KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science
Department of Electrical Engineering

Distributed Signal Processing Algorithms for Acoustic Sensor Networks

Joseph C. Szurley

Supervisor:
Prof. dr. ir. M. Moonen
Co-supervisor:
Prof. dr. ir. A. Bertrand

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering Science

June 2015

Distributed Signal Processing Algorithms for Acoustic Sensor Networks

Joseph C. SZURLEY

Examination committee:
Em. prof. dr. ir. P. Van Houtte, chair
Prof. dr. ir. M. Moonen, supervisor
Prof. dr. ir. A. Bertrand, co-supervisor
Prof. dr. ir. H. Van hamme
Prof. dr. ir. T. van Waterschoot
Prof. dr. ir. I. Moerman
(Ghent University)
Prof. dr. I. Proudler
(Loughborough University)
Prof. dr. S. Gannot
(Bar-Ilan University)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor in Engineering Science

June 2015

© 2015 KU Leuven – Faculty of Engineering Science
Uitgegeven in eigen beheer, Joseph C. Szurley, Kasteelpark Arenberg 10, bus 2446, B-3001 Leuven (Belgium)

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of the publication may be reproduced in any form by print, photoprint, microfilm, electronic or any other means without written permission from the publisher.

Abstract

In recent years, there has been a proliferation of wireless devices for individual use to the point of being ubiquitous. Recent trends have been incorporating many of these devices (or nodes) together, which acquire signals and work in unison over wireless channels, in order to accomplish a predefined task. This type of cooperative sensing and communication between devices form the basis of a so-called wireless sensor network (WSN).

Due to the ever increasing processing power of these nodes, WSNs are being assigned more complicated and computationally demanding tasks. Recent research has started to exploit this increased processing power in order for the WSNs to perform tasks pertaining to audio signal acquisition and processing forming so-called wireless acoustic sensor networks (WASNs). Audio signal processing poses new and unique problems when compared to traditional sensing applications as the signals observed often have temporal characteristics that fluctuate rapidly requiring significantly faster transmission and computation speeds.

In order for WASNs to cope with these increased demands, it is incontrovertibly apparent that efficient algorithms need to be developed. These algorithms must not only be computationally efficient but must also take into consideration several network wide design constraints in order to efficiently allocate the limited resources of the nodes. The main aim of this thesis is, therefore, to incorporate these constraints into the design of distributed signal processing algorithms for WASNs. The envisaged WASNs are primarily tasked with signal estimation that is exemplified by binaural noise reduction systems that utilize additional remote microphones. These systems focus on speech enhancement and noise reduction as their signal estimation task, where spatial cue preservation plays a role.

The general problem statement of this thesis, the current state of the art and the mathematical framework for (distributed) signal estimation are given in the

introductory chapters. WSNs will first be introduced in a centralized framework, where all of the information is aggregated and processed at a central location in order to perform signal estimation, e.g., for speech enhancement. This is then extended to a distributed framework, where the processing is distributed across the nodes and where each node performs a (possible node-specific) signal estimation task. As a general aim, the distributed algorithms developed in this thesis look to perform the same estimation as in the centralized framework while reducing the amount of data exchange between the nodes.

The first part of this thesis focuses on a concrete application, investigating the improvements that can be achieved in terms of noise reduction performance and spatial cue preservation by adding a single remote sensor (microphone) to a binaural noise reduction system, e.g. binaural hearing aids or cochlear implants. The increase in the noise reduction performance by the addition of a remote sensor signal serves as the motivation for later chapters where many nodes consisting of multiple sensors are available. In the case where multiple remote microphones or nodes are available, a method for choosing which (sub)set of signals or nodes should be included in the estimation for maximum benefit is also discussed. A method is introduced that uses a so-called utility measure which limits the effect of node removal and can be used to aid in node addition. While first introduced in a centralized scenario, the utility is extended to that of a distributed scenario. However, due to the distributed nature of the signal estimation, the utility is shown to represent a bound as opposed to an exact quantity in the centralized scenario.

The second part of this thesis introduces linear compression and fusion rules of the distributed adaptive node-specific signal estimation (DANSE) algorithm for use in heterogeneous and mixed-topology WSNs. These mixed-topology WSNs are divided into a set of smaller substructures that are based on either clique or cluster formation. It is shown, that the nodes are able to converge to the same optimal solution as if they were to receive all of the uncompressed signals from every other node. Supporting techniques are also introduced relating to different broadcast strategies and topology formation.

In the third part of this thesis, each node implements a novel method to linearly compress its sensor signals in order to transmit to the other nodes in the WSN. This leads to the introduction of the topology-independent DANSE (TI-DANSE) algorithm. While the TI-DANSE algorithm is first introduced in a fully connected topology, the convergence properties are shown to be applicable in any topology, as long as the nodes have access to a network wide summed signal. An attractive attribute of the TI-DANSE algorithm is that since it relies on a network wide summed signal, it is less sensitive to link failures and also becomes applicable in WSNs with dynamic topologies. A method to compute this network wide summed signal is proposed that relies on a maximum of two

transmissions per node.

Finally, conclusions are given reiterating the contributions of the thesis as well as exploring possible future research directions.

Korte Inhoud

De laatste jaren is het individueel gebruik van draadloze apparaten zo sterk toegenomen, dat ze ondertussen alomtegenwoordig zijn. Deze toestellen, of nodes, verzamelen signalen en werken onderling samen over draadloze kanalen. Recente trends integreren vele van deze toestellen om een op voorhand gedefinieerde taak te volbrengen. Dit type van coöperatief meten en de communicatie tussen de toestellen vormt de basis van een zogenaamd draadloos sensornetwerk (wireless sensor network, of WSN).

Ten gevolge van de eeuwig toenemende verwerkingskracht van deze nodes, worden de WSN's beladen met steeds gecompliceerdere en computationeel veeleisendere taken. Recent onderzoek is begonnen met het uitbuiten van deze toegenomen verwerkingskracht om de WSN's taken te laten uitvoeren met betrekking tot het verwerven en verwerken van audiosignalen. In deze context spreken we over draadloze akoestische sensornetwerken (wireless acoustic sensor networks, of WASN's). Audiosignaalverwerking stelt nieuwe en unieke problemen vergeleken met traditionele WSN-toepassingen, aangezien de geobserveerde signalen vaak snel variërende temporele karakteristieken hebben, waardoor significant snellere transmissie- en berekeningssnelheden nodig zijn.

Om met deze strengere eisen voor WASN's om te gaan, staat het onomstotelijk vast dat efficiëntere algoritmes ontwikkeld moeten worden. Deze algoritmes moeten niet enkel computationeel efficiënt zijn, maar moeten ook rekening houden met verscheidene ontwerpbeperkingen, opgelegd door het netwerk, zodat de gelimiteerde middelen van de nodes efficiënt ingezet kunnen worden. Het hoofddoel van deze thesis is daarom om deze beperkingen in het design van gedistribueerde signaalverwerkingsalgoritmes voor WASN's te integreren.

De hoofdprobleemstelling van deze thesis, de huidige state of the art en het wiskundig kader voor (gedistribueerde) signaalschatting worden behandeld in de inleidende hoofdstukken. WSN's worden eerst geïntroduceerd in een gecentraliseerd kader waar alle informatie verzameld en verwerkt wordt in

een centrale locatie om signaalschatting mogelijk te maken, bijvoorbeeld voor spraakverbeteringstoepassingen. Dit gecentraliseerd kader wordt verder uitgebreid naar een gedistribueerd kader, waar de verwerking gedistribueerd wordt over de nodes en waar elke node een (mogelijks node-specifieke) signaalschattingstaak uitvoert. Het algemene doel van de in deze thesis ontwikkelde gedistribueerde algoritmes is om dezelfde schatting als in het gecentraliseerde kader te verkrijgen, en dit aan een gereduceerde hoeveelheid data-uitwisseling tussen de nodes.

Het eerste deel van deze thesis concentreert zich op een concrete toepassing en onderzoekt welke verbeteringen behaald kunnen worden op vlak van ruisonderdrukking en het behoud van spatiale cues in een binauraal ruisonderdrukkingsprobleem door het toevoegen van een enkele, (ver)afgelegen sensor (microfoon). Zulke binaurale ruisonderdrukkingsproblemen zijn belangrijk in bijvoorbeeld binaurale hoorapparaten of cochleaire implantaten. De performantietoename door het toevoegen van het signaal van een afgelegen sensor dient als motivatie voor de latere hoofdstukken waar vele nodes, die elk meerdere sensoren bevatten, beschikbaar zijn. In het geval waarbij meerdere afgelegen microfoons of nodes beschikbaar zijn wordt ook een methode besproken om te bepalen welke sub(set) van deze signalen of nodes de voordeligste keuze is in het schattingsprobleem. De bruikbaarheid (utility) wordt geïntroduceerd als een manier om het effect van de verwijdering van een node te begrenzen en kan nuttig zijn bij het toevoegen van nodes. De utility, eerst geïntroduceerd voor een gecentraliseerd scenario, wordt uitgebreid voor een gedistribueerd scenario. Echter, omwille van het gedistribueerde karakter van de signaalschatting, wordt in dit geval aangetoond dat de utility een grens voorstelt eerder dan een exacte hoeveelheid zoals in het gecentraliseerde scenario.

Het tweede deel van deze thesis introduceert lineaire compressie- en fusieregels voor gedistribueerde signaalschatting voor het gebruik in heterogene WSN's en WSN's met een gemengde topologie. Deze gemengde topologie WSN's zijn onderverdeeld in een set van kleinere substructuren die gebaseerd zijn op kliekjes- of clustervorming. Er wordt aangetoond dat de nodes in staat zijn te convergeren naar dezelfde optimale oplossing als wanneer ze alle ongecomprimeerde signalen van elke andere node zouden ontvangen. Ondersteunende technieken worden ook geïntroduceerd betreffende verschillende broadcast strategieën en topologievormingen.

In het derde deel van deze thesis implementeert elke node een nieuwe methode om zijn sensorsignalen lineair te comprimeren vooraleer naar de andere nodes in het WSN te verzenden. Dit leidt tot de introductie van het topologie-onafhankelijke gedistribueerd adaptief node-specifiek signaalschattingsalgoritme (TI-DANSE). TI-DANSE wordt eerst geïntroduceerd in een volledige verbonden topologie waarna ook aangetoond wordt dat de convergentie-eigenschappen

blijven gelden voor eender welke topologie zolang de nodes toegang hebben tot het gesommeerd signaal van het volledige netwerk. Een aantrekkelijke eigenschap van het TI-DANSE algoritme is dat, aangezien het steunt op het netwerkwijde gesommeerde signaal, het minder gevoelig is aan het falen van verbindingen en het bovendien ook toepasbaar is in WSN's met dynamische topologieën. Een methode om dit netwerkwijd gesommeerd signaal te berekenen wordt voorgesteld waarbij elke node maximaal twee transmissies moet verzorgen.

Tenslotte worden besluiten geformuleerd die de contributies van deze thesis herhalen alsook suggesties geven voor mogelijke richtingen voor toekomstig onderzoek.

Glossary

Acronyms and Abbreviations

AO	alternating optimization
AP	audio prostheses
BSS	blind source separation
BTE	behind-the-ear
CH	cluster head
CI	cochlear implant
DANSE	distributed adaptive node-specific signal estimation
DOA	direction of arrival
DSP	digital signal processor
EEG	electroencephalography
FC	fusion center
GSBCD	Gauss-Seidel block-coordinate descent
HA	hearing aid
HI	hearing impaired
HRIR	head-related impulse response
Hz	hertz
ILD	interaural level difference
ITD	interaural time difference

ITF	interaural transfer function
LMMSE	linear minimum mean squared error
LS	least-squares
MAP	maximum *a-posteriori* probability
ML	maximum likelihood
MN	member node
MSE	mean square error
MST	minimum spanning tree
MVDR	minimum variance distortionless response
MWF	multi-channel Wiener filter
MWF_η	multi-channel Wiener filter with partial noise estimation
R1-MWF	rank-1 multi-channel Wiener filter
RFC	receiver feedback cancellation
SNR	signal-to-noise ratio
SVD	singular value decomposition
T-DANSE	distributed adaptive node-specific signal estimation in a tree topology
TFC	transmitter feedback cancellation
TI-DANSE	topology-independent distributed adaptive node-specific signal estimation
VAD	voice activity detector
WASN	wireless acoustic sensor network
WBAN	wireless body area network
WL	widely linear
WL-MVDR	widely linear minimum variance distortionless response
WL-MWF	widely linear multi-channel Wiener filter
WMSN	wireless multimedia sensor network

WSN	wireless sensor network
cfr.	*conferatur*, compare with, see also
e.g.	*exempli gratia*, for example
i.e.	*id est*, that is
i.i.d.	independent and identically distributed
s.t.	subject to
w.l.o.g.	without loss of generality

Mathematical Notation

a	scalar
$\mathbf{a}$	vector
$\mathbf{A}$	matrix
$\mathbf{I}$	identity matrix
$\mathbf{0}$	null matrix or vector
$\mathbb{C}^L$	L-dimensional complex space
$\mathbb{R}^L$	L-dimensional real space
$\mathbb{N}$	the set of natural numbers
$\mathcal{E}\{\cdot\}$	mathematical expectation
$\mathrm{Tr}\{\cdot\}$	trace of a matrix
$\{\cdot\}^*$	scalar conjugate transpose
$\{\cdot\}^H$	vector or matrix conjugate transpose
$\{\cdot\}^T$	vector or matrix transpose
$\{\cdot\}^{-1}$	matrix inverse
$\mathrm{rank}(\mathbf{A})$	rank of a matrix $\mathbf{A}$
$\angle(\cdot)$	phase of a complex number
$\approx$	approximately equal to
$\forall$	for all

$\Leftrightarrow$	material equivalence
$\lim_{x \to a} f(x)$	limit of function f as x approaches a
$\ll$	much less than
$\lvert\cdot\rvert$	cardinality of a set or absolute value
$\lVert\cdot\rVert$	norm
$\lVert\cdot\rVert_F$	Frobenius norm
$\mathcal{N}_k$	the set of neighbors of a node k, excluding itself
$\Rightarrow$	material implication
$\triangleq$	is defined as
$\arg\min_x$	argument of the minimum
$\underset{x}{\text{minimize}}$	minimize over x
$diag\{\cdot\}$	diagonal elements of a matrix
mod	modular operator
$\setminus$	set-theoretic complement
$\cup$	set-theoretic union
$\emptyset$	empty set
$\in$	set membership
$\notin$	set membership, is not an element of
$\subseteq$	subset

Contents

Abstract i

Korte Inhoud v

Glossary ix

Contents xiii

List of Figures xxi

List of Tables xxvii

1 Introduction 1

1.1 Single-channel noise reduction 3

1.2 Multi-channel noise reduction 5

1.2.1 Beamforming 5

1.2.2 Optimal linear filtering 7

1.3 Wireless sensor networks 7

1.4 Design challenges of WSNs 10

1.4.1 Communication bandwidth 11

1.4.2 Network lifetime 12

1.4.3 Topology control and maintenance 12
1.4.4 Scalability 14
1.4.5 Self-healing 14
1.5 Overview of thesis 15
1.5.1 General overview 15
1.5.2 Thesis outline 16

2 Centralized and distributed signal estimation 19
2.1 Data model 19
2.2 Multi-channel Wiener filtering 20
2.2.1 Special case : single node-specific desired signal 21
2.2.2 Estimation of signal statistics 21
2.3 Centralized MWF for node-specific signal estimation 23
2.3.1 Relationship between node-specific desired signals and estimators 25
2.3.2 Widely linear filtering 25
2.4 The distributed adaptive node-specific signal estimation (DANSE) algorithm 26

Part I Noise reduction with a remote signal and subset selection 31

3 Binaural noise cue preservation in a binaural noise reduction system with a remote microphone signal 33
3.1 Introduction 33
3.2 Binaural data model 35
3.3 Performance measures 38
3.3.1 Output SNR 38
3.3.2 Binaural cues 39

3.4 Data model with a remote microphone signal 42
3.5 Performance measures with a remote microphone signal 44
3.5.1 Output SNR with a remote microphone signal 44
3.5.2 Binaural noise cues with a remote microphone signal . . 45
3.6 Adaptive partial noise estimation 46
3.6.1 Increased output SNR 47
3.6.2 Increased binaural noise cue preservation 47
3.7 Simulations 48
3.7.1 Performance measures with changing η and $\mathrm{SNR}_{\mathrm{IN},REM}$ 49
3.7.2 Adaptive η 50
3.7.3 Influence of noise source position 52
3.8 Conclusion 57

4 Greedy distributed node selection for node-specific signal estimation in wireless sensor networks 59

4.1 Introduction 59
4.2 Utility 61
4.2.1 Utility for node removal 62
4.2.2 Utility for node addition 66
4.2.3 Definition of a common network-wide utility 67
4.2.4 Greedy centralized node selection 69
4.3 Distributed computation of utility bounds 70
4.3.1 Node removal : utility upper bound 72
4.3.2 Node addition : utility lower bound 73
4.3.3 Greedy distributed node selection 74
4.4 Simulations 74
4.4.1 Batch mode 74
4.4.2 Adaptive implementation 81

4.5 Conclusion 86

Part II Mixed-topology distributed signal estimation and supporting techniques 87

5 Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks 89

5.1 Introduction 89

5.2 Heterogeneous WSNs 91

5.2.1 Intra- and inter-cluster communication 93

5.2.2 Data-driven signal flow 94

5.2.3 Node-specific local parameter updating 95

5.2.4 Convergence and optimality 98

5.3 Mixed-topology ad-hoc WSNs 99

5.3.1 Distributed clique formation 99

5.3.2 Intra- and inter-clique communication 101

5.3.3 Data-driven signal flow 102

5.3.4 Convergence and optimality 103

5.3.5 Changing topologies 106

5.4 Simulations 108

5.4.1 Single sensing environment 108

5.4.2 Monte-Carlo simulations 111

5.5 Conclusion 115

6 Alternative broadcast strategies and topology control 117

6.1 Introduction 117

6.2 The T-DANSE algorithm 119

6.2.1 Fusion flow 119

6.2.2 Diffusion flow 120
6.2.3 Feedback cancellation 120
6.2.4 Transmitter feedback cancellation (TFC) 121
6.2.5 Receiver feedback cancellation (RFC) 121
6.2.6 Parameterized solution space 121
6.3 Modified signal flow in the T-DANSE algorithm 122
6.3.1 Root node broadcast signal and feedback cancellation . 123
6.3.2 Reduction in input-output delay and transmission energy 124
6.4 Estimation errors of second-order statistics 125
6.5 Weighted eigenvector centrality 125
6.6 Simulations 128
6.6.1 Simulations using the modified root node broadcasting strategy 128
6.6.2 Topology construction 130
6.7 Conclusion 130

Part III Topology-independent distributed signal estimation 133

7 Topology-independent distributed adaptive node-specific signal estimation 135

7.1 Introduction 135
7.2 TI-DANSE in a fully connected WSN 137
7.2.1 Convergence and optimality 139
7.3 TI-DANSE without topology constraints 140
7.3.1 Data-driven signal flow 141
7.3.2 Equivalence to fully connected topology 143
7.4 Simulations 144

7.4.1 Single sensing environment 144
7.4.2 Comparison of T-DANSE and TI-DANSE with dynamic connections 147
7.4.3 Monte-Carlo simulations 148
7.5 Conclusion 150

8 Conclusion **151**

A Appendix to Chapter 2 **157**
A.1 On the use of time-domain widely linear filtering for binaural speech enhancement 157
A.2 Linear minimum variance distortionless response filter 158
A.3 Widely linear filtering 160
A.3.1 Widely linear multi-channel Wiener filter 160
A.3.2 Widely linear minimum variance distortionless response filter 161
A.4 Equivalence of the MVDR scaling factors under a Rank-1 model 162
A.4.1 Linear MVDR scaling factor 163
A.4.2 Widely linear MVDR scaling factor 164
A.5 Computational complexity 164
A.6 Conclusion 165

B Appendix to Chapter 3 **167**

C Appendix to Chapter 4 **169**
C.1 Relationship between LMMSE cost matrices between nodes . . 169
C.2 Calculation of optimal fall-back estimator and utility 170
C.3 Utility for node addition 172

D Appendix to Chapter 5 **173**

E Appendix to Chapter 7 **177**

Bibliography **187**

Curriculum Vitae **205**

Publication List **207**

List of Figures

1.1 Single-channel noise reduction. 4

1.2 Multi-channel noise reduction. 6

1.3 A depiction of a WASN with a set of distributed devices, a single desired speaker and background noise. 9

1.4 A WSN where all of the information is sent and processed at a central location or fusion center (FC). 9

1.5 The encoding, decoding, and filtering process of a node. 11

1.6 Two example topologies possible when there are no power constraints for communication with the nodes (a) and when the nodes only communicate with their nearest neighbors (b). . 13

1.7 Scalability difference between a relay node and a node that employs a local fusion rule where the line width is indicative of the size of the transmitted signal. 14

2.1 Centralized MWF for a $K = 3$ WSN. 24

2.2 The DANSE algorithm for a $K = 3$ WSN. 27

3.1 A general binaural noise reduction system. 35

3.2 The general binaural noise reduction system with a remote microphone signal. 42

3.3 $\Delta\text{SNR}_{\text{OUT},R}$ for various values of η and $\text{SNR}_{\text{IN},REM}$ along with target of $\frac{1}{\eta^2}$. 50

3.4 The noise ILD and relative noise ITD error for a single frequency ($\omega_l = 1500\ Hz$) for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$. 51

3.5 Adapted η and $\Delta\mathrm{SNR}_{\mathrm{OUT},R}$ based on the noise ILD constraint for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$. Note that the x-y axis of (a) are flipped for an easier graphical representation. 53

3.6 The noise ILD and relative noise ITD error based on the noise ILD constraint for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$. 54

3.7 Adapted η based on the $\Delta\mathrm{SNR}_{\mathrm{OUT},L}$ constraint and the resulting $\Delta\mathrm{SNR}^{e}_{\mathrm{OUT},L}(\eta_e)$ (Note that the x-y axis of (a) are flipped for an easier graphical representation). 55

3.8 The $\Delta\mathrm{SNR}_{\mathrm{OUT},L}$ and noise ILD error for a single frequency ($\omega_l = 1500\ Hz$) for various values of θ_n where $\theta_s = 0^\circ$ and η is at a constant value, namely, $\eta = 0.2$. 56

4.1 Computational complexity for finding $U_{k_{-q}}$ using the naive approach compared to (4.12). 65

4.2 Comparison of the LS cost for a centralized (a) and distributed (b) node removal process. 78

4.3 Utility upper bound in a distributed scenario compared to centralized (optimal) utility. 79

4.4 Utility upper bound in a distributed scenario compared to centralized (optimal) utility with $M_k \geq K(Q-1)$. 80

4.5 Distributed scenario where a node is kept in the network that would have been removed if the centralized utility was used. . . . 81

4.6 Comparison of centralized (a) and distributed (b) node addition process. 82

4.7 Depiction of simulated scenario. The network contained 30 nodes each with 5 sensors (). There were two sources (,) that follow paths indicated by the colored lines respective of the color of the source. There were also five spatially located white noise sources present (). 83

4.8 Comparison of the LS cost (a) and the total number of active nodes (b) in the centralized scenario with and without node selection and the distributed scenario with node selection. . . . 84

4.9 Node selection during various times in the estimation process using distributed utility bounds. The active nodes are represented as (●) and the nodes that are only receiving signals from the other nodes and not transmitting their $\mathbf{z}_k^i$ signals are represented as (●). 85

5.1 A hierarchical representation of the envisaged heterogeneous WSN. The top layer consists of a set of cluster heads (CHs) that form a fully connected network with one another. The member nodes (MNs) of each cluster are formed in, e.g., a tree topology where the CHs act as root nodes for their individual clusters. . 93

5.2 Data-driven flow in a heterogeneous WSN utilizing the DANSE algorithm : The MNs of each cluster send observations of their intra-cluster signals (5.1) to their respective CH (a). The CHs broadcast observations of their inter-cluster signals (5.2) to the other CHs in the WSN (b). The CHs broadcast observations of their intra-cluster signals (5.3) to their respective MNs where the MNs use (5.4) to cancel out their feedback portion (c). 95

5.3 A depiction of the partitioning of an ad-hoc configuration (a) into a set of non-overlapping cliques (b). This is then further abstracted to a tree topology (c). 100

5.4 A depiction of the data-driven signal flow of DANSE in a mixed-topology WSN where node 5 is the closest to the root node. . . 104

5.5 A possible updating order for the mixed-topology (a) is $P = [5, 4, 3, 2, 1, 2, 4]$ and for the tree topology (b) is $P = [5, 4, 2, 3, 2, 1, 2, 4]$. 105

5.6 (a) A simulated environment with 2 desired signal sources (■), 4 noise sources (◆) and a WSN consisting of 17 nodes where each node has 3 sensors and where the root node is represented as ●. The nodes are grouped in $N = 5$ cliques containing 3 nodes each and two non-clique nodes. (b) A spanning tree using the original node positions given in (a). 109

5.7 The LS cost of the root node (a) and the sum of LS cost of all of the nodes in the network (b) versus the number of iterations for DANSE-FC with round robin updating (DANSE-FC-RR), and a tree updating order (DANSE-FC-TUO), T-DANSE, DANSE-MT using an update order as described in Section 5.3.3 (DANSE-MT-MUO), and a tree updating order (DANSE-MT-TUO). 112

5.8 Sum of the energy used for transmission using DANSE-FC, T-DANSE, and DANSE-MT. 113

5.9 The median of the sum of the LS cost of all of the nodes in the network versus the number of iterations for the previously defined DANSE algorithms. 114

5.10 The average sum of energy used for transmission for DANSE-FC, T-DANSE and DANSE-MT for the Monte-Carlo simulations. . 114

6.1 Signal flow in the original and modified T-DANSE algorithm where the root node (node 2) is represented as . The solid arrows in (b) indicate the fusion flow which is equivalent to the original T-DANSE formulation (a). The dotted arrows in (b) indicate the broadcast signal flow from the root node to all other nodes in the WSN. 122

6.2 Performance measures using the modified broadcasting strategy. 129

6.3 A WASN with a desired speech source (), a spatially located white noise source () and ten nodes () with corresponding network topologies. 131

7.1 The proposed filtering and broadcast scheme for node k. 138

7.2 A WSN with an ad-hoc topology (a). The WSN with a pruned tree topology where the suppressed links are represented by dashed lines and where active links represented as bold solid lines (b). 142

7.3 The signal flow of the TI-DANSE algorithm for a non-root node k. Note that for a leaf node, $\mathcal{T}_k \backslash \{q\} = \emptyset$ and the transmitted signal, $\mathbf{z}_k^i$, is equivalent to that of (7.5). 144

7.4 A simulated environment with 2 sources (), 4 spatially correlated white noise sources (), and 15 nodes ($K = 15$) each with $M_k = 5$ sensors. The dashed lines indicate the ad-hoc connections and the solid lines represent the MST formed using the Euclidean distance between nodes as edge weights. 146

7.5 Cost of the DANSE, T-DANSE, TI-DANSE-FC and TI-DANSE-T algorithms versus the number of iterations. 147

7.6 Cost of the T-DANSE and TI-DANSE-T algorithms with randomized links in the tree. 148

7.7 Normalized cost of the DANSE, T-DANSE and TI-DANSE algorithms versus the number of iterations averaged over 1000 Monte-Carlo runs. 149

A.1 Graphical representation between the linear MWF and linear MVDR solutions. 159

A.2 Graphical representation between the WL-MWF and linear WL-MVDR solutions. 163

List of Tables

2.1 The DANSE algorithm in a fully connected WSN 28

4.1 Greedy centralized node selection 71

4.2 Greedy distributed node selection 75

5.1 The DANSE algorithm in a heterogeneous WSN 97

6.1 SNR_{IN} (dB), eigenvector centrality, SNR_{OUT} (dB) using eigenvector centrality, SNR_{IN} weighted eigenvector centrality and SNR_{OUT} (dB) using SNR_{IN} weighted eigenvector centrality for each node. Root nodes for each centrality measure are given in bold. 132

7.1 TI-DANSE in a fully connected WSN 139

E.1 Alternating optimization procedure 179

E.2 Description of the D-AO3 algorithm in a fully connected WSN 183

Chapter 1

Introduction

With the current trend of miniaturization, many electronic devices have increased functionality due to being equipped with a variety of sensors to capture information of the surrounding environment. The digital signal processors (DSPs) which are used to process this information are becoming more powerful and energy efficient, allowing for more sophisticated tasks to be performed while reducing the overall power consumption. Recent trends have been incorporating many of these devices (or nodes) together which work in unison over wireless channels, in order to accomplish a predefined task. This type of cooperative sensing and communication between devices form the basis of a wireless sensor network (WSN).

WSNs have many attractive attributes when compared to other traditional sensing and monitoring systems, such as but not limited to: low cost of deployment when compared to equivalent wired systems, the ability to monitor large areas and continually perform a network wide task without supervision, resilience to single point failure and the ability to be placed in hostile or inaccessible environments [1]. These attributes make them ideal for a variety of civilian and military uses [2–5]. These include: tracking and surveillance [6], environmental and seismic activity monitoring [7, 8], and health care delivery in wireless body area networks (WBANs) [5].

Due to the increased processing capabilities of the devices, WSNs can also be tasked with acquisition and estimation of multimedia content, forming so-called wireless multimedia sensor networks (WMSNs) [9, 10]. WMSNs differ from most other WSNs, in that the signals observed often have characteristics that fluctuate much quicker in time which require significantly faster sampling rates, and hence increased transmission and computation costs. A direct consequence

of an increase in transmission rates is a higher power consumption of the nodes.

Recent research has started to exploit a subclass of WMSNs with regard to acoustic signals forming a wireless acoustic sensor network (WASN) [11–14]. One application of WASNs is performing noise reduction by using spatially distributed microphones in an acoustic environment. For example, in so-called binaural noise reduction, two multi-microphone audio prostheses (APs), such as hearing aids (HAs) or cochlear implants (CIs), share microphone signals with one another over a wireless link to provide an enhanced version of a speech signal to the user [15, 16]. The performance of this binaural noise reduction system, or two node WASN, can further be enhanced by the incorporation of an external (set of) microphone(s) in the environment, thereby forming a larger WASN [12, 17, 18].

These external microphones can be statically deployed, where the locations of the desired and noise sources are assumed to be known. This is often done in noise reduction algorithms for automotive and aircraft design, where microphones are placed close to the engine and used as a reference to cancel out their sound [19]. However, this relies on *a-priori* knowledge of the sensing environment, i.e., the locations of the noise and speech signals. A more versatile approach is of course by allowing the microphones to have an unknown or ad-hoc deployment where no assumptions are made on the sensing environment.

In order to fully exploit the versatility of such a deployed WASN, and WSNs in general, there are a multitude of design challenges. Solving a variety of these design challenges is the main focus of this thesis. While many of the contributions of this thesis are posed in a more general framework, i.e., not specifically pertaining to acoustic applications, as long as the underlying assumptions of the signal characteristics are met, they are easily applicable to acoustic problems.

This chapter introduces the problem of estimating a desired signal that has been corrupted by additive noise and reviews the current state of the art pertaining to distributed signal processing algorithms for WSNs. A basic description of the problem statement is given in Section 1.1 where only single-channel noise reduction is performed, i.e., there is only one sensor available to capture a signal of interest from the sensing environment. Various methods for performing the noise reduction in the single-channel case will be discussed, albeit briefly. The drawbacks associated with the single-channel noise reduction techniques will be highlighted, such as the unavoidable introduction of signal distortion, which provides the motivation for the use of multi-channel noise reduction techniques.

Section 1.2 introduces multi-channel noise reduction techniques, discussing their ability to exploit the spatial characteristics of a set, or array, of sensors, thereby increasing the noise reduction performance compared to the single-channel

techniques. The spatial information can be used to form a so-called beamformer, which can be accomplished in a fixed or adaptive fashion. Optimal linear filtering will also be discussed, which takes the form of the multi-channel Wiener filter (MWF). Besides being an optimal filter, the MWF has other desirable properties, such as the ability to control the amount of signal distortion compared to the amount of noise reduction, which will be highlighted.

However, if the sensor array used for multi-channel noise reduction lacks spatial diversity, e.g., a size constrained device where the sensors are placed close together, or due to poor position or placement of the array, the performance can be severely hampered. These drawbacks can be overcome by incorporating other arrays in the sensing environment, which serve to further increase the spatial diversity of the sensors despite device size constraints, as well as decrease the effects of poor placement of a single array.

In Section 1.3, a set of nodes each containing an array of sensors is considered, which communicate over wireless channels and form the basis of a WSN. WSNs will first be introduced in a centralized fashion, where all of the information is aggregated and processed at a central location or fusion center. However, due to drawbacks such as large computational burdens, large communication bandwidth and the vulnerability to single point failure, it will be discussed how WSNs can operate in a distributed fashion, i.e., where each node in the WSN contributes to the processing so that there is not a reliance on a fusion center.

In order to exploit the versatility of WSNs there are several key design challenges that are addressed in Section 1.4. While an exhaustive list is not presented, the design challenges given serve as the main research objectives of this thesis.

Finally, in Section 1.5, an outline of the thesis is given where the contributions of each chapter are summarized.

1.1 Single-channel noise reduction

Noise reduction is a fundamental challenge that occurs in a variety of signal processing domains [7, 20–27] which, for a single-channel case, is depicted in Figure 1.1. It is assumed that there is a device with a sensor to capture measurements in a sensing environment and which is equipped with a DSP in order to analyze and process the captured measurements. The sensing environment consists of a desired signal, or set of desired signals along with undesired additive noise components that impinge on the sensor of the device. This sensor can, for example, be a microphone that is capturing an audio signal with background noise. This input signal can be characterized by its

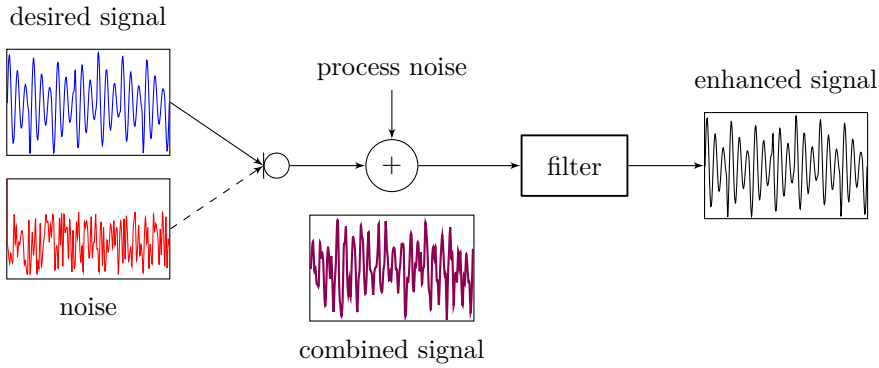


Figure 1.1: Single-channel noise reduction.

signal-to-noise ratio (SNR), which is the ratio of the input desired signal power to the input noise power.

Once the signal is captured by the sensor, additional sensor or process noise may also be present, which can be indicative of internal thermal or electrical noise. This signal is then filtered in order to produce an estimate of the desired signal as it impinges on the sensor without noise. It should be noted that the noise reduction covered in this thesis focuses on estimating the desired signal component as it impinges on the sensor of the device and not the original signal itself. Other methods, such as blind source separation (BSS), look to estimate this unobservable, or blind source [28, 29] and will not be covered. The aim of the filter, in this noise reduction context, is then to provide a higher SNR at the output than at the input.

The question then becomes how to design and implement the filter in order to provide an estimate of the desired signal while suppressing the noise. There are various methods to perform single-channel noise reduction such as spectral subtraction, subspace methods and statistical modeling [27, 30–35]. While filtering methods can be performed in the time domain, the signals are often converted to their frequency domain representations, which allows for the noise reduction algorithms to take advantage of the different spectral characteristics of the desired signal and noise. These methods therefore rely on collecting and estimating the underlying statistical properties of the noise and possibly the desired signal.

The various methods provide advantages and disadvantages when compared with one another, which can range from simplicity to greater noise reduction performance. It is apparent from Figure 1.1, that only a noisy version of

the signal is available at the device. In fact, single-channel noise reduction techniques only consider the temporal and spectral characteristics of the signal, which has the negative byproduct of introducing distortion to the estimate of the desired signal [32, 36] such as musical noise [37, 38].

Therefore, single-channel noise reduction techniques introduce trade-offs between the output SNR, signal distortion and other artifacts that are introduced during the filtering process. However, if an auxiliary sensor was available to include in the filtering process it could be used as a possible noise reference, which would not only allow for an increase in the noise reduction but at the same time provide a decrease in the amount of signal distortion. Therefore, in the following section, it is discussed how the disadvantages of the single-channel noise reduction can be overcome by the addition of one or more sensors in the filtering process.

1.2 Multi-channel noise reduction

The single-channel noise reduction presented in Section 1.1 is now extended to the case where the device contains multiple sensors and performs multi-channel noise reduction. It should be noted, however, that many of the single-channel noise reduction techniques previously mentioned can easily be extended to that of the multi-channel case. A filter is now applied to each channel, or sensor signal, where the resulting filtered signals are summed to produce an estimate of the desired signal. A depiction of this envisaged multi-channel noise reduction is given in Figure 1.2, where the process noise is not shown for brevity. Again it should be noted that the noise reduction aims to provide an estimate of the desired signal component that is acquired at one or more of the sensors and not the original signal(s) itself.

1.2.1 Beamforming

As stated in the previous section, single-channel noise reduction techniques often introduce distortion and artifacts into the enhanced signal. Multi-channel noise reduction techniques can mitigate these effects by exploiting the spatial and temporal structure of the sensor signals [32, 39, 40]. This spatial separation can be used to infer the spatial characteristics of the sensing environment and perform so-called beamforming [41], which often leads to not only increased noise reduction performance but also decreased distortion in the estimated desired signal. Beamforming essentially controls the directivity pattern of the

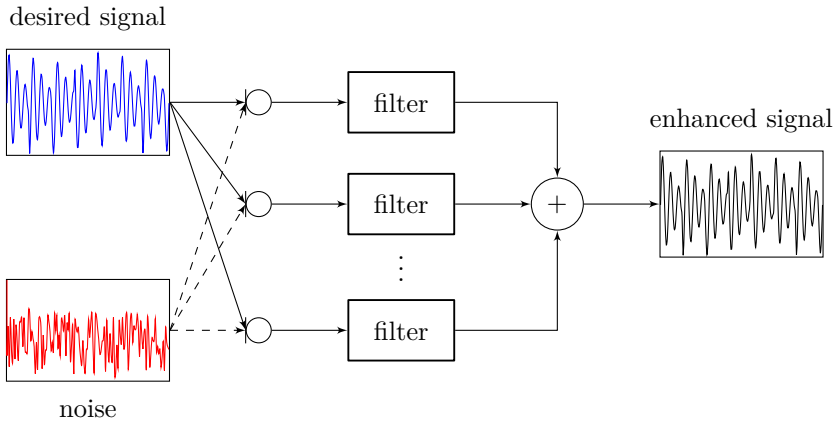


Figure 1.2: Multi-channel noise reduction.

sensors, which allows for information from the direction of the desired signal to pass, while blocking the direction of the noise.

These beamformers can be placed into two general categories, fixed or adaptive. Fixed beamformers are data independent as they rely on *a-priori* assumptions of the desired signals and noise positions as well as the array geometry of the sensors [42, 43]. However, they suffer from performance degradation due to model errors, i.e., if the *a-priori* assumptions are violated, or if there is calibration mismatch from the assumed array geometry. Adaptive beamformers, on the other hand, are data dependent and can be implemented with only minor assumptions [44–48] which allow for non-stationary signals and, in some cases, allow for unknown array geometries.

Nonetheless, when the desired signal and noise originate from the same direction, beamformers are unable to exploit the spatial characteristics and must resort to only using the temporal and spectral characteristics. This then amounts to a single-channel case where trade-offs are introduced between the output SNR and signal distortion. However, this may be mitigated by additional sensors that capture additional characteristics of the desired signal and noise as well as collect more spatial information of the environment (a method for this via a remote sensor is presented in Chapter 3).

1.2.2 Optimal linear filtering

There is another class of multi-channel noise reduction techniques that are based on minimizing the mean square error (MSE) between the desired signal and the filtered signal. If linear estimation is considered, then the optimal filter in a linear minimum mean squared error (LMMSE) sense is referred to as the multi-channel Wiener filter (MWF) and will be explained in detail in Section 2.

An attractive feature of the MWF is that it does not rely on any *a-priori* assumptions of the signals or sensor placement. It instead relies on the second-order statistics, i.e., the mean and correlations, of the signals. While the statistics of the signals are assumed to be stationary, or time invariant, this is often relaxed to quasi-stationarity, or slowly time varying, which will be described in Section 2.2.2. There are several extensions to the frequency domain MWF focusing on controlling the signal distortion and residual noise that will be described more in depth in the following chapters.

Due to its optimality, the ability to be implemented in the frequency domain, and the ability to control the amount of distortion compared to the amount of noise reduction, the MWF is adopted as the basis for the algorithms presented in this thesis.

1.3 Wireless sensor networks

While increasing the amount of sensors leads to an increase in the noise reduction performance [12, 32, 49] there are instances where it is impractical or impossible to place more sensors on a device. For example, many APs come with multiple microphones, typically to capture front facing speech and cancel out background noise from the auxiliary microphones. However, due to the size constraint of the device adding additional microphones is unfeasible. The noise reduction performance can also be hampered by environmental factors such as poor sensor placement, e.g., when the speech and noise come from the same direction.

Therefore, if the multi-channel node was able to receive auxiliary channels from another location, not only would more spatial information of the sensing environment be captured but it could also receive signals from a more favorable location. Expanding on a similar motivation from single-channel noise reduction to multi-channel noise reduction, the sensing environment is now considered to contain two or more multi-sensor (channel) nodes.

These nodes could be connected in a traditionally wired manner; however, this poses not only scalability issues, but also restricts the sensor locations to

fixed positions. These problems coupled with large sensing environments, e.g., agricultural monitoring, will typically result in high installation and maintenance costs [19]. Although, with the current state of wireless communication, the same connectivity and communication can be achieved over wireless links, while still maintaining the same performance of larger, more sophisticated wired sensing networks. The nodes can therefore share their sensor signals with one another over wireless channels where this type of cooperative sensing and communication between devices form the basis of a WSN. In the WSNs envisaged for this thesis, it is assumed that each node contains multiple sensors as presented in Figure 1.2. Along with their own sensor signals, the nodes now also receive signals from other nodes in the WSN.

Each node is tasked with estimating a node-specific desired signal as observed by their own sensors, i.e., a node only tries to estimate a desired signal as it impinges on its local sensors. This type of node-specific estimation is important if spatial information needs to be retained. For example, in a WASN, where each node is tasked with estimating a node-specific audio signal, the nodes can perform local noise reduction by incorporating (pre-processed) microphone signals from other nodes [14]. Estimating the signals as they impinge on the sensors of a node allows the node-specific signal estimates to not loose the spatial cues for directional hearing [12, 14, 50]. A depiction of a WASN is given in Figure 1.3 where there is a desired speaker (signal) and background noise.

Each node could transmit its local sensor observations to a central location or fusion center in order to perform estimation of a desired signal as in Figure 1.4. However, as the number of devices or the number of sensors per node increases, it quickly becomes difficult to transmit all of the sensor signals to the fusion center. This is due, in part, to a significant increase in the amount of communication bandwidth necessary and may also require a substantial amount of transmission energy if there is a large distance between the nodes and the fusion center.

In order to reduce the communication bandwidth, the nodes can employ some type of compress-and-fuse type strategy, along with the use of relays in order to disseminate signals through the WSN [51, 52]. However, since the estimation problems are disjoint, i.e., each node compresses its own signal without information from the other nodes, this type of strategy is often suboptimal. The fusion center is therefore unable to reach the same solution as if it were to receive all of the uncompressed signals from the nodes. Another drawback of centralized processing is that it is prone to single point failure, i.e., if the fusion center fails then no information can be processed.

Therefore, instead of requiring that each node transmits all of its observations to a fusion center, either directly or by the use of relays, it is beneficial to

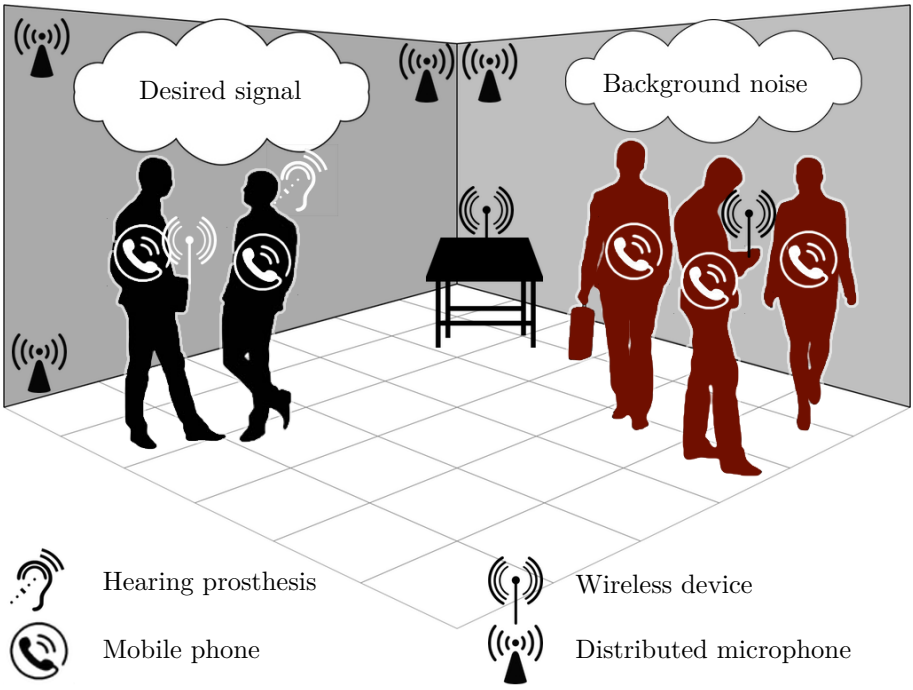


Figure 1.3: A depiction of a WASN with a set of distributed devices, a single desired speaker and background noise.

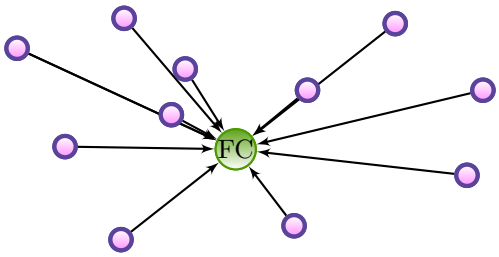


Figure 1.4: A WSN where all of the information is sent and processed at a central location or fusion center (FC).

have a distributed[1] WSN framework where the computational load may be divided among the nodes while still being able to reach the same solution as with a fusion center. The devices in this distributed context could reduce their transmission burden by only transmitting to their direct neighbors. As a general aim, the algorithms developed in this thesis look to perform the same estimation as in the centralized case, where each node would have access to all of the sensor signals of the entire network.

To this end, the so-called distributed adaptive node-specific signal estimation (DANSE) algorithm is used as the basis of many algorithms in this thesis [53–56]. In the DANSE algorithm, the nodes update their local compression rules to compress their sensor signal observations and transmit this compressed signal to other nodes in the WSN. Under certain conditions, it has been shown that, even though the nodes transmit a compressed version of their sensor signal observations, each node converges to the same solution as if it had access to all of the uncompressed signals in the WSN, i.e., the centralized case. Due to its convergence properties, the DANSE algorithm has been used in a variety of applications such as speech enhancement and direction of arrival (DOA) estimation in WASNs, binaural hearing aids [11, 12, 57, 58] and artifact removal in wireless electroencephalography (EEG) sensor networks [59].

While the WSNs envisaged in this thesis are tasked with estimating node-specific desired signals, there is another large body of work for WSNs that are tasked with estimating a parameter vector of interest. While not specifically addressed in this thesis, it is important to mention for a complete overview of the current state of the art. The parameters of interest in these WSNs typically change much slower in time, when compared to signal estimation, which allows the WSN to take advantage of local processing by consensus, incremental, or diffusion based algorithms [60–65]. However, it should be noted, that even in the case of node-specific parameter estimation [66, 67], the underlying problem formulations are quite different.

1.4 Design challenges of WSNs

There are a multitude of implementation aspects and design challenges associated with algorithm development for WSNs [9]. While the following is not meant to be an exhaustive list it serves as the main research aims of this thesis. Many of

[1] It should be noted that while the nodes are physically distributed through the sensing environment, a distributed WSN in the context of this thesis means that along with being physically distributed the computation is divided among the nodes.

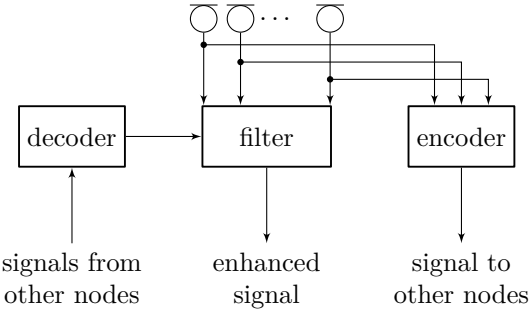


Figure 1.5: The encoding, decoding, and filtering process of a node.

the design challenges are complementary, in that by mitigating the effects of one will also aid the others.

1.4.1 Communication bandwidth

The nodes in the envisaged WSNs have three main tasks: acquisition, processing and transmission of signals. Depending on the application upwards of 80% of the total power consumed on a per node basis is used for transmission of the signals [68]. Since the nodes are often assumed to have finite energy budgets, reducing the transmission burden is a primary design challenge. The devices may also be bandwidth constrained, due to either network or device constraints, e.g., it may not be possible for a node to transmit the full set of its sensor signal observations.

In order to reduce the communication bandwidth, some type of encoding or compression can be applied to the sensor signals observations to reduce the size of their transmitted signal. The nodes may also apply some type of decoding or decompression to the received signals from other nodes in the WSN. The node's own sensor signals and received signals from other nodes are then filtered to produce an enhanced signal. A basic depiction of the filtering, encoding (compression) and decoding (decompression) is given in Figure 1.5.

Several methods propose applying a local dimensionality constraint, which is a by-product of a bandwidth constraint and transmit this compressed signal, either in a centralized or distributed scenario [51, 52, 69, 70]. However, this imposed dimensionality constraint on the transmitted signals can result in reduced estimation performance, i.e., suboptimal solutions that are not equivalent to the centralized solution.

1.4.2 Network lifetime

The nodes that constitute the WSN, typically have finite energy budgets, which restricts the network lifetime. In order to prolong the lifetime of the WSN, a subset of nodes can be placed in an inactive or sleep mode. However, placing a subset of nodes in an inactive state should not considerably affect the signal estimation performance of the WSN. Indeed, due to a dense deployment of nodes, or situations where there is an appreciable overlap between the sensor signals, not all of the nodes signals are necessary to perform signal estimation as they contain redundant information.

However, finding the optimal subset is often an intractable task, due to its combinatorial nature, therefore a way to assess the signals in their order of importance to the current estimation is essential. Several statistical methods for subset selection have been proposed based on maximum *a-posteriori* probability (MAP) and maximum likelihood (ML) estimators [71].

Even when the optimal subset is found using such techniques as branch and bound, the run time of the algorithms are often too long and computationally intensive for practical use in the envisaged distributed WSNs. Typically, due to the combinatorial nature, some type of heuristic method is usually employed [72]. These heuristics can place bounds on the performance of the WSN, ensuring that the removal or addition of nodes do not exceed some predefined threshold [73].

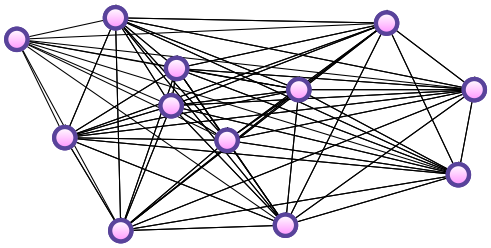
1.4.3 Topology control and maintenance

The links that the nodes use to communicate with one another form the communication backbone or topology of the WSN. This topology plays an important role in resource allocation as it can affect how much power the nodes utilize to transmit to one another and the data flow through the WSN.

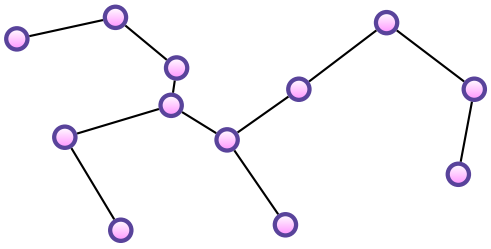
The WSNs can be formed with some upper level coordination, in order to exhibit a certain topology[2], based on different design constraints in mind such as : energy conservation [74,75]; communication bandwidth [76,77]; and security [78].

Figure 1.6 shows two possible topologies constructed from the same WSN with and without a transmission power constraint on the nodes. In Figure 1.6a there is no transmission power constraint and the nodes can communicate directly with every other node in the WSN forming a fully connected topology. In Figure 1.6b, the nodes only communicate with their nearest neighbors, which corresponds to a minimum transmission power constraint, forming a minimum

[2]It is noted that the topology does not necessarily need to correspond to the physical location of the devices.



(a) A fully connected topology.



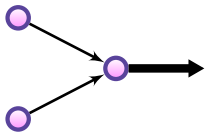
(b) A minimum spanning tree topology.

Figure 1.6: Two example topologies possible when there are no power constraints for communication with the nodes (a) and when the nodes only communicate with their nearest neighbors (b).

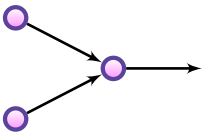
spanning tree (MST). In the case of a heterogeneous WSN, i.e., where some of the nodes have greater processing capabilities as well as increased energy budgets, it can benefit the resource allocation of the WSN to place more of the burden on these larger capacity nodes [79, 80].

Adherence to a predefined topology, however, can affect the performance of a WSN, especially in WSNs with mobile nodes. Without some upper level coordination or design goal in mind the WSN can therefore be formed in an ad-hoc fashion. Ad-hoc WSNs do not rely on a fixed infrastructure which allows for greater flexibility in node deployment. However, due to this lack of fixed infrastructure, ad-hoc WSNs can suffer from scalability issues due to the fact that nodes must monitor and retain network wide routing tables which can require a significant portion of the available network resources [81, 82].

Various topologies, from fully connected to ad-hoc will be used throughout this thesis with different design goals in mind. In the distributed signal estimation case, different broadcast strategies will be developed depending on the underlying topology of the WSN.



(a) A node that acts as a relay.



(b) A node that utilizes a local fusion rule to generate a combined signal.

Figure 1.7: Scalability difference between a relay node and a node that employs a local fusion rule where the line width is indicative of the size of the transmitted signal.

