$$

$$\widetilde{\mathbf{d}}(t) = \mathbf{U}(t)\widetilde{\mathbf{d}}(t-1), \tag{3.13}$$

$$\widetilde{Z}_i(t) = \frac{\widetilde{Y}_i(t)}{\widetilde{d}_i(t)}, \tag{3.14}$$

where $\widetilde{Z}_i(t)$ denotes the estimated output signal of node i at iteration t, and $\mathbf{U}(t)$ is a randomly selected $N \times N$ dimensional update matrix. The matrix $\mathbf{U}(t)$ is selected independently across time and it is computed as

$$\mathbf{U}(t) = \mathbf{I} - \frac{1}{2}\sum_{i,j\in C(t)}(e_i - e_j)(e_i - e_j)^T, \tag{3.15}$$

where $\mathbf{I}$ denotes the $N \times N$ dimensional identity matrix, $e_i = [0, \cdots, 0, 1, 0, \cdots, 0]^T$ is an N-dimensional unit vector with the ith component equal to 1, and $C(t)$ is a set of all communicating node pairs in the tth time-slot. The update matrix is a doubly stochastic matrix, which implies $\mathbf{U}(t)\mathbf{1} = \mathbf{1}$ and $\mathbf{1}^T\mathbf{U}(t) = \mathbf{1}^T$ with $\mathbf{1}$ denoting a vector of all ones. These properties are necessary for the randomized gossip algorithm to converge [19].

Given the initial vectors $\widetilde{\mathbf{Y}}(0)$ and $\widetilde{\mathbf{d}}(0)$, the DDSB algorithm is realized by the following steps:

1. Initialize the iteration index $t = 0$.

2. Select communicating neighboring nodes i and j via the chosen communication scheme.

3. Update the estimates $\widetilde{Y}_i(t)$, $\widetilde{Y}_j(t)$, $\widetilde{d}_i(t)$ and $\widetilde{d}_j(t)$ of all selected averaging node pairs $(i,\ j)$ as in (3.8). This implies that the weight matrix $\mathbf{U}(t)$ in (3.15) is updated in the general vector form of the DDSB, and thus, all nodes update their local information $\widetilde{\mathbf{Y}}(t)$ and $\widetilde{\mathbf{d}}(t)$ by using equations (3.12) and (3.13).

4. Update the DDSB output $\widetilde{Z}_i(t)$ of each node i in the network in (3.14).

5. $t \rightarrow t+1$.

6. Return to step 2 until convergence has been achieved (see Section 3.6) or after a fixed amount of iterations.

The time domain signal is then obtained by applying a windowed frame-wise inverse DFT followed by overlap-add.

3.6 Convergence Analysis

Given that the network is connected, the iterative randomized gossip algorithm guarantees that all nodes' estimates converge to the optimal average value when the update matrix in each time-slot is a doubly stochastic matrix [19]. Since the update matrix $\mathbf{U}(t)$ of the DDSB is symmetric and doubly stochastic in each iteration, the convergence of $\lim_{t\rightarrow\infty} \tilde{\mathbf{Y}}(t)$ to $\tilde{Y}_{\text{ave}}\mathbf{1}$ and $\lim_{t\rightarrow\infty} \tilde{\mathbf{d}}(t)$ to $\tilde{d}_{\text{ave}}\mathbf{1}$ is guaranteed for any $\tilde{\mathbf{Y}}(0)$ and $\tilde{\mathbf{d}}(0)$. The convergence of the parameters $\tilde{\mathbf{Y}}(t)$ and $\tilde{\mathbf{d}}(t)$ guarantees that the output $\tilde{Z}_i$ of the DDSB converges to the optimal centralized solution Z if $\tilde{d}_{\text{ave}} \neq 0$.

To analyze the convergence rate of the presented algorithms, we use the convergence error defined as

$$CE = \frac{\left\| \tilde{\mathbf{Y}}(t) - \tilde{Y}_{\text{ave}}\mathbf{1} \right\|}{\left\| \tilde{\mathbf{Y}}(0) \right\|}. \tag{3.16}$$

With the convergence error CE, the convergence rate of the algorithm can in analogy with [19] be defined as the first time-slot where the convergence error is smaller than a desired error ϵ with high probability $1-\epsilon$. This time is referred to as the ϵ-averaging time and is given by

$$T_{\text{ave}}(\epsilon) = \sup_{\tilde{\mathbf{Y}}(0)} \inf_{t=0,1,\cdots} \left\{ P\left(CE \geq \epsilon\right) \leq \epsilon \right\}. \tag{3.17}$$

The averaging time $T_{\text{ave}}(\epsilon)$ can be shown to be bounded by the second largest eigenvalue of the expected value of the update matrix, $E[\mathbf{U}]$, as [19]

$$\frac{0.5\log\epsilon^{-1}}{\log\lambda_2\left(E\left[\mathbf{U}\right]\right)^{-1}} \leq T_{\text{ave}}(\epsilon, E\left[\mathbf{U}\right]) \leq \frac{3\log\epsilon^{-1}}{\log\lambda_2\left(E\left[\mathbf{U}\right]\right)^{-1}}. \tag{3.18}$$

As a consequence, the convergence rate of the DDSB depends on the second largest eigenvalue of $E[\mathbf{U}]$; the smaller the magnitude of $\lambda_2(E[\mathbf{U}])$, the faster the convergence. The general definition of the expected value of the update matrix $E[\mathbf{U}]$ is given as follows:

1. The entry in the i-th row and the j-th column of the update matrix is $\mathbf{U}_{ij} = \frac{1}{2}$ for $i \neq j$, with probability $G_{ij} + G_{ji}$; otherwise, $\mathbf{U}_{ij} = 0$. Thus, the entry of the expected value $E[\mathbf{U}]_{ij}$ is

$$E[\mathbf{U}]_{ij} = \frac{1}{2}(G_{ij} + G_{ji}). \tag{3.19}$$

2. When $i = j$, the entry of the update matrix is $\mathbf{U}_{ii} = \frac{1}{2}$ with probability $\sum_{j=1}^{N}(G_{ij} + G_{ji}) - 2G_{ii}$; otherwise $\mathbf{U}_{ii} = 1$. Then the expected value is

$$E[\mathbf{U}]_{ii} = 1 - \frac{1}{2}\sum_{j=1}^{N}(G_{ij} + G_{ji}) + G_{ii}, \tag{3.20}$$

where G_{ij} is the probability that nodes i and j are selected to update their estimates. Note that in this work we assume that there is no self-communication in the network, i.e., $\text{tr}(\mathbf{G}) = 0$, as this will not lead to changes in the data. Similarly, we denote the expected value of the ADDSB and the ISDDSB as $E_A[\mathbf{U}]$ and $E_I[\mathbf{U}]$, respectively. From the above definitions of the expected values, it follows that $E[\mathbf{U}]$ can be written in a general vector form as

$$E\left[\mathbf{U}\right] = \mathbf{I} - \frac{\mathbf{m}}{2} + \frac{\mathbf{G} + \mathbf{G}^T}{2}, \tag{3.21}$$

where $\mathbf{m} = \text{diag}\left([m_1, \cdots, m_N]\right)$ is a diagonal matrix with $m_i = \sum_{j=1}^{N}\left[G_{ij} + G_{ji}\right]$. As we discussed two different communication schemes in Section 3.4, matrix $\mathbf{G}$ has two different possible expressions (G_{ij}^A and G_{ij}^I) depending on the communication scheme.

In this section, based on the bound given in (3.18), and in combination with the expected values $E_A[\mathbf{U}]$ and $E_I[\mathbf{U}]$ of the DDSB using the ADDSB and ISDDSB, respectively, we first give a convergence analysis of the ADDSB, and then we present convergence rate comparisons between the different DDSB algorithms.

3.6.1 Convergence Analysis of Asynchronous Gossip

The upper bound given in (3.18) is the minimum averaging time of the algorithm for a given connected network to guarantee $P(CE \geq \epsilon) \leq \epsilon$. In practice, the exact network topology is unknown. To be more specific about the averaging time of the ADDSB algorithm expressed in terms of sensors in the network, we now derive bounds under certain conditions for the fastest and the slowest asynchronous gossip algorithms for a network of a given size.

As defined in Section 3.5, the probability matrix $\mathbf{p}$ is a stochastic matrix. In the following derivations we will assume for ease of analysis that the matrix $\mathbf{p}$ is doubly stochastic. In that case, from (3.21) in combination with G_{ij}^A, it follows that the expected value of the ADDSB $E_A[\mathbf{U}]$ is given by (see also [19])

$$E_A[\mathbf{U}] = (1 - \frac{1}{N})\mathbf{I} + \frac{1}{N}\mathbf{r}, \tag{3.22}$$

with $\mathbf{r} = (\mathbf{p} + \mathbf{p}^T)/2$. From the bound given in (3.18), and in combination with (3.22), we see that $\lambda_2(E_A[\mathbf{U}])$, and thus, $T_{\text{ave}}(\epsilon, E_A[\mathbf{U}])$, depend on the matrix $\mathbf{p}$ and hence, on the underlying network topology.

Given the network size, the connectivity of a randomly connected network will be between the connectivity of the worst connected network and the best connected network. We will first derive an upper bound for the averaging time for the best connected network, and then an upper bound for the averaging time for the worst connected network, under the constraint that $\mathbf{p}$ is doubly stochastic.

Best Connected Networks

Since the expected value $E_A[\mathbf{U}]$ is a symmetric positive semi-definite doubly stochastic matrix [19], the eigenvalues of $E_A[\mathbf{U}]$ are non-negative and equal to or smaller than 1 in magnitude. We denote them as

$$\lambda_1(E_A[\mathbf{U}]) = 1 \geq \lambda_2(E_A[\mathbf{U}]) \geq \cdots \geq \lambda_N(E_A[\mathbf{U}]) \geq 0. \tag{3.23}$$

By the definition of the probability matrix $\mathbf{p}$, we have $\text{tr}(\mathbf{p}) = 0$, which means that $\sum_{i=1}^{N} \lambda_i(\mathbf{p}) = 0$. Combining this with (3.22), it then follows that

$$\text{tr}(E_A[\mathbf{U}]) = 1 + \sum_{i=2}^{N} \lambda_i(E_A[\mathbf{U}]) = N - 1. \tag{3.24}$$

From (3.23) in combination with (3.24), it follows that $\lambda_2(E_A[\mathbf{U}])$ is at its minimal when all $\lambda_i(E_A[\mathbf{U}])$ for $i \in 2, \cdots, N$ are equal. From (3.24), it then follows that $\text{tr}(E_A[\mathbf{U}]) = 1 + \lambda_2(E_A[\mathbf{U}])(N-1) = N - 1$, and thus, the smallest second largest eigenvalue is $\lambda_2(E_A[\mathbf{U}]) = 1 - \frac{1}{N-1}$ and the corresponding second largest eigenvalue of $\mathbf{r}$ is $\lambda_2(\mathbf{r}) = -\frac{1}{N-1}$.

An example of a $\mathbf{p}$-matrix with such an eigenvalue distribution is the matrix given by

$$\mathbf{p} = \frac{1}{N-1}(\mathbf{1}\mathbf{1}^T - \mathbf{I}). \tag{3.25}$$

This is intuitively satisfying, as this probability matrix $\mathbf{p}$ is the $\mathbf{p}$-matrix corresponding to a fully connected network where the probability that a node i communicates with any other neighboring node is uniformly distributed.

Altogether, the network that converges fastest when using the asynchronous gossip algorithm has a second eigenvalue $\lambda_{2,\text{FA}}(E_A[\mathbf{U}]) = 1 - \frac{1}{N-1}$. For this $\lambda_{2,\text{FA}}(E_A[\mathbf{U}])$, we get the upper bound of the N-size network as

$$T_{\text{ave,FA}}(\epsilon, N) \leq \frac{3 \log \epsilon^{-1}}{\log\left(1 - \frac{1}{N-1}\right)^{-1}}. \tag{3.26}$$

Using the Taylor series expansion $\log\left(1 - \frac{1}{N-1}\right)^{-1} = \sum_{n=1}^{\infty} \frac{\left(\frac{1}{N-1}\right)^n}{n} \geq \frac{1}{N-1}$, the upper bound of the averaging time $T_{\text{ave,FA}}(\epsilon, N)$ can be written in terms of the number of nodes N as

$$T_{\text{ave,FA}}(\epsilon, N) \leq 3(N-1) \log \epsilon^{-1}. \tag{3.27}$$

In summary, the upper convergence bound grows less than linear with the number of microphones. Furthermore, it can be shown that a network with a corresponding eigenvalue distribution is given by a fully connected network with uniform probabilities on the graph.

Worst Connected Networks

On the other hand, an example of a worst-possible connected network is given by a set of sensors that are connected as a string. In this section we assume that there is no self-loop in the network and the probability matrix $\mathbf{p}$ is a doubly stochastic matrix. Therefore, the string should form a closed circle (ring), where the probability that a node connects to the next (clockwise) node is denoted by q and the probability that it connects to the previous (anti-clockwise) node is $1-q$. This leads to the following probability matrix,

$$\mathbf{p} = \begin{bmatrix} 0 & q & & & & 1-q \\ 1-q & 0 & q & & 0 & \\ & 1-q & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & \ddots & \ddots & q \\ q & & & & 1-q & 0 \end{bmatrix}. \tag{3.28}$$

For this doubly stochastic matrix $\mathbf{p}$, matrix $\mathbf{r}$ in (3.22) is also doubly stochastic with real eigenvalues and is given by

$$\mathbf{r} = \begin{bmatrix} 0 & 0.5 & & & & 0.5 \\ 0.5 & 0 & 0.5 & & 0 & \\ & 0.5 & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & \ddots & \ddots & 0.5 \\ 0.5 & & & & 0.5 & 0 \end{bmatrix}. \tag{3.29}$$

This $\mathbf{r}$-matrix is a special case of a Toeplitz matrix and is known as a Gear-matrix [26]. More specifically, it is a Gear-matrix scaled by a factor 0.5. The eigenvalues of a scaled Gear-matrix have a special form and are given by [26] $\lambda_i = 2\beta\cos(2\pi n/N)$, with $n \in \{0, ..., N-1\}$, and $\beta = 0.5$. Since $(1-\frac{1}{N})\mathbf{I}$ is an identity matrix with eigenvalues $\lambda_i = 1-\frac{1}{N}$, $\forall i$, the second largest eigenvalue of $E_A[\mathbf{U}]$ is given by $\lambda_{2,\mathrm{WA}}(E_A[\mathbf{U}]) = 1-\frac{1}{N}+\frac{1}{N}\cos(2\pi/N)$, where the subscript WA indicates the second eigenvalue of the worst converging network when the asynchronous gossip is used. Using (3.18), this leads to the following upper bound of the averaging time $T_{\mathrm{ave,WA}}(\epsilon, N)$

$$T_{\mathrm{ave,WA}}(\epsilon, N) \leq \frac{3\log\epsilon^{-1}}{-\log\left(1-\frac{1}{N}(1-\cos(2\pi/N))\right)}. \tag{3.30}$$

Using the Taylor series expansion $\log(1-x) = -\sum_{k=1}^{\infty} \frac{x^k}{k}$ for $-1 \leq x < 1$ and $\cos x = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k}}{(2k)!}$ [27] we can write the following worst case upper bound for the averaging time in terms of N

$$T_{\text{ave,WA}}(\epsilon, N) \leq \frac{3N^3 \log \epsilon^{-1}}{(2\pi)^2/2 + \sum_{k=2}^{\infty}(-1)^{k+1} N^2 \left(\frac{2\pi}{N}\right)^{2k}/(2k)!}. \tag{3.31}$$

The averaging time in the worst connected network, grows thus with the order $\mathcal{O}(N^3)$, while the averaging time for the best connected network grows with the order $\mathcal{O}(N)$. However, in many practical applications, the network graph will certainly be better connected than the worst case scenario, but worse connected than the best connected network, as we will be show in Section 3.7.

3.6.2 Convergence Rate Comparisons

The synchronous communication schemes are proposed to converge faster than the asynchronous gossip algorithm since they allow multiple node pairs to update simultaneously. To investigate this, we compare the convergence rate of the asynchronous gossip algorithm with the ODSA algorithm in [19] and the IGDSA algorithm. The convergence analysis will be made for a regular network.

In a b-regular graph, where each node has exactly b neighbors, we define the probability matrix $\mathbf{p}$ as $p_{ij} = 1/b$ if node i is connected with node j and $p_{ij} = 0$ otherwise. Combining this probability matrix $\mathbf{p}$ with G_{ij}^A, the simplification of G_{ij}^I for regular graphs given in (3.10) and with (3.21), the expected values $E_{\text{RA}}[\mathbf{U}]$ and $E_{\text{RI}}[\mathbf{U}]$ in a b-regular graph are given by

$$E_{\text{RA}}[\mathbf{U}] = (1 - \frac{1}{N})\mathbf{I} + \frac{\mathbf{p}}{N}, \tag{3.32}$$

$$E_{\text{RI}}[\mathbf{U}] = (1 - \hat{b}_I)\mathbf{I} + \hat{b}_I \mathbf{p}, \tag{3.33}$$

where the subscripts RA and RI indicate that these are the expected values of the asynchronous gossip and IGDSA algorithm, respectively, and $\hat{b}_I = \frac{1}{2}\left(1 - \left(1 - \frac{1}{2b}\right)^b\right)$. The expected value of the ODSA algorithm can be shown to be [19]

$$E_{\text{RO}}[\mathbf{U}] = (1 - \hat{b})\mathbf{I} + \hat{b}\mathbf{p}, \tag{3.34}$$

with $\hat{b} = \frac{1}{4}(1 - \frac{1}{2b})^{b-1}$. The number of neighboring nodes is then given by the range $2 \leq b \leq N-1$ and we assume that $N > 2$. Since both $\hat{b}_I$ and $\hat{b}$ are monotonically decreasing functions as a function of b, they can be bounded as

$$\frac{1}{4}(1 - \frac{1}{2(N-1)})^{N-2} \leq \hat{b} \leq \frac{3}{16}, \tag{3.35}$$

and

$$\frac{1}{2}\left(1 - \left(1 - \frac{1}{2(N-1)}\right)^{N-1}\right) \leq \hat{b}_I \leq \frac{7}{32}. \tag{3.36}$$

To compare the convergence rate of the asynchronous gossip algorithm with the ODSA algorithm in a b-regular graph, their second largest eigenvalues can be compared as

$$\lambda_{2,\mathrm{RO}}(E_{\mathrm{RO}}\left[\mathbf{U}\right]) - \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right]) = \frac{1-N\hat{b}}{N}(1-\lambda_2(\mathbf{p})). \tag{3.37}$$

where the subscript RO indicates that this is for a regular graph and the ODSA algorithm.

From (3.35), it follows that the upper bound of $1-N\hat{b}$ is monotonically decreasing as a function of N. Then, using the fact that $-1 \leq \lambda_2(\mathbf{p}) < 1$, we have that

$$\lambda_{2,\mathrm{RO}}(E_{\mathrm{RO}}\left[\mathbf{U}\right]) > \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right])$$

for $N \leq 5$ and

$$\lambda_{2,\mathrm{RO}}(E_{\mathrm{RO}}\left[\mathbf{U}\right]) < \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right])$$

for $N \geq 7$, which indicates that the ODSA algorithm converges faster than the asynchronous gossip algorithm with high probability if $N \geq 7$ and it converges slower with high probability if $N \leq 5$. A similar eigenvalue comparison can be given between the IGDSA algorithm and the asynchronous gossip algorithm as

$$\lambda_{2,\mathrm{RI}}(E_{\mathrm{RI}}\left[\mathbf{U}\right]) - \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right]) = \frac{1-N\hat{b}_I}{N}(1-\lambda_2(\mathbf{p})). \tag{3.38}$$

From (3.38) and the bound given in (3.36), in combination with $-1 \leq \lambda_2(\mathbf{p}) < 1$ and the fact that the upper bound of $1-N\hat{b}_I$ is monotonically decreasing as a function of N, it follows that $\lambda_{2,\mathrm{RI}}(E_{\mathrm{RI}}\left[\mathbf{U}\right]) > \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right])$ for $N \leq 4$ and $\lambda_{2,\mathrm{RI}}(E_{\mathrm{RI}}\left[\mathbf{U}\right]) < \lambda_{2,\mathrm{RA}}(E_{\mathrm{RA}}\left[\mathbf{U}\right])$ for $N \geq 5$. Thus, the IGDSA algorithm converges faster than the asynchronous gossip algorithm with high probability if $N \geq 5$, while the IGDSA algorithm converges slower than the asynchronous gossip algorithm when there are less than 5 nodes in the regular network.

The convergence rate comparison between the IGDSA algorithm and the ODSA algorithm in a b-regular graph is given by

$$\lambda_{2,\mathrm{RI}}(E_{\mathrm{RI}}\left[\mathbf{U}\right]) - \lambda_{2,\mathrm{RO}}(E_{\mathrm{RO}}\left[\mathbf{U}\right]) = (\hat{b}-\hat{b}^I)(1-\lambda_2(\mathbf{p})). \tag{3.39}$$

Similarly, from (3.39) and the fact that $-1 \leq \lambda_2(\mathbf{p}) < 1$, we have $\lambda_{2,\mathrm{RI}}(E_{\mathrm{RI}}\left[\mathbf{U}\right]) - \lambda_{2,\mathrm{RO}}(E_{\mathrm{RO}}\left[\mathbf{U}\right]) \leq 0$ for all $N > 2$. This implies that the IGDSA algorithm converges faster than the ODSA algorithm with high probability.

The above convergence rate comparisons show that the synchronous communication schemes converge faster than the asynchronous communication scheme if there are enough nodes in a regular network. In [19], the authors also proposed a distributed synchronous averaging algorithm for more general graphs, i.e., bounded degree graphs. Although it is interesting to directly compare the convergence rate of the presented IGDSA algorithm with the ODSA algorithm in a randomly (non-regular) connected network, it is not straightforward to do this using analytic expressions, due to the general nature of the IGDSA algorithm. Therefore, in order to compare the convergence behavior of the two algorithms in a randomly connected network, we will use simulations as discussed in Section 3.7.

3.7 Simulations

In this section, we illustrate the performance of all the presented algorithms via a simulated WASN. We first provide simulation results to demonstrate the accuracy of the convergence analysis of the distributed averaging algorithms in Section 3.6 using synthetic data. Then, we will consider speech data to evaluate the performance of the DDSB algorithm using the different communication schemes.

3.7.1 Synthetic Data

In this subsection, we perform simulations using synthetic data in which each node i in the network has the initial value V_i, and V_i, $\forall i$ are independent and identically distributed Gaussian variables. We first consider a randomly generated WASN, to compare the convergence error CE with the bounds for the fastest and slowest averaging time of the asynchronous gossip algorithm. Then, we compare the convergence rate of the asynchronous gossip algorithm with the proposed IGDSA algorithm and the ODSA algorithm from [19] for regular networks. Finally, we give a comparison of the convergence behavior between the IGDSA algorithm, the ODSA algorithm, and the asynchronous gossip algorithm for a randomly connected network.

Worst and Best Case Bounds for A WASN of A Given Size

To illustrate that the derived bounds for the worst and the best case averaging time of the randomized gossip algorithm for a WASN of a given size guarantee a desired convergence error ϵ with high probability $1-\epsilon$, we simulate a WASN where 20 nodes are randomly connected with 60 edges. We repeat the simulation 20 times and use different initial values at all nodes. To compare how different the CE is from the desired convergence error for $\epsilon = 0.01$, we evaluate the CE for the asynchronous gossip algorithm using different fixed numbers of iterations. In the asynchronous gossip algorithm, we first use $T_{\text{ave,PA}}$ which is based on the upper bound in (3.18) combined with the optimal $\mathbf{p}$-matrix from [19]. Then we compare this to the upper bound that would be obtained for best connected network $T_{\text{ave,FA}}$ in (3.26) and the upper bound that would be obtained for the worst connected network $T_{\text{ave,WA}}$ in (3.30).

Figure 3.1 shows that both with $T_{\text{ave,WA}}$ and with the optimal $T_{\text{ave,PA}}$, the CE of the asynchronous gossip algorithm is lower than the desired CE, and that with $T_{\text{ave,FA}}$ the CE is higher than the desired CE. As expected, for a given ϵ, $T_{\text{ave,PA}}$ of the asynchronous gossip algorithm is the least number of iterations to guarantee convergence ϵ for a given connected network, and $T_{\text{ave,WA}}$ is the least number of iterations to guarantee convergence ϵ given only the network size N when using the asynchronous gossip algorithm with the assumption that matrix $\mathbf{p}$ is doubly stochastic.

Convergence Comparison in Regular Graphs

In Section 3.6, we showed a comparison of the convergence rate of the asynchronous gossip algorithm with the IGDSA algorithm for regular graphs. To demonstrate the accuracy of the convergence analysis of the distributed algorithms, we simulate four

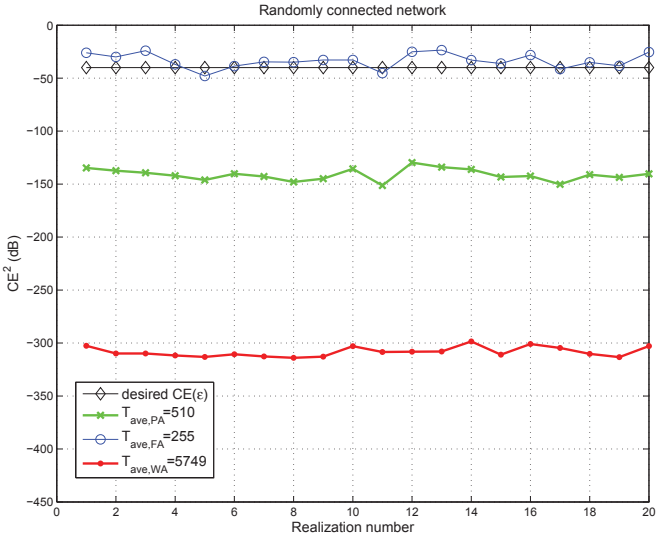


Figure 3.1: The convergence error CE across different realizations.

simple regular graphs where $N = \{4, 5, 6, 7\}$ nodes are fully connected. At each iteration t, we will use the convergence error CE given in (3.16) as a measure to assess the performance of the algorithms.

Figure 3.2(a) shows a simulation result with four fully connected nodes. The curves in Fig. 3.2(a) correspond to the three different communication schemes of the randomized gossip algorithm and show that the asynchronous scheme converges faster than the IGDSA and the ODSA algorithm. The simulation results with five, six and seven fully connected nodes are shown in Figs. 3.2(b)-3.2(d), respectively, and show that the asynchronous gossip algorithm converges slower than the IGDSA algorithm when there are more than four nodes in the network. Fig. 3.2(a) and 3.2(b) show that the asynchronous gossip algorithm converges faster than the ODSA algorithm if $N \leq 5$, and in Fig. 3.2(d) we see that the ODSA algorithm converges faster than the asynchronous communication scheme when $N \geq 7$. These results are in line with the convergence analysis in Section 3.6.2.

Convergence Comparison in Non-regular Graphs

Since it is not straightforward to perform a convergence rate comparison in a non-regular graph using analytic expressions, we show in this subsection simulation results to compare the convergence rates of the proposed IGDSA algorithm with the

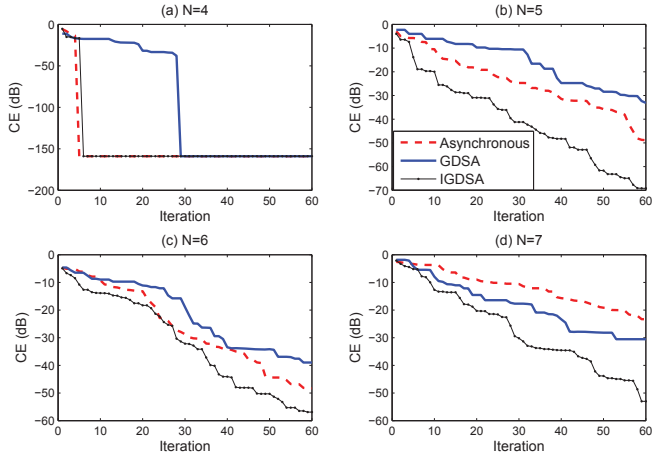


Figure 3.2: The convergence error CE versus number of iterations.

ODSA algorithm and the asynchronous gossip algorithm in non-regular networks. We simulate three different randomly connected networks where 20 nodes are randomly connected with 60, 80, and 100 edges. The probability matrix $\mathbf{p}$ in this simulation is defined as $p_{ij} = 1/b_i$ if node i and node j are neighbors, where b_i is the number of neighbors of node i; $p_{ij} = 0$ otherwise. We investigate the convergence error CE given in (3.16) versus the number of iterations.

In Figs. 3.3(a)-3.3(c), we show a results of the randomized gossip algorithm in a randomly connected network where 20 nodes are randomly connected with 60, 80 and 100 edges respectively. Not surprisingly, the IGDSA algorithm converges faster than the ODSA algorithm and the asynchronous gossip algorithm. However, note that the asynchronous gossip algorithm converges faster than the ODSA algorithm. This can be explained by the fact that in the ODSA algorithm, the probability that two neighboring nodes average is inversely proportional to the maximum degree of the network. The detailed mathematical analysis of the ODSA was provided in [19], which showed that the probability that two neighboring nodes average in the ODSA is smaller than the probability in asynchronous gossip algorithm, if the maximum degree of the network is relatively large.

Comparing Fig. 3.3(a), 3.3(b) and 3.3(c), we can also observe that by increasing the number of edges, the convergence speed of the IGDSA increases. This can be explained by the fact that increasing the number of edges will lead to more disjoint pairs of nodes that can communicate simultaneously in the IGDSA. However, the convergence speed of the ODSA has no significant change, since increasing the number of edges will increase the maximum degree of the network and partly decrease the probability that two neighboring nodes perform averaging.

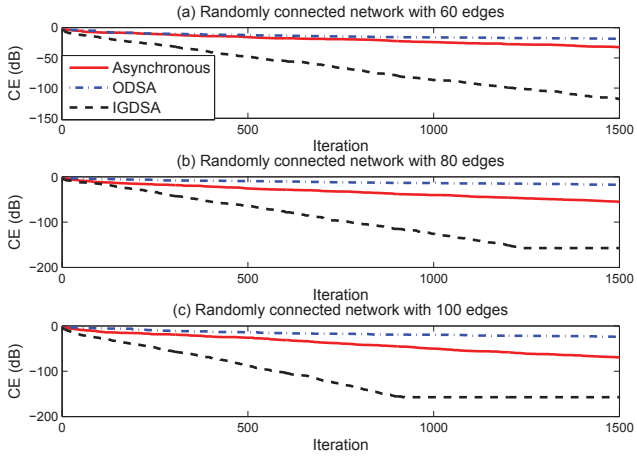


Figure 3.3: The convergence error CE versus number of iterations.

Figure 3.4 depicts the total number of required transmissions of the presented algorithms for reaching a desired convergence error $\epsilon = 0.01$ as a function of the number of edges. We simulate some randomly connected networks where 20 nodes are randomly connected with 30, 40, 50, 60, 70, 80, 90 and 100 edges, respectively. For each simulated network, we repeat the experiment 1000 times and average the required transmissions over the 1000 realizations. The simulation results show that the required transmissions for reaching the desired convergence error ϵ is decreased by increasing the number of edges of a given size network. This is consistent with the simulation results in Fig. 3.1, where a better connected network requires less transmissions to reach the desired convergence error. Notice that the difference between the total number of required transmissions of the three algorithms is very small. The reason is that all three distributed algorithms are based on the pairwise communication scheme. However, as the IGDSA allows multiple pairs of nodes to communicate simultaneously per iteration, it needs much less iterations compared to the ODSA and the asynchronous averaging as shown in Fig. 3.3.

3.7.2 Wireless Acoustic Sensor Networks

In this section, we provide experimental results obtained using speech data. First, the simulation environment and performance measures are described. Then, the performance of the DDSB algorithm in regular and non-regular networks is discussed. Lastly, the performance of the DDSB algorithm is compared with some existing distributed noise reduction algorithms.

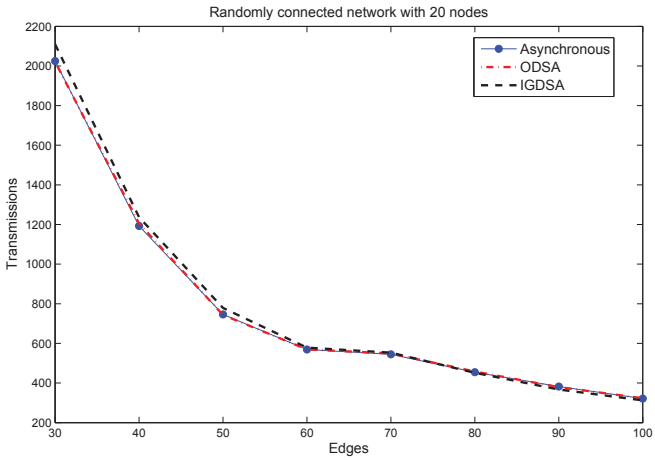


Figure 3.4: The transmissions versus number of edges.

Simulation Environment and Performance Measures

We simulate regular networks and non-regular networks with acoustic sensor nodes. In each network, we consider that wireless microphones, a speech source and a noise source are randomly distributed in a $10\text{m} \times 10\text{m}$ rectangular area. Each node gathers noisy speech signals at a sampling frequency of $f_s = 16$ kHz. We use a 30 s speech signal originating from the Timit database [28] as the clean speech source and a white Gaussian signal as the noise source. The noise PSD is estimated during noise-only periods using an ideal VAD. Assuming a free-field situation, the steering vector $\mathbf{d}$ is determined by gain and delay values as $\mathbf{d} = \left[a_1 e^{-j\omega_k \tau_1}, \cdots, a_N e^{-j\omega_k \tau_N}\right]^T$, where a_i is the damping coefficient, and τ_i denotes the delay in number of samples. In this work, we assume that the distance l_i between microphone i and the desired source is known. Then, with damping $a_i = 1/l_i$, delay $\tau_i = \frac{l_i}{c} f_s$, and the speed of sound $c = 340$m/s, the steering vector d_i of microphone i is known. All nodes process the signals in the frequency domain using frame-based processing, with a frame length of 32 ms and a 50%-overlapping Hann window.

We use the mean square-error (MSE) as a measure to assess the noise reduction performance of the presented DDSB algorithm, since we are mainly interested in the performance difference compared to the centralized noise reduction algorithms. We also assess speech quality by means of the segmental SNR, and the speech intelligibility of the enhanced signal using the short-time objective intelligibility measure (STOI)

[29]. The MSE for node i is averaged over all time frames and is defined as

$$\text{MSE}_i = \frac{1}{MK} \sum_{m=1}^{M} \sum_{k=1}^{K} \left\| \hat{Z}_i(k,m) - S(k,m) \right\|^2, \tag{3.40}$$

where K denotes the number of frequency bins, M is the number of time-frames and $\hat{Z}_i(k,m)$ and $S(k,m)$ denote the frequency domain DFT coefficient of the beamformer output and the desired speech signal, respectively, at frequency-bin index k and time-frame index m. The segmental SNR for node i is averaged over all time frames and is given by

$$\text{SNR}_i = \frac{1}{M} \sum_{m=1}^{M} 10\log_{10} \frac{\sum_{k=1}^{K} |S(k,m)|^2}{\sum_{k=1}^{K} \left| \hat{Z}_i(k,m) - S(k,m) \right|^2}. \tag{3.41}$$

The DDSB Algorithm in Regular Networks

We simulate two different regular networks with 20 microphones, a fully connected and a ring-connected network, which are the best and worst connected networks, respectively, for a doubly stochastic $\mathbf{p}$-matrix. In the simulation, the input SNR of microphone 1 in the network is set to 1 dB. We investigate the performance of the DDSB algorithm using the different communication schemes and compare the convergence rate of the ADDSB with the OSDDSB and the ISDDSB.

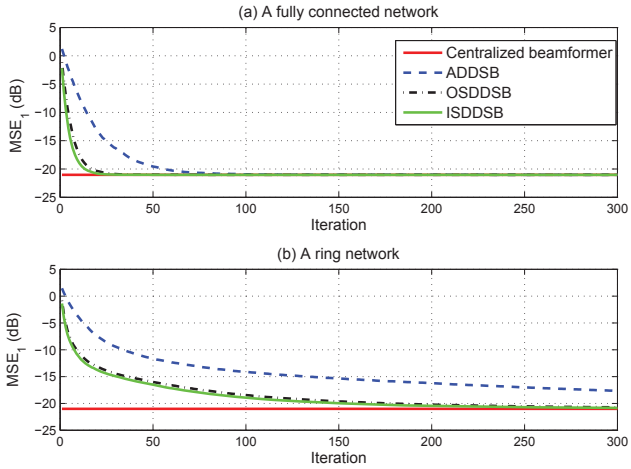


Figure 3.5: The MSE of node 1 with 1 dB input SNR versus iteration.

Figure 3.5 shows the MSE between the output of all DDSB algorithms at node 1 and the desired speech signal and the MSE between the CDSB and the desired

speech signal. It is observed that the MSE of the DDSB algorithm using the different communication schemes decreases with increasing number of iterations. It is also seen that all presented DDSB algorithms reach the same performance as the CDSB when enough iterations are used. As expected, the DDSB algorithm using synchronous communication schemes converges faster than the ADDSB algorithm in both subfigures, since there are enough nodes in the regular network. The ISDDSB has the fastest convergence, although the difference with OSDDSB in these regular networks is relatively small. The simulation results corroborate the convergence rate analysis of the DDSB algorithm in regular networks.

The DDSB Algorithm in Non-regular Networks

We now show a convergence rate comparison of all presented DDSB algorithms for a randomly connected network. We simulate a non-regular network where 20 microphones are randomly connected with 60 edges. The input SNR of microphone 1 in the network is 2 dB. In [19], the authors described a distributed method for finding an optimal probability matrix $\mathbf{p}$ in the asynchronous gossip algorithm. We use this optimal probability matrix $\mathbf{p}$ in the experiment for all presented DDSB algorithms, since the ADDSB has the fastest convergence speed using the optimal probability matrix in a randomly connected network.

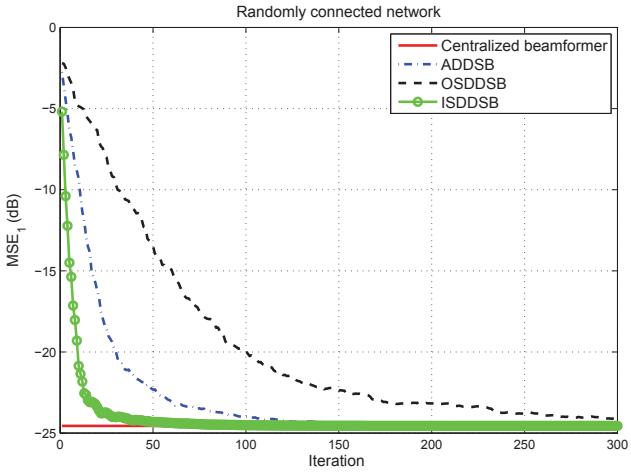


Figure 3.6: The MSE of node 1 with 2 dB input SNR versus iteration.

Figure 3.6 shows that the DDSB algorithm using the different communication schemes reaches the same performance as the CDSB when each pair of neighboring nodes communicates frequently enough. As expected, the ISDDSB converges faster

than the ADDSB and the OSDDSB algorithm. On the other hand, the ADDSB algorithm converges faster than the OSDDSB algorithm, since the maximum degree of the network is not small enough.

Comparison with Reference Methods

Here, we compare the presented framework for distributed beamforming with a method from the literature [17] in terms of performance and required number of transmissions. We simulate a randomly connected WASN where nine acoustic sensor nodes are randomly connected with 24 edges. One target speech source and ten noise sources are present in a 10×10 meter rectangular area. The ten noise sources are simulated by five independent white Gaussian noise signals and five independent babble noise signals. Each node consists of one microphone and the input SNR of microphone 1 is 1 dB. The noise PSD tracking algorithm in [25] is used to estimate the noise PSD $\sigma^2_{V_i}$.

Several existing methods are used to compare the performance of the ISDDSB. First, we consider the single-microphone Wiener filter in order to compare the performance to a single-microphone algorithm. The Wiener filter was implemented using the decision-directed approach [3] to estimate the SNR and the MMSE based noise PSD estimator from [25]. Second, we consider the DANSE algorithm [17]. Since the single-microphone Wiener filter can be applied as a post-filter on the beamformer output, we additionally include simulation results of the ISDDSB and DANSE with the single channel Wiener filter as post-processor, referred to as ISDDSB-WF and DANSE-WF, respectively. To compare the distributed beamformers with their centralized versions and evaluate any performance loss, we also use the centralized adaptive node-specific signal estimation (CANSE) algorithm, the CDSB, the CANSE-WF and the CDSB-WF, which incorporate a Wiener as post-processor. Since the DANSE algorithm is confined to perform in a network with a tree topology, we convert the randomly connected network into a network with tree topology when the DANSE algorithm is used. Furthermore, since the DANSE algorithm is time recursive and needs some initialization time, we remove the first initializing 15 s when calculating the MSE, segmental SNR and STOI.

Figures 3.7(a) and 3.7(c) show the speech quality of the distributed beamformers and their centralized version in terms of MSE and SNR, respectively. Figure 3.7(b) shows the predicted speech intelligibility performance of the beamformers output. From the perspective of the centralized algorithms, we observed that both the noise reduction and speech intelligibility performance of the CANSE algorithm and CANSE-WF are better than the CDSB and CDSB-WF. This is reasonable since the CANSE algorithm can essentially be implemented as an MVDR beamformer with single-channel Wiener post-filter, and the MVDR beamformer generally has better speech quality and intelligibility than the CDSB algorithm when the noise signals of the microphones are correlated. Figures 3.7(a) and 3.7(c) show that the noise reduction performance of the CANSE-WF and CDSB-WF is better than the CANSE and CDSB. This is consistent with the fact that the single-microphone Wiener filter can efficiently reduce noise power. However, Fig. 3.7(b) shows that the speech intelligibility of the beamformers that do not use a post-filter is better. This is because the single-channel Wiener filter leads to much speech distortion and relatively poor speech

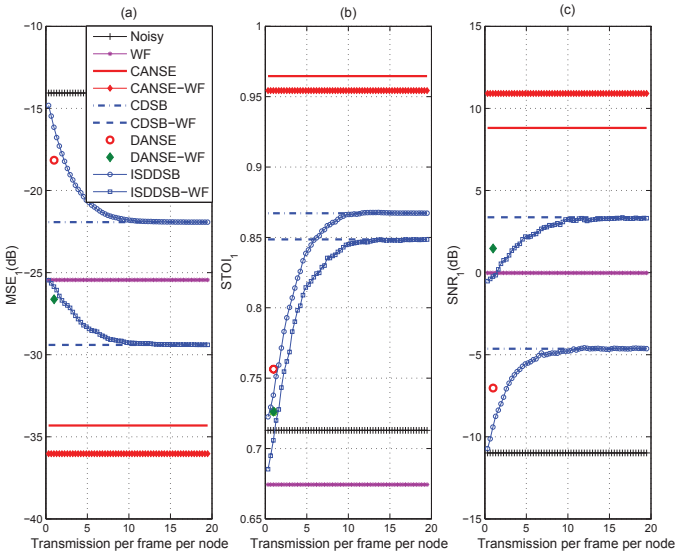


Figure 3.7: (a) The MSE of node 1 with 1 dB input SNR versus average number of transmissions per time frame per node. (b) The STOI of node 1 with 1 dB input SNR versus average number of transmissions per time frame per node. (c) The SNR of node 1 with 1 dB input SNR versus average number of transmissions per time frame per node.

intelligibility, which is also shown by comparing the STOI value of the noisy signal and the single-channel Wiener filter output. From the perspective of the distributed algorithms, it is observed that the ISDDSB and the ISDDSB-WF reach the same performance as their centralized counterparts, the CDSB and the CDSB-WF algorithms, respectively, when enough transmissions are used. However, both the speech quality and the intelligibility of the DANSE and DANSE-WF are worse than the CANSE and CANSE-WF, respectively. An interesting observation is that the performance of the DANSE and DANSE-WF are somewhat worse than the DDSB and the DDSB-WF algorithm in terms of MSE, SNR and STOI. These differences can partly be explained by the following. First, in contrast to the DDSB, DANSE assumes no knowledge about the steering vector, but estimates this implicitly using estimates of the noise, the noisy correlation matrices and using information on the on-off behavior of the desired signal. Secondly, the time-recursive DANSE algorithm does not fully converge to the CANSE algorithm, which already implies some performance loss. This might be due to the fact that 1) the DANSE algorithm performs subsequent iterations over differ-

ent signal segments and only allows one node to update its beamformer coefficients at each iteration, while other nodes only gather their neighbors' information, 2) the used observation window length is too short for the algorithm to estimate the signal statistics, and 3) low-SNR nodes might affect the estimation in the reference node.

Next, Figs. 3.7(a), 3.7(b) and 3.7(c) show the trade-off between the performance and the communication cost of all the distributed algorithms. Despite the small performance improvement of the DDSB algorithm compared to the DANSE algorithm it should be mentioned that this is at the expense of a higher communication load. The main reason for this difference is the fact that the DANSE algorithm employs a broadcast protocol and performs time-recursive updates over signal frames, while the DDSB based algorithms use a point-to-point transmission protocol per signal frame. The communication cost of the randomized gossip based distributed beamformers can be reduced via clique or cluster based randomized gossip algorithms, [30], [31].

Furthermore, the DDSB assumes that the noise field is spatially uncorrelated. This is not necessarily a problem, as shown by the experimental results in Fig. 3.7. This experiment is based on point non-stationary noise sources where clearly this assumption is not completely valid, but where validity depends on the inter-microphones distance. To further improve the performance of the distributed beamformer, the proposed algorithm can be combined with the message passing algorithm from [13] in order to incorporate noise correlation. To demonstrate this, we present a final experiment where we compare the performance of the CDSB and an MVDR, with their distributed counterparts, that are, the ISDDSB and a distributed MVDR (DMVDR) based on the message passing algorithm from [13] combined with the proposed in this article randomized gossip algorithm for distributed beamforming.

The message passing algorithm can be used to compute $\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}$ in a distributed fashion. Subsequently, the proposed randomized gossip algorithm can be used to compute $\mathbf{Y}^H\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}$ and $\mathbf{d}^H\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}$ in distributed fashion. As the noise field is assumed to be stationary in this experiment, the message passing algorithm is applied only once, in case the noise field is changing, the algorithm must be applied repeatedly for every time-frame.

Figure 3.8 depicts the noise reduction performance of the ISDDSB and the DMVDR beamformer versus the number of transmissions. We simulate a network where ten nodes are fully connected while the input SNR of microphone 1 is 1 dB. As expected, we see that indeed a gain of approximate -5 dB can be obtained by taking noise correlation into account in the DMVDR beamformer. Of course, the potential improvement by incorporating noise correlation depends on the number of noise sources and their locations. In addition, both the DMVDR and ISDDSB converge to their centralized version, after sufficient iterations. Note that the DMVDR takes some extra transmissions to estimate $\mathbf{Y}^H\mathbf{R}_{\mathbf{VV}}^{-1}\mathbf{d}$ compared to the ISDDSB algorithm.

3.8 Conclusions

In this work we introduced a distributed delay and sum beamformer (DDSB) algorithm using both asynchronous and synchronous communication schemes for decentralized estimation of the clean speech signal in a randomly connected wireless acoustic sensor

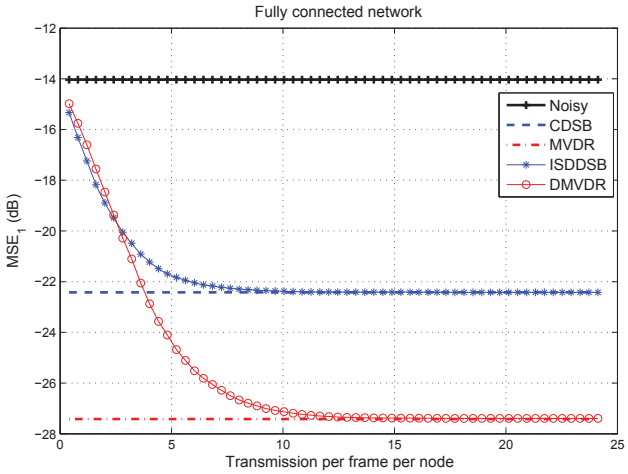


Figure 3.8: The MSE of node 1 with 1 dB input SNR versus average transmission per time frame.

network. The algorithm is based on randomized gossip. In addition, we presented an improved general distributed synchronous averaging (IGDSA) algorithm that can be applied to any connected network. The DDSB algorithm using the different communication schemes converges asymptotically to the centralized beamformer. We described worst and best case convergence bounds for the asynchronous DDSB algorithm for a network of a given size and we compared analytically the convergence rate of the DDSB algorithm using the proposed IGDSA with the asynchronous DDSB (ADDSB) algorithm and the original synchronous DDSB (OSDDSB) algorithm in an unbounded regular network. The simulation results demonstrated that the proposed algorithm for simultaneous updating increases the convergence rate of the DDSB when there is a sufficient amount of nodes in the regular network. Furthermore, simulation results with non-regular networks showed a large gain in convergence speed for DDSB using the proposed IGDSA compared to the DDSB using the existing communication algorithm for non-regular networks.

Experiments on the comparisons between the proposed algorithm and several distributed speech enhancement reference algorithms from literature indicated the trade-off between the speech enhancement performance and the communication cost of the distributed algorithms. Specifically, with the advantage of not having a topology constraint, the proposed algorithm has better performance than the referenced distributed adaptive node-specific signal estimation (DANSE) algorithm at the expense of a higher communication cost. To further reduce the communication costs, use can be made of clique and cluster based distributed beamforming. This is studied in [30],

where the communication cost of the DDSB is further decreased, by investigating the use of cliques and clusters for the randomized gossip algorithm in a randomly connected network. In contrast to the DANSE algorithm where the steering vector is estimated implicitly, the proposed algorithms make use of prior knowledge on the steering vector. Ongoing research investigates how these steering vectors can be estimated in a distributed way and how correlated noise fields and reverberation can be taken into account explicitly. Finally, to bring distributed noise reduction algorithms to practice, practical aspects such as clock synchronization of the different sensors in the WASN has to be taken into account.

References

[1] J. Jensen and R. C. Hendriks. Spectral magnitude minimum mean-square error estimation using binary and continuous gain functions. *IEEE Trans. Audio, Speech, Lang. Process.*, 20(1):92–102, Jan. 2012.

[2] Philipos C. Loizou. *Speech Enhancement - Theory and Practice*. CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, 2007.

[3] Y. Ephraim and D. Malah. Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Trans. Audio, Speech, Lang. Process.*, 32(6):1109–1121, Dec. 1984.

[4] T. Lotter and P. Vary. Speech enhancement by MAP spectral amplitude estimation using a super-Gaussian speech model. *EURASIP Journal on Applied Signal Processing*, 2005(7):1110–1126, Jan. 2005.

[5] S. Doclo and M. Moonen. GSVD-based optimal filtering for single and multimicrophone speech enhancement. *IEEE Trans. Signal Process.*, 50(9):2230–2244, Sep. 2002.

[6] S. Gannot, D. Burshtein, and E. Weinstein. Signal enhancement using beamforming and nonstationarity with applications to speech. *IEEE Trans. Signal Process.*, 49(8):1614–1626, Aug. 2001.

[7] Y. Jia, Y. Luo, Y. Lin, and I. Kozintsev. Distributed microphone arrays for digital home and office. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 1065–1068, May. 2006.

[8] A. Betrand. Applications and trends in wireless acoustic sensor networks: a signal processing perspective. In *Proc. IEEE Symposium on Communications and Vehicular Technology*, pages 1–6, Ghent, Nov. 2011.

[9] I. Himawan, I. Mccowan, and S Sridharan. Clustered blind beamforming from ad-hoc microphone arrays. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(4):661–676, Jun. 2010.

[10] S. Doclo, M. Moonen, T. Van den Bogaert, and J. Wouters. Reduced-bandwidth and distributed MWF-based noise reduction algorithms for binaural hearing aids. *IEEE Trans. Audio, Speech, Lang. Process.*, 17(1):38–51, Jan. 2009.

[11] A. Bertrand and M. Moonen. Distributed node-specific LCMV beamforming in wireless sensor networks. *IEEE Trans. Signal Process.*, 60(1):233–246, Jan. 2012.

[12] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer for speech enhancement in wireless sensor networks via randomized gossip. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 4037–4040, 2012.

[13] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng, and W. B. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[14] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[15] S. Markovich-Golan, S. Gannot, and I. Cohen. A reduced bandwidth binaural MVDR beamformer. In *Int. Workshop on Acoustic Echo and Noise Control*, Israel, Aug. 2010.

[16] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[17] A. Bertrand and M. Moonen. Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology. *IEEE Trans. Signal Process.*, 59(5):2196–2210, May. 2011.

[18] G. Zhang and R. Heusdens. Linear coordinate-descent message-passing for quadratic optimization. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 2005–2008, 2012.

[19] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[20] S. Haykin and K. J. R. Liu. *Handbook on array processing and sensor networks*. Wiley Online Library, 2009.

[21] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[22] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer in regular networks based on synchronous randomized gossip. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[23] D. Brandwood. A complex gradient operator and its application in adaptive array theory. *Proc. IEEE*, 130(1):11 – 16, Feb. 1983.

[24] H. L. Van Trees. *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing*. John Wiley and Sons, New York, 2002.

[25] R. C. Hendriks, R. Heusdens, and J. Jensen. MMSE based noise PSD tracking with low complexity. In *IEEE Int. Conf. Acoust, Speech, Signal Processing*, pages 4266–4269, 2010.

[26] C. W. Gear. A simple set of test matrices for eigenvalue programs. *Mathematics of Computation*, 23(105):pp. 119–125, 1969.

[27] I. Gradshteyn and I. Ryzhik. *Table of Integrals, Series and Products*. New York: Academic, 6th ed. edition, 2000.

[28] J. S. Garofolo. DARPA TIMIT acoustic-phonetic speech database. *National Institute of Standards and Technology (NIST)*, 1988.

[29] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen. An algorithm for intelligibility prediction of time-frequency weighted noisy speech. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(7):2125–2136, Sep. 2011.

[30] Y. Zeng, R. C. Hendriks, and R. Heusdens. Clique-based distributed beamforming for speech enhancement in wireless sensor networks. In *Proc. European Signal Proc. Conf. (EUSIPCO)*, Marrakesh, Morocco, 2013.

[31] W. Li and H. Dai. Cluster-based distributed consensus. *IEEE Trans. Wireless Communications*, 8(1):28 –31, Jan. 2009.

Chapter 4

Clique-Based Distributed Beamforming for Speech Enhancement in Wireless Sensor Networks

This chapter is published as "Clique-Based Distributed Beamforming for Speech Enhancement in Wireless Sensor Networks", by Y. Zeng, R. C. Hendriks and R. Heusdens in the Proceedings of *EURASIP Europ. Signal Process. Conf. (EUSIPCO)*, Marrakesh, Morocco, September 2013.

4.1 Introduction

In applications like hearings aids and mobile telephony, beamforming algorithms for noise reduction, e.g., [1], are often used to improve quality and intelligibility of noisy speech. However, conventional centralized beamforming algorithms generally use a rather limited number of microphones at fixed locations, which limits the performance. This can be improved using wireless acoustic sensor networks (WASNs), where many low-cost microphones each with its own individual processor are distributed over the environment. For large WASNs, traditional centralized beamformers are neither robust nor scalable. In contrast to centralized beamformers, distributed beamformers only need to perform local communication and local processing, they scale well as the network grows, and exhibit robustness as there is no centralized processor. Recently, there has been a growing interest in distributed beamforming in WASNs, e.g., [2, 3, 4].

We introduced an asynchronous distributed delay-and-sum beamformer (DDSB) in [3], which is based on the asynchronous randomized gossip algorithm [5]. Without topology constraint, the DDSB converges to the optimal centralized beamformer. In [6] it was shown that this approach can also be combined with a message passing algorithm to compute a minimum variance distortionless response (MVDR) beamformer in a distributed way. However, the convergence rate of the asynchronous DDSB is relatively slow, since only one pair of neighboring nodes can update its estimates per time-slot. An obvious way to improve the convergence speed is to apply synchronous randomized gossip [5] as applied in [7]. Although this improves the convergence speed, other approaches are required to further improve the convergence speed of randomized gossip-based beamforming.