1.4.4 Scalability

The algorithms designed should be scalable, in that they should have the same performance and efficiency regardless of the size of the WSN. However, there are many aspects that affect the scalability of an algorithm. The addition of nodes should not significantly increase the amount of transmitted signals nor should the computational complexity be increased. By limiting these effects, the energy usage of the nodes will also be unaffected by the size of the WSN.

In the case of a centralized or fully connected WSNs each time a node is added the communication bandwidth increases. Instead of requiring a direct connection, the nodes could accomplish this data dissemination via relays to ensure that the signals are diffused through the entire WSN [83, 84] as depicted in Figure 1.7a. However, the number of direct neighbors a node has will directly affect the amount of signals that need to be relayed. A more elegant and efficient method can be accomplished by allowing nodes to utilize some type of local combination, or fusion rule, that combines their local signals with the received signals of their neighbors as depicted in Figure 1.7b.

1.4.5 Self-healing

It is unavoidable that some of the nodes will fail or that links between nodes will fail, thereby changing the topology of the WSN. It is apparent from the centralized processing depicted in Figure 1.4, that this type of WSN is susceptible to single point failure, i.e., if the central node, or fusion center, fails then the WSN is unable to process the signals. In the MST depicted in Figure 1.6b, if one of the links fail, then the WSN is split into two distinct subnetworks that are unable to share information with one another which will also affect the performance of the algorithms. Therefore, the WSN should have self-healing

properties, in that if a node or links fails, the nodes can reorganize themselves and still be able to accomplish their signal estimation.

1.5 Overview of thesis

1.5.1 General overview

Along with the introductory chapters that make up the first part of this thesis, as well as the final concluding chapter, this thesis is composed of three main parts.

The **first part** (Chapters 3-4) of this thesis focuses on the improvements that can be achieved in terms of noise reduction performance and spatial cue preservation by first adding a single sensor (microphone) to a binaural noise reduction system. The increase in the noise reduction performance by the addition of a single sensor signal serves as the motivation for later chapters where many nodes consisting of multiple sensors are available. A method on choosing which (sub)set of signals or nodes that should be included in the estimation for maximum benefit is also discussed. A method is introduced that uses a so-called utility measure to limit the effect of node removal and can be used to aid in node addition. While first introduced in a centralized scenario, this utility is extended to a distributed scenario.

The **second part** (Chapters 5-6) of this thesis looks to redefine the linear compression and fusion rules of the DANSE algorithm for use in mixed-topology WSNs. These mixed-topology WSNs are divided to a set of smaller substructures that are based on either clique or cluster formation. It is shown, that based on the redefined linear compression and fusion rules, the same convergence properties compared to single fully connected or tree topologies are achievable. Supporting techniques are also introduced relating to different broadcast strategies and topology formation.

In the **third part** (Chapter 7) of this thesis, each node implements a novel method to linearly compress its sensor signals in order to transmit to the other nodes in the WSN. This leads to the introduction of the so-called topology-independent distributed adaptive node-specific signal estimation (TI-DANSE) algorithm. While the TI-DANSE algorithm is first introduced in a fully connected topology, the convergence properties are shown to be applicable in any topology, as long as the nodes have access to a network wide summed signal.

An outline of the thesis and its contributions are given as follows:

1.5.2 Thesis outline

Chapter 2 is a continuation of this introductory chapter that defines the general framework and multi-channel noise reduction techniques used throughout this thesis. A review of the MWF is given using a single device with multiple sensors and then extends the MWF to a WSN, which operates in a centralized fashion, i.e., every signal is accessible by every node in the WSN. The WSN then employs the DANSE algorithm, which can be viewed as a distributed version of the MWF. A concise outline is given highlighting its convergence and optimality properties in a fully connected topology.

Chapter 2 also references Appendix A which is based on:

- **Szurley J.**, Bertrand A., and Moonen M., "On the use of time-domain widely linear filtering for binaural speech enhancement", IEEE Signal Processing Letters, vol. 20, Jul. 2013, pp. 649-652.

showing the equivalence between linear and widely linear filtering in a two node WASN using only real time domain signals.

Chapter 3 considers a general binaural system for noise reduction that employs a slightly modified version of the MWF presented in Chapter 2, namely the multi-channel Wiener filter with partial noise estimation (MWF_η). Along with the binaural system, it is assumed that a remote microphone signal with high input SNR is available thereby forming a three node WASN. It is shown that in using a remote microphone signal there is an increase in the noise reduction performance, when compared to the original binaural system, and that the so-called binaural cues can be better preserved.

Chapter 3 is based on:

- **Szurley J.**, Bertrand A., and Moonen M., "Binaural noise cue preservation in a binaural noise reduction system with a remote microphone signal", *submitted for publication*, 2015.

Chapter 4 extends the three node system introduced in Chapter 3 to a more general WSN consisting of many nodes. A method is introduced that uses a so-called utility measure as a means to facilitate subset selection, where the WSN selects the most important nodes to remain on and continually broadcast their signals, which are used for signal estimation while removing the less important nodes. While an optimal node selection strategy is NP-hard due to its combinatorial nature, a greedy heuristic selection procedure is given that can add or remove nodes in an iterative fashion until a set of constraints are

satisfied which are based on the utility. It is shown, that when the estimation is performed in a centralized fashion, i.e., each node has access to every other nodes raw signal, a centralized algorithm can efficiently compute the utility, which is then used for the subset selection. When the estimation is performed where the DANSE algorithm is in place, the greedy selection procedure will rely on upper and lower utility bounds to facilitate the subset selection. However, due to the greedy nature of the algorithm, the distributed selection procedure has similar performance properties to that of the centralized selection procedure.

Chapter 4 is based on:

- **Szurley J.**, Bertrand A., Ruckebusch P., Moerman I., and Moonen M., "Greedy distributed node selection for node-specific signal estimation in wireless sensor networks", Signal Processing, vol. 94, Jan. 2014, pp. 57-73.

Chapter 5 focuses on mixed-topology formation of a WSN based on one of two approaches, either a top-down approach where the WSN is composed of heterogeneous nodes, or a bottom-up approach where the nodes are not necessarily heterogeneous. In the top-down approach, the nodes with the largest energy budgets are designated as cluster heads and the remaining nodes form clusters around these nodes. In the bottom-up approach, an ad-hoc WSN is partitioned into a set of smaller substructures consisting of non-overlapping cliques that are arranged in a tree topology. These two approaches are shown to be conceptually equivalent, in that the same building blocks constitute both envisaged topologies, where the functionality of the DANSE algorithm is extended to such topologies.

Chapter 5 is based on:

- **Szurley J.**, Bertrand A., and Moonen M., "Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks", *accepted for publication in* Signal Processing, 2015.

Chapter 6 deals with distributed estimation in WASNs with tree topologies where the tree-DANSE (T-DANSE) algorithm is employed. First, it is assumed that there is at least one node with a virtually infinite energy budget, which is selected as the root node of the tree. A modification to the T-DANSE algorithm is presented that allows the root node to broadcast a single signal to all of the other nodes in the WSN. This modification is shown to be theoretically equivalent to the original T-DANSE algorithm in terms of the signal estimation performance and it comes with the added benefits of not only reducing the input-output delay but also shifting a large part of the communication burden to the root node.

Next it is shown, that due to the data-driven flow of the T-DANSE algorithm and unavoidable errors in the estimation of certain second-order statistics, the selection of the root node plays an important role in the overall performance of the network wide signal estimation. The concept of eigenvector centrality is introduced as a way to prune an ad-hoc WASN to that of a tree topology, which provides improved network wide signal estimation.

Chapter 6 is based on:

- **Szurley J.**, Bertrand A., Moerman I., and Moonen M., "A Modified broadcast strategy for distributed signal estimation in a wireless sensor network with a tree topology", in the Proc. of the 39th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, May 2014, pp. 7620-7624.
- **Szurley J.**, Bertrand A., Moerman I., and Moonen M., "Network topology selection for distributed speech enhancement in wireless acoustic sensor networks", in the Proc. of the 21st European Signal Processing Conference (EUSIPCO), Marrakesh, Morocco, 2013, pp. 1-5.

In chapter 7, the topology-independent distributed adaptive node-specific signal estimation (TI-DANSE) algorithm is presented which is able to accomplish the same signal estimation as presented in the preceding chapters but no longer relies on a given topology. The nodes in the TI-DANSE algorithm rely on an in-network summation that is found via a data-driven signal flow that takes place on a tree, where the topology of the tree is allowed to change in each iteration. The TI-DANSE algorithm is therefore less sensitive to link failures and can be used in WSNs with dynamic topologies.

Chapter 7 is based on:

- **Szurley J.**, Bertrand A., and Moonen M., "Topology-independent distributed adaptive node-specific signal estimation in wireless sensor networks", *submitted for publication*, 2015.

Chapter 8 concludes the thesis by re-iterating the overall objectives, highlighting the key contributions and exploring future research directions.

Chapter 2

Centralized and distributed signal estimation

2.1 Data model

A device, or node, is assumed, which for the time being is designated as node k, that contains a sensor array of M_k sensors in an arbitrary geometry. The received signal at the mth sensor, $y_{k,m}$, is given in the short-time Fourier transform (STFT) domain as

$$y_{k,m}(\omega_l, t) = x_{k,m}(\omega_l, t) + n_{k,m}(\omega_l, t) \tag{2.1}$$

where $x_{k,m}$ is the desired signal component, $n_{k,m}$ is additive noise, ω_l is the discrete frequency with $l \in \{1, \ldots, L\}$, and $t \in \mathbb{N}$ is the frame index. For the sake of brevity, the discrete frequencies and time indices will be omitted from the following derivations bearing in mind that operations take place in the STFT domain. Time domain filtering will be discussed briefly in section 2.3.2 with widely-linear filtering, however, all other derivations as assumed to take place in the STFT domain.

All of the M_k sensor signal observations of node k are placed in a stacked vector represented as

$$\mathbf{y}_k = [y_{k,1}, \ldots, y_{k,M_k}]^T \tag{2.2}$$

where ^{T} is the transpose operator and $\mathbf{x}_k$ and $\mathbf{n}_k$ are defined similarly to (2.2) such that

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{n}_k \ . \tag{2.3}$$

The desired signal components, $\mathbf{x}_k$, share a latent Q-dimensional signal subspace given as

$$\mathbf{x}_k = \mathbf{A}_k \mathbf{s} \tag{2.4}$$

where $\mathbf{s}$ is a stochastic Q-dimensional vector that contains the source signals and $\mathbf{A}_k$ is a deterministic but unknown full row rank $M_k \times Q$-dimensional steering matrix which contains the transfer functions between the source signals and the sensors.

2.2 Multi-channel Wiener filtering

The node is tasked with estimating a J-dimensional node-specific desired signal, $\mathbf{d}_k$, by applying a linear filter-and-sum operation to all of its received sensor signals

$$\widetilde{\mathbf{d}}_k = \mathbf{H}_k^H \mathbf{y}_k \tag{2.5}$$

where $\widetilde{\mathbf{d}}_k$ represents the estimate of $\mathbf{d}_k$ and ^{H} is the conjugate transpose operator. While it is not a strict requirement, the node-specific desired signal, $\mathbf{d}_k$, is assumed to be a subset of the desired signals, $\mathbf{x}_k$, in which case they can be represented as

$$\mathbf{d}_k = \overline{\mathbf{A}}_k \mathbf{s} \tag{2.6}$$

where $\overline{\mathbf{A}}_k$ contains a subset of rows of $\mathbf{A}_k$ in (2.4). It is also assumed without loss of generality (w.l.o.g.), that $J = Q$, i.e., $\overline{\mathbf{A}}_k$ is a square matrix.

The filter, $\mathbf{H}_k$, is found by performing a LMMSE estimation between the node-specific desired signal and the filtered version of the received sensor signals, i.e.,

$$\widehat{\mathbf{H}}_k = \arg\min_{\mathbf{H}_k} \mathcal{E}\left\{\|\mathbf{d}_k - \mathbf{H}_k^H \mathbf{y}_k\|_2^2\right\} \tag{2.7}$$

$$= \arg\min_{\mathbf{H}_k} J_k(\mathbf{H}_k) \tag{2.8}$$

where $\mathcal{E}\{\cdot\}$ is the expectation operator and where $J_k(\mathbf{H}_k)$ in (2.8) is the cost function.

The solution to (2.7) is given as the MWF which has the form [85–87]

$$\widehat{\mathbf{H}}_k = \mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}^{-1} \mathbf{R}_{\mathbf{y}_k\mathbf{d}_k} \tag{2.9}$$

where $\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k} = \mathcal{E}\{\mathbf{y}_k\mathbf{y}_k^H\}$ is the signal+noise correlation matrix and $\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_k\mathbf{d}_k^H\}$ is the cross-correlation between the received signal and the node-specific desired signal. Methods for estimating these statistics are presented in Section 2.2.2.

An attractive attribute of the MWF is that it only relies on the collection of the second-order statistics, namely $\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}$ and $\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k}$, and does not require an estimate $\mathbf{A}_k$ or $\mathbf{s}$ which is typically done in BSS algorithms.

2.2.1 Special case : single node-specific desired signal

An alternative formulation of (2.9) can be derived in the special case when there is a single node-specific desired signal, d_k ($Q = 1$) [49, 88, 89]. In this case, the solution to (2.9) may be given as the so-called rank-1 MWF (R1-MWF) [49, 90, 91]

$$\widehat{\mathbf{h}}_k = \frac{\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}^{-1}\mathbf{r}_{\mathbf{y}_k d_k}}{1 + \mathrm{Tr}\{\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}^{-1}\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}\}} \tag{2.10}$$

where $\mathrm{Tr}\{\cdot\}$ is the trace of a matrix, $\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k} = \mathcal{E}\{\mathbf{n}_k\mathbf{n}_k^H\}$ is the noise correlation matrix and $\mathbf{r}_{\mathbf{y}_k d_k} = \mathcal{E}\{\mathbf{y}_k d_k^*\}$ is again the cross-correlation between the received signal and the node-specific desired signal, where * denotes the complex conjugate. The desired signal correlation matrix, $\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}$, is a rank-1 matrix that is given as

$$\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k} = \mathcal{E}\{\mathbf{x}_k\mathbf{x}_k^H\} \tag{2.11}$$

$$= \mathcal{E}\{s\mathbf{a}_k\mathbf{a}_k^H s^*\} \tag{2.12}$$

$$= P_s\mathbf{a}_k\mathbf{a}_k^H \tag{2.13}$$

where $P_s = \mathcal{E}\{ss^*\}$ is the power of the desired signal and where $\mathbf{a}_k$ is a M_k-dimensional steering vector (which is the vector form of $\mathbf{A}_k$ in (2.4) for $Q = 1$). In [90], it has been shown that the R1-MWF often outperforms the MWF given in (2.9) as it is less sensitive to errors in the second-order statistics.

2.2.2 Estimation of signal statistics

In order to compute the MWF in (2.9) and (2.10), it is assumed that the second-order statistics are known throughout the estimation procedure. The collection of the second-order statistics typically relies on some type of on-off behavior of the signal as well as implicit (short-term) stationarity. For example, in speech applications, this on-off behavior is the pause between speech segments, which can be observed by means of a voice-activity detector (VAD).

When the node-specific desired signal is active, the signal+noise correlation matrix, $\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}$, can be estimated by time averaging the sensor observations with

a forgetting factor $0 < \gamma < 1$ [12, 22, 92], i.e.,

$$\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}[t] = \gamma \mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}[t-1] + (1-\gamma)\mathbf{y}_k[t]\mathbf{y}_k[t]^H. \tag{2.14}$$

Likewise during noise-only periods, where $\mathbf{y}_k = \mathbf{n}_k$, the noise correlation matrix, $\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}$ can be estimated in a similar fashion, i.e.,

$$\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}[t] = \gamma \mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}[t-1] + (1-\gamma)\mathbf{n}_k[t]\mathbf{n}_k[t]^H\ . \tag{2.15}$$

While the estimation of the signal+noise and noise correlation matrix, $\mathbf{R}_{\mathbf{y}_k\mathbf{y}_k}$ and $\mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}$, can be done directly based on temporal averaging, the estimation of the cross-correlation between the sensor signals and the node-specific desired signal, $\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k}$ in (2.9) and the desired signal correlation matrix $\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}$ in (2.10), is more difficult as it requires statistical information from the unobservable node-specific desired signal $\mathbf{d}_k$ or $\mathbf{x}_k$.

Since the desired signal components and noise are assumed to be uncorrelated, the desired signal correlation matrix may be estimated[1] by subtracting the noise correlation matrix from the signal+noise correlation matrix, i.e.,

$$\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k} = \mathbf{R}_{\mathbf{y}_k\mathbf{y}_k} - \mathbf{R}_{\mathbf{n}_k\mathbf{n}_k}. \tag{2.16}$$

Subsequently, the cross-correlation matrix, $\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k}$, can be given as

$$\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_k\mathbf{d}_k^H\} \tag{2.17}$$

$$= \mathcal{E}\{\mathbf{x}_k\mathbf{d}_k^H\} + \mathcal{E}\{\mathbf{n}_k\mathbf{d}_k^H\} \tag{2.18}$$

where $\mathcal{E}\{\mathbf{n}_k\mathbf{d}_k^H\} = 0$ due to the statistical independence of the node-specific desired signal and the noise. Since the node-specific desired signal is a subset of the signal components in $\mathbf{x}_k$ (compare (2.4) and (2.6)), the cross-correlation between the sensor signals and the node-specific desired signal using (2.18) can be given as

$$\mathbf{R}_{\mathbf{y}_k\mathbf{d}_k} = \mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}\mathbf{E}_k \tag{2.19}$$

where $\mathbf{E}_k$ is an $M_k \times Q$-dimensional matrix containing only zeros, except for a single entry equal to one in each column which selects the columns of $\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}$ that correspond to the node-specific desired signal.

However, there will inevitably be errors when collecting the second-order statistics due to finite window sizes, errors in the VAD, etc. In [90, 93], the impact of estimation errors on the second-order statistics is analyzed in terms of performance degradation and excess MSE and Chapter 6 takes these estimation errors into account when forming the topology of a WSN.

[1] Note that this is just one possible strategy to estimate $\mathbf{R}_{\mathbf{x}_k\mathbf{x}_k}$.

2.3 Centralized MWF for node-specific signal estimation

It is now assumed that node k belongs to a set $\mathcal{K}$ of $|\mathcal{K}| = K$ nodes that form a fully connected WSN. The neighbors of a node k are represented as the set $\mathcal{N}_k$, which for a fully connected WSN is $\mathcal{N}_k = \mathcal{K}\backslash\{k\}$. Each node, $\forall k \in \mathcal{K}$, collects its own set of sensor signal observations of the form given in (2.2) and broadcasts this to every other node in the WSN such that each node has access to $M = \sum_{k=1}^{K} M_k$ sensor signals.

Each node constructs an M-dimensional stacked vector of all of the received sensor signal observations of the entire WSN given as

$$\mathbf{y} = [\mathbf{y}_1^T \dots \mathbf{y}_K^T]^T \tag{2.20}$$

where $\mathbf{x}$ and $\mathbf{n}$ are defined similarly to (2.20) such that

$$\mathbf{y} = \mathbf{x} + \mathbf{n} \,. \tag{2.21}$$

Each node estimates its own node-specific desired signal, $\mathbf{d}_k$, by a similar filter-and-sum operation as in (2.5), where the filter $\mathbf{W}_k$ can now be thought of as a stacked matrix that contains a partitioned set of filters that are applied to the sensor signal observations of each node, i.e.,

$$\mathbf{W}_k = \begin{bmatrix} \mathbf{W}_{k1} \\ \vdots \\ \mathbf{W}_{kK} \end{bmatrix} \tag{2.22}$$

where each $\mathbf{W}_{kq} \in \mathbb{C}^{M_q \times Q}$ represents the filter component that node k applies to a node q's received sensor signal observations $\mathbf{y}_q$, and $\mathbf{W}_{kk} \in \mathbb{C}^{M_k \times Q}$ represents the local filter component that node k applies to its own received sensor signal observations. This partitioning as well as broadcasting of the received sensor signal observations, $\mathbf{y}_k$ is depicted in Figure 2.1 for a $K = 3$ node WSN.

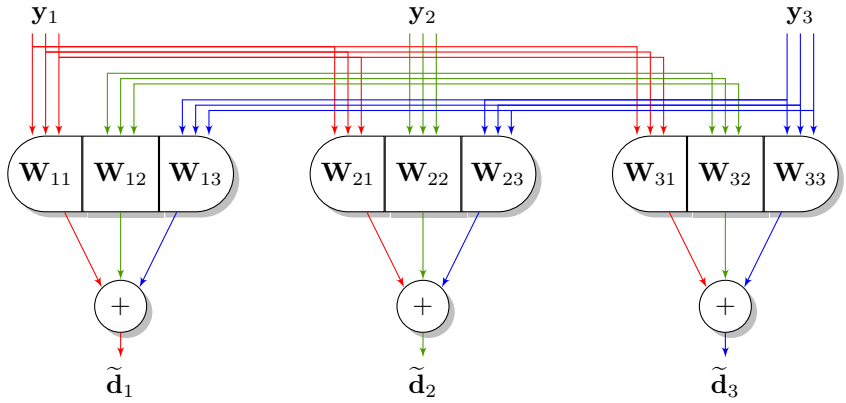


Figure 2.1: Centralized MWF for a $K = 3$ WSN.

The node-specific filter for node k, $\mathbf{W}_k$, is found in the same manner as in (2.7)

$$\widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{k1} \\ \vdots \\ \widehat{\mathbf{W}}_{kK} \end{bmatrix} = \underset{\mathbf{W}_{k1},\ldots,\mathbf{W}_{kK}}{\arg\min}\ \mathcal{E}\left\{ \left\| \mathbf{d}_k - \sum_{q=1}^{K} \mathbf{W}_{kq}^H \mathbf{y}_q \right\|_2^2 \right\} \tag{2.23}$$

$$= \underset{\mathbf{W}_k}{\arg\min}\ \mathcal{E}\left\{ \left\| \mathbf{d}_k - \mathbf{W}_k^H \mathbf{y} \right\|_2^2 \right\} \tag{2.24}$$

$$= \underset{\mathbf{W}_k}{\arg\min}\ J_k(\mathbf{W}_k)\ . \tag{2.25}$$

The solution to (2.23) takes the same form given in (2.9) as

$$\widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{k1} \\ \vdots \\ \widehat{\mathbf{W}}_{kK} \end{bmatrix} = \mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{yd}_k} = \mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{E}_k \tag{2.26}$$

where $\mathbf{R}_{\mathbf{yd}_k} = \mathcal{E}\{\mathbf{y}\mathbf{d}_k^H\}$, $\mathbf{R}_{\mathbf{yy}} = \mathcal{E}\{\mathbf{y}\mathbf{y}^H\}$, $\mathbf{R}_{\mathbf{xx}} = \mathcal{E}\{\mathbf{x}\mathbf{x}^H\}$ and $\mathbf{E}_k$ is now an $M \times Q$-dimensional matrix containing only zeros, except for a single entry equal to one in each column which selects the columns of $\mathbf{R}_{\mathbf{xx}}$ that correspond to the node-specific desired signal of node k.

2.3.1 Relationship between node-specific desired signals and estimators

It is assumed that the desired signal components of each node, $\mathbf{x}_k$, $\forall k \in \mathcal{K}$, share a latent Q-dimensional signal subspace as in (2.4) such that the stacked vector of all desired signal components, $\mathbf{x}$, can be represented as

$$\mathbf{x} = \mathbf{A}\mathbf{s} \tag{2.27}$$

where $\mathbf{A}$ is a stacked $M \times Q$-dimensional steering matrix that contains the steering matrices of each node $\mathbf{A}_k$, $\forall k \in \mathcal{K}$.

The node-specific desired signals can be related to one another by using (2.6) as

$$\mathbf{d}_k = \overline{\mathbf{A}}_k \left(\overline{\mathbf{A}}_q\right)^{-1} \mathbf{d}_q \ , \forall k, q \in \mathcal{K}. \tag{2.28}$$

Using this relationship and (2.26), it directly follows that the optimal LMMSE estimators at each node are also related by

$$\widehat{\mathbf{W}}_k = \widehat{\mathbf{W}}_q \overline{\mathbf{A}}_{kq} \ , \forall k, q \in \mathcal{K} \tag{2.29}$$

where

$$\overline{\mathbf{A}}_{kq} = \left(\overline{\mathbf{A}}_q\right)^{-H} \overline{\mathbf{A}}_k^H \ . \tag{2.30}$$

The relationships given in (2.28) - (2.30) will be used extensively throughout this thesis.

2.3.2 Widely linear filtering

In the previous derivations, it was assumed that the collected complex statistics are circular, i.e., the pseudo-variance of the complex random variables are equal to zero while their variances are non-zero [94, 95]. With the previously defined linear filters, due to the circularity assumptions, any non-circular second-order statistics could result in a suboptimal solution.

In order to fully incorporate the circular and non-circular statistics, a widely linear (WL) filter can be used [96, 97], which augments a complex signal with its conjugate to derive an optimal filter in the WL sense which can improve the MSE performance.

However, in some applications, [98–102] WL filtering has been applied to a two node system, where two real signals are combined to form a single complex signal between the nodes in the time-domain. In Appendix A, it is shown

that while this presents a novel approach, WL filtering on two real signals in the time-domain offers no improvement compared to linear filtering while increasing the computational complexity. It is reiterated though, that besides the formulation presented in Appendix A, the derivations in this thesis take place in the STFT domain and that the assumed circularity assumption holds.

2.4 The distributed adaptive node-specific signal estimation (DANSE) algorithm

In the previous section, the nodes broadcast their full M_k-dimensional signal to every other node in the WSN. However, as the number of nodes and the number of sensors per node increases, a larger communication bandwidth is required. It is with this in mind that the distributed adaptive node-specific signal estimation (DANSE) algorithm is introduced which allows the nodes to broadcast a linearly compressed version of their signal while still being able to converge to the centralized solution given in (2.26). Here, the DANSE algorithm for a fully-connected network is described. An extension for a tree-topology has been proposed in [54] and will be reviewed in Chapter 6.

The DANSE algorithm uses an iterative updating scheme where, at every iteration i, a node updates its node-specific parameters in a round-robin fashion. Each node k now broadcasts a linearly compressed version of its sensor signal observations given as

$$\mathbf{z}_k^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k \tag{2.31}$$

where $\mathbf{W}_{kk}^i$ is the local filter component that node k applies to its own received sensor signal observations at iteration i. Node k now has access to observations of its own sensor signal observations and $K-1$ linearly compressed signals, $\mathbf{z}_q^i,\ \forall q \in \mathcal{K}\backslash\{k\}$, from other nodes.

Node k collects observations of the fused signals from other nodes in a stacked vector defined as

$$\mathbf{z}_{-k}^i = [\mathbf{z}_1^{iT}, \ldots, \mathbf{z}_{k-1}^{iT}, \mathbf{z}_{k+1}^{iT}, \ldots, \mathbf{z}_K^{iT}]^T \tag{2.32}$$

where the subscript $_{-k}$ indicates that the linearly compressed signals of node k are not included in the vector. The sensor signals at node k and the fused signals from the other nodes are placed into a stacked vector given as

$$\widetilde{\mathbf{y}}_k^i = \begin{bmatrix} \mathbf{y}_k \\ \mathbf{z}_{-k}^i \end{bmatrix}. \tag{2.33}$$

Since node k only has access to its own sensor signals, $\mathbf{y}_k$ and the linearly compressed signals from other nodes, it can only control a specific part of its

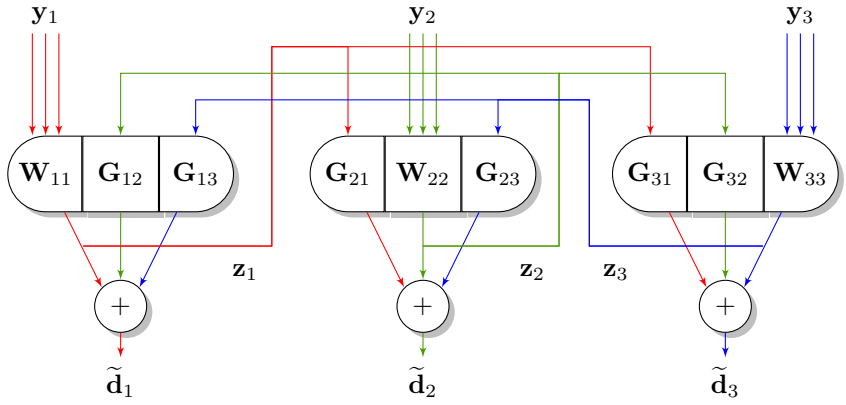


Figure 2.2: The DANSE algorithm for a $K = 3$ WSN.

estimator (2.26), namely $\mathbf{W}_{kk}$, while applying a transform $\mathbf{G}_{kq} \in \mathbb{C}^{Q\times Q}$ to each of the received signals from other nodes. These $\mathbf{G}_{kq}$'s are placed in a stacked vector which is given as

$$\mathbf{G}^i_{k_{-k}} = [\mathbf{G}^{iT}_{k1}, \ldots, \mathbf{G}^{iT}_{k(k-1)}, \mathbf{G}^{iT}_{k(k+1)}, \ldots, \mathbf{G}^{iT}_{kK}]^T \tag{2.34}$$

which consists of $K-1$, $Q \times Q$-dimensional matrices that are applied to the linearly compressed signals from other nodes.

At iteration i, node k performs an update of its local LMMSE estimator $\widetilde{\mathbf{W}}^i_k$ which is similar to (2.23) and is given as

$$\widetilde{\mathbf{W}}^{i+1}_k = \begin{bmatrix} \widetilde{\mathbf{W}}^{i+1}_{kk} \\ \widetilde{\mathbf{G}}^{i+1}_{k_{-k}} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_{k_{-k}}}{\arg\min}\, \mathcal{E}\left\{ \left\| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}^H_{kk} & \mathbf{G}^H_{k_{-k}} \end{bmatrix} \widetilde{\mathbf{y}}^i_k \right\|_2^2 \right\}, \tag{2.35}$$

which is computed based on a block of B most recent observations of $\widetilde{\mathbf{y}}_k$ that node k has collected. A depiction of the DANSE algorithm is given in Figure 2.2 for a $K = 3$ node WSN.

The solution to (2.35) is given as

$$\begin{bmatrix} \widetilde{\mathbf{W}}^{i+1}_{kk} \\ \widetilde{\mathbf{G}}^{i+1}_{k_{-k}} \end{bmatrix} = (\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k})^{-1}\mathbf{R}^i_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}\mathbf{E}_k \tag{2.36}$$

where $\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k} = \mathcal{E}\{\widetilde{\mathbf{y}}^i_k\widetilde{\mathbf{y}}^{iH}_k\}$, $\mathbf{R}^i_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k} = \mathcal{E}\{\widetilde{\mathbf{x}}^i_k\widetilde{\mathbf{x}}^{iH}_k\}$, $\widetilde{\mathbf{x}}^i_k$ is defined similarly to (2.33) and can be estimated in the same fashion as presented in Section 2.2.2, and $\mathbf{E}_k$

is now an $(M_k + K - 1) \times Q$-dimensional matrix containing only zeros, except for a single entry equal to one in each column which selects the columns of $\mathbf{R}^i_{\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k}$ that correspond to the node-specific desired signal of node k.

Table 2.1: The DANSE algorithm in a fully connected WSN

1. Initialize $0 \rightarrow i$, $1 \rightarrow k$.
   Initialize $\mathbf{W}^0_{qq}$ and $\mathbf{G}^0_{q_{-q}}$ randomly, $\forall q \in \mathcal{K}$.
2. Node k updates it node-specific local parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_{k_{-k}}$, by minimizing its LMMSE criterion based on the B most recent observations of its own local sensor signals and the $K-1$ linear compressed signals from neighboring nodes

$$\begin{bmatrix} \mathbf{W}^{i+1}_{kk} \\ \\ \mathbf{G}^{i+1}_{k_{-k}} \end{bmatrix} = \underset{\mathbf{W}_{kk}, \mathbf{G}_{k_{-k}}}{\arg\min} \; \mathcal{E}\left\{ \left|\left| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}^H_{kk} & \mathbf{G}^H_{k_{-k}} \end{bmatrix} \widetilde{\mathbf{y}}^i_k \right|\right|^2_2 \right\} \tag{2.37}$$

   for which the solution is given by (2.36).

   The other nodes do not update their node-specific local parameters:

$$\forall q \in \mathcal{K} \backslash \{k\} : \mathbf{W}^{i+1}_{qq} = \mathbf{W}^i_{qq}, \mathbf{G}^{i+1}_{q_{-q}} = \mathbf{G}^i_{q_{-q}} \; . \tag{2.38}$$

3. $i{+}1 \rightarrow i$.
4. $(i \bmod K) + 1 \rightarrow k$.
5. return to 2 .

It is noted that the nodes share many compressed sensor signal observations in between two iterations of the DANSE algorithm, i.e., the block length B should be chosen long enough such that a sufficient number of observations are collected to accurately estimate the correlation elements corresponding to $\mathbf{z}^i_k$.

The same compressed sensor signal observations will not be transmitted in sequential iterations, i.e., the broadcast signal of each node at iteration i, $\mathbf{z}^i_k$, will consist of a different time segment than the broadcast signal at iteration $i+1$, $\mathbf{z}^{i+1}_k$. This differs from other distributed algorithms where the same signal observation is transmitted over multiple iterations, resulting in an increased and

non-scalable communication cost, in particular when used for signal estimation [103, 104].

The node-specific signal estimate, $\widetilde{\mathbf{d}}_k$, for a node can be found at any point in the iterative procedure by

$$\widetilde{\mathbf{d}}_k^i = \widetilde{\mathbf{W}}_k^{iH} \widetilde{\mathbf{y}}_k^i \tag{2.39}$$

$$= \mathbf{W}_{kk}^{iH} \mathbf{y}_k + \mathbf{G}_{k_{-k}}^{iH} \mathbf{z}_{-k}^i \, . \tag{2.40}$$

Based on (2.31) and (2.40) the network-wide estimator LMMSE $\mathbf{W}_k^i$ that defines the estimate of $\mathbf{d}_k$ at iteration i is parameterized as

$$\mathbf{W}_k^i = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{k1}^i \\ \vdots \\ \mathbf{W}_{KK}^i \mathbf{G}_{kK}^i \end{bmatrix} \tag{2.41}$$

where node k controls $\mathbf{W}_{kk}^i$ and $\mathbf{G}_{k_{-k}}^i$. Since node k can control its own estimator matrix $\mathbf{W}_{kk}^i$, then $\mathbf{G}_{kk} = \mathbf{I}$ by definition to minimize the degrees of freedom, where $\mathbf{I}$ is a $Q \times Q$-dimensional identity matrix. The parameterization of (2.41) simultaneously defines a solution space for all $\mathbf{W}_k^i$, $\forall k \in \mathcal{K}$.

By setting $\mathbf{W}_{kk}^i = \widehat{\mathbf{W}}_{kk}$ and $\mathbf{G}_{kq}^i = \overline{\mathbf{A}}_{kq}$, $\forall k, q \in \mathcal{K}$, it is seen that the optimal LMMSE estimators, $\widehat{\mathbf{W}}_k$, $\forall k \in \mathcal{K}$, lie in the parameterized solution space of (2.41). It was shown in [53] that if the node-specific desired signals are described by (2.6), and if the nodes perform the update (2.35) in a sequential round robin fashion, then the parameterized network-wide estimators (2.41) indeed converge to the corresponding centralized LMMSE estimators (2.26), i.e.,

$$\lim_{i \to \infty} \widetilde{\mathbf{W}}_k^i = \widehat{\mathbf{W}}_k, \ \forall k \in \mathcal{K} \tag{2.42}$$

and hence each node, $k \in \mathcal{K}$, is able to estimate its node-specific desired signal, $\mathbf{d}_k$ as if it had access to all the sensor signal observations of all other nodes, i.e.,

$$\lim_{i \to \infty} \widetilde{\mathbf{d}}_k^i = \mathbf{d}_k, \ \forall k \in \mathcal{K}. \tag{2.43}$$

Part I

Noise reduction with a remote signal and subset selection

Chapter 3

Binaural noise cue preservation in a binaural noise reduction system with a remote microphone signal

3.1 Introduction

Hearing impaired (HI) individuals exhibit poor speech intelligibility in the presence of background noise. Depending on the nature and profoundness of the hearing loss as well as the type of background noise HI individuals may need a higher SNR, with a difference of up to 15 dB in order to have the same intelligibility as non-HI individuals [105–107]. In order to reduce the amount of background noise, various APs such as hearing HAs or CIs, which come equipped with one or more microphones, perform a noise reduction operation on the microphone signals to provide the individual with an enhanced signal.

Furthermore, for individuals who exhibit hearing impairment in both ears, this can be extended to a bilateral system where two devices are used, e.g., two HAs, two CIs or a bi-modal system, which is a combination of APs. In using bilateral APs that share the microphone signals between one another, i.e., perform *binaural* processing, improved speech perception in noisy environments is observed when compared to bilateral APs that operate independently of one another [108, 109]. Another advantageous aspect of binaural processing is that

it allows the HI individual to localize a sound source by exploiting the so-called binaural cues [3, 110].

Due to the limited size of the APs, which restricts the spatial diversity of the microphones, as well as challenging noise environments, there has been an increased interest in incorporating remote microphone signals to the APs [12, 111, 112]. It is assumed that these remote microphones are either worn or placed close to the desired speech source thereby providing an improved SNR resulting in increased speech intelligibility. There are currently various systems that are able to transmit a remote microphone signal to the APs, e.g., traditional telecoil systems by FM transmission [113, 114]. However, these systems either disable the local microphones or significantly amplify the remote microphone signal thereby degrading the binaural cues.

A general binaural system is envisaged, where each device performs noise reduction with the local microphone signals as well as a remote microphone signal. The devices perform this noise reduction by employing the MWF. However, it has been shown that while the MWF perfectly preserves the binaural speech cues, the noise binaural cues are distorted. This distortion may place the listener in danger in situations where the binaural noise cues may be important for the safety, e.g. hazard warnings, traffic, etc.

In order to preserve the binaural speech and noise cues, several methods have been proposed by either conditioning the microphone signals or modifying the MWF. Binaural cue preservation was studied in [115] which relies on applying a gain to the signal, albeit only to preserve the binaural speech cues. In [3,116,117] the authors extended the cost function used to derive the MWF to include a portion that also preserves the binaural noise cues. However, since there is no closed form expression for the solution, iterative techniques must be used, or additional parameters are adjusted by trial and error. In [16, 118] an extension to the MWF was introduced, the MWF with partial noise estimation (MWF_η), which allows for an explicit trade-off between the amount of noise reduction and the preservation of the binaural noise cues in a closed-form solution.

It will be shown that when a high input SNR remote microphone signal is available to the binaural system, not only is the noise reduction performance increased, but also a better preservation of the binaural noise cues is achieved. This is surprising considering the fact that the remote microphone signal does not contain binaural cue information. In order to further increase the performance, a modification to the partial noise estimation variable is proposed which relies on exploiting the aforementioned trade-off, either constraining the output SNR or binaural noise cues to the same level before and after the addition of the remote microphone signal. Also when the noise and speech signal come from the same direction, due to the blocking structure of the MWF_η, almost no SNR

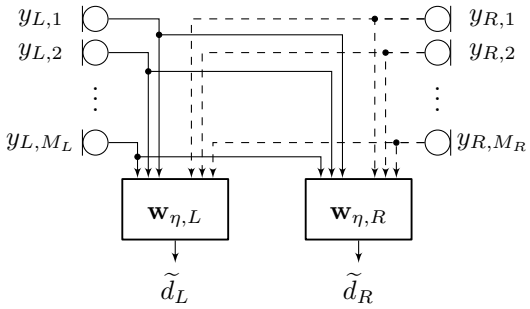


Figure 3.1: A general binaural noise reduction system.

improvement is observed [117, 119]. The addition of the remote microphone can then help to improve the SNR, as it adds more spatial diversity in the signal subspace.

This chapter is organized as follows. Section 3.2 introduces the general binaural noise reduction system along with the MWF$_\eta$. In Section 3.3 the performance measures are defined emphasizing the distortion in the binaural noise cues at the output of the system. In Section 3.4 the general binaural system is extended to include a remote microphone signal and in Section 3.5 the effect on the output SNR and binaural noise cues is derived. In Section 3.6 a modification to the partial noise estimation variable η is given in order to further increase noise reduction performance or binaural noise cue preservation. Simulations are performed in Section 3.7, using head-related impulse responses (HRIRs) to generate various acoustic environments where a remote microphone signal is available for inclusion into the MWF$_\eta$. Finally, conclusions are given in Section 3.8.

3.2 Binaural data model

A general binaural noise reduction system is considered as depicted in Figure 3.1 that consists of a left and right device, denoted as L and R respectively, each equipped with $M_L = M_R$ microphones where the combined number of microphones is given as $M = M_L + M_R$. The signal of the mth microphone of the left device, denoted with the subscript $_{L,m}$, is given as in (2.1) with $k = L$ as

$$y_{L,m} = x_{L,m} + n_{L,m} \, . \tag{3.1}$$

A stacked M_L-dimensional vector containing the microphone signals of the left device is defined in the same manner as (2.2), i.e.,

$$\mathbf{y}_L = \begin{bmatrix} y_{L,1} \\ \vdots \\ y_{L,M_L} \end{bmatrix} \tag{3.2}$$

where $\mathbf{x}_L$ and $\mathbf{n}_L$ are defined similarly and contain the speech components and noise components respectively. A stacked M_R-dimensional vector containing the microphone signals of the right device, $\mathbf{y}_R$, as well as the speech and noise components, $\mathbf{x}_R$ and $\mathbf{n}_R$, are also defined in the same manner as (3.2). A stacked M-dimensional vector is defined that contains all of the microphone signals of the system and is given as

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_L \\ \mathbf{y}_R \end{bmatrix} \tag{3.3}$$

where again $\mathbf{x}$ and $\mathbf{n}$ are defined similarly.

Each device is tasked with estimating a desired speech signal, $d_{\{L,R\}}$, which w.l.o.g., is assumed to be the speech component of the first microphone of each device, i.e., $d_{\{L,R\}} = x_{\{L,R\},1}$, by applying a multi-channel filter $\mathbf{w}_{\eta,\{L,R\}}$ to the microphone signals given in (3.3). The filtered output of each device is given as

$$\widetilde{d}_L = \mathbf{w}_{\eta,L}^H \mathbf{y}, \quad \widetilde{d}_R = \mathbf{w}_{\eta,R}^H \mathbf{y} \tag{3.4}$$

which can be expanded to

$$\widetilde{d}_L = \mathbf{w}_{\eta,L}^H \mathbf{x} + \mathbf{w}_{\eta,L}^H \mathbf{n}, \quad \widetilde{d}_L = \mathbf{w}_{\eta,R}^H \mathbf{x} + \mathbf{w}_{\eta,R}^H \mathbf{n}\,. \tag{3.5}$$

Each device finds its multi-channel filter based on the LMMSE criterion between the speech and noise component in its reference microphone signal and the filtered version of the microphone signals [16], i.e.,

$$\begin{aligned} \mathbf{w}_{\eta,L} &= \underset{\mathbf{w}_{\eta,L}}{\arg\min}\, \mathcal{E}\{|d_L - \mathbf{w}_{\eta,L}^H \mathbf{x}|^2\} + \mu \mathcal{E}\{|\eta n_{L,1} - \mathbf{w}_{\eta,L}^H \mathbf{n}|^2\} \\ \mathbf{w}_{\eta,R} &= \underset{\mathbf{w}_{\eta,R}}{\arg\min}\, \mathcal{E}\{|d_R - \mathbf{w}_{\eta,R}^H \mathbf{x}|^2\} + \mu \mathcal{E}\{|\eta n_{R,1} - \mathbf{w}_{\eta,R}^H \mathbf{n}|^2\} \end{aligned} \tag{3.6}$$

where $0 \leq \mu$ is a trade-off parameter emphasizing the noise reduction at the cost of higher speech distortion, $0 \leq \eta \leq 1$ is the partial noise estimation variable that allows a certain amount of the noise to be present in the output signal. Note that η essentially controls how much residual noise is present in the filtered output and can be thought of as producing a given target SNR increase of $\frac{1}{\eta^2}$. This will explained in more detail in Section 3.5.1.

The solution to (3.6) is the MWF with partial noise estimation (MWF$_\eta$) which is given as [16]

$$\mathbf{w}_{\eta,L} = (1-\eta)\frac{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{e}_L}{\mu + \mathrm{Tr}\{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\}} + \eta\mathbf{e}_L$$

$$\mathbf{w}_{\eta,R} = (1-\eta)\frac{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{e}_R}{\mu + \mathrm{Tr}\{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\}} + \eta\mathbf{e}_R \tag{3.7}$$

where $\mathbf{e}_{\{L,R\}}$ are unit vectors which selects the columns of $\mathbf{R}_{\mathbf{xx}}$ that correspond to the desired speech signal of the device and the speech and noise correlation matrices are defined in the same fashion as in Chapter 2, i.e.,

$$\mathbf{R}_{\mathbf{xx}} = \mathcal{E}\{\mathbf{x}\mathbf{x}^H\},\ \mathbf{R}_{\mathbf{nn}} = \mathcal{E}\{\mathbf{n}\mathbf{n}^H\}\ . \tag{3.8}$$

When $\eta = 0$, (3.7) corresponds to the MWF with no partial noise estimation (2.10), whereas $\eta = 1$ corresponds to the trivial case where no noise reduction is performed and the reference microphone signal is passed to the output of the device.

The derivation of (3.7) relies on the assumption that there is a single speech source of interest [16] [49], such that $\mathbf{R}_{\mathbf{xx}}$ is rank-1 and given as in (2.13), i.e.,

$$\mathbf{R}_{\mathbf{xx}} = P_s\mathbf{a}\mathbf{a}^H \tag{3.9}$$

where $\mathbf{a}$ is a stacked M-dimensional speech steering vector containing the left and right device speech steering vectors $\mathbf{a}_{\{L,R\}}$ respectively. If the device speech steering vectors, $\mathbf{a}_{\{L,R\}}$, are defined in the same fashion as (3.2), then the input speech power at the reference microphones can be written as

$$P_{s,\{L,R\}} = \mathbf{e}_{\{L,R\}}^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_{\{L,R\}} = P_s|a_{\{L,R\},1}|^2\ . \tag{3.10}$$

It is assumed that the noise correlation matrix is given similarly to that of [86] which is comprised of a single coherent (spatially correlated) noise source with noise power normalized to unity and additive incoherent process noise represented as

$$\mathbf{R}_{\mathbf{nn}} = \mathbf{v}\mathbf{v}^H + \delta\mathbf{I} \tag{3.11}$$

where $\mathbf{v}$ is the M-dimensional spatially correlated noise steering vector and δ is the noise power of the incoherent noise. Again, the noise steering vectors $\mathbf{v}_{\{L,R\}}$ are defined in the same fashion as (3.2), then the input noise power at the reference microphones is given in the same fashion as (3.10), i.e.,

$$P_{n,\{L,R\}} = \mathbf{e}_{\{L,R\}}^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_{\{L,R\}} = |v_{\{L,R\},1}|^2 + \delta\ . \tag{3.12}$$

Using the matrix inversion lemma, the inverse of (3.11) is given as

$$\mathbf{R}_{\mathbf{nn}}^{-1} = \frac{1}{\delta}\left[\mathbf{I} - \frac{\mathbf{v}\mathbf{v}^H}{\delta + ||\mathbf{v}||^2}\right] . \tag{3.13}$$

The $\mathrm{Tr}\{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\}$ is now denoted as λ, which will be used in the sequel, and when using (3.9) and (3.13) yields

$$\mathrm{Tr}\{\mathbf{R}_{\mathbf{nn}}^{-1}\mathbf{R}_{\mathbf{xx}}\} = \frac{P_s}{\delta}\left[||\mathbf{a}||^2 - \frac{|\mathbf{a}^H\mathbf{v}|^2}{\delta + ||\mathbf{v}||^2}\right] \tag{3.14}$$

$$= \lambda . \tag{3.15}$$

3.3 Performance measures

3.3.1 Output SNR

The input SNR for the left and right device using (3.10) and (3.12) is given as

$$\mathrm{SNR}_{\mathrm{IN},\{L,R\}} = \frac{\mathbf{e}_{\{L,R\}}^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_{\{L,R\}}}{\mathbf{e}_{\{L,R\}}^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_{\{L,R\}}} \tag{3.16}$$

$$= \frac{P_s|a_{\{L,R\},1}|^2}{|v_{\{L,R\},1}|^2 + \delta} \tag{3.17}$$

and the output SNR for the left and right device using (3.7) is given as

$$\mathrm{SNR}_{\mathrm{OUT},\{L,R\}} = \frac{\mathbf{w}_{\eta,\{L,R\}}^T\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\eta,\{L,R\}}}{\mathbf{w}_{\eta,\{L,R\}}^T\mathbf{R}_{\mathbf{nn}}\mathbf{w}_{\eta,\{L,R\}}} \tag{3.18}$$

$$= \frac{\frac{(1-\eta)^2\lambda^2 P_{s,\{L,R\}}}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta\lambda P_{s,\{L,R\}}}{(\mu+\lambda)} + \eta^2 P_{s,\{L,R\}}}{\frac{(1-\eta)^2\lambda P_{s,\{L,R\}}}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta P_{s,\{L,R\}}}{(\mu+\lambda)} + \eta^2 P_{n,\{L,R\}}} . \tag{3.19}$$

Looking at the limiting cases of η, the output SNR is given as [16, 49]

$$\mathrm{SNR}_{\mathrm{OUT},\{L,R\}} = \begin{cases} \lambda & : \eta = 0 \\ \mathrm{SNR}_{\mathrm{IN},\{L,R\}} & : \eta = 1 \end{cases} \tag{3.20}$$

where (3.19) lies in this range for any other value of $0 < \eta < 1$. For $\eta = 0$, it is also observed that a uniform output SNR is achieved equal to λ as defined in (3.14), i.e., $\mathrm{SNR}_{\mathrm{OUT},L} = \mathrm{SNR}_{\mathrm{OUT},R}$.

As a particular case, a scenario where the speech and noise have equal steering vectors is now considered, such that $\frac{\mathbf{a}}{\|\mathbf{a}\|} = \frac{\mathbf{v}}{\|\mathbf{v}\|}$. This results in a minimum value for λ in (3.14) equal to $\frac{P_s\|\mathbf{a}\|^2}{\delta+\|\mathbf{v}\|^2}$. Since λ is the upper limit for the output SNR, a smaller value for λ will produce a smaller overall range of SNR improvement in (3.20). In the worst case, this can lead to $\text{SNR}_{\text{OUT},\{L,R\}} \approx \text{SNR}_{\text{IN},\{L,R\}}$ as shown experimentally in [116, 119] and in Section 3.7.3.

3.3.2 Binaural cues

The speech interaural transfer function (ITF) at the input and output is defined as [116]

$$\text{ITF}_{\text{IN},s} = \frac{\mathbf{e}_L^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_L}{\mathbf{e}_R^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_L} = \frac{\mathbf{e}_L^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_R}{\mathbf{e}_R^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_R} = \frac{P_{s,L}}{\phi_s} \tag{3.21}$$

$$\text{ITF}_{\text{OUT},s} = \frac{\mathbf{w}_{\eta,L}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\eta,L}}{\mathbf{w}_{\eta,R}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\eta,L}} = \frac{\mathbf{w}_{\eta,L}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\eta,R}}{\mathbf{w}_{\eta,R}^H\mathbf{R}_{\mathbf{xx}}\mathbf{w}_{\eta,R}} \tag{3.22}$$

where $\phi_s = \mathbf{e}_R^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_L$. Substituting (3.7) into (3.22) shows that the speech ITF is unaffected by the filtering process, i.e.,

$$\text{ITF}_{\text{IN},s} = \text{ITF}_{\text{OUT},s} \tag{3.23}$$

The noise ITF is defined in a similar manner at the input and output, i.e.,

$$\text{ITF}_{\text{IN},n} = \frac{\mathbf{e}_L^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_L}{\mathbf{e}_R^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_L} = \frac{\mathbf{e}_L^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_R}{\mathbf{e}_R^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_R} = \frac{P_{n,L}}{\phi_n} \tag{3.24}$$

$$\text{ITF}_{\text{OUT},n} = \frac{\mathbf{w}_{\eta,L}^H\mathbf{R}_{\mathbf{nn}}\mathbf{w}_{\eta,L}}{\mathbf{w}_{\eta,R}^H\mathbf{R}_{\mathbf{nn}}\mathbf{w}_{\eta,L}} = \frac{\mathbf{w}_{\eta,L}^H\mathbf{R}_{\mathbf{nn}}\mathbf{w}_{\eta,R}}{\mathbf{w}_{\eta,R}^H\mathbf{R}_{\mathbf{nn}}\mathbf{w}_{\eta,R}} \tag{3.25}$$

where $\phi_n = \mathbf{e}_R^T\mathbf{R}_{\mathbf{nn}}\mathbf{e}_L$. Substituting (3.7) into (3.25) results in

$$\text{ITF}_{\text{OUT},n} = \frac{\frac{(1-\eta)^2\lambda P_{s,L}}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta P_{s,L}}{(\mu+\lambda)} + \eta^2 P_{n,L}}{\frac{(1-\eta)^2\lambda\phi_s}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta\phi_s}{(\mu+\lambda)} + \eta^2\phi_n} \tag{3.26}$$

which means that the $\text{ITF}_{\text{OUT},n}$ will change depending on the value of η. Looking at the limiting cases of η

$$\text{ITF}_{\text{OUT},n} = \begin{cases} \text{ITF}_{\text{OUT},s} = \text{ITF}_{\text{IN},s} & : \eta = 0 \\ \text{ITF}_{\text{IN},n} & : \eta = 1 \end{cases} \tag{3.27}$$

which means that at lower values of η the noise ITF at the output will be closer to the speech ITF at the input and output whereas at higher values of η the noise ITF at the output is closer to the noise ITF at the input.