Recent improvements of the randomized gossip algorithm exploit the principle of broadcasting, e.g., [8, 9]. In [9] this is done by forming overlapping clusters of nodes, and subsequently averaging per cluster instead of per node-pair. This improves the convergence speed. A further improvement is obtained by averaging across two neighboring non-overlapping clusters as proposed in [8]. Both algorithms depend on cluster heads, which makes them sensitive to changes in network topology, in particular if a cluster head disappears from the network. In that case, the remaining nodes in the cluster become useless and require a new formation of clusters. Instead of using clusters, we propose in this article to improve the convergence speed of the randomized gossip [5] using non-overlapping cliques. The randomized gossip is then based on averaging across two neighboring non-overlapping cliques, which will lead to a large improvement of the convergence speed of randomized gossip. Moreover, as cliques are generally better connected than clusters, the presented approach will be more robust (in terms of node failures) than the cluster-based approach [8].

The presented framework is subsequently combined with the DDSB [3]. We refer to this algorithm as clique-based distributed beamformer (CbDB). Without central unit and network routing requirements, the CbDB converges to the centralized beamformer. Since the CbDB performs only local communication and local processing, there is no constraint on the number and location of microphones and no risk of having a single point failure. Moreover, we prove that the CbDB converges faster than the DDSB in [3]. The convergence analysis of the CbDB is tested in a simulated WASN, which shows that the convergence rate compared to the DDSB is significantly

improved, and that the robustness of the CbDB is improved compared to the cluster-based distributed beamformer.

4.2 Problem Formulation and Notation

We consider an WASN as a randomly connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with $\mathcal{V}$ the vertex set consisting of N acoustic sensor nodes and $\mathcal{E}$ the edge set of undirected communication links between every set of two connected nodes. We assume that each node $i \in \mathcal{V}$ has $|\mathcal{N}_i|$ neighbors with $\mathcal{N}_i$ the set of neighbors of node i. Every node captures a mix between a desired target speech source and noise sources present in the environment. Assuming that speech and noise sources are uncorrelated and additive, the signal model for each node i in the discrete Fourier transform (DFT) domain is given as

$$Y_i(k,m) = S_i(k,m) + V_i(k,m), \tag{4.1}$$

with $Y_i(k,m)$, $S_i(k,m)$ and $V_i(k,m)$ the noisy speech, target speech and noise DFT coefficient, respectively, at frequency-bin index k and time-frame index m. We assume the DFT coefficients to have zero-mean and to be independent across time and frequency, which allows us to omit the time and frequency indices for notational convenience. Further, we consider the case of a single desired speech source. The speech S_i at node i can then be written as $S_i = d_i S$ with S the speech DFT coefficient at the source location and d_i the acoustic transfer function from the speech source S to node i. In general, Y_i, S_i, V_i and d_i, $\forall i \in \mathcal{V}$ are stacked in N dimensional vectors $\mathbf{Y}$, $\mathbf{S}$, $\mathbf{V}$ and $\mathbf{d}$, respectively. A vector notation of the signal model in DFT domain is then given by $\mathbf{Y} = \mathbf{d}S + \mathbf{V}$.

The clean speech DFT coefficient S can be estimated by applying a spatial filter $\mathbf{w}$ to $\mathbf{Y}$, i.e., $Z = \mathbf{w}^H\mathbf{Y}$, with Z the estimated clean speech DFT coefficient and $(\cdot)^H$ Hermitian transposition. An often used filter is the MVDR beamformer, that is [1]

$$\mathbf{w} = \frac{\mathbf{R}_{VV}^{-1}\mathbf{d}}{\mathbf{d}^H\mathbf{R}_{VV}^{-1}\mathbf{d}}, \tag{4.2}$$

with $\mathbf{R}_{VV}$ the noise spectral covariance matrix. For simplicity, we assume that the WASN is in an uncorrelated noise field. With this assumption, V_i, $\forall i$ can be argued to be approximately spatially uncorrelated with power spectral density (PSD) $\sigma_{V_i}^2$, so that $\mathbf{R}_{VV} = \text{diag}\left\{\sigma_{V_1}^2, \cdots, \sigma_{V_N}^2\right\}$. In case this does not hold, the presented theory can still be used in combination with the message passing algorithm from [6]. The clean speech DFT coefficient can then be estimated as

$$Z = \frac{\sum_{i=1}^{N} d_i^* \sigma_{V_i}^{-2} Y_i}{\sum_{i=1}^{N} d_i^* \sigma_{V_i}^{-2} d_i}, \tag{4.3}$$

where $(\cdot)^*$ denotes conjugation. Note that this beamformer is a special case of the MVDR beamformer, but more general than the delay-and-sum beamformer [1]. More specifically, it still allows different noise PSDs per microphone. The centralized beamformer in (4.3), can be used when there is a central processor gathering the information

from all nodes in $\mathcal{G}$. However, the constraints of the communication reliability and radius of WASNs make the centralized beamformer neither robust nor scalable in a large WASN. Instead, (4.3) can be implemented in a distributed way using the DDSB [3], which is based on finding consensus across pairs of nodes in the network. To increase convergence speed, we investigate in this article the use of cliques. This requires to find the cliques in a network in a distributed way.

Finally, notice that distributed beamforming requires the clocks of all nodes in the WASN to be synchronized. As clock synchronization is well studied, see e.g., [10] and we want to focus this work, we assume all clocks to be synchronized.

4.3 Distributed Determination of Cliques

A maximal clique is a fully connected sub-graph that cannot be extended by including more nodes without ceasing to be a clique. Since each node can belong to multiple maximal cliques, the maximal cliques of $\mathcal{G}$ can be overlapping. To exploit the concept of maximal cliques for beamforming based on randomized gossip, we consider non-overlapping cliques only. Here we briefly discuss how to find a set of non-overlapping cliques in a distributed way, such that each node belongs to only one clique.

The approach consists of two steps. First, each node $i \in \mathcal{V}$ finds all its maximal cliques in a distributed way. This can be done using a slightly modified version of the first Bron-Kerbosch algorithm [11]. In this modified version, each node i runs the Bron-Kerbosch algorithm, where the set of candidate nodes that can form a clique with node i consists of the set $\mathcal{N}_i$ of neighboring nodes. For each node this results in a set of maximal cliques. Subsequently, each node should be assigned to just one clique by local communication. A more detailed description on how to make the cliques non-overlapping are given in appendix A. This can be done using only local information of the neighboring nodes. The constraint to make the cliques non-overlapping can lead to situations where a smaller clique is selected instead of a maximal clique. Given the set of non-overlapping cliques, we propose in the next section a distributed consensus algorithm via non-overlapping cliques (DCvNC).

4.4 Distributed Consensus Algorithm

The non-overlapping cliques can be used to compress the graph by representing each clique by one single node. Let C denote the number of non-overlapping cliques and K_c denote the number of nodes in clique c. Consider a connected non-overlapping clique network $\mathcal{G}_R = (\mathcal{V}_R, \mathcal{E}_R)$ consisting of a set of non-overlapping cliques $\mathcal{V}_R = \{1, \cdots, C\}$ and a set of edges $\mathcal{E}_R$, where each edge $(c, l) \in \mathcal{E}_R$ is an undirected link between two non-overlapping cliques c and l. A clique c has $|\mathcal{N}_c|$ neighboring cliques. Note that $\mathcal{G}_R$ is a compressed version of $\mathcal{G}$, since all nodes in a clique c are represented as a single node in $\mathcal{G}_R$ and multiple edges between two cliques are compressed as one edge in $\mathcal{G}_R$.

Each node i, $i = 1, \cdots, N$, in the original graph $\mathcal{G}$ has an initial value $g_i(0)$. First, each non-overlapping clique c in the corresponding compressed graph $\mathcal{G}_R$ computes

its initial value $h_c(0)$ as $h_c(0) = \sum_{i \in c} g_i(0)$. To do so, each node i in a clique c broadcasts its values $g_i(0)$ to all other nodes in the clique. The average value over all cliques' initial values is $h_{ave} = \frac{1}{C}\sum_{c=1}^{C} h_c(0)$. We assume that each node $i \in c$ has the list of all neighboring cliques $\mathcal{N}_c$ of clique c. To facilitate a convergence analysis using the Laplacian matrix, we assume, similar as in [8], that each edge in the compressed graph is activated with uniform probability. Therefore, each node $i \in c$ runs a rate $\frac{|\mathcal{N}_c|}{2K_c}$ Poisson process independently. Clique c becomes active when the clock of any node $i \in c$ ticks. This means that a clique c runs a Poisson process with rate $\frac{|\mathcal{N}_c|}{2}$. This corresponds to a global clock of rate $|\mathcal{E}_R|$, and implies that each clique becomes active with probability $\frac{|\mathcal{N}_c|}{2|\mathcal{E}_R|}$. In each time-slot t, two neighboring nodes i and j in neighboring cliques c and l, respectively, communicate with probability $p_{cl} = \frac{1}{2|\mathcal{E}_R|}$ and update their current values as

$$h_c(t) = h_l(t) = \left(h_c(t-1) + h_l(t-1)\right)/2, \tag{4.4}$$

where c and l denote clique index and $h_c(t)$ denotes the value of clique c at the end of time-slot t. Subsequently, nodes i and j broadcast the updated estimates $h_c(t)$ and $h_l(t)$ to all the nodes in clique c and l, respectively. Notice that after a sufficient number of iterations, (4.4) does lead to the average h_{ave}, but generally not to the average $g_{ave} = \frac{1}{N}\sum_{i=1}^{N} g_i(0)$. In case one is interested in the average g_{ave}, a second consensus algorithm can be run in order to compute $K_{ave} = \frac{1}{C}\sum_{c=1}^{C} K_c$ (the average number of nodes per clique), after which g_{ave} after a sufficient number of iterations is given by $g_{ave} = h_{ave}/K_{ave}$.

4.5 Clique-based Distributed Beamformer

The beamformer in (4.3) can be seen as a ratio of two averages. Using a consensus algorithm over non-overlapping cliques (see Section 4.4), it is possible to estimate these averages in distributed fashion. We refer to this distributed beamformer as clique-based distributed beamformer (CbDB).

We assume that each node i in the WASN for a given time frame has the initial values $\tilde{Y}_i(0) = d_i^* \sigma_{V_i}^{-2} Y_i$ and $\tilde{d}_i(0) = d_i^* \sigma_{V_i}^{-2} d_i$, where Y_i is obtained by the microphone at node i. To focus on the distributed processing, we estimate $\sigma_{V_i}^2$ during noise only periods and assume that the acoustic transfer function d_i of each node i to be known.

After finding all non-overlapping cliques, each non-overlapping clique c in the network has the initial values $\hat{Y}_c(0) = \sum_{i \in c} \tilde{Y}_i(0)$ and $\hat{d}_c(0) = \sum_{i \in c} \tilde{d}_i(0)$. With the initial values $\hat{Y}_c(0)$ and $\hat{d}_c(0)$ per clique $c \in \mathcal{V}_R$, the beamformer output in (4.3) is given by

$$Z = \hat{Y}_{ave}/\hat{d}_{ave}, \tag{4.5}$$

where $\hat{Y}_{ave} = \frac{1}{C}\sum_{c=1}^{C} \hat{Y}_c(0)$ and $\hat{d}_{ave} = \frac{1}{C}\sum_{c=1}^{C} \hat{d}_c(0)$. To find the average value $\hat{Y}_{ave}$ and $\hat{d}_{ave}$ in a distributed way, the CbDB uses the proposed DCvNC algorithm. Let $\hat{\mathbf{Y}}(t)$ be a C-dimensional vector defined as $\hat{\mathbf{Y}}(t) = \left[\hat{Y}_1(t), \cdots, \hat{Y}_C(t)\right]^T$, similarly, all $\hat{d}_c(t)$ are stacked in a C-dimensional vector $\hat{\mathbf{d}}(t)$. In vector form, the CbDB

at iteration t is given by

$$\hat{\mathbf{Y}}(t) = \mathbf{U}(t)\hat{\mathbf{Y}}(t-1) \tag{4.6}$$

$$\hat{\mathbf{d}}(t) = \mathbf{U}(t)\hat{\mathbf{d}}(t-1) \tag{4.7}$$

$$\tilde{Z}_i(t) = \hat{Z}_c(t) = \hat{Y}_c(t)/\hat{d}_c(t), \;\; i \in c, \tag{4.8}$$

with $\tilde{Z}_i(t)$ the CbDB output of node $i \in c$ at iteration t and $\mathbf{U}(t)$ is a $C \times C$-dimensional update matrix, which is selected independently across time. Matrix $\mathbf{U}(t)$ is given by

$$\mathbf{U}(t) = \mathbf{I} - \frac{1}{2}\left(e_c - e_l\right)\left(e_c - e_l\right)^T, \tag{4.9}$$

where e_c is a C-dimensional unit vector with the cth component equal to 1 and $\mathbf{I}$ is the C-dimensional identity matrix.

4.6 Convergence Analysis

The probabilities p_{cl} that neighboring cliques c and l communicate can be stacked in a $C \times C$-dimensional probability matrix as $\mathbf{p} = \frac{\mathbf{A}_R}{2|\mathcal{E}_R|}$, where $\mathbf{A}_R$ is a $C \times C$ symmetric matrix with $a_{cl} = 1$ if $(c,l) \in \mathcal{E}_R$. The expectation of the update matrix in $\mathcal{G}_R$ can then be computed as [8],

$$E[\mathbf{U}] = \mathbf{I} - \mathbf{L}_R/\left(2\left|\mathcal{E}_R\right|\right), \tag{4.10}$$

where $\mathbf{L}_R = \mathbf{D}_R - \mathbf{A}_R$ is the Laplacian matrix of graph $\mathcal{G}_R$ with

$$\mathbf{D}_R = \text{diag}\left\{\left|\mathcal{N}_1\right|, \cdots, \left|\mathcal{N}_C\right|\right\}.$$

Since the expectation matrix $E[\mathbf{U}]$ is positive semi-definite doubly-stochastic, and the graph corresponding to $E[\mathbf{U}]$ is connected, $\hat{\mathbf{Y}}(t)$ and $\hat{\mathbf{d}}(t)$ are guaranteed to converge to the average value $\hat{Y}_{ave}\mathbf{1}$ and $\hat{d}_{ave}\mathbf{1}$ in expectation [5], where $\mathbf{1}$ denotes the vector of all ones. This guarantees that the output $\tilde{Z}_i$ of the CbDB converges to the optimal output Z as long as $\hat{d}_{ave} \neq 0$.

To assess the convergence rate of the CbDB, we consider the convergence error $\epsilon(t) = \frac{\left\|\hat{\mathbf{Y}}(t) - \hat{Y}_{ave}\mathbf{1}\right\|}{\left\|\hat{\mathbf{Y}}(0)\right\|}$, and in analogy with [5] define the convergence time of the CbDB $T_{ave}(\xi)$ as

$$T_{ave}(\xi) = \sup_{\hat{\mathbf{Y}}(0)} \inf_{t=0,1,\cdots} \left\{Pr\left(\epsilon(t) \geq \xi\right) \leq \xi\right\}. \tag{4.11}$$

From the definition of $T_{ave}(\xi)$ given in (4.11), the upper and lower bounds for $T_{ave}(\xi)$ are given by [5]

$$\frac{0.5\log\xi^{-1}}{\log\lambda_2(E[\mathbf{U}])^{-1}} \leq T_{ave}(\xi, E[\mathbf{U}]) \leq \frac{3\log\xi^{-1}}{\log\lambda_2(E[\mathbf{U}])^{-1}}. \tag{4.12}$$

Equation (4.12) shows that the convergence rate of the CbDB depends on the second largest eigenvalue of $E[\mathbf{U}]$. The smaller the magnitude of $\lambda_2(E[\mathbf{U}])$, the faster the convergence. From (4.10), $\lambda_2(E[\mathbf{U}])$ can be computed as

$$\lambda_2(E[\mathbf{U}]) = 1 - \frac{1}{2|\mathcal{E}_R|}\lambda_{C-1}(\mathbf{L}_R), \tag{4.13}$$

with $\lambda_{C-1}(\mathbf{L}_R)$ the second smallest eigenvalue of $\mathbf{L}_R$. Substituting (4.13) into (4.12) and using the Taylor series expansion, the upper bound for $T_{ave}(\xi)$ can now be written in terms of the eigenvalue $\lambda_{C-1}(\mathbf{L}_R)$. That is

$$T_{ave,U}(\xi, \mathbf{L}_R) = \frac{3\log\xi^{-1}}{\log\left(1-\frac{1}{2|\mathcal{E}_R|}\lambda_{C-1}(\mathbf{L}_R)\right)^{-1}} \leq \frac{6|\mathcal{E}_R|\log\xi^{-1}}{\lambda_{C-1}(\mathbf{L}_R)}. \tag{4.14}$$

From the lower bound on the eigenvalue $\lambda_{C-1}(\mathbf{L}_R)$ given in [12], $\lambda_{C-1}(\mathbf{L}_R)$ can be shown to be bounded by

$$\frac{4}{CD(\mathcal{G}_R)} \leq \lambda_{C-1}(\mathbf{L}_R), \tag{4.15}$$

with $D(\mathcal{G}_R)$ the diameter of $\mathcal{G}_R$. Combining (4.15) with (4.14), $T_{ave,U}(\xi, \mathcal{G}_R)$ can be written in terms of the diameter and number of cliques in the graph $\mathcal{G}_R$. That is

$$T_{ave,U}(\xi, \mathcal{G}_R) \leq \frac{3}{2}CD(\mathcal{G}_R)|\mathcal{E}_R|\log\xi^{-1}. \tag{4.16}$$

The convergence rate of the CbDB and the DDSB can now be compared using the upper bounds of $T_{ave,U}(\xi, \mathcal{G}_R)$ and $T_{ave,U}(\xi, \mathcal{G})$, respectively. Since the DDSB is performed in graph $\mathcal{G}$ and the CbDB is performed in $\mathcal{G}$'s compressed graph $\mathcal{G}_R$, we have $C \leq N$, $|\mathcal{E}_R| \leq |\mathcal{E}|$ and $D(\mathcal{G}_R) \leq D(\mathcal{G})$. In combination with (4.16), it follows that $T_{ave,U}(\xi, \mathcal{G}_R) \leq T_{ave,U}(\xi, \mathcal{G})$. This implies that, with high probability, the CbDB converges faster than the DDSB.

4.7 Computer Simulations

In this section, the performance of the presented DCvNC and CbDB is illustrated via a simulated WASN. First, we compare the convergence rate and robustness of the DCvNC with the randomized gossip algorithm [5] and the cluster-based gossip algorithm [8] using synthetic data. After that, the performance of the CbDB is demonstrated on speech data.

We simulate a network of 20 nodes and 40 edges and consider that each node i has the initial value $\tilde{X}_{i,m} = X + V_{i,m}$, where m is a realization index, X is a constant that is to be estimated in this experiment which is degraded by independent and identically distributed (i.i.d.) zero-mean Gaussian variables $V_{i,m}$. To compare the DCvNC with the randomized gossip and the cluster-based gossip algorithm, we measure the mean convergence error (MCE) as a function of used transmissions as,

$$\text{MCE} = \frac{1}{M}\sum_{m=1}^{M}\left\|\tilde{\mathbf{X}}_m(t) - X\mathbf{1}\right\| / \left\|\tilde{\mathbf{X}}_m(0)\right\|, \tag{4.17}$$

with $M = 5000$ the number of realizations. Here, one transmission is the sending of data from one node. The MCE is shown in Fig. 4.1(a) as a function of transmissions of the overall network. Both the proposed DCvNC and the cluster-based gossip algorithm converge much faster than the randomized gossip algorithm. Due to the centralized structure of clusters, the cluster-based algorithm will be more sensitive to nodes failure than the DCvNC. In order to test the robustness of these algorithms, we repeat this experiment for the case that one of the nodes randomly disappears and also average this performance over M realizations. The result is shown in Fig. 4.1(b), from which we see that the DCvNC is more robust than the cluster-based gossip algorithm, since the disappearing node can be a cluster head in the cluster-based algorithm.

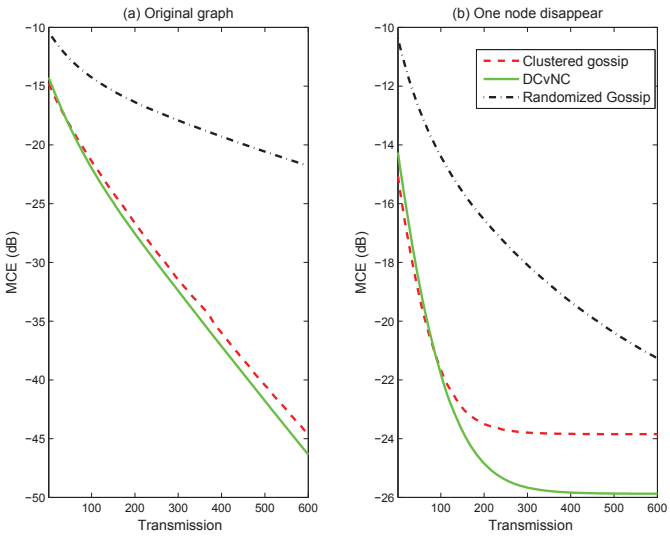


Figure 4.1: 20 nodes randomly connected with 40 edges.

Next, we again simulate an WASN where 20 microphones are randomly connected with 40 edges. We assume that the 20 wireless microphones, a speech source and a noise source are randomly distributed in a 10m $\times$ 10m $\times$ 5m room, and each microphone gathers noisy speech at a sampling frequency of $f_s = 16$ kHz. We use a 30 sec. speech signal [13] as a speech source and a single zero-mean white Gaussian signal as a point noise source. To demonstrate the distributed algorithms, we assume that the distance l_i between microphone i and the desired signal source is known, and the acoustic transfer function d_i of each node i is determined by gain and delay values as $d_i = x_i e^{-j\omega_k \tau_i}$, where $x_i = 1/l_i$ and $\tau_i = \frac{l_i}{c} f_s$ denote the damping and delay coefficient, respectively, with c the speech of sound. All nodes process the signals

frame-by-frame in the DFT domain with a 50%-overlapping Hann window of 25 ms. To assess the performance, we make use of the mean-square error (MSE) between the estimated clean speech coefficients $\hat{Z}_i(k,m)$ from the distributed beamformers and the desired speech coefficient S, given by

$$\mathrm{MSE}_i = \frac{1}{MK} \sum_{m=1}^{M} \sum_{k=1}^{K} \left\| \hat{Z}_i(k,m) - S(k,m) \right\|^2 . \tag{4.18}$$

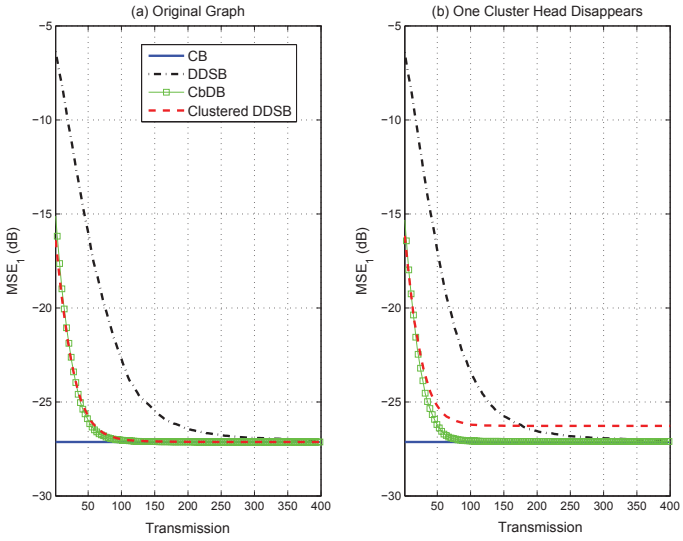


Figure 4.2: MSE of node 1 with -1dB input SNR versus transmission.

Fig. 4.2 shows the comparison in terms of MSE between the proposed CbDB, the DDSB and the cluster-based DDSB outputs of node 1 and the MSE of the optimal centralized beamformer (CB) output. Similar result are obtained for the other nodes. The results in Fig. 4.2(a) show that all distributed algorithms reach the same performance as the centralized beamformer after enough transmissions, but both the CbDB and the cluster-based DDSB converge much faster than the DDSB. We also compare the robustness of the CbDB and the cluster-based DDSB in the case that one microphone which served as a cluster head in the cluster-based DDSB disappears. Fig. 4.2(b) shows that the CbDB has better performance and is more robust than the cluster-based DDSB when nodes disappear, since the cluster-based DDSB converges to a larger MSE.

4.8 Conclusions

To improve the convergence speed of the previously distributed delay-and-sum beamformer (DDSB), we proposed in this article a clique-based distributed beamformer (CbDB) for speech enhancement via non-overlapping cliques in a randomly connected WSN. Without any central processor and network topology constraint, the CbDB converges asymptotically to the optimal centralized beamformer. Furthermore, we investigate the convergence rate of the distributed beamformers which is inversely proportional to the second smallest eigenvalue of the Laplacian matrix of the graph and compare the convergence rate of the CbDB with the DDSB. The simulation results show that both the CbDB and the cluster-based DDSB converge much faster than the DDSB while the robustness of the CbDB is better than the cluster-based DDSB.

References

[1] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[2] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[3] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer for speech enhancement in wireless sensor networks via randomized gossip. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 4037–4040, 2012.

[4] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[5] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[6] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng, and W. B. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[7] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer in regular networks based on synchronous randomized gossip. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[8] W. Li and H. Dai. Cluster-based distributed consensus. *IEEE Trans. Wireless Communications*, 8(1):28 –31, Jan. 2009.

[9] M. Zheng, M. Goldenbaum, S. Stanczak, and H. Yu. Fast average consensus in clustered wireless sensor networks by superposition gossiping. In *IEEE Wireless communications and networking conference*, pages 1982–1987, Jun. 2012.

[10] L. Schenato and F. Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, 2011.

[11] C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Communication of the ACM*, 16(9):575–577, 1973.

[12] B. Mohar. Eigenvalues, diameter, and mean distance in graphs. *Graphs and combinatorics*, 7(1):53–64, 1991.

[13] J. S. Garofolo. DARPA TIMIT acoustic-phonetic speech database. *National Institute of Standards and Technology (NIST)*, 1988.

Chapter 5

Distributed Estimation of the Inverse of the Correlation Matrix for Privacy Preserving Beamforming

---

This chapter is published as "Distributed Estimation of the Inverse of the Correlation Matrix for Privacy Preserving Beamforming", by Y. Zeng and R. C. Hendriks in the Proceedings of *Elsevier signal processing*, July 2014.

5.1 Introduction

To improve the quality and intelligibility of speech processing applications under noisy environments, it is customary to equip them with a single- or multi-microphone noise reduction algorithm (for an overview see e.g., [1] [3] [2]). As multi-microphone noise reduction algorithms can take advantage of spatial diversity, they usually lead to better speech quality and intelligibility than their single-microphone counterparts. It is in particular the number of microphones and their placement that determine the potential performance of a multi-microphone noise reduction algorithm. However, as most mobile speech processing devices have relatively small dimensions, the number of microphones as well as their placement is rather restricted.

Using so-called wireless acoustic sensor networks (WASNs), it is possible to use a much larger number of microphones that are distributed in the environment and where their placement is not restricted by the device itself. This allows a further increase in noise reduction performance. However, the conventional multi-microphone noise reduction algorithms (e.g. [3] [2]) are characterized by having one processor where all data is processed centrally. Such centralized algorithms are less suitable for a WASN, as they may require higher energy consumption or transmission bandwidth than necessary. The fact that the sensors in a WASN are all equipped with a (simple) processor potentially owned by different users, allows to perform intermediate processing of data without the need to first send all data to a single point in the network. This has recently led to an increased research interest to distributed speech enhancement algorithms, see e.g., [4] [5] [6] [7].