Using the ITF, two binaural cues for the speech and noise are also defined, namely the interaural level difference (ILD), and the interaural time difference (ITD) which are both known to contribute to the sound localization by the human auditory system [120]. In [16] these cues have been shown to be related to the amplitude and phase of the ITFs. The speech ILD and ITD at the input are given by

$$\mathrm{ILD}_{\mathrm{IN},s} = |\mathrm{ITF}_{\mathrm{IN},s}| \tag{3.28}$$

$$= \frac{\mathbf{e}_L^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_L}{\mathbf{e}_R^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_R} \tag{3.29}$$

$$= \frac{P_{s,L}}{P_{s,R}} \tag{3.30}$$

$$\mathrm{ITD}_{\mathrm{IN},s} = \angle\left(\mathrm{ITF}_{\mathrm{IN},s}\right) \tag{3.31}$$

$$= \angle\left(\frac{\mathbf{e}_L^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_R}{\mathbf{e}_R^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_R}\right) \tag{3.32}$$

$$= \angle\left(\phi_s\right) \tag{3.33}$$

and at the output are given by

$$\mathrm{ILD}_{\mathrm{OUT},s} = \frac{\mathbf{w}_{\eta,L}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}_{\eta,L}}{\mathbf{w}_{\eta,R}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}_{\eta,R}} \tag{3.34}$$

$$\mathrm{ITD}_{\mathrm{OUT},s} = \angle\left(\mathbf{w}_{\eta,L}^H \mathbf{R}_{\mathbf{xx}} \mathbf{w}_{\eta,R}^H\right) \ . \tag{3.35}$$

Since the speech ITF is unaffected by the filtering process, the binaural speech cues are also preserved independent of the value of η, i.e.,

$$\mathrm{ILD}_{\mathrm{OUT},s} = \mathrm{ILD}_{\mathrm{IN},s}, \quad \mathrm{ITD}_{\mathrm{OUT},s} = \mathrm{ITD}_{\mathrm{IN},s} \ . \tag{3.36}$$

The same analysis can be applied to the binaural noise cues where the noise ILD and ITD at the input are given by

$$\mathrm{ILD}_{\mathrm{IN},n} = |\mathrm{ITF}_{\mathrm{IN},n}| \tag{3.37}$$

$$= \frac{\mathbf{e}_L^T \mathbf{R_{nn}} \mathbf{e}_L}{\mathbf{e}_R^T \mathbf{R_{nn}} \mathbf{e}_R} \tag{3.38}$$

$$= \frac{P_{n,L}}{P_{n,R}} \tag{3.39}$$

$$\mathrm{ITD}_{\mathrm{IN},n} = \angle\left(\mathrm{ITF}_{\mathrm{IN},n}\right) \tag{3.40}$$

$$= \angle\left(\frac{\mathbf{e}_L^T \mathbf{R_{nn}} \mathbf{e}_R}{\mathbf{e}_R^T \mathbf{R_{nn}} \mathbf{e}_R}\right) \tag{3.41}$$

$$= \angle\left(\phi_n\right) \ . \tag{3.42}$$

However, as shown in (3.26), since the noise ITF is affected by the filtering process, the binaural noise cues will not be preserved. Using (3.7) the noise ILD and ITD at the output are given as

$$\mathrm{ILD}_{\mathrm{OUT},n} = \frac{\mathbf{w}_{\eta,L}^H \mathbf{R_{nn}} \mathbf{w}_{\eta,L}}{\mathbf{w}_{\eta,R}^H \mathbf{R_{nn}} \mathbf{w}_{\eta,R}} \tag{3.43}$$

$$= \frac{\frac{(1-\eta)^2 \lambda P_{s,L}}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta P_{s,L}}{(\mu+\lambda)} + \eta^2 P_{n,L}}{\frac{(1-\eta)^2 \lambda P_{s,R}}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta P_{s,R}}{(\mu+\lambda)} + \eta^2 P_{n,R}} \tag{3.44}$$

$$\mathrm{ITD}_{\mathrm{OUT},n} = \angle\left(\mathbf{w}_{\eta,L}^H \mathbf{R_{nn}} \mathbf{w}_{\eta,R}^H\right) \tag{3.45}$$

$$= \angle\left(\frac{(1-\eta)^2 \lambda \phi_s}{(\mu+\lambda)^2} + \frac{2(1-\eta)\eta\phi_s}{(\mu+\lambda)} + \eta^2 \phi_n\right) \ . \tag{3.46}$$

Again, looking at the limiting cases of η results in

$$\mathrm{ILD}_{\mathrm{OUT},n} = \begin{cases} \mathrm{ILD}_{\mathrm{OUT},s} = \mathrm{ILD}_{\mathrm{IN},s} & : \eta = 0 \\ \mathrm{ILD}_{\mathrm{IN},n} & : \eta = 1 \end{cases} , \tag{3.47}$$

$$\mathrm{ITD}_{\mathrm{OUT},n} = \begin{cases} \mathrm{ITD}_{\mathrm{OUT},s} = \mathrm{ITD}_{\mathrm{IN},s} & : \eta = 0 \\ \mathrm{ITD}_{\mathrm{IN},n} & : \eta = 1 \end{cases} \tag{3.48}$$

which shows that the noise ILD and ITD at the output will be closer to the speech ILD and ITD at the input and output with lower values of η, whereas at higher values of η the noise ILD and ITD at the output are closer to the noise ILD and ITD at the input.

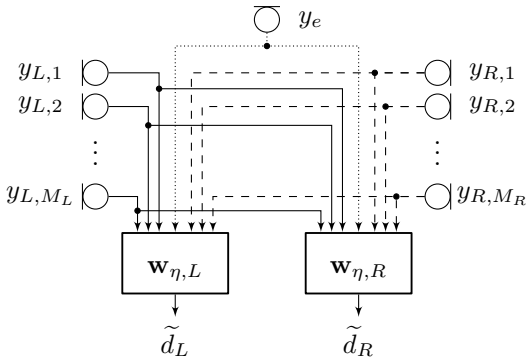


Figure 3.2: The general binaural noise reduction system with a remote microphone signal.

3.4 Data model with a remote microphone signal

In the previous section it was shown that when using the MWF$_\eta$, the binaural noise cues will always be distorted for $\eta \neq 1$. In fact, as more emphasis is put on preserving the binaural noise cues, i.e., a higher value of η is used, this will result in a decrease in the output SNR. In this section it is shown that with the addition of a remote microphone signal with a high input SNR, that not only is there an increase in the output SNR up to the targeted $\frac{1}{\eta^2}$ mentioned in Section 3.2, but also an increase in preservation of the binaural noise cues.

It is now supposed that a remote microphone signal, y_e, becomes available that can be used by both devices for the filtering process, where a depiction of this modified binaural noise reduction system is given in Figure 3.2. Unlike current binaural systems that mostly disable the local microphones when a remote microphone signal is available, y_e is incorporated into the noise reduction.

The speech and noise steering vectors with this remote microphone signal included are defined as $\mathbf{a}_e = \begin{bmatrix}\mathbf{a}^T & a_e\end{bmatrix}^T$, $\mathbf{v}_e = \begin{bmatrix}\mathbf{v}^T & v_e\end{bmatrix}^T$, respectively, and the new correlation matrices that incorporate these statistics as

$$\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e} = \mathcal{E}\{\mathbf{x}_e\mathbf{x}_e^H\},\ \mathbf{R}_{\mathbf{n}_e\mathbf{n}_e} = \mathcal{E}\{\mathbf{n}_e\mathbf{n}_e^H\} \tag{3.49}$$

where $\mathbf{x}_e = \begin{bmatrix}\mathbf{x}^T & x_e\end{bmatrix}^T$ and $\mathbf{n}_e = \begin{bmatrix}\mathbf{n}^T & n_e\end{bmatrix}^T$. The speech correlation matrix, which is still a rank-1 matrix, can be represented as

$$\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e} = P_s\mathbf{a}_e\mathbf{a}_e^H \tag{3.50}$$

and the noise correlation matrix can be represented as

$$\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e} = \mathbf{v}_e\mathbf{v}_e^H + \left[\begin{array}{c|c} \delta\mathbf{I} & \mathbf{0} \\ \hline \mathbf{0} & \delta_e \end{array}\right] \tag{3.51}$$

where δ_e is the incoherent noise power in the remote microphone signal. Similarly to (3.14), λ_e is defined using (3.50) and (3.51) as

$$\mathrm{Tr}\{\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}^{-1}\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}\} = P_s\left[\frac{||\mathbf{a}||^2}{\delta} + \frac{|a_e|^2}{\delta_e} - \frac{\left|\frac{\mathbf{a}^H\mathbf{v}}{\delta} + \frac{a_e^* v_e}{\delta_e}\right|^2}{1 + \frac{||\mathbf{v}||^2}{\delta} + \frac{|v_e|^2}{\delta_e}}\right] \tag{3.52}$$

$$= \lambda_e \tag{3.53}$$

where it is shown in Appendix B, that with the addition of the remote microphone signal $\lambda_e \geq \lambda$. In the case that the incoherent noise power in the remote microphone signal is the same as the other microphones, $\delta_e = \delta$, (3.52) reduces to

$$\lambda_e = \frac{P_s}{\delta}\left[||\mathbf{a}_e||^2 - \frac{|\mathbf{a}_e^H\mathbf{v}_e|^2}{\delta + ||\mathbf{v}_e||^2}\right] . \tag{3.54}$$

In the extreme case, when the remote microphone signal does not contain any noise component that is coherent with the noise in the other microphones, i.e., $v_e \to 0$ (3.53) can be given as

$$\lambda_e = \lambda + \frac{P_s|a_e|^2}{\delta_e} \tag{3.55}$$

where δ_e is the noise power of the incoherent noise in the remote microphone signal.

In the particular case when the speech and noise have similar steering vectors, an improvement in the output SNR is still possible, whereas the left and right array alone may not improve the output SNR as described in Section 3.3.1 . Note that, even if the remote microphone contains a noise component that is correlated to the coherent noise in the other microphones, the additional spatial diversity usually disambiguates the noise and the speech steering vector, i.e., $\frac{\mathbf{v}_e}{\|\mathbf{v}_e\|} \neq \frac{\mathbf{a}_e}{\|\mathbf{a}_e\|}$, even when $\frac{\mathbf{v}}{\|\mathbf{v}\|} = \frac{\mathbf{a}}{\|\mathbf{a}\|}$.

3.5 Performance measures with a remote microphone signal

Since the binaural speech cues obviously continue to be preserved for any value of η, only the performance of the output SNR and the binaural noise cues with a remote microphone signal will be analyzed.

3.5.1 Output SNR with a remote microphone signal

It is first shown that the output SNR with the addition of a remote microphone signal, $\mathrm{SNR}^{e}_{\mathrm{OUT},\{L,R\}}$, for a fixed value of μ and η is always greater than or equal to the output SNR without the remote microphone signal, i.e., $\mathrm{SNR}^{e}_{\mathrm{OUT},\{L,R\}} \geq \mathrm{SNR}_{\mathrm{OUT},\{L,R\}}$. This relies on the fact that $\lambda_e \geq \lambda$, as shown in the appendix.

The output SNR with the addition of a remote microphone signal is given the same as in (3.19) with λ_e substituted for λ, i.e.,

$$\mathrm{SNR}^{e}_{\mathrm{OUT},\{L,R\}} = \frac{\frac{(1-\eta)^2\lambda_e^2 P_{s,\{L,R\}}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta)\eta\lambda_e P_{s,\{L,R\}}}{(\mu+\lambda_e)} + \eta^2 P_{s,\{L,R\}}}{\frac{(1-\eta)^2\lambda_e P_{s,\{L,R\}}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta)\eta P_{s,\{L,R\}}}{(\mu+\lambda_e)} + \eta^2 P_{n,\{L,R\}}}. \tag{3.56}$$

The derivative of the numerator of (3.19) with respect to λ is given as

$$\frac{2\mu(1-\eta)(\eta\mu+\lambda)P_{s,\{L,R\}}}{(\mu+\lambda)^3} \geq 0 \tag{3.57}$$

which shows that the numerator is an increasing function with respect to λ. Therefore, the numerator of (3.56) will always be greater than or equal to the numerator of (3.19) since $\lambda_e \geq \lambda$. Likewise, by inspection, the denominator of (3.19) is always decreasing toward $\eta^2 P_{n,\{L,R\}}$ with increasing λ which means that the denominator of (3.56) will always be less than or equal to the denominator of (3.19) again since $\lambda_e \geq \lambda$. This then shows that $\mathrm{SNR}^{e}_{\mathrm{OUT},\{L,R\}} \geq \mathrm{SNR}_{\mathrm{OUT},\{L,R\}}$. However, observing the limiting case of $\lambda_e \to +\infty$, e.g., in the case that the noise power in the remote microphone signal $\delta_e \to 0$ (cfr. (3.55)),

$$\lim_{\lambda_e \to +\infty} \mathrm{SNR}^{e}_{\mathrm{OUT},\{L,R\}} = \frac{\mathrm{SNR}_{\mathrm{IN},\{L,R\}}}{\eta^2} \tag{3.58}$$

which shows that with increasing values of λ_e, the closer the solution approaches the target SNR increase of $\frac{1}{\eta^2}$. This is of course because a reduction of output noise amplitude with a factor η has specifically been targeted in the original cost function (3.6). Indeed, the aim was to preserve some residual noise containing the original binaural noise cues.

3.5.2 Binaural noise cues with a remote microphone signal

A similar analysis can be performed with the binaural noise cues using a remote microphone signal. The noise ITF, ILD and ITD at the output are now denoted as $\mathrm{ITF}^{e}_{\mathrm{OUT},n}$, $\mathrm{ILD}^{e}_{\mathrm{OUT},n}$ and $\mathrm{ITD}^{e}_{\mathrm{OUT},n}$ respectively with λ_e substituted for λ in (3.26), (3.44) and (3.46).

The numerator and denominator of the noise ITF in (3.26) both exhibit a monotonic decrease with increasing λ to $\eta^2 P_{n,L}$ and $\eta^2\phi_n$ respectively. This implies that with larger values of λ, the noise ITF will be better preserved, i.e., since $\lambda_e \geq \lambda$ the noise ITF will always be equivalent or better preserved with the addition of a remote microphone signal. A similar limiting case for the noise ITF can be observed with the addition of a remote microphone signal as $\lambda_e \to +\infty$. Using this assumption and substituting λ_e for λ in (3.26) yields

$$\lim_{\lambda_e \to +\infty} \mathrm{ITF}^{e}_{\mathrm{OUT},n} = \frac{P_{n,L}}{\phi_n} \tag{3.59}$$

$$= \mathrm{ITF}_{\mathrm{IN},n} \tag{3.60}$$

which shows that noise ITF will be preserved. Since the noise ITF will be preserved, it is therefore straightforward to show that the noise ILD and ITD will also be preserved. The numerator and the denominator of the noise ILD (3.44), exhibit the same monotonic decrease with increasing λ as the noise ITF. Therefore, it will also be better preserved for larger value of λ which, in the limiting case, results in

$$\lim_{\lambda_e \to +\infty} \mathrm{ILD}^{e}_{\mathrm{OUT},n} = \mathrm{ILD}_{\mathrm{IN},n} \ . \tag{3.61}$$

The noise ITD (3.46) also exhibits the same monotonic decrease with increasing λ as the denominator of the noise ITF which, in the limiting case, results in

$$\lim_{\lambda_e \to +\infty} \mathrm{ITD}^{e}_{\mathrm{OUT},n} = \mathrm{ITD}_{\mathrm{IN},n} \ . \tag{3.62}$$

If fact, it is observed that all three binaural noise cues are affected in the same manner by λ where a larger value results in better preservation of the cues. Due to the same dependence on λ_e, the noise $\mathrm{ITF}^{e}_{\mathrm{OUT},n}$, $\mathrm{ILD}^{e}_{\mathrm{OUT},n}$ and $\mathrm{ITD}^{e}_{\mathrm{OUT},n}$, all converge to the limiting case at the same rate.

A remarkable aspect of this is that the remote microphone signal, which does not itself contain any information pertaining to the binaural noise cues, still aids in their preservation. If (3.7) is defined using the correlation matrices

containing the statistics with the remote microphone signal given in (3.49) then

$$\mathbf{w}_{\eta,L} = (1-\eta)\frac{\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}^{-1}\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}\mathbf{e}_L}{\mu + \mathrm{Tr}\{\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}^{-1}\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}\}} + \eta\mathbf{e}_L \tag{3.63}$$

$$= (1-\eta)\mathbf{w}_{0,L} + \eta\mathbf{e}_L \tag{3.64}$$

where $\mathbf{w}_{0,L}$ is (3.63) with $\eta = 0$ (note $\mathbf{w}_{\eta,R}$ and $\mathbf{w}_{0,R}$ are defined in the same manner). It is observed that the first term produces the speech with the correct binaural cues, plus residual noise with the wrong binaural cues, and the second term produces the correct binaural noise (and speech) cues, where the noise level is reduced by the target amount η. The function of the remote microphone signal is to improve the performance of $\mathbf{w}_{0,L}$ and $\mathbf{w}_{0,R}$, leading to less residual noise produced by the first term. As a consequence, $\mathbf{w}_{\eta,L}$ and $\mathbf{w}_{\eta,R}$ lead to better binaural noise cues due to the second term.

3.6 Adaptive partial noise estimation

In the previous section, it was shown that adding a remote microphone signal was able to increase the output SNR to the targeted $\frac{1}{\eta^2}$ while simultaneously improving the binaural noise cues. In the derivations, it was assumed that η was constant before and after the addition of the remote microphone signal. However, η can be adapted to a new value of η_e to further improve one performance measure while limiting the improvement of the other, e.g., some of the improvement in the binaural noise cues could be sacrificed to further increase the output SNR and vice versa.

Therefore, in this section, different methods of adapting η to a new value of η_e are proposed while constraining the performance of either the output SNR or binaural noise cues. It is shown, that by constraining the binaural noise cues to the same performance before and after the addition of a remote microphone signal by adjusting the value of η to a new value of η_e, the output SNR is able to further be increased. Conversely, if the output SNR performance is constrained to the same value before and after the addition of a remote microphone signal by adjusting η to a new value of η_e, then the binaural noise cues are better preserved.

3.6.1 Increased output SNR

The noise ILD is constrained in order to adjust the value of η to a new value of η_e. The choice to constrain the noise ILD[1] is done for the ease of exposition, however, since $\text{ITF}^e_{\text{OUT},n}$, $\text{ILD}^e_{\text{OUT},n}$ and $\text{ITD}^e_{\text{OUT},n}$ all have the same dependence on η and λ_e, the change to η_e will affect all three identically. The new value of η_e is found by equating $\text{ILD}_{\text{OUT},n}$ to $\text{ILD}^e_{\text{OUT},n}$ which yields

$$\text{ILD}_{\text{OUT},n}(\eta) = \text{ILD}^e_{\text{OUT},n}(\eta_e) \tag{3.65}$$

$$= \frac{\frac{(1-\eta_e)^2\lambda_e P_{s,L}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta_e)\eta_e P_{s,L}}{\mu+\lambda_e} + \eta_e^2 P_{n,L}}{\frac{(1-\eta_e)^2\lambda_e P_{s,R}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta_e)\eta_e P_{s,R}}{\mu+\lambda_e} + \eta_e^2 P_{n,R}} \tag{3.66}$$

which is a quadratic expression in terms of η_e and has a closed-form solution. As shown in Section 3.5, with the addition of a remote microphone signal, the binaural noise cues are better preserved, i.e., $\text{ILD}^e_{\text{OUT},n}$ will be closer to $\text{ILD}_{\text{IN},n}$ if the constraint of (3.65) is not present. This implies then, in order to reach the same performance before the addition of the remote microphone signal, $\text{ILD}_{\text{OUT},n}(\eta)$, that $\eta_e \leq \eta$. When plugging η_e in (3.56) this results in an increase in the output SNR compared to (3.56) with η. This method of adapting η, therefore, sacrifices the increase in the binaural noise cue preservation while simultaneously increasing the output SNR.

3.6.2 Increased binaural noise cue preservation

In a similar manner to Section 3.6.1, the value of η is adjusted to a new value of η_e based on constraining the output SNR. The new value of η_e is found by equating the output SNR without the remote microphone signal, $\text{SNR}_{\text{OUT},\{L,R\}}$, to the output SNR with the remote microphone signal, $\text{SNR}^e_{\text{OUT},\{L,R\}}$, yielding

$$\text{SNR}_{\text{OUT},\{L,R\}}(\eta) = \text{SNR}^e_{\text{OUT},\{L,R\}}(\eta_e) \tag{3.67}$$

$$= \frac{\frac{(1-\eta_e)^2\lambda_e^2 P_{s,\{L,R\}}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta_e)\eta_e\lambda_e P_{s,\{L,R\}}}{(\mu+\lambda_e)} + \eta_e^2 P_{s,\{L,R\}}}{\frac{(1-\eta_e)^2\lambda_e P_{s,\{L,R\}}}{(\mu+\lambda_e)^2} + \frac{2(1-\eta_e)\eta_e P_{s,\{L,R\}}}{(\mu+\lambda_e)} + \eta_e^2 P_{n,\{L,R\}}} \tag{3.68}$$

which is again a quadratic expression in terms of η_e. Since, in Section 3.5.1 it was shown that $\text{SNR}^e_{\text{OUT},\{L,R\}}(\eta) \geq \text{SNR}_{\text{OUT},\{L,R\}}(\eta)$, then in order to achieve

[1]Note that instead of equating $\text{ILD}_{\text{OUT},n}(\eta)$ and $\text{ILD}^e_{\text{OUT},n}(\eta_e)$ in order to find η_e, a predefined threshold for the noise ILD could also be used to ensure a certain level of binaural noise cue preservation.

the same output SNR before and after the addition of the remote microphone signal $\eta_e \geq \eta$, thereby sacrificing the improvement in the output SNR. This larger value of η_e, however, simultaneously further improves the binaural noise cue preservation. Again since $\text{ITF}^e_{\text{OUT},n}(\eta_e)$, $\text{ILD}^e_{\text{OUT},n}(\eta_e)$ and $\text{ITD}^e_{\text{OUT},n}(\eta_e)$ all have the same dependence on η_e, this will lead to an equivalent preservation in all three binaural noise cues.

However, since $\text{SNR}_{\text{OUT},L} \neq \text{SNR}_{\text{OUT},R}\ , \forall \eta \neq 0$, the device with the smallest difference between $\text{SNR}_{\text{OUT},\{L,R\}}$ and $\text{SNR}^e_{\text{OUT},\{L,R\}}$ should be used as not to decrease the output SNR of either device to below the case without a remote microphone signal.

3.7 Simulations

In this section the effects on the previously defined performance measures using the MWF_η were analyzed with the addition of a remote microphone signal. The performance measures were also analyzed when η was adapted based on Section 3.6. Simulations were performed using the head-related impulse responses (HRIRs) provided by the database in [121]. The acoustic signals for the noise and speech were convolved with the HRIRs in order to produce the desired acoustic environment. The location of the speech and noise source are represented as θ_s and θ_n respectively, where $\theta_{\{s,n\}} = 0^\circ$ indicates the source was directly in front of the devices and $\theta_{\{s,n\}}$ increases in a clock-wise direction, e.g., $\theta_{\{s,n\}} = 90^\circ$ indicates the source was at the right ear.

The speech and noise were positioned at a distance of 3m and the microphone setup consisted of three behind-the-ear (BTE) microphones on each device, $M = 3$, located at the front, middle and rear. Independent identically distributed noise that was equal to 1% of the average noise signal power at the microphones was also added to the signals. The trade-off parameter μ was set to the same value for all simulations, namely $\mu = 1$.

The simulations were performed in batch mode, which means that the filters were found using the full length of the speech and noise signals to estimate $\mathbf{R}_{\mathbf{xx}}$, $\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}$, $\mathbf{R}_{\mathbf{nn}}$ and $\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}$. In real-time scenarios, however, the devices typically use a VAD as described in Section 3.2, to gather the necessary statistics.

The performance measures presented in the previous sections were used in order to find the improvement in the output SNR and the binaural noise cues. The binaural speech cues were not analyzed as they are perfectly preserved as shown with the equivalence of the speech ITF (3.23). The improvement in the SNR

for the left and right devices is defined as

$$\Delta\text{SNR}_{\text{OUT},\{L,R\}} = 10\log_{10}(\text{SNR}^{e}_{\text{OUT},\{L,R\}}) - 10\log_{10}(\text{SNR}_{\text{IN},\{L,R\}}) \; . \quad (3.69)$$

The noise ILD error is given as

$$\Delta\text{ILD}_{\text{OUT},n} = 10\log_{10}(\text{ILD}^{e}_{\text{OUT},n}) - 10\log_{10}(\text{ILD}_{\text{IN},n}) \quad (3.70)$$

and a relative noise ITD error is given as

$$\Delta\text{ITD}_{\text{OUT},n} = \frac{|\text{ITD}^{e}_{\text{OUT},n} - \text{ITD}_{\text{IN},n}|}{\pi} \quad (3.71)$$

where $0 \leq \Delta\text{ITD}_{\text{OUT},n} \leq 1$.

The remote microphone signal consisted of the clean speech signal, not convolved with a HRIR, with additive noise that was incoherent with the spatially located noise source, i.e., $v_e = 0$. The additive noise on the remote microphone signal, δ_e (cfr. 3.55), was scaled in order to produce various input SNRs which is denoted as $\text{SNR}_{\text{IN},REM}$.

3.7.1 Performance measures with changing η and $\text{SNR}_{\text{IN},REM}$

The first simulation was performed in order to highlight the benefit of using a remote microphone signal with a high $\text{SNR}_{\text{IN},REM}$. The speech and noise sources were located at $\theta_s = 30°$ and $\theta_n = -90°$ where the input SNR of the reference microphone of the right device was $\text{SNR}_{\text{IN,R}} = 0$ dB. The $\text{SNR}_{\text{IN},REM}$ was varied from 0 dB to 40 dB in increments of 5 dB and the value for η was varied from 0 (no partial noise estimation) to 1 (no noise reduction).

Figure 3.3 shows $\Delta\text{SNR}_{\text{OUT},R}$ as a function of both η and $\text{SNR}_{\text{IN},REM}$ for a frequency of $\omega_l = 1500\ Hz$. Since $1500\ Hz$ is roughly the cut-off when ITDs become less important and ILDs become more important this frequency gives a good overview of the performance with respect to both binaural noise cues.

For small values of η, it is observed that the remote microphone signal is able to provide a large increase in $\Delta\text{SNR}_{\text{OUT},R}$, while at larger values of η, since almost no noise reduction is performed, there is little to no improvement of $\Delta\text{SNR}_{\text{OUT},R}$. It is also observed that $\Delta\text{SNR}_{\text{OUT},R}$ reaches the target SNR increase $\frac{1}{\eta^2}$ (represented by ■ in Figure 3.3) for large values of η even with a low input SNR of the remote microphone signal. However, at lower values of η, the remote microphone signal is able to boost the output SNR performance of the system to the target SNR increase of $\frac{1}{\eta^2}$ for sufficiently high values of $\text{SNR}_{\text{IN},REM}$. It is noted that while the broadband $\text{SNR}_{\text{IN},REM}$ was in the range

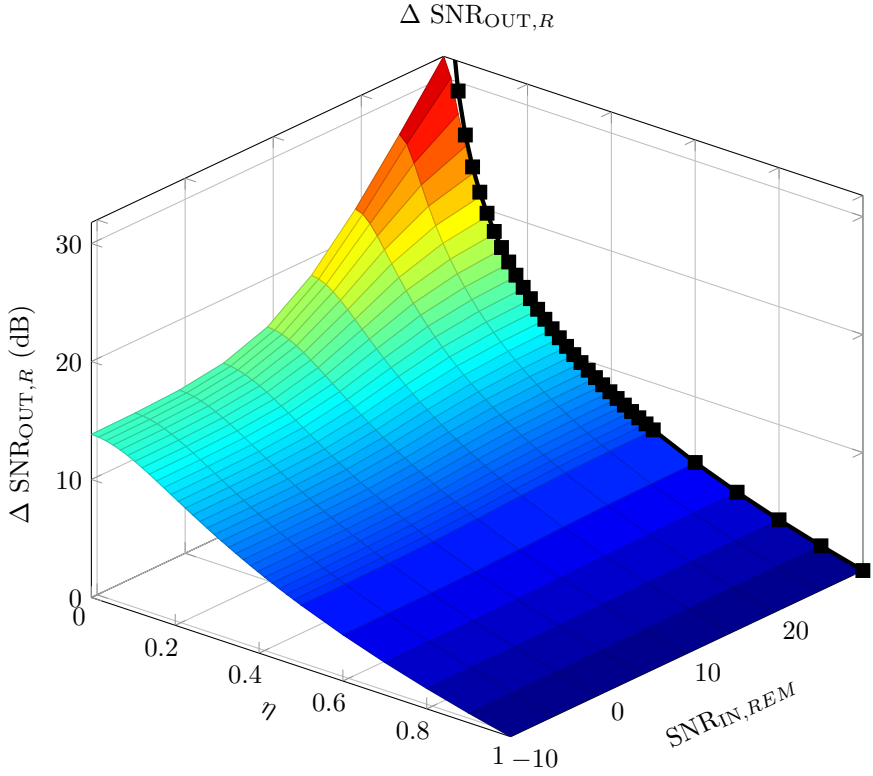


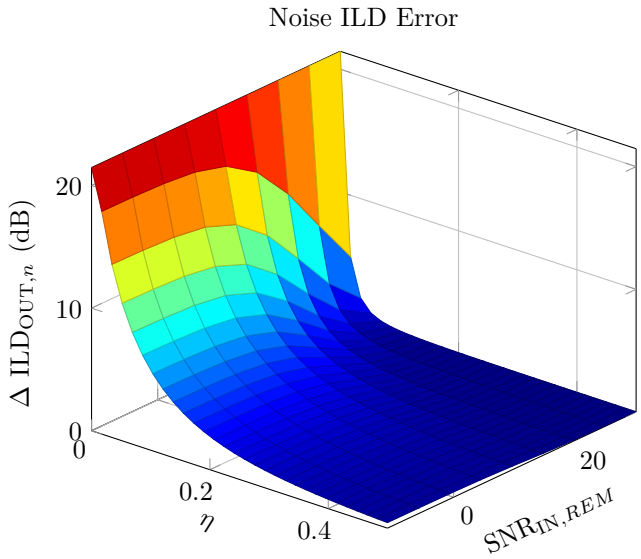
Figure 3.3: $\Delta\text{SNR}_{\text{OUT},R}$ for various values of η and $\text{SNR}_{\text{IN},REM}$ along with target of $\frac{1}{\eta^2}$.

of 0 dB and 40 dB, the range of the narrowband $\text{SNR}_{\text{IN},REM}(\omega_l)$ is somewhat lower.

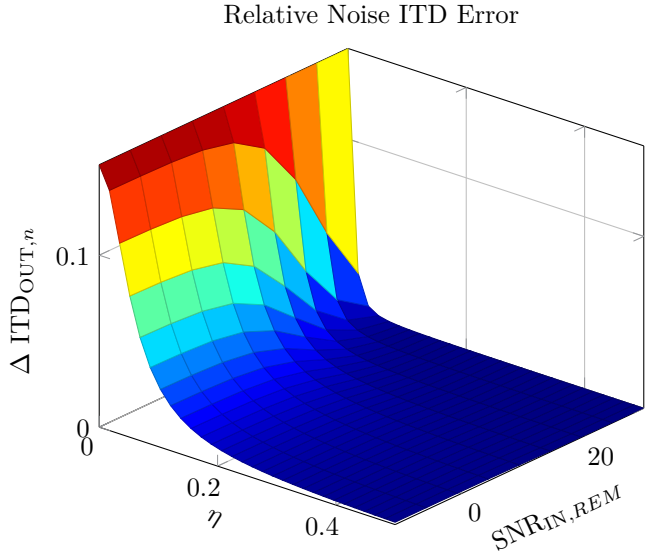
The noise ILD and relative noise ITD error for $\omega_l = 1500\ Hz$ are shown in Figure 3.4. It is observed that for small values of η the largest error is produced as there is very little partial noise estimation. The errors become smaller with increasing η and $\text{SNR}_{\text{IN},REM}$. Unlike Figure 3.3, η is in the range of $0 \leq \eta \leq 0.5$ as larger values offer little benefit to those already shown.

3.7.2 Adaptive η

Simulations are now performed where η is adapted as described in Section 3.6. The adapted η_e is first found by constraining the ILD as described in Section



(a) Noise ILD error for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$.



(b) Relative noise ITD error for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$.

Figure 3.4: The noise ILD and relative noise ITD error for a single frequency ($\omega_l = 1500\ Hz$) for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$.

3.6.1 which allows for an increase in the output SNR.

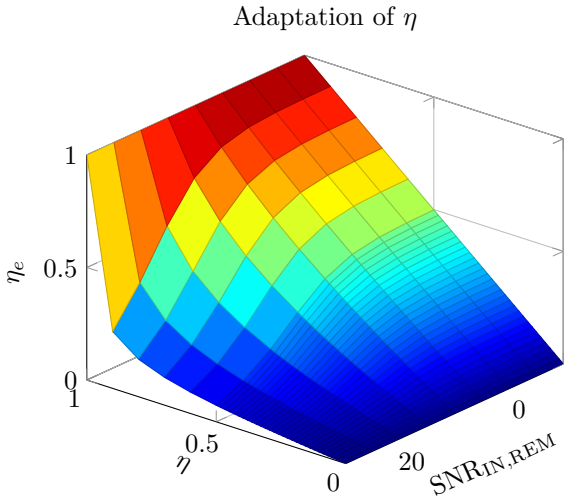
Figure 3.5 and Figure 3.6 show the adapted η_e, $\Delta\text{SNR}_{\text{OUT},R}$, the noise ILD errors and the relative noise ITD errors respectively for various values of η and $\text{SNR}_{\text{IN},REM}$. It is observed that by sacrificing the improvement in the binaural cue preservation, $\Delta\text{SNR}_{\text{OUT},R}$ is further improved (the same holds true for $\Delta\text{SNR}_{\text{OUT},L}$). However, as shown in Figures 3.6a and 3.6b this comes at the expense of keeping the binaural noise cues the same with and without the remote microphone signal.

Next, η_e is adapted by constraining the SNR as described in Section 3.6.2 which allows for an increase in the binaural noise cue preservation. The output SNR of the left device was used as the constraint as $\text{SNR}^e_{\text{OUT},L} - \text{SNR}_{\text{OUT},L} < \text{SNR}^e_{\text{OUT},R} - \text{SNR}_{\text{OUT},R}$ which guarantees that the output SNR of either device is not reduced to below $\text{SNR}_{\text{OUT},\{L,R\}}$. However, since the left device has essentially reached the target SNR increase $\frac{1}{\eta^2}$ even without the remote microphone signal, there is only a marginal adaption of η as shown in Figure 3.7. This marginal adaptation of η also results in almost no further increase in preservation in the binaural noise cues. This lack of adaptation is typically the result of a large increase in the output SNR of the device that has the lowest input SNR, due to using signals from the higher input SNR device, and results in the device reaching the $\frac{1}{\eta^2}$ even without a remote microphone signal.

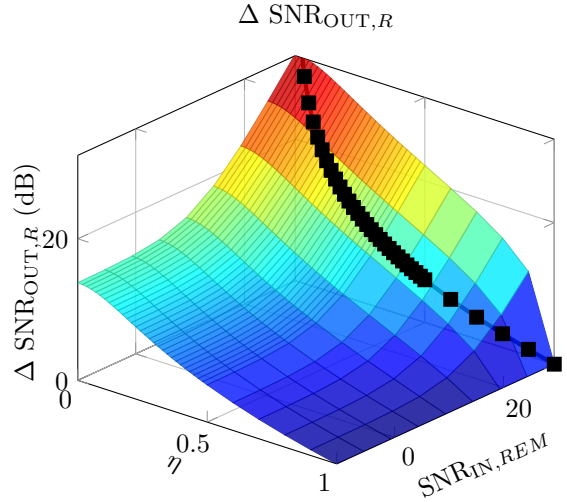
3.7.3 Influence of noise source position

Another interesting aspect is to observe the effect on the binaural noise cues for different θ_n values relative to θ_s. The speech source was placed at $\theta_s = 0°$ and the noise source position was varied in the range of $-180° \leq \theta_n < 180°$ in increments of 10°. The $\Delta\text{SNR}_{\text{OUT},R}$ and the noise ILD error were found at each noise position without a remote microphone signal, with a remote microphone signal where $\eta = 0.2$ and $\text{SNR}_{\text{IN},REM} = 40$ dB, and finally where η was adapted. The relative noise ITD is not plotted as the performance in terms of θ_n mimics the ILD.

The top of Figure 3.8 shows the $\Delta\text{SNR}_{\text{OUT},R}$ versus various values of θ_n. It is observed that without the remote microphone signal, when $\theta_s = \theta_n$ there is no improvement in the SNR. This is due to the fact that the MWF_η, and beamformers in general, cannot provide a benefit when the speech and noise come from similar directions relative to the array [22]. Remarkably, it is observed that with the addition of the remote microphone signal, when $\theta_s \approx \theta_n$ there is a significant SNR increase, reaching the target SNR increase $\frac{1}{\eta^2}$. In using η_e, where the improvement of the binaural noise cues is sacrificed, a larger

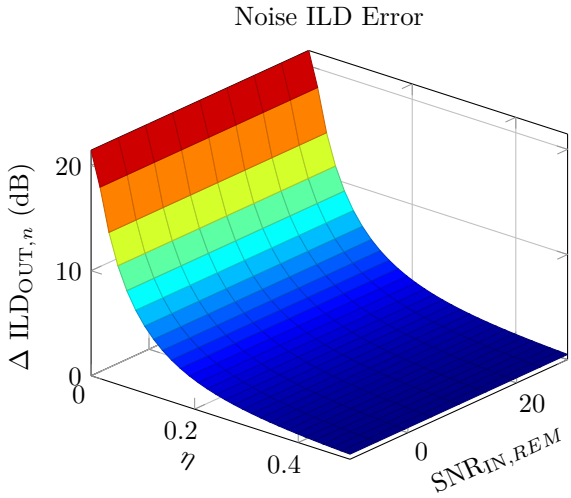


(a) Adaptation of η based on constraining the noise ILD.

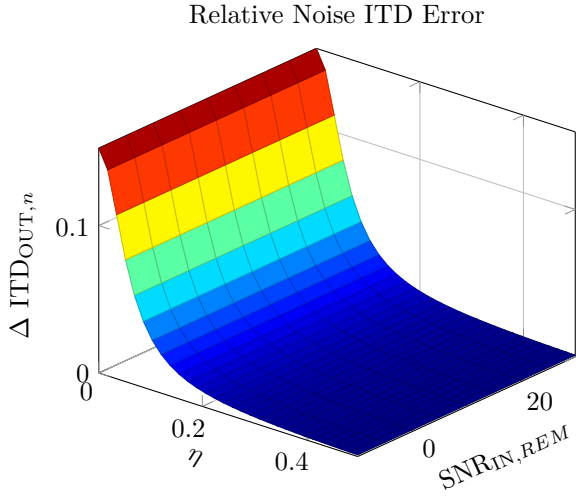


(b) $\Delta\text{SNR}_{\text{OUT},R}$ (overlay) with adapting η where the target $\frac{1}{\eta^2}$ is exceeded.

Figure 3.5: Adapted η and $\Delta\text{SNR}_{\text{OUT},R}$ based on the noise ILD constraint for various values of η and $\text{SNR}_{\text{IN},REM}$. Note that the x-y axis of (a) are flipped for an easier graphical representation.

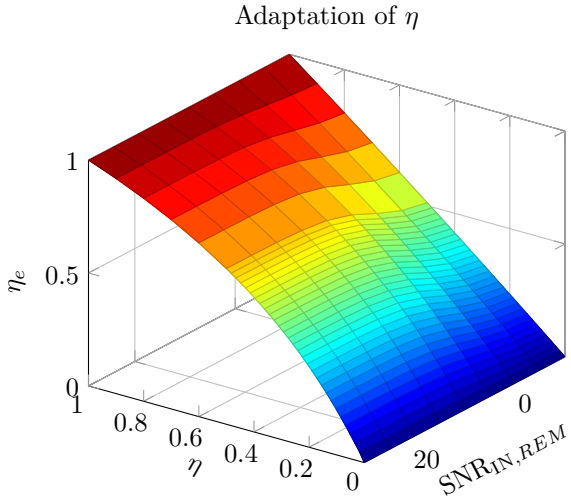


(a) Noise ILD error with adapting η.

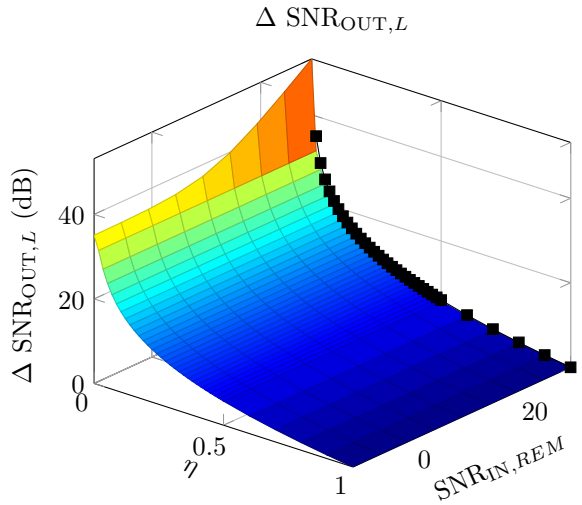


(b) Relative noise ITD with adapting η.

Figure 3.6: The noise ILD and relative noise ITD error based on the noise ILD constraint for various values of η and $\mathrm{SNR}_{\mathrm{IN},REM}$.

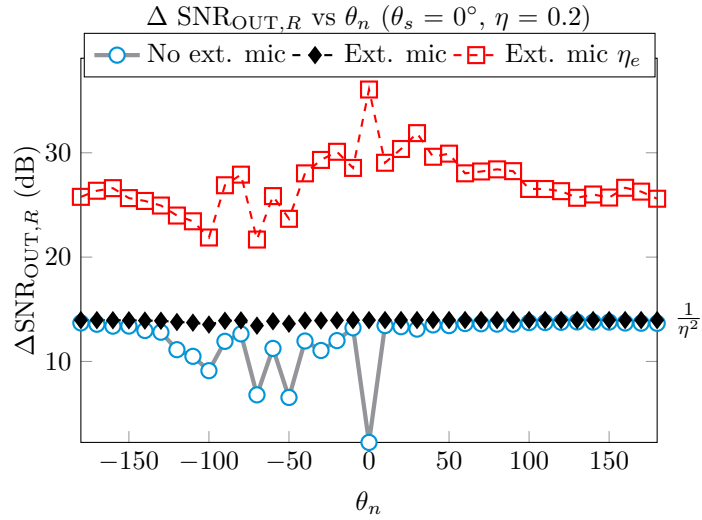


(a) Adaptation of η based on constraining $\Delta\text{SNR}_{\text{OUT},L,}$.

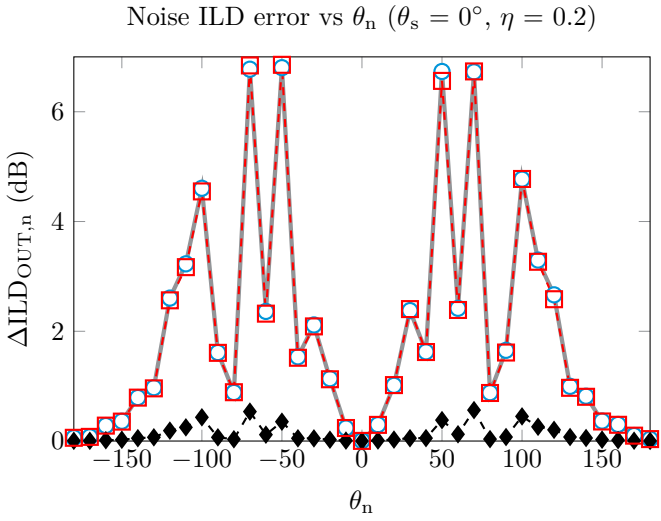


(b) $\Delta\text{SNR}_{\text{OUT},L}$ with constraining η_e.

Figure 3.7: Adapted η based on the $\Delta\text{SNR}_{\text{OUT},L}$ constraint and the resulting $\Delta\text{SNR}^{e}_{\text{OUT},L}(\eta_e)$ (Note that the x-y axis of (a) are flipped for an easier graphical representation).



(a) $\Delta\text{SNR}_{\text{OUT},R}$ for various values of θ_n.



(b) Noise ILD error for various values of θ_n.

Figure 3.8: The $\Delta\text{SNR}_{\text{OUT},L}$ and noise ILD error for a single frequency ($\omega_l = 1500\ Hz$) for various values of θ_n where $\theta_s = 0°$ and η is at a constant value, namely, $\eta = 0.2$.

improvement in $\Delta\text{SNR}_{\text{OUT},R}$ is observed. Again this comes at the expense of constraining the ILD to the same performance before and after the addition of the remote microphone signal as shown in Figure 3.8b.

3.8 Conclusion

A general binaural noise reduction system was presented that used the MWF_η in order to preserve the binaural speech and noise cues with the addition of a remote microphone signal. It was shown that the addition of this remote microphone signal not only resulted in an increase in the output SNR, but also simultaneously improved the binaural noise cue preservation. A remarkable aspect of this was that the remote microphone signal did not contain any information about the binaural noise cues but was still able to increase the binaural noise cue preservation. A modification was proposed allowing the partial noise estimation variable η to adapt to a new value of η_e by exploiting the trade-off between noise reduction performance and binaural noise cue preservation. This allowed for an increase in one of the performance measures while constraining the other to the same value before and after the addition of the remote microphone signal.

Chapter 4

Greedy distributed node selection for node-specific signal estimation in wireless sensor networks

4.1 Introduction

In the previous chapter, it was shown that adding a single signal to a current two node noise reduction system resulted in an increase in performance. This is now extended to a K node WSN, where each node has multiple sensors and shares these received sensor signal observations with every other node. As stated in Chapter 1, by using multiple nodes in a sensing environment, not only is more spatial information captured but signals from a more favorable location can be acquired, e.g. the remote microphone signal in the previous chapter was able to provide a benefit in terms of noise reduction due to increased spatial information.

However, due to the nodes being deployed over large distances and their limited battery life, it is often desired to limit the number of active nodes at any given time. Indeed, if the WSN is densely deployed, many sensors record redundant data and may be placed in an inactive or sleep mode in order to preserve the network lifetime. Therefore, the number of total active nodes in the system, K, should be reduced to a smaller subset, N. This is an inherent combinatorial

optimization problem with $\binom{K}{N}$ combinations. As the number of nodes K increases, the computational time required to find the optimal subset becomes unfeasible.

There are several methods that have been proposed in order to perform node selection in a WSN [122], where each of these methods differ in one or more ways from the techniques proposed in this chapter. Joshi *et. al.* have proposed a formulation that relies on a MAP estimator and uses a convex cone in order to measure the impact of removing a sensor, which relies on knowing the underlying statistics of the system [72]. Other methods are able to evaluate performance bounds compared to the optimal solution [71], using MAP or ML estimators. In using either an MAP or ML estimator it is assumed that the WSN is performing parameter estimation of the underlying statistics of the system, whereas the method proposed in this chapter aims to perform signal estimation. In [123] a distributed strategy has been proposed that is compared to a centralized approach but is cast as a multi-armed bandit problem which looks to activate a certain subset of sensors to aid in estimation in consensus based averaging which again aims to estimate a desired parameter of the system. Thatte *et. al.* have proposed placing bounds on the LMMSE under various network topologies but rely on a FC to perform the estimation [73] whereas the proposed method looks to accomplish this in a completely distributed fashion.

In order to select nodes from a given set, the concept of *utility* is introduced which is assigned to a node as a way to determine its importance to the signal estimation problem at hand [124–126]. It is defined as the increase or decrease of the LMMSE after removing or adding the respective node and re-optimizing the estimators. Since each node is assumed to have multiple sensors, a new utility computation algorithm is developed which efficiently computes the utility of a set of sensors at once, which then corresponds to the utility of the node.

Due to the distributed computation of the proposed utility as well as the combinatorial nature of node selection a distributed algorithm is devised that uses a greedy procedure to add and remove nodes from the network as in [127]. For node removal, at each iteration, the greedy procedure in this chapter will remove the node with the lowest utility. Likewise, for node addition, this greedy procedure will add the node with the highest utility. Similar greedy techniques have been applied to radar arrays [128, 129] where the change of the LMMSE is used for target detection. It should also be noted that the utility measure proposed in the presented framework differs from other definitions such as [71, 123, 130, 131] which rely on the concept of submodularity which the proposed utility does not have.

Although the exact utility can be computed in the centralized case, when

distributed estimation is performed by way of the DANSE algorithm the utility will be shown to represent upper and lower bounds on the LMMSE after node addition or removal, i.e., the increase or decrease of the LMMSE when removing or adding nodes respectively. By using the convergence and optimality properties of the DANSE algorithm, it is shown that the nodes can independently decide whether to stay active in the current network based on their local utility estimation. The network, therefore, does not need to rely on a FC in order to facilitate node selection. However, since the DANSE algorithm allows each node to estimate a node-specific signal, each sensor will also have a different utility for each individual estimation problem. Therefore, the computed utilities (and their bounds) are referenced to a common network-wide utility to circumvent this problem.

This chapter is organized as follows. In Section 4.2 the utility is described in a centralized scenario with a greedy node selection procedure, and a network-wide utility is defined that is common for all the node-specific estimation problems. In Section 4.3 the utility is described in a distributed scenario where the DANSE algorithm is in place and it is shown how it can be used in the greedy node selection as an upper and lower bound with respect to the increase or decrease to the LMMSE. Simulations are performed in Section 4.4 where the centralized and distributed scenarios are compared. An adaptive scenario is also simulated that shows the use of the utility and the greedy node selection procedure in a real-time environment. Finally, conclusions are given in Section 4.5.

4.2 Utility

Suppose that each node in the network has calculated its own $M \times Q$-dimensional optimal estimator (2.26) and, due to constraints imposed on the system and the possibility of new nodes becoming available, nodes should be removed or added to the network while controlling the effect on the LMMSE at each node. This problem is inherently combinatorial in nature, therefore greedy heuristics will be used. In order to determine the effect of removing or adding a node, a utility measure is introduced that quantifies how much a node contributes to the current estimation.

The utility of a node is defined as the difference in the LMMSE when one node is added or removed from the estimation after re-optimizing the estimator $\mathbf{W}_k$. In [124] it is described how the utility of a sensor can be computed with a relatively small computational complexity when compared to a naive approach which would be to remove one sensor from the system and then to recalculate the minimum cost to check its contribution and to do this for all sensors.

While in the centralized approach computational complexity may not be a concern, in the distributed case nodes often have a smaller processing capability. In this section, the techniques in [124] will be generalized to find an expression that computes the utility of a group of sensors (e.g., the M_k sensors corresponding to node k) instead of a single sensor. This expression will then later be used for node selection in the distributed scenario.

4.2.1 Utility for node removal

For convenience the minimum cost at node k using the optimal estimator (2.23) is repeated here as

$$\begin{aligned} J_k(\widehat{\mathbf{W}}_k) &= \mathcal{E}\left\{\left\|\mathbf{d}_k - \widehat{\mathbf{W}}_k^H \mathbf{y}\right\|_2^2\right\} \\ &= \sum_{j=1}^{Q} P_{k_{dj}} - \mathbf{r}_{\mathbf{yd}_{kj}}^H \widehat{\mathbf{w}}_{kj} \end{aligned} \tag{4.1}$$

where $\widehat{\mathbf{w}}_{kj}$ and $\mathbf{r}_{\mathbf{yd}_{kj}}$ represent the jth column of $\widehat{\mathbf{W}}_k$ and $\mathbf{R}_{\mathbf{yd}_k}$ respectively and $P_{k_{dj}} = \mathcal{E}\{|d_{kj}|^2\}$ represents the desired signal power in the jth channel.

A $Q \times Q$-dimensional LMMSE cost matrix is defined as

$$\widehat{\mathbf{J}}_k \triangleq \mathbf{R}_{\mathbf{d}_k\mathbf{d}_k} - \mathbf{R}_{\mathbf{yd}_k}^H \widehat{\mathbf{W}}_k \tag{4.2}$$

where the minimum cost can then be compactly represented as the trace of (4.2), i.e.,

$$J_k(\widehat{\mathbf{W}}_k) = \mathrm{Tr}\{\widehat{\mathbf{J}}_k\}. \tag{4.3}$$

Since it is assumed that the desired signal components of each node are linear mixtures of the same Q source signals in $\mathbf{s}$, it can be shown that the LMMSE cost matrices and LMMSE estimators between nodes are related to one another by their steering matrices. The LMMSE cost matrices between node k and node q are related, as shown in Appendix C.1, as

$$\widehat{\mathbf{J}}_k = \left(\overline{\mathbf{A}}_{qk}\right)^{-H} \widehat{\mathbf{J}}_q \left(\overline{\mathbf{A}}_{qk}\right)^{-1} \tag{4.4}$$

where $\overline{\mathbf{A}}_{qk} = \left(\overline{\mathbf{A}}_k\right)^{-H} \overline{\mathbf{A}}_q^H$. This is a direct result of $\mathbf{d}_k$ and $\mathbf{d}_q$ being related by their steering matrices $\overline{\mathbf{A}}_k$ and $\overline{\mathbf{A}}_q$, respectively.

The utility, $U_{k_{-q}}$, of node q's sensors, $\mathbf{y}_q$, with respect to the estimation problem of node k, is defined as the increase in LMMSE when $\mathbf{y}_q$ is removed from the estimation problem of node k, i.e.,

$$U_{k_{-q}} = J_{k_{-q}}(\widehat{\mathbf{W}}_{k_{-q}}) - J_k(\widehat{\mathbf{W}}_k) \tag{4.5}$$

where the subscript $-q$ indicates that node q's sensors are removed from the function and $\widehat{\mathbf{W}}_{k_{-q}}$ is referred to as the optimal fall-back estimator when node q is removed. Note that $\widehat{\mathbf{W}}_{k_{-q}}$ is *not* equal to $\widehat{\mathbf{W}}_k$ with M_q rows removed rather it is equal to the re-optimized LMMSE estimator that minimizes $J_{k_{-q}}$ in which the sensors of node q are removed. Using (4.3), the utility of node q with respect to the estimation problem of node k is given by

$$\begin{aligned} U_{k_{-q}} &= \mathrm{Tr}\{\widehat{\mathbf{J}}_{k_{-q}} - \widehat{\mathbf{J}}_k\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\mathbf{d}_k}\widehat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\widehat{\mathbf{W}}_{k_{-q}}\}. \end{aligned} \tag{4.6}$$

In order to calculate $\widehat{\mathbf{W}}_{k_{-q}}$ without having to take a full inverse as given in (2.26), the $M \times M$-dimensional sensor signal correlation matrix, $\mathbf{R}_{\mathbf{yy}}$, is first partitioned as follows

$$\mathbf{R}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} & \mathbf{R}_{\mathbf{y}_q\mathbf{y}_{-q}} \\ \mathbf{R}^H_{\mathbf{y}_q\mathbf{y}_{-q}} & \mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} \end{bmatrix} \tag{4.7}$$

where, for the sake of an easy exposition but w.l.o.g., it is assumed that the first $M_q \times M_q$ elements of the matrix correspond to node q's sensors (i.e., $q = 1$). Notice that to remove the sensors from another node, the indices would need to be shifted accordingly, which does not affect generality of the utility computation described in the sequel.