As the processors and sensors in the WASN context are not necessarily anymore owned by a single user, distributed processing might come with serious privacy risks. These could range from an increased risk of being eavesdropped to an increased risk that private data or information becomes public. Within the speech enhancement context, such privacy issues were first addressed in [8] and [9] for two scenarios. The scenario in [8] considered the case where a user keeps the exact source of interest private for other users, while [9] considered the scenario where eavesdropping by untrusted third parties is overcome. Both contributions employed homomorphic encryption [10] to provide the necessary privacy. However, homomorphic encryption is computational very complex, and requires very high bit rates for data transmission. In this work, we consider a different approach and develop a framework for distributed signal estimation employing a WASN, while providing the user a certain level of privacy with respect to the source of interest. We consider the case where the users of the network do not want to share to which specific source in the environment they are listening, while they do want to make use of the WASN to estimate their signal of interest.

More specifically, the application scenario that we consider in this work, is the one where multiple users make use of a WASN that consists of many processors (including their own) and where each processor is equipped with multiple microphones. The users can use the additional sensors in the WASN to obtain an improved estimate of their signal of interest, which can be different for each user and is usually determined by the steering vector of the beamformer. However, because of privacy reasons, the users want to keep the specific source they are interested in private, i.e., the steering

vector. Although the microphone signals might be public, hiding the steering vector will overcome that the exact combination of microphone signals required for specific target signal estimation is publicly known. Moreover, hiding the steering vector makes sure that none of the entities with access to the network is able to reveal which conversation or source is apparently of interest for a particular user. This will guarantee a certain amount of privacy to the users of the network. One way to guarantee privacy preservation on the source of interest, would be to send all data to all nodes and compute a conventional beamformer in every node. In this way, users do not need to make the steering vector public. However, this requires a lot of data transmission. Performing calculations in a distributed way will reduce the number of data transmissions, due to the fact that local nodes perform intermediate calculations. This leads to a data compression depending on the number microphones per node. We investigate thus the possibility that each user in the WASN estimates his signal of interest by performing distributed computations on the WASN data, while keeping the particular source of interest private. To do so, we concentrate on distributed estimation of one of the most well-known beamformers, the minimum variance distortionless response (MVDR) beamformer.

The MVDR beamformer depends on the inverse of the noise or noise+target spectral correlation matrix. Computing this inverse in a distributed manner is not trivial, as the data in a WASN is not centrally present and each element of the inverse of the correlation matrix is a function of the statistics of the noise or the noise+target at multiple microphones. In [7], a distributed MVDR was presented based on a randomized gossip algorithm [11] where the inversion of the noise correlation matrix was overcome by assuming the correlation matrix to be diagonal. This simplifies distributed computation of the MVDR, but it also compromises the performance as the noise is assumed to be uncorrelated across microphones. In [12] the MVDR was computed using a message passing algorithm [13]. However, this requires the network topology to be consistent with the noise correlation matrix, where two nodes are neighbors if their noise cross correlation is unequal to zero. This would require to adjust the transmission range of the nodes in the network to the correlation matrix and consequently, increase the energy usage for transmission or decrease the connectivity in the network. In [5] and [6], computation of the inverse of the correlation matrix was overcome by employing the generalized sidelobe canceler structure. However, this algorithm also constrains the topology of the network to be fully connected.

The contribution of this work is twofold. First, we present a method were each user can estimate a different signal of interest from a mix of many different signals by means of a distributed MVDR beamformer without the need to reveal the source of interest to other entities in the network. In order to do this, we develop an algorithm that enables distributed estimation of the inverse of a correlation matrix, which is the second contribution of this work. This algorithm for distributed estimation of the matrix inverse is based on the observation that in practice, correlation matrices are usually estimated recursively by exponential smoothing. Using the Sherman-Morrison formula [14], estimation of the inverse of the correlation matrix can be seen as a consensus problem and can be realized using gossip algorithms. Although the convergence error per time frame is decreased with increasing number of iterations when using gossip

algorithms, estimation errors might accumulate. This is caused by the fact that the correlation matrix is recursively estimated across time. These convergence errors can be eliminated using the distributed clique-based algorithm that we propose in this article. The performance of the proposed clique-based distributed estimation of the inverse correlation matrix is compared with the centralized estimation approach in terms of data transmissions in the scenario at hand where users want to estimate their signal of interest without revealing this to other entities. In addition, we show that this algorithm for distributed matrix inverse estimation can be used in the privacy preserving scenario to estimate a certain signal of interest.

The remainder of this article is organized as follows. In Section 25.2 we introduce the notation that we will use throughout this article and describe the problem. To guide the reader, we give in Section 35.3 a brief overview of gossip based algorithms. In Section 45.4 we show how estimation of the inverse correlation matrix can be seen as a consensus problem, and in Section 55.5 we introduce a framework to compute a privacy preserving MVDR beamformer in a distributed way, after which we show in Section 65.6 how this can be turned into a distributed estimation problem using gossip techniques. Then, in Section 75.7 we introduce a clique-based distributed algorithm in order to reduce the convergence error of the estimated inverse correlation matrix, which might otherwise accumulate across time. In Section 85.8 we present simulation results to demonstrate the presented algorithm and compare its performance in terms of computational costs with centralized estimation. Finally, in Section 95.9 conclusions are drawn.

5.2 Notation and Problem Description

Let $Y_m(f,k)$ denote a degraded speech short-time discrete Fourier transform (DFT) coefficient obtained on a frame-by-frame basis at a microphone with index-number m, frequency-bin index f and time-frame index k. The challenge for a speech enhancement algorithm is to estimate the underlying clean speech, given realizations of the noise+target DFT coefficients.

A common model that often underlies such algorithms is an additive noise model where the different sources are assumed to be mutually uncorrelated. Let $S_m(f,k)$ and $V_m(f,k)$ denote the target and disturbance DFT coefficient. The noise+target DFT coefficients are then given by

$$Y_m(f,k) = S_m(f,k) + V_m(f,k), \tag{5.1}$$

with $S_m(f,k) = d_m(f,k)S(f,k)$ and $S(f,k)$ the clean speech at the target location, and $d_m(f,k)$ the acoustic transfer function. Let index i denote a user (i.e., node) in the network. In the scenario that we consider, each user (i.e., each node) in the network can have a different target source $S(f,k)$, say $S_i(f,k)$, with thus a different acoustic transfer function (notice that this allows multiple microphones per node/user). The remaining sources are considered as disturbance (noise) for this user and are symbolized by the disturbance or noise DFT coefficient $V_i(f,k)$. What is considered to be noise for one user might be the target signal for another user. As such, (1) is different

for all users. However, to simplify notation, we consider here the noise model for one specific user.

The target and noise DFT coefficients are often assumed to be independent across time and frequency. This allows to omit the time and frequency indices for notational convenience. Further, we will use a stacked vector notation, that is, $\mathbf{Y} = [Y_1, \cdots, Y_M]^T$, with M the total number of microphones in the network and where $(\cdot)^T$ denotes transposition of a vector or a matrix. We use bold symbols to represent vectors or matrices, while scalars are denoted by non-bold symbols. For symbols representing random variables, we use the upper case to denote the random variable, and the corresponding lower case to denote its realization. The speech and noise vector $\mathbf{S}$ and $\mathbf{V}$ are defined in the same way as $\mathbf{Y}$. Let $\mathbf{d} = [d_1, \cdots, d_M]^T$ denote the steering vector representing the acoustic transfer function from the speech source to all microphones. Altogether this gives for one specific user

$$\mathbf{Y} = \mathbf{d}S + \mathbf{V} = \mathbf{S} + \mathbf{V}.$$

We assume that the M microphones in the WASN are grouped in N nodes. Each node has M_i microphones with $M = \sum_{i=1}^{N} M_i$. The different nodes in the network are connected via wireless links, while the microphones within the same node are assumed to be connected via wired connections. Each node in the network symbolizes a different device in the network potentially owned by a different user (hearing aid, mobile phone, etc.).

The goal of a multi-microphone noise reduction algorithm is to make an estimate of the clean speech DFT coefficient, say $\hat{S}$. Although many alternatives exist, an often used multi-microphone noise reduction algorithm is the MVDR beamformer. The MVDR beamformer is given by [3]

$$\mathbf{w} = \frac{\mathbf{R}_{\mathbf{Y}}^{-1}\mathbf{d}}{\mathbf{d}^{\mathbf{H}}\mathbf{R}_{\mathbf{Y}}^{-1}\mathbf{d}}. \tag{5.2}$$

where $\mathbf{R}_{\mathbf{Y}} = E\left[\mathbf{Y}\mathbf{Y}^H\right]$, with $E[\cdot]$ denoting the statistical expectation operator, and $(\cdot)^H$ denoting the Hermitian transposition. Similarly we define $\mathbf{R}_{\mathbf{V}} = E\left[\mathbf{V}\mathbf{V}^H\right]$. Alternatively, applying the matrix inversion lemma to (5.2) in combination with the assumption that target and noise are uncorrelated, the MVDR beamformer can also be written as

$$\mathbf{w} = \frac{\mathbf{R}_{\mathbf{V}}^{-1}\mathbf{d}}{\mathbf{d}^{\mathbf{H}}\mathbf{R}_{\mathbf{V}}^{-1}\mathbf{d}}. \tag{5.3}$$

The problem statement in this work is to allow all users in the network to estimate their own signal of interest in a distribute way using a WASN with the MVDR beamformer by means of (5.2), while keeping the source of interest private. This implies that the steering vector $\mathbf{d}_i$ for user (node) i should not be shared with other users. In practical applications, the steering vector $\mathbf{d}_i$ has to be estimated. In order to concentrate on the distributed estimation of the inverse of the correlation matrix, we assume here that each user i knows the steering vector towards the source of his or her interest. Among other methods, the steering vector can be determined by estimation of the

microphone locations and location/direction of the source of interest, or, by estimation of the relative transfer function [15].

By keeping the steering vector private, each node i can estimate its own source of interest S_i without revealing to other users which particular source this is. In this scenario it is thus the exact linear combination specified by the steering vector $\mathbf{d}_i$ that is kept secret. With this assumption, an estimate of the target S_i can be obtained as

$$\hat{S}_i = \frac{\mathbf{d}_i^H \mathbf{R}_\mathbf{Y}^{-1} \mathbf{Y}}{\mathbf{d}_i^H \mathbf{R}_\mathbf{Y}^{-1} \mathbf{d}_i}. \tag{5.4}$$

We will consider distributed estimation of the MVDR beamformer using the noisy correlation matrix $\mathbf{R}_\mathbf{Y}$ in the remaining part of this article. In case the objective is to estimate the MVDR based on the noise correlation matrix (as in (5.3)), the proposed algorithm can be combined with a voice activity detector (VAD) to distinguish between noise+target and noise-only segments.

An often used procedure to estimate the correlation matrix is recursive exponential smoothing, that is,

$$\hat{\mathbf{R}}_\mathbf{Y}(k) = \lambda \hat{\mathbf{R}}_\mathbf{Y}(k-1) + (1-\lambda)\mathbf{Y}(k)\mathbf{Y}^H(k), \tag{5.5}$$

where $0 \leq \lambda \leq 1$ denotes the exponential weighting factor and $\hat{\mathbf{R}}_\mathbf{Y}(k)$ denotes an estimate of $\mathbf{R}_\mathbf{Y}$ at time-frame k. With conventional centralized processing, this operation would be performed in a fusion center, where the observations for all nodes are gathered and the correlation matrix is estimated and transmitted to other nodes in the network that would require this estimate. In this work we employ an alternative way to estimate the inverse correlation matrix based on the Sherman-Morrison formula. This appears to be an important aspect, as it not only enables to compute the inverse correlation matrix in a distributed way, but also enables to compute the MVDR beamformer in a distributed fashion without the need to share the steering vector with other users.

5.3 Gossip Algorithms

To guide the reader, this section presents a brief overview of gossip algorithms. Gossip algorithms have been widely studied for in-network information processing in wireless sensor networks [11]. They can be used to solve consensus problems in a distributed way without any requirement of network topology. Given a randomly connected network of N nodes and an initial value $g_i(0)$ at each node i, a possible objective of a gossip algorithm could be to estimate the average value $g_{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N} g_i(0)$ of the initial values at each node i by using only local processing. By allowing neighboring nodes to exchange information and update this with a convex combination of their own and neighboring values (i.e., a linear combination of points with non-negative weights that sum up to one), convergence will be reached under certain conditions.

Gossip algorithms can be categorized into two classes, randomized, where each pair of neighboring nodes is chosen randomly based on a probabilistic model to update information, and deterministic, where neighboring nodes are chosen in a deterministic

way (e.g., by using knowledge on the network topology) to update information. With deterministic gossip the consensus will be achieved asymptotically, and with randomized gossip, consensus will be achieved asymptotically almost surely [16]. In typical gossip algorithms, nodes in the connected network make a convex combination of their own value and their values received from their neighbors. Let $g_i(t)$ denote the value of node i at the end of iteration t. In a given time-slot t, a typical iteration of a gossip algorithm consists of the selection of multiple nodes, e.g., the pair (i, j), and communication and update of their estimates, for example, $g_i(t) = g_j(t) = \frac{g_i(t-1)+g_j(t-1)}{2}$. Depending on the exact protocol, the averaging operations can be performed asynchronously or synchronously.

5.4 The Estimated Correlation Matrix

This section discusses that estimation of the inverse correlation matrix can be seen as a consensus problem.

The Sherman-Morrison formula [14] provides an explicit formula for the inverse of a matrix $\mathbf{B} = \mathbf{A}+\mathbf{u}\mathbf{v}^T$, where $\mathbf{A}$ is an invertible $M \times M$ matrix and $\mathbf{u}$ and $\mathbf{v}$ are M-dimensional column vectors. Matrix $\mathbf{B}$ is invertible if and only if $1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u} \neq 0$. In this case, the Sherman-Morrison formula is given by

$$\mathbf{B}^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\mathbf{u}\mathbf{v}^T\mathbf{A}^{-1}}{1 + \mathbf{v}^T\mathbf{A}^{-1}\mathbf{u}}. \tag{5.6}$$

From (5.5) and (5.6), the inverse correlation matrix $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$ can be obtained as [17]

$$\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k) = \lambda^{-1}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1) - \frac{\lambda^{-2}(1-\lambda)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)}{1 + \lambda^{-1}(1-\lambda)\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)}. \tag{5.7}$$

Notice that the computational complexity of (5.7) in each time-frame is only $O(M^2)$ when a previous time-frame estimate $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$ is available, while the computational complexity of directly computing the inverse in $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$ is $O(M^3)$. Besides the lower computational complexity of (5.7) over the inverse that results from (5.5), the structure of (5.7) makes it possible to estimate the inverse correlation matrix in a distributed fashion. More specifically, each node will have an estimate $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$ available from the iteration performed in the previous time frame $k-1$. To compute $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$, it is required to compute $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ and $\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ in a distributed way.

Let $\mathbf{r}_1, \cdots, \mathbf{r}_M$ be the columns of $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$. We then have

$$\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k) = [\mathbf{r}_1, \cdots, \mathbf{r}_M]\,\mathbf{Y}(k) = \sum_{m=1}^{M} \mathbf{r}_m Y_m, \tag{5.8}$$

i.e., a weighted sum of the columns of $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$, where the weights are determined by the noise+target DFT coefficients in $\mathbf{Y}(k)$. In addition, let $\mathbf{a} = \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$.

We can then write

$$\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k) = \mathbf{Y}^H(k)\mathbf{a} = \sum_{m=1}^{M} Y_m^* a_m, \tag{5.9}$$

which is a weighted sum over $\mathbf{a}$ with the noise+target DFT coefficients as weights. Obviously, given $\mathbf{Y}(k)$ and $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$, the expressions in (5.8) and (5.9) can also be seen as averaging operations, if a normalization over the number of nodes N would be included. Such averaging operations can be computed in a distributed manner using gossip algorithms. To do so, we need two rounds of gossip iterations. The first gossip round is used to compute

$$\mathcal{E}_{\text{ave}}^1(k) = \frac{1}{N}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k), \tag{5.10}$$

and the second round is used to compute

$$\mathcal{E}_{\text{ave}}^2(k) = \frac{1}{N^2}\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k). \tag{5.11}$$

For notational convenience we will denote intermediate estimates of $\mathcal{E}_{\text{ave}}^1(k)$ in frame k, iteration t and node i by $\mathcal{E}_{i,t}^1(k)$ and intermediate estimates of $\mathcal{E}_{\text{ave}}^2(k)$ in frame k and iteration t by $\mathcal{E}_{i,t}^2(k)$.

5.5 Distributed Privacy Preserving MVDR Computation

This section introduces an approach to perform a distributed MVDR beamformer based on the presented distributed estimation of the inverse correlation matrix with privacy preservation of a source of interest. We first demonstrate how any user in the network can employ the presented framework for (distributed) matrix inverse estimation in order to compute an MVDR beamformer in a distributed fashion without revealing their source of interest.

Given that (5.8) and (5.9) are computed in a distributed fashion, every user can compute an estimate of the inverse correlation matrix by means of (5.7), that is $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$. By means of (5.8), every user has a local estimate of $\mathbf{a} = \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$. Given that the user knows the steering vector for his source of interest, say $\mathbf{d}_i$, the target that is of interest for the user at node i can be estimated as $\hat{S}_i = \frac{\mathbf{d}_i^H\mathbf{a}}{\mathbf{d}_i^H\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{d}_i}$. Here $\mathbf{d}_i$ denotes the steering vector towards the source of interest for user i (at node i). Both $\mathbf{a}$ and $\mathbf{R}_{\mathbf{Y}}$ are computed in distributed fashion using gossip techniques. This means that no shared central processor is needed, but that every user has its own estimates of $\mathbf{a}$ and $\mathbf{R}_{\mathbf{Y}}$. The steering vector $\mathbf{d}_i$ is only known locally by the user. In this way, every user can compute his signal of interest without sharing the steering vector. An alternative to the presented distributed MVDR would be the use of one centralized MVDR that calculates ${\mathbf{R}_{\mathbf{Y}}}^{-1}$ in one centralized node and subsequently transmit this matrix together with $\mathbf{Y}(k)$ to all other nodes. However, as will be shown in the analysis in

Section 5.6, this will lead to a much larger transmission cost compared to distributed computation. Using distributed computations, intermediate calculations can be performed that compress the data prior to transmission.

5.6 Gossip-based Distributed Estimation of the Correlation Matrix

In this section, we first discuss a gossip-based algorithm for distributed estimation of the inverse correlation matrix. Next, we give a convergence error analysis of the algorithm and show that the convergence errors accumulate across time due to the recursive estimation procedure. Therefore, we present in Section 5.7 an alternative approach that eliminates the accumulating convergence error. Although our interest is to estimate the inverse correlation matrix, the error analysis will be based on the correlation matrix as this will be more insightful.

5.6.1 Estimation of $\mathbf{R}_{\mathbf{Y}}^{-1}(k)$ Using Gossip

Let $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$ denote the estimated inverse of the correlation matrix at node i and time-frame k. To estimate $\mathbf{R}_{\mathbf{Y}}^{-1}(k)$ recursively as given in (5.7), we assume that at each node i initializes the inverse of the noise+target correlation matrix as $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(0) = \mathbf{I}$, where $\mathbf{I}$ is a $M \times M$ dimensional unit matrix. This requires that the dimension M, i.e., the total number of microphones in the network is known. When this is unknown, it can be estimated using gossip based techniques, see e.g., [18] and references therein.

Before starting gossip iterations between nodes in a time-frame k, first the initial value $\mathcal{E}_{i,0}^1(k)$ needs to be determined for each node i. This is obtained by computing $\mathcal{E}_{i,0}^1(k) = \sum_{m \in M_i} \mathbf{r}_m Y_m$. Notice that $\mathcal{E}_{i,0}^1(k)$ is an $M \times 1$ dimensional vector, since $\mathbf{r}_m$ is $M \times 1$ dimensional vector. Then, gossip iterations can be used to estimate $\frac{1}{N}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ in frame k in a distributed manner, by computing the average in each iteration t between two nodes i and j, i.e., $\frac{\mathcal{E}_{i,t}^1(k)+\mathcal{E}_{j,t}^1(k)}{2}$. Let $\mathcal{E}_{i,T_1}^1(k)$ denote the final estimate of round 1 at node i and iteration T_1. According to the convergence properties of gossip algorithms, the estimates $\mathcal{E}_{i,T_1}^1$ at all nodes are guaranteed to converge to the average value (5.10) after a sufficient number of iterations.

As soon as $\mathcal{E}_{i,T_1}^1(k)$ is known accurately enough at all nodes, a second gossip round can be started to estimate $\frac{1}{N^2}\mathbf{Y}^H(k)\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$. First, each node i determines the initial value $\mathcal{E}_{i,0}^2(k)$, that is, $\mathcal{E}_{i,0}^2(k) = \sum_{m \in M_i} Y_m^*[\mathcal{E}_{i,T_1}^1(k)]_m$, where $[\cdot]_m$ indicates the mth element of the corresponding vector. Given $\mathcal{E}_{i,0}^2(k)$, gossip iterations can be performed to estimate $\mathcal{E}_{\text{ave}}^2(k)$, by computing the average in each iteration t between two nodes i and j, e.g., $\frac{\mathcal{E}_{i,t}^2(k)+\mathcal{E}_{j,t}^2(k)}{2}$. The final $\mathcal{E}_{i,T_2}^2$ at all nodes are guaranteed to converge to the average value (5.11) when using enough iterations T_1 and T_2 in both gossip rounds. Notice that the convergence error in the second estimation round depends on the convergence error in the first estimation round. Based on the estimates of $\mathcal{E}_{i,T_1}^1(k)$ and $\mathcal{E}_{i,T_2}^2(k)$, each node i can locally update the estimate $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$

using (5.7) as

$$\hat{\mathbf{R}}^{-1}_{\mathbf{Y},i}(k) = \lambda^{-1}\hat{\mathbf{R}}^{-1}_{\mathbf{Y},i}(k-1) - \frac{\lambda^{-2}(1-\lambda)N^2\mathcal{E}^1_{i,T_1}(k)\mathcal{E}^{1,H}_{i,T_1}(k)}{1+\lambda^{-1}(1-\lambda)N^2\mathcal{E}^2_{i,T_2}(k)}. \tag{5.12}$$

5.6.2 Convergence Error Analysis

To assess the performance of the gossip-based distributed estimation algorithm, we define the squared error (SE) between the inverse of the gossip-based distributed estimate of the inverse correlation matrix and the centralized optimal estimate of the correlation matrix for a given time-frame k as

$$\text{SE}(k) = \left\| \hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) - \hat{\mathbf{R}}_{\mathbf{Y},c}(k) \right\|_{\text{fro}}, \tag{5.13}$$

where $||\cdot||_{\text{fro}}$ denotes the Frobenius norm, $\hat{\mathbf{R}}_{\mathbf{Y},c}(k)$ denotes the centralized estimated correlation matrix using (5.5) and $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k)$ is the inverse of the estimated inverse correlation matrix using T_1 and T_2 iterations in the first and second gossip round, respectively. Depending on T_1, T_2 and the network topology, a convergence error might be introduced by the gossip-based algorithm, and as the matrix inverse is computed recursively across time, these errors might accumulate. To investigate this, we define the error introduced by the gossip operation at time-frame k by the matrix Δ_k as

$$\Delta_k = \hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) - \lambda\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k-1) - (1-\lambda)\mathbf{Y}(k)\mathbf{Y}^H(k). \tag{5.14}$$

From (5.14) in combination with the initial value $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0)$, we can write $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k)$ as

$$\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(k) = \sum_{n=1}^{k}\lambda^{k-n}\Delta_n + \lambda^k\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0) + \sum_{n=1}^{k}\lambda^{k-n}(1-\lambda)\mathbf{Y}(n)\mathbf{Y}^H(n). \tag{5.15}$$

Further, we can write $\hat{\mathbf{R}}_{\mathbf{Y},c}(k)$ as

$$\hat{\mathbf{R}}_{\mathbf{Y},c}(k) = \lambda^k\hat{\mathbf{R}}_{\mathbf{Y},c}(0) + \sum_{n=1}^{k}\lambda^{k-n}(1-\lambda)\mathbf{Y}(n)\mathbf{Y}^H(n). \tag{5.16}$$

From (5.15) and (5.16) in combination with (5.13) and the fact that $\hat{\mathbf{R}}_{\mathbf{Y},T_1,T_2}(0) = \hat{\mathbf{R}}_{\mathbf{Y},c}(0) = \mathbf{I}$, it then follows that

$$\text{SE}(k) = \left\| \sum_{n=1}^{k}\lambda^{k-n}\Delta_n \right\|_{\text{fro}}. \tag{5.17}$$

Equation (5.17) shows that the SE depends on the summation of the gossip error across all time frames, which indicates that the SE between the output of the gossip-based distributed estimation and the output of the centralized estimation accumulates with increasing number of time frames. The reason that the SE accumulates across

time frames is that the correlation matrix is recursively updated across time frames and the gossip algorithms have a convergence error at each time frame. This depends on the number of iterations and on the way the sequence of gossip operations is performed. Although it is interesting to analyze the changes of the SE as a function of time frames, it is not straightforward to do this using analytic expressions. In Section 5.8, we will use simulations to illustrate the SE behavior versus time frames.

5.7 Clique-based Distributed Estimation of the Inverse Correlation Matrix

One way to eliminate the convergence error described in Section 5.6.2, is to make sure that the gossip algorithm aggregates the information from all nodes. To do this in an efficient manner, we first compress the graph using non-overlapping cliques as previously presented in [19] for gossip-based estimation. For this compressed graph we then determine the spanning tree.

A clique is a fully connected sub-graph. The cliques of a graph $\mathcal{G}$ can be overlapping, since each node can belong to multiple cliques. Here we consider non-overlapping cliques only, where each node belongs to only one clique. In this section, we first present a clique-based distributed (CbD) algorithm based on the non-overlapping cliques of a graph $\mathcal{G}$. Then we study the performance of the CbD algorithm and compare the clique-based distributed estimation of the inverse correlation matrix with a centralized estimation algorithm in terms of required data transmissions. The performance comparison will be made for a fully connected and string connected network.

5.7.1 Clique-based Distributed Algorithm

In [19] it was proposed how a method to determine all non-overlapping cliques in a distributed fashion given a randomly connected graph $\mathcal{G}$. We assume that there are C non-overlapping cliques in graph $\mathcal{G}$. These cliques can be used to compress the original graph $\mathcal{G}$ by representing each clique by a single node in $\mathcal{G}_1$. An example is given in Fig. 5.1. The nodes are here denoted by g_i, with i the node index, while the cliques are denoted by h_j, with j the index of the clique. Notice that graph $\mathcal{G}_1$ is a compressed version of the original graph $\mathcal{G}$, since all nodes in a clique are represented by a single node in $\mathcal{G}_1$ and multiple edges between two neighboring cliques are compressed into a single edge in $\mathcal{G}_1$. In the compressed graph $\mathcal{G}_1$, two cliques are said to be neighbors if there is at least one direct link joining them. If more than one such links exist, only one of them is activated by random selection. The end nodes of active links are called gateway nodes. To eliminate the convergence error of gossip-based distributed algorithms and reduce the computational complexity of the CbD algorithm, the compressed graph $\mathcal{G}_1$ is further pruned to a spanning tree. Many approaches were proposed to define and compute spanning trees, see e.g., [20]. Let $\mathcal{G}_t$ denote a tree graph which is pruned from the compressed graph $\mathcal{G}_1$, let L denote the total number of levels of the graph $\mathcal{G}_t$ and let C^l denote the number of cliques in the lth level of

$\mathcal{G}_t$. We assume that each node i in $\mathcal{G}$ has an initial value $g_i(0)$. In the current case of estimating $N\mathcal{E}^1_{\text{ave}}$, $g_i(0)$ is given by $g_i(0) = \mathcal{E}^1_{i,0}(k) = \sum_{m\in M_i} \mathbf{r}_m Y_m$. Based on the tree graph $\mathcal{G}_t$, the CbD algorithm is described in Table 5.1. For the given initialization, this will finally lead to $N\mathcal{E}^1_{\text{ave}}$. In a similar way $N^2\mathcal{E}^2_{\text{ave}}$ can be estimated by initializing $g_i(0)$ as $g_i(0) = \sum_{m\in M_i} Y^*_m[N\mathcal{E}^1_{\text{ave}}(k)]_m$.

Table 5.1: CbD algorithm.