The calculation of $\widehat{\mathbf{W}}_{k_{-q}}$ using the definition given in (2.26), requires the inverse of only a portion of the sensor signal correlation matrix, namely $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, which is currently unknown. In order to calculate the inverse without having to first remove the corresponding rows and columns that pertain to the node q's sensors and calculate $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, the current inverse $\mathbf{R}^{-1}_{\mathbf{yy}}$ is block partitioned as

$$\mathbf{R}^{-1}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C} \end{bmatrix} \tag{4.8}$$

where $\mathbf{S}$ is an invertible $M_q \times M_q$-dimensional matrix, $\mathbf{V}$ is an $M_q \times (M - M_q)$-dimensional matrix and $\mathbf{C}$ is an $(M - M_q) \times (M - M_q)$-dimensional matrix. It is noted that this matrix inverse (including all of its block components) is already known from the computation of the current LMMSE estimator at node

k. Using the block form of the matrix inversion lemma [132], $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ may be calculated using only the known values in the current $\mathbf{R}^{-1}_{\mathbf{yy}}$ matrix as

$$\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} = \mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V} \tag{4.9}$$

which is the Schur complement of $\mathbf{S}$ in $\mathbf{R}^{-1}_{\mathbf{yy}}$.

The current optimal estimator $\widehat{\mathbf{W}}_k$ is also block partitioned as

$$\widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{kq} \\ \\ \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} \end{bmatrix} \tag{4.10}$$

where $\widehat{\mathbf{W}}_{kq}$ is an $M_q \times Q$-dimensional matrix that represents the estimator values applied to node q's sensors and $\widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}$ is an $(M - M_q) \times Q$-dimensional matrix that represents the estimator values applied to the other sensors. In Appendix C.2, it is shown that the optimal fall-back estimator is given as

$$\widehat{\mathbf{W}}_{k_{-q}} = \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H\mathbf{S}^{-1}\widehat{\mathbf{W}}_{kq} \tag{4.11}$$

which are all known values of the current estimate. Again shown in Appendix C.2, using (4.11) the utility is given as

$$U_{k_{-q}} = \mathrm{Tr}\{\widehat{\mathbf{W}}^H_{kq}\mathbf{S}^{-1}\widehat{\mathbf{W}}_{kq}\} \tag{4.12}$$

$$= \mathrm{Tr}\{\mathbf{U}_{k_{-q}}\} \tag{4.13}$$

where

$$\mathbf{U}_{k_{-q}} \triangleq \widehat{\mathbf{W}}^H_{kq}\mathbf{S}^{-1}\widehat{\mathbf{W}}_{kq} \tag{4.14}$$

which only relies on an inversion[1] of an $M_q \times M_q$ matrix $\mathbf{S}$, and the current estimator values. In order to find the impact of any other node to the current estimation of node k, $U_{k_{-i}}, i \in \mathcal{K}$, the estimator coefficients and partial inverse of (4.12) can be changed to the corresponding indices.

Instead of using (4.12), the utility could be found naively by removing a node's signals and calculating the new estimator $\widehat{\mathbf{W}}_{k_{-q}}$ which relies on the inverse of $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ having a worst case scenario $O(M - M_q)^3$ computational complexity. Figure 4.1 shows the computational complexity of finding $U_{k_{-q}}$ using this naive approach compared to using (4.12) where all nodes are assumed to have 4 sensor signals. The dashed line at the bottom indicates the computational complexity

[1] It should be noted that when a single channel is considered for removal then this becomes a scalar inversion, and (4.12) reduces to the formulation presented in [124].

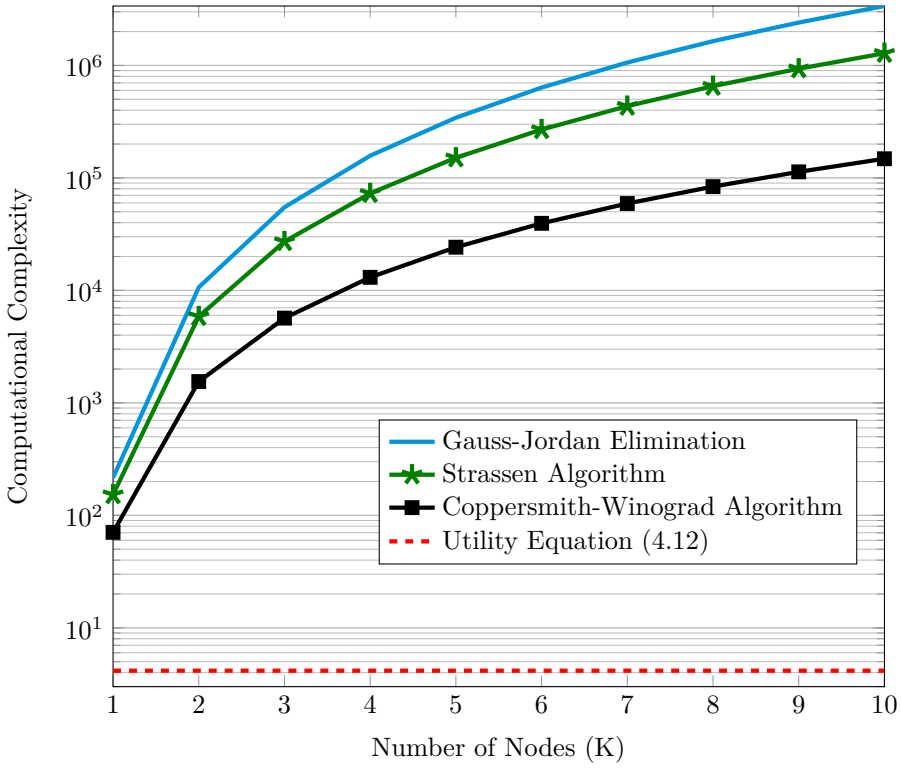


Figure 4.1: Computational complexity for finding $U_{k_{-q}}$ using the naive approach compared to (4.12).

of calculating the utility using (4.12) which is constant due to the fact of only relying on the inverse of the same, in this case, 4×4-dimensional matrix. The total computational complexity is given for different matrix inversion algorithms [133] which have an increasingly large number of calculations for an increasing number of nodes. In [124] a less generalized expression compared to (4.12) was derived which finds the utility of a single sensor at computational cost of $O(M)$. The utility computed using (4.12) could instead be found in an iterative fashion based on this single sensor utility where in each iteration, a single sensor of node q is removed and this process is repeated until all of the sensors from node q are removed. The utility of node q with respect to the estimation problem of node k in this iterative approach is then given as

$$U_{k_{-q}} = \sum_{m=1}^{M_q} U_{k_{-q_m}} \tag{4.15}$$

where m is the mth sensor index of node q defined similarly as in (2.2). In the iterative approach, a new optimal fall-back estimator must be calculated after each sensor removal which was found to have a computational complexity of $O((M-m)^2)$ [124]. The overall computational complexity therefore would be $\sum_{m=1}^{M_q} O((M-m)^2)$. However using (4.12) the utility can be found with a single matrix inversion with a maximum computational complexity of $O(M_q^3)$. Since often $M >> M_q$, using this iterative approach is usually more computationally intensive than using (4.12).

4.2.2 Utility for node addition

For node addition, the utility of node q with respect to the estimation problem of node k is defined as the decrease in LMMSE when all sensors of node q are added to the estimation problem of node k. The current estimator without node q, $\widehat{\mathbf{W}}_{k_{-q}}$, is assumed to be known which also implies that the current sensor signal correlation matrix, $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, and its inverse, $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$, are known. The cross-correlation matrix between the sensor signals and the desired signal components, $\mathbf{R}_{\mathbf{y}\mathbf{d}_k}$, is partitioned as

$$\mathbf{R}_{\mathbf{y}\mathbf{d}_k} = \begin{bmatrix} \mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} \\ \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} \end{bmatrix} \tag{4.16}$$

where $\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_q\mathbf{d}_k^H\}$ is not included in the current estimation problem and $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_{-q}\mathbf{d}_k^H\}$ represents the current cross-correlation matrix.

The utility is defined identically as in the case of node removal, which is repeated here for convenience, as

$$\begin{aligned} U_{k_{-q}} &= J_{k_{-q}}(\widehat{\mathbf{W}}_{k_{-q}}) - J_k(\widehat{\mathbf{W}}_k) \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\mathbf{d}_k}\widehat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\widehat{\mathbf{W}}_{k_{-q}}\} \end{aligned} \tag{4.17}$$

which relies on the new estimator with the addition of node q, $\widehat{\mathbf{W}}_k$. Unlike the node removal case, the statistics to estimate the contribution of node q are unknown at node k because no information is sent when the node is not connected to the network. To circumvent this limitation it is presupposed that node q periodically sends part of its observations to node k from which the required statistics can be measured, but these are not included in the estimation at node k hence only an $(M-M_q)\times(M-M_q)$-dimensional inverse is taken. Notice that this makes the calculation of the utility when a node is added to the estimation substantially different than for the node removal case.

With the above mentioned strategy of periodically sending node q's statistics to node k, the sensor signal correlation matrix is partitioned the same as given in (4.7). However, a computationally efficient manner in calculating the utility without having to take the full inverse of $\mathbf{R}_{\mathbf{yy}}$ to compute $\widehat{\mathbf{W}}_k$ should be found.

For the sake of an easy exposition, two intermediate variables are defined as

$$\mathbf{\Gamma} = \mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} \mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_q} \tag{4.18}$$

$$\mathbf{\Sigma} = \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{y}_q} \mathbf{\Gamma} \tag{4.19}$$

which incorporate the statistics of the current connected nodes and those of node q. In Appendix C.3 it is shown that the utility, when node q is added to the estimation problem of node k using these two intermediate variables along with the previously defined notation, is given as

$$U_{k_{-q}} = \mathrm{Tr}\{(\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k})^H \mathbf{\Sigma}^{-1} (\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k})\} \tag{4.20}$$

where

$$\mathbf{U}_{k_{-q}} \triangleq (\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k})^H \mathbf{\Sigma}^{-1} (\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}) \tag{4.21}$$

which is again a generalization of the work presented in [124]. Since $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ is already known from the current estimation, the computational complexity of finding $\mathbf{\Gamma}$ is $O((M - M_q)^2 M_q)$ and $\mathbf{\Sigma}$ relies on the inverse of an $M_q \times M_q$-dimensional matrix. Therefore the overall computational complexity of finding the utility will be $O((M - M_q)^2 M_q + M_q^3)$. If this utility is compared to a naive approach of calculating the utility, i.e., including node q's signals in the current estimation and taking a worst case scenario $O(M^3)$ to find the change in the LMMSE, then (4.20) offers a substantial decrease in computational complexity for large M.

4.2.3 Definition of a common network-wide utility

The utility calculated with (4.12) and (4.20) gives the difference in the LMMSE when a node is removed from or added to the network. However, the utilities calculated at an individual node are biased to that node's desired signals, i.e., the utility calculated for the signals of node k, $U_{k_{-k}}$, may differ significantly for another node, $U_{q_{-k}}$. This conflict in the utilities stems from the fact that each node estimates its own node-specific desired signals. This makes it difficult to quantify the network-wide utility of the sensor signals of a node, i.e., a single utility measure that incorporates the estimation problem of every node.

One approach could be to use the sum, $\sum_k U_{k-q}$, to define the network-wide impact of node q's signals. First of all, this would require the computation of K^2 utility values to evaluate the network-wide utility of each node. Secondly, and more importantly, this measure is heavily biased towards estimation problems at nodes with a large signal power, as the LMMSE directly depends on the signal power of the desired signal. The utility values corresponding to these estimation problems will dominate the summation.

Therefore, a scaling is proposed of the utilities by means of (4.4) in which the utilities are now in terms of a virtual node s. This modifies the utilities of the nodes as if they were estimating the dry source signals ($\mathbf{s}$ in (2.4)) which effectively removes the bias toward a single node's desired signals resulting in a common utility-reference. The intuition behind this approach is that a reliable estimate of the dry source signal(s) also allows each node to compute a reliable estimate of their locally observed source signal(s), i.e., if the sensor signals of a node have a large utility with respect to this dry source estimation problem, they will be important for every node-specific estimation problem too. This is because each node actually estimates node-specific scaled versions of the dry source signals.

For ease of exposition, it is assumed that each node will scale its utilities as if it were estimating the unobservable dry source signals in $\mathbf{s}$. Note that the estimation of $\mathbf{s}$ is not possible in practice, as the cross-correlation matrix in (2.26) cannot be computed from the local sensor signals at node k for the case where $\mathbf{d}_k = \mathbf{s}$. However, the remarkable aspect of this is that information about the dry source signals is not needed to calculate the utility of a node with respect to it. The desired dry source signals in terms of a $Q \times Q$-dimensional identity matrix is given as $\overline{\mathbf{A}}_s = \mathbf{I}_s$ for a virtual node s so that

$$\begin{aligned} \mathbf{d}_s &= \overline{\mathbf{A}}_s \mathbf{s} \\ &= \mathbf{s}. \end{aligned} \tag{4.22}$$

The LMMSE cost at this virtual node is given as

$$\begin{aligned} J_s(\widehat{\mathbf{W}}_s) &= \mathcal{E}\left\{ \left|\left| \mathbf{d}_s - \widehat{\mathbf{W}}_s^H \mathbf{y} \right|\right|_2^2 \right\} \\ &= \mathrm{Tr}\{\mathbf{R}_{\mathbf{d}_s\mathbf{d}_s} - \mathbf{R}_{\mathbf{y}\mathbf{d}_s}^H \widehat{\mathbf{W}}_s\} \end{aligned} \tag{4.23}$$

where $\mathbf{R}_{\mathbf{y}\mathbf{d}_s}^H = \mathcal{E}\{\mathbf{y}\mathbf{d}_s^H\}$. For the sake of an easy exposition, but w.l.o.g, it is assumed that the dry source signals have also been power normalized to unity which, relying on the assumption that the signals are statistically independent, gives $\mathbf{R}_{\mathbf{d}_s\mathbf{d}_s} = \mathcal{E}\{\mathbf{s}\mathbf{s}^H\} = \mathbf{I}_\mathbf{s}$.

The utility is defined similarly to (4.5) where the utility of signals of node k with respect to estimation problem of node s is given by

$$U_{s_{-k}} = J_{s_{-k}}(\widehat{\mathbf{W}}_{s_{-k}}) - J_s(\widehat{\mathbf{W}}_s). \tag{4.24}$$

Using the relationship between steering matrices and the LMMSE cost matrices, as given in (4.4), results in

$$\begin{aligned} U_{s_{-k}} &= \mathrm{Tr}\{\widehat{\mathbf{J}}_{s_{-k}}\} - \mathrm{Tr}\{\widehat{\mathbf{J}}_s\} \\ &= \mathrm{Tr}\{\left(\overline{\mathbf{A}}_{ks}\right)^{-H}\left(\widehat{\mathbf{J}}_{k_{-k}} - \widehat{\mathbf{J}}_k\right)\left(\overline{\mathbf{A}}_{ks}\right)^{-1}\} \\ &= \mathrm{Tr}\{\left(\widehat{\mathbf{J}}_{k_{-k}} - \widehat{\mathbf{J}}_k\right)\left(\overline{\mathbf{A}}_{ks}\right)^{-1}\left(\overline{\mathbf{A}}_{ks}\right)^{-H}\} \end{aligned} \tag{4.25}$$

where $\overline{\mathbf{A}}_{ks} = \mathbf{I}_s\overline{\mathbf{A}}_k^H$. Using this and the fact that $\mathbf{R}_{\mathbf{d}_k\mathbf{d}_k} = \overline{\mathbf{A}}_k\overline{\mathbf{A}}_k^H$, it is then shown that

$$U_{s_{-k}} = \mathrm{Tr}\{\mathbf{U}_{k_{-k}}\mathbf{R}_{\mathbf{d}_k\mathbf{d}_k}^{-1}\} \tag{4.26}$$

which can then be applied to both (4.14), and (4.21).

This definition of a common network-wide utility allows each node to track the network-wide utility of its own sensor signals. Furthermore, it provides a common reference such that the utilities computed at the different nodes can be easily compared with each other, or with a common threshold (see also Section 4.2.4).

4.2.4 Greedy centralized node selection

To allow maintaining a minimal network-wide estimation performance, nodes are only removed if this removal does not result in a network-wide LMMSE increase[2] of more that ε, where ε is a user-defined threshold, which can be adapted depending on the current LMMSE. Similarly, a node is only added if this addition can guarantee a minimal increase in the network-wide LMMSE, i.e., larger than ε. It is noted that, since a common network-wide utility is now used, each node can use the same value for ε. As an example in terms of a constraint, if the number of nodes in the network were to be limited so that 50% were removed, this ε could be adjusted until this constraint was met.

To facilitate a distributed node selection algorithm, each node should be able to independently decide whether it should add itself or remove itself from the network which requires that each node calculates the network-wide utility of its own signals. To this end, in the case of node addition, instead of node k

[2]This is the LMMSE with respect to the (virtual) estimation of the dry source signals.

broadcasting its signal periodically to the other nodes to measure its utility, it is assumed that it does not transmit its signal in order to conserve energy but that it can still receive the $M - M_k$ broadcast signals from the other connected nodes in the network. In this case node k uses its own $\mathbf{y}_k$ sensor signals as well as the $M - M_k$ signals from the other nodes to compute its utility, U_{k-k}, for node addition by means of (4.12) instead of (4.20) since it is able to calculate the full optimal estimator $\widehat{\mathbf{W}}_k$.

In the node removal case, the selection process picks the node with the lowest utility below ε and removes this from the network. In the case there are multiple nodes for which the utility value is smaller than ε, a greedy choice is made, i.e., the node with the smallest utility is removed. The network continues this selection process until there are no more nodes whose utility fall below ε. Likewise if a node is not connected to the network and its utility is greater than the threshold value, it is added to the network where again a greedy choice is made in case multiple nodes exceed the threshold.

Since the selection process is greedy, no claims on optimality are made, however, it is argued that because of the prohibitive computational complexity of an exhaustive search, the utility based approach offers a safe bound on the impact on the network-wide performance (in terms of the LMMSE of the dry source signal estimation) while being computationally efficient. The greedy centralized node selection is summarized in Table 4.1.

Remark 1. *Notice also that once a node is removed or added from or to the network the inverse sensor signal correlation matrix (4.8) must be recalculated which effects the utilities of the nodes, therefore the future utility values cannot be predicted.*

4.3 Distributed computation of utility bounds

In Section 4.2 it was assumed that each node has access to all M signals of $\mathbf{y}$ to compute the optimal $\widehat{\mathbf{W}}_k$ for estimating its node-specific desired signal components. A distributed scenario is now considered where each node employs the DANSE algorithm as described in Section 2.4.

In the distributed scenario, nodes only have access to their own sensor signals and linearly compressed sensor signals from the other nodes, e.g., node k only has access to its own M_k sensor signals, $\mathbf{y}_k$, and the $Q(K-1)$ broadcast signals from the other nodes, $\mathbf{z}^i_{-k}$. Using these signals, each node computes its own utility locally, $U_{s_{-k}}$, to determine if it should add itself to or remove itself from the network. For node addition, it is assumed that while a node is not

Table 4.1: Greedy centralized node selection

1. Each node $\forall k \in \mathcal{K}$ collects observations of the signals $\mathbf{y}_q,\ \forall q \in \mathcal{K}$.
2. Each node calculates its utility $U_{s_{-k}}$ based on (4.26)
   - **If** $U_{s_{-k}} \leq \varepsilon$ **and** node is connected to network
     - Remove node from network
     - Update $\mathbf{R}_{\mathbf{y}_{-k}\mathbf{y}_{-k}}$ and $\widehat{\mathbf{W}}_{-k}, \quad \forall q \in \mathcal{K} \setminus \{k\}$
       Note: if there are multiple nodes for which the utility is below the threshold the node with the smallest utility is removed.
   - **ElseIf** $U_{s_{-k}} > \varepsilon$ **and** node is not connected to network
     - Add node to network
     - Update $\mathbf{R}_{\mathbf{yy}}$ and $\widehat{\mathbf{W}}_k, \quad \forall q \in \mathcal{K} \setminus \{k\}$
       Note: if there are multiple nodes for which the utility exceeds the threshold the node with the largest utility is added.
3. return to 1.

transmitting it can still receive the other $\mathbf{z}^i_{-k}$ broadcast signals. For the sake of easy exposition, it is assumed that node k computes the utility $U_{k_{-k}}$ with respect to its own estimation problem, rather than $U_{s_{-k}}$ with respect to the dry source signals (see Section 4.2.3). However, everything in this section can easily be extended to also compute $U_{s_{-k}}$ by using the appropriate transformation given in (4.26).

The utility for node deletion given in (4.12) relies on the availability of $\mathbf{S}$, a sub-matrix of $\mathbf{R}^{-1}_{\mathbf{yy}}$, which is never available in the distributed case. Therefore, (4.12) cannot be used, and the original definition of the utility in (4.5) is needed. This relies on the ability for a node to calculate its optimal fall-back estimator, e.g., the fall-back estimator of node k when removing itself from the network is $\widehat{\mathbf{W}}_{k_{-k}}$. However, in the distributed scenario, the optimal estimator is found by iteratively passing information from one node to the next until the system reconverges. When node k is removed from the network this not only changes the partial estimator $\mathbf{W}^i_{kk}$ but also $\mathbf{G}^i_{k_{-k}}$ which both rely on the statistics of the other nodes. Therefore if node k were removed from the network, the fall-back estimator at node k, $\widetilde{\mathbf{W}}^i_{k_{-k}}$ is initially suboptimal and only becomes optimal once all of the nodes in the network have converged again. To avoid the

explicit computation of the K different fall-back estimators for each possible node removal, $\widetilde{U}_{k_{-k}}$ is defined that is based on (4.12) where $\mathbf{R}_{\mathbf{yy}}^{-1}$ is now replaced with $(\mathbf{R}^{i}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k})^{-1}$. In the case of node removal this $\widetilde{U}_{k_{-k}}$ will be shown to be an upper bound on the increase in the LMMSE, i.e., the actual utility $U_{k_{-k}}$.

Likewise for node addition it will be shown that with a similar argument removed nodes are able to calculate $\widetilde{U}_{k_{-k}}$ which produces a utility lower bound on the actual utility $U_{k_{-k}}$, i.e., if the node adds itself to the existing network the actual decrease in LMMSE after addition will be greater than that given by the utility. Therefore even though the exact utility cannot be computed, bounds for the change in the LMMSE can be found for node removal and addition, which will facilitate node selection.

4.3.1 Node removal : utility upper bound

Assuming the DANSE algorithm has converged[3] the following quantity for node k with respect to its node-specific estimation problem is defined

$$\widetilde{U}^{i}_{k_{-k}} = J_{k_{-k}}(\widetilde{\mathbf{W}}^{i+1}_{k_{-k}}) - J_k(\widehat{\mathbf{W}}_k) \tag{4.27}$$

where $\widetilde{\mathbf{W}}^{i+1}_{k_{-k}}$ is equal to (2.41) with $\mathbf{W}^{i+1}_{kk}$ removed.

It is noted that (4.27) does not represent the true utility of node k since, in principle, once the node is removed the other nodes must update their local estimator parameters until the DANSE algorithm has reconverged to the re-optimized fall-back estimator, i.e., $\widetilde{\mathbf{W}}^{\infty}_{k_{-k}} = \widehat{\mathbf{W}}_{k_{-k}}$ where $\widehat{\mathbf{W}}_{k_{-k}}$ represents the optimal estimator without the signals of node k.

Due to the convergence of DANSE, the cost function then decreases until

$$J_{k_{-k}}(\widetilde{\mathbf{W}}^{\infty}_{k_{-k}}) = J_{k_{-k}}(\widehat{\mathbf{W}}_{k_{-k}}). \tag{4.28}$$

Since $\widehat{\mathbf{W}}_{k_{-k}}$ minimizes $J_{k_{-k}}$, the $\widetilde{U}^{i}_{k_{-k}}$ using the suboptimal estimator $\widetilde{\mathbf{W}}^{i+1}_{k_{-k}}$ is an upper bound of the increase in LMMSE, i.e.,

$$J_{k_{-k}}(\widetilde{\mathbf{W}}^{\infty}_{k_{-k}}) - J_k(\widehat{\mathbf{W}}_k) \leq J_{k_{-k}}(\widetilde{\mathbf{W}}^{i+1}_{k_{-k}}) - J_k(\widehat{\mathbf{W}}_k)$$

$$\widetilde{U}^{\infty}_{k_{-k}} \leq \widetilde{U}^{i}_{k_{-k}}. \tag{4.29}$$

The utility upper bound, $\widetilde{U}^{i}_{k_{-k}}$, can be efficiently computed by means of (4.12) based on $\mathbf{R}^{i}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$

$$\widetilde{U}^{i}_{k_{-k}} = \mathrm{Tr}\{\widehat{\mathbf{W}}^{H}_{kk}(\mathbf{S}^{i}_{k})^{-1}\widehat{\mathbf{W}}_{kk}\} \tag{4.30}$$

[3]For simulations presented in Section 4.4 convergence is reached after each node updates its local parameters 5 times, i.e., $5 \times K$ DANSE iterations.

where $\mathbf{S}_k^i$ is defined similarly to $\mathbf{S}$ in (4.8) and is the part of $(\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k})^{-1}$ pertaining to the signals of node k only. A corresponding upper bound for the network-wide utility $\widetilde{U}_{s_{-k}}$ corresponding to the dry source signals can also be computed similarly to (4.26).

With this, node k can decide to remove itself from the network knowing the maximum impact it will have in terms of increase in LMMSE. It should also be noted that $\widetilde{\mathbf{W}}_{k_{-k}}^{i+1}$ is not explicitly computed when calculating the utility upper bound, and only exists once the node has been removed from the network.

4.3.2 Node addition : utility lower bound

As in the centralized scenario, the addition of a node is substantially different from node removal due to the fact that node k does not broadcast $\mathbf{z}_k$ when not included in the network. In the distributed scenario when a node is not connected to the network it is assumed that it is still able to receive signals, possibly awaking periodically to judge its current importance to the network estimation.

In the sequel, it is assumed that node k still performs an estimation of its own desired signals using $\mathbf{z}^i_{-k}$ and its own $\mathbf{y}_k$ signals. In this case, it is shown that the node is able to determine a utility lower bound with the same formula that is used for node removal (4.30) without having to broadcast its $\mathbf{z}_k^i$ signal.

Assuming the DANSE algorithm has converged with node k not broadcasting its signals, the following utility for node k with respect to the estimation problem of node k is defined,

$$\widetilde{U}^i_{k_{-k}} = J_{k_{-k}}(\widehat{\mathbf{W}}_{k_{-k}}) - J_k(\widetilde{\mathbf{W}}_k^{i+1}) \tag{4.31}$$

where again $\widehat{\mathbf{W}}_{k_{-k}}$ represents the optimal estimator without the signals of node k and $\widetilde{\mathbf{W}}_k^{i+1}$ is given in (2.41). Notice that the cost function with the signals of node k removed, $J_{k_{-k}}(\widehat{\mathbf{W}}_{k_{-k}})$, is currently minimized as all of the nodes in the network have performed their estimation without the signals of node k using the DANSE algorithm.

Assuming that node k would include itself in the network, then due to the convergence of the DANSE algorithm, the cost decreases to

$$J_k(\widehat{\mathbf{W}}_k) = J_k(\widetilde{\mathbf{W}}_k^{\infty}) \tag{4.32}$$

where again the estimators have converged to their optimal values. The $\widetilde{U}^i_{k_{-k}}$ using the suboptimal estimator therefore is a lower bound for the LMMSE

decrease when a node is added to the network, i.e.,

$$J_{k_{-k}}(\widehat{\mathbf{W}}_{k_{-k}}) - J_k(\widetilde{\mathbf{W}}_k^{\infty}) \geq J_{k_{-k}}(\widehat{\mathbf{W}}_{k_{-k}}) - J_k(\widetilde{\mathbf{W}}_k^{i+1})$$

$$\widetilde{U}_{k_{-k}}^{\infty} \geq \widetilde{U}_{k_{-k}}^{i}. \tag{4.33}$$

Therefore if node k was to add itself to the network, i.e., begin broadcasting its $\mathbf{z}_k$, its LMMSE will decrease by at least the utility

$$\widetilde{U}_{k_{-k}}^{i} = \mathrm{Tr}\{\widetilde{\mathbf{W}}_{kk}^{iH}(\mathbf{S}_k^i)^{-1}\widetilde{\mathbf{W}}_{kk}^{i}\}\ . \tag{4.34}$$

It should be noted that since node k uses its $\mathbf{y}_k$ in its current estimation it does not need to rely on (4.20) to calculate its utility. However, the calculation of the utility from either equation would be equivalent.

4.3.3 Greedy distributed node selection

In the distributed scenario the same method for adding and removing nodes can be used as in the centralized case (Table 4.1). However instead of calculating the exact utilities, only upper and lower bounds can be computed. The greedy selection procedure in Section 4.2.4 is therefore modified to take the utility bounds into consideration. During estimation nodes will calculate their utility bounds based on (4.30) and (4.34) and scale them to the common dry source reference using (4.26). A distributed version of the node selection algorithm is given in Table 4.2. Since the nodes compute upper and lower bounds, it is known that it is safe to remove or add nodes, i.e., without risking an LMMSE increase or decrease that is larger than ε.

4.4 Simulations

4.4.1 Batch mode

In this section, the greedy utility based node selection process is demonstrated in batch mode, which means that all iterations of DANSE were performed on data obtained from the entire length of the signals. For the simulations, the MSE cost function at each node J_k was effectively replaced by the corresponding finite-length equivalent least-squares (LS) cost function given as

$$J_k(\mathbf{W}_k) = \sum_{t=0}^{T-1} \left\| \mathbf{d}_k[t] - \mathbf{W}_k^H \mathbf{y}_k[t] \right\|^2 \tag{4.35}$$

Table 4.2: Greedy distributed node selection

1. Each node $\forall k \in \mathcal{K}$ collects its own $\mathbf{y}_k$ signals and broadcast signals, $\mathbf{z}^i_{-k}$ from other nodes connected in the network.
2. Each node calculates its utility $\widetilde{U}_{s_{-k}}$ based on (4.30) and scaled to the dry source as in (4.26)
   - **If** $\widetilde{U}_{s_{-k}} \leq \varepsilon$ **and** node is connected to network
     - Remove node from network
     - Update $\mathbf{R}^{i+1}_{\tilde{\mathbf{y}}_{k_{-k}}\tilde{\mathbf{y}}_{k_{-k}}}$ and $\widetilde{\mathbf{W}}^{i+1}_{k_{-k}}$, $\forall q \in \mathcal{K} \setminus \{k\}$
       Note: if there are multiple nodes for which the utility is below the threshold the node with the smallest utility is removed.
   - **ElseIf** $\widetilde{U}_{s_{-k}} > \varepsilon$ **and** node is not connected to network
     - Add node to network
     - Update $\mathbf{R}^{i+1}_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$ and $\widetilde{\mathbf{W}}^{i+1}_{k}$, $\forall q \in \mathcal{K} \setminus \{k\}$
       Note: if there are multiple nodes for which the utility exceeds the threshold the node with the largest utility is added.
3. return to 1.

where the signal length is given as T and where $\mathbf{d}_k[t]$ and $\mathbf{y}_k[t]$ denote the observation of the signals $\mathbf{d}_k$ and $\mathbf{y}_k$ at sample time t, respectively. Although batch mode is not a practical implementation, a batch mode simulation gives a reasonable view on the performance limits of the algorithm. In Section 4.4.2 an adaptive implementation with moving sources is presented where, instead of processing the entire length of the signal, shorter blocks are processed which change the utilities throughout the simulation.

The greedy node selection process used a utility bound based on (4.30), (4.34) and (4.26). The dimension of the desired signal space, Q, was varied depending on the simulation, where each source signal consisted of $T = 10000$ samples which were independent and identically distributed (i.i.d.) uniformly on the interval [-0.5 0.5]. The coefficients of $\mathbf{A}_k$ were generated by a uniform process on the unit interval. A single spatially located white noise source was added where the signal was generated with a similar process as the source signals and scaled with a random number in (0,1] for each node. Spatially uncorrelated white noise, representative of process noise, that was half the average power of

the desired signals, was added to each sensor.

As each node in the network estimates a different signal, the network-wide performance at one particular *visualization* node is analyzed. It was assumed[4] that $\overline{\mathbf{A}}_k = \mathbf{I}$ in (2.6) at this node, i.e., the node estimates the dry source signals. This allowed for a comparison of the increase and decrease in the LMMSE when compared to the network-wide utility bounds which was also referenced to the dry source signal estimation problem (see Section 4.2.3). The algorithm was constrained such that this visualization node was never removed from the network.

In order to compare the difference in the dry source MSE during the centralized and distributed node selection process, the order of the node selection was forced to be the same, i.e., the order was stored during the centralized selection process and used in lieu of a node selection process in the distributed scenario. However, it should be noted that without this forced selection, the same order of nodes in general is not chosen in the centralized and distributed node selection process (see Section 4.4.2). Since the following figures are used for illustrative purposes to compare the centralized and distributed node selection process, the bound for removal was first set to $\varepsilon = \infty$ so that all of the nodes would be considered for removal.

In the distributed scenario, the DANSE parameters, $\mathbf{W}_{kk}^i$ and $\mathbf{G}_{k_{-k}}^i$, $\forall k \in \mathcal{K}$, were updated in a round-robin fashion for $5 \times K$ iterations, i.e., 5 updates per node, which was deemed sufficient for network convergence, i.e., the difference in cost between iterations was below machine precision. After DANSE had reached a steady state the utility bounds pertaining to the dry source signal, $\widetilde{U}_{s_{-k}}$, were calculated. The DANSE parameters were allowed to reconverge before the selection process resumed.

The centralized and distributed selection procedures for removal were compared as shown in Figure 4.2. There were $K = 20$ nodes in the system each with $M_k = 5$ sensors ($M = 100$), and $Q = 3$. Figure 4.2a is the centralized selection process, where the full matrix $\mathbf{R}_{\mathbf{yy}}^{-1}$ can be used in (4.12). The dotted lines indicate how much the dry source MSE increased after the removal of a node, i.e., $J_s(\widehat{\mathbf{W}}_s) + U_{s_{-k}}$.

Figure 4.2b shows the DANSE algorithm that performs the same removal process where (4.30) was used and scaled by (4.26). The dashed lines indicate the maximum increase a node will have on the dry source LS cost after removal, i.e., $J_s(\widehat{\mathbf{W}}_s) + \widetilde{U}_{s_{-k}}^i$. Each increment on the horizontal axis in the distributed scenario corresponds to a full iteration cycle of DANSE, i.e., every node has

[4]Node that this only for illustrative purposes since $\overline{\mathbf{A}}_k = \mathbf{I}$ usually does not occur in practice.

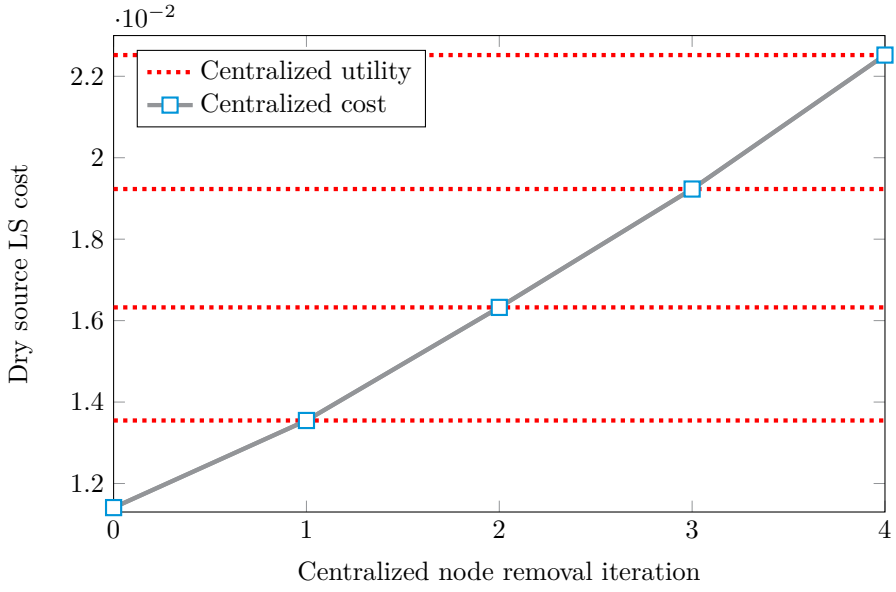
updated its node-specific parameters ($\mathbf{W}_{kk}^i$ and $\mathbf{G}_{k_{-k}}^i$) once. It was observed in [53] that convergence of the node-specific parameters occurs after each node has updated them 5 times. Therefore each DANSE cycle in Figure 4.2 corresponds to each node in the network updating their node-specific parameters once ($i = 20$ DANSE iterations) with a node being removed after each node has updated its node-specific parameters 5 times ($i = 20 \times 5$ DANSE iterations).

During the node selection process the utility upper bound and centralized utility were added to the current dry source LS cost in order to observe the increase in LS cost after node removal. This is indicated by the dashed and dotted lines respectively. Notice the large decrease in the LS cost in the first few iterations of the DANSE algorithm compared to the subsequent reconvergence iteration when a node is removed from the network. This is due to the random initialization of the parameters at the beginning of the DANSE algorithm. Once the DANSE algorithm has converged and a node is removed, the suboptimal estimators lie close to the new optimal estimators as shown in the relatively small decrease in MSE after the first set of iterations.

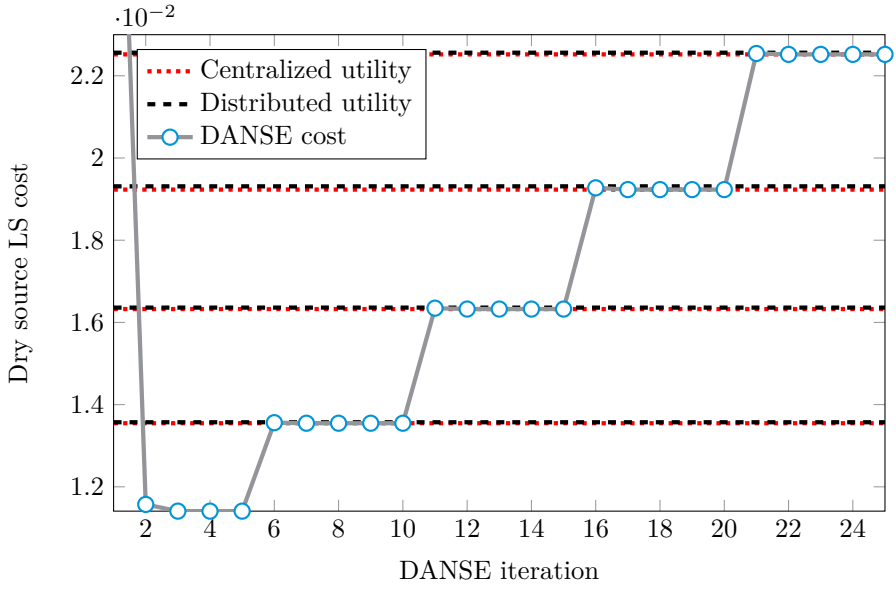
Figure 4.3 shows a magnified view of the selection process between DANSE iterations 15-21 of Figure 4.2. The utility upper bound lies above the utility that would have been found if the optimal estimator would have been used. Again a relatively small decrease in the LS cost is observed when compared to the original initialization of the DANSE algorithm.

The next simulation contained $K = 5$ nodes in the system each with $M_k = 20$ sensors ($M = 100$) and $Q = 3$, i.e., the network loses many more sensor signals per node removal than in the previously simulated network. This normally has a larger impact on the utility bound, as the fall-back estimators must reconverge from a larger difference. Figure 4.4 shows the increase in the LS cost with the utility bound during the selection process. Notice that the utility bounds are less tight than in the previous network ($K = 20$, $M = 100$). This is because there are many more degrees of freedom in the DANSE parameters at the other nodes. As the utility bound does not take the future DANSE updates of these parameters at other nodes into account, there is a larger gap between the centralized utility and the utility upper bound computed in the distributed case.

Due to the node removal case calculating a utility upper bound in the distributed scenario, there are times when the node selection fails to remove nodes that would have been removed had the centralized utility been used. Using the previous simulated network, the threshold value was set to $\varepsilon = 0.03$ so that this type of failure in the node selection process could be observed. In Figure 4.5, the threshold value of ε, the centralized utility, the distributed utility bound and the LS cost of the DANSE algorithm are shown. In this scenario, the



(a)



(b)

Figure 4.2: Comparison of the LS cost for a centralized (a) and distributed (b) node removal process.

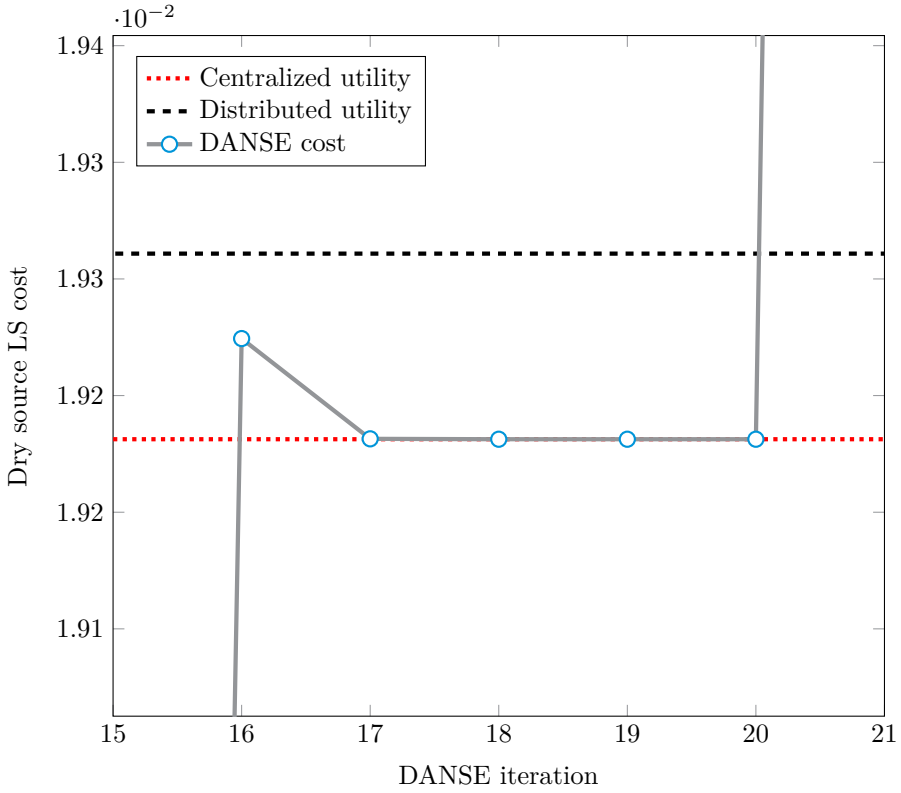


Figure 4.3: Utility upper bound in a distributed scenario compared to centralized (optimal) utility.

centralized utility was below ε, however the distributed utility bound fell above ε and so the node was not considered for removal. While this does not have a negative effect on the estimation, i.e., the LS cost stays at a lower value, it prolongs the usage of a node that would have otherwise had its transmission capabilities turned-off possibly shortening the lifetime of the network.

For node addition a network was constructed with $K = 20$ nodes each with $M_k = 5$ sensors ($M = 100$) and $Q = 3$. At the beginning of the selection process, a single node broadcast its $\mathbf{z}_k^i$ signal and other nodes used this signal along with their local signals to determine their utility. The utility threshold was set to $\varepsilon = 0$ so that all of the available nodes would eventually add themselves to the network. In the centralized case, Figure 4.6a, the utility was found using (4.26) and was then subtracted from the current LS cost to find the new LS

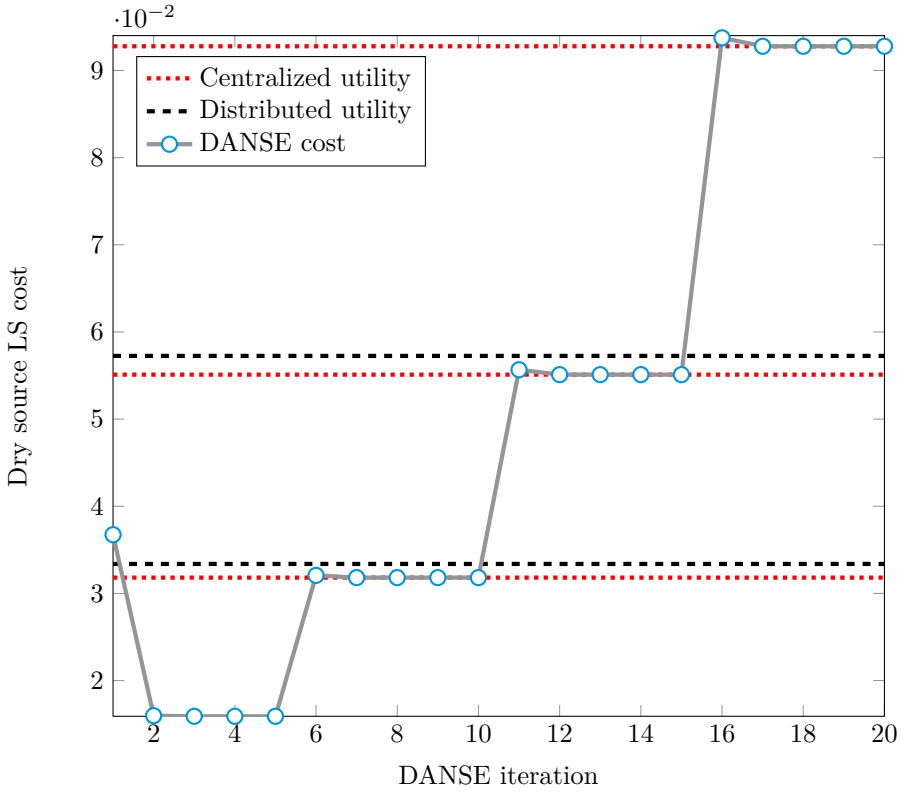


Figure 4.4: Utility upper bound in a distributed scenario compared to centralized (optimal) utility with $M_k \geq K(Q-1)$.

cost after node addition, i.e., $J_{s_{-k}}(\widehat{\mathbf{W}}_s) - U_{s_{-k}}$ and is represented by the dotted line. In the distributed case, the utility was calculated by means of (4.30) and subtracted from the current LS cost, i.e., $J_{s_{-k}}(\widehat{\mathbf{W}}_s) - \widetilde{U}^i_{s_{-k}}$ as shown in Figure 4.6b represented by the dashed lines. It should be noted that since the utility was subtracted from the current LS cost, the true LS cost after addition was lower than that calculated LS cost which used the suboptimal estimator as shown in (4.33). In Figure 4.6 the utility bound is shown to provide at least a minimal decrease in the system, i.e., after convergence in the distributed scenario the LS cost is lower than that given by the utility bound.

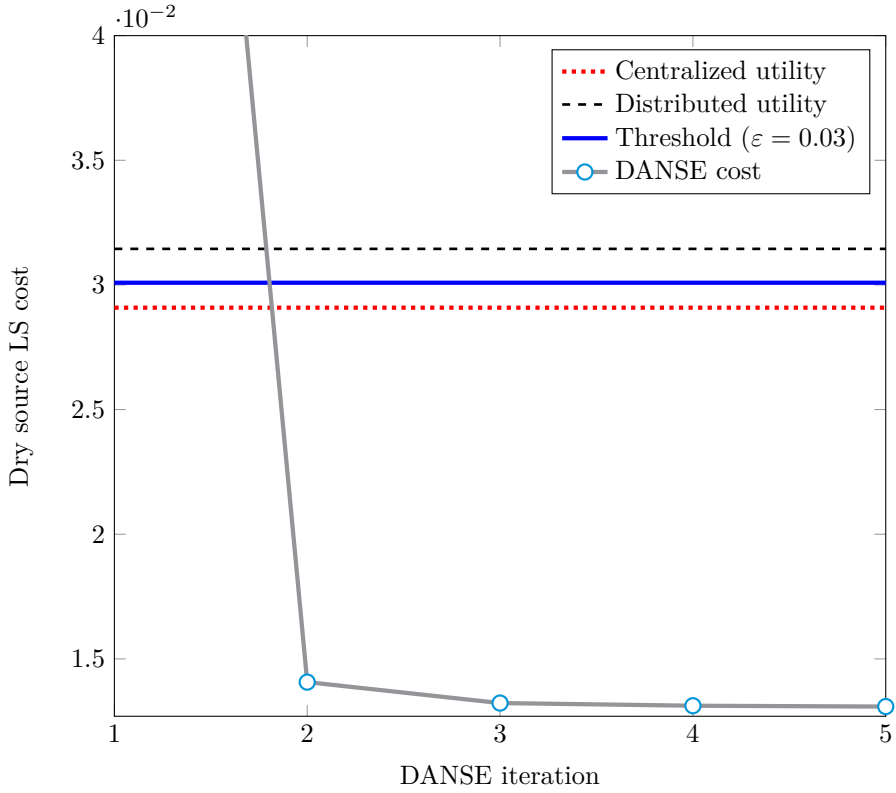
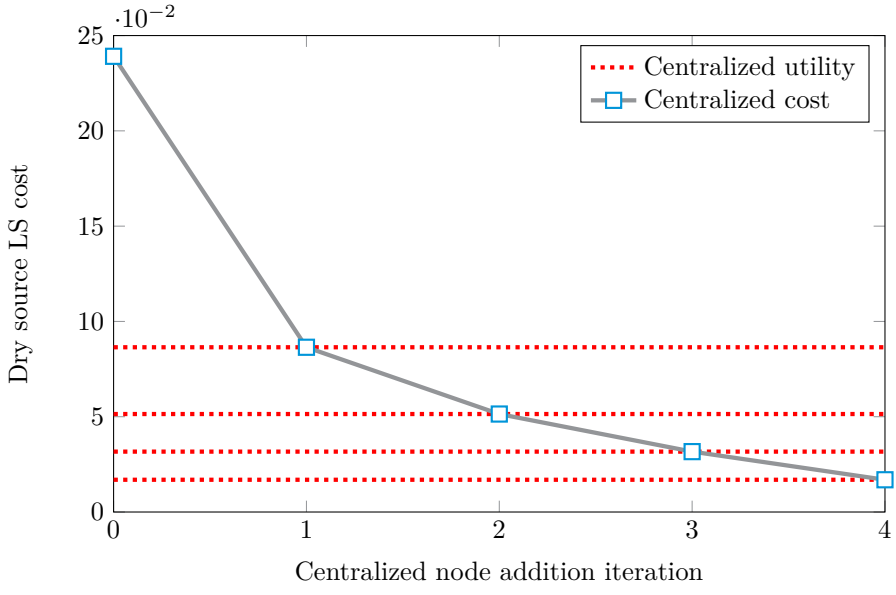


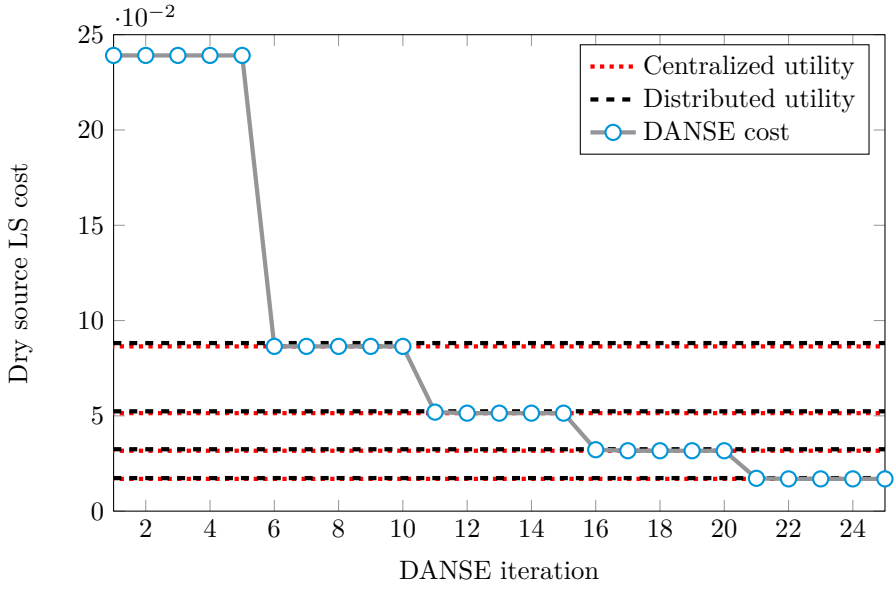
Figure 4.5: Distributed scenario where a node is kept in the network that would have been removed if the centralized utility was used.

4.4.2 Adaptive implementation

For the adaptive implementation a simulated environment, depicted in Figure 4.7, was considered where there were $Q = 2$ moving source signals. The source signals (■, ■), were generated in the same manner as in Section 4.4.1 and followed paths indicated by the colored lines respective of the color of the source. The source signals moved at a speed of 0.3 m/s and stopped for 3 seconds at each corner. After the source signals reached the end of the path, they followed the same route to their starting point. This movement repeated until the end of the simulation. There were also 5 spatially located white noise sources (◆), which were generated with a similar process as the source signals. Spatially uncorrelated white noise that was 5% of the average power of the noise sources was added to each sensor. There were $K = 30$ nodes each with $M_k = 5$ sensors



(a)



(b)

Figure 4.6: Comparison of centralized (a) and distributed (b) node addition process.

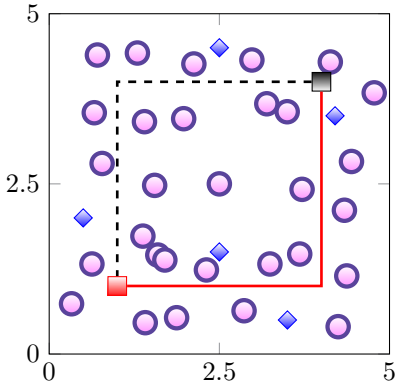


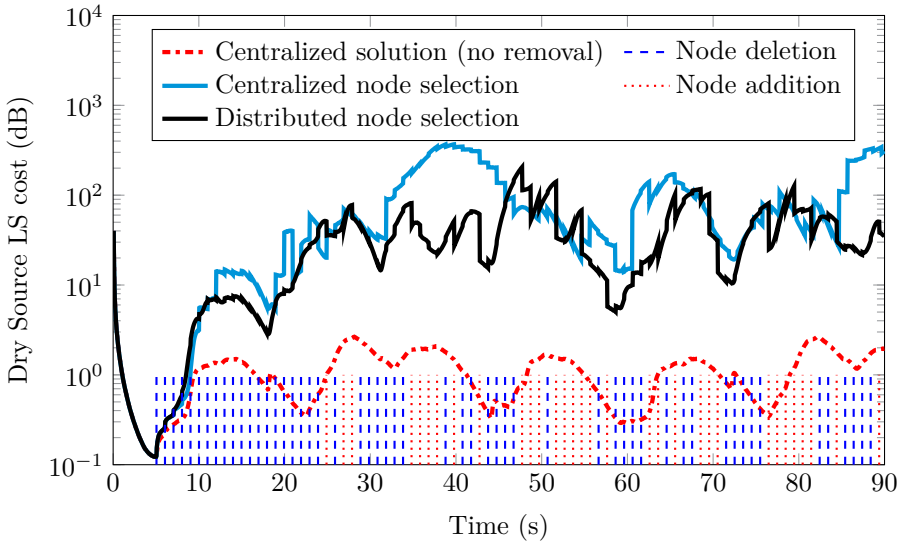
Figure 4.7: Depiction of simulated scenario. The network contained 30 nodes each with 5 sensors (●). There were two sources (■, ■) that follow paths indicated by the colored lines respective of the color of the source. There were also five spatially located white noise sources present (◆).