1. Initialize the level index $l = 1$ and $g_i = g_i(0)$, where g_i denotes the current value of node i.
2. Each clique c^l, where c^l denote a clique in the lth level, updates its estimation as $h_{c^l} = \sum_{i\in c^l} g_i$. To do so, each node i in the clique c^l broadcasts its value g_i to all other nodes in the clique.
3. Each clique c^l sends its estimates h_{c^l} to its neighboring clique c^{l+1} via gateway nodes one level up in the tree. The gateway node $j \in c^{l+1}$ updates its estimation as $g_j = g_j(0) + h_{c^l}$.
4. $l \to l + 1$.
5. Return to step 2 until $l = L$.
6. The root clique c^L at the top level updates its estimation as $h_{c^L} = \sum_{i\in c^L} g_i$ and sends the updated estimation h_{c^L} back to all other nodes in the lower levels of $\mathcal{G}_t$.

It is worth pointing out that the CbD algorithm can reach the summation of all initial node values in the network in a distributed manner. Thus, using the CbD algorithm algorithm, exact values for $N\mathcal{E}^1_{\text{ave}}$ and $N^2\mathcal{E}^2_{\text{ave}}$ in (5.10) and (5.11), respectively, are obtained, while the randomized gossip approach described in the previous section will always have a (small) convergence error. In combination with the Sherman-Morrison formula (5.6), the inverse correlation matrix can be obtained in a distributed manner in a similar way as with (5.12). We refer to this distributed algorithm as a clique-based distributed estimation of the correlation matrix (CbDECM).

5.7.2 Transmission Cost Analysis

This subsection discusses the required number of transmissions of the CbD algorithm. We assume that one transmission is the sending of a scalar value from one node to another. Given a connected network $\mathcal{G}$, the required number of transmissions of the

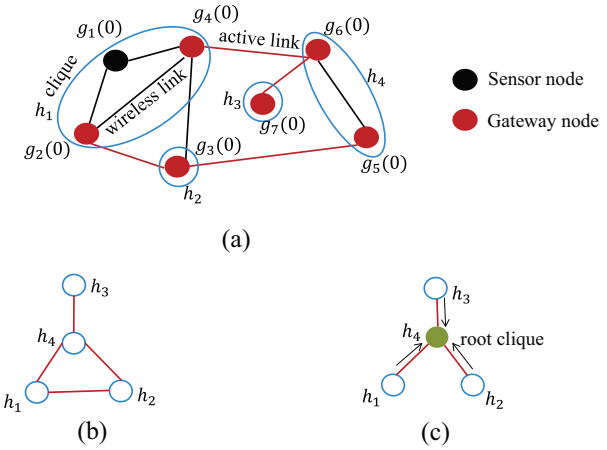


Figure 5.1: (a) The original graph $\mathcal{G}$. (b) The compressed graph $\mathcal{G}_1$. (c) The tree graph $\mathcal{G}_t$.

CbD algorithm will be explained below and is given by $T_d = \sum_{l=1}^{L} t^l$ with

$$t^l = \begin{cases} \sum_{c^l=1}^{\hat{C}^l} K_{c^l} + C^l & 1 \leq l < L, \\ K_{c^L} + 2(C-1) & l = L, \end{cases} \tag{5.18}$$

where t^l is the required number of transmissions of the CbD algorithm in the lth level of $\mathcal{G}_t$, K_{c^l} is the number of nodes in a clique c^l, and $\hat{C}^l$ is the number of non-overlapping cliques which consist of more than one node. Notice that $\hat{C}^l \leq C^l$, since C^l is the total number of non-overlapping cliques in the lth level of $\mathcal{G}_t$.

In $\mathcal{G}$, C non-overlapping cliques can be determined and represent C nodes in $\mathcal{G}_1$. The compressed graph $\mathcal{G}_1$ is then pruned to a tree graph $\mathcal{G}_t$ and the C cliques are distributed in the different levels of $\mathcal{G}_t$. For the CbD algorithm, we go through the tree $\mathcal{G}_t$ from the lowest level, $l = 1$ up to the root L. When l is smaller than L, each node i in a clique c^l broadcasts their data to all other nodes in the clique. The required number of transmissions for all nodes in the lth level is $\sum_{c^l=l}^{\hat{C}^l} K_{c^l}$. In addition, each clique c^l needs one transmission to send the updated estimates to its neighboring clique c^{l+1} in the $(l+1)$th level. Thus, the required number of transmissions in lth ($l < L$) level is equal to $\sum_{c^l=1}^{\hat{C}^l} K_{c^l} + C^l$.

At the top level of $\mathcal{G}_t$, there is only one clique which is the root clique of $\mathcal{G}_t$. Similar as the other cliques, each node i in the root clique broadcasts their data to all other nodes in the clique. On the other hand, the root clique has to send the updated estimates to all other nodes in the lower levels. Since the number of remaining cliques in $\mathcal{G}_t$ is $C-1$ and each clique requires one transmission to receive data from its neighboring clique and one transmission to synchronize all nodes in this clique with

the updated data, the required number of transmissions for sending data from the root clique to all other nodes is $2(C-1)$. Therefore, at the top level of $\mathcal{G}_t$, the total required number of transmissions is $K_{c^L} + 2(C-1)$.

Notice that t^l is the upper bound of the required number of transmissions, since there are some cliques that might consist of only one node. Moreover, the CbD algorithm costs one transmission less when a gateway node has two neighboring gateway nodes in two different cliques.

5.7.3 Performance Comparison

This subsection analyzes the relative performance between the proposed CbDECM algorithm and a centralized way for matrix inverse estimation in terms of data transmissions. This comparison is done in the given scenario where the source of interest (by means of the steering vector) is considered to be private. In such a scenario, each user is in need of the estimated inverse correlation matrix, i.e., $\hat{\mathbf{R}}_Y^{-1}$, as well as $\hat{\mathbf{R}}_Y^{-1}\mathbf{Y}$. Given knowledge of these two quantities, in combination with the locally known steering vector $\mathbf{d}_i$, every user (node) can estimate his source of interest by constructing the MVDR beamformer (by means of $\hat{S}_i = \frac{\mathbf{d}_i^H \hat{\mathbf{R}}_Y^{-1}\mathbf{Y}}{\mathbf{d}_i^H \hat{\mathbf{R}}_Y^{-1}\mathbf{d}_i}$). Although the proposed algorithm also inherently delivers an estimate of $\hat{\mathbf{R}}_Y^{-1}\mathbf{Y}$, we concentrate this comparison solely on the number of data transmissions required to compute the matrix inverse at each user's processor. For the centralized approach, which is our reference method, we therefore first gather all data at a single processor (which is assumed to be one of the nodes), after which the inverse correlation matrix is computed in a centralized fashion according to (5.7) and transmitted back to all users such that it can subsequently be used to estimate their signal of interest. Similarly, for the proposed approach, where (5.7) is computed in a distributed way based on the CbD algorithm presented in this section.

Consider a WASN with given size, the number of data transmissions for both the CbDECM and the centralized algorithm depend on the network topology. However, in practice, the exact network topology is unknown, complicating a comparison using analytic expressions in a randomly connected network. We therefore compare the CbDECM with the centralized algorithm in terms of the number of data transmissions in a fully connected network and a string connected network. In general, the fully connected topology has the best connectivity, while the string connected network has the worst connectivity. To do the performance comparison with analytic expressions, we assume that each clique in the network consists of K nodes and each node i has $M_i = u$ microphones. This means there are $N = KC$ nodes and $M = Nu$ microphones in the network and the inverse correlation matrix is an $M \times M$ dimensional matrix. This assumption is only made for ease of analytical analysis, but not required in practice.

In a fully connected network, $M - u$ data transmissions are needed to gather all observed signals in the central processor and then $\frac{M^2}{2} + \frac{M}{2}$ data transmissions are needed to send the lower or upper triangle of the estimated inverse correlation matrix back to all other nodes. Thus, the required number of data transmissions $T_{C,F}$ is given

by

$$T_{C,F} = \frac{M^2}{2} + \frac{3M}{2} - u, \tag{5.19}$$

where the subscripts C and F indicate the centralized algorithm and a fully connected network, respectively. Further, we use subscripts D and S to indicate the CbDECM algorithm and a string connected network, respectively.

The CbDECM algorithm requires two rounds of processing in order to compute the sums in (5.8) and (5.9), respectively. As a fully connected network can be seen as a single clique, the compressed graph $\mathcal{G}_1$ will consist of just one node and tree pruning is not necessary. Thus, the number of levels is given by $L = 1$. Based on (5.18), it then follows that for a scalar value it requires $T_d = K_{c^1}$ transmissions. As the number of cliques in a fully connected network is $C = 1$, and the number of nodes in that clique is $K = N = \frac{M}{u}$, it requires $K = \frac{M}{u}$ transmissions for one scalar value. The distributed estimation of the matrix inverse requires an estimate of a vector of length M (by means of (5.8)) and a scalar value (by means of (5.9)), which then leads to $\frac{M^2}{u}$ and $\frac{M}{u}$ data transmissions, respectively. Thus, the total required number of data transmissions $T_{D,F}$ is given by

$$T_{D,F} = \frac{M^2 + M}{u}. \tag{5.20}$$

To compare the computational cost of the CbDECM with the centralized algorithm in a fully connected network, their required number of data transmissions can be compared as

$$T_{C,F} - T_{D,F} = \frac{M^2 u + 3Mu - 2M^2 - 2M}{2u} - u. \tag{5.21}$$

We assume that $M > 2$. From (5.21), we then have that $T_{C,F} < T_{D,F}$ for $u = 1$ and $T_{C,F} > T_{D,F}$ for $u \geq 2$. This indicates that the computational cost of the centralized algorithm is larger than those of the CbDECM algorithm if there is more than one microphone per node. Table 5.1 gives a numerical comparison between $T_{C,F}$ and $T_{D,F}$ for $M = 500$ and $C = 10$ and various combinations of K and u. This shows that for a fully connected network, and when the number of microphones per node is $u > 1$, the proposed CbDECM algorithm always requires fewer data transmissions than the centralized approach.

In a string connected network where all C cliques are connected as a string, we assume that one node in the center of the string serves as the fusion center. The data is transmitted from both sides to this fusion center. The number of transmissions depends on C being odd or even. For compactness we only consider the case that C is even. For odd C a similar analysis can be carried out. When C is even, one side consists of $\frac{C}{2}$ cliques and the other side consists of $\frac{C}{2} - 1$ cliques. The required number of data transmissions for both sides sending data to the gateway nodes in the central clique is $2Ku\sum_{c=1}^{\frac{C}{2}} c + 2Ku\sum_{c=1}^{\frac{C}{2}-1} c$. Next, the required number of data transmissions for the central clique sending the data to the central node is $Ku(C-1) + (K-1)u$. The central node updates the estimates of the inverse correlation matrix and sends the estimates back to all other nodes in $\mathcal{G}$. Since the correlation matrix is a symmetric

matrix, the central node only needs to transmit the upper or lower triangle of the correlation matrix, which consists of $\frac{M^2}{2} + \frac{M}{2}$ variables. The required number of transmissions for all nodes to receive this information is $2(C-1)+1$, since the central node needs one transmission to broadcast the estimates to all other nodes in the central clique and $2(C-1)$ transmissions to send the data to the nodes in the remaining $C-1$ cliques. Here, each clique requires one transmission to send data to its neighboring clique and one transmission to synchronize all nodes in this clique with the updated estimates. Therefore, the required number of data transmissions of the centralized algorithm $T_{C,S}$ is given by

$$T_{C,S} = (C - \frac{1}{2})M^2 + (\frac{3}{2}C + \frac{1}{2})M - u. \tag{5.22}$$

For the CbDECM algorithm, the data transmissions can be obtained using the transmission analysis given in Section 5.7.2. In a string connected network, there are $L = C$ levels of graph $\mathcal{G}_t$, and each level consists of only one clique (by means of $C^l = \hat{C}^l = 1$). Since we assume that each clique consists of K nodes, we have $K_{c^l} = K$, $\forall l$. Thus, the required number of transmissions for the level $l < L$ is $K+1$. For the top level of $\mathcal{G}_t$, the required number of transmissions is $K + 2(C-1)$. Combining the transmissions with the fact that an M-dimensional vector is transmitted per transmission in the first round and a scalar value is transmitted per transmission in the second round, the required number of data transmissions $T_{D,S}$ is given by

$$T_{D,S} = (M+1)\left\{(K+1)(C-1) + K + 2(C-1)\right\} = (M+1)(N+3C-3). \tag{5.23}$$

We compare $T_{C,S}$ and $T_{D,S}$ for a given size network with $M = 500$ and $C = 10$. The required number of data transmissions of both the CbDECM and the centralized algorithm are given in Table 5.2 for various combinations of u and K, such that $\frac{M}{C} = Ku = 50$.

Table 5.2: The required number of data transmissions of the CbDECM algorithm and the centralized algorithm.

	$T_{C,F}$	$T_{D,F}$	$T_{C,S}$	$T_{D,S}$
K=1,u=50	125700	5010	2382700	18537
K=2,u=25	125725	10020	2382725	23547
K=5,u=10	125740	25050	2382740	38577
K=10,u=5	125745	50100	2382745	63627
K=25,u=2	125748	125250	2382748	138777
K=50,u=1	125759	250500	2382749	264027

This shows that the number of data transmissions for the proposed CbDECM algorithm is always smaller than for the centralized algorithm for the string connected network under the given parameter setting. It should be noted that this comparison in terms of required data transmissions is under the privacy preserving scenario where

each user (processing node) is in need of the matrix inverse. If the constraints on the privacy preserving aspect are loosened, and the centralized processor directly estimates the target signal followed by transmission of the estimated target signal, the required transmissions are significantly reduced.

5.8 Simulations

In this section, we first illustrate the performance of the gossip-based algorithm and the CbDECM algorithm when estimating the inverse correlation matrix in a simulated WASN. Secondly, we demonstrate the use of the proposed CbDECM algorithm in combination with the MVDR.

5.8.1 Simulation Environment

We simulate a wireless network with 6 cliques ($C = 6$), where each clique consists of 3 acoustic sensor nodes ($K = 3$ and $N = 18$) and where each acoustic node consists of 2 microphones ($u = 2$). The distance between the two microphones is 2 cm. The 6 cliques are (wirelessly) connected as depicted in the network in Fig. 5.2. A network like this could be obtained from a randomly connected network by first compressing it into a compressed network by finding the non-overlapping cliques in the network. Subsequently, the network can be pruned into a spanning tree. We consider a scenario where all sensor nodes, three speech sources and a noise source are placed in a $10\text{m} \times 8\text{m}$ rectangular area. The overall node positions relative to the target signal are shown in Fig. 5.3. Furthermore, the speech sources consist of 30 s speech signals sampled at 16 kHz originating from the Timit database [21], and the noise source is a White Gaussian noise signal. To model the microphone-self noise, an independent additive white noise source is added to each microphone signal at the SNR of 40 dB measured at the microphone that is furthest away from the target source. In this work, the simulation environment is assumed to be free of reverberation. The steering vectors $\mathbf{d}_i$ per user can then be calculated by gain and delay values as $\mathbf{d}_i = [a_{i,1}e^{-j\omega_f \tau_{i,1}}, \cdots, a_{i,M}e^{-j\omega_f \tau_{i,M}}]^T$, where $a_{i,m} = 1/l_{i,m}$ is the damping coefficient, and $\tau_{i,m} = \frac{l_{i,m}}{c} f_s$ is the delay in number of samples with $l_{i,m}$ the distance between microphone m and the desired speech source and $c = 340$ m/s the speed of sound. The smoothing constant λ, see e.g., (5.5), is set at $\lambda = 0.997$. All nodes process the signals in the frequency domain using frame-based processing, with a frame length of 32 ms and a 50%-overlapping Hann window. Notice that all simulations in the following subsections are performed according to the environment given in this subsection.

5.8.2 Estimation of $\mathbf{R}_{\mathbf{Y}}^{-1}$

In this subsection, we compare the two methods presented in Section 5.6 and 5.7, that are, the gossip-based algorithm and the CbDECM algorithm for distributed estimation of the inverse correlation matrix, with a centralized estimator for the inverse correlation matrix.

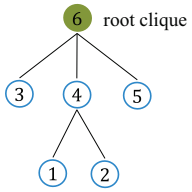


Figure 5.2: Example of a network with tree topology with six cliques.

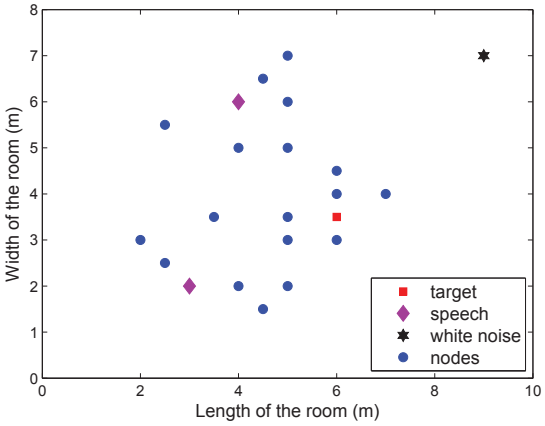


Figure 5.3: WASN with 18 nodes, 3 speech sources and a noise source.

The gossip operations in the gossip-based distributed processing algorithm are based on the clique-based algorithm presented in [19], which performs randomized gossip [11] across non-overlapping cliques.

To quantify the performance of the distributed estimation algorithms, we define the error between the estimated $\mathbf{R}_{\mathbf{Y}}^{-1}$ from the distributed algorithms and the estimation results from the centralized algorithm as

$$\mathrm{Err}_i(k) = 20\log_{10}\frac{1}{F}\sum_{f=1}^{F}\frac{\left\|\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(f,k)-\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(f,k)\right\|_{\mathrm{fro}}}{\left\|\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(f,k)\right\|_{\mathrm{fro}}}. \tag{5.24}$$

where $\mathrm{Err}_i(k)$ is the normalized square error at node i and frame k, and F denotes the number of frequency bins, and $\hat{\mathbf{R}}_{\mathbf{Y},i}^{-1}(k)$ and $\hat{\mathbf{R}}_{\mathbf{Y},c}^{-1}(k)$ denote the estimated inverse correlation matrix obtained from one of the distributed algorithms and the centralized algorithm, respectively. To aggregate information across frequency and time, $\mathrm{Err}_i(k)$

is averaged across time, that is

$$\mathrm{ME}_i = \frac{1}{\tilde{K}} \sum_{k=1}^{\tilde{K}} \mathrm{Err}_i(k), \tag{5.25}$$

with $\tilde{K}$ the number of time-frames.

Fig. 5.4 shows the error Err_1 between the gossip-based estimated inverse correlation matrix and the centralized estimated inverse correlation matrix for various number of randomized gossip iterations, and the error Err_1 between the inverse correlation matrix estimated using the proposed CbDECM method and a centralized estimate. It is observed that the error Err_1 when using the gossip-based distributed algorithm increases across time. This is in line with the convergence analysis given in Section 5.6.2. As expected, the error Err_1 is decreased by increasing the number of iterations. In addition, as the number of iterations is increased, the increase of the error across time slows down. Moreover, we observe that the CbDECM algorithm converges to a very accurate estimate of the inverse correlation matrix after an initial increase of the error (which is due to the floating-point relative accuracy of the Matlab).

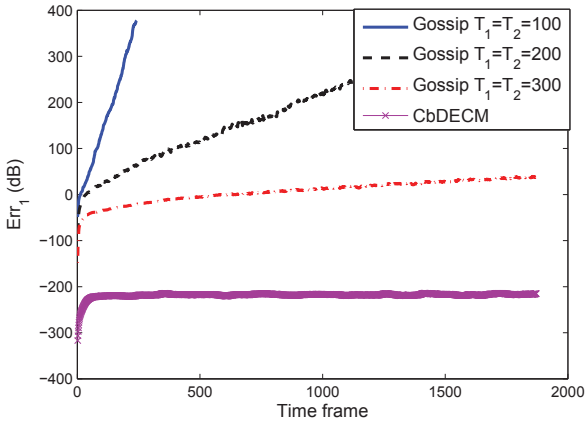


Figure 5.4: The error Err of node 1 with 1 dB input SNR versus time frame.

Fig. 5.5 depicts the estimation performance of the gossip-based distributed estimation algorithm and the proposed CbDECM algorithm as a function of the number of data transmissions per time frame. From Fig. 5.5, we see that the error ME_1 of the gossip-based distributed estimation algorithm is decreased by increasing the number of data transmissions per time frame. In addition, we observe that both the transmission costs and the error of the gossip-based processing for distributed estimation of the inverse correlation matrix are higher than for the proposed CbDECM algorithm.

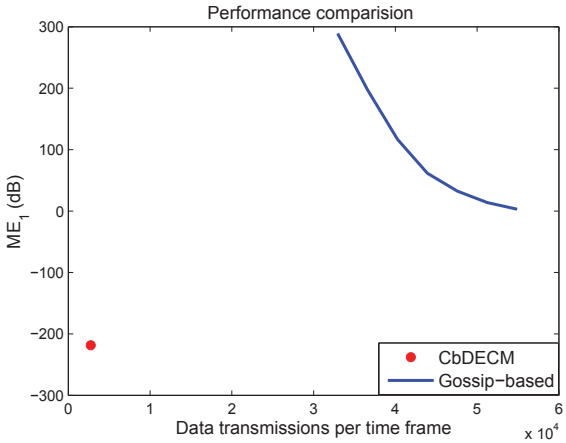


Figure 5.5: The mean error ME of node 1 with 1 dB input SNR versus data transmissions per time frame.

5.8.3 Estimation of a Target Signal

This section discusses the performance when the proposed framework for distributed matrix inverse estimation is used in combination with an MVDR beamformer. The scenario in this experiment considers the case where the source of interest for a node i is private and selected from one of the available sources in the environment depicted in Fig. 5.3. The steering vector $\mathbf{d}_i$ from the desired source of node i to all microphones in the network is assumed to be known only locally. For comparison, we use the distributed delay and sum beamformer (DDSB) presented in [19]. This DDSB is essentially an MVDR defined as in (5.3) where $\mathbf{R_V}$ is assumed to be diagonal. This DDSB thus does not need the full correlation matrix, but is only in need of the diagonal elements, i.e., the noise PSD per microphone. The distributed MVDR beamformer with the CbDECM algorithm and the DDSB are based on similar setup where a clique-based graph is used and the steering vectors are assumed to be known a priori. To compare the distributed beamformers with their centralized versions and evaluate their performance, we also compare to the centralized MVDR (CMVDR) beamformer and the centralized delay and sum beamformer (CDSB) in this experiment. For the DDSB and the CDSB, the noise power spectral density (PSD) tracking algorithm in [22] is used to estimated the noise PSD. Moreover, for a fair comparison we set the number of data transmissions in the DDSB such that it equals to the number of required data transmissions in the CbDECM.

To compare the performance of the proposed CbDECM algorithm and the centralized estimation algorithm for estimating the inverse correlation matrix, we employ the estimated inverse correlation matrices of the CbDECM algorithm and the centralized

algorithm in the MVDR beamformer as

$$\hat{S}_i(k) = \frac{\mathbf{d}_i^H \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)}{\mathbf{d}_i^H \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{d}_i}, \tag{5.26}$$

where $\hat{S}_i(k)$ is the frequency domain DFT coefficient of the beamformer output, and $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}$ is the estimated inverse correlation matrix. Notice that we explicitly mention here the dependency on time-frame k for clarity. Since each node in the network has the estimates of $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)$, $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)$ and $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)$ at time frame k, an alternative approach to incorporate the estimated inverse correlation matrix from frame k instead of $k-1$ as in (5.26) is given by

$$\hat{S}_i(k) = \frac{\mathbf{d}_i^H \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)(\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1))^{-1}\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1)\mathbf{Y}(k)}{\mathbf{d}_i^H \hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)\mathbf{d}_i}. \tag{5.27}$$

Obviously, the computational complexity of (5.27) is higher, since (5.27) requires in addition to compute $\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k)(\hat{\mathbf{R}}_{\mathbf{Y}}^{-1}(k-1))^{-1}$. To distinguish the estimation methods in (5.26) and (5.27), we denote the algorithm in (5.26) by CbDECM$_1$, and the algorithm in (5.27) by CbDECM$_2$.

To evaluate the performance of the proposed algorithm we compute the mean square-error (MSE) between the estimated desired signal and the clean speech signal. The MSE for node i is averaged over all time frames and frequency bins, and is give by

$$\text{MSE}_i = \frac{1}{\tilde{K}F}\sum_{k=1}^{\tilde{K}}\sum_{f=1}^{F}\left|\hat{S}_i(f,k) - S_i(f,k)\right|^2, \tag{5.28}$$

where $S_i(f,k)$ is the frequency domain DFT coefficient of the desired speech signal of node i. In addition, we qualify the speech quality and the speech intelligibility of the estimated desired signal in terms of the segmental SNR and the short-time objective intelligibility measure (STOI) [23], respectively. The segmental SNR for node i is defined as

$$\text{SNR}_{i,\text{seg}} = \frac{1}{\tilde{K}}\sum_{k=1}^{\tilde{K}} 10\log_{10}\frac{\sum_{f=1}^{F}|S_i(f,k)|^2}{\sum_{f=1}^{F}\left|\hat{S}_i(f,k) - S_i(f,k)\right|^2}. \tag{5.29}$$

Figs. 5.6(a) and 5.6(b) show the noise reduction performance of the distributed beamformers and their centralized versions in terms of the segmental SNR and the MSE, respectively, while Fig. 5.6(c) show the instrumental speech intelligibility of the beamformer output. In Figs. 5.6(a) and 5.6(b), we observe that the speech quality of the CbDECM$_1$ is very close to the CMVDR and the CbDECM$_2$. More specifically, the difference between the segmental SNR of the CbDECM$_1$ algorithm and the CMVDR in Fig. 5.6(a) is decreased from 0.3 dB to 0.1 dB with increasing SNR of the noise+target input signal. Similarly, the difference between the MSE of the CbDECM$_1$ algorithm and the CMVDR in Fig. 5.6(b) decreases from 0.3 dB to 0.1 dB. This is reasonable since the CMVDR uses the estimated inverse correlation matrix

of the current time-frame k to estimate the desired signal, while the CbDECM$_1$ algorithm uses the estimated inverse correlation matrix of the previous time frame $k-1$. Fig. 5.6(c) shows that the speech intelligibility in terms of STOI of the CbDECM$_1$ algorithm is identical to the CMVDR and the CbDECM$_2$ algorithm. All three figures in Fig. 5.6 show that both the speech quality and the speech intelligibility of the CbDECM$_2$ reaches the same performance as the CMVDR. This can be explained by the fact that the estimated inverse correlation matrix of both CbDECM based algorithms converge to the estimated inverse correlation matrix of the centralized estimation algorithm. This is consistent with the experiment results given in Fig. 5.4 and Fig. 5.5. In addition, all three figures in Fig. 5.6 show that the improvement of the MVDR beamformers (CMVDR, CbDECM$_1$ and CbDECM$_2$) over the DDSB and the CDSB is increased from 1 to 10 dB in terms of the segmental SNR with increasing input SNR. A similar performance improvement is shown in terms of MSE and STOI. This should not come as a surprise, since the potential improvement of the MVDR beamformer is obtained by taking noise correlation into account. Moreover, all distributed beamformers (DDSB, CbDECM$_1$ and CbDECM$_2$) converge to their centralized versions with sufficient data transmissions per time frame.

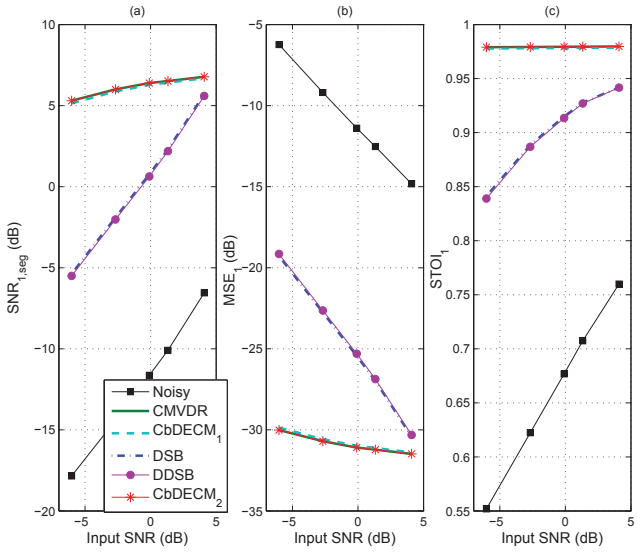


Figure 5.6: (a) The segmental SNR of node 1 versus the global input SNR. (b) The MSE of node 1 versus the global input SNR. (c) The STOI of node 1 versus the global input SNR.