(●) so that the total number of sensor observations were $M = 150$.

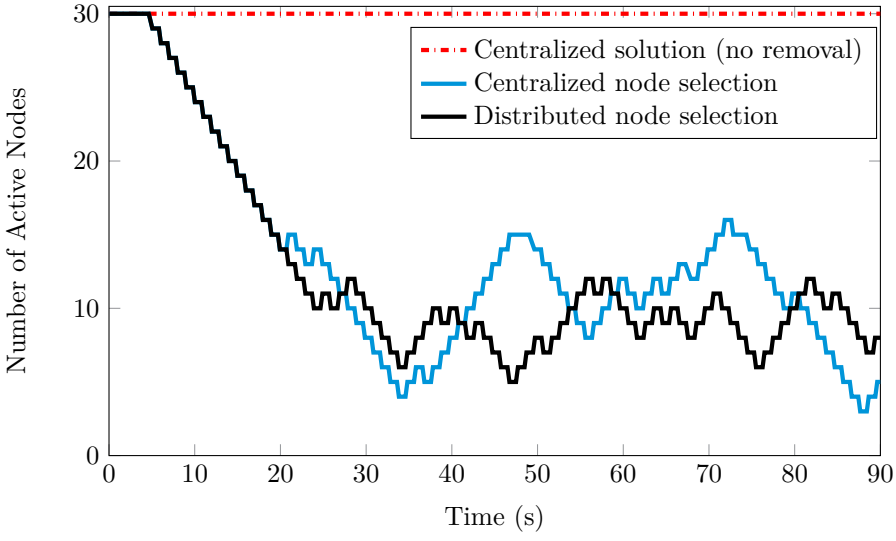
The individual sensor measurements originating from the source signals and noise were attenuated and summed at each sensor. The attenuation factor was given as $\frac{1}{d}$ where d denotes the distance from the source to the sensor. It was assumed that the source signals and noise statistics were estimated at each node as given in Section 2.2.2 and where the correlation matrices were updated with a forgetting factor of $\gamma = 0.97$ and the sensors observed their signals at a sampling frequency of $f_s = 8$ kHz.

The source signals were stationary for the first 5 seconds of the simulation in order to populate the necessary signal statistics and all nodes were considered active during this time. After this initialization, the node selection process was started in order to remove and add nodes depending on their utility bounds when compared to the predefined threshold of $\varepsilon = 5$. After the addition or removal of a node from the system a full DANSE cycle occurred, i.e., all nodes updated once, before the selection algorithm began again. This was done in order to allow the DANSE algorithm to reconverge after the addition or removal process.

Figure 4.8 compares the LS cost when no nodes were removed by the centralized and distributed node selection process, as well as the number of active nodes in the system. The vertical dashed lines indicate node addition and node deletion in the distributed scenario. The selection of nodes in the centralized and distributed scenario do not follow the same order due to the use of an upper

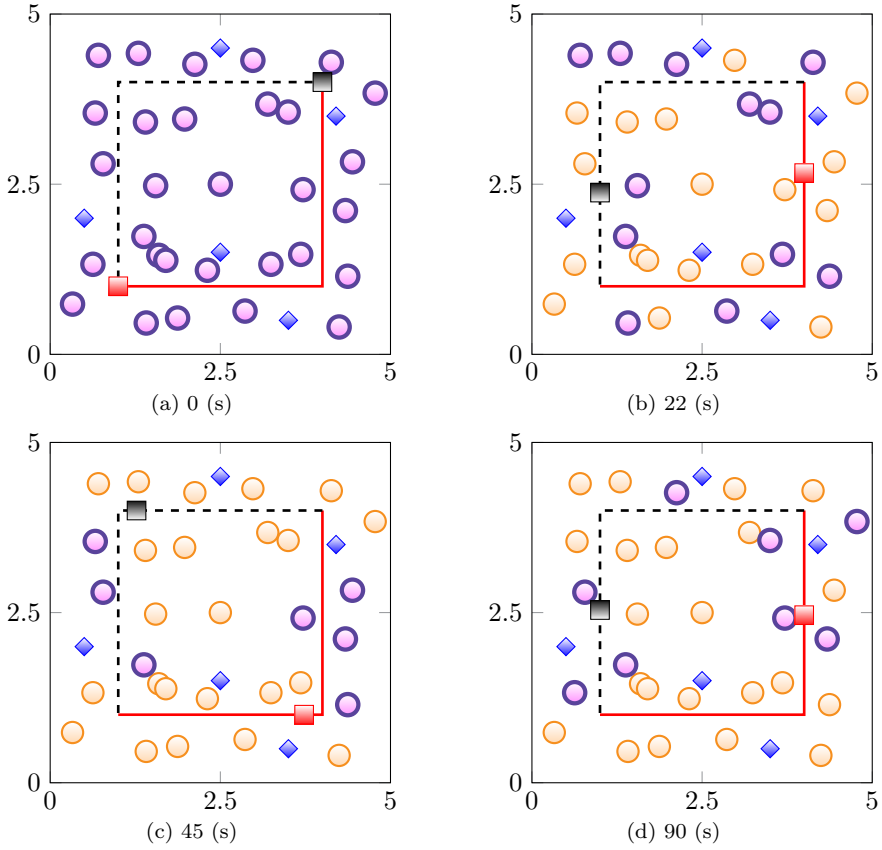


(a)



(b)

Figure 4.8: Comparison of the LS cost (a) and the total number of active nodes (b) in the centralized scenario with and without node selection and the distributed scenario with node selection.



(a) 0 (s) (b) 22 (s)

(c) 45 (s) (d) 90 (s)

Figure 4.9: Node selection during various times in the estimation process using distributed utility bounds. The active nodes are represented as (●) and the nodes that are only receiving signals from the other nodes and not transmitting their $\mathbf{z}_k^i$ signals are represented as (○).

and lower bound and due to the limited tracking capabilities of DANSE, which may generate errors in the utility bounds.

However, the utility bound was able to limit the effect on the LS cost similarly to that in the centralized scenario. There were even times that the LS cost and number of active nodes in the distributed scenario is a lower value than that of the centralized scenario which is possible due to the fact that greedy node selection is often suboptimal.

The scenario is shown in Figure 4.9 during various times (0s, 22s, 45s, 90s)

of the node selection process. The active nodes are represented as ● and the nodes that are only receiving signals from the other nodes and not transmitting their $\mathbf{z}_k^i$ signals are represented as ○. At 0 seconds no nodes were removed from the network and after 22 seconds there was already a large reduction in the amount of active nodes in the network. It should be noted that this was dependent on ε and could be adjusted to fit the desired scenario.

4.5 Conclusion

In this chapter, the utility was introduced as a means to facilitate node selection in a distributed wireless sensor network that performs node-specific signal estimation. This was accomplished by using the convergence and optimality properties of the DANSE algorithm in unison with an MSE threshold and a greedy selection process. While the distributed utility bounds were shown to be suboptimal they were successfully used to limit the LMMSE increase or decrease during node selection. The centralized and distributed node selection were compared to one another and it was shown that the utility bounds offer an efficient way to perform node selection while still allowing to control the MSE performance. Simulation results using the distributed node selection process often have a similar performance to the centralized node selection process and show that there are significant power savings in the network while only slightly effecting the MSE.

Part II

Mixed-topology distributed signal estimation and supporting techniques

Chapter 5

Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks

5.1 Introduction

The DANSE algorithm was introduced in a fully connected WSN in Chapter 2 and has been extended to tree topologies [54] (T-DANSE), which will be briefly reviewed in Chapter 6. However, in these existing versions of the DANSE algorithm, the adherence to a single topological structure can have a negative impact on the resources and performance of the algorithm. In the fully connected case, each node must communicate with every other node in the WSN, which over large distances can quickly deplete the energy resources of the nodes. Conversely, while a tree topology can rely on a nearest neighbor communication strategy to reduce energy consumption, the branching of the tree will lead to an increased number of hops in between nodes which effects the input-output delay as well as the convergence speed of the DANSE algorithm.

In this chapter, a method for implementing the DANSE algorithm in WSNs with

more than a single topological structure is proposed, which will allow for a trade-off in performance which lies in between that of a single fully connected network and a single tree topology. The topology of these WSNs will be constructed based on one of two approaches, namely either a top-down approach where the WSN is composed of heterogeneous nodes, or a bottom-up approach where the nodes are not necessarily heterogeneous.

With the multitude of different devices that can form a WSN, it is natural to assume that some of these devices come equipped with larger energy budgets and processing capabilities. In the **top-down** approach the nodes with the largest energy budgets are designated as cluster heads and the remaining nodes of the WSN form clusters around these nodes. Such heterogeneous WSNs offer many benefits compared to homogeneous WSNs and have been explored in order to extend the lifetime of WSNs [79, 80]. They have been used for such applications as wireless body area networks [134, 135] and their benefits have been outlined for use in wireless multimedia sensor networks in [9]. The heterogeneous nodes can be thought of as a partitioning of the WSN into two layers, where the top layer consists of the cluster heads, and the bottom layer consists of the member nodes of the clusters. This type of partitioning can then easily be abstracted to a WSN that is primarily hierarchical in nature.

WSNs with a hierarchical structure for distributed parameter estimation have been studied where the actual estimation is performed at a fusion center [136] whereas in this chapter, estimation is performed in a distributed fashion. These methods have also been extended to distributed hierarchical WSNs, as the one envisaged in this chapter, where the parameter estimation takes place using diffusion or consensus based algorithms [137, 138]. In [139, 140] the WSN is tasked with estimating a desired signal where each of the received signals at the individual clusters are correlated with one another. However, all of the information is again transmitted to a fusion center which is the only location in the WSN that performs an estimation of the desired signals.

Contrary to the top-down approach, in the **bottom-up** approach, the nodes are no longer necessarily heterogeneous in nature and the WSN is first formed in an ad-hoc topology. An attractive attribute of an ad-hoc WSN is a lack of fixed infrastructure which allows for greater flexibility in node deployment. This ad-hoc WSN can initially be formed with a variety of constraints in mind such as: energy conservation [74,75]; communication bandwidth [76,77]; and security [78]. However, due to the lack of fixed infrastructure, it has been shown that ad-hoc WSNs can suffer from scalability issues[1] due to the fact that nodes must monitor

[1]Since the DANSE algorithm relies on in-network signal fusion, the nodes do not perform explicit routing tasks through the network, and hence these scalability issues do not really apply here. Nevertheless, it will be shown that pruning an ad-hoc network into a set of smaller substructures is still advantageous within the context of DANSE.

and retain network wide routing tables which can require a significant portion of the available network resources [81, 82]. This problem becomes even more prevalent if the nodes are mobile requiring constant reconfiguration of these routing tables to pass information throughout the network [141].

Fortunately many of the routing and scalability challenges associated with ad-hoc WSNs can be mitigated by partitioning the ad-hoc configuration into a set of smaller, simpler topologies or substructures [142–145]. In order to determine these substructures the WSN can rely on a so-called *topology control*, which looks to extract these substructures from the original ad-hoc configuration [103,143,146–148]. In the bottom-up approach the WSN is partitioned into a set of smaller substructures, namely non-overlapping cliques, which are connected with each other in a tree topology.

The proposed top-down and bottom-up approaches will be shown to be conceptually equivalent, in that the same building blocks constitute both the envisaged heterogeneous and mixed-topology WSNs. By way of pre-defined fusion rules, the functionality of the DANSE algorithm is then extended to such mixed-topology WSNs. The DANSE algorithm in a mixed-topology will be shown to converge to the same solution as if the nodes had access to all of the sensor signal observations in the WSN. Simulations will show that the mixed-topology WSN consumes a significantly lower amount of energy on a per-node basis when compared to the fully connected case. Furthermore, since the total number of links is larger than in an exact (clique-free) tree, the average number of input signals per-node increases, yielding more degrees of freedom at some nodes to perform an update. On average, this results in an overall faster convergence time than T-DANSE, i.e., the DANSE algorithm operating in a clique-free tree, with only a slight increase in per-node energy usage.

This chapter is organized as follows. Section 5.2 introduces the per-node signal fusion rules using the top-down approach where the WSN consists of heterogeneous nodes. Section 5.3 introduces the per-node signal fusion rules using the bottom-up approach where the WSN first has an ad-hoc topology and is partitioned into smaller substructures. Simulations are performed in Section 5.4, which highlight the improved convergence properties in the envisaged WSNs and also compare the energy usage to that of DANSE in fully connected and clique-free tree topologies. Finally, conclusions are given in Section 5.5.

5.2 Heterogeneous WSNs

A heterogeneous WSN is now assumed which contains N nodes with larger energy budgets. The N nodes, hence have larger broadcast capabilities making

them more suitable for long-range communication [149], which are designated as cluster heads (CH). In Section 5.3 the heterogeneity of the nodes is relaxed so that each node is assumed to have similar broadcast capabilities. Clusters are now formed around these CHs with the remaining $K - N$ nodes, which are designated as member nodes (MN) of the N clusters. The CHs and MNs are placed into subsets $\mathcal{K}_{\text{CH}}$ and $\mathcal{K}_{\text{MN}}$, respectively. The set of nodes in cluster n is defined as $\mathcal{K}_n$, where each cluster is composed of a subset of MNs, $\mathcal{K}_{n,\text{MN}} \subseteq \mathcal{K}_{\text{MN}}$, and a single CH, $\mathcal{K}_{n,\text{CH}} \subseteq \mathcal{K}_{\text{CH}}$, so that $\mathcal{K}_{n,\text{MN}} \cup \mathcal{K}_{n,\text{CH}} = \mathcal{K}_k$.

Since the CHs have larger broadcast capabilities, it is assumed that they are able to communicate to the other CHs in the WSN in a single-hop fashion, i.e., that they form a fully connected network. However, they may also be formed in a tree topology where the methods presented in Section 5.3 can be applied to the CHs. The MNs of the clusters can be formed in either a fully connected fashion or in a tree topology. This type of WSN can then be thought of as being hierarchical in nature, i.e., the WSN contains two layers, an upper layer with CHs, where information is exchanged across clusters (inter-cluster communication) and where these CHs communicate to the MNs in the lower layer (intra-cluster communication). A hierarchical representation of the envisaged heterogeneous WSN is given in Figure 5.1.

For the ease of exposition, this description is further simplified in the sequel by assuming a star topology within each cluster instead of a multi-level tree, i.e., a two-level tree where the CH acts as the root node. This means that each MN can directly communicate with its corresponding CH. However, it is re-iterated, that this is merely for the sake of an easy exposition in order to avoid the elaborate description of the DANSE algorithm in a multi-level tree topology (T-DANSE), which is explained in Chapter 6.

The CHs are not only responsible for broadcasting information from their local cluster to the other CHs, but also broadcasting the information from the other CHs to their local cluster. The CHs could simply act as relays, where they would broadcast all information from the MNs in their cluster to the other CHs and vice versa, i.e., from the CHs to the MNs. However, this broadcast strategy quickly becomes infeasible as the number of CHs and the number of MNs becomes large. Therefore, a way to allow the CHs to broadcast information is proposed such that the amount of information is independent of not only the number of CHs but also the number of MNs in any cluster. In the following description of the DANSE algorithm in a heterogeneous WSN it is shown that the MNs of the clusters transmit a set of fused sensor signal observations that are defined identically to that of the DANSE algorithm in a fully connected WSN. The CHs need only broadcast two sets of fused sensor signal observations, one for inter-cluster communication that is broadcast to all other CHs, and one for intra-cluster communication that is broadcast to the MNs of the cluster. It

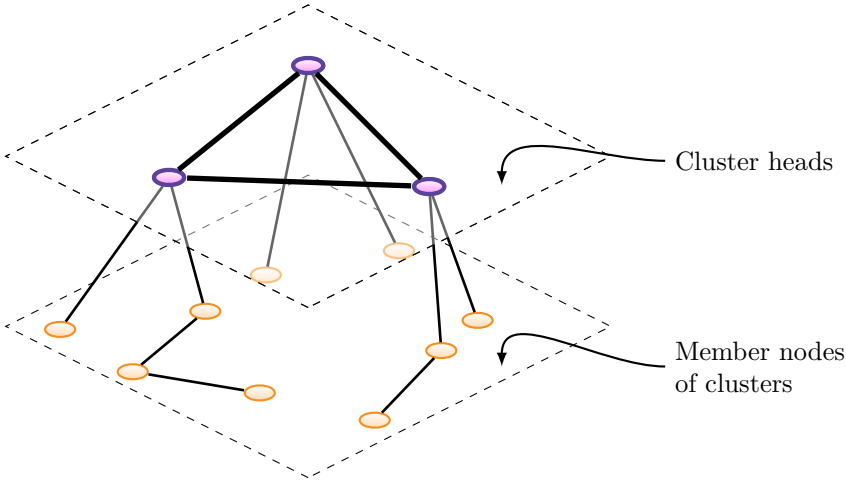


Figure 5.1: A hierarchical representation of the envisaged heterogeneous WSN. The top layer consists of a set of cluster heads (CHs) that form a fully connected network with one another. The member nodes (MNs) of each cluster are formed in, e.g., a tree topology where the CHs act as root nodes for their individual clusters.

is noted that the dimension of these signals will be independent of the number of CHs or MNs in any cluster.

5.2.1 Intra- and inter-cluster communication

The fused signals are now denoted as $\mathbf{z}_{kq}^i$ where the subscripts k and q denote that the fused sensor signal observations are transmitted from node k to node q. The intra-cluster signals for a MN k and a CH q are given as

$$\mathbf{z}_{kq}^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k, \ k \in \mathcal{K}_{n,\text{MN}} \ , \forall q \in \mathcal{K}_{n,\text{CH}} \ , \forall n \ . \tag{5.1}$$

This fusion rule is identical as the one in fully connected DANSE (see (2.31)). It is noted that if the cluster is formed in a fully connected fashion instead of a star topology, the same fusion rule applies, although each MN would then also receive intra-cluster signal observations from other MNs in the cluster.

The CHs now fuse the intra-cluster signals with their own sensor signals in order to generate the inter-cluster signals. For node k which is the CH of cluster n,

the inter-cluster signal is given as

$$\boldsymbol{\zeta}_k^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \sum_{q\in\mathcal{K}_{n,\mathrm{MN}}} \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^i \ , k \in \mathcal{K}_{n,\mathrm{CH}} \ , \forall n \tag{5.2}$$

which are broadcast to all other CHs.

In order to disseminate the information from the other CHs to their own cluster, the CHs now form fused sensor signals, $\mathbf{z}_k^i$, that contain all of the received intra-cluster, $\mathbf{z}_{kq}^i$, and inter-cluster, $\boldsymbol{\zeta}_q^i$, signals as well as their own sensor signals as

$$\mathbf{z}_k^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \sum_{q\in\mathcal{K}_{n,\mathrm{MN}}} \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^i + \sum_{q\in\mathcal{K}_{\mathrm{CH}}\setminus\{k\}} \mathbf{G}_{kq}^{iH}\boldsymbol{\zeta}_q^i \ , k \in \mathcal{K}_{n,\mathrm{CH}} \ , \forall n \tag{5.3}$$

of which observations are broadcast to all MNs in the cluster.

However, in broadcasting observations of (5.3) to the MNs, a feedback path develops as $\mathbf{z}_k^i$ contains all of the MN intra-cluster signals. This type of feedback has been shown to prevent the nodes from converging to their LMMSE solution [54]. The MNs can remove this feedback component by subtracting their own intra-cluster signal $\mathbf{z}_{qk}^i$ from the CH's intra-cluster broadcast signal, $\mathbf{z}_k^i$, to generate the signal

$$\mathbf{z}_{kq}^i = \mathbf{z}_k^i - \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^i, \ \forall k \in \mathcal{K}_{n,\mathrm{CH}} \ , q \in \mathcal{K}_{n,\mathrm{MN}} \ , \forall n \ . \tag{5.4}$$

Note that this requires the CHs to periodically transmit their $\mathbf{G}_{kq}^i, \ q \in \mathcal{K}_{n,\mathrm{MN}}$, to all of their MNs. Although this does increase the transmission bandwidth it is negligible compared to the continuous transmission of the observations.

5.2.2 Data-driven signal flow

Since there are dependencies between the signals defined in (5.1)-(5.4), the nodes have to transmit their signals observations in a specific order. However, this can be done in a purely data-driven fashion, without network-wide coordination.

This data-driven signal flow first starts in the individual MNs of each cluster since (5.1) does not depend on the signals from any other node. When a new set of sensor signal observations becomes available, a MN fuses its observations based on (5.1) and sends these fused sensor signal observations to its CH. The CH generates observations of the inter-cluster signals based on (5.2) and broadcasts this to the other CHs. Once all of the observations of the inter-cluster signals from the other CHs have been received, a CH generates its intra-cluster signal observations based on (5.3), and broadcasts these to its MNs. A MN

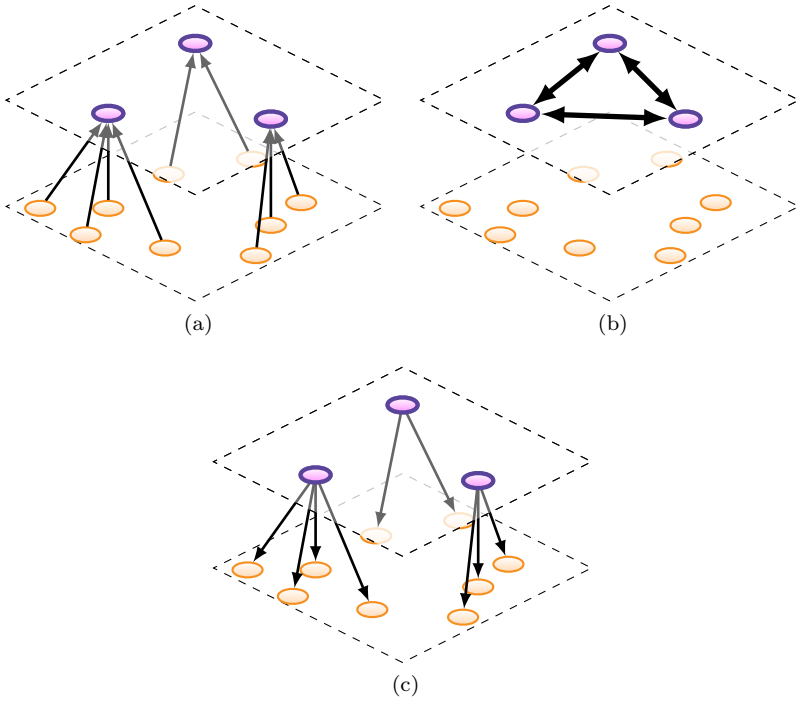


Figure 5.2: Data-driven flow in a heterogeneous WSN utilizing the DANSE algorithm : The MNs of each cluster send observations of their intra-cluster signals (5.1) to their respective CH (a). The CHs broadcast observations of their inter-cluster signals (5.2) to the other CHs in the WSN (b). The CHs broadcast observations of their intra-cluster signals (5.3) to their respective MNs where the MNs use (5.4) to cancel out their feedback portion (c).

then uses the intra-cluster signal observations from its CH (5.3) as well as the $\mathbf{G}_{kq}^{i}$ transformation matrix to cancel out the feedback portion based on (5.4). This data-driven signal flow is depicted in Figure 5.2.

5.2.3 Node-specific local parameter updating

Notice that, because of the star (or more generally tree) topology in the clusters and the fully connected topology amongst the CHs in Figure 5.1, there is a unique shortest path, P, between any two nodes in the network. This unique shortest path essentially defines an ordered set of nodes that is represented as

$P_{p_1 \to p_t} = (p_1, p_2, \ldots, p_{t-1}, p_t)$ for the path from node p_1 to node p_t and where the inverse path is given as $P_{p_1 \leftarrow p_t}$ (in the case of star-topology clusters, these paths have a maximum length of 3 hops, and pass through a maximum 4 nodes). A matrix, $\mathbf{G}_{p_1 \leftarrow p_t}$, is now defined that is a product of $\mathbf{G}$'s following the same path $P_{p_1 \leftarrow p_t}$, i.e.,

$$\mathbf{G}_{p_1 \leftarrow p_t} = \mathbf{G}_{p_{t-1} p_t} \mathbf{G}_{p_{t-2} p_{t-1}} \ldots \mathbf{G}_{p_2 p_1} \ . \tag{5.5}$$

The parameterization given in (2.41) can then be similarly defined using (5.5) as

$$\mathbf{W}_k^i = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{k \leftarrow 1}^i \\ \vdots \\ \mathbf{W}_{KK}^i \mathbf{G}_{k \leftarrow K}^i \end{bmatrix} . \tag{5.6}$$

Again this parameterization defines a solution space for all $\mathbf{W}_k^i, \ k \in \mathcal{K}$, simultaneously and is similar to the one defined in the fully connected case (2.41).

Theorem 1. *If (2.6) holds, then the LMMSE estimators $\widehat{\mathbf{W}}_k$ given in (2.26) are in the solution space defined by the parameterization (5.6).*

Proof. By setting the $\mathbf{G}_{kq}^i$ matrices equal to $\mathbf{G}_{kq}^i = \overline{\mathbf{A}}_{kq} = \left(\overline{\mathbf{A}}\right)_q^{-H} \overline{\mathbf{A}}_k^H$ and because of (5.5), which results in a product of matrices of the form $\overline{\mathbf{A}}_{\kappa l} \overline{\mathbf{A}}_{k\kappa} = \overline{\mathbf{A}}_{kl}$ for any κ, l, and k, then automatically it is shown that $\mathbf{G}_{k \leftarrow l}^i = \overline{\mathbf{A}}_{kl}$. By also setting $\mathbf{W}_{qq}^i = \widehat{\mathbf{W}}_{qq}$, $\forall q \in \mathcal{K}$ and using (2.29), then it is shown that the solution space defined by the parameterization (5.6) contains the optimal solutions given in (2.26). □

The matrix $\mathbf{G}_{k-k}^i$ now denotes the stacked version of all $\mathbf{G}_{kq}$ matrices for which $q \in \mathcal{N}_k$. The stacked vector of the sensor signals of node k is now denoted as $\widetilde{\mathbf{y}}_k^i$, similarly to (2.33), $\mathbf{y}_k$, and the fused sensor signals of all of the neighbors of node k, $\mathbf{z}_{\mathcal{N}_k}^i$, i.e.,

$$\widetilde{\mathbf{y}}_k^i = [\mathbf{y}_k^T \ \mathbf{z}_{\mathcal{N}_k}^{iT}]^T \ . \tag{5.7}$$

Table 5.1 shows how the node-specific local parameter updating is performed during the operation of the DANSE algorithm in the envisaged heterogeneous WSN where the updating order is defined by an arbitrary path through the network, i.e., the updating node should be a neighbor of the node that updated last (this will be elaborated on in Subsection 5.2.4). To this end, an arbitrary path P is defined through the network that starts at node k and ends at node $q \in \mathcal{N}_k$, visiting each node at least once. The nodes will then perform updates in the order defined by a periodic repetition of P. The signal flow described

Table 5.1: The DANSE algorithm in a heterogeneous WSN

1. Initialize $0 \rightarrow i$, $p_1 \rightarrow k$.
   Initialize $\mathbf{W}_{qq}^0$ and $\mathbf{G}_{q_{-q}}^0$ randomly, $\forall q \in \mathcal{K}$.
2. Node k updates its node-specific local parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_{k_{-k}}$, by minimizing its LMMSE criterion based on the B most recent observations of its own local sensor signals and the fused sensor signals received from neighboring nodes as defined in Section 5.2.1:

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \mathbf{G}_{k_{-k}}^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_{k_{-k}}}{\arg\min}\, \mathcal{E}\left\{ \left\| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}_{kk}^H & \mathbf{G}_{k_{-k}}^H \end{bmatrix} \widetilde{\mathbf{y}}_k^i \right\|_2^2 \right\} \quad (5.8)$$

   .
   The other nodes do not update their node-specific local parameters :

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i, \mathbf{G}_{q_{-q}}^{i+1} = \mathbf{G}_{q_{-q}}^i \; . \quad (5.9)$$

3. $i{+}1 \rightarrow i$.
4. $p_t \rightarrow k$ with $t = (i \bmod |P|) + 1$.
5. return to 2 .

in Subsection 5.2.2 is not included in Table 5.1 as the node updating and the signal flow are performed independently, albeit in parallel.

The update order, or path, can be described in the following fashion for the WSN in Figure 5.2 (assuming the individual clusters have a star topology); a CH, k, of cluster n updates its local parameters by way of (5.8). The next update occurs in a MN of the CH after which the next update again occurs at the CH. This update procedure is then repeated between every MN in cluster n and the CH. Once every node in cluster k has updated its local parameters once, this process can begin in a neighboring cluster. This process repeats for every cluster until finally arriving back at the CH of cluster n.

The signal estimate, $\widetilde{\mathbf{d}}_k^i$, at node k can be found at any point in the iterative process as (compared with (2.40))

$$\widetilde{\mathbf{d}}_k^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \mathbf{G}_{k_{-k}}^{iH}\mathbf{z}_{\mathcal{N}_k}^i . \quad (5.10)$$

5.2.4 Convergence and optimality

Using the heterogeneous hierarchical representation given in Figure 5.1 it is now shown that even though the WSN consists of a fully connected topology and several tree topologies (here assumed to be stars), the DANSE algorithm is still able to converge to the same solution, at each node, as if each node had access to all of the sensor signal observations in the entire WSN. Even though convergence and optimality of the DANSE algorithm in a fully connected topology and in a tree have been proven in [53] and [54], respectively, these proofs can not straightforwardly be generalized to the case of mixed-topologies.

For one specific (but naive) updating procedure, the convergence of the DANSE algorithm in a heterogeneous WSN can be straightforwardly established by applying the convergence proofs of both DANSE in a fully connected network and the T-DANSE algorithm, which is explained as follows. A single cluster n is first considered where the fused sensor signals from the other clusters are temporarily viewed as fixed additional (virtual) sensor signals for the CH. The cluster n can then be viewed as a subnetwork with a tree topology, in which the T-DANSE algorithm can be run until convergence. From [54], it is known that the signal estimate $\widetilde{\mathbf{d}}_k$, for every node k in the cluster, will then converge to the same LMMSE solution as if it had access to all sensor signal observations within the cluster. This convergence of the cluster can be thought of as a single DANSE update in the fully connected network that is composed of the CHs, i.e., it is as if the CH of cluster n has solved (2.35), where the MNs of cluster n are viewed as virtual sensors of the CH. After this first cluster has converged, the same process happens in the next cluster where again the T-DANSE algorithm is run until convergence and this repeated for all N clusters in the network. Since the CHs converge to the same solution as if they had access to all of the sensor signal observations within their cluster, the DANSE algorithm that operates among the CHs will then converge to the same solution as if each CHs had access to all of the sensor signal observations of each cluster.

However this updating scheme is not practical as one cluster (in theory) has to perform $i \to \infty$ number of iterations until convergence is reached before the next cluster can start the T-DANSE algorithm. The aim is therefore to show that the same solution can be found by following an arbitrary updating path, P, through the heterogeneous WSN that does not need to wait for each cluster in the WSN to converge. This will allow for faster convergence of all of the nodes in the WSN.

Theorem 2. *Consider a WSN with a heterogeneous topology as described above. Let P represent a path through the WSN that traverses through every node and begins at node k and ends with node $q \in \mathcal{N}_k$. If (2.6) holds then the DANSE*

algorithm as described in Table 5.1 converges for every node k and for any initialization of its parameters to the centralized estimator given in (2.26).

Proof. See Appendix D. □

The above results also hold for general tree topologies within each cluster. It is noted that updating the nodes according to a path through the network is a sufficient condition for convergence, but not a necessary condition, i.e., convergence is also often (but not always) observed if nodes update in a random order. Nevertheless, updating orders that follow a path through the network generally yield a faster convergence (which was also observed in [54]).

5.3 Mixed-topology ad-hoc WSNs

The case where the nodes in the WSN are not necessarily heterogeneous is described where they are first formed in an ad-hoc configuration. Since the nodes are not considered to be heterogeneous, the hierarchical structure presented in Section 5.2 is not as readily apparent in the envisaged network. Instead, the WSN is partitioned into smaller substructures, namely non-overlapping fully connected networks, or cliques, which are either directly connected to one another or which are connected by cycle-free subnetworks with nodes in the WSN. It is shown that the DANSE algorithm can be implemented in such mixed-topology networks by employing similar fusion rules to those defined in Section 5.2.

5.3.1 Distributed clique formation

A partitioning of the WSN into smaller substructures can be accomplished by performing so-called *topology control* on the WSN, where substructures are formed by adding or removing links in order to form a given set of topological structures [143, 146–148]. The particular partitioning of the WSN into non-overlapping cliques, as envisaged in this chapter, can be done in a distributed fashion, by means of a modified Bron-Kerbosch algorithm [150], where nodes need only exchange local connection information with neighboring nodes. It is noted that there are several other methods to find cliques in a given graph but this is not the main focus of this work. During clique formation, it is assumed that the WSN is partitioned into N non-overlapping cliques, where the set of nodes that form clique n is given as $\mathcal{C}_n$. A constraint is also applied to ensure there exists a unique path between adjacent cliques which can be accomplished

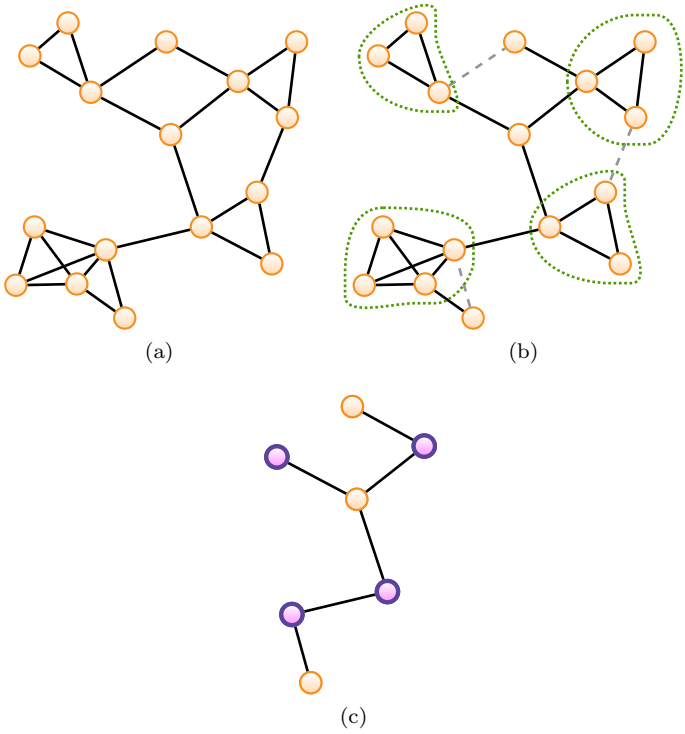


Figure 5.3: A depiction of the partitioning of an ad-hoc configuration (a) into a set of non-overlapping cliques (b). This is then further abstracted to a tree topology (c).

by pruning some links in the network. To be more precise, if the cliques would be replaced by a single node (see Fig. 5.3c), there exists a unique path between any two nodes in the resulting (virtual) network.

Once the network has been partitioned into a set of non-overlapping cliques and the unique paths are defined between the cliques, the network may be further abstracted to that of a single tree topology where the cliques are again viewed as single nodes. The process of topology control in which the WSN is first formed into non-overlapping cliques and then given as an abstraction to a tree topology is depicted in Figure 5.3.

As shown in Figure 5.3b, there are three distinct types of nodes : non-clique nodes or nodes that do not belong to any clique; clique nodes whose neighboring set only comprises nodes in the same clique, i.e., if node k belongs to clique

n then $\mathcal{N}_k = \mathcal{C}_n \backslash \{k\}$; and clique nodes whose neighboring set does not only comprise nodes in the same clique. In the following section it is described how each type of node fuses its sensor signal observations with the fused signal observations from its neighbors.

5.3.2 Intra- and inter-clique communication

Since the partitioned WSN can be abstracted to that of a tree topology, the fused sensor signals for *non-clique* nodes are defined in exactly the same manner as in the T-DANSE algorithm (cfr. Chapter 6). It is assumed that the non-clique nodes use a point-to-point communication protocol where a different signal is transmitted to each of its neighbors, i.e., a non-clique node k transmits

$$\mathbf{z}_{kq}^i = \mathbf{W}_{kk}^{iH} \mathbf{y}_k + \sum_{l \in \mathcal{N}_k \backslash \{q\}} \mathbf{G}_{kl}^{iH} \mathbf{z}_{lk}^i \ , \forall k \notin \mathcal{C}_n \ , \forall n \tag{5.11}$$

to each of its neighbors $\forall q \in \mathcal{N}_k$. This transmission strategy (where node q is removed from $\mathcal{N}_k$ in the summation) effectively removes a feedback path that would otherwise develop by sending a signal including the fused signal of node q, $\mathbf{z}_{qk}$. It is noted that a similar transmission strategy and feedback removal could instead be employed as in (5.3) and (5.4) which then allows a node to broadcast the same signal to each of its neighbors. However, both can be shown to be theoretically equivalent [54] and (5.11) is now used to avoid an overly complex description of the T-DANSE algorithm.

Clique nodes whose neighboring nodes are not solely comprised of nodes in the same clique have a similar functionality to that of the CHs presented in Section 5.2 in that they are responsible for intra-clique communication (between nodes in the same clique) and inter-clique communication (between a clique and nodes not in the same clique). In fact, there is a duality between the inter-cluster signals (Section 5.2) and intra-clique signals. The signal that a clique node $k \in \mathcal{C}_n$ broadcasts to all other nodes in the clique $\mathcal{C}_n$ (intra-clique communication), is denoted as $\boldsymbol{\zeta}_k^i$. This signal is computed by fusing the sensor signal observations at node k along with the fused sensor signal observations of its neighbors that do not belong to the same clique n as node k, i.e., $q \in \mathcal{N}_k \backslash \mathcal{C}_n$ which is given as (compare with (5.2))

$$\boldsymbol{\zeta}_k^i = \mathbf{W}_{kk}^{iH} \mathbf{y}_k + \sum_{q \in \mathcal{N}_k \backslash \mathcal{C}_n} \mathbf{G}_{kq}^{iH} \mathbf{z}_{qk}^i \ , \forall k \in \mathcal{C}_n \ , \forall n \ . \tag{5.12}$$

In order for a clique node to disseminate information from its clique n, it uses a similar transmission strategy as in (5.11) where a different inter-clique

signal, $\mathbf{z}_{kq}^i$, is sent to each of its neighbors that are not in the same clique, i.e., non-clique nodes and nodes belonging to other cliques, given as

$$\mathbf{z}_{kq}^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \sum_{l\in\mathcal{N}_k\backslash\{\mathcal{C}_n,q\}} \mathbf{G}_{kl}^{iH}\mathbf{z}_{lk}^i + \sum_{l\in\mathcal{C}_n\backslash\{k\}} \mathbf{G}_{kl}^{iH}\boldsymbol{\zeta}_l^i \ , \forall k \in \mathcal{C}_n \ , \forall n \ . \quad (5.13)$$

This is in fact the same transmission strategy used by the non-clique nodes in (5.11) where the right most summation is set to zero. By plugging (5.3) into (5.4) the feedback cancellation for intra-cluster signals is theoretically equivalent to (5.13) as alluded to earlier. This inter-clique signal is therefore defined in the same manner to that of the intra-cluster broadcast signals given in (5.3).

Finally, for clique nodes whose neighbors only belong to the same clique, e.g., node k for which $\mathcal{N}_k = \mathcal{C}_n\backslash\{k\}$, the sum on the right hand side of (5.12) vanishes as $\mathcal{N}_k\backslash\mathcal{C}_n = \emptyset$ which then results in the fusion rule (2.31) for DANSE in a fully connected network. As previously noted, the intra-clique signals are defined similarly to that of the inter-cluster signals (5.2).

5.3.3 Data-driven signal flow

Since the nodes are not grouped into the same hierarchical structure as in Section 5.2, the same data-driven signal flow cannot be readily applied to the mixed-topology WSN. However, the definitions of the intra- and inter-clique signals (5.12), (5.13) as well as the non-clique nodes, who transmit the same signals as in the T-DANSE algorithm, have implicit causality constraints. This again allows for the signal flow to occur in a purely data-driven fashion that does not rely on network-wide coordination. The signal flow in the envisaged mixed-topology WSN can be thought of analogously to the T-DANSE algorithm in that it consists of a data flow toward the root node of the tree (fusion flow) and a data flow away from the root node (diffusion flow) [54]. However, if data-driven 'firing' rules are used at each node, the signal flow emerges naturally without an explicit definition and coordination of the fusion flow, the diffusion flow, or the root node. This is described in the sequel and illustrated in Figure 5.4.

The non-clique nodes transmit $\mathbf{z}_{kq}^i$ when they have received the (fused) signal observations from all of their neighbors, except of a neighbor node q. Note that this data-driven 'firing' rule holds both in the fusion and diffusion flow, and it will automatically initiate the former due to the presence of the so-called leaf nodes of the tree, which only have a single neighbor.

Once new sensor signal observations become available for clique nodes whose neighboring set is comprised only of nodes in the same clique, the intra-

clique signal, $\boldsymbol{\zeta}_k^i$ as defined in (5.12) (with vanishing rightmost sum), can be immediately computed and broadcast to all clique neighbors.

Clique nodes whose neighboring set is not comprised only of nodes in the same clique can broadcast their intra-clique signal (5.12) once they have received the signals from all of their non-clique neighbors. They can transmit their inter-clique signal (5.13) when they have received the (fused) signal observations from all of their neighbors, except of a neighbor node q.

If the matrix $\mathbf{G}_{k-k}^i$ again denotes the stacked versions of all $\mathbf{G}_{kq}^i, \forall q \in \mathcal{N}_k$ and $\widetilde{\mathbf{y}}_k^i$ denote the stacked vector of a node's sensor signals and fused sensor signals from neighboring nodes, the DANSE algorithm in a mixed-topology WSN can be implemented in the same fashion as in a heterogeneous WSN given in Table 5.1.

It is re-iterated that the signal flow or the in-network signal fusion happens independently from the distributed algorithm that updates the local estimator parameters at each node (see Table 5.1), i.e., the estimate of $\mathbf{d}_k$, at node k, can be found at any point in the iterative process as

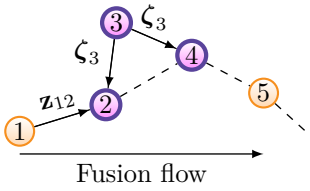
$$\widetilde{\mathbf{d}}_k^i = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \mathbf{G}_{k-k}^{iH}\mathbf{z}_{\mathcal{N}_k}^i. \tag{5.14}$$

5.3.4 Convergence and optimality

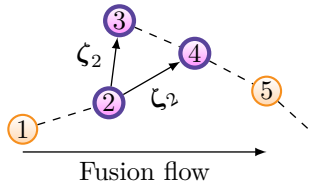
The mixed-topology WSN can be thought of as an extension to the hierarchical structure presented in Section 5.2, which now contains many fully connected components that are connected via their MNs. Due to the constraint that there must be a unique path between any two cliques, there also exists a unique *shortest* path, $P_{p_1 \leftarrow p_t}$, between any two nodes in the network. Defining a matrix that follows this shortest path $\mathbf{G}_{p_1 \leftarrow p_t}$, in the same fashion as (5.5), then it can be shown that the parameterization given in (5.6) is the same for the mixed-topology WSN and Theorem 1 still holds.

Since the mixed-topology WSN can be abstracted to that of a tree topology, the same convergence as that of the T-DANSE algorithm can be guaranteed as long as the individual cliques also converge. Convergence for the mixed-topology WSN can therefore be shown in a similar manner as in Theorem 2.

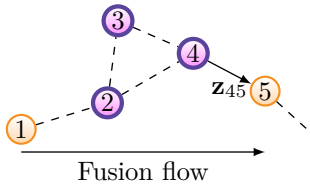
Corollary 1. *Consider a WSN with a mixed-topology. Let P represent an ordered set of nodes that defines a unique path through the WSN that traverses through every node and begins at node k and ends with a neighbor, $q \in \mathcal{N}_k$. If (2.6) holds, then the DANSE algorithm as described in Table 5.1 converges for every node k and for any initialization of its parameters to the centralized estimator given in (2.26).*



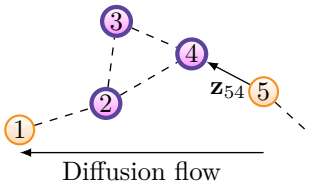
(a) Leaf nodes and clique nodes which $\mathcal{N}_k = \mathcal{C}_n \backslash \{k\}$ transmit their signals.



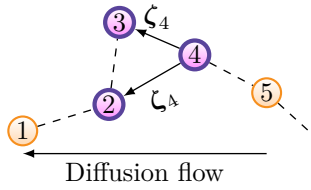
(b) Node 2 now has the signals needed for its intra-clique signal (5.12).



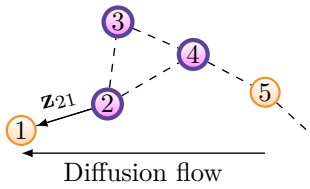
(c) Node 4 now has the signals needed for its inter-clique signal (5.13).



(d) The diffusion flow originates at the root node and has reached node 5, who transmits the inter-clique signal $\mathbf{z}_{54}$ to its neighbors.

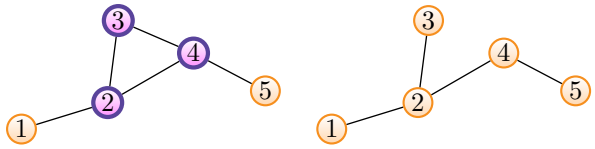


(e) Node 4 now has signals needed for intra-clique signal (5.12).



(f) Node 2 now has signals needed for inter-clique signal (5.13).

Figure 5.4: A depiction of the data-driven signal flow of DANSE in a mixed-topology WSN where node 5 is the closest to the root node.



(a) A mixed-topology WSN with a single clique.

(b) Tree topology based on the mixed-topology WSN.

Figure 5.5: A possible updating order for the mixed-topology (a) is $P = [5, 4, 3, 2, 1, 2, 4]$ and for the tree topology (b) is $P = [5, 4, 2, 3, 2, 1, 2, 4]$.

Due to the similarities between the heterogeneous WSN and mixed-topology WSN, i.e., the cliques may be viewed similarly to the CHs and the non-clique nodes similarly to the MNs, the proof of convergence follows from a generalization of the proof of Theorem 2. The corollary then follows by recursively applying Lemmas 1 and 2, in a similar fashion as in the proof of Theorem 2.

Corollary 1 states that the updating order must correspond to a unique path through the WSN that traverses every node at least once. The convergence of the DANSE algorithm in a mixed-topology WSN also implies that if node k were to update its node-specific parameters then the next node, say q, in the updating order must be a neighbor of node k, i.e., $q \in \mathcal{N}_k$. However this is only a sufficient not a necessary condition which is explained in more detail in Remark 2. An example comparing a possible updating order for DANSE in a mixed-topology WSN to that of T-DANSE is given in Figure 5.5. For the mixed-topology in Figure 5.5a a possible updating order is $P_{\text{mixed}} = [5, 4, 3, 2, 1, 2, 4]$ and for the tree topology in Figure 5.5b a possible updating order is $P_{\text{tree}} = [5, 4, 2, 3, 2, 1, 2, 4]$. It is shown that the length of P_{mixed} is smaller than the length of P_{tree}. This means that it takes fewer iterations to update each node at least once, which often leads to a faster overall convergence, as shown in Section 5.4. Another reason for this increased convergence can be explained by larger degrees of freedom available at the clique nodes and is explained in more detail in Remark 3. It is noted that P_{tree} is also a valid updating order in a mixed-topology WSN, but P_{mixed} is not valid in a tree topology.

Remark 2. *It was noted in [54] that the condition that successive updates occur in neighboring nodes was a sufficient, but not a necessary condition in order to guarantee convergence, meaning that convergence is also sometimes possible if the nodes update in any order regardless of their place in the WSN. However, this does not hold in general, i.e., simulations show that convergence is only guaranteed if the updating order indeed follows a path through the network. Although this imposes some additional coordination, it was noted in [54] that such a path-based updating order not only guarantees convergence, but it usually*

also yields the fastest convergence.

Remark 3. *In a mixed-topology WSN, the total number of links is obviously larger than in an exact (clique-free) tree that can be obtained by pruning some of the links of the mixed-topology WSN. Hence, the number of fused sensor signals at the clique-nodes is larger, yielding more degrees of freedom to perform an update. On average, this results in an overall faster convergence than for the T-DANSE algorithm operating in a clique-free tree. It is noted that this is purely due to the additional degrees of freedom in the per-node LMMSE optimization (5.8) and not due to a faster dissemination or diffusion of information as in consensus- or gossip-based approaches such as, e.g., [103]. Indeed, in this framework, the signal fusion itself is not an iterative process, i.e., it is a finite in-network fusion process, as explained in Sections 5.2.2 and 5.3.3, which does not have any influence on the convergence speed of the distributed node-updating algorithm.*

5.3.5 Changing topologies

For changing topologies, links are added and removed between nodes. It is assumed, however, that the WSN can still be partitioned into a set of fully connected and tree topologies that form a connected WSN.

Using Figure 5.5 as an example, the WSN is first in a mixed-topology where the link between node 3 and 4 fails, creating a tree topology. It is first assumed that the neighbors of a node after the link removal are a subset of the nodes before the link removal, i.e., if $\mathcal{N}_k^i$ represents the neighbors of node k at iteration i before link removal, and $\mathcal{N}_k^{i+1}$ represents the neighbors of node k at iteration $i+1$ after link removal, then $\mathcal{N}_k^{i+1} \subseteq \mathcal{N}_k^i$.

If the DANSE algorithm has converged such that the LMMSE estimators at each node are $\mathbf{W}_{kk}^i = \widehat{\mathbf{W}}_{kk}$ and $\mathbf{G}_{kq}^i = \overline{\mathbf{A}}_{kq}$, then removing the link does not affect the optimality at each node if $\mathcal{N}_k^{i+1} \subseteq \mathcal{N}_k^i$. This can be described using the parameterization of (5.6) before and after the removal of the link. Using (5.5) and the proof of Theorem 2, the parameterization of (5.6) for node 3 before link removal is given as

$$\mathbf{W}_3^i = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{3\leftarrow 1}^i \\ \mathbf{W}_{22}^i \mathbf{G}_{3\leftarrow 2}^i \\ \mathbf{W}_{33}^i \\ \mathbf{W}_{44}^i \mathbf{G}_{3\leftarrow 4}^i \\ \mathbf{W}_{55}^i \mathbf{G}_{3\leftarrow 5}^i \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{21}^i \mathbf{G}_{32}^i \\ \mathbf{W}_{22}^i \mathbf{G}_{32}^i \\ \mathbf{W}_{33}^i \\ \mathbf{W}_{44}^i \mathbf{G}_{34}^i \\ \mathbf{W}_{55}^i \mathbf{G}_{45}^i \mathbf{G}_{34}^i \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{11} \overline{\mathbf{A}}_{31} \\ \widehat{\mathbf{W}}_{22} \overline{\mathbf{A}}_{32} \\ \widehat{\mathbf{W}}_{33} \\ \widehat{\mathbf{W}}_{44} \overline{\mathbf{A}}_{34} \\ \widehat{\mathbf{W}}_{55} \overline{\mathbf{A}}_{35} \end{bmatrix} = \widehat{\mathbf{W}}_3 \,. \quad (5.15)$$

When removing the link, the paths from nodes 4 and 5 to node 3 now go through node 2. Using the fusion signal rules based on (5.11) the parameterization of node 3 now becomes

$$\mathbf{W}_3^i = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{3\leftarrow 1}^i \\ \mathbf{W}_{22}^i \mathbf{G}_{3\leftarrow 2}^i \\ \mathbf{W}_{33}^i \\ \mathbf{W}_{44}^i \mathbf{G}_{3\leftarrow 4}^i \\ \mathbf{W}_{55}^i \mathbf{G}_{3\leftarrow 5}^i \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{21}^i \mathbf{G}_{32}^i \\ \mathbf{W}_{22}^i \mathbf{G}_{32}^i \\ \mathbf{W}_{33}^i \\ \mathbf{W}_{44}^i \mathbf{G}_{24}^i \mathbf{G}_{32}^i \\ \mathbf{W}_{55}^i \mathbf{G}_{45}^i \mathbf{G}_{24}^i \mathbf{G}_{32}^i \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{11} \overline{\mathbf{A}}_{31} \\ \widehat{\mathbf{W}}_{22} \overline{\mathbf{A}}_{32} \\ \widehat{\mathbf{W}}_{33} \\ \widehat{\mathbf{W}}_{44} \overline{\mathbf{A}}_{34} \\ \widehat{\mathbf{W}}_{55} \overline{\mathbf{A}}_{35} \end{bmatrix} \widehat{\mathbf{W}}_3 \, . \tag{5.16}$$

Since the $\mathbf{G}_{kq}$ coefficients are known optimally, even though a link was removed, the network wide LMMSE estimators are still found.

When links are added or more generally when nodes are mobile with changing neighbors, such that $\mathcal{N}_k^{i+1} \not\subset \mathcal{N}_k^i$, the parameterization becomes more complicated. Using the previous example, going from topology in Figure 5.5b to the one in Figure 5.5a node 3 must initialize a new variable $\mathbf{G}_{34}^i$ to apply to $\boldsymbol{\zeta}_4^i$ and node 4 initializes a new variable $\mathbf{G}_{43}^i$ to apply to $\boldsymbol{\zeta}_3^i$ which would require re-convergence. However, with some coordination and extra broadcasting between nodes this re-converge can be circumvented. Using the same example, before the link between node 4 and 3 is made, the estimator of node 2 is given as

$$\widehat{\mathbf{W}}_2 = \begin{bmatrix} \widehat{\mathbf{W}}_{22} \\ \mathbf{G}_{2_-2}^i \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{22} \\ \mathbf{G}_{21}^i \\ \mathbf{G}_{23}^i \\ \mathbf{G}_{24}^i \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{22} \\ \overline{\mathbf{A}}_{21} \\ \overline{\mathbf{A}}_{23} \\ \overline{\mathbf{A}}_{24} \end{bmatrix} \, . \tag{5.17}$$

Once the link between node 3 and node 4 is made, then if node 2 were to transmit $\mathbf{G}_{24}^i$ to node 3, $\mathbf{G}_{34}^i$ can be found optimally at node 3 by a simple product given as

$$\mathbf{G}_{34}^i = \mathbf{G}_{24}^i \mathbf{G}_{32}^i = \overline{\mathbf{A}}_{34} \, . \tag{5.18}$$

This can directly be applied to $\boldsymbol{\zeta}_4^i$, which again produces the parameterization of (5.15). Obviously, as the topology becomes more complicated and with several changing links, more $\mathbf{G}$ coefficients will need to be exchanged between not necessarily neighboring nodes.

If the DANSE algorithm has not yet converged and the topology continually changes then this will affect convergence as the parameterization of (5.6) is topology dependent due to $\mathbf{G}_{p_1\leftarrow p_t}$. Essentially this means that at every iteration, the nodes will be optimizing a different problem with regards to $\mathbf{G}_{k_-k}^i$ in (5.8) preventing convergence of the algorithm.

5.4 Simulations

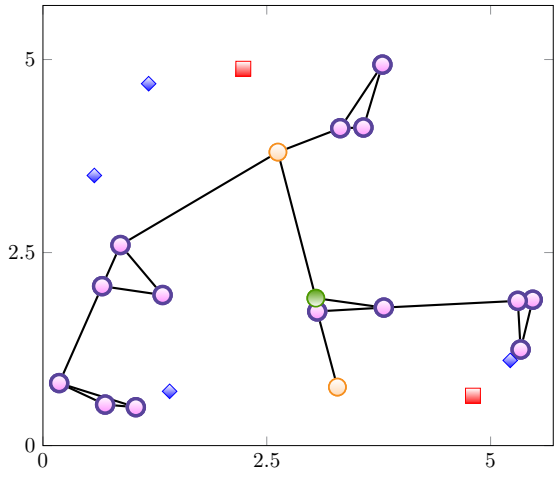
In this section numerical simulations were performed by way of the DANSE algorithm in a mixed-topology WSN. The simulations were implemented in batch mode as described in Section 4.4.1 where the MSE cost function at each node was replaced by the corresponding finite-length equivalent LS cost (4.35).

An example is first given of the DANSE algorithm in a mixed-topology WSN compared to the DANSE algorithm in a fully connected WSN as well as in a tree topology WSN with various node updating orders. In Section 5.4.2, Monte-Carlo simulations were performed on 1000 randomly generated sensing environments.

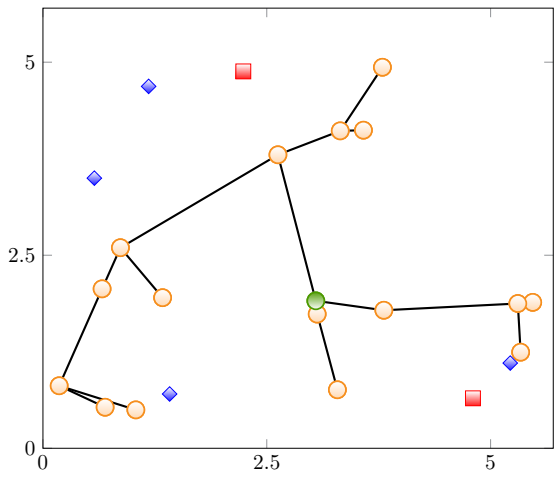
5.4.1 Single sensing environment

An example sensing environment as depicted in Figure 5.6a is first considered. The WSN consisted of 17 nodes deployed throughout a sensing environment. Each node had $M_k = 3$ sensors that were spaced uniformly around a 10 cm radius from the center of the node. There were $Q = 2$ sources distributed throughout the environment (■), where each source signal consisted of $T = 10000$ samples which were i.i.d uniformly on the interval [-0.5 0.5]. The coefficients of $\mathbf{A}$ in (2.27) were proportional to an attenuation factor of $\frac{1}{d}$ where d was the distance between the source and the sensor. The desired signal $\mathbf{d}_k$ at node k was defined as in (2.6) where the matrix $\overline{\mathbf{A}}_k$ was a 2×2 matrix that consisted of a subset of rows of $\mathbf{A}$ corresponding to two randomly selected sensors of node k. There were also 4 spatially located white noise sources (◆) where the signals were generated with a similar process as the source signals. Spatially uncorrelated white noise, representative of process noise, that was half the average power of the source signals, was added to each sensor.

There were $N = 5$ cliques, each containing 3 nodes, which were first deployed in the sensing environment along with 2 other non-clique nodes. A nearest neighbor broadcasting scheme was then used, where the transmission ranges of the nodes were increased until the network was connected, i.e., every node was reachable in the WSN. An MST algorithm was then performed where each clique was considered to be a single node in the topology. The resultant topology was then a set of cliques that are connected via a tree topology and is depicted in Figure 5.6a. In order to compare the convergence results to that of the T-DANSE algorithm the links between clique nodes, who do not have a non-clique neighbor, were pruned so that the WSN had a tree topology which is depicted in Figure 5.6b.



(a)



(b)

Figure 5.6: (a) A simulated environment with 2 desired signal sources (■), 4 noise sources (◆) and a WSN consisting of 17 nodes where each node has 3 sensors and where the root node is represented as ●. The nodes are grouped in $N = 5$ cliques containing 3 nodes each and two non-clique nodes. (b) A spanning tree using the original node positions given in (a).

An ideal transmission scheme was used as in [151] where the transmission rate was constant for every sensor and delays in the system were ignored. The power required to transmit from node k to node q was given as

$$P(r_{k\to q}) = Cd_{k\to q}^{\alpha} \tag{5.19}$$

where C is a constant ($C \approx 10^{-10} J/m^{\alpha}/bit$), $\alpha = 2.3$ is a power loss factor (nominally between 2 and 6), and $d_{k\to q}$ is the distance between node k and node q. Note that if the WSN is in a large, heterogeneous environment, α could be defined uniquely for each node [152]. A sensor link capacity, S, of 212kb/s was assumed. The transmission energy required for each sensor e_n is then given by

$$e_k(d_{k\to q}, S) = \max_{q\in\mathcal{N}_k} SCd_{k\to q}^{\alpha} \,. \tag{5.20}$$

In the case of a fully connected WSN, $d_{k\to q}$ was chosen as the distance between node k and its furthest neighbor.

For the sake of brevity, DANSE in a fully connected WSN will be denoted as DANSE-FC and DANSE in a mixed-topology WSN will be denoted as DANSE-MT.

The minimal LS cost was first found in a centralized fashion, where it was assumed that all sensor signal observations of the entire WSN were available to compute (4.35). The updating order of DANSE-FC was also run in two different variations, first where the nodes updated their node-specific parameters in a round-robin fashion denoted as DANSE-FC-RR, and second where the nodes updated their node-specific parameters according to P, i.e., following the same updating order as T-DANSE denoted as DANSE-FC-TUO. This allows a comparison of both the effect of the per-node updating frequency and the per-node degrees of freedom on the convergence speed. The T-DANSE algorithm was implemented with a path-based tree updating order, P, where the WSN was connected as in Figure 5.6b. DANSE-MT was also implemented in two different variations, first where the nodes updated their node-specific parameters using an update order as described in Section 5.3.4 denoted as DANSE-MT-MUO, and second where the nodes updated their node-specific parameters following the update order according to P, i.e., in the same order as in T-DANSE, denoted as DANSE-MT-TUO.

Figure 5.7 (top) shows the LS cost at the root node, represented as ●, (bottom) shows the sum of the LS cost functions at each node, i.e., $\sum_{k=1}^{K} J_k(\mathbf{W}_k^i)$ for the various implementations of the DANSE algorithm. It is shown that DANSE-FC-RR converges the fastest to the optimal solution. DANSE-MT-MUO and DANSE-MT-TUO both outperform the T-DANSE algorithm.

The improved convergence of DANSE-MT compared to that of T-DANSE may also be attributed to the fact that the nodes in the cliques have an

increased number of signals, and hence more degrees of freedom in the local LS minimization problem, whereas the degrees of freedom for the T-DANSE is restricted to that of a smaller set of neighbors. However, this improved convergence only holds on average, i.e., it has been observed in rare cases that DANSE-MT or even DANSE-FC converged slower than the T-DANSE algorithm. This actually occurs more often in smaller networks, e.g, when $K < 4$.

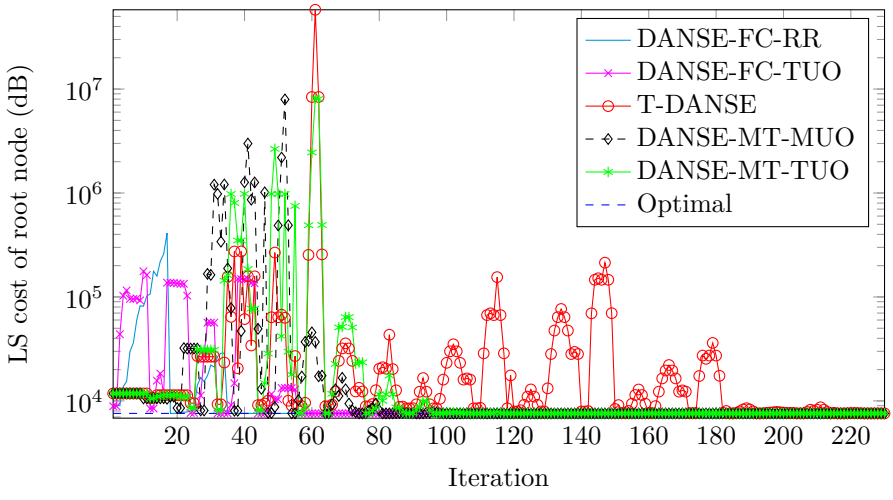
The update order of T-DANSE must also traverse a specific path through the WSN as stated in Remark 2. With a highly branched tree, this updating order may have a considerable number of steps. Since the updating order in a clique can be relaxed to mimic that of DANSE-FC, i.e., where nodes with non-clique neighbors update in a round-robin fashion, the update order for DANSE-MT can always be defined using a smaller number of steps than in T-DANSE. This also contributes to the faster convergence of DANSE-MT compared to that of T-DANSE. This effect is demonstrated in Figure 5.7 by observing that DANSE-MT-MUO converges faster than DANSE-MT-TUO. Indeed as discussed in Section 5.3.4, the length of P_{mixed} is smaller to that of P_{tree} which means that it takes fewer iterations to update each node at least once, which often leads to a faster overall convergence.

The total sum of energy used for transmission for all of the nodes in the WSN using DANSE-FC, T-DANSE and DANSE-MT is shown in Figure 5.8. Although DANSE-FC converges much faster, it also requires a larger amount of energy when compared to that of DANSE-MT and T-DANSE. DANSE-MT converges significantly faster than T-DANSE, but requires hardly any additional energy. Since the tree topology is a pruned version of the mixed-topology, there is a 3% difference in the total energy used in the WSN between the DANSE-MT and T-DANSE.

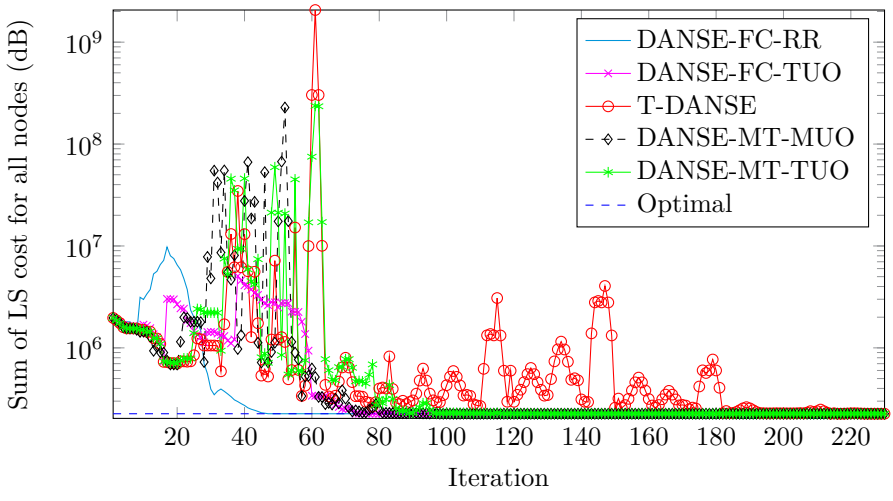
5.4.2 Monte-Carlo simulations

Monte-Carlo simulations were performed on 1000 sensing environments with similar parameters as those presented in Section 5.4.1. The sensing environments were constructed with $N = 4$ cliques all consisting of 3 nodes and 5 additional non-clique nodes. The mixed-topology WSN and tree topology WSN were derived using the same method as presented in Section 5.4.1.

The optimal LS cost for each Monte-Carlo run differs due to the random nature of the generated signals and the placement of nodes throughout the sensing environment. In order to account for this, the summed difference between the LS cost at each iteration and the optimal summed LS cost, $\sum_{k=1}^{K} J_k(\mathbf{W}_k^i) - J_k(\widehat{\mathbf{W}}_k)$,



(a)



(b)

Figure 5.7: The LS cost of the root node (a) and the sum of LS cost of all of the nodes in the network (b) versus the number of iterations for DANSE-FC with round robin updating (DANSE-FC-RR), and a tree updating order (DANSE-FC-TUO), T-DANSE, DANSE-MT using an update order as described in Section 5.3.3 (DANSE-MT-MUO), and a tree updating order (DANSE-MT-TUO).

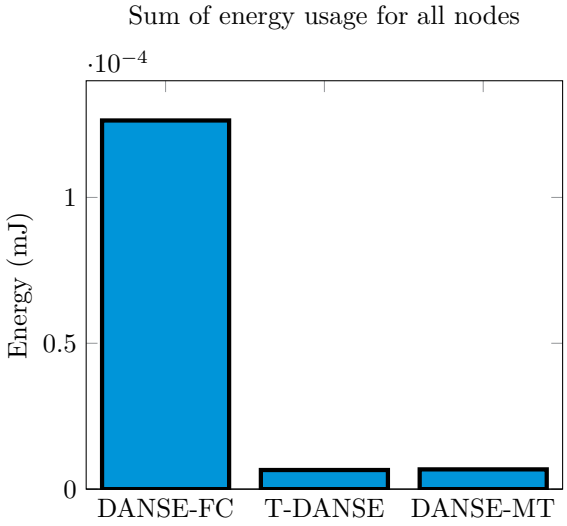


Figure 5.8: Sum of the energy used for transmission using DANSE-FC, T-DANSE, and DANSE-MT.

was normalized by the optimal summed LS cost, i.e.,

$$\tilde{J}^i_{Tot} = \frac{\sum_{k=1}^{K} J_k(\mathbf{W}_k^i) - J_k(\widehat{\mathbf{W}}_k)}{\sum_{k=1}^{K} J_k(\widehat{\mathbf{W}}_k)} \tag{5.21}$$

for each Monte-Carlo run.

Figure 5.9 shows the median of the sum of the normalized LS cost of all of the nodes in the network versus the number of DANSE iterations where again DANSE in a mixed-topology has convergence properties in between that of DANSE in a fully connected and tree topology. Figure 5.10 shows the median over the 1000 simulated sensing environments of the total sum of energy used for transmission for all of the nodes in the WSN using DANSE-FC, T-DANSE and DANSE-MT.

There was a 28% decrease in the number of iterations needed for convergence when comparing T-DANSE to DANSE-MT-MUO. This decrease in the number of iterations only comes at a 2.6% increase in the sum of energy consumption between T-DANSE and DANSE-MT.

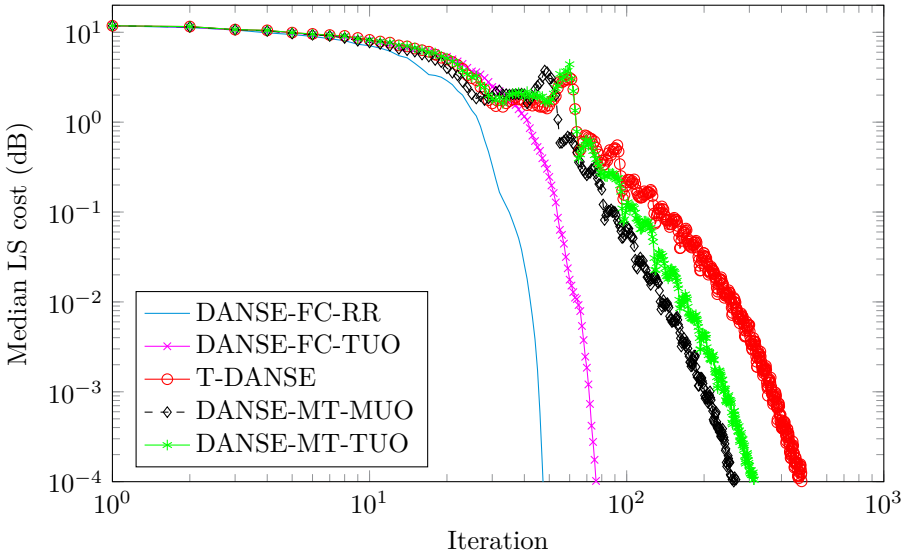


Figure 5.9: The median of the sum of the LS cost of all of the nodes in the network versus the number of iterations for the previously defined DANSE algorithms.

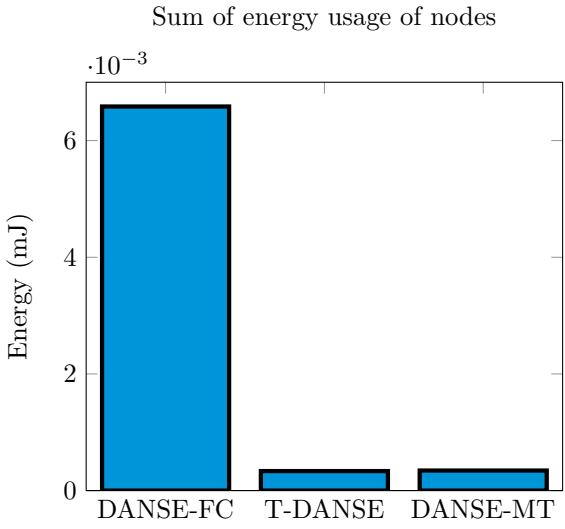


Figure 5.10: The average sum of energy used for transmission for DANSE-FC, T-DANSE and DANSE-MT for the Monte-Carlo simulations.

5.5 Conclusion

In this chapter, a WSN was partitioned into a set of substructures that encompassed both heterogeneous and mixed-topology WSNs. The fusion rules of the DANSE algorithm were modified to take into account this type of partitioning showing a duality between the heterogeneous and mixed-topology case. It was shown that the DANSE algorithm in both cases converges to the same solution as if every node had access to all of the sensor signal observations in the WSN, which was also confirmed by means of numerical simulations. Using the DANSE algorithm in a mixed-topology WSN also improved the convergence speed when compared to the T-DANSE algorithm with only a slight increase in the per-node energy usage. This is due to two complementary effects, i.e., the increased per-node updating frequency, and the increased number of degrees of freedom in the local optimization task at certain nodes.

Chapter 6

Alternative broadcast strategies and topology control

6.1 Introduction

In this chapter, the T-DANSE algorithm is briefly reviewed, focusing not only on the signal flow but also on the formation of the tree topology from a pre-existing ad-hoc topology. The methods presented in this chapter using the T-DANSE algorithm can be readily applied to the topologies in Chapter 5 as tree topologies were considered building blocks of the presented mixed-topologies. In fact, the fusion rules that the nodes use in the T-DANSE algorithm are equivalent to the fusion rules in Chapter 5 when considering only a single cluster or cliques that consist of a single node.

The T-DANSE algorithm has two modes of signal flow, much like the mixed-topologies: a fusion flow where signals travel from the outer most nodes or leaf nodes which have a single neighbor, toward the root node; and a diffusion flow, where the signals travel away from the root node. However, this entails that each non-root and non-leaf node transmits two signals, one for each signal flow, which uses a significant portion of the already limited energy resources.

A modification to the signal flow between nodes in the T-DANSE algorithm is proposed, which is also applicable in any of the abstracted tree topologies in Chapter 5. The fusion flow in the proposed modification is identical to the

fusion flow in the original T-DANSE algorithm; however, the diffusion flow is now replaced with a single broadcast signal from the root node. This is shown to shift the majority of energy usage for signal transmission to the root node, thereby reducing the energy usage of non-root nodes, which again lends itself to such applications as heterogeneous WSNs, as presented in Chapter 5, where it can be assumed that the sink node typically has larger communication and energy capabilities [153, 154]. In many applications it is assumed that the sink node already communicates directly to the other nodes [153, 155], therefore this type of broadcasting exploits this type of connectivity. The proposed data flow is also shown to reduce the input-output delay in the tree topology.

One aspect of the previously introduced mixed-topologies and the T-DANSE algorithm in general is that the network topology is already formed, and in the case of the T-DANSE algorithm, that the topology has already been pruned to that of a spanning tree. However, due to the signal flow in the T-DANSE algorithm, the way that the tree topology is formed, especially the selection of the root node, is shown to have a significant impact on the performance of the T-DANSE algorithm. This is mainly due to the fact that there are estimation errors in certain second-order statistics estimates at each node which tend to become more prevalent with decreasing $\mathrm{SNR}_{\mathrm{IN}}$.

Since all signals have to pass through the root node, it is important that the signal fusion rule at this node is correctly estimated, which is only possible if the $\mathrm{SNR}_{\mathrm{IN}}$ at the root node is sufficiently high. A similar statement (be it to a lesser degree) holds for the direct neighbors of the root node, as well as their respective neighbors, etc. Furthermore, to minimize the input-output delay within the network, the root node should be close (in terms of number of hops) to any other node in the network.

Before the T-DANSE algorithm begins, it is first assumed that the network initially has an ad-hoc topology where each node only communicates with nearby nodes. The concept of eigenvector centrality is introduced with an $\mathrm{SNR}_{\mathrm{IN}}$ weighted adjacency matrix which will allow a suitable root node to be identified and to prune the network to a tree topology which satisfies the earlier mentioned properties. A particularly convenient property of the eigenvector centrality is that it can be computed in a distributed fashion.

This chapter is organized as follows. In Section 6.2 the T-DANSE algorithm is reviewed, with emphasis on its signal flow, and ways to remove signal feedback. Section 6.3 introduces the proposed modification to the signal flow of the T-DANSE algorithm and discusses the equivalence in terms of signal estimation performance to the original T-DANSE algorithm. In Section 6.4 the impact of estimation errors of the second-order statistics on the T-DANSE algorithm is described. In Section 6.5 a weighted eigenvector centrality measure is introduced

which is used in pruning an ad-hoc network to a tree topology as well as root node selection which helps mitigate the effect of the errors in the second-order statistics. Section 6.6 contains simulations showing first, that the proposed modification to the signal flow shifts the majority of the transmission energy to that of the root node and second that the proposed method of topology construction based on eigenvector centrality improves the performance of the WSN. Finally, conclusions are given in Section 6.7.

6.2 The T-DANSE algorithm

The nodes, in this chapter, estimate their node-specific desired signals by using the T-DANSE algorithm[1] where each node uses its own sensor signals and linearly compressed and combined signals from neighboring nodes. The fusion rules of the T-DANSE algorithm can be described in a similar fashion to the mixed-topologies of Chapter 5 with some slight modifications (see (6.2) - (6.6)).

In the T-DANSE algorithm, the nodes perform an LMMSE estimation by using their sensor signal observations $\mathbf{y}_k$ and $|\mathcal{N}_k|$ $\mathbf{z}$-signals from their neighbors in the tree which are defined in the sequel. The LMMSE estimation is similar to (2.35) which is repeated here for convenience

$$\widetilde{\mathbf{W}}_k^{i+1} = \begin{bmatrix} \widetilde{\mathbf{W}}_{kk}^{i+1} \\ \\ \widetilde{\mathbf{G}}_{k_{-k}}^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_{k_{-k}}}{\arg\min}\, \mathcal{E}\left\{ \left\| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}_{kk}^H & \mathbf{G}_{k_{-k}}^H \end{bmatrix} \widetilde{\mathbf{y}}_k^i \right\|_2^2 \right\} \tag{6.1}$$

where $\widetilde{\mathbf{y}}_k^i$ is defined as in (5.7) and $\mathbf{G}_{k_{-k}}^i$ is a $|\mathcal{N}_k| \times Q$-dimensional matrix.

6.2.1 Fusion flow

The broadcast rules for leaf nodes, i.e., nodes with a single neighbor, are given in the same manner as (5.1) which is repeated here for convenience,

$$\mathbf{z}_{kq}^i = \mathbf{W}_{kk}^{iH} \mathbf{y}_k \tag{6.2}$$

where the subscript kq indicates this is the transmitted signal from node k to node q.

[1] It should be noted that the following description is only meant to be a succinct summary of the T-DANSE algorithm where [54] contains a more in depth discussion as well as convergence proofs.

After all of the leaf nodes have transmitted their signals, the receiving (non-root) nodes, combine these signals with their own sensor signals by

$$\mathbf{z}_{kq}^{i} = \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \sum_{l\in\mathcal{N}_k\setminus\{q\}} \mathbf{G}_{kl}^{iH}\mathbf{z}_{lk}^{i} \tag{6.3}$$

and forward the result to the single node q from which they have not received any signal yet, i.e., the neighbor that is one hop closer to the root node. The fusion process continues until the root node has been reached.

6.2.2 Diffusion flow

The diffusion flow is initiated once the root node receives all the signals from its neighboring nodes. The root node then locally broadcasts to its neighbors the combination of its own sensor signals as well as the signals from all of its neighbors which is equivalent to (5.3) without the neighboring cluster signals, i.e.,

$$\mathbf{z}_{k}^{i} = \mathbf{W}_{kk}^{iH}\mathbf{y}_k^{i} + \sum_{q\in\mathcal{N}_k} \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^{i}. \tag{6.4}$$

Each node now broadcasts (6.4) to its neighbors, and this process is repeated up the tree and terminates at the leaf nodes.

6.2.3 Feedback cancellation

As in the case with the mixed-topologies signal fusion rules presented in Chapter 5, a feedback path develops in the T-DANSE algorithm where the received diffusion flow signal contains the fusion flow signal of a node, which again prevents the nodes from converging to their optimal LMMSE solution. Therefore, the T-DANSE algorithm relies on feedback cancellation to restore its convergence properties. Feedback cancellation can be accomplished by one of two means in the T-DANSE algorithm, transmitter feedback cancellation (TFC), which is similar to the strategy proposed in Chapter 5 for the MNs, and receiver feedback cancellation (RFC) which relies on the broadcasting of $\mathbf{G}_{kq}$'s to neighboring nodes. However, both methods are equivalent from a theoretical point of view. TFC signals are used when each node pair has a reserved point-to-point communication link, whereas RFC signals also allow for local broadcasts where a node transmits the same signal to all its neighbors.

6.2.4 Transmitter feedback cancellation (TFC)

Transmitter feedback cancellation is performed in the same manner as discussed in Section 5.3.2 where nodes use a point-to-point communication protocol and transmit a different signal to each of their neighbors, e.g. the TFC signal from node k to node q is given as

$$\begin{aligned}\mathbf{z}_{kq}^{i} &= \mathbf{W}_{kk}^{iH}\mathbf{y}_k + \sum_{l\in N_k\setminus\{q\}} \mathbf{G}_{kl}^{iH}\mathbf{z}_{lk}^{i} \\ &= \mathbf{z}_k^i - \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^{i}. \end{aligned} \tag{6.5}$$

6.2.5 Receiver feedback cancellation (RFC)

The feedback signal can be equivalently removed by using so-called receiver feedback cancellation RFC. In RFC the nodes have access to the $\mathbf{G}_{kq}^{i}$ coefficient that is applied to their signal during the fusion flow, which allows the nodes to effectively cancel out their own contribution from the received diffusion flow signal. For example, node q receives the $\mathbf{G}_{kq}^{i}$ that was applied to its signal during the fusion flow from its neighbor node k and applies this to its received diffusion flow signal (6.4) as

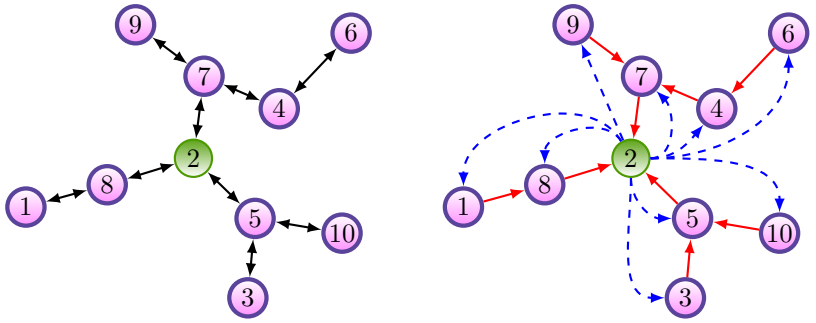
$$\mathbf{z}_{kq}^{i} = \mathbf{z}_k^i - \mathbf{G}_{kq}^{iH}\mathbf{z}_{qk}^{i} \tag{6.6}$$

which is equivalent to (6.5). It is noted that while this requires an additional data flow of parameters, $\mathbf{G}_{kq}^{i}$, these are updated at a much slower rate than the rate of the fusion and diffusion, which must be executed at every instance of new data.

6.2.6 Parameterized solution space

A path is now defined in the same fashion as in Section 5.2.3, $P_{p_1\to p_t}$, which consists of an ordered set of nodes. In Figure 6.1a, going from node 1 to node 6, the path is $P_{1\to 6} = \{1, 8, 2, 7, 4, 6\}$ where the inverse path, $P_{p_1\leftarrow p_t}$ is equivalent. The matrix $\mathbf{G}_{p_1\leftarrow p_t}$ is also defined, which follows the previously defined inverse path $P_{p_1\leftarrow p_t}$, which using the previous example, corresponds to $\mathbf{G}_{1\leftarrow 6} = \mathbf{G}_{46}\mathbf{G}_{74}\mathbf{G}_{27}\mathbf{G}_{82}\mathbf{G}_{18}$.

Using this definition of $\mathbf{G}_{p_1\leftarrow p_t}$, the T-DANSE algorithm can therefore be thought of as a parameterization of (2.26), that at node k, effectively corresponds



(a) Data flow in the original T-DANSE algorithm.

(b) Data flow in the modified T-DANSE algorithm.

Figure 6.1: Signal flow in the original and modified T-DANSE algorithm where the root node (node 2) is represented as ●. The solid arrows in (b) indicate the fusion flow which is equivalent to the original T-DANSE formulation (a). The dotted arrows in (b) indicate the broadcast signal flow from the root node to all other nodes in the WSN.

to

$$\mathbf{W}_k^i = \begin{bmatrix} \mathbf{W}_{11}^i \mathbf{G}_{k\leftarrow 1}^i \\ \vdots \\ \mathbf{W}_{KK} \mathbf{G}_{k\leftarrow K}^i \end{bmatrix} \tag{6.7}$$

which is defined in the same manner as (5.6).

Using this parameterization, it was shown in [54] that the optimal estimators given in (2.26) are in the solution space defined by (6.7).

6.3 Modified signal flow in the T-DANSE algorithm

In Section 6.2 the T-DANSE algorithm was described where each node transmits signals to its neighboring nodes in the fusion and diffusion signal flow. However, it is apparent that each non-leaf and non-root node must transmit two signals, one for the diffusion and one for the fusion flow, together with the portion of their node-specific estimators that are applied during the fusion flow, $\mathbf{G}_{kq}^i$.

6.3.1 Root node broadcast signal and feedback cancellation

In order to reduce the transmission burden at each node, a method is proposed to eliminate the diffusion signal flow and allow the root node to broadcast a single signal (6.4) to all other nodes assuming the root node has a large (virtually infinite) energy budget. An example of the envisaged signal flow is given in Figure 6.1b. This method, however, still requires that the $\mathbf{G}_{kq}^i$'s be transmitted to neighboring nodes, however this is done over longer time intervals.

Although each node receives this signal from the root node, it still needs a way to cancel out its own signals contribution in order to arrive at the optimal LMMSE estimators. This can be accomplished in the same manner as the RFC feedback cancellation, where nodes transmit their $\mathbf{G}_{kq}^i$'s to neighboring nodes but now, instead of only being used by a neighboring node, they are multiplied with one another as they are transmitted further up the tree.

It is now assumed, that node r is the root node of the tree and a path from the root node to a node k is defined as $P_{k\to r}$ where the inverse path is again given as $P_{k\leftarrow r}$. In the proposed diffusion flow each node receives the broadcast signal from the root node and $\mathbf{G}_{k\leftarrow r}^i$ that originates at the root node and is a product of all of the $\mathbf{G}_{mn}$'s (with $n \in N_m$ and $m, n \in P_{k\leftarrow r}$) between the node and the root node. This differs from the parameter flow given in Section 6.2.5 where the $\mathbf{G}_{kq}^i$'s are only used from neighboring nodes.

Node k then receives the broadcast signal from the root node as well as $\mathbf{G}_{k\leftarrow r}^i$, which is applied to the root node broadcast signal as

$$\mathbf{z}_{k\leftarrow r}^i = \mathbf{z}_r^i - \mathbf{G}_{k\leftarrow r}^{iH}\mathbf{z}_{kq}^i \tag{6.8}$$

where $\mathbf{z}_{kq}^i$ is the signal transmitted by node k in the fusion flow to node q which is one hop closer to the root node. This effectively allows a node to cancel out its own contribution from the broadcast signal from the root node. The diffusion flow signal $\mathbf{z}_{k_k}^i$, used in the LMMSE estimation problem (6.1), now consists of $\mathbf{z}_{k\leftarrow r}^i$, rather than the signal defined in (6.6) with corresponding $\mathbf{G}_{k\leftarrow r}^i$ applied to the root node broadcast signal. Note, that while (6.6) and (6.8) are similar, they are only equivalent for the neighbors of the root node.

Despite the modified signal flow, the parameterization still lies in the solution space given in (6.7), i.e., $\mathbf{G}_{1\leftarrow 6} = \mathbf{G}_{46}\mathbf{G}_{74}\mathbf{G}_{r7}\mathbf{G}_{1r}$ and when setting $\mathbf{G}_{kq} = \overline{\mathbf{A}}_{kq}$, $\mathbf{G}_{1\leftarrow 6} = \overline{\mathbf{A}}_{46}\overline{\mathbf{A}}_{74}\overline{\mathbf{A}}_{r7}\overline{\mathbf{A}}_{1r} = \overline{\mathbf{A}}_{16}$. Simulations indicate that convergence is reached and that the $\mathbf{G}_{kq}^i$'s indeed converge to the $\overline{\mathbf{A}}_{kq}$'s using the modified signal flow. However, it should be noted that while the new parameterization still lies in the solution space, it is not equivalent to the original T-DANSE parameterization at every iteration i. Therefore, while the solutions still converge

to the same solution, the LMMSE at every iteration may be different between the two algorithms.

6.3.2 Reduction in input-output delay and transmission energy

Allowing the root node to broadcast a signal to all other nodes in the WSN has the added advantage of reducing the input-output delay that occurs from the signal flow. This is due to the fact that in the original T-DANSE scheme data must travel between every node in the network, whereas in the modified scheme the longest path data must travel is from the furthest leaf node to the root node plus the single hop broadcast from the root node to all other nodes in the network.

In using the modified scheme, half of the burden of transmission is shifted to the root node. A simplified energy model for transmission of signals is used that is based on the distance between nodes given as

$$E_{kq} = b \times d_{kq}^{\alpha} \tag{6.9}$$

where d_{kq} is the distance between nodes k and q, and $\alpha > 2$ is a constant [156,157] and b is the number of bits. $E_{kq,F}$ denotes the energy required for a node k to reach its single neighbor to which a fusion flow signal is transmitted and b_s represents the number of bits for signal transmission. $E_{kq,D}$ denotes the energy required to transmit the diffusion flow signal to all the neighboring nodes of node k (assuming the original T-DANSE algorithm is applied). Since a node transmits the same signal to all of its neighbors in the diffusion flow, the transmission energy required to broadcast is proportional to the largest distance between the node and its neighbors that receive the diffusion flow signal. $E_{kq,G}$ denotes the energy required to transmit a nodes $\mathbf{G}_{kq}$'s to its neighbors in the diffusion flow, which unlike the diffusion signal flow, must be transmitted to each individual neighbor that receives the diffusion flow signal and b_c represents the number of bits for the transmission. $E_{k,\text{total}}$ denotes the total energy used for transmission in the original T-DANSE algorithm, which is given as a summation of all of a nodes energy terms, i.e.,

$$E_{k,\text{total}} = E_{kl,F} + \sum_{q \in \mathcal{N}_k \setminus l} (E_{kq,G} + E_{kq,D}) \tag{6.10}$$

In the modified T-DANSE algorithm the right most term in (6.10), $E_{kq,D}$, vanishes as there is no signal flow in the diffusion flow for all non-root nodes.

6.4 Estimation errors of second-order statistics

In Chapter 2, it was shown that the MWF and the DANSE algorithm not only rely on the perfect estimation of the second-order statistics of the received signals, $\mathbf{R}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$, but also on the unobservable desired signals, $\mathbf{R}^i_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}$. In practice, however, the second-order statistics can only be imperfectly estimated by collected observations denoted as $\widetilde{\mathbf{R}}^i_{\tilde{\mathbf{y}}_k\tilde{\mathbf{y}}_k}$, $\widetilde{\mathbf{R}}^i_{\tilde{\mathbf{n}}_k\tilde{\mathbf{n}}_k}$ and $\widetilde{\mathbf{R}}^i_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}$. Notice that when a node k has a low SNR_{IN} at its sensors the desired signal correlation matrix is poorly estimated resulting in large relative errors on the entries of $\widetilde{\mathbf{R}}^i_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}$.

In [90] it was shown that in the single node case, when using imperfectly estimated correlation matrices and if the errors are considered small with respect to $\mathbf{R}_{\mathbf{nn}}$, the MWF becomes the optimal filter given by (2.26) and an extra bias term. It is therefore assumed that nodes which contain signals with larger SNR_{IN} tend to have smaller estimation errors, which leads to smaller errors in the optimal filter (2.26).

Due to the data-driven flow of the T-DANSE algorithm, if a root node is chosen with a low SNR_{IN} this not only places a bias term on its own estimation but also on the TFC-signals that the root node transmits to its neighbors. This in turn effects the estimation within those nodes, and their respective neighbors, etc., i.e., the error ripples throughout the whole network. Therefore since all data is fused through the root node it is important to have a good estimation of the signal statistics in the root node in order to decrease the added bias term as much as possible.

It is with these low SNR_{IN} nodes and the data-driven flow of T-DANSE in mind that not only a root node with a high SNR_{IN} is chosen, thereby reducing the errors spread throughout the network, but also to construct the entire tree topology based around this idea.

6.5 Weighted eigenvector centrality

In Section 6.4 it was assumed that nodes with low SNR_{IN} tend to have larger estimation errors in the estimated correlation matrices which increases the bias term on their calculated filters. In the T-DANSE algorithm this effect is compounded due to the fact that the signals are linearly compressed with these imperfect filters and fused with signals from other nodes. It is therefore assumed that nodes with high SNR_{IN} have lower estimation errors and are used to help prune an ad-hoc network to a tree topology with the goal of trying

to keep the highest SNR_{IN} toward the root node and nodes with low SNR_{IN} toward the leaf nodes.

It is first assumed, that before the commencement of the T-DANSE algorithm, the network is connected in an ad-hoc fashion, i.e., nodes only communicate with the neighbors that fall within their transmission range. The so-called adjacency matrix $\mathbb{A}$ is defined from the original ad-hoc topology as

$$[\mathbb{A}]_{k,q} = [\mathbb{A}]_{q,k} = \begin{cases} 1 & \text{if } k \text{ is connected to } q \\ 0 & \text{otherwise} \end{cases} \tag{6.11}$$

which categorizes the links of the individual nodes.

The first aim is to obtain a small input-output delay, i.e., the maximum distance between two nodes (in number of hops) should be as small as possible. This means that the tree should be highly branched, i.e., finding a root node which has many neighbors who - at their turn - also have many neighbors, etc. An interesting approach to identify such a potential root node is based on the so-called eigenvector centrality, which calculates the centrality of each node [158]. It is based on the principle that a node should get a high eigenvector centrality if it has many neighbors who - at their turn - also have a high eigenvector centrality. This leads to the following definition for the eigenvector centrality of node k [158]

$$c_k = \frac{1}{\lambda} \sum_{q \in N_k} c_q, \ \forall k \in \mathcal{K} \tag{6.12}$$

where λ is an arbitrary normalization factor, which can be re-written in terms of the adjacency matrix $\mathbb{A}$ as [158]

$$\lambda \mathbf{c} = \mathbb{A}\mathbf{c} \ . \tag{6.13}$$

It can be shown that if only non-negative coefficients are allowed in $\mathbf{c}$, then there is a unique solution to (6.12) given by the principle eigenvector of $\mathbb{A}$ that corresponds to the largest eigenvalue λ_{max}. It is important to note that this principle eigenvector can easily be computed in a distributed fashion (see [158]).

A root node is then selected which corresponds to the node with the highest eigenvector centrality. The ad-hoc network may then be pruned to a tree topology by connecting nodes based on their eigenvector centrality. This cycle is repeated until all nodes that were available in the ad-hoc network are included in the tree topology. Note that there is no assumption on the maximum number of connections per node, but this can easily be included as a constraint in the algorithm.

However, in terms of the estimation problem at hand, the eigenvector centrality of a node may not lead to a good choice for the root node, e.g., a node may

have a low SNR_{IN} and have many connections with other nodes which, if chosen as the root node, will affect the performance of the T-DANSE algorithm.

Therefore, a weighted adjacency matrix is used to not only take into account the eigenvector centrality of a node but also use the SNR_{IN} in order to select a good root node. This will also be used to prune the original ad-hoc network to a tree topology. Similar weighting strategies have been studied in [159] which rely only on the degree centrality of a node and SNR_{IN} was used for designing topologies in [160].

It is first assumed that the sensors of a node have a similar SNR_{IN} where the SNR_{IN} at node k is denoted as $\text{SNR}_{\text{IN},k}$. A new weighted eigenvector centrality is defined based on the following principle. A node should receive a high weighted eigenvector centrality if it has many neighbors with a high-SNR_{IN}, who also have many neighbors with high-SNR_{IN}, etc. Furthermore the weighted eigenvector of each node centrality should also be weighted with its own SNR_{IN} in order to avoid that a low-SNR_{IN} node surrounded by high-SNR_{IN} nodes be given a high weighted eigenvector centrality. This leads to the following definition of a weighted eigenvector centrality,

$$c_k = \frac{1}{\lambda}\text{SNR}_{\text{IN},k} \sum_{q \in N_k} c_q, \ \forall k \in \mathcal{K}. \tag{6.14}$$

This weighted eigenvector centrality can again be shown to have a unique solution given by the principle eigenvector of a weighted adjacency matrix $\mathbb{A}_{\text{SNR}}$ [158]. In order to construct the weighted adjacency matrix, $\mathbb{A}_{\text{SNR}}$, corresponding to (6.14), a diagonal matrix, $\mathcal{D}_{\text{SNR}_{\text{IN}}}$, is constructed where the diagonal entries are equal to the SNR_{IN} of the nodes

$$\mathcal{D}_{\text{SNR}_{\text{IN}}} = \begin{bmatrix} \text{SNR}_{\text{IN},1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \text{SNR}_{\text{IN},K} \end{bmatrix}. \tag{6.15}$$

The weighted adjacency matrix can then be given as

$$\mathbb{A}_{\text{SNR}} = \mathcal{D}_{\text{SNR}_{\text{IN}}}\mathbb{A}. \tag{6.16}$$

The root node is selected as the node with the highest weighted eigenvector centrality found by replacing $\mathbb{A}$ by $\mathbb{A}_{\text{SNR}}$ in (6.12), which can again be easily computed in a distributed fashion. The tree topology is then formed in a similar fashion where nodes are connected based on their weighted eigenvector centrality.

6.6 Simulations

6.6.1 Simulations using the modified root node broadcasting strategy

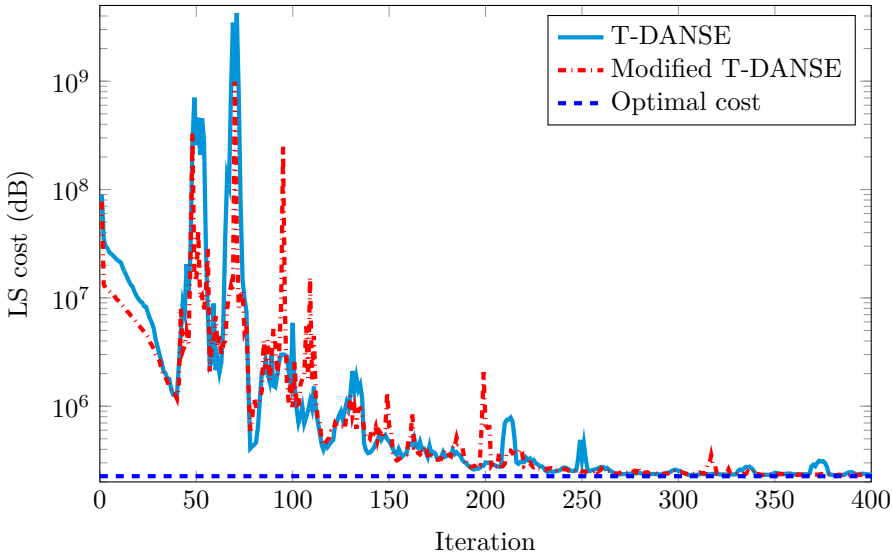
The simulations were implemented in batch mode as described in Section 4.4.1 where the MSE cost function at each node was replaced by the corresponding finite-length equivalent LS cost (4.35). Monte-Carlo simulations were performed on 100 sensing environments where there were $Q = 3$ source signals randomly distributed throughout the environment, and where each source signal consisted of $T = 10000$ samples i.i.d uniformly on the interval [-0.5 0.5]. The WSN consisted of 20 nodes deployed throughout the sensing environment. Each node had $M_k = 5$ sensors[2] that were spaced uniformly around a 10 cm radius from the center of the node. The coefficients of $\mathbf{A}_k$ were proportional to an attenuation factor of $\frac{1}{d}$ where d was the distance between the source and the sensor. There were also 5 spatially located white noise sources where the signals were generated with a similar process as the source signals. Spatially uncorrelated white noise, representative of process noise, that was half the power of the source signals, was added to each sensor.

An MST was found during each simulation using Prim's algorithm [161]. An $\alpha = 2.3$ was used to find the energy of each node during the fusion and diffusion flow of signals and a $b_s = 640$ kbits was used for each transmitted signal from a node and a $b_c = 576$ bits was used for the transmission of the $\mathbf{G}$ coefficients in (6.9).

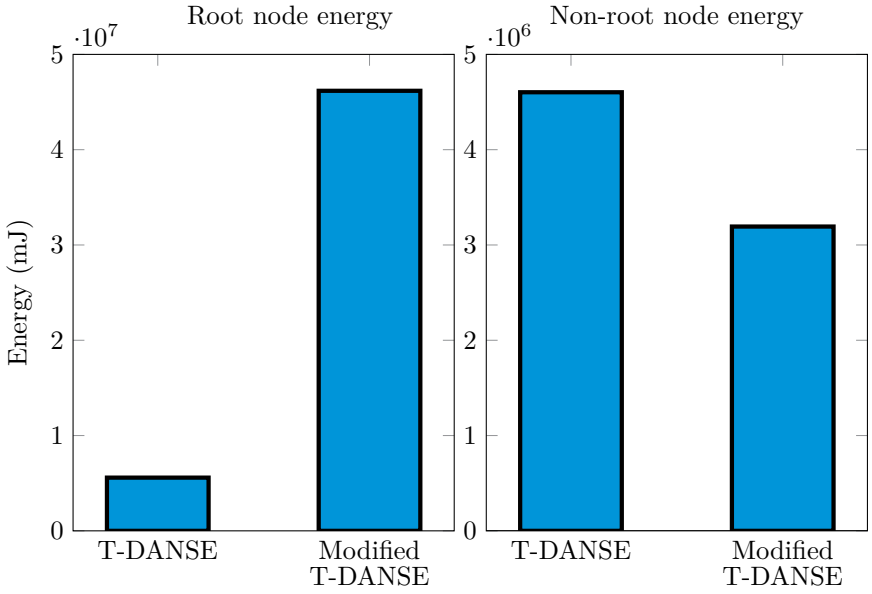
Figure 6.2a shows the convergence of the average LS cost, as defined in (4.35), of the sum of all of the nodes versus the number of iterations for both the original and modified T-DANSE algorithm compared to the average optimal LS cost. The modified algorithm is shown to have similar convergence properties of the original T-DANSE algorithm.

Figure 6.2b shows the average energy usage for transmission using (6.10) and the truncated version for non-root nodes for both the original and modified T-DANSE algorithm at the root and non-root nodes. The average energy is shown to be reduced for the non-root nodes using the modified broadcast strategy, and the transmission energy is shifted more to the root node which is in-line with the original goal of this method, as it is assumed the root node has a virtually infinite energy budget. It should also be noted that the average transmission energy is reduced in each node as given in Figure 6.2b, therefore

[2]It is reiterated that in order to guarantee convergence of the T-DANSE algorithm $M_k \geq Q$.



(a) Average LS cost of the sum of all nodes for the, T-DANSE, modified T-DANSE and optimal LS cost.



(b) Average transmission energy of the root node compared to the non-root nodes for the 100 sample environments.

Figure 6.2: Performance measures using the modified broadcasting strategy.

the total reduction in energy over the WSN for all non-root nodes is 19 times the reduction in transmission energy.

6.6.2 Topology construction

A WASN was simulated, as depicted in Figure 6.3a, where the nodes initially formed an ad-hoc network as in Figure 6.3b and consisted of 10 nodes (●) each with a communication radius of 2.5 m. Each node was assumed to have $M_k = 3$ microphones that were placed equidistantly around the center of the node at a radius of 1 cm. There was single target speech source (■), $Q = 1$, and a single spatially located white noise source (◆). Spatially uncorrelated white noise, representative of process noise, that was 10% of the average power of the noise source was added to each microphone.

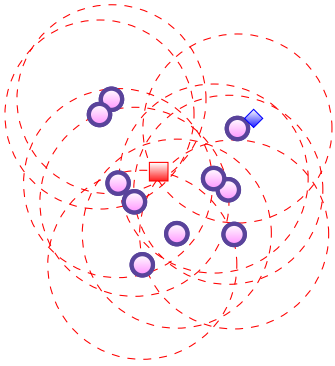
The room dimensions were 5m×5m×5m where a reflection coefficient of 0.2 was used for all surfaces where the microphone signals were generated using the image method [162]. A STFT block length of $L = 128$ was used with a sampling frequency of $f_s = 8000$ Hz. The estimation of the signal+noise (speech+noise in this case) and noise correlation matrices used an ideal VAD.

Before starting the T-DANSE algorithm the root node was selected as the node with either the highest eigenvector centrality (6.12) or SNR_{IN} weighted eigenvector centrality (6.14). The values for the different eigenvector centrality measures are given in Table 6.1 along with the corresponding SNR_{IN} of each reference microphone and the nodes selected as the root node are in bold. While the SNR_{IN} of the reference microphones is given in dB in Table 6.1 the values used in the weighted adjacency matrix (6.16) are not in dB. The ad-hoc network was then pruned to a tree, Figure 6.3c and Figure 6.3d, by using the method given in Section 6.5.

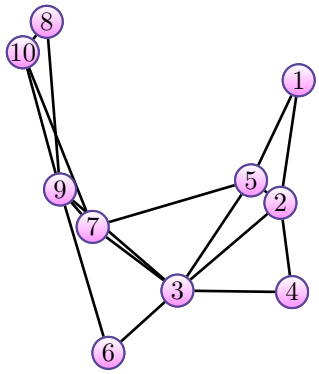
Notice that in using the weighted eigenvector centrality to prune the ad-hoc network, the output signal-to-noise ratio performance on all the nodes increases. In using the weighted eigenvector centrality the root node was selected as the node with the highest SNR_{IN} which also has higher SNR_{IN} nodes connected to it.

6.7 Conclusion

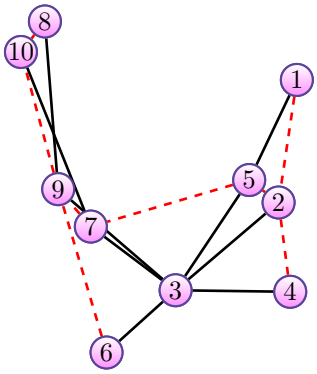
A modified T-DANSE algorithm was first proposed which was able to reduce the overall input-output delay of the system and shift the energy usage to that of the root node. This modified algorithm lends itself readily to tree topologies



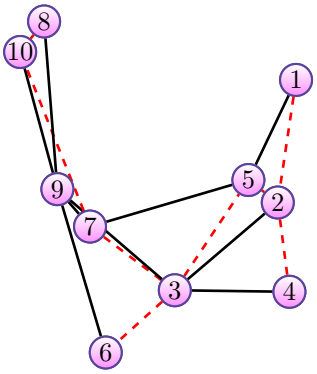
(a) Ad-hoc network with sensing radius per node with desired speech source and noise source.



(b) Ad-hoc network topology.



(c) Tree pruning using eigenvector centrality.



(d) Tree pruning using SNR weighted eigenvector centrality.

Figure 6.3: A WASN with a desired speech source (■), a spatially located white noise source (◆) and ten nodes (●) with corresponding network topologies.

Node	SNR_{IN}	eigenvector centrality	SNR_{OUT}	weighted eigenvector centrality	SNR_{OUT}
1	-14.63	0.161	3.26	0.001	3.64
2	-3.32	0.303	7.63	0.027	8.65
3	0.96	**0.473**	10.44	0.282	10.67
4	-3.33	0.273	6.15	0.027	7.23
5	-1.25	0.381	8.72	0.102	10.63
6	-0.84	0.291	5.53	0.180	7.70
7	5.16	0.403	12.68	0.641	13.43
8	-0.82	0.133	7.59	0.097	8.22
9	5.24	0.356	11.20	**0.648**	13.01
10	0.02	0.210	8.65	0.194	8.77

Table 6.1: SNR_{IN} (dB), eigenvector centrality, SNR_{OUT} (dB) using eigenvector centrality, SNR_{IN} weighted eigenvector centrality and SNR_{OUT} (dB) using SNR_{IN} weighted eigenvector centrality for each node. Root nodes for each centrality measure are given in bold.

in which a base station has larger broadcast abilities as well as larger energy resources. The proposed signal flow was demonstrated to be equivalent in terms of signal estimation to that of the original T-DANSE algorithm in that every node in the WSN was still able to converge to the centralized solution.

Next, due to the errors in the estimation of the second-order signal statistics, it was shown that the selection of the root node is of paramount importance. The method of using a so-called weighted eigenvector centrality was presented to not only select a good root node for the network but to also prune an already existing ad-hoc network to a tree topology. Simulations showed that using a weighted eigenvector centrality compared to that of an unweighted eigenvector centrality not only changed the pruned network topology but also improved the performance of the T-DANSE algorithm.