5.9 Conclusions

In this work, we proposed a framework for distributed estimation of the inverse correlation matrix in a randomly connected network. The proposed framework is based on the fact that using recursive exponential smoothing in combination with the Sherman-Morrison formula, the estimation of the inverse correlation matrix can be structured as two rounds of averaging consensus problems. We first investigated the use of gossip processing for distributed estimation of the inverse correlation matrix, since gossip processing is an well known approach for solving averaging consensus problems. However, due to the fact that the inverse correlation matrix is updated recursively across time, the convergence error between the gossip-based estimated correlation matrix and the centralized estimated correlation matrix accumulates across time. In addition, we therefore also proposed a clique-based distributed algorithm to eliminate this convergence error. This algorithm is referred to as clique-based distributed estimation of the inverse correlation matrix (CbDECM).

The proposed CbDECM is analyzed and compared with a centralized estimator in terms of transmission costs. The comparison is done in a scenario, where the source of interest of a user in the network is considered to be privacy preserving by hiding the information of the steering vectors. In such a privacy preserving scenario, each user in the network is assumed to know the steering vectors locally, and thus each user requires an estimate of the inverse correlation matrix.

Simulation results with the gossip-based estimation approach showed that the convergence error of the estimated inverse correlation matrix increases across time, while the CbDECM algorithm converges to the same estimate as the centralized estimator. Moreover, experiments on the comparisons between the proposed CbDECM algorithm and the distributed delay and sum beamformer in referenced literature illustrated the performance improvement of the proposed CbDECM algorithm by incorporating noise correlation. Compared with other distributed adaptive beamformers, the proposed distributed MVDR beamformer in this article make use of prior knowledge on the steering vectors. For future research it will be interesting to investigate how to estimate the steering vectors in a distributed way while still preserving the users' privacy with respect to his source of interest.

References

[1] R. C. Hendriks, T. Gerkmann, and J. Jensen. *DFT-Domain Based Single-Microphone Noise Reduction for Speech Enhancement: A Survey of the State of the Art*. Morgan & Claypool, 2013.

[2] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[3] J. Benesty, M. M. Sondhi, and Y. Huang (Eds). *Springer handbook of speech processing*. Springer, 2008.

[4] A. Bertrand and M. Moonen. Distributed node-specific LCMV beamforming in wireless sensor networks. *IEEE Trans. Signal Process.*, 60(1):233–246, Jan. 2012.

[5] S. Markovich Golan, S. Gannot, and I. Cohen. Distributed GSC beamforming using the relative transfer function. In *European Signal Processing Conference*, Bucharest, Aug. 2012.

[6] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[7] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer for speech enhancement in wireless sensor networks via randomized gossip. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 4037–4040, 2012.

[8] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 7005–7009, Canada, May 2013.

[9] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy preserving distributed beamforming based on homomorphic encryption. In *Proc. European Signal Proc. Conf. Eusipco*, pages 7005–7009, Marrakesh, Morocco, 2013.

[10] C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security.*, 2007:1–10, Jan. 2007.

[11] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[12] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng, and W. B. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[13] G. Zhang and R. Heusdens. Linear coordinate-descent message-passing for quadratic optimization. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 2005–2008, 2012.

[14] J. Sherman and W. J. Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127, 1950.

[15] S. Gannot, D. Burshtein, and E. Weinstein. Signal enhancement using beamforming and nonstationarity with applications to speech. *IEEE Trans. Signal Process.*, 49(8):16141626, 2001.

[16] J. Liu, S. Mou, A. S. Morse, B. D. Anderson, and C. Yu. Deterministic gossiping. *Proceedings of the IEEE*, 99(9):1505–1524, 2011.

[17] H. L. Van Trees. *Detection, Estimation, and Modulation Theory, Part IV, Optimum Array Processing*. John Wiley and Sons, New York, 2002.

[18] R. Bovenkamp, F. Kuipers, and P. V. Mieghem. Gossip-based counting in dynamic networks. *Networking 2012.*, pages 404–417, 2012.

[19] Y. Zeng, R. C. Hendriks, and R. Heusdens. Clique-based distributed beamforming for speech enhancement in wireless sensor networks. In *Proc. European Signal Proc. Conf. (EUSIPCO)*, Marrakesh, Morocco, 2013.

[20] H. Chen, A. Campbell, B. Thomas, and A. Tamir. Minimax flow tree problems. *Networks.*, 54:117–129, Mar. 2009.

[21] J. S. Garofolo. DARPA TIMIT acoustic-phonetic speech database. *National Institute of Standards and Technology (NIST)*, 1988.

[22] R. C. Hendriks, R. Heusdens, and J. Jensen. MMSE based noise PSD tracking with low complexity. In *IEEE Int. Conf. Acoust, Speech, Signal Processing*, pages 4266–4269, 2010.

[23] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen. An algorithm for intelligibility prediction of time-frequency weighted noisy speech. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(7):2125–2136, Sep. 2011.

Chapter 6

On Clock Synchronization for Multi-microphone Speech Processing in Wireless Acoustic Sensor Networks

This chapter is based on the publication "On Clock Synchronization for Multi-microphone Speech Processing in Wireless Acoustic Sensor Networks", by Y. Zeng, R. C. Hendriks and N. D. Gaubitch in the Proceedings of *IEEE Int. Conf. Acoust., Speech, Signal Processing (ICASSP)*, Brisbane, Australia, April 2015.

6.1 Introduction

Wireless acoustic sensor networks (WASNs) have been proposed for speech enhancement by means of multi-microphone noise reduction [1] [2] [3] [4]. Compared with conventional multi-microphone signal processing systems, where all microphones are connected via wired links, and the observed signals of all microphones are gathered and processed in a fusion center, multi-microphone signal processing in WASNs aims to distribute computational load across all sensors, while reaching comparable performance as conventional systems. Multi-microphone noise reduction algorithms such as beamforming, heavily depend on timing information as they usually employ the delay that is experienced when an acoustic signal is observed at different positions. However, in WASNs, each node usually has its own processor with an independent internal clock. Employing multi-microphone noise reduction algorithms, in practice, requires that these clocks are synchronized. Most of the multi-microphone signal processing algorithms for WASNs (see e.g., [1] [2] [3] [4]) are based on the often implicit assumption that the internal clocks are synchronized. An unsynchronized clock can cause drift of time differences of the observed signal at the different nodes, and, as a consequence degrade the performance of the multi-microphone noise reduction algorithm.

Since clock synchronization is an important aspect for signal processing in wireless sensor networks (WSNs), several algorithms addressing this issue (not specifically for speech enhancement) have been presented [5] [6] [7] [8]. However, most studies on beamforming/speech enhancement in WASNs neglect the clock synchronization problem and simply assume the clocks to be synchronized. Studies on the required precision and the applicability of such algorithms in terms of data transmissions and robustness in practical scenarios is generally lacking. In this work we therefore present an in-depth comparison of several clock synchronization algorithms for distributed speech enhancement. The consequence of an unsynchronized clock is a clock skew and a clock offset. To focus this work, we will only consider the clock skew, which causes signal drift and results in a poor performance of speech enhancement algorithms. Many algorithms for clock skew compensation employ a series of timing message transmissions, e.g., [6] [7]. In [6], a joint ranging and clock synchronization (JCS) algorithm is proposed to estimate relative clock skews, clock offsets and pairwise distances in a WSN using a single clock reference. This algorithm requires that the node with the reference clock serves as a central processor, connected with all other nodes in the network. Specifically, with the JCS, relative clock skews and offsets among nodes are estimated by minimizing the least squares norm of the measurement noise of the time-stamps. The gossip-based clock synchronization (GbCS) algorithm in [7] is based on time stamps in combination with the randomized gossip algorithm [9]. Unlike the JCS algorithm where the clocks of all nodes are synchronized with respect to the clock of a reference node, the GbCS algorithm synchronizes them with a virtual clock. Thus, the GbCs algorithm synchronizes the clocks in a distributed way without the requirements of a reference clock or a specific network topology. The accuracy of these time-stamp based algorithms is, generally, proportional to the number of timing message transmissions.

Another class of clock synchronization algorithm is based on the observed sig-

nal, such as the blind sampling-rate offset estimation (BSrOE) algorithm in [8] and the blind synchronization algorithm in [10]. Assuming that there is a reference node in the WASN, the BSrOE estimates relative clock skews using the phase drift in the coherence between the observed signals of two communicating nodes. Similar to the JCS, the BSrOE requires that the node with the reference clock serves as a central processor. Although the algorithms in [6], [7] and [8] can all synchronize clocks in WSNs, they have different advantages and disadvantages when used in WASNs for practical multi-microphone signal processing, as they are based on different assumptions and frameworks.

In this work we present a study of the effect of clock synchronization problems on multi-microphone signal processing where each node has an individual clock. We perform theoretical and experimental investigations of the effects of clock synchronization on the delay-and-sum beamformer (DSB) and the minimum variance distortionless response (MVDR) beamformer using three state-of-the art algorithms (i.e., the JCS, the GbCS and the BSrOE). In particular we analyze the communication costs of the three algorithms and investigate their robustness to noise on the parameters used to synchronize the clocks. In addition, we discuss the advantages and drawbacks of the three clock synchronization algorithms in view of their theoretical frameworks and simulation results.

The remainder of this article is organized as follows. In Section 6.2 we state the problem and introduce the notation that we will use throughout this article. In Section 6.3 we analyze the effects of clock synchronization problems on beamforming technologies. Then, in Section 6.4 we give a brief overview of three different clock synchronization algorithms, which are the JCS, the GbCS and the BSrOE. In Section 6.5 we illustrate the estimation accuracy of the three clock synchronizations and evaluate their effect on the MVDR beamformer in terms of speech quality and speech intelligibility. Finally, in Section 6.6 conclusions are drawn.

6.2 Problem Statement and Notation

Consider a WASN comprising N nodes randomly distributed in a noisy environment, where each node is driven by it's own processor with an internal clock and contains one microphone. Let $y_i(t)$ denote a continuous-time signal observed at node i. Assuming that the signal $y_i(t)$ consists of a target source signal $x_i(n)$ degraded by additive noise $v_i(n)$, a common data model of $y_i(t)$ is given by

$$y_i(t) = x_i(t) + v_i(t), \tag{6.1}$$

The challenge for noise reduction algorithms is to estimate the target signal from the noisy observations. With a conventional microphone array, the speech signal can be estimated using beamforming methods, such as the DSB or the MVDR beamformer, since all microphones have the same clock and sampling rate. However, in a WASN, each node is equipped with an independent clock oscillator. Clock differences between nodes are therefore inevitable. Let t_i denote the local clock reading at node i, given by

$$t_i = \alpha_i t + \beta_i, \tag{6.2}$$

where t is the global time or the local time of a reference node, α_i is the clock skew and β_i is the clock offset. We assume the clock offset parameter to be known and we concentrate on the clock skew. The clock model t_i in (6.2) can then be simplified to

$$t_i = \alpha_i t. \tag{6.3}$$

Without loss of generality, we assume that the first node is the reference node (e.g., $t_1 = t$). Based on the time model in (6.3), the sampling rate at node i is given by

$$f_{s,i} = \alpha_i f_s, \tag{6.4}$$

where $f_{s,i}$ is the sampling rate at node i, and f_s is the sampling rate at a reference node (e.g., $f_s = f_{s,1}$). Let $y_i[n]$ denote the discrete-time observed signal at time-sampling index n. The discrete-time signal $y_i[n]$ can be obtained by sampling the continuous-time signal $y_i(t)$ at time $\frac{n}{f_{s,i}}$, i.e.,

$$y_i[n] = y_i\left(n/f_{s_i}\right), \;\; t = n/f_{s_i} \text{ and } -\infty < n < +\infty. \tag{6.5}$$

Similarly, $x_i[n]$ and $v_i[n]$ denote the discrete-time speech signal and noise signal, respectively.

Equation (6.5) indicates that different sampling rates cause drift of time difference between the observed digital signals. This problem can be solved by synchronizing sampling rates of all nodes, which can be realized by synchronizing clock skews of all nodes. In this work, we study the problem of clock skew synchronization and investigate its impact on multi-microphone noise reduction. Moreover, we investigate and compare three different algorithms for clock skew synchronization and study their implications on multi-microphone noise reduction.

6.3 Analysis of the Clock Synchronization Problem for Beamforming Technologies

In this section, we analyze the effect and importance of clock synchronization on beamforming technologies. To facilitate a simple and clear insight into the problem, we use a synthetic signal. The multi-microphone noise reduction algorithm that we use in these illustrations is the DSB, since this is the most simple way to exemplify the consequences of having non-synchronized clock skews. Later on, in Section 6.5 we will use the MVDR beamformer and speech signals to compare the impact of different clock synchronization algorithms.

Let $w(\cdot)$ denote a window function of length J and let K be the window shift. As beamforming algorithms are usually conducted in the short-time discrete Fourier transform (DFT) domain, signals are windowed and transformed into the frequency domain by applying a DFT, i.e.,

$$Y_i(f,m) = \sum_{n=mK+1}^{mK+J} y_i(n) w(n-mK) e^{-j\omega_f (n-mK)}, \tag{6.6}$$

where $Y_i(f,m)$ is the noisy DFT coefficient at frequency-bin index f and discrete-time frame index m, and $\omega_f = 2\pi f/J$ is the discrete frequency variable at frequency-bin index f. Similarly, $X_i(f,m)$ and $V_i(f,m)$ denote the speech DFT coefficient and the noise DFT coefficient, respectively. Consider a single target speech source in the network, the speech DFT coefficient $X_i(f,m)$ is given by

$$X_i(f,m) = d_i(f,m)S_i(f,m),$$

where $d_i(f,m)$ is the acoustic transfer function (ATF), and $S_i(f,m)$ is the clean speech at the target location, both with sampling rate $f_{s,i}$. To estimate the clean speech signal, the noisy DFT coefficients $Y_i(f,m)$ can be stacked into a vector, say $\mathbf{Y}(f,m) = [Y_1(f,m), ...Y_N(f,m)]^T$, with $[\cdot]^T$ the transposition of a vector or a matrix, followed by filtering with $\mathbf{W}(f,m)$ (i.e., $\hat{S}_1(f,m) = \mathbf{W}^H(f,m)\mathbf{Y}(f,m)$). However, since all nodes have different sampling rates, the beamformer performance will be degraded depending on the differences between sampling rates.

Consider a WASN with two nodes, each node with one microphone. The sampling-rate of node 1 is $f_{s_1} = f_s = 16$ kHz (i.e., the reference node) and the sampling-rate of node 2 is $\alpha_2 f_s$. To assess the performance of the DSB versus the clock skew of node 2, we use the spatial directivity pattern $Q(\omega)$ of the DSB, that is

$$Q(\omega) = \frac{\mathbf{d}^H\tilde{\mathbf{d}}}{\mathbf{d}^H\mathbf{d}}, \tag{6.7}$$

with $(\cdot)^H$ Hermitian transposition, $\tilde{\mathbf{d}} = [d_1(f,m), d_2(f,m)]^T$ the ATF under sampling-rate mismatch, and $\mathbf{d}$ the ATF without sampling-rate mismatch. If there is a sampling-rate mismatch, and $\mathbf{d} \neq \tilde{\mathbf{d}}$, then $Q(\omega)$ measures the amount of mismatch, which reflects the amount of distortion that the clean signal will undergo after processing by the beamformer. In a free-field scenario without damping (i.e., $|d_i| = 1$), we then have $\mathbf{d} = \left[e^{-j\omega\frac{l_1}{c}f_s}, e^{-j\omega\frac{l_2}{c}f_s}\right]^T$ and $\tilde{\mathbf{d}} = \left[e^{-j\omega\frac{l_1}{c}f_s}, e^{-j\omega\frac{l_2}{c}f_{s,2}}\right]^T$, where l_i is the distance from the source to the ith node, and $c = 340$ m/s is the speed of sound. Then, $Q(\omega)$ can be expressed as

$$Q(\omega) = \frac{1}{2} + \frac{1}{2}e^{j\omega\frac{l_2}{c}(1-\alpha_2)f_s} = \cos\left(\omega\frac{l_2}{2c}(1-\alpha_2)f_s\right)e^{j\omega\frac{l_2}{2c}(1-\alpha_2)f_s}. \tag{6.8}$$

Notice that damping can easily be included, but is left out here for simplicity.

Equation (6.8) shows that $|Q(\omega)|$ is periodic as a function of the clock skew, see also Fig. 6.1 for a plot of $|Q(\omega)|$. Ideally, for the case that there is no clock skew (i.e., $\alpha_i = 1, \forall\, i$), $Q(\omega) = 1$. Obviously, when there is clock skew, $|Q(\omega)|$ deviates from 1 and distortions are introduced in the spatial directivity pattern. That is, the beamformer response in the target direction may be suppressed depending on α. This dependency of $Q(\omega)$ on α is periodic as is clear from (6.8) and also shown in Fig. 6.1.

To further assess the distortions introduced in the estimated clean DFT coefficients, we investigate the output of the DSB when applied to the clean input only, i.e., $V_i = 0, \forall i$. In this case, we set the clean target signal to a sinusoidal signal given by the expression $s(t) = \cos(2\pi\nu t)$ with $\nu = 1250$ Hz. Sampling this signal with

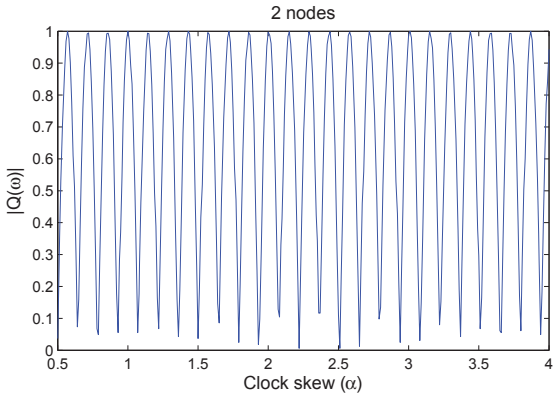


Figure 6.1: The distortion $|Q(\omega)|$ versus clock skew

the sampling frequency of node 1 and node 2 (i.e., $f_{s_1} = f_s$ and $f_{s_2} = \alpha_2 f_s$), leads to $s_1[n] = \cos(2\pi \frac{\nu}{f_s})$ and $s_2[n] = \cos(2\pi \frac{\nu}{\alpha_2 f_s} n)$ with the two different frequencies ν and ν/α, respectively. Let $S_1(f,m)$ and $S_2(f,m)$ denote the DFT of a windowed frame of $s_1[n]$ and $s_2[n]$, respectively. Stacking these DFT coefficients in a vector, and including the delays τ_1 and τ_2 due to the signal propagation over distances l_1 and l_2, respectively, we get $\mathbf{X}(f,m) = \left[S_1(f,m)e^{-j\omega_f \tau_1}, S_2(f,m)e^{-j\omega_f \tau_2}\right]^T$, with $\tau_1 = l_1 f_{s_1}/c$ and $\tau_2 = l_2 f_{s_2}/c$.

Let $\hat{S}(f,m)$ denote the DSB output when applying the beamformer to the clean signals only (i.e., to $\mathbf{X}(f,m)$). When there is no clock skew, the output equals the clean signal, $\hat{S}(f,m) = \frac{1}{2}\left(S_1(f,m) + S_2(f,m)\right)$. However, the DSB output under clock skew is given by

$$\hat{S}(f,m) = \frac{\mathbf{d}^H \mathbf{X}(f,m)}{\mathbf{d}^H \mathbf{d}} = \frac{1}{2}\left(S_1(f,m) + S_2(f,m)e^{j2\pi f(1-\alpha_2)l_2/c}\right). \quad (6.9)$$

Two effects become apparent. What should be the average of the DFT coefficient $\hat{S}(f,m)$ of two windowed sinusoids with similar frequency and compensated delay such that they constructively add, becomes the sum of the DFT coefficients of two windowed sinusoids with a) two different frequencies, and b) a delay with respect to each other that is not correctly compensated.

As an example, Fig. 6.2 shows the value $|\hat{S}(f,m)|$ with and without clock skew for a fixed frequency bin $f = 20$ (chosen as the bin with center frequency closest to $\nu = 1250$) across time frames. The blue dashed line shows the estimated $|\hat{S}(f,m)|$ when there is no clock skew ($\alpha_1 = \alpha_2$) and the red solid line shows for a fixed $\alpha_2 = 1.0032$. The distortion $|\hat{S}(f,m)|$ varies periodically across time and that the distortion in $\hat{S}(f,m)$ is upper and lower bounded.

The above analysis and simulations show that asynchronous clocks in WASNs can severely degrade the performance of beamformers. First, the delay compensations by

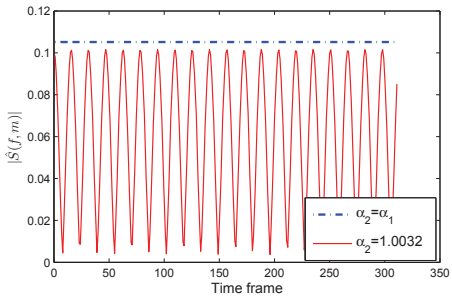


Figure 6.2: The distortion $|\hat{S}(f,m)|$ versus time frames

$\mathbf{d}$ are incorrect, leading to an undesired beamformer response. Secondly, at nodes with clock skew $\alpha_i \neq 1$, the signal gets translated to another frequency. Depending on α_i this is audibly perceived as the sum of two speech signals that are not aligned with respect to each other and have different sampling frequencies. A solution is to perform clock synchronization and/or sampling-rate offset compensation.

6.4 Clock Synchronization

In this section, we give a brief overview of three clock synchronization algorithms for wireless sensor networks (WSNs). The three algorithms are the JCS [6], the GbCS [7] and the BSrOE [8]. Specifically, the JCS and GbCS are time-stamp based algorithms. Clock synchronization in these algorithms is based on local time-stamp communication, while the BSrOE is a signal based algorithm, where clock skews are estimated based on the phase drift in the coherence between the observed signals. The three algorithms make use of different underlying principles and rely on different assumptions. In this section we will describe the underlying principle of each of these methods. Although each algorithm has its own requirements (e.g., the number of transmissions, the network topology, centralized or decentralized processing), they can all be used to solve the sampling-rate synchronization problem in WASNs albeit at different costs and requirements.

6.4.1 JCS

The JCS algorithm proposed in [6] is based on a two-way communication scheme, where two neighboring nodes communicate with each other during each time-slot. We give here an overview of the pairwise synchronization, while for further information about this algorithm, the reader is referred to reference [6].

Consider two nodes (i, j) in a connected WSN. Node i becomes active and sends a message to node j and node j sends a response message back to node i. Both nodes i and j record the transmission time and the reception time. Nodes i and j communicate

with each other K times and record the transmission and reception times as

$$\mathbf{t}_{ij} = \left[t_{ij,\mathrm{s}}^{(1)}, t_{ij,\mathrm{r}}^{(1)}, t_{ij,\mathrm{s}}^{(2)}, \cdots, t_{ij,\mathrm{r}}^{(K)}\right]^T$$

and

$$\mathbf{t}_{ji} = \left[t_{ji,\mathrm{r}}^{(1)}, t_{ji,\mathrm{s}}^{(1)}, t_{ji,\mathrm{r}}^{(2)}, \cdots, t_{ji,\mathrm{s}}^{(K)}\right]^T$$

where $t_{ij,\mathrm{s}}^{(k)}$ is the transmission time at node i for sending the kth message to node j, $t_{ij,\mathrm{r}}^{(k)}$ is the response time at node i for receiving kth message from node j, and $\mathbf{t}_{ij}$ is the $2K \times 1$ dimensional vector with the transmission and response times measured at node i. Let η_{ij} denote the propagation time between nodes i and j. Based on the time model in (6.3), the relation between the transmission and response time can be written as [6]

$$t_{ij,\mathrm{s}}^{(k)} = \alpha_i \left[\alpha_j^{-1}(t_{ji,\mathrm{r}}^{(k)} + g_{j,\mathrm{r}}^{(k)}) - \eta_{ij}\right] + g_{i,\mathrm{s}}^{(k)}, \tag{6.10}$$

and

$$t_{ij,\mathrm{r}}^{(k)} = \alpha_i \left[\alpha_j^{-1}(t_{ji,\mathrm{s}}^{(k)} + g_{j,\mathrm{s}}^{(k)}) + \eta_{ij}\right] + g_{i,\mathrm{r}}^{(k)}. \tag{6.11}$$

where $\left\{g_{j,\mathrm{r}}^{(k)}, g_{i,\mathrm{s}}^{(k)}, g_{j,\mathrm{s}}^{(k)}, g_{i,\mathrm{r}}^{(k)}\right\}$ are additive Gaussian independent identical distributed random variables that model the noise (error) on the time recording. Assuming that node i is the reference node with $\alpha_i = 1$, the parameters α_j, and η_{ij} can then be estimated by solving the following least squares algorithm problem [6]

$$\min_{\mathbf{p}_j} \|\mathbf{T}_{ji}\mathbf{p}_j - \mathbf{t}_{ij}\|_2^2, \tag{6.12}$$

where

$$\mathbf{T}_{ji} = [\mathbf{t}_{ji}, \mathbf{e}]$$
$$\mathbf{p}_j = [\alpha_j, \eta_{ij}]^T$$

with $2K \times 1$ dimensional vector $\mathbf{e} = [-1, 1, -1, \cdots, 1]^T$. The estimate of $\mathbf{p}_j$ can be obtained as

$$\hat{\mathbf{p}_j} = \left(\mathbf{T}_{ji}^T\mathbf{T}_{ji}\right)^{-1}\mathbf{T}_{ji}^T\mathbf{t}_{ij}. \tag{6.13}$$

This pairwise estimation method can be extended to the entire network as long as the network has a special structure such as, a fully connected or star connected network, where the clock of one of the nodes serves as a reference clock. The reference node must be connected with all other nodes in the network, since the relative clock skews $\frac{\alpha_j}{\alpha_i}$ are estimated with respect to its clock. Furthermore, if extended to clock synchronization for the entire network, a common fusion center is assumed in order to calculate the extended version of Eq. (6.13).

6.4.2 GbCS

Given the local clock model in (6.3), the local clock reading at node j at time t can be expressed in terms of the local clock of node i as

$$t_j = \frac{\alpha_j}{\alpha_i}t_i. \tag{6.14}$$

where $\frac{\alpha_j}{\alpha_i}$ is the relative clock skew between node j and node i. To estimate $\frac{\alpha_j}{\alpha_i}$, the GbCS algorithm makes two assumptions on the clock-reading procedure. First, there is no noise during clock reading. Secondly, clock reading of two communicating nodes is performed instantaneously, since the propagation time between two communicating nodes is assumed to be negligible. It remains to be seen to which extend these assumptions are valid in practice. In Section 6.5 we will test this by introducing noise on the clock reading procedure. Based on those two assumptions, the relative skew $\frac{\alpha_j}{\alpha_i}$ between nodes i and j can be estimated as

$$\frac{\alpha_j}{\alpha_i} = \frac{t_{ji}^{(1)} - t_{ji}^{(2)}}{t_{ij}^{(1)} - t_{ij}^{(2)}}. \tag{6.15}$$

Since α_i and α_j are unknown, it is impossible to directly estimate α_i and α_j. To achieve clock synchronization, the GbCS algorithm uses a clock skew compensation parameter s_i. The local clock of node i can then be compensated as

$$\hat{t}_i = s_i t_i = s_i \alpha_i t, \tag{6.16}$$

where $\hat{t}_i$ is the updated local clock of node i. The randomized gossip algorithm is performed to achieve the consensus of $\hat{t}_i$ at all nodes. In each time-slot u, two neighboring nodes i and j are randomly selected. To update their synchronization of the clock skew we first write

$$s_i(u+1)\alpha_i = \frac{1}{2} s_i(u)\alpha_i + \frac{1}{2} s_j(u)\alpha_j. \tag{6.17}$$

Dividing left and right hand side with α_i and combining the estimated relative clock skew $\frac{\alpha_j}{\alpha_i}$ in (6.15) with (6.17), it follows that

$$s_i(u+1) = \frac{1}{2} s_i(u) + \frac{\alpha_j}{2\alpha_i} s_j(u). \tag{6.18}$$

Similar as node i, the skew compensation parameter s_j of node j can be updated as

$$s_j(u+1) = \frac{\alpha_i}{2\alpha_j} s_i(u) + \frac{1}{2} s_j(u). \tag{6.19}$$

Given any initialization of s_i, clock skews of all nodes are guaranteed to converge to the average value $\frac{1}{N}\sum_{i=1}^{N} s_i(0)\alpha_i$ as long as each pair of neighboring nodes in a connected network gossips frequently enough. Thus, the updated local clocks of all nodes are synchronized with $\frac{1}{N}\sum_{i=1}^{N} s_i(0)\alpha_i t$, when a sufficient number of iterations are used in the GbCS algorithm. In Section 6.4.4, we analyze the communication cost of this algorithm.