Part III

Topology-independent distributed signal estimation

Chapter 7

Topology-independent distributed adaptive node-specific signal estimation

7.1 Introduction

The DANSE algorithm has been implemented in the preceding chapters in fully connected, tree, and combinations thereof (mixed) topologies. However, the convergence of the DANSE algorithm in these topologies is directly dependent on the neighbors of a node. This implies that the nodes must communicate with the same neighbors during the entire estimation procedure, i.e., that the topology must remain static. In case the topology would change, the DANSE algorithm would then have to reconverge to a new set of parameters to again obtain the node-specific LMMSE signal estimates (cfr. Section 5.3.5).

It is with this in mind that the topology-independent DANSE (TI-DANSE) algorithm is introduced, where the aim is to let the nodes converge to a new set of estimator and compression parameters, which always yield the LMMSE signal estimates, independent of the underlying topology. Furthermore, the topology may even change in between iterations, without the need to let the algorithm re-converge to a new set of estimators and compression rules for the new topology. It accomplishes this by letting each node apply a transform to

the signal that it transmits and to the sum of the transformed linearly combined signals of the other nodes. In fact, as long as the WSN remains connected, i.e., there exists a path between any two nodes, the TI-DANSE algorithm is robust against changes in topology that can occur due to mobile nodes, link failure, etc. Furthermore, it can be shown that the convergence speed of the TI-DANSE algorithm is independent of the topology or changes therein.

In order to converge to the optimal solution, the nodes in the TI-DANSE must have access to the sum of all of the transformed linearly combined signals that are shared by the other nodes in the WSN. There are various means to calculate an in-network sum, e.g., relying on gossip or consensus based algorithms [64,103,104,163]. Although these methods are useful for the summation of fixed or slowly varying parameters, they become impractical for the summation of sensor signal observations that are collected at high sampling rates. Indeed, these methods typically need many iterations to converge to the solution, as well as multiple (re)-broadcasts of the intermediary summed variables. A method for this in-network summation is proposed which takes place on a tree topology that is formed from the set of available links[1]. The method can be described in a completely data-driven way, i.e., no upper layer coordination is needed between nodes. Since a tree topology is used, this in-network sum can be accomplished in a maximum of 2 transmissions per node. The tree used for the data-driven signal flow can be chosen randomly at every iteration which differs from the T-DANSE algorithm where the tree must remain static during the entire estimation.

This chapter is organized as follows. In Section 7.2 the TI-DANSE algorithm is first described in a fully connected topology for the sake of an easy exposition along with a proof of convergence. In Section 7.3 it is explained how the specific nature of the TI-DANSE algorithm allows it to be applied in any topology, relying on an in-network summation of signals. A method for this in-network summation is proposed that, at every iteration, partitions the WSN into a different tree followed by a data-driven signal flow. Numerical simulations are performed in Section 7.4 showing the convergence of the TI-DANSE algorithm compared to previous implementations of the DANSE algorithm. Finally, conclusions are given in Section 7.5.

[1]It is noted that there are numerous ad-hoc routing protocols to prune a tree from an ad-hoc topology based on different design constraints [164,165], however, this is not the main focus of the work.

7.2 TI-DANSE in a fully connected WSN

For the ease of exposition, the TI-DANSE algorithm is first described in a fully connected WSN, where each node is able to directly communicate with all of the other nodes in the WSN. In Section 7.3 the TI-DANSE algorithm is described in WSNs with any topology, where it will be shown that with a slight modification the same local linear combination rule that is introduced in this section can be also be applied.

Unlike the previous implementation of the DANSE algorithm in a fully connected WSN [53], where the linearly combined signals are placed in a stacked vector (2.32), node k now sums these signals which, for convenience, is denoted as $\boldsymbol{\eta}_{-k}^{i}$, where the subscript $-k$ indicates that there is no contribution from the signals of node k and is given as

$$\boldsymbol{\eta}_{-k}^{i} = \sum_{\forall q \in \mathcal{K} \setminus \{k\}} \mathbf{z}_{q}^{i} \,. \tag{7.1}$$

Node k places $\boldsymbol{\eta}_{-k}^{i}$ in a stacked vector with its own sensor signal observations represented as

$$\breve{\mathbf{y}}_{k}^{i} = \begin{bmatrix} \mathbf{y}_{k} \\ \boldsymbol{\eta}_{-k}^{i} \end{bmatrix} \,. \tag{7.2}$$

Since node k only has access to its own sensor signal observations $\mathbf{y}_k$ and sums the linearly combined signals of the other nodes, it can only control a specific part of its node-specific estimator $\mathbf{W}_k$, namely, $\mathbf{W}_{kk}$ (see (2.22)) and applies a transformation $\mathbf{G}_k \in \mathbb{C}^{Q \times Q}$ to $\boldsymbol{\eta}_{-k}^{i}$. The node finds its LMMSE estimator with respect to (7.2) at iteration i, $\widetilde{\mathbf{W}}_{k}^{i+1}$, by solving

$$\widetilde{\mathbf{W}}_{k}^{i+1} = \begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \\ \mathbf{G}_{k}^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk}, \mathbf{G}_k}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}_{kk}^{H} & \mathbf{G}_{k}^{H} \end{bmatrix} \breve{\mathbf{y}}_{k}^{i} \right|\right|_2^2 \right\} \,. \tag{7.3}$$

The solution to (7.3) is similar to (2.35) which is given as

$$\widetilde{\mathbf{W}}_{k}^{i+1} = \begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \\ \mathbf{G}_{k}^{i+1} \end{bmatrix} = (\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^{i})^{-1} \mathbf{R}_{\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k}^{i} \mathbf{E}_k \tag{7.4}$$

where $\mathbf{R}_{\tilde{\mathbf{y}}_k \tilde{\mathbf{y}}_k}^{i} = \mathcal{E}\{\tilde{\mathbf{y}}_{k}^{i} \tilde{\mathbf{y}}_{k}^{iH}\}$, $\mathbf{R}_{\tilde{\mathbf{x}}_k \tilde{\mathbf{x}}_k}^{i} = \mathcal{E}\{\tilde{\mathbf{x}}_{k}^{i} \tilde{\mathbf{x}}_{k}^{iH}\}$, $\tilde{\mathbf{x}}_{k}^{i}$ defined similarly to (7.2), and $\mathbf{E}_k$ is now an $(M_k + Q) \times Q$-dimensional matrix containing only zeros, except for a single entry equal to one in each column to select the desired

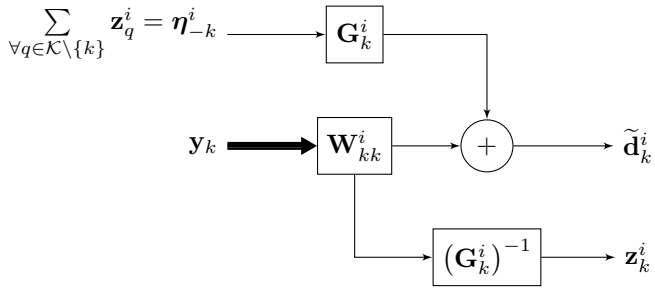


Figure 7.1: The proposed filtering and broadcast scheme for node k.

columns of $\mathbf{R}_{\tilde{\mathbf{x}}_k\tilde{\mathbf{x}}_k}^i$ that correspond to the node-specific desired signals of node k.

After node k has updated its local node-specific estimator, $\mathbf{W}_{kk}^i$ and $\mathbf{G}_k^i$, it updates the compression rule to generate its own broadcast signal, $\mathbf{z}_k^i$ by first linearly combining its sensor signal observations $\mathbf{y}_k$ by the part of the estimator applied to its own sensor signal observations, $\mathbf{W}_{kk}^i$, and then transforming this result by the inverse of the estimator applied to $\boldsymbol{\eta}_{-k}^i$, $\left(\mathbf{G}_k^i\right)^{-1}$, i.e.,

$$\mathbf{z}_k^i = \left(\mathbf{G}_k^i\right)^{-H} \mathbf{W}_{kk}^{iH} \mathbf{y}_k \tag{7.5}$$

$$= \mathbf{P}_k^{iH} \mathbf{y}_k \tag{7.6}$$

$$\mathbf{P}_k^i \triangleq \mathbf{W}_{kk}^i \left(\mathbf{G}_k^i\right)^{-1} . \tag{7.7}$$

This process is outlined in Table 7.1 and is depicted for node k in Figure 7.1. It is noted that, although $\mathbf{z}_k^i$ contains an iteration index, node k does not re-transmit the observations of $\mathbf{z}_k^i$ each time node k performs an update. An update at node k only means that *future* sensor observations of $\mathbf{y}_k$ will be compressed as $\mathbf{z}_k^{i+1}$ instead of $\mathbf{z}_k^i$ (using the new $\mathbf{P}_k^{i+1}$). The iterations of TI-DANSE are only performed on the estimator $\mathbf{W}_{kk}$, $\mathbf{G}_k$ and the compressor $\mathbf{P}_k$, but not on the signal observations.

The estimate of $\mathbf{d}_k$, denoted as $\widetilde{\mathbf{d}}_k^i$, at any node k and at any point in the iterative process is given as

$$\widetilde{\mathbf{d}}_k^{i+1} = \widetilde{\mathbf{W}}_k^{i+1\,H} \breve{\mathbf{y}}_k^i \tag{7.11}$$

$$= \mathbf{W}_{kk}^{i+1\,H} \mathbf{y}_k + \mathbf{G}_k^{i+1\,H} \boldsymbol{\eta}_{-k}^i . \tag{7.12}$$

Table 7.1: TI-DANSE in a fully connected WSN

1. Initialize $0 \to i$, $1 \to k$.
   Initialize $\mathbf{W}_{qq}^0$, $\mathbf{P}_q^0$ and $\mathbf{G}_q^0$ randomly, $\forall q \in \mathcal{K}$.
2. Node k updates its node-specific local parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_k$, by minimizing its LMMSE criterion based on the B most recent samples observations of its own local sensor signals and the summed received signals from the other $K-1$ neighboring nodes

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_k}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}_{kk}^H & \mathbf{G}_k^H \end{bmatrix} \breve{\mathbf{y}}_k^i \right|\right|_2^2 \right\} \tag{7.8}$$

   for which the solution is given by (7.4), and updates its compression matrix as

$$\mathbf{P}_k^{i+1} = \mathbf{W}_{kk}^{i+1} \left(\mathbf{G}_k^{i+1}\right)^{-1} . \tag{7.9}$$

   The other nodes do not update their node-specific local parameters:

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i, \mathbf{G}_q^{i+1} = \mathbf{G}_q^i \Rightarrow \mathbf{P}_q^{i+1} = \mathbf{P}_q^i . \tag{7.10}$$

3. $i+1 \to i$.
4. $(i \bmod K)+1 \to k$.
5. return to 2 .

7.2.1 Convergence and optimality

When node k solves its local LMMSE problem, it is essentially finding a parameterized version of the node-specific filter given in (2.26), i.e.,

$$\mathbf{W}_k = \begin{bmatrix} \mathbf{P}_1\mathbf{G}_k \\ \vdots \\ \mathbf{W}_{kk} \\ \vdots \\ \mathbf{P}_K\mathbf{G}_k \end{bmatrix} = \begin{bmatrix} \mathbf{W}_{11}\mathbf{G}_1^{-1}\mathbf{G}_k \\ \vdots \\ \mathbf{W}_{kk} \\ \vdots \\ \mathbf{W}_{KK}\mathbf{G}_K^{-1}\mathbf{G}_k \end{bmatrix} \tag{7.13}$$

where it can only manipulate the entries in $\mathbf{W}_{kk}$ and $\mathbf{G}_k$. The parameterization in (7.13) defines a solution space $\mathbf{W}_k, \forall k \in \mathcal{K}$ simultaneously. The next theorem shows that this solution space contains the network-wide LMMSE solutions (2.26) as a subset.

Theorem 3. *If (2.6) holds then the network-wide LMMSE solutions given in (2.26) lie in the parameterized solution space given by (7.13).*

Proof. Setting $\mathbf{G}_k = \overline{\mathbf{A}}_k^H$, $\forall k$, then

$$\mathbf{G}_q^{-1}\mathbf{G}_k = \left(\overline{\mathbf{A}}_q\right)^{-H} \overline{\mathbf{A}}_k^H \tag{7.14}$$

$$= \overline{\mathbf{A}}_{kq} \, . \tag{7.15}$$

Substituting (7.14) into (7.13) and setting $\mathbf{W}_{kk} = \widehat{\mathbf{W}}_{kk}$, $\forall k \in \mathcal{K}$ yields

$$\forall \, k \in \mathcal{K} \, : \, \widehat{\mathbf{W}}_k = \begin{bmatrix} \widehat{\mathbf{W}}_{11}\mathbf{G}_1^{-1}\mathbf{G}_k \\ \vdots \\ \widehat{\mathbf{W}}_{kk} \\ \vdots \\ \widehat{\mathbf{W}}_{KK}\mathbf{G}_K^{-1}\mathbf{G}_k \, . \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{W}}_{11}\overline{\mathbf{A}}_{k1} \\ \vdots \\ \widehat{\mathbf{W}}_{kk} \\ \vdots \\ \widehat{\mathbf{W}}_{KK}\overline{\mathbf{A}}_{kK} \end{bmatrix} \tag{7.16}$$

which, when comparing with (2.29), shows that the solution space defined by these parameterized filters contain the optimal network-wide LMMSE solution (2.26). □

Theorem 4. *Consider a WSN with a fully connected topology. If (2.6) holds, then for any initialization in step 1 of Table 7.1, the TI-DANSE algorithm obtains the node-specific signal estimates corresponding to the optimal LMMSE solution (2.26) for every node $k \in \mathcal{K}$.*

Proof. See Appendix E □

7.3 TI-DANSE without topology constraints

It is now described how the TI-DANSE algorithm presented in Section 7.2, can be implemented in a WSN without topology constraints. This is accomplished with a slight modification to the broadcast signal of a node (7.5), which now fuses its transformed linearly combined signal with the transformed linearly combined signal from its neighbors. This will allow for the broadcast signals of

the nodes to disperse through the WSN by means of an in-network summation. However, the broadcast TI-DANSE in a fully connected WSN will be shown to be a special case of the proposed modification.

In using the TI-DANSE algorithm, in order to converge to the optimal solution, each node needs to compute $\boldsymbol{\eta}_{-k}^i$, which can be found from a summed version of all $\mathbf{z}_q^i$'s as defined in (7.6), i.e., (7.1) can instead be given as

$$\boldsymbol{\eta}_{-k}^i = \sum_{\forall q \in \mathcal{K}} \mathbf{z}_q^i - \mathbf{z}_k^i \tag{7.17}$$

$$= \boldsymbol{\eta}^i - \mathbf{z}_k^i \tag{7.18}$$

where $\boldsymbol{\eta}^i = \sum_{\forall q \in \mathcal{K}} \mathbf{z}_q^i$.

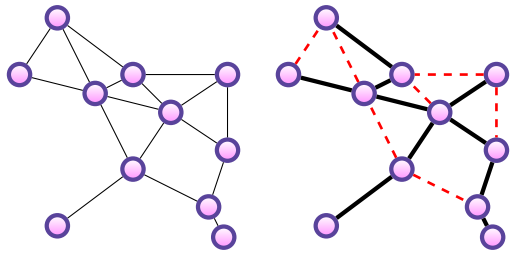
A method to perform this in-network summation is proposed via a data-driven signal flow that takes place in a tree topology[2]. While, at first this may seem like a restriction, this is done merely for the signal flow throughout the WSN. The node-specific estimation at each node can be performed independent of the actual topology, i.e., equivalently to Table 7.1, such that the algorithm becomes robust to link failures or dynamic topologies. If one of the branches of the tree would get disconnected, a new tree can be grown without the need to let the TI-DANSE algorithm reconverge (as would be the case in the original T-DANSE algorithm [54]). Also the sequential updating order does not need to follow a path though the WSN as in the case of the T-DANSE algorithm[3]. It is noted that a similar approach to an in-network summation was presented in [17] in order to provide a summed output signal to all nodes performing a distributed generalized sidelobe canceler technique.

7.3.1 Data-driven signal flow

It is assumed that when new sensor signal observations become available at the nodes, a tree is pruned from the ad-hoc topology in which these signal observations will be fused and disseminated. This tree can be formed with various constraints in mind, e.g., the weighted and unweighted eigenvector centrality presented in Chapter 6. An example is depicted in Figure 7.2 where the links not used in the tree topology can either be suppressed or removed. It is assumed that after the tree is formed, the nodes only communicate with

[2]It is noted that there are several distributed algorithms for tree formation which are dependent on the design constraints of the WSN [166–168].

[3]It is noted that, in most cases, the T-DANSE algorithm also converges if the nodes update in a round-robin fashion (based on empirical results), i.e., the specific updating order according to a path through the tree is sufficient, but not necessary.



(a) (b)

Figure 7.2: A WSN with an ad-hoc topology (a). The WSN with a pruned tree topology where the suppressed links are represented by dashed lines and where active links represented as bold solid lines (b).

their neighbors in the tree represented as $\mathcal{T}_k$ which is a subset of the total number of neighbors of the ad-hoc topology, i.e., $\mathcal{T}_k \subseteq \mathcal{N}_k$. Every time a tree is formed, one arbitrary node has to be assigned as the root node. The following data-driven signal flow is executed for each new block of observations collected at the sensors of each node:

1. Any non-root node k which has only a single neighbor can immediately fire and broadcast its compressed observations based on (7.5) which is repeated here for convenience,

$$\mathbf{z}_k^i = \mathbf{P}_k^{iH}\mathbf{y}_k \,. \tag{7.19}$$

   Any non-root node k with more than a single neighbor waits until it has received the compressed observations of all their neighbors except for a single neighbor that has yet to fire, say node q, and then computes the compressed observations

$$\mathbf{z}_k^i = \mathbf{P}_k^{iH}\mathbf{y}_k + \sum_{l\in\mathcal{T}_k\setminus\{q\}} \mathbf{z}_l^i \tag{7.20}$$

   and transmits this to node q. This process repeats at every node in the tree until the root node of the tree is reached.

2. Once the data-driven signal flow has reached the root node, say node r, it generates a block of observations of, $\boldsymbol{\eta}^i$, based on (7.20)

$$\boldsymbol{\eta}^i = \mathbf{P}_r^{iH}\mathbf{y}_r + \sum_{q\in\mathcal{T}_k} \mathbf{z}_q^i \tag{7.21}$$

which contains all of the linearly combined sensor signal observations in the WSN. This signal is now flooded through the WSN so that it reaches every node, where the nodes simply act as relays to pass $\boldsymbol{\eta}^i$ further through the tree.

7.3.2 Equivalence to fully connected topology

Now that the nodes have access to (7.21), they subtract out their locally linearly combined signal $\mathbf{P}_k^{iH}\mathbf{y}_k$, i.e.,

$$\boldsymbol{\eta}^i - \mathbf{P}_k^{iH}\mathbf{y}_k = \sum_{\forall q \in \mathcal{K}\backslash\{k\}} \mathbf{z}_q^i \tag{7.22}$$

$$= \boldsymbol{\eta}_{-k}^i \tag{7.23}$$

which is equivalent to (7.1). This equivalence means that even though the signal flow took place in a tree topology, it can be viewed from the same perspective as in the fully connected scenario. This shows that at each iteration of the algorithm, since the nodes have access to the same signals as in the fully connected case, the same node updating procedure can be applied as given in Table 7.1. The TI-DANSE in an ad-hoc topology is therefore equivalent at every iteration and hence the same convergence and optimality holds[4] as in Section 7.2.

A depiction of the node-specific estimation procedure and proposed signal flow for a non-root node is given in Figure 7.3. In Figure 7.3 it can be seen that the TI-DANSE in a fully connected topology can be considered a special case of the TI-DANSE in a tree topology. Each node in the fully connected topology broadcasts its signal as if it is a leaf node, i.e., $\mathbf{z}_k^i = \mathbf{P}_k^{iH}\mathbf{y}_k$ and also simultaneously acts as a root node due to the fact that in a fully connected topology $\mathcal{T}_k = \mathcal{N}_k = \mathcal{K}\backslash\{k\}$, $\forall k \in \mathcal{K}$ and therefore they have access to $\boldsymbol{\eta}_{-k}^i$.

Due to the proposed in-network summation the neighbors between nodes in the tree do not have to remain the same during the estimation procedure. The tree for the signal flow can be randomized at every iteration such that a new root node can be chosen and a different set of neighbors can be chosen, $\mathcal{T}_k^i \subseteq \mathcal{N}_k$. In fact, the signal flow does not need to occur on the same tree toward and away from the root node.

[4]It is noted that this is not the case when comparing the DANSE algorithm in a fully connected topology to the T-DANSE algorithm.

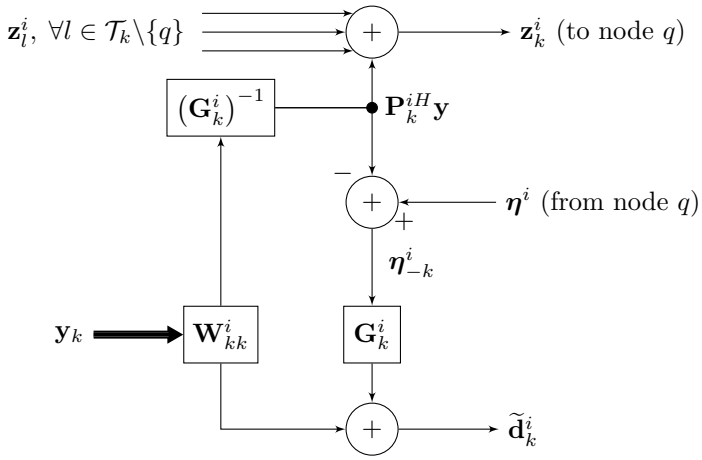


Figure 7.3: The signal flow of the TI-DANSE algorithm for a non-root node k. Note that for a leaf node, $\mathcal{T}_k \backslash \{q\} = \emptyset$ and the transmitted signal, $\mathbf{z}_k^i$, is equivalent to that of (7.5).

7.4 Simulations

Numerical simulations are first performed in a single sensing environment comparing the convergence of the DANSE, T-DANSE, TI-DANSE in a fully connected topology (TI-DANSE-FC) and TI-DANSE in a tree topology (TI-DANSE-T). The T-DANSE algorithm is then compared to the TI-DANSE-T algorithm when the links between nodes are chosen randomly during every iteration. Finally Monte-Carlo simulations are performed on 1000 sensing environments that are generated in the same fashion as in the single sensing environment to show a broader comparison between the convergence properties of the algorithms.

The simulations are implemented in batch mode as described in Section 4.4.1 where the MSE cost function at each node is replaced by the corresponding finite-length equivalent LS cost (4.35).

7.4.1 Single sensing environment

The WSN consisted of 15 nodes each deployed throughout a sensing environment. Each node had $M_k = 5$ sensors that were spaced uniformly around a 10 cm radius from the center of the node. There were $Q = 2$ sources distributed throughout the environment (■), where each source signal consisted of $T = 10000$ samples

which are i.i.d. uniformly on the interval of $[-0.5\ 0.5]$. The coefficients of $\mathbf{A}$ were proportional to an attenuation factor of $\frac{1}{d}$ where d was the distance between the source and the sensor. There were also 4 spatially located white noise sources (◆) where the signals were generated with a similar process as the source signals. Spatially uncorrelated white noise, representative of process noise, that was equal to 10% of the average noise power was also added to each sensor.

The given dimension of the desired signal space ($Q = 2$) and the number of sensors per node ($M_k = 5$) resulted in a compression[5] of the sensor signal observations of each node by a factor of $\frac{M_k}{Q} = \frac{5}{2}$.

The optimal centralized solution was first found at each node, assuming that each node transmitted its full M_k sensor signal observations to the other nodes in the WSN. The DANSE and TI-DANSE algorithms were then performed where it was assumed that the WSN was fully connected. Next, an ad-hoc WSN was generated by first setting the communication radius of each node to 0 and then expanding the communication radius of each node until the WSN was connected, i.e., every node was reachable by some set of links to every other node in the WSN. This ad-hoc WSN was then pruned to a MST using Prim's algorithm where the edge weights are equal to the Euclidean distance between nodes. The T-DANSE algorithm and the TI-DANSE algorithm were then performed on the resulting MST. A depiction of the WSN with the ad-hoc and MST links is given in Figure 7.4.

The total LS cost at every iteration, J^i_{Tot}, is found as the sum of individual LS costs, J_k, of the nodes using their current node-specific filters, $J_k(\mathbf{W}^i_k)$,

$$J^i_{Tot} = \sum_{k=1}^{K} J_k(\mathbf{W}^i_k)\ . \tag{7.24}$$

The convergence is compared to the number of iterations of the DANSE algorithms in Figure 7.5. The DANSE algorithm in the fully connected WSN converges to the optimal solution in the fewest number of iterations. The TI-DANSE algorithm in the fully connected and in the MST topologies converge identically due to the independence of the algorithm to the actual topology albeit slower than the other implementations of the DANSE algorithm.

The slower convergence of the TI-DANSE algorithm can be attributed to the number of available degrees of freedom that a node has when finding its LMMSE

[5]Note that the main compression is not due to the reduction of fusing observations of M_k-channel sensor signals into observations of a Q-channel signals per node, but due to the fact that there is in-network fusion of all the signals. Indeed, when compared to the relay case where all raw sensor data is routed through the network to a single FC (using multi-hop communication protocols), the reduction in communication is actually much higher than $\frac{M_k}{Q}$.

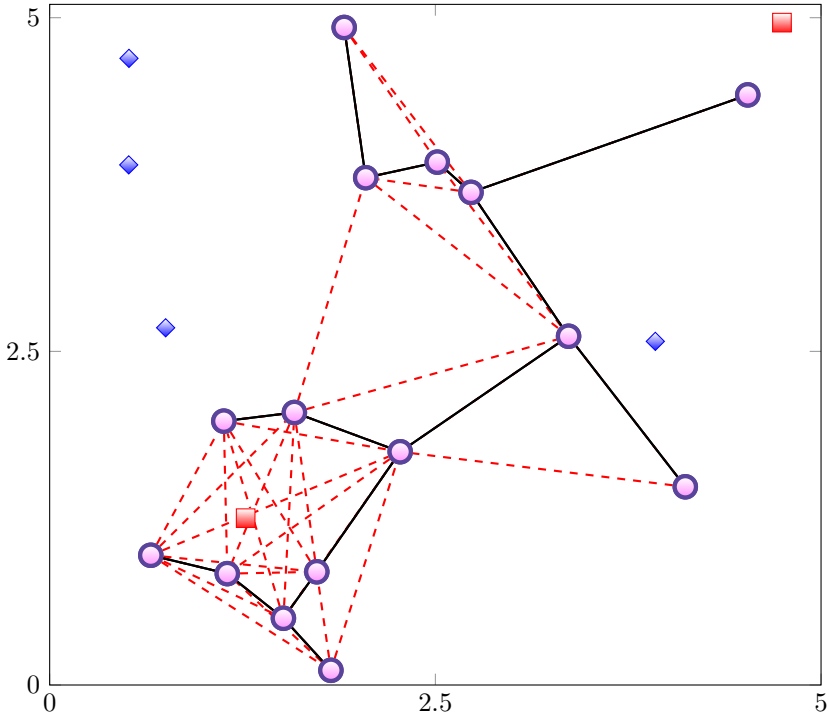


Figure 7.4: A simulated environment with 2 sources (■), 4 spatially correlated white noise sources (◆), and 15 nodes ($K = 15$ ●) each with $M_k = 5$ sensors. The dashed lines indicate the ad-hoc connections and the solid lines represent the MST formed using the Euclidean distance between nodes as edge weights.

estimator. In the fully connected case the DANSE algorithm has $M_k + (K-1)Q$ degrees of freedom per update and in a tree topology the T-DANSE algorithm has $M_k + |\mathcal{T}_k|Q$ degrees of freedom when performing an update at node k. The TI-DANSE algorithm has only $M_k + Q$ degrees of freedom when finding its its LMMSE estimator in (7.3). However, this is a trade-off to allow the DANSE algorithm to run in a topology-independent fashion yielding more robustness to topology changes such as link failures.

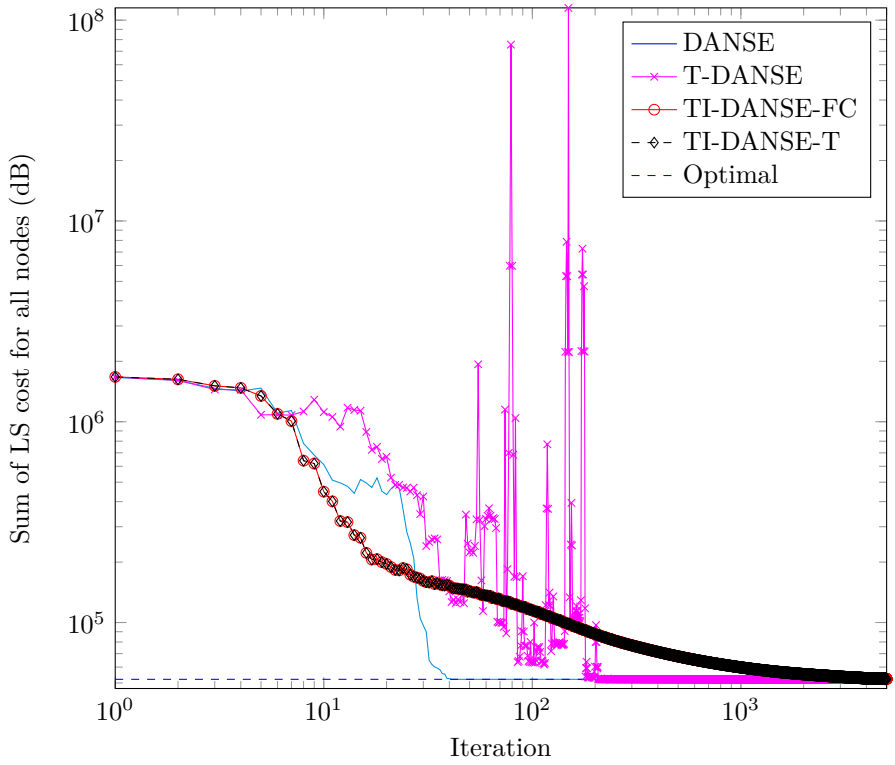


Figure 7.5: Cost of the DANSE, T-DANSE, TI-DANSE-FC and TI-DANSE-T algorithms versus the number of iterations.

7.4.2 Comparison of T-DANSE and TI-DANSE with dynamic connections

Simulations were then performed comparing the T-DANSE and TI-DANSE algorithm when the links in Figure 7.4 were chosen randomly at every iteration, i.e., $\mathcal{T}_k^i \subseteq \mathcal{N}_k$. This randomization was accomplished by randomizing the link weights, instead of using the Euclidean distances as in the single sensing environment, and again performing an MST algorithm based on the new weights. In order to apply the T-DANSE algorithm to the ad-hoc WSN, a different $\mathbf{G}_{kq}^i$ (see [54] for its definition) was initialized for every possible connection of a node k where $q \in \mathcal{N}_k$. The $\mathbf{G}_{kq}^i$'s were then updated depending on the current active links of a node during the corresponding iteration $q \in \mathcal{T}_k^i$ and they were kept fixed if the corresponding link disappeared (until it appeared again in a future

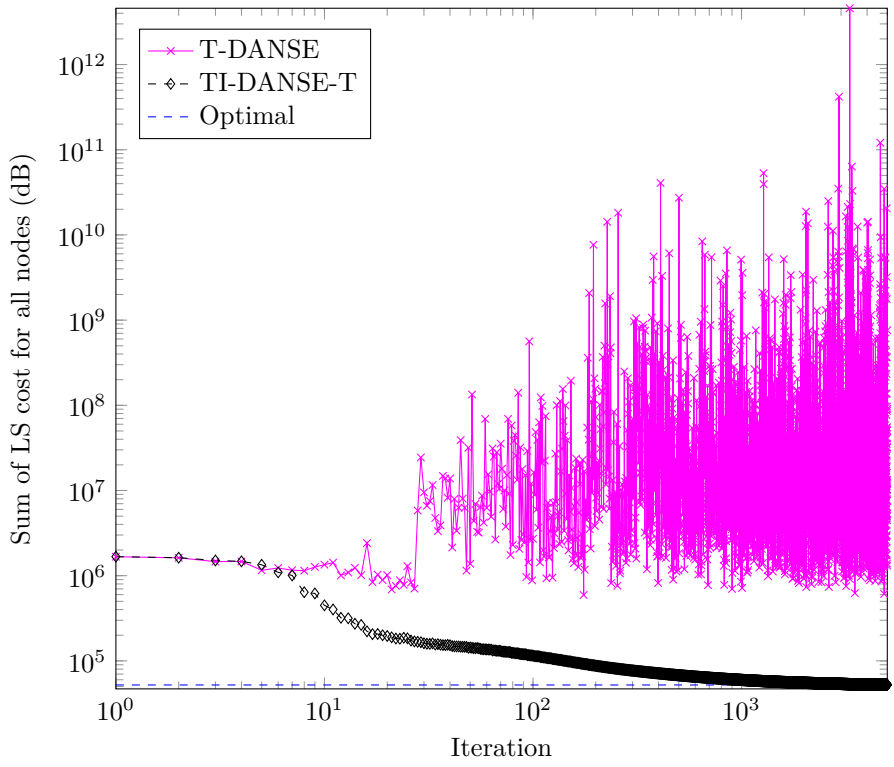


Figure 7.6: Cost of the T-DANSE and TI-DANSE-T algorithms with randomized links in the tree.

tree).

The TI-DANSE algorithm converges identically as in the static tree given in Figure 7.5. However, the T-DANSE algorithm fluctuates which is due to the fact that the updates are topology-dependent, and hence cannot converge if the topology changes in between each iteration.

7.4.3 Monte-Carlo simulations

Monte-Carlo simulations were performed on 1000 simulated environments that are initialized in the same to manner as the single scenario presented in Subsection 7.4.1. For every environment the sources, spatially located noise sources and 15 nodes were randomly placed. An MST was found using the

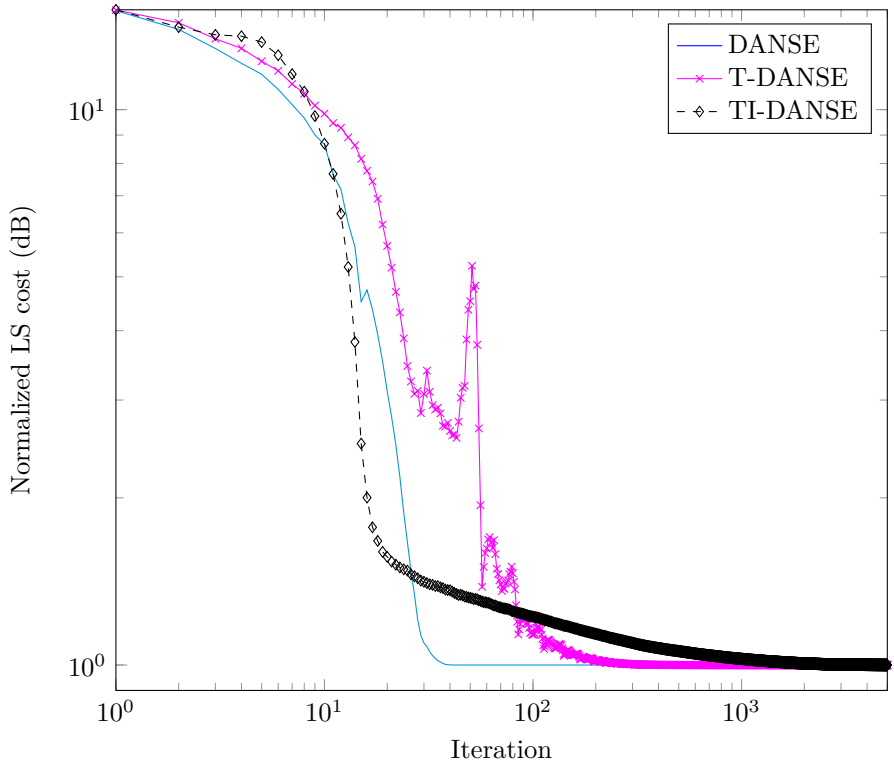


Figure 7.7: Normalized cost of the DANSE, T-DANSE and TI-DANSE algorithms versus the number of iterations averaged over 1000 Monte-Carlo runs.

Euclidean distances between nodes and kept constant for each Monte-Carlo run. The TI-DANSE-FC algorithm was not implemented as the convergence properties are identical to that of the TI-DANSE-T algorithm.

Due to the random nature of the generated signals, the optimal value of the summed LS cost was often different for each Monte-Carlo run. In order to account for this, for every Monte-Carlo run, the summed LS cost was normalized by the optimal value, i.e.,

$$\tilde{J}^i_{Tot} = \frac{\sum_{k=1}^{K} J_k(\mathbf{W}^i_k)}{\sum_{k=1}^{K} J_k(\widehat{\mathbf{W}}_k)} \,. \tag{7.25}$$

The normalized sum of the LS cost for all nodes was then averaged for the 1000 Monte-Carlo runs and shown in Figure 7.7. On average the convergence of the

algorithms is similar to that in the single sensing environment.

7.5 Conclusion

In this chapter the TI-DANSE algorithm was introduced where nodes estimated a node-specific desired signal in a WSN in a distributed fashion, and independent from the per-iteration topology of the WSN. In using the TI-DANSE, the nodes were shown to be able to arrive to their optimal node-specific solution, as if every node had transmitted all of its sensor signals observations to all other nodes. As opposed to the original T-DANSE algorithm, the TI-DANSE algorithm achieves these solutions using any of the available links in the WSN. While the TI-DANSE algorithm typically converges slower when compared to other variations of the DANSE algorithm, it offers the flexibility in being able to be implemented in any topology as long as an in-network signal summation is performed.

Chapter 8

Conclusion

This thesis has introduced many distributed signal processing algorithms for WSNs. The research objectives provided in the introduction included such topics as: the improvement in noise reduction performance when nodes shared multi-channel signals with one another, efficient allocation of communication bandwidth, prolonging the lifetime of the WSN, topology construction, scalability and self-healing. As stated in the introduction, many of the research objectives are complementary, therefore, the presented algorithms are often applicable to multiple objectives simultaneously. The research objectives were realized resulting in the following contributions.

Improvements in estimation performance with the addition of sensor signals

As a general motivation, it was argued in the introduction, that by increasing the amount of signals available to the system, this would lead in an increase in the noise reduction performance. Chapter 3 showed the benefit in noise reduction performance when adding a single sensor signal to a two node WASN. By adding this single sensor signal the output SNR of the devices was simultaneously improved with an increase in the binaural noise cue preservation.

A remarkable consequence of using the remote microphone is that while it did not contain any information pertaining to the binaural noise cues, it was still able to provide a benefit in terms of binaural noise cue preservation. Another important side observation was that even if the noise and speech came from similar directions, which limits the efficacy of the MWF, the remote microphone was still able to provide an improvement in the output SNR due to its increased spatial information. A novel method was also proposed to adapt the partial

noise estimation tuning parameter to further emphasize noise reduction or binaural noise cue preservation by exploiting trade-off between the performance measures.

These improvements from a single sensor, therefore, motivated an extension to including many more nodes in the sensing environment and led to the developments in subsequent chapters. However, the addition of nodes and the consequent formation of WSNs, led to novel design challenges.

Scalability and communication bandwidth

The nodes that formed the WSNs received observations from neighboring nodes over a wireless link in order to further improve their LMMSE estimates. As the number of nodes increases, though, this uses a significant amount of communication bandwidth. In order to reduce the communication bandwidth, the algorithms introduced in Chapters 4-7 all utilized some type of local compression, based on the node's local filters. This local compression was based on the dimension of the latent signal subspace shared by all nodes. Even though this dimensionality constraint was used, the nodes were still able to reach the same optimal solution as if they had access to all of the uncompressed signals in the WSN.

In Chapters 5-7 the scalability of the WSNs was improved by allowing the nodes to broadcast signals based on modified fusion rules, instead of acting like relays, in order to diffuse the signals through the WSN. The fusion rules presented allowed nodes to communicate with only their direct neighbors, which reduced the number of direct connections required when compared to fully connected WSNs.

In Chapter 5 these modified fusion rules were shown to be applicable in clustered and heterogeneous WSNs. This also allowed for nodes to be added to the WSN with a modest increase in the computational complexity and communication bandwidth. The scalability of the WSN was further improved upon in Chapter 7 by means of the TI-DANSE algorithm. In using the TI-DANSE algorithm the estimation performance was completely independent of the number of neighbors of a node, and relies on a single in-network summed signal. A method to compute this in-network summation was proposed that relied on a maximum of two transmissions per node and could be computed independently of the topology.

Extended network lifetime

Since the nodes communicate over a wireless link, a substantial amount of transmission energy may be needed, especially if there is a large distance between the nodes. However, the algorithms introduced in Chapters 4-7 all

resulted in a reduction in communication bandwidth, which directly led to a reduction in the energy the nodes utilized for communication. This reduction in bandwidth also reduced the computational burden at each node, since the number of signals that a node had to process was reduced, thereby further extending the lifetime of the nodes.

In Chapter 4 a means to facilitate node selection in a WSN was introduced by way of the utility. By only selecting a certain active set of nodes at any one time, the other nodes were able to conserve their energy thereby prolong the lifetime of the network. The utility was used to determine the increase in the LMMSE when removing a node and was used to determine the decrease in the LMMSE when one or more nodes were available for addition. The utility was then coupled with a greedy selection process and a MSE threshold to perform subset selection. While in a distributed scenario the utility was shown to represent a bound, compared to an exact value in the centralized scenario, simulations showed that the distributed and centralized node selection process had similar effects on the MSE.

In Chapter 6 a modification was proposed to the broadcast strategy of the T-DANSE algorithm which allowed for half of the communication burden to be shifted to the root node, while still being equivalent in terms of signal estimation performance. This relied on the broadcast of a single compressed signal from the root node to all other nodes in the WSN as well as the RFC strategy which allowed nodes to cancel out their local signals. Since the updates of the node's local parameters happen at a much slower rate than the broadcasting of the sensor signal observations, this resulted in a significant energy saving at the non-root nodes.

This modification can also directly be applied to the heterogeneous WSNs envisaged in the WSNs of Chapter 5, especially with the assumed increased energy capabilities of the cluster heads. By implementing the TI-DANSE algorithm with a similar strategy, i.e., by assuming the root node broadcasts to all nodes, this would essentially remove one of the transmissions needed from all non-root nodes and further extend the lifetime of the WSN.

Topology control and self-healing

The algorithms presented in this thesis all relied on accurate collection of the second-order statistics. However, due to inaccuracies in the second-order statistics, the way in which the topology was formed was shown in Chapter 6 to have an impact on the estimation performance. In fact, it was argued that the closer a node lies to the root node, the more accurate the estimation of the second-order statistics should be. A method for tree topology formation was considered in Chapter 6 that relied on either unweighted or weighted

eigenvector centrality. The weighted eigenvector centrality was based on the input SNR of the nodes, and assumed that nodes with a higher input SNR had more accurate estimations of their second-order statistics. The root node was first chosen based on this weighted eigenvector centrality, and a tree topology was formed around this node. It was shown that topologies formed using weighted eigenvector centrality improved the performance of the T-DANSE algorithm. Again, due to the tree substructures present in the mixed-topologies of Chapter 5 and the unavoidable errors in the estimation of the signal statistics, this weighted eigvenvector centrality can directly be applied to improve the estimation performance.

By allowing the WSN to be formed into sets of fully connected and tree topologies as in Chapter 5, many of the problems that are associated with restricting the WSN to a single topology were overcome. These substructures lessen the impact of changes in the topology, e.g., from mobile nodes, as only a relatively small number of these substructures is affected at any one time. This allows for efficient data flow through the WSN without the need for some type of upper level coordination. Finally, the TI-DANSE algorithm of Chapter 7 allowed for the estimation to occur in a completely topology independent fashion. This allows for the TI-DANSE algorithm to be implemented in WSNs with dynamic topologies which can be the byproduct of link failures.

Along with the material presented in this thesis there are other aspects which could serve as interesting directions for future research.

Suggestions for future research

The algorithms presented in this thesis were all considered to be implemented in the so-called application layer of the WSN. This, however, neglects other host and media layers of the node and WSN in general. These other layers also typically have constraints which are solved independently, i.e., without interaction with other layers, in order to optimize their performance. By implementing cross-layer solutions, which directly couple several layer constraints to a single estimation problem, or perform *joint*-estimation, further performance improvements may be achieved in terms of bandwidth and energy conservation.

The objective of many of the distributed signal estimation algorithms presented in this thesis was to reach the same optimal solutions as in the centralize case. This optimality requirement, however, forced diffusion of all of the signals of the WSN to every other node. A direct consequence of this was that a connectivity requirement was imposed on the topology of the WSN, i.e., a path must exist between every node in the WSN. However, this stringent requirement could be

relaxed to allow the formation of independent WSNs, which, while resulting in suboptimal performance, would allow for smaller processing delays. The optimality requirement may be able to be relaxed, much like bounds proposed for subset selection in Chapter 4, so that improvements in other areas of the algorithm may be possible, i.e., nodes could further reduce the dimensionality of their broadcast signals in order to reduce the communication bandwidth. This relaxation in optimality would also decrease the number of iterations needed until the algorithms were below a suboptimal threshold, which would be applicable in real-time scenarios.

Many of the algorithms were proposed with a general data model in mind, although specific applications were mentioned. However, when applying these algorithms to specific applications further unique aspects of the data model can be exploited to further benefit the algorithms. For example in WASNs, the perceptual information of the audio signals can be exploited to further increase the noise reduction performance. In terms of bandwidth reduction, further compression may be possible by applying audio coding techniques on the broadcast signals of the nodes.

Practical implementation aspects

While some of the chapters utilized energy models for communication, other aspects of the WSN and the nodes themselves were assumed to be ideal. While these assumptions aid in the ease of exposition for the conceptual aspects of the algorithms, in practical implementations many of these assumptions are violated which will lead to suboptimal performance.

Each of the nodes that compose these WSNs rely on a system clock to control the sampling and processing of the signals. However, due to temperature fluctuations, battery drain and other variables, these clocks inevitably differ from one node to another. This means that if no synchronization is present in the WSN, the samples sent and received by nodes will typically correspond to different time realizations of the signals. Since the proposed algorithms rely on a time-invariant coherence between the different microphone signals, a way to have time synchronization on a global scale should be implemented in the WSN.

With many devices communicating over an ever decreasing available wireless spectrum and due to imperfections in the communications channels and protocols, there will inevitably be some type of data or packet loss during the estimation. This will again have an impact on how the estimation of the signals in the WSN as it assumed that the data is collected perfectly at each node.

Therefore, making the algorithms robust to such problems makes them more adequate for practical implementations. However, in adding this robustness, the algorithms should still be computationally efficient as to not place too much burden on the nodes.

Privacy

In using many distributed nodes throughout a sensing environment, it has been shown that the noise reduction performance is improved. However, as in the case of WASNs, the nodes may not wish to broadcast information about their desired signals, i.e., they wish to keep a conversation private. Although, if nodes were still able to broadcast information pertaining to the noise, then this spatial information could still be used to improve the noise reduction performance in other nodes. Also the WSNs should be impervious to malicious attacks, which would allow outside devices to eavesdrop on the desired signals.

This raises privacy and security issues which are an ever present concern in today's digital world [169]. This security could be implemented in various stages of the WSN protocol stack. A privacy preserving distributed beamforming algorithm has recently been explored in [170, 171], however, it relies on computationally demanding encryption schemes which will affect the lifetime of the WSN. Therefore, if the algorithms could be designed with a privacy preserving aspect in mind, then perhaps the same level of network performance would be achievable at almost no discernible increase in computational complexity.

Appendix A

Appendix to Chapter 2

A.1 On the use of time-domain widely linear filtering for binaural speech enhancement

A binaural system that contains 2 nodes ($|\mathcal{K}| = 2$) is considered where the subscripts 1 and 2 represent the left and right device respectively. Each node contains an M_k microphone array where the total number of microphones of the system is given as M. The microphone signals are given similarly to (2.1) but now in the time domain as

$$y_{k,m}(t) = x_{k,m}(t) + n_{k,m}(t),\ k = 1, 2,\ m = 1 \dots M_k, \tag{A.1}$$

where $x_{k,m}(t)$ is the speech component and $n_{k,m}(t)$ is the additive noise component. An $M_k L$-dimensional stacked microphone signal vector, $\mathbf{y}_k \in \mathbb{R}^{M_k L}$, of each array is defined as

$$\mathbf{y}_k = \left[\mathbf{y}_{k,1}^T, \dots, \mathbf{y}_{k,M_k}^T\right]^T,\ k = 1, 2 \tag{A.2}$$

where $\mathbf{y}_{k,m} \in \mathbb{R}^L$ is defined as

$$\mathbf{y}_{k,m} = \begin{bmatrix} y_{k,m}(t) \\ \vdots \\ y_{k,m}(t - L + 1) \end{bmatrix},\ k = 1, 2 \tag{A.3}$$

and an ML-dimensional signal vector $\mathbf{y} \in \mathbb{R}^{ML}$ is defined as

$$\mathbf{y} = \left[\mathbf{y}_1^T\ \mathbf{y}_2^T\right]^T \tag{A.4}$$

where $\mathbf{x}$ and $\mathbf{n}$ are defined similarly. Each device is tasked with estimating a desired speech signal, d_k, which w.l.o.g., is assumed to be the speech component of the first microphone of each device, i.e., $d_k = x_{k,1}$.

A.2 Linear minimum variance distortionless response filter

It is known that MWF presented in Chapter 2 suppresses noise with the adverse effect of distorting the speech. In order to avoid this a minimum variance distortionless response filter (MVDR) filter can be used, which minimizes the output power while imposing a linear constraint to enforce a distortionless filter response.

It is noted that a true distortionless response can only be accomplished if the $\mathbf{R}_{\mathbf{xx}}$ has rank-1, which is why MVDR filtering is usually applied in the frequency domain where this rank-1 model holds as in the case of a single target speaker. In [100], MVDR filtering is proposed for time domain speech enhancement. However, since the rank-1 assumption then generally does not hold, this MVDR approach is not strictly distortionless, in the sense of delivering an undistorted speech signal, despite its name.

The MVDR cost function and linear constraint are given as [98]

$$\begin{aligned} \underset{\mathbf{w}_{k_{\mathrm{MVDR}}}}{\text{minimize}} \quad & \mathbf{w}_{k_{\mathrm{MVDR}}}^T \mathbf{R}_{\mathbf{yy}} \mathbf{w}_{k_{\mathrm{MVDR}}} \\ \text{subject to} \quad & \mathbf{w}_{k_{\mathrm{MVDR}}}^T \mathbf{a}_k = 1 \end{aligned} \tag{A.5}$$

where the steering vector $\mathbf{a}_k$ (which is the vector form of (2.4) where $Q = 1$) can be given as a scaled version of $\mathbf{R}_{\mathbf{xx}}\mathbf{e}_k$, i.e.,

$$\mathbf{a}_k = \frac{\mathbf{R}_{\mathbf{xx}}\mathbf{e}_k}{\mathbf{e}_k^T \mathbf{R}_{\mathbf{xx}}\mathbf{e}_k}. \tag{A.6}$$

The solution to (A.5) is given by [172]

$$\widehat{\mathbf{w}}_{k_{\mathrm{MVDR}}} = \frac{\mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{a}_k}{\mathbf{a}_k^T \mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{a}_k} \tag{A.7}$$

and using the definition of the steering vector (A.6) the MVDR filter is shown to be equivalent up to a scaling factor, α_k, to the MWF (2.26), i.e.,

$$\widehat{\mathbf{w}}_{k_{\mathrm{MVDR}}} = \alpha_k \mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{e}_k \tag{A.8}$$

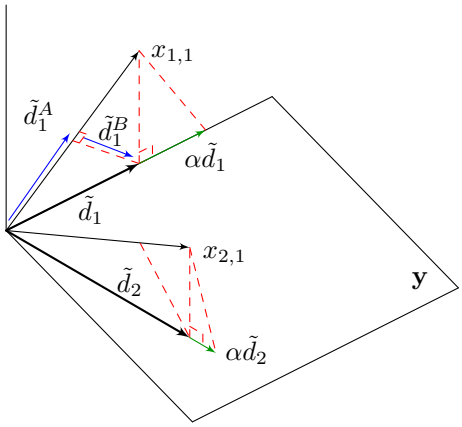


Figure A.1: Graphical representation between the linear MWF and linear MVDR solutions.

where

$$\alpha_k = \frac{\mathbf{e}_k^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_k}{\mathbf{e}_k^T \mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{R}_{\mathbf{xx}} \mathbf{e}_k}. \tag{A.9}$$

The estimated desired speech component at each array using the MVDR is then given as

$$\begin{aligned} \widetilde{d}_{k_{\mathrm{MVDR}}} &= \alpha_k \mathbf{e}_k^T \mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{y} \\ &= \alpha_k \widetilde{d}_k \, . \end{aligned} \tag{A.10}$$

The similarity between the MWF and MVDR is shown graphically in Figure A.1. The linear MWF projects $x_{1,1}$ and $x_{2,1}$ orthogonally into the $\mathbf{y}$-plane. The MWF estimate for $x_{1,1}$, denoted here again as $\widetilde{d}_1$, has a component $\widetilde{d}_1^A$ along $x_{1,1}$ and a component $\widetilde{d}_1^B$ orthogonal to $x_{1,1}$, where $\widetilde{d}_1 = \widetilde{d}_1^A + \widetilde{d}_1^B$. The MVDR is then a stretched version of the MWF solution (scaling with α_1) until the $\widetilde{d}_1^A$ component lands in $x_{1,1}$. The same process happens for the estimate of $x_{2,1}$.

Since the MWF is known to distort the speech, and since (A.8) is equivalent to an MWF (up to a fixed scaling), then indeed the time domain MVDR is also not distortionless. Basically, the linear constraint in (A.6) only ensures that the covariance between the MVDR filter output $\widetilde{d}_{k_{\mathrm{MVDR}}}$ and the desired signal $x_{k,1}$ is equal to the variance of $x_{k,1}$, i.e., $\mathcal{E}\{x_{k,1} d_{k_{\mathrm{MVDR}}}\} = \mathcal{E}\{x_{k,1}^2\}$ [100].

Despite their theoretical equivalence, an adaptive implementation of (2.26) or (A.8) based on block-processing may result in different output signals due to time variations in α_k such that $\mathcal{E}\{x_{k,1}d_{k_{\mathrm{MVDR}}}\} = \mathcal{E}\{x_{k,1}^2\}$ in each block.

Finally, it is noted that the MWF is known to preserve the binaural cues of the speech [16] (cfr. Chapter 3). Since the time domain MVDR filter corresponds to an MWF with an additional scaling α_k which is different for $k = 1, 2$ this will result in distortion of the binaural cues.

A.3 Widely linear filtering

The derivation of the linear MWF (2.26) and MVDR filter (A.8) also holds for complex-valued signals, but then the transpose operator ^{T} should be replaced by a transpose conjugation ^{H} in every equation. However, complex signals allow to also exploit the non-circularity of the signals if widely linear filtering techniques are used instead [96, 97, 173].