6.4.3 BSrOE

In [8], it was proposed to perform a blind sampling-rate offset estimation based on the phase drift in the coherence between two node signals, which are sampled at different

sampling rates. Considering a connected WASN, where all nodes are placed in a noisy speech environment, two neighboring nodes communicate with each other by exchanging their observed noisy speech signals. It is assumed that the noise-only segments are detected using a voice activity detector (VAD) and used to estimate the sampling-rate difference between two nodes.

Assuming that the sampling rate of the first node as the reference sampling rate (e.g., $f_{s,1} = f_s$), the relative sampling-rate offset between the reference node and a node i is given by

$$e_{1i} = \frac{f_{s,i} - f_s}{f_s}. \tag{6.20}$$

Let $\gamma_{1i}[f]$ denote cross-coherence between the reference node 1 and node i at frequency bin f, which is defined as

$$\gamma_{1i}[f] = \frac{R_{1i}[f]}{\sqrt{R_1[f] R_i[f]}}, \tag{6.21}$$

where $R_{1i}[f]$ is the cross-spectrum of nodes 1 and i, and $R_1[f]$ and $R_i[f]$ are the auto-spectrum of nodes 1 and i, respectively. To estimate the relative sampling-rate offset e_{1i}, the microphone signals are first segmented to P segments. Then, the Welch method is applied to estimate the cross-spectrum and the auto-spectrum for f_{max} frequency bins and the estimation of the relative sampling-rate offset e_{1i} is obtained as

$$\hat{e}_{1i} = \frac{1}{f_{\text{max}}} \sum_{f=1}^{f_{\text{max}}} \frac{F}{2\pi L f} \angle \left\{ \frac{1}{P-1} \left(\sum_{p=1}^{P-1} \frac{\hat{\gamma}_{1i}^{p}[f]}{\hat{\gamma}_{1i}^{p-1}[f]} \right) \right\}, \tag{6.22}$$

where L is the number of samples in each segment, f_{max} is the maximum frequency bin, which is used to guarantee that the phase difference between $\gamma_{1i}^{p}[f]$, and $\gamma_{1i}^{p-1}[f]$ is bounded in the range $[-\pi\ \pi]$, and is defined as [8]

$$f_{\text{max}} = \frac{F}{2L e_{\text{max}}}, \tag{6.23}$$

with e_{max} the bound of relative sampling-rate offsets. The estimation accuracy of e_{1i} depends on the value of e_{max}. When $e_{\text{max}} >> |e_{1i}|$, few frequency bins contribute in the estimation, and the estimation of e_{1i} will be incorrect if $e_{\text{max}} < |e_{1i}|$ [8]. In [8], the authors assume that the WASN is a fully connected network and a node in the network is selected as a fusion center, gathering all microphone signals and synchronizing all clocks with a selected reference clock in the network. Although the algorithm assumes that the WASN is a fully connected network, it can also synchronize clock skews in a non-fully connected network by using other nodes as relay stations to send the data to a central processor (which is the reference node). This algorithm may result in a large number of data transmissions, in particular for non-fully connected networks since all signals need to be sent to this central processor.

6.4.4 Communication Cost Analysis

To analyze communication cost of the three clock synchronization algorithms, we define a data transmission as the sending of a scalar value from one node to another.

In both the JCS and GbCS, clocks are synchronized by exchanging time information, which is a scalar value of the time-stamp. In a fully connected network with N nodes, the number of data transmissions of the JCS algorithm is given by

$$T_J = 2K(N-1), \tag{6.24}$$

since all $N-1$ pairs of neighboring nodes (each pair includes the reference node and one other node) communicate K times with 2 transmissions each time.

The number of data transmissions of the GbCS algorithm T_G is

$$T_G = 4C, \tag{6.25}$$

with C number of iterations. At each iteration, two neighboring nodes communicate a time message and clock skew compensation parameter. This means twice the transmission of two variables per iteration.

In the BSrOE, all $N-1$ nodes (all nodes except the reference node) send the DFT coefficients of their observed signals to the reference node, which serves as the central processor. Thus, the data transmission of the BSrOE can be computed as

$$T_B = (N-1)Pf_{\text{max}}, \tag{6.26}$$

where there are P segments and f_{max} frequency bins per segment. Notice that the required number of data transmissions of the JCS and BSrOE given in (6.24) and (6.26), respectively, consider only a fully connected network. More data transmissions are required for clock synchronization when used in non-fully connected networks as the time message information or microphone signals need to be sent to the central processor using relay nodes. Only the data transmissions of the GbCS given in (6.25) is directly applicable to randomly connected networks.

6.5 Simulations

In this section, we study the performance of the three clock synchronization algorithms described in Section 6.4 and evaluate their effect on the performance of the MVDR beamformer in terms of instrumental speech quality and speech intelligibly metrics. The comparison is first performed in an ideal scenario without noise on the information required by each algorithm, followed by a practical scenario with additive noise.

6.5.1 Simulation Environment and Performance Measurements

We simulate a WASN with five fully connected nodes, and consider a free-field scenario. Thus, the steering vector $\mathbf{d}$ is determined by gain and delay values. The node positions relative to the sources are shown in Fig. 6.3. The speech source consists of a 30 seconds speech signal sampled at 16 kHz originating from the Timit [11] database, and the noise source is a babble noise signal. All nodes in the WASN first synchronize their clocks using one of the three algorithms, and then process the signals in the frequency domain using a frame-based MVDR beamformer, with a frame

length of 32 ms and a 50%-overlapping Hann window. The following parameters are used in the BSrOE. The Welch method is used with a DFT size of 4096 and 75% overlap. Each segment consists of $L = 16000$ samples and $P = 32$ segments with 50% overlap are used to estimate the sampling-rate offset, which is bounded by $e_{\max} = 800$ ppm with ppm $= 10^{-6}$. The clock skews of the five nodes are set to $\alpha = [1, 1.0001, 1.0002, 1.0003, 1.0004]^T$. Notice that all simulations in this section are based on the scenario given in this subsection.

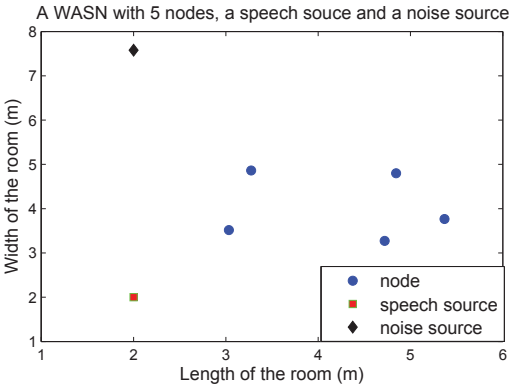


Figure 6.3: WASN with 5 nodes, a speech source and a noise source.

To assess the estimation accuracy of the clock skew, we define the mean square error (MSE) between the estimated clock skews $\hat{\alpha}_i$ of all nodes and the reference clock skew α_{ref} as

$$\text{MSE}_{\text{T}} = 10\log_{10}\frac{1}{N}\sum_{i=1}^{N}\left|\hat{\alpha}_i - \alpha_{\text{ref}}\right|^2. \tag{6.27}$$

The reference clock skew in the JCS algorithm and the BSrOE algorithm is the clock skew of the first node, while the reference clock skew in the GbCS algorithm is the average value $\frac{1}{N}\sum_{i=1}^{N} s_i(0)\alpha_i$. In order to evaluate the effect of the clock synchronization algorithms on the performance of the MVDR beamformer, we use the segmental signal-to-noise ratio (SNR_{seg}) to assess the speech quality of the MVDR beamformer and the short-time objective intelligibility measure (STOI) [12] to assess the speech intelligibility of the MVDR beamformer. The SNR_{seg} of node i is averaged over all time frames and defined as

$$\text{SNR}_{\text{seg}} = \frac{1}{M}\sum_{m=1}^{M} 10\log_{10}\frac{\sum_{f=1}^{F}|S(f,m)|^2}{\sum_{f=1}^{F}\left|\hat{S}_i(f,m) - S(f,m)\right|}, \tag{6.28}$$

where M is the number of time-frames, F denotes the number of frequency bins, and $\hat{S}_i(f,m)$ is the frequency domain DFT coefficient of the MVDR beamformer output,

calculated as

$$\hat{S}_i(f,m) = \frac{\mathbf{d}^H \mathbf{R}_{\mathbf{YY}}^{-1} \mathbf{Y}}{\mathbf{d}^H \mathbf{R}_{\mathbf{YY}}^{-1} \mathbf{d}}. \tag{6.29}$$

As reference signal in STOI and SNR_{seg} we use the clean signal sampled by the reference clock at the reference node. For notational convenience, we denote the MVDR with perfect clock synchronization by C-MVDR, the MVDR beamformer without clock synchronization by E-MVDR, the MVDR beamformer with the JCS by J-MVDR, the MVDR beamformer with the GbCS by G-MVDR, and the MVDR beamformer with the BSrOE by B-MVDR.

6.5.2 Ideal Clock Synchronization

We begin with the assumption that there is no measurement noise on the time-stamp in the JCS and GbCS, and the observed signal used in the BSrOE is a babble noise-only. This follows the ideal circumstances described in the original papers.

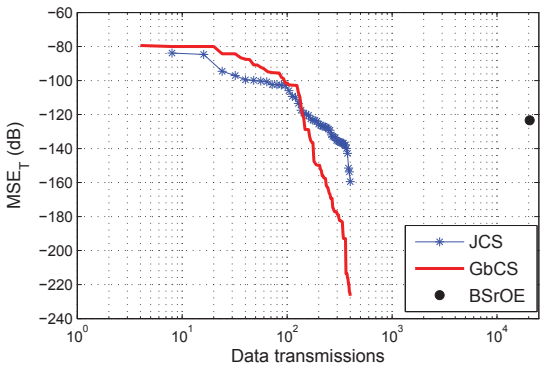


Figure 6.4: The MSE versus number of transmissions.

Figure 6.4 shows that all three algorithms can reach the same accuracy of clock synchronization in terms of MSE_{T} with enough data transmissions. The estimation accuracy of the clock skew in the JCS and GbCS is increased with increasing number of data transmissions. Further, the BSrOE needs more data transmissions to reach a performance similar to that of the JCS and GbCS.

Figure 6.5 shows the effect of clock synchronization on the MVDR beamformer. In Fig. 6.5(a), we see that the SNR_{seg} of the E-MVDR output is even lower than those of the input noisy signal for global input SNRs larger than 2 dB. In Fig. 6.5(b), it can be seen that the STOI values of the E-MVDR output are smaller than those of the noisy input signal. This indicates that the predicted speech quality and intelligibility of the MVDR is severely degraded without clock synchronization. Note that to obtain absolute intelligibility scores, the STOI output needs to be mapped using for

example a logistic function. Moreover, these results also indicate that noise reduction performance of the MVDR with clock synchronization (i.e., J-MVDR, G-MVDR and B-MVDR) can reach the same performance as the C-MVDR, where clocks of all microphones are perfectly synchronized with the reference clock.

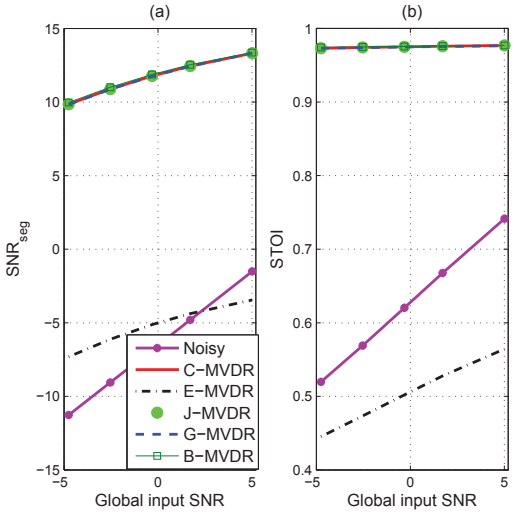


Figure 6.5: (a) The segmental SNR of node 1 versus the global input SNR. (b) The STOI of node 1 versus the global input SNR.

6.5.3 Clock Synchronization with Noisy Parameters

Next, we investigate the performance of the clock synchronization algorithms in a realistic setup where the measurements are subject to imperfections. For the JCS and GbCS, this means that we add white Gaussian noise to the time-stamps with a variance that is normalized by the precision of the internal clock. The system clock in modern PCs runs at 66 MHz. The minimum difference between two time-stamps is thus $1/(66 \times (10^6))$. The variance on the measurement noise is then given by $66 \times 10^6 \times \sigma^2$, with σ^2 the variance of the white Gaussian noise process. Since the performance of the JCS and GbCS depends on the number of transmissions, we use for both algorithms the same amount of 400 data transmissions. For the BSrOE we use a noisy speech signal instead of the pure noise. The noisy signal consists of a speech signal degraded by additive babble noise at several input SNRs. Note that the BSrOE requires much more data transmissions than the other two reference algorithms, namely 20480.

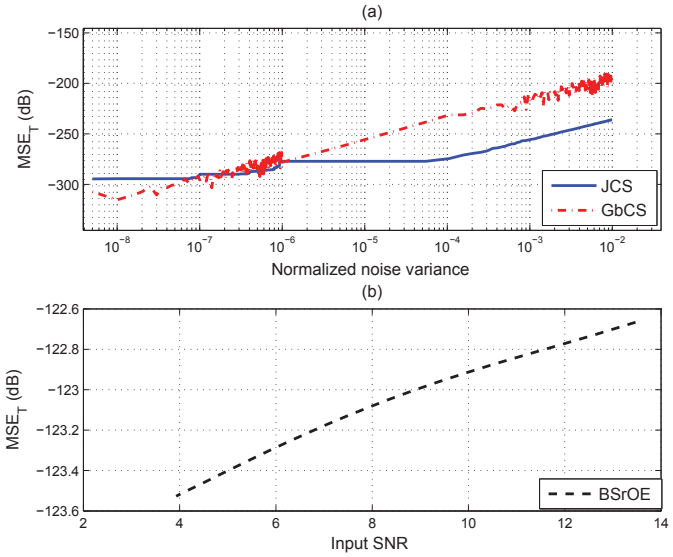


Figure 6.6: (a) The MSE versus the noise variance in time recording. (b) The MSE versus the global input SNR.

The effect of the measurement noise of time-stamp on the estimation accuracy of clock skew is shown in Fig. 6.6. In Fig. 6.6(a), the $\mathrm{MSE_T}$ for both the JCS and the GbCS is seen to increase with increasing measurement-noise variance. In addition, the results show that the $\mathrm{MSE_T}$ of the JCS increases slower than that of the GbCS. This is reasonable, since the time model in the JCS can take measurement noise of time-stamps into account, and the clock parameters in the JCS are estimated by minimizing the least squares norm of the measurement noise of time-stamps, while the time model in the GbCS assumes that there is no measurement noise on the time-stamps. The JCS shows a better estimation accuracy of the clock skew than the gossip based algorithm. In Fig. 6.6(b), the $\mathrm{MSE_T}$ of the BSrOE is increased from -123.5 dB to -122.7 dB with increasing global input SNR, which indicates that the effect of the SNR of the observed signal on the estimation accuracy of the BSrOE in terms of MSE is small, around 1 dB in this SNR range.

To illustrate the performance of the MVDR beamformer in the situation with measurement noise on the time-stamp, we investigate the $\mathrm{SNR_{seg}}$ and the STOI of the MVDR output of reference node 1 versus the normalized noise variance. The global input SNR of the signal at node 1 is -2.5 dB. In Figs. 6.7(a) and 6.7(b), both the $\mathrm{SNR_{seg}}$ and the STOI of the J-MVDR and the G-MVDR are decreased with increasing noise variance on the time stamps. Although the B-MVDR uses the noisy speech

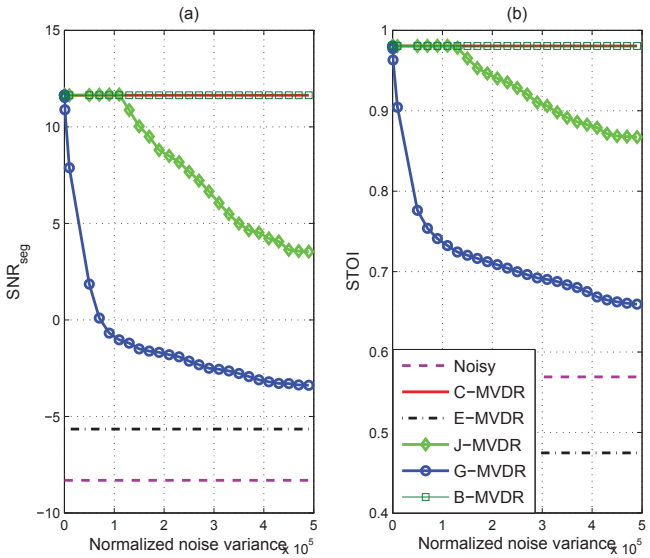


Figure 6.7: (a) The segmental SNR of node 1 versus the noise variance in time recording. (b) The STOI of node 1 versus the noise variance in time recording.

signal for clock synchronization, it reaches the same performance as the C-MVDR, since the B-MVDR is a signal-based algorithm, not sensitive for time-stamp noise. The performance of the J-MVDR decreases slower than that of the G-MVDR. This is consistent with the simulation results in Fig. 6.6. Moreover, for small noise variances, the J-MVDR reaches the same performance as the C-MVDR, while this is at a much lower transmission cost than the B-MVDR.

To further access the performance of the B-MVDR versus the global input SNR, we set the normalized measurement-noise variance in the JCS and GbCS to 0.0001, and set the global input SNR of the signal at reference node 1 is in the range from -5 dB up to 5 dB. In Fig. 6.8, the simulation results show that both the SNR_{seg} and the STOI of all three clock synchronization algorithms can reach the same performance as the centralized MVDR beamformer, which indicates that the accuracy of the clock synchronization of the three algorithms is still enough for beamforming technologies, when noisy speech is used in the B-MVDR and measurement noise with noise variance 0.0001 is added to the time-stamps used by the J-MVDR and the G-MVDR.

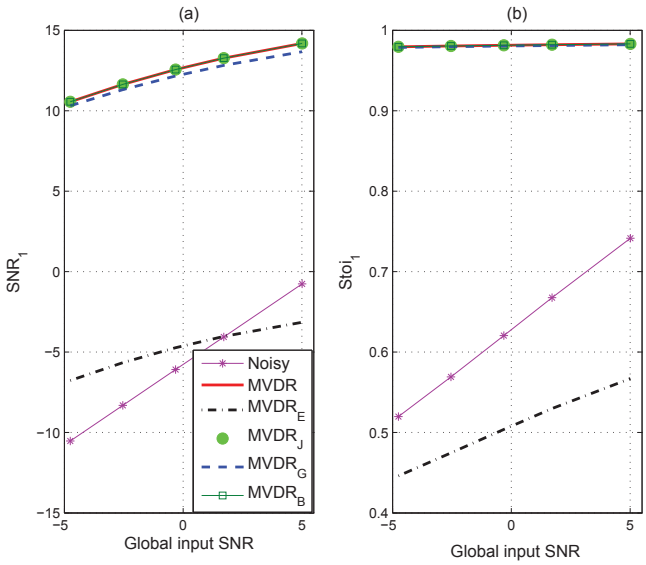


Figure 6.8: (a) The segmental SNR of node 1 versus the global input SNR. (b) The STOI of node 1 versus the global input SNR.

6.5.4 Discussion

Despite fact that all three algorithms can synchronize the clocks in WSNs, they do differ on four aspects that are important for use in WASNs, which are: 1) requirement on the network structure; 2) requirement on a central processor; 3) including noise or measurement uncertainty; 4) required number of data transmissions. In Table 6.1 we give an overview on each of these aspects for the three algorithms, which we further discuss below.

First, in the JCS and BSrOE, choosing or setting a reference clock in the network is required, since these two algorithms estimate the relative clock skew between a reference node and the other nodes. This requirement leads to the fact that these two algorithms can only operate in a network with a special topology, such as star connected networks or fully connected networks, where the node with the reference clock is connected with all other nodes. However, WASNs may be dynamic as nodes may join or leave the network due to a defect or an empty battery, resulting in unpredictable changes in network size and topology. Thus, the limitation of network topology of the JCS and BSrOE may constraint their practical application. On the other hand, the GbCS can operate in a randomly connected network, and the clocks of all nodes in the GbCS are synchronized with a virtual clock.

Table 6.1: Four aspects of the three clock synchronization algorithms.

Aspects	JCS	GbCS	BSrOE
Requirement on network structure	Yes, fully connected or star connected	No	Yes, fully connected or star connected
Requirement on a central processor	Yes	No	Yes
Including noise or measurement uncertainty	The accuracy of clock synchronization is decreased, the beamformer output get degraded	The accuracy of clock synchronization is decreased, the beamformer output get degraded	The accuracy of clock synchronization is slightly decreased, but is enough for beamforming technologies
Required number of data transmissions	$2K(N-1)$ in fully connected or star connected network	$4C$ in a randomly connected network	$(N-1)Pf_{\max}$ in fully connected or star connected network

Second, both the JCS and BSrOE require a central processor to synchronize all clocks with the reference clock, where the observed signals or timing messages are gathered and processed in the central processor. This requirement is consistent with the requirement on network topology, which may limit their applications in practical WASNs. On the contrary, the GbCS algorithm has no such requirements, since it is based on local processing and local communication.

Furthermore, to access the effects of noise or measurement uncertainty on the synchronization accuracy of the three algorithms and on the noise reduction performance of beamforming technologies, we took measurement uncertainty into account. The simulation results show that the synchronization accuracy of the three compared algorithms is sufficient for synchronizing the clock skew in scenarios without measurement noise. In practical scenarios with measurement uncertainty or noise, the beamformer output with time-stamp based clock synchronization algorithms get degraded, while the accuracy of the signal based clock synchronization algorithms is still sufficient for beamforming technologies.

Finally, we investigated the communication cost of the three algorithms in terms of data transmissions. Since both the JCS and the GbCS are time-stamp based clock synchronization algorithms, their communication cost is proportional to the number of transmissions. On the other hand, the required number of data transmissions of the

BSrOE algorithm depends on the number of nodes and the size of the observed signals. Furthermore, the latter algorithm generally requires many more data transmissions. Moreover, since both the JCS and BSrOE require a central processor to do signal processing, they will require additional data transmissions if the network topology is neither fully connected nor star connected.

6.6 Conclusions

In this work, we first analyzed effects of clock synchronization on the DSB with a synthetic signal. Then, we analyzed communication cost of three different clock synchronization algorithms. From this, it follows that the BSrOE requires a significantly larger amount of transmissions than the JCS and the GbCS approaches due to the fact that it is signal based. To which extent this high data-transmissions is a problem for distributed signal processing, depends on the processing in the subsequent steps. The experimental study has shown that the accuracy of clock synchronization of the three algorithms is sufficient for the MVDR beamformer under ideal circumstances. In scenarios with measurement uncertainty or noise, the output of the MVDR with the JCS and the GbCS degrades, but the MVDR with the BSrOE reaches the same performance as the centralized MVDR beamformer. For small amounts of measurement noise, the JCS gives similar performance as the BSrOE, but, at a significantly lower amount of data transmissions.

References

[1] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[2] R. Heusdens, G. Zhang, R. C. Hendriks, Y. Zeng, and W. B. Kleijn. Distributed MVDR beamforming for (wireless) microphone networks using message passing. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[3] S. Markovich-Golan, S. Gannot, and I. Cohen. Distributed multiple constraints generalized sidelobe canceler for fully connected wireless acoustic sensor networks. *IEEE Trans. Audio, Speech, Lang. Process.*, 21:343–356, Oct. 2012.

[4] Y. Zeng and R. C. Hendriks. Distributed delay and sum beamformer for speech enhancement via randomized gossip. *IEEE Trans. Audio, Speech, Lang. Process.*, 22:260 – 273, Jan. 2014.

[5] Y. C. Wu, Q. Chaudhari, and E. Serpedin. Clock synchronization of wireless sensor networks. *IEEE Signal Processing Magazine*, 21:260 – 273, Nov. 2013.

[6] R. T. Rajan and A. Veen. Joint ranging and clock synchronization for a wireless network. In *IEEE Int. Workshop on Computational Advances in Multi-Sensor Adaptibe Processing*, pages 297–300, 2011.

[7] L. Schenato and F. Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, 2011.

[8] S. Markovich-Golan, S. Gannot, and I. Cohen. Blind sampling rate offset estimation and compensation in wireless acoustic sensor networks with application to beamforming. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[9] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[10] D. Cherkassky and S. Gannot. Blind synchronization in wireless sensor networks with application to speech enhancement. In *Int. Workshop on Acoustic Echo and Noise Control*, 2014.

[11] J. S. Garofolo. DARPA TIMIT acoustic-phonetic speech database. *National Institute of Standards and Technology (NIST)*, 1988.

[12] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen. An algorithm for intelligibility prediction of time-frequency weighted noisy speech. *IEEE Trans. Audio, Speech, Lang. Process.*, 19(7):2125–2136, September 2011.

Chapter 7

Conclusions and Future Work

In this thesis we focused on the field of distributed noise reduction for speech enhancement in wireless acoustic sensor networks (WASNs). The objective of distributed noise reduction algorithms is to obtain the same performance as their centralized equivalents, as their centralized equivalents by performing computations in a distributed way.

7.1 Conclusions and Discussion

This section gives answers to the research questions, which we identified in Section 1.2, and discusses the results of our research work.

As WASNs are general randomly connected, making sense of distributed signal processing for randomly connected WASNs is an important task. The following research question was thus investigated.

1) How can we develop distributed algorithms that perform in randomly connected WASNs via local processing and improve the speech quality and intelligibility in a similar way as centralized algorithms?

To answer this first question, we proposed a distributed delay-and-sum beamformer (DDSB) algorithm in Chapter 3, a clique-based distributed beamformer (CbDB) in Chapter 4, and a distributed minimum variance distortionless response (DMVDR) beamformer in Chapter 5. The three distributed beamformers are proposed for decentralized estimation of clean speech signal in WASNs. Since they are based on only local communication and local processing, they have no requirements on a special network topology and no risk of having a single point of failure. The theoretical analysis and simulation results shown that the proposed distributed beamformers asymptotically converge to the optimal estimation of their centralized equivalents.

In Chapters 3 and 4, both the DDSB and the CbDB assume that WASNs are in incoherent noise field or in diffuse noise field. However, the experimental results shown that the DDSB and the CbDB can be used sucessfully in other noise fields and reduce noise as well. To further improve their noise reduction performance, we proposed the DMVDR beamfromer in Chapter 5 for coherent noise fields where noise signals between different microphones are correlated. In addition, we have shown that the DMVDR beamformer has better noise reduction performance than the DDSB and the CbDB in coherent fields.

Since the proposed distributed speech enhancement algorithms asymptotically converge to the optimal estimation of their centralized equivalents, we analyzed the convergence speed and communication cost of the distributed algorithms and answered the research question as follows.

2) How to effectively reduce communication cost in distributed speech enhancement algorithms?

This second question was addressed by reducing the communication cost of randomized gossip algorithms [1], since the proposed distributed speech enhancement algorithms are mainly based on the randomized gossiping. The randomized gossip algorithms are classified into an asynchronous communication scheme and a synchronous

communication scheme. To improve the convergence speed of the synchronous randomized gossip algorithm, we presented an improved general distributed synchronous averaging (IGDSA) algorithm for randomly connected networks in Chapter 3.

However, the proposed IGDSA algorithm did not reduce the communication cost of the randomized gossip algorithms, which was shown in Fig. 3.4. Furthermore, experiments on the comparisons between the DDSB and several existing distributed speech enhancement algorithms shown that besides the advantage of not having a topology constraint, the DDSB has better noise reduction performance than the referenced distributed adaptive node-specific signal estimation (DANSE) algorithm [2] at the expense of a higher communication cost. The main reason for this is that the DANSE algorithm employs a broadcast protocol and performs time-recursive updates over signal frames, while the DDSB based algorithms use a point-to-point transmission protocol (pairwise communication scheme) per signal frame.

Therefore, to reduce the communication costs of the DDSB, we first proposed a clique-based gossip algorithm, and then we presented the CbDB algorithm based on the principle of clique-based gossiping for distributed speech enhancement in Chapter 4. The CbDB algorithm not only improved the convergence speed of the DDSB, but also reduced the communication cost of the DDSB. Instead of the pairwise neighboring nodes updating their estimates per iteration as with the randomized gossip algorithm, the nodes in two neighboring non-overlapping cliques update their estimates simultaneously per iteration with the clique-based gossip algorithm. This algorithm performs a broadcast protocol and allows more nodes to update their estimates simultaneously. Moreover, simulations in Chapter 4 demonstrated that the proposed CbDB algorithm is robust for node failures. The CbDB algorithm was further compared to a reference algorithm that is based on clusters, which reduces the communication cost of the randomized gossip algorithm via cluster structures. Simulations in Chapter 4 also showed that the robustness of the CbDB is better than the cluster-based algorithm due to the fact that cliques generally have a better connectivity than clusters.

To develop a distributed speech enhancement algorithm for coherent noise fields, we further answered the research question as follows.

3) How to estimate the inverse correlation matrix in distributed way?

In Chapter 5, we address this third research question on how to estimate inverse correlation matrix in distributed way. The distributed estimation of the inverse correlation matrix was based on the fact that using the Sherman-Morrison formula, the estimation of the inverse of the correlation matrix can be seen as a consensus problem and can be solved using distributed consensus algorithms. Although randomized gossip algorithms can be used to solve consensus problems in distributed way, they are not good for distributed estimation of correlation matrix. The reason is that the correlation matrix is recursively estimated across time, and randomized gossip algorithms have estimation errors per time frame. Although the convergence error between gossip-based estimated correlation matrix and the centralized estimated correlation matrix per time frame is decreased with increasing number of iterations, the error for the estimated inverse matrix accumulates across time. Therefore, we presented a clique-based distributed algorithm to eliminate this convergence error. The simulation results

shown that the proposed algorithm for estimation of the inverse correlation matrix can reach the same performance as the centralized algorithm. According to the results in this chapter, distributed adaptive beamforming technologies for multi-microphone noise reduction in coherent noise fields can be developed.

Furthermore, we considered privacy preservation problem in performing distributed speech enhancement and answered the research question as follows.

4) How to develop distributed methods for speech enhancement in privacy preserving WASNs?

This fourth research question was considered in Chapter 5. We considered scenarios where multiple users make use of a WASN consisting of many processors (including their own). The users use the WASN to estimate their signal of interest, which can be different for each user, and want to keep private to which specific source they are interested. Based on the distributed estimation of the inverse of the correlation matrix, we introduced the DMVDR beamformer, where each user in the WASN estimates his signal of interest by performing distributed computations on the WASN data, while keeping the particular source of interest private. The privacy preservation was provided by hiding the steering vector, where each user in the network is assumed to know the steering vector only locally. Our results shown that the DMVDR beamformer generally requires less communication cost than its centralized version in string connected networks and fully connected networks when considering privacy persevering WASNs. Since in Chapter 5 we assume that the steering vector is privacy sensitive, it will be interesting to investigate how to develop a distributed way to estimate the steering vector in more complicated acoustic scenarios (e.g. involving reverberation) and keep the knowledge on the source of interest private. This is also one of the future research topics that will be discussed in the next section.

Existing distributed multi-microphone signal processing generally assume that clocks between different microphones are synchronized, we thus studied the effects of clock synchronization problems on multi-microphone noise reduction and answered the research question as follows.

5) How does clock synchronization problems affect multi-microphone noise reduction and what effect do the clock synchronization algorithms have on multi-microphone noise reduction?

This research question was answered in Chapter 6. We investigated the effect of clock synchronization problems on multi-microphone signal processing in WASNs via theoretical and experimental analysis of the noise reduction performance of beamforming technologies. Results have shown that the noise reduction performance of both the DSB and the MVDR beamformer was severely degraded when clocks of microphones were not synchronized. Three issues arise. At first, the beamformer response is not identical to the desired response. Secondly, speech signals at different nodes get translated to different frequencies, and finally, speech signals at the different nodes will not get aligned with each other.

Furthermore, the effects of three existing clock synchronization algorithms on the MVDR beamformer in simulated WASNs was studied in Chapter 6. The three clock

synchronization algorithms are the joint ranging and clock synchronization (JCS) algorithm [3], the gossip-based clock synchronization (GbCS) [4] and the blind sampling-rate offset estimation (BSrOE) [5]. We found that the clock synchronization of the three algorithms is accurate enough for beamforming technologies under ideal circumstances without measurement noise. When measurement uncertainty is present on the communicated time information, the output of the MVDR beamformer with the JCS and the GbCS get distorted, while the accuracy of the BSrOE algorithm is still accurate enough for beamforming technologies.

The results in this chapter give a theoretical reference on using the three clock synchronization algorithms for multi-microphone noise reduction in WASNs. For multi-microphone signal processing in randomly connected WASNs, the GbCS algorithm has more advantages than the JCS and the BSrOE, since both the JCS and the BSrOE require a central processor to do clock synchronization, which constraints the network topology to be fully connected or star connected. For practical WASNs with specialized network topologies, the BSrOE performs better than the JCS, and the JCS performs better than the GbCS. Both the JCS and the GbCS are time-stamp based algorithms. The time model in the GbCS ignores measurement noise, which explains the worse performance when measurement noise is present. Moreover, when considering communication cost in WASNs, the GbCS and the JCS generally require less communication cost than the BSrOE, since the communication cost of the BSrOE depends on the number of nodes and and the size of the observed signals, while the communication cost of the GbCS and the JCS depends on the number of transmissions. Due to the importance of clock synchronization in practical WASNs, it would be interesting to perform and compare the three clock synchronization algorithms in practical setup, where a WASN is constructed with mobile phones and/or computers.

7.2 Directions for Future Research

Based on the methods and findings presented in this thesis, we suggest the following recommendations for future work.

1. Distributed estimation of the recursive least squares filter The basic concept that was used in Chapter 5 to derive the distributed algorithm for matrix inverse calculation and formulate this as a consensus problem, is the Sherman-Morrison formula. The Sherman-Morrison formula has been used before to define recursive filters, e.g., the recursive least squares (RLS) filter [6]. This implies that the basic idea of this distributed algorithm can also be employed to estimate the RLS filter in a distributed way. With the RLS filter, filter coefficients are recursively estimated by minimizing a weighted linear least squares cost function relating to the input signals [6]. Estimation of the RLS filter coefficients can thus be seen as a distributed consensus problem and can be solved using three rounds of gossip iterations. Two rounds of gossip iterations are used to estimate the inverse of the correlation matrix, and the third round is used to update the correction vector of the filter coefficient.

2. Distributed estimation of the acoustic transfer function

In this thesis and we assumed that the acoustic transfer function (ATF) is known a priori, which should be estimated in practical application. The ATF depends on acoustic environments, e.g., the room scale and type of materials. In free field scenarios where reverberation can be neglected, the acoustic transfer function (ATF) is determined by damping and delay, which can be obtained by estimating the distance between the microphones and the target source. Therefore, distributed estimation of the ATF in free field can be obtained by estimating the location of the target source and the microphones in a distributed way, see e.g., [7][8]. In practical acoustic environments, which consist of both reverberation and background noise, the ATF is determined by the acoustic room impulse response (RIR) [8]. An algorithm for a distributed estimation of relative ATF has been proposed in [9]. This algorithm is constrained to work in fully connected WASNs. For distributed estimation of RIRs in fully connected networks, one could also use the proposed clique-based distributed consensus algorithm in Chapter 4 and combine this with the RIR estimation algorithm in [10]. With each node broadcasting its observations to its neighbors, the local RIR estimates can be obtained via distributed consensus processing. Furthermore, the proposed algorithm for distributed estimation of the correlation matrix in Chapter 5 can be used to develop a distributed multi-channel Wiener filter, which can be used for randomly connected WASNs in practical environments. The attractiveness of this is the fact that estimation of the ATF is implicitly included in the multi-channel Wiener filter by estimating the noise and noisy correlation matrices.

3. Privacy preserving beamforming technologies in a WASN

We have discussed the concept of privacy preserving beamforming in Chapter 5. However, the distributed noise reduction algorithm in Chapter 5 only made privacy preservation of the ATF possible, rather than complete preservation of the privacy of the different target signals against eavesdroppers. Combining secure signal processing [11] with the proposed DDSB algorithm in Chapter 3, privacy preserving distributed speech enhancement was proposed in [12] and [13]. However, the homomorphic encryption for privacy preservation in WASNs requires very high bit rates for data transmission, and is computationally very complex. For further research on privacy preserving distributed speech enhancement, one can rely on the distributed speech enhancement algorithms derived in this thesis, combined with secret sharing [14]. Secret sharing is less complex and requires lower data rates than homomorphic encryption. In this case, the reconstruction of the desired signals is only possible when a sufficient number of shares are combined.

4. Node subset selection for distributed noise reduction

The distributed noise reduction algorithms proposed in this thesis, estimate the target clean speech source using the observed signals of all nodes of a connected WASN. Although WASNs can employ a large number of microphones to cover a larger spatial field and provide more information, WASNs with more nodes will require more transmissions or communication cost and have therefore a higher power consumption. An important aspect for distributed noise reduction algorithms is the tradeoff between noise reduction performance and power consumption. One research direction to guar-

antee noise reduction performance and reduce power consumption is selection of the most useful nodes in WASNs to perform signal estimation, rather than using all of the nodes. This means that some pre-processing can be used to prune the original WASN to be a smaller WASN. For example, the microphones with highest SNR are chosen [15] or a set of microphones with strong cross-correlation is chosen [16]. In this case, less useful nodes (e.g., far away from the target sources) can be switched off and save their energy for other tasks, and each node will communicate with less nodes. As a consequence the distributed estimation of the target source in each node will be based on less information. The tradeoff between noise reduction performance of distributed beamforming technologies and power consumption can be evaluated by assessing the mean square error between the output of distributed algorithms and the output of the centralized estimation versus data transmissions.

References

[1] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Trans. Inf. Theory*, 52(6):2508 – 2530, Jun. 2006.

[2] A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: Sequential node updating. *IEEE Trans. Signal Process.*, 58(10):5277 –5291, Oct. 2010.

[3] R. T. Rajan and A. Veen. Joint ranging and clock synchronization for a wireless network. In *IEEE Int. Workshop on Computational Advances in Multi-Sensor Adaptibe Processing*, pages 297–300, 2011.

[4] L. Schenato and F. Fiorentin. Average timesynch: a consensus-based protocol for time synchronization in wireless sensor networks. *Automatica*, 47(9):1878–1886, 2011.

[5] S. Markovich-Golan, S. Gannot, and I. Cohen. Blind sampling rate offset estimation and compensation in wireless acoustic sensor networks with application to beamforming. In *Int. Workshop on Acoustic Echo and Noise Control*, 2012.

[6] S. Haykin. *Adaptive filter theory (3rd Edition)*. Prentice Hall, 1995.

[7] S. Haykin and K. J. R. Liu. *Handbook on array processing and sensor networks*. Wiley Online Library, 2009.

[8] M. Brandstein and D. Ward (Eds.). *Microphone arrays*. Springer, 2001.

[9] S. Markovich Golan, S. Gannot, and I. Cohen. Distributed GSC beamforming using the relative transfer function. In *European Signal Processing Conference*, Bucharest, Aug. 2012.

[10] Y. A. Huang and J. Benesty. Adaptive multi-channel least mean square and newton algorithms for blind channel identification. *Signal Processing*, 82(8):1127–1138, August 2002.

[11] R. L. Lagendijk, Z. Erkin, and M. Barni. Encrypted signal processing for privacy protection. *IEEE Signal Process. Mag.*, pages 82–105, Jan. 2013.

[12] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy-preserving distributed speech enhancement for wireless sensor networks by processing in the encrypted domain. In *IEEE Int. Conf. Acoust, Speech, Signal Process. (ICASSP)*, pages 7005–7009, Canada, May 2013.

[13] R. C. Hendriks, Z. Erkin, and T Gerkmann. Privacy preserving distributed beamforming based on homomorphic encryption. In *Proc. European Signal Proc. Conf. Eusipco*, pages 7005–7009, Marrakesh, Morocco, 2013.

[14] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

[15] M. Wolfel, C. Fugen, S. Ikbal, and J. W. Mcdonough. Multi-source far-distance microphone selection and combination for automatic transcription of lectures. *Proc. INTERSPEECH.*, 2006.

[16] K. Kumatani, J. McDonough, J. Lehman, and B. Raj. Channel selection based on multichannel cross-correlation coefficients for distant speech recognition. *Joint Workshop on Hands-free Speech Communication and Microphone Arrays (HSCMA).*, pages 1–6, 2011.

Appendix A

Derivations for Chapter 4

For completeness we give in this appendix the most important steps to find all non-overlapping cliques in randomly connected wireless sensor networks.

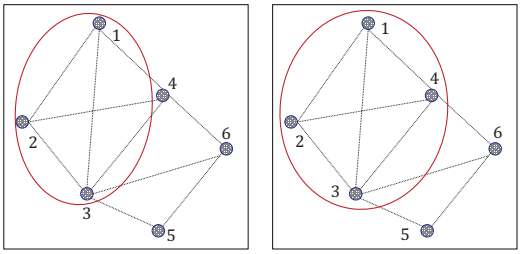
A.1 Non-overlapping Cliques

Consider a WASN as a randomly connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the vertex set with N acoustic sensor nodes, and $\mathcal{E}$ is the edge set of communication links between every set of two neighboring nodes. We assume that each node $i \in \mathcal{V}$ has $|\mathcal{N}_i|$ neighboring nodes with $\mathcal{N}_i$ the set of neighbors of node i. Further, we assume each node i consists of M_i microphones. The total number of microphones then is $M = \sum_{i=1}^{N} M_i$.

The algorithm that we present in Chapter 4 makes use of non-overlapping cliques. Therefore, in this appendix a discussion is given on non-overlapping cliques and how to find them. In $\mathcal{G}$, a clique is a fully connected sub-graph (Fig. A.1(a)), and a maximal clique is a fully connected sub-graph that cannot be extended by including more nodes without ceasing to be a clique, see Fig. A.1(b). Since each node i can belong to several maximal cliques, the maximal cliques of $\mathcal{G}$ can be overlapping (Fig. A.1(c)). In Chapters 4 and 5 we exploit the concept of non-overlapping cliques, such that each node belongs to only one clique, see Fig. A.1(d). In this section, we discuss an approach to find a set of non-overlapping cliques in a distributed way. The approach consists of two steps. First, each node $i \in \mathcal{G}$ finds all maximal cliques which it belongs to in a distributed way, and subsequently each node is assigned to just one clique by local communication.

Given a connected undirected graph $\mathcal{G}$, a slightly modified version of the first Bron-Kerbosch algorithm [1] can be used to find maximal cliques in a distributed way. To do so, each node i runs the first Bron-Kerbosch algorithm, where the set with candidate nodes that can form a clique with node i consists of the set $\mathcal{N}_i$ of neighboring nodes. For each node this results in a set of maximal cliques.

As the maximal cliques should be non-overlapping, i.e., each node should belong to just one clique, each node independently runs an algorithm to find the largest



(a) Clique (b) Maximal Clique

(c) Overlapping maximal cliques (d) Non-overlapping cliques

Figure A.1: Schematic diagram of network topologies.

possible clique in distributed fashion. Notice that this can lead to situations where a smaller clique is selected instead of a maximal clique, in order to make the cliques non-overlapping. Let K^i_{max} denote the size of the maximum maximal clique node i belongs to and let K^i_{M} denote the maximum value of K^j_{max} with $j \in \mathcal{N}^c_i$ and $\mathcal{N}^c_i$ the neighboring nodes in node i's current candidate maximal cliques. We consider that each node i is numbered with a unique identification number and receives the information K^j_{max}, $j \in \mathcal{N}^c_i$ from the neighboring nodes in its current candidate maximal cliques. Each time instant, the unique number of each node i is decreased by 1. When node i's number equals 0, it computes the maximal value K^i_{M} and compares K^i_{M} with its maximum size K^i_{max}. If K^i_{M} and K^i_{max} are identical, it randomly selects one maximal clique with size K^i_{max} and informs the neighboring nodes in the selected clique to select the same maximal clique. All nodes in the selected clique then set their number to 0 and send a message to their neighboring nodes that are not in the current selected clique and remove themselves from these neighbors' current list of candidate maximal cliques, after which these neighbors update their K^i_{M}. When all node identification numbers become negative, all the largest possible non-overlapping cliques are formed.

To further explain this distributed approach for finding all non-overlapping cliques in a graph $\mathcal{G}$, an example is given on the graph given in Fig. A.1. Let $\{\cdots\}$ denote a set of nodes that are in a maximal clique. Each node i in iteration 0 first finds its maximal cliques using a slightly modified version of the first Bron-Kerbosch algorithm [1].

In iteration 1, the unique number of the node 1 equals 0 and becomes active. Node 1 computes K_{M}^{1} and compares K_{M}^{1} with K_{max}^{1}. Since $K_{\mathrm{M}}^{1} = K_{\mathrm{max}}^{1} = 4$, it selects the maximal clique $\{1, 2, 3, 4\}$, and the neighboring nodes 2, 3 and 4 set their identification number to zero. Notice that, if node 1 has several cliques with the size K_{max}^{1}, it will randomly select one. Further, if node 1 does not have clique with the size K_{max}^{1}, it will become inactive. Nodes 3 and 4 send a message to nodes 5 and 6 and remove themselves from their list of candidate maximal cliques $\{3, 4, 6\}$ and $\{3, 5, 6\}$. Similarly, node 5 in time instant 5 becomes active and selects the clique $\{5, 6\}$. In time instant 6, the processing ends with negative identification numbers for all nodes.

Table A.1: An example of finding all non-overlapping cliques in a WSN with 6 nodes.

	1 (1)	2 (2)	3 (3)	4 (4)	5 (5)	6 (6)
Time 0	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$ $\{3, 4, 6\}$ $\{3, 5, 6\}$	$\{1, 2, 3, 4\}$ $\{3, 4, 6\}$	$\{3, 5, 6\}$	$\{3, 4, 6\}$ $\{3, 5, 6\}$
	1 (0)	2 (1)	3 (2)	4 (3)	5 (4)	6 (5)
Time 1	$\{1, 2, 3, 4\}$ $K_{\mathrm{max}}^{1} = K_{\mathrm{M}}^{1} = 4$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{5, 6\}$	$\{6\}$ $\{5, 6\}$
	1 (-1)	2 (-1)	3 (-1)	4 (-1)	5 (3)	6 (4)
Time 2	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{5, 6\}$	$\{6\}$ $\{5, 6\}$
	1 (-4)	2 (-4)	3 (-4)	4 (-4)	5 (0)	6 (1)
Time 5	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{5, 6\}$ $K_{\mathrm{max}}^{5} = K_{\mathrm{M}}^{5} = 2$	$\{5, 6\}$
	1 (-5)	2 (-5)	3 (-5)	4 (-5)	5 (-1)	6 (-1)
Time 6	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{1, 2, 3, 4\}$	$\{5, 6\}$	$\{5, 6\}$

References

[1] C. Bron and J. Kerbosch. Algorithm 457: Finding all cliques of an undirected graph. *Communication of the ACM*, 16(9):575–577, 1973.

Samenvatting

In toepassingen van digitale spraakcommunicatie zoals handsfree mobiele telefonie, gehoorapparaten en mens-machine communicatie systemen, worden de opgenomen spraak signalen vaak verstoord door achtergrondgeluiden. Hierdoor kan de kwaliteit en spraakverstaanbaarheid ernstig verslechteren. Traditionele ruisonderdrukkings technieken maken doorgaans gebruik van een microfoonarray en bewerken de signalen met gecentraliseerde beamforming technieken. Recente ontwikkelingen in de micro-elektromechanische systemen en draadloze communicatie maken de ontwikkeling van draadloze sensornetwerken mogelijk, waar goedkope, low-power en multi-functionele sensoren draadloos met elkaar zijn verbonden. Vergeleken met de conventionele microfoon arrays, kunnen draadloze sensoren willekeurig geplaatst worden in de omgeving, waardoor ze een groter ruimtelijk gebied bedekken en meer informatie kunnen geven over de waargenomen signalen. Dit proefschrift verkent een aantal problemen op het gebied van multi-microfoon spraakverbetering voor draadloze akoestische sensornetwerken, zoals gedistribueerde ruisreductie, kloksynchronisatie en privacy waarborging.

Ten eerste hebben we een gedistribueerde *delay-en-sum beamformer* (DDSB) ontwikkeld voor spraakverbetering in draadloze akoestische sensornetwerken. Vanwege de beperkte energie van draadloze sensor nodes, hebben signaalverwerkings algoritmen met een lage rekencomplexiteit en lage communicatiekosten de voorkeur. Met gedistribueerde signaalverwerking communiceert ieder node alleen met zijn direct naburige knooppunten en voert alleen lokaal berekeningen uit, waarbij communicatie kosten en rekencomplexiteit worden verdeeld over alle nodes in het netwerk. Zonder centrale processor en beperking op de netwerktopologie, schat het DDSB algoritme het gewenste spraaksignaal via lokale bewerkingen en lokale communicatie. Het DDSB algoritme is gebaseerd op een iteratief schema, waarbij in elke iteratie paren van aangrenzende nodes hun schattingen bijwerken volgens het principe van de traditionele *delay-en-sum* (DSB) beamformer. De schatting van de DDSB convergeert asymptotisch naar de optimale oplossing van de gecentraliseerde DSB beamformer. Echter, de ruisonderdrukking van de DDSB gaat samen met hogere communicatiekosten die een ernstige nadeel kunnen zijn voor praktische toepassingen.

Daarom wordt in het tweede deel van dit proefschrift, een gedistribueerde beamformer voorgesteld op basis van cliques, afgekort als CbDB, met als doel de communicatiekosten van het oorspronkelijke DDSB algoritme te verminderen. Met de CbDB worden per iteratie tegelijkertijd de schattingen van alle nodes in twee naburige

niet-overlappende cliques bijgewerkt. Aangezien elke niet-overlappende clique uit meerdere nodes bestaat, staat de CbDB het toe om meer nodes tegelijkertijd hun schattingen te laten actualiseren. Dit leidt tot lagere communicatie kosten dan met het oorspronkelijke DDSB algoritme. Bovendien tonen theoretische en experimentele studies aan dat de CbDB convergeert naar de gecentraliseerde beamformer en dat de CbDB robuuster is tegen storingen in sensor nodes.

In het derde deel van dit proefschrift presenteren we een privacy gewaarboorde minimum variance distortionless response (MVDR) beamformer voor spraakverbetering in draadloze akoestische sensornetwerken. Verschillende draadloze apparaten in draadloze akoestische sensornetwerken behoren in het algemeen tot verschillende gebruikers. We beschouwen een scenario waarin een gebruiker deel neemt aan een draadloos akoestisch sensornetwerk en schattingen van zijn gewenste bron via dit netwerk gaat maken. Echter, deze gebruiker wil zijn bron van interesse priv houden. Om tot een gedistribueerde MVDR beamformer met behoud van privacy te komen, wordt eerst een gedistribueerd algoritme voorgesteld om op een recursieve manier de inverse van de correlatie matrix in willekeurig gevormde draadloze akoestische sensornetwerken te schatten. Deze gedistribueerde techniek is gebaseerd op het feit dat met de Sherman-Morrison formule een schatting van de inverse van de correlatiematrix kan worden beschouwd als consensus probleem. Door het verbergen van de stuurvector van de beamformer, kan de privacy behoudende MVDR beamformer dezelfde ruisonderdrukking prestaties als zijn gecentraliseerd versie bereiken.

In het laatste deel van dit proefschrift, onderzoeken we klok synchronisatie problemen voor multi-microfoon spraakverbetering in draadloze akoestische sensornetwerken. Elk draadloos apparaat in het netwerk is voorzien van een onafhankelijke klok oscillator, waardoor de interne klokken onderling onvermijdelijk verschillen. Deze klokverschillen tussen verschillende apparaten in het draadloze netwerk zullen leiden tot drift tussen de signalen zoals opgenomen door de verschillende apparaten, wat uiteindelijk zal resulteren in ernstige vermindering van de prestaties van multi-microfoon ruisonderdrukking. In dit deel presenteren we theoretische analyse van het effect van kloksynchronisatie problemen op beamforming technologien en beoordelen drie verschillende kloksynchronisatie algoritmen in de context van multi-microfoon ruisonderdrukking. Uit experimenteel onderzoek blijkt dat in ideale scenarios, de bereikte nauwkeurigheid van de drie kloksynchronisatie algoritmen voldoende nauwkeurigheid hebben voor de MVDR beamformer. Echter, in de praktische scenario's waar de klocksynchronisatie onzekerheid en ruis bevat, is het resultaat van de MVDR beamformer met zogenaamde time-stamp gebaseerde kloksynchronisatie verslechterd, terwijl de nauwkeurigheid van signaal gebaseerde kloksynchronisatie algoritmes nog voldoende is voor de MVDR beamformer, zij het tegen veel hogere communicatiekosten.

Acknowledgements

The four years PhD life not only teach me how to be a science researcher, but also teach me how to face inevitable changes in my life. In these four years, I have got a lot of guidance and help from the people around me and it would not have been possible to finish this thesis without their support. I would therefore like to acknowledge all of them.

First of all, I would like to express my deep thanks to my daily supervisor Richard Hendriks for your step by step guidance, your support and your suggestions during the whole period of the study. The thesis would not have been possible without your warm encouragement, your patience and great support during the most difficult time when writing this thesis. The way you guide me and the discussions we had have benefit me a lot.

Second, I want to thank Richard Heusdens for your excellent comments on my research work and your support during my study. I am so happy that I can study in the SIP lab. The group is warm and I had so much fun in our social events, coffee time and lunch time. That bring me to thank my promoter Inald for accepting me as a PhD student in this group and your great comments and suggestions of the thesis.

I also owe a lot of thanks to my colleges for their great support and sincere friendship. First, thank Nick and Guoqiang for your advices and your support of this thesis. Many of the research works in this thesis have been discussed with you and have got a lot of great advices from you. Thank Zhijie, Joao, Jorge, Cees and Bastiaan for sharing your knowledge and helping me understand Dutch culture. Additionally, I would like to extend my acknowledgment to other colleges and friends in the department. Sepeda, Bahnaz, Zeki, Jos, Christam, Ahmad, Feifei, Lu Zhang, Peng Xu and many others, thank you for all your friendship and sharing all the moments with me that made my time in Delft enjoyable.

This thesis would not be finished without the support and the love of my family. Therefore, I would like to express my gratitude and deepest appreciation to my parents, you have been supporting me in all aspects of my life with all you can or cannot afford, thank you for your endless love. I also would like to thank my brother Yunlong and my sister Fangfang for your encouragement and assistance during these years. And finally, Xinchao, being with you is like enjoying the sunshine, which makes me always feel warm and loved and gives me the courage and the confidence to face life's challenges, thank you for all your support and love.

Curriculum Vitae

Yuan Zeng was born on December 18th, 1984 in city Fuzhou, Jiangxi, P. R. China. She received her Bachelor degree in Biomedical Engineering from Nanchang Hangkong University in 2006.

In 2007 she worked as a chinese youth volunteer in the Wuxuan Environmental Protection Agency, Guangxi, P. R. China and received the bronze medal award of the China youth volunteer service.

She started her master study at Northwestern Polytechnical University in September 2007 and received her Master degree after presenting her graduate work, which was on management system for wireless sensor networks and research on aero-engine fault diagnosis, in 2010.

In September 2010, she started her PhD work in the Signal and Information Processing lab at Delft University of Technology. She was supervised by Dr. ir. Richard C. Hendriks and worked on the research topic speech enhancement in wireless acoustic sensor networks.