In [98–101], a complex signal vector is artificially constructed from the real signals received at both arrays,

$$\overline{\mathbf{y}} = \mathbf{y}_1 + j\mathbf{y}_2 \tag{A.11}$$

where $j = \sqrt{-1}$, to be able to apply WL filtering techniques. WL filtering then amounts to using the original linear filters on an augmented ML-dimensional signal vector, $\check{\mathbf{y}} = \mathbb{C}^{ML}$, defined as

$$\check{\mathbf{y}} = \begin{bmatrix} \overline{\mathbf{y}} \\ \overline{\mathbf{y}}^* \end{bmatrix} \tag{A.12}$$

where $\check{\mathbf{x}}$ and $\check{\mathbf{n}}$ are defined similarly. This augmented signal vector can easily be shown to be a transform of the original signal vector in (A.4),

$$\check{\mathbf{y}} = \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix} \mathbf{y}. \tag{A.13}$$

This transformation may then be used to show the equivalence between the estimated desired speech component found with the WL filters and the estimated desired speech components, ($k = 1, 2$), found with the linear filters when applied to real signals.

A.3.1 Widely linear multi-channel Wiener filter

The WL-MWF of (2.26) is given as

$$\check{\mathbf{w}} = \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1}\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}}\mathbf{e}_1, \tag{A.14}$$

where $\mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}} = \mathcal{E}\{\check{\mathbf{y}}\check{\mathbf{y}}^H\}$ and $\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} = \mathcal{E}\{\check{\mathbf{x}}\check{\mathbf{x}}^H\}$. The estimated desired speech component using (A.14) is then given as

$$\check{d} = \mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \check{\mathbf{y}} \tag{A.15}$$

which, using (A.13), can be expanded to

$$\check{d} = \mathbf{e}_1^T \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix} \mathbf{R}_{\mathbf{xx}} \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix}^H \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix}^{-H} \mathbf{R}_{\mathbf{yy}}^{-1} \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix} \mathbf{y} \,. \tag{A.16}$$

Simplifying (A.16), it can be seen that the estimated desired speech component for both devices is given by

$$\begin{aligned} \check{d} &= [1|j] \begin{bmatrix} \mathbf{e}_1^T \\ \mathbf{e}_2^T \end{bmatrix} \mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{y} \\ &= [1|j] \begin{bmatrix} \widetilde{d}_1 \\ \widetilde{d}_2 \end{bmatrix} . \end{aligned} \tag{A.17}$$

The WL-MWF output then fully corresponds to the linear MWF outputs.

A.3.2 Widely linear minimum variance distortionless response filter

In [98] a WL steering vector, $\check{\mathbf{a}}$, is used for the solution to the WL-MVDR filter given as

$$\check{\mathbf{a}} = \frac{\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1}{\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1} \,. \tag{A.18}$$

The WL-MVDR of [98] may then be given as

$$\check{\mathbf{w}}_{\mathrm{MVDR}} = \frac{\mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \check{\mathbf{a}}}{\check{\mathbf{a}}^H \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \check{\mathbf{a}}} \tag{A.19}$$

and using the definition of the response vector (A.18) the WL-MVDR filter is shown to be equivalent up to a scaling factor, $\check{\alpha}$, to the WL-MWF (A.14), i.e.,

$$\check{\mathbf{w}}_{\mathrm{MVDR}} = \check{\alpha} \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1 \tag{A.20}$$

where

$$\check{\alpha} = \frac{\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1}{\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1} \,. \tag{A.21}$$

The estimated desired speech component using (A.19) is then given as

$$\check{d}_{\text{MVDR}} = \check{\alpha}\mathbf{e}_1^T\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}}\mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1}\tilde{\mathbf{y}} \tag{A.22}$$

which uses the same expansion as in (A.16). Simplifying (A.22) with (A.16), the estimated desired speech component is given by

$$\begin{aligned}\check{d}_{\text{MVDR}} &= \check{\alpha}[1|j]\begin{bmatrix}\mathbf{e}_1^T\\ \mathbf{e}_1^T\end{bmatrix}\mathbf{R}_{\mathbf{xx}}\mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{y}\\ &= \check{\alpha}[1|j]\begin{bmatrix}\tilde{d}_1\\ \tilde{d}_2\end{bmatrix}\\ &= \check{\alpha}[1|j]\begin{bmatrix}\frac{1}{\alpha_1}\tilde{d}_{1_{\text{MVDR}}}\\ \frac{1}{\alpha_2}\tilde{d}_{2_{\text{MVDR}}}\end{bmatrix}.\end{aligned} \tag{A.23}$$

The WL-MVDR output then corresponds to the linear MVDR outputs up to a real-valued scaling with $\frac{\check{\alpha}}{\alpha_1}$ and $\frac{\check{\alpha}}{\alpha_2}$ (or to the linear MWF up to a joint scaling with $\check{\alpha}$).

It is noted that

$$\check{\alpha} = \frac{\mathbf{e}_1^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_1 + \mathbf{e}_2^T\mathbf{R}_{\mathbf{xx}}\mathbf{e}_2}{\mathbf{e}_1^T\mathbf{R}_{\mathbf{xx}}\mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{e}_1 + \mathbf{e}_2^T\mathbf{R}_{\mathbf{xx}}\mathbf{R}_{\mathbf{yy}}^{-1}\mathbf{R}_{\mathbf{xx}}\mathbf{e}_2} \tag{A.24}$$

and so $\check{\alpha}$ can also be computed from quantities available in the linear filtering approach hence both approaches yield the exact same information.

The WL-MWF gives the same estimates for $x_{1,1}$ and $x_{2,1}$ as in the linear case. However the WL-MVDR is obtained by equally stretching the MWF solutions by $\check{\alpha}$. If the $\widetilde{d}_1^A$ component is stretched into something longer than $x_{1,1}$, then the $\widetilde{d}_2^A$ is stretched into something shorter than $x_{2,1}$, and vice versa because the two stretched components, $\widetilde{d}_1^A$ and $\widetilde{d}_2^A$, now jointly satisfy one equation, i.e., $\check{\alpha}\mathcal{E}\{x_{1,1}d_1 + x_{2,1}d_2\} = \mathcal{E}\{x_{1,1}^2 + x_{2,1}^2\}$.

The similarity between the WL-MWF and WL-MVDR is shown graphically in Figure A.2 where an equal $\check{\alpha}$ is applied to both MWF solutions as compared to different scalings in Figure A.1.

A.4 Equivalence of the MVDR scaling factors under a Rank-1 model

Originally, the MVDR approach was designed for scenarios with a rank-1 model for $\mathbf{R}_{\mathbf{xx}}$ or $\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}}$. It is now shown that when such a rank-1 model is used for $\mathbf{R}_{\mathbf{xx}}$

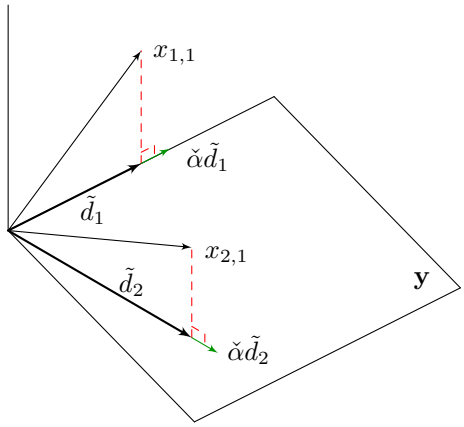


Figure A.2: Graphical representation between the WL-MWF and linear WL-MVDR solutions.

or $\mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}}$, the scaling factors for the linear MVDR and WL-MVDR are equivalent.

A.4.1 Linear MVDR scaling factor

The singular value decomposition (SVD) of the assumed rank-1 $\mathbf{R}_{\mathbf{xx}}$ matrix is given as

$$\mathbf{R}_{\mathbf{xx}} = \mathbf{U}_x \mathbf{\Sigma}_x \mathbf{V}_x^T \tag{A.25}$$

where $\mathbf{\Sigma}_x = diag(\sigma_x, 0, \ldots, 0)$ and the elements of $\mathbf{U}_x$ and $\mathbf{V}_x$ are given as $u_{x_{i,j}}$ and $v_{x_{i,j}}$ respectively. Using this SVD of the $\mathbf{R}_{\mathbf{xx}}$ matrix the numerator of (A.9) is shown to be

$$\mathbf{e}_1^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_1 = \mathbf{e}_1^T \mathbf{U}_x \mathbf{\Sigma}_x \mathbf{V}_x^H \mathbf{e}_1 = \sigma_x u_{x_{1,1}} v_{x_{1,1}} \ . \tag{A.26}$$

For the denominator of (A.9), the SVD is expressed as

$$\mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{R}_{\mathbf{xx}} = \mathbf{U}_x \mathbf{\Sigma}_{xy} \mathbf{V}_x^T \tag{A.27}$$

where $\mathbf{\Sigma}_{xy} = \mathbf{\Sigma}_x \mathbf{V}_x^T \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{U}_x \mathbf{\Sigma}_x$ which is another diagonal matrix with a single non-zero element, i.e., $diag(\sigma_{xy}, 0, \ldots, 0)$. Therefore

$$\begin{aligned} \mathbf{e}_1^T \mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{R}_{\mathbf{xx}} \mathbf{e}_1 &= \mathbf{e}_1^T \mathbf{U}_{xy} \mathbf{\Sigma}_{xy} \mathbf{V}_{xy}^T \mathbf{e}_1 \\ &= \sigma_{xy} u_{x_{1,1}} v_{x_{1,1}} \end{aligned} \tag{A.28}$$

and the scaling factor α_1 can be shown to be equal to

$$\alpha_1 = \frac{\mathbf{e}_1^T \mathbf{R}_{\mathbf{xx}} \mathbf{e}_1}{\mathbf{e}_1^T \mathbf{R}_{\mathbf{xx}} \mathbf{R}_{\mathbf{yy}}^{-1} \mathbf{R}_{\mathbf{xx}} \mathbf{e}_1} = \frac{\sigma_x}{\sigma_{xy}} \tag{A.29}$$

which is also the same for the $k = 2$ array, i.e., $\alpha_1 = \alpha_2$.

A.4.2 Widely linear MVDR scaling factor

In the WL case, the numerator of (A.21) using (A.13), and the SVD of $\mathbf{R}_{\mathbf{xx}}$, (A.25), is given as

$$\begin{aligned}\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1 &= \mathbf{e}_1^T \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix} \mathbf{R}_{\mathbf{xx}} \begin{bmatrix} \mathbf{I} & j\mathbf{I} \\ \mathbf{I} & -j\mathbf{I} \end{bmatrix}^T \mathbf{e}_1 \\ &= [1|j] \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix}^T \mathbf{R}_{\mathbf{xx}} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} [1|j]^H . \end{aligned} \tag{A.30}$$

However, since $\mathbf{R}_{\mathbf{xx}}$ is symmetric, the multiplication of the vectors $[1|j]$ and $[1|j]^H$ cancel out the off-diagonal terms while summing the diagonal terms. Therefore, (A.30) can be represented as

$$\begin{aligned}\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1 &= \mathrm{Tr}\left\{ \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix}^T \mathbf{R}_{\mathbf{xx}} \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} \right\} \\ &= \mathrm{Tr}\left\{ \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix}^T \mathbf{U}_x \mathbf{\Sigma}_x \mathbf{V}_x^T \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} \right\} \\ &= \sigma_x (u_{x_{1,1}} v_{x_{1,1}} + u_{x_{M_k L+1,1}} v_{x_{M_k L+1,1}}) . \end{aligned} \tag{A.31}$$

The denominator is expanded in a similar fashion as

$$\begin{aligned}\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1 &= \mathrm{Tr}\left\{ \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix}^T \mathbf{U}_x \mathbf{\Sigma}_{xy} \mathbf{V}_x^T \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 \end{bmatrix} \right\} \\ &= \sigma_{xy} (u_{x_{1,1}} v_{x_{1,1}} + u_{x_{M_k L+1,1}} v_{x_{M_k L+1,1}}) . \end{aligned} \tag{A.32}$$

The WL-MVDR scaling factor is therefore equivalent to the linear MVDR scaling factor (A.29),

$$\check{\alpha} = \frac{\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1}{\mathbf{e}_1^T \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{R}_{\check{\mathbf{y}}\check{\mathbf{y}}}^{-1} \mathbf{R}_{\check{\mathbf{x}}\check{\mathbf{x}}} \mathbf{e}_1} = \frac{\sigma_x}{\sigma_{xy}} . \tag{A.33}$$

A.5 Computational complexity

The WL filtering approach computes a single complex valued filter of length ML, while the linear filtering approach computes two real valued filters of

length ML (one for each desired signal d_k, where $k = 1, 2$). However complex arithmetic is 4 times more expensive than real arithmetic. As a result the WL filtering approach is actually twice as expensive compared to the linear filtering approach with no increase in performance. Furthermore, the two real-valued linear filters can share many of their computations (e.g., the inversion of $\mathbf{R}_{\mathbf{yy}}$), which makes the WL filtering approach actually more than twice as expensive compared to the linear filtering approach.

A.6 Conclusion

An equivalence was shown between the estimated desired speech components using time-domain linear and widely linear filters in binaural speech enhancement applications when only real signals are used. While the WL filters offer a novel way to represent the received real signals as a single complex signal there is no added benefit in terms of speech enhancement. However, by using an artificially constructed complex signal, the memory requirement of the system is increased as well as the computational complexity.

Appendix B

Appendix to Chapter 3

In this appendix, it is shown that adding a remote microphone signal results in $\lambda_e \geq \lambda$. It should be noted that while in [49] it was shown that when there is no coherent noise, i.e., $\mathbf{v} = 0$, $\lambda_e \geq \lambda$, this was not shown for the general case with the addition of a coherent noise source.

By definition (3.49), $\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}$ and $\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}$ are Hermitian and positive definite where the matrix pencil is given as

$$\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e} - \widetilde{\lambda_e}\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e} \ . \tag{B.1}$$

As $\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}$ is a rank-1 matrix, there will be only one nonzero eigenvalue, λ_e which is then in the same form as (3.54), repeated here for convenience

$$\mathrm{Tr}\{\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}^{-1}\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}\} = \lambda_e \ . \tag{B.2}$$

Let $\mathbf{C}$ be a $(M+1) \times M$-dimensional matrix defined as

$$\mathbf{C} = \begin{bmatrix} \mathbf{I} \\ \mathbf{0}^T \end{bmatrix} \tag{B.3}$$

where $\mathbf{I}$ is an $M \times M$-dimensional identity matrix and $\mathbf{0}$ is a M-dimensional row vector of all zeros. Using (B.3) the so-called *Rayleigh Quotient Pencil* is defined as

$$\mathbf{C}^T\mathbf{R}_{\mathbf{x}_e\mathbf{x}_e}\mathbf{C} - \widetilde{\lambda}\mathbf{C}^T\mathbf{R}_{\mathbf{n}_e\mathbf{n}_e}\mathbf{C} \tag{B.4}$$

$$= \mathbf{R}_{\mathbf{xx}} - \widetilde{\lambda}\mathbf{R}_{\mathbf{nn}} \ . \tag{B.5}$$

As $\mathbf{R_{xx}}$ is again a rank-1 matrix, it will have only one nonzero eigenvalue λ, which can be represented as

$$\mathrm{Tr}\{\mathbf{R_{nn}^{-1}R_{xx}}\} = \lambda \, . \tag{B.6}$$

Since $\mathbf{C} \in \mathbb{R}^{(M+1)\times M}$ with $\mathrm{rank}(\mathbf{C}) = M$ then, as shown in [174], by way of Cauchy interlacing inequalities the eigenvalue of the Rayleigh Quotient Pencil follows

$$\lambda \leq \lambda_e \, . \tag{B.7}$$

Appendix C

Appendix to Chapter 4

C.1 Relationship between LMMSE cost matrices between nodes

The cost function of node q evaluated with the node-specific linear LMMSE estimator, $\widehat{\mathbf{W}}_q$, is given as

$$J_q(\widehat{\mathbf{W}}_q) = \mathcal{E}\left\{\left|\left|\mathbf{d}_q - \widehat{\mathbf{W}}_q^H \mathbf{y}\right|\right|_2^2\right\}. \tag{C.1}$$

Using this optimal estimator, the LMMSE cost matrix for node q is given as

$$\widehat{\mathbf{J}}_q = \mathbf{R}_{\mathbf{d}_q\mathbf{d}_q} - \mathbf{R}_{\mathbf{y}\mathbf{d}_q}^H \widehat{\mathbf{W}}_q \tag{C.2}$$

where $\mathbf{R}_{\mathbf{d}_q\mathbf{d}_q} = \mathcal{E}\{\mathbf{d}_q\mathbf{d}_q^H\}$ is now the node q's desired signal correlation matrix.

The optimal estimators are related to one another by (2.29), which when used in (C.2) produces

$$\widehat{\mathbf{J}}_q = \mathbf{R}_{\mathbf{d}_q\mathbf{d}_q} - \mathbf{R}_{\mathbf{y}\mathbf{d}_q}^H \widehat{\mathbf{W}}_k \overline{\mathbf{A}}_{qk}. \tag{C.3}$$

By expanding the desired components of node q into its complex-valued steering matrix and source signal vector (C.3) is then given as

$$\widehat{\mathbf{J}}_q = \overline{\mathbf{A}}_q \mathcal{E}\{\mathbf{s}\mathbf{s}^H\}\overline{\mathbf{A}}_q^H - \overline{\mathbf{A}}_q \mathcal{E}\{\mathbf{s}\mathbf{y}^H\}\widehat{\mathbf{W}}_k \overline{\mathbf{A}}_{qk}. \tag{C.4}$$

Now the product of $\left(\overline{\mathbf{A}}_{qk}\right)^{-H}\widehat{\mathbf{J}}_q\left(\overline{\mathbf{A}}_{qk}\right)^{-1}$ is given as

$$\begin{aligned}\left(\overline{\mathbf{A}}_{qk}\right)^{-H}\widehat{\mathbf{J}}_q\left(\overline{\mathbf{A}}_{qk}\right)^{-1} &= \left(\overline{\mathbf{A}}_{qk}^{-H}\right)\overline{\mathbf{A}}_q\mathcal{E}\{\mathbf{s}\mathbf{s}^H\}\overline{\mathbf{A}}_q^H - \overline{\mathbf{A}}_q\mathcal{E}\{\mathbf{s}\mathbf{y}^H\}\widehat{\mathbf{W}}_k\overline{\mathbf{A}}_{qk}\left(\overline{\mathbf{A}}_{qk}\right)^{-1}\\ &= \overline{\mathbf{A}}_k\left(\overline{\mathbf{A}}_q\right)^{-1}\overline{\mathbf{A}}_q\mathcal{E}\{\mathbf{s}\mathbf{s}^H\}\overline{\mathbf{A}}_q^H\left(\overline{\mathbf{A}}_q\right)^{-H}\overline{\mathbf{A}}_k^H\\ &\quad - \overline{\mathbf{A}}_k\left(\overline{\mathbf{A}}_q\right)^{-1}\overline{\mathbf{A}}_q\mathcal{E}\{\mathbf{s}\mathbf{y}^H\}\widehat{\mathbf{W}}_k\\ &= \overline{\mathbf{A}}_k\mathcal{E}\{\mathbf{s}\mathbf{s}^H\}\overline{\mathbf{A}}_k^H - \overline{\mathbf{A}}_k\mathcal{E}\{\mathbf{s}\mathbf{y}^H\}\widehat{\mathbf{W}}_k\\ &= \widehat{\mathbf{J}}_k \qquad \text{(C.5)}\end{aligned}$$

which shows the equivalence stated in (4.4).

C.2 Calculation of optimal fall-back estimator and utility

For ease of exposition it is re-iterated the block partitioning of the inverse signal+noise correlation matrix as

$$\mathbf{R}_{\mathbf{yy}}^{-1} = \begin{bmatrix}\mathbf{S} & \mathbf{V}\\ \mathbf{V}^H & \mathbf{C}\end{bmatrix} \qquad \text{(C.6)}$$

and of the cross-correlation matrix $\mathbf{R}_{\mathbf{y}\mathbf{d}_k}$ as

$$\mathbf{R}_{\mathbf{y}\mathbf{d}_k} = \begin{bmatrix}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k}\\ \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}\end{bmatrix} \qquad \text{(C.7)}$$

where $\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_q\mathbf{d}_k^H\}$ and $\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} = \mathcal{E}\{\mathbf{y}_{-q}\mathbf{d}_k^H\}$. The estimator without node q's signals is given as

$$\widehat{\mathbf{W}}_{k_{-q}} = \mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}^{-1}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}. \qquad \text{(C.8)}$$

Now using the previously defined inverse signal+noise correlation matrix (4.9)

$$\begin{aligned}\widehat{\mathbf{W}}_{k_{-q}} &= (\mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V})\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}\\ &= \mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}. \qquad \text{(C.9)}\end{aligned}$$

The current estimator values are given as

$$\widehat{\mathbf{W}}_k = \begin{bmatrix}\widehat{\mathbf{W}}_{kq}\\ \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}\end{bmatrix} = \begin{bmatrix}\mathbf{S} & \mathbf{V}\\ \mathbf{V}^H & \mathbf{C}\end{bmatrix}\begin{bmatrix}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k}\\ \mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}\end{bmatrix}. \qquad \text{(C.10)}$$

Now using (C.10) and re-arranging the expression for $\widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}}$ results in

$$\widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} = \mathbf{V}^H \mathbf{R}_{\mathbf{y}_q \mathbf{d}_k} + \mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}$$

$$\mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} = \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H \mathbf{R}_{\mathbf{y}_q \mathbf{d}_k}. \tag{C.11}$$

Using (C.11) into (C.9) produces

$$\widehat{\mathbf{W}}_{k_{-q}} = \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H (\mathbf{R}_{\mathbf{y}_q \mathbf{d}_k} + \mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}). \tag{C.12}$$

Now using the fact that $\mathbf{S}^{-1}\widehat{\mathbf{W}}_{kq} = \mathbf{R}_{\mathbf{y}_q \mathbf{d}_k} + \mathbf{S}^{-1}\mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}$, (C.12) may be represented as (4.11), i.e.,

$$\widehat{\mathbf{W}}_{k_{-q}} = \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} - \mathbf{V}^H \mathbf{S}^{-1} \widehat{\mathbf{W}}_{kq}. \tag{C.13}$$

Now using the optimal fall-back estimator (4.11) the utility given in (4.12) can be calculated. Using (C.13) and the definition of the utility (4.6) gives

$$\begin{aligned} U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\mathbf{d}_k} \widehat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \widehat{\mathbf{W}}_{k_{-q}}\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}\mathbf{d}_k} \widehat{\mathbf{W}}_k - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \mathbf{V}^H \mathbf{S}^{-1} \widehat{\mathbf{W}}_{kq}\}. \end{aligned} \tag{C.14}$$

Block partitioning the first element in the trace of (C.14) gives

$$\mathbf{R}^H_{\mathbf{y}\mathbf{d}_k} \widehat{\mathbf{W}}_k = \begin{bmatrix} \mathbf{R}^H_{\mathbf{y}_q \mathbf{d}_k} & \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \end{bmatrix} \begin{bmatrix} \widehat{\mathbf{W}}_{kq} \\ \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} \end{bmatrix} \tag{C.15}$$

which expands the utility to

$$\begin{aligned} U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q \mathbf{d}_k} \widehat{\mathbf{W}}_{kq} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} \\ &\qquad - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \widehat{\mathbf{W}}_{k_{\mathbf{y}_{-q}}} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \mathbf{V}^H \mathbf{S}^{-1} \widehat{\mathbf{W}}_{kq}\} \\ &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q \mathbf{d}_k} \widehat{\mathbf{W}}_{kq} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \mathbf{V}^H \mathbf{S}^{-1} \widehat{\mathbf{W}}_{kq}\}. \end{aligned} \tag{C.16}$$

Now using (C.10) results in

$$\widehat{\mathbf{W}}_{kq} = \mathbf{S}\mathbf{R}_{\mathbf{y}_q \mathbf{d}_k} + \mathbf{V}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}$$

$$\mathbf{R}^H_{\mathbf{y}_q \mathbf{d}_k} = \widehat{\mathbf{W}}^H_{kq} \mathbf{S}^{-1} - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k} \mathbf{V}^H \mathbf{S}^{-1} \tag{C.17}$$

which when used with the previous result gives the utility

$$U_{k_{-q}} = \mathrm{Tr}\{\widehat{\mathbf{W}}^H_{kq} \mathbf{S}^{-1} \widehat{\mathbf{W}}_{kq}\}. \tag{C.18}$$

C.3 Utility for node addition

The inverse signal+noise correlation matrix is partitioned using the Woodbury identity and two intermediate variables, $\mathbf{\Gamma} = \mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{y}_q}$ and $\mathbf{\Sigma} = \mathbf{R}_{\mathbf{y}_q\mathbf{y}_q} - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{y}_q}\mathbf{\Gamma}$, as

$$\mathbf{R}^{-1}_{\mathbf{yy}} = \begin{bmatrix} \mathbf{S} & \mathbf{V} \\ \mathbf{V}^H & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{\Sigma}^{-1} & -\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H \\ -\mathbf{\Gamma}\mathbf{\Sigma}^{-1} & \mathbf{C} \end{bmatrix}. \tag{C.19}$$

Equation (4.17) is first expanded using the definition of the optimal estimator (4.10) to

$$U_{k_{-q}} = \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{yd}_k}\mathbf{R}^{-1}_{\mathbf{yy}}\mathbf{R}_{\mathbf{yd}_k} - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}\}. \tag{C.20}$$

Using the previously defined intermediate variables it is seen that

$$\begin{aligned}\mathbf{R}^H_{\mathbf{yd}_k}\mathbf{R}^{-1}_{\mathbf{yy}}\mathbf{R}_{\mathbf{yd}_k} = \mathbf{R}^H_{\mathbf{y}_q\mathbf{d}_k}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{R}^H_{\mathbf{y}_q\mathbf{d}_k}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} \\ - \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\mathbf{C}\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}.\end{aligned} \tag{C.21}$$

Now using (4.9) and (C.19), $\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}}$ is given as

$$\begin{aligned}\mathbf{R}^{-1}_{\mathbf{y}_{-q}\mathbf{y}_{-q}} &= \mathbf{C} - \mathbf{V}^H\mathbf{S}^{-1}\mathbf{V} \\ &= \mathbf{C} - \mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H.\end{aligned} \tag{C.22}$$

Now combining (C.21), (C.22) and (C.20) produces

$$\begin{aligned}U_{k_{-q}} &= \mathrm{Tr}\{\mathbf{R}^H_{\mathbf{y}_q\mathbf{d}_k}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{R}^H_{\mathbf{y}_q\mathbf{d}_k}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k} - \\ &\qquad \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} + \mathbf{R}^H_{\mathbf{y}_{-q}\mathbf{d}_k}\mathbf{\Gamma}\mathbf{\Sigma}^{-1}\mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k}\} \\ &= \mathrm{Tr}\{(\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k})^H\mathbf{\Sigma}^{-1}(\mathbf{R}_{\mathbf{y}_q\mathbf{d}_k} - \mathbf{\Gamma}^H\mathbf{R}_{\mathbf{y}_{-q}\mathbf{d}_k})\}.\end{aligned} \tag{C.23}$$

Appendix D

Appendix to Chapter 5

Proof of Theorem 2

This proof relies on additional concepts and lemmas that have been introduced in [54] which, due to their length and complexity, will only be re-iterated here without proof for convenience.

A partitioning, $\mathcal{P}$, is introduced of the nodes into non-overlapping subsets, i.e., $\mathcal{P} = (F, C_1, \ldots, C_t)$ where the specific node subsets, $F, C_1, \ldots, C_t$, will be defined later. The first subset, F, will be referred to as the free subset, for reasons explained in the sequel, and the other subsets will be referred to as the constrained subsets. It is assumed that for a certain subset of nodes, say S, the corresponding stacked version of all the $\mathbf{W}_{kq}$'s (see (2.23)) for which $q \in S$ is given as $\mathbf{W}_{k|S}$. A centralized updating scheme is now introduced that updates the entries of $\mathbf{W}_k$ in a sequence of alternating optimizations (AO). At each AO-step, i, a LMMSE optimization similar to (2.23) is performed where constraints are added that correspond to the constrained subsets of a given partitioning $\mathcal{P}^i$. For the set of partitionings $\mathcal{P}^i = (F^i, C_1^i, \ldots, C_{t_i}^i),\ \forall i \in \{1, \ldots, l\}$, the

corresponding AO update is then given as

$$\{\mathbf{W}_{k1}^{i+1},\ldots,\mathbf{W}_{kK}^{i+1},\mathbf{C}_1,\ldots,\mathbf{C}_{t_i}\} = \tag{D.1}$$

$$\underset{\mathbf{W}_{k1},\ldots,\mathbf{W}_{kK},\mathbf{C}_1,\ldots,\mathbf{C}_{t_i}}{\arg\min}\ \mathcal{E}\left\{\left\|\mathbf{d}_k-\sum_{l=1}^{K}\mathbf{W}_{kl}^H\mathbf{y}_l\right\|_2^2\right\} \tag{D.2}$$

$$\text{s.t.}\begin{cases}\mathbf{W}_{k|C_1^i}=\mathbf{W}_{k|C_1^i}^i\mathbf{C}_1\\ \vdots\\ \mathbf{W}_{k|C_{t_i}^i}=\mathbf{W}_{k|C_{t_i}^i}^i\mathbf{C}_{t_i}\end{cases}$$

where $\mathbf{C}_1,\ldots,\mathbf{C}_{t_i}$ are $Q\times Q$-dimensional matrices. Notice that the part of $\mathbf{W}$ corresponding to the constrained subset C, $\mathbf{W}_{k|C}$, is constrained to the subspace spanned by the columns of $\mathbf{W}_{k|C}^i$ from the previous iteration, whereas the part of $\mathbf{W}$ corresponding to the free subset F has no such constraint.

The MSE cost function of node k, defined in (2.25), using the full estimator, $\mathbf{W}_k$, is denoted as

$$J_k(\mathbf{W}_k)=J_k([\mathbf{W}_{k1}^T\ldots\mathbf{W}_{kK}^T]^T). \tag{D.3}$$

It is easy to see that for every AO-step defined in (D.1), $J_k(\mathbf{W}_k^i)$, will monotonically decrease, i.e.,

$$J_k(\mathbf{W}_k^{i+1})\le J_k(\mathbf{W}_k^i). \tag{D.4}$$

Lemma 1. *Consider an AO sequence defined by (D.1) where the constraints are defined by a given sequence of partitionings $(\mathcal{P}^i)_{i=0,\ldots,l-1}$ with $\mathcal{P}^i=(F^i,C_1^i,\ldots,C_{t_i}^i)$. This AO sequence is simultaneously applied to the cost function $J_k(\mathbf{W}_k)$ of node k to update $\mathbf{W}_k^i$ and to the cost function $J_q(\mathbf{W}_q)$ of node q to update $\mathbf{W}_q^i$, assuming that the two initial centralized estimators $\mathbf{W}_k^0$ and $\mathbf{W}_q^0$ are identical. If (2.6) is satisfied, then the following holds for any $i\in\{1,\ldots,l-1\}$:*

$$\mathbf{W}_k^i=\mathbf{W}_q^i\overline{\mathbf{A}}_{kq}, \tag{D.5}$$

with $\overline{\mathbf{A}}_{kq}=\left(\overline{\mathbf{A}}_q\right)^{-H}\overline{\mathbf{A}}_k^H$.

Proof. See the proof of Lemma A.1 in [54]. □

Lemma 1 shows that in a centralized scenario, where the node-specific estimators are updated according to the AO procedure defined in (D.1), then the resultant network-wide $\mathbf{W}_k$'s span the same Q-dimensional subspace.

Node k is now observed in a single tree topology, i.e., the tree is not yet connected in the envisaged heterogeneous topology. In a tree topology there is always a unique graph cut that cuts the edge (k,q) and no other edge in the network. It is assumed that the tree topology is cut along the edge (k,q) so that the tree is now divided into two complementary sets denoted as C_{kq}, which contains node k, and C_{qk}, which contains node q. The network-wide estimator for the estimation problem at node k is considered, $\mathbf{W}_k$, which is parameterized according to the tree topology (see also (5.6)). The part $\mathbf{W}_{k|C_{qk}}$ of $\mathbf{W}_k$ can only be manipulated by means of the $Q \times Q$-dimensional transformation matrix, $\mathbf{G}_{kq}$, that is applied in (5.8) to the signal received from node q.

This means that an update at node k corresponds to an AO step (D.1) in which C_{qk} is a constrained subset, since it can only transform $\mathbf{W}_{k|C_{qk}}$ through a $Q \times Q$-dimensional transformation matrix $\mathbf{G}_{kq}$. A similar constrained subset is now defined for all the other neighbors $q \in \mathcal{N}_k$ of node k. In this way, a one-to-one relationship between a node k and a specific partitioning can be defined, i.e.,

$$k \Leftrightarrow \mathcal{P} = (\{k\}; C_{\mathcal{N}_k}) \tag{D.6}$$

where the constrained subsets are given by $C_{\mathcal{N}_k} = \{C_{q_1 k}, C_{q_2 k}, \ldots, C_{q_t k}\}$ with $q_i \in \mathcal{N}_k$, i.e., $C_{\mathcal{N}_k}$ contains all the subsets C_{qk} for $q \in \mathcal{N}_k$. Considering the corresponding AO step in (D.1), node k is free to manipulate its own local estimator $\mathbf{W}_{kk}$ and its $\mathbf{G}_{kq}$ matrices in order to manipulate the constrained variables. Similar to (D.6), a one-to-one relationship between a path of length l through the network and a sequence of partitionings can be described that defines the constraints in l iterations of the AO sequence defined in (D.1), i.e.,

$$P \Leftrightarrow (\mathcal{P}^i)_{i=0\ldots l-1}. \tag{D.7}$$

Lemma 2. *Consider a network with a tree topology and a path $P_{k\to k}$ through this network with length $l-1$, that never passes through node k, except at the start and at the end. Consider the T-DANSE updating sequence equivalent to the l-step AO sequence defined by the partitioning $(\mathcal{P}^i)_{i=0,\ldots,l-1} \Leftrightarrow P_{k\to k}$. Then if (2.6) is satisfied the resulting $\mathbf{W}_k^l$, parameterized by (5.6), will be the same as if node k had access to all sensor signal observations and performed all optimizations in the AO sequence by itself with respect to its own cost function J_k.*

Proof. See [54]. □

A CH k of cluster n of the envisaged heterogeneous topology is now considered. Applying Lemma 2 to the CH, it can be seen that, after each node k in cluster n has performed at least one update (in the order defined by a path $P_{k\to k}$, starting

and ending at node k), it is as if node k itself has performed l subsequent AO iterations, resulting in a decrease of its local LMMSE cost function J_k (note that, by definition, each AO step yields a decrease in cost). The partitionings of the CHs are defined in the same manner as in (D.6) where now the constrained subset C_{qk} for $k, q \in \mathcal{K}_{CH}$, is defined as above, but based on a graph cut in the virtual tree where the links between any two CHs (except these of node k) are removed.

Based on Lemma 1, an important observation is now that another CH, say q, would obtain the same solution[1] (up to a $Q \times Q$-dimensional transformation) as if it were to apply the same AO sequence, but this time based on its own cost function J_q. Therefore if CH q would apply the appropriate $Q \times Q$-dimensional transformation to its received signal from node k, then it would still achieve the same reduction in cost even though the update occurred in another cluster in the network. This shows then, that each time a CH updates, this update decreases the cost function of the other $N-1$ CHs, as long as the other CH can adjust their $\mathbf{G}$-coefficients, which will eventually occur when the CHs update their local parameters.

Thus, $J_k(\mathbf{W}_k^i)$, $\forall k \in \mathcal{K}$, will monotonically decrease when evaluated after each update at node k. Since the cost is bounded below by zero, the cost at each node must converge to a fixed value, i.e.,

$$\lim_{i\to\infty} J_k(\mathbf{W}_k^i) = J_k^\infty \ . \tag{D.8}$$

It can then be shown, using similar strategies as in [54], that this convergence of the MSE values also results in convergence of the individual estimation parameters at each node and that $\lim_{i\to\infty} \mathbf{W}_k^i = \widehat{\mathbf{W}}_k$ (details omitted). □

[1] It is noted that Lemma 1 requires the initial network-wide estimators of node k and node q to be equal. This is achieved if all $\mathbf{G}_{q-q}, \forall q \in \mathcal{K}$ in the T-DANSE algorithm are initialized as an identity matrix. However, this is not a strict requirement to achieve convergence as shown in [54].

Appendix E

Appendix to Chapter 7

Proof of Theorem 4

A centralized case is first considered where it is assumed that all sensor signals are available at a node. For any arbitrary node ν, the centralized cost function as in (2.23) is defined as

$$J_\nu(\mathbf{W}_\nu) = \mathcal{E}\left\{\left|\left|\mathbf{d}_\nu - \mathbf{W}_\nu^H \mathbf{y}\right|\right|_2^2\right\} \tag{E.1}$$

where the filter $\mathbf{W}_\nu$ is given in the same form as (2.22) which can be thought of as a stacked matrix that contains a partitioned set of estimators that are applied to each nodes sensor signal observations, i.e.,

$$\mathbf{W}_\nu^T = \left[\ \mathbf{W}_{\nu 1}^T, \ldots,\ \mathbf{W}_{\nu K}^T\ \right]^T \tag{E.2}$$

where $\mathbf{W}_{\nu 1}$ is applied to the first nodes sensor signal observations up to $\mathbf{W}_{\nu K}$ for node K.

A cost function that is similar to (E.1) is also introduced where now the desired signal $\mathbf{d}_\nu$ is transformed by an arbitrary $Q \times Q$-dimensional matrix, $\mathbf{Z}^i$, which yields

$$J_z^i(\mathbf{W}_z) = \mathcal{E}\left\{\left|\left|\mathbf{Z}^{i\,H}\mathbf{d}_\nu - \mathbf{W}_z^H \mathbf{y}\right|\right|_2^2\right\} \tag{E.3}$$

where $\mathbf{W}_z$ is partitioned in that same manner as (E.2). The iteration index i used here does not refer to the TI-DANSE iterations, but can refer to any generic iterative algorithm.

A sequence of transform matrices that are used to transform the node-specific desired signal in (E.3) at every iteration i is defined as

$$(\mathbf{Z}^i)_{i\in\mathbb{N}} = (\mathbf{Z}^0, \mathbf{Z}^1, \ldots) \tag{E.4}$$

which implies that at every iteration a different $\mathbf{Z}^i$ is applied to $\mathbf{d}_\nu$.

A sequence of AOs is now considered where at each iteration i the LMMSE optimization corresponding to (E.1) is performed but with constraints added to all partitions of (E.2) except one, say $\mathbf{W}_{\nu k}$ where k changes in each iteration. The constraints ensure that the columns of $\mathbf{W}_{\nu,-k}$ remain in the current column space of $\mathbf{W}^i_{\nu,-k}$, where the notation $\mathbf{X}_{-k}$ is used to denote a node-by-node stacked matrix as in (E.2), but where the partition $\mathbf{X}_k$ corresponding to node k is removed. The formal description of this AO process is defined as AO1 in the left column of Table E.1.

A similar AO procedure AO2 can be described for the more general cost function in (E.3) where the same partitioning is applied to the optimization variable $\mathbf{W}_z$, but where the sequence of transformations (E.4) is used to define a new cost function (E.3) in each iteration of the AO procedure, as formalized in the right column of Table E.1. Note that, if $\mathbf{Z}^i = \mathbf{I}$, $\forall i$, where $\mathbf{I}$ is an identity matrix of appropriate size, AO1 and AO2 become equivalent.

AO1 will generate the AO sequence

$$(\mathbf{W}^i_\nu)_{i\in\mathbb{N}} = (\mathbf{W}^0_\nu, \mathbf{W}^1_\nu, \ldots)\,. \tag{E.5}$$

It is then clear from the definition of the AO procedure that J_ν will monotonically decrease at each step, i.e.,

$$J_\nu(\mathbf{W}^{i+1}_\nu) \leq J_\nu(\mathbf{W}^i_\nu)\,. \tag{E.6}$$

This is because the current estimate $\mathbf{W}^i_\nu$ itself is always in the constraint set of the optimization problem in step 2, and hence the updated version should yield an MSE that is at least as good as the previous estimator. In fact, AO1 is actually a relaxed version of a Gauss-Seidel block-coordinate descent (GSBCD) method, where the latter puts an additional constraint that $\mathbf{\Psi}_\nu = \mathbf{I}$, i.e., all entries in $\mathbf{W}^i_{\nu,-k}$ remain fixed. It is known that such a GSBCD method (and hence also its relaxed version) converges to a stationary point of the objective function if this objective function is convex [175]. Therefore, it follows that

$$\lim_{i\to\infty} \mathbf{W}^i_\nu = \widehat{\mathbf{W}}_\nu \tag{E.7}$$

where $\widehat{\mathbf{W}}_\nu$ is the global minimum of J_ν, corresponding to the network-wide LMMSE solution of node ν.

Before continuing the proof, the following result is necessary

Table E.1: Alternating optimization procedure

AO1, using (E.1)	AO2, using (E.3)
	A pre-defined sequence of $Q \times Q$-dimensional matrices $(\mathbf{Z}^i)_{i\in\mathbb{N}}$ as given in (E.4) is assumed.
1. Initialize $i \to 0$, $1 \to k$. Initialize $\mathbf{W}_\nu^0$ randomly .	1. Initialize $i \to 0$, $1 \to k$. Initialize $\mathbf{W}_z^0$ randomly .
2. $(\widehat{\mathbf{X}}_\nu,\ \widehat{\mathbf{\Psi}}_\nu) = \underset{\mathbf{X}_\nu,\mathbf{\Psi}_\nu}{\arg\min} J_\nu(\mathbf{X}_\nu)$ s.t. $\mathbf{X}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \mathbf{\Psi}_\nu$.	2. $(\widehat{\mathbf{X}}_z,\ \widehat{\mathbf{\Psi}}_z) = \underset{\mathbf{X}_z,\mathbf{\Psi}_z}{\arg\min} J_z^{i+1}(\mathbf{X}_z)$ s.t. $\mathbf{X}_{z,-k} = \mathbf{W}_{z,-k}^i \mathbf{\Psi}_z$.
3. $\widehat{\mathbf{X}}_\nu \to \mathbf{W}_\nu^{i+1}$.	3. $\widehat{\mathbf{X}}_z \to \mathbf{W}_z^{i+1}$.
4. $i+1 \to i$.	4. $i+1 \to i$.
5. $(i \bmod K)+1, \to k$.	5. $(i \bmod K)+1, \to k$.
6. return to 2 .	6. return to 2 .

Lemma 3. *Consider the two AO procedures AO1 and AO2 as given in Table E.1, and assume that AO1 and AO2 are initialized in step 1 such that* $\mathbf{W}_z^0 = \mathbf{W}_\nu^0 \mathbf{Z}^0$. *Then AO2 will produce the AO sequence*

$$(\mathbf{W}_z^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^i \mathbf{Z}^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^0 \mathbf{Z}^0, \mathbf{W}_\nu^1 \mathbf{Z}^1, \ldots) \tag{E.8}$$

i.e., the sequences $(\mathbf{W}_\nu^i)_{i\in\mathbb{N}}$ *and* $(\mathbf{W}_z^i)_{i\in\mathbb{N}}$ *are equivalent up to a* $Q \times Q$*-dimensional transformation for every iteration* i.

Proof. The constrained optimization problems in AO1 and AO2 in Table E.1 can be transformed into unconstrained optimization problems by using the substitutions

$$\mathbf{X}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \mathbf{\Psi}_\nu \tag{E.9}$$

$$\mathbf{X}_{z,-k} = \mathbf{W}_{z,-k}^i \mathbf{\Psi}_z \tag{E.10}$$

inside the objective function $J_\nu(\mathbf{X}_\nu)$ and $J_z^{i+1}(\mathbf{X}_z)$, yielding:

$$\begin{bmatrix} \widehat{\mathbf{X}}_{\nu,k} \\ \widehat{\mathbf{\Psi}}_\nu \end{bmatrix} = \underset{\mathbf{X}_{\nu,k},\mathbf{\Psi}_\nu}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{d}_\nu - \begin{bmatrix} \mathbf{X}_{\nu,k}^H & \mathbf{\Psi}_\nu^H \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H}\, \mathbf{y}_{-k} \end{bmatrix} \right|\right|_2^2 \right\} \tag{E.11}$$

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{\Psi}}_z \end{bmatrix} = \underset{\mathbf{X}_{z,k},\mathbf{\Psi}_z}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{Z}^{i+1\,H}\mathbf{d}_\nu - \begin{bmatrix} \mathbf{X}_{z,k}^H & \mathbf{\Psi}_z^H \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{z,-k}^{i\,H}\, \mathbf{y}_{-k} \end{bmatrix} \right|\right|_2^2 \right\} \tag{E.12}$$

with $\mathbf{y}_{-k}$ denoting the vector $\mathbf{y}$ with the vector $\mathbf{y}_k$ removed. The full matrix $\widehat{\mathbf{X}}_\nu^{i+1}$ is eventually found by including $\widehat{\mathbf{X}}_{\nu,k}$ into $\widehat{\mathbf{X}}_{\nu,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{\Psi}}_\nu$ again at the correct place (corresponding to node k), and similarly for $\widehat{\mathbf{X}}_z^{i+1}$.

This lemma is proved using the following inductive argument. To this end, it is first assumed that the lemma holds up to iteration i, i.e., $\mathbf{W}_z^i = \mathbf{W}_\nu^i \mathbf{Z}^i$, from which it will be shown that also $\mathbf{W}_z^{i+1} = \mathbf{W}_\nu^{i+1}\mathbf{Z}^{i+1}$. If it is indeed assumed that $\mathbf{W}_z^i = \mathbf{W}_\nu^i \mathbf{Z}^i$, then (E.12) can be rewritten as

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{\Psi}}_z \end{bmatrix} = \underset{\mathbf{X}_{z,k},\mathbf{\Psi}_z}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{Z}^{i+1\,H}\mathbf{d}_\nu - \begin{bmatrix} \mathbf{X}_{z,k}^H & \mathbf{\Psi}_z^H \mathbf{Z}^{i\,H} \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H}\, \mathbf{y}_{-k} \end{bmatrix} \right|\right|_2^2 \right\} \tag{E.13}$$

and using the substitution $\mathbf{B} = \mathbf{Z}^i \mathbf{\Psi}_z$ yields

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{B}} \end{bmatrix} = \underset{\mathbf{X}_{z,k},\mathbf{B}}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{Z}^{i+1\,H}\mathbf{d}_\nu - \begin{bmatrix} \mathbf{X}_{z,k}^H & \mathbf{B}^H \end{bmatrix} \begin{bmatrix} \mathbf{y}_k \\ \mathbf{W}_{\nu,-k}^{i\,H}\, \mathbf{y}_{-k} \end{bmatrix} \right|\right|_2^2 \right\} \tag{E.14}$$

where $\widehat{\mathbf{X}}_{z,-k}$ is found as

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{B}}\,. \tag{E.15}$$

which can be derived from the fact that

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}_{z,-k}^i \widehat{\mathbf{\Psi}}_z = \mathbf{W}_{\nu,-k}^i \mathbf{Z}^i \widehat{\mathbf{\Psi}}_z = \mathbf{W}_{\nu,-k}^i \mathbf{Z}^i \left(\mathbf{Z}^i\right)^{-1} \widehat{\mathbf{B}} = \mathbf{W}_{\nu,-k}^i \widehat{\mathbf{B}}\,. \tag{E.16}$$

Comparing (E.11) and (E.14), it is observed that they both define a LMMSE problem with a similar form as (7.3), and hence their solution will have a similar form as (7.4). Furthermore, (E.11) and (E.14) are identical optimization problems except for a multiplication of the desired signal $\mathbf{d}_\nu$ with $\mathbf{Z}^{i+1\,H}$. Considering (7.4), their solutions will therefore only differ up to a right multiplication with $\mathbf{Z}^{i+1}$, i.e.,

$$\begin{bmatrix} \widehat{\mathbf{X}}_{z,k} \\ \widehat{\mathbf{B}} \end{bmatrix} = \begin{bmatrix} \widehat{\mathbf{X}}_{\nu,k} \\ \widehat{\mathbf{\Psi}}_\nu \end{bmatrix} \mathbf{Z}^{i+1}\,. \tag{E.17}$$

Plugging the lower part of (E.17) in (E.15) yields

$$\widehat{\mathbf{X}}_{z,-k} = \mathbf{W}^i_{\nu,-k}\widehat{\mathbf{\Psi}}_\nu \mathbf{Z}^{i+1} \tag{E.18}$$

and since

$$\widehat{\mathbf{X}}_{\nu,-k} = \mathbf{W}^i_{\nu,-k}\widehat{\mathbf{\Psi}}_\nu \tag{E.19}$$

this yields

$$\widehat{\mathbf{X}}_{z,-k} = \widehat{\mathbf{X}}_{\nu,-k}\mathbf{Z}^{i+1} \ . \tag{E.20}$$

Combining this with the upper part in (E.17), then the combined matrices $\widehat{\mathbf{X}}_z$ and $\widehat{\mathbf{X}}_\nu$ are obtained and are related as

$$\widehat{\mathbf{X}}_z = \widehat{\mathbf{X}}_\nu \mathbf{Z}^{i+1} \tag{E.21}$$

and hence, from Table E.1, also

$$\mathbf{W}^{i+1}_z = \mathbf{W}^{i+1}_\nu \mathbf{Z}^{i+1} \ . \tag{E.22}$$

It has thus been shown that, if $\mathbf{W}^i_z = \mathbf{W}^i_\nu \mathbf{Z}^i$, then also $\mathbf{W}^{i+1}_z = \mathbf{W}^{i+1}_\nu \mathbf{Z}^{i+1}$. Since the former holds for $i = 0$ due to the particular initialization of both AOs, the lemma is proven for any iteration i by an induction argument. □

An important corollary from this lemma is that, since the AO sequence $(\mathbf{W}^i_\nu)_{i\in\mathbb{N}}$ converges, (see E.7) then it immediately follows from Lemma 3 that $(\mathbf{W}^i_z)_{i\in\mathbb{N}}$ will also converge up to a $Q \times Q$-dimensional transformation of its columns, i.e.,

$$\lim_{i\to\infty} \min_{\mathbf{U}} \left|\left| \widehat{\mathbf{W}}_\nu - \mathbf{W}^i_z \mathbf{U} \right|\right|_F = 0 \tag{E.23}$$

for any given sequence $(\mathbf{Z}^i)_{i\in\mathbb{N}} = (\mathbf{Z}^0, \mathbf{Z}^1, \ldots)$. Note that, in Lemma 3, $\mathbf{Z}^0$ merely defines the initialization of AO2 with respect to the initialization of AO1. However, since both initializations are arbitrary, it is assumed that $\mathbf{Z}^0 = \mathbf{I}$ for any choice of the $\mathbf{Z}$-sequence (E.4) in the sequel.

A new AO procedure is now considered, referred to as AO3, which is defined as AO1, but where the objective function J_ν is replaced with the objective function $J_{k(i)}$, where the node-index $k(i)$ now increments in each iteration, looping over all nodes of $\mathcal{K}$ in a round-robin fashion. Similar as AO1 being a special case of AO2 (where $\mathbf{Z}^i = \mathbf{I}$, $\forall\, i \in \mathbb{N}$), also AO3 can be shown to be a special case of AO2. Indeed, AO3 is obtained by choosing

$$\mathbf{Z}^{i+1} = \left(\overline{\mathbf{A}}_\nu\right)^{-H} \overline{\mathbf{A}}^H_{k(i)} \tag{E.24}$$

in AO2. This means that the sequence of AO3 will also converge up to a $Q \times Q$-dimensional transformation similar to (E.23).

The changing node index in the objective function $J_{k(i)}$ used in AO3 allows to implement it in a distributed fashion in a fully connected WSN, which is explained next. This distributed algorithm will be referred to as the D-AO3 algorithm. Similar to the TI-DANSE algorithm, a specific compression rule at node k is defined, which is denote as $\mathbf{V}_k^i$ (having a similar function as $\mathbf{P}_k^i$ in the TI-DANSE algorithm), and its corresponding broadcast signals are again denoted as $\mathbf{z}_k^i = \mathbf{V}_k^{i\,H}\mathbf{y}_k$. Using this notation, the D-AO3 algorithm is described in Table E.2 (with the introduction of some auxiliary variables $\mathbf{G}_k$ and $\mathbf{W}_{kk}$, to make it easier to compare it with the TI-DANSE algorithm later). In the description of D-AO3, an incremental node-index k is used instead of the notation $k(i)$.

If all of the $\mathbf{V}_k^i$'s defined in the D-AO3 algorithm are stacked into a larger matrix

$$\mathbf{V}^i = \begin{bmatrix} \mathbf{V}_1^i \\ \vdots \\ \mathbf{V}_K^i \end{bmatrix} \tag{E.25}$$

then the sequence $(\mathbf{V}^i)_{i\in\mathbb{N}}$ of the D-AO3 algorithm will be identical to the sequence produced by the AO3 algorithm[1], i.e.,

$$(\mathbf{V}^i)_{i\in\mathbb{N}} = (\mathbf{W}_z^i)_{i\in\mathbb{N}} = (\mathbf{W}_\nu^i \mathbf{Z}^i)_{i\in\mathbb{N}} \tag{E.26}$$

where $(\mathbf{Z}^i)_{i\in\mathbb{N}}$ is defined in (E.24). Therefore, using the result (E.23), it is shown that

$$\lim_{i\to\infty} \min_{\mathbf{U}} \left|\left| \widehat{\mathbf{W}}_\nu - \mathbf{V}^i\mathbf{U} \right|\right|_F = 0\,. \tag{E.27}$$

Assume that, in the beginning of step 2 of iteration i in the D-AO3 algorithm, all of the compression rules $\mathbf{V}_q^i$ are replaced as follows

$$\forall\, q \in \mathcal{K} : \mathbf{V}_q^i \leftarrow \mathbf{V}_q^i\mathbf{T} \tag{E.32}$$

with an arbitrary full rank $Q \times Q$-dimensional matrix $\mathbf{T}$. Then it can be shown that this will not have any influence on the D-AO3 algorithm in the sense that $\mathbf{V}^{i+1}$ will be the same for any choice of $\mathbf{T}$. This can be proven as follows. First, recall that an iteration of D-AO3 is equivalent to an iteration of AO3, which at its turn is equivalent to an iteration of AO2, for the specific choice of $(\mathbf{Z}^i)_{i\in\mathbb{N}}$ defined in (E.24), and where $\mathbf{V}^i$ corresponds to $\mathbf{W}_z^i$. Therefore, step 2

[1] This follows from the fact that (E.28) is essentially the same as (E.12) where $\mathbf{V}_{-k}^i$ replaces $\mathbf{W}_{z,-k}^i$, such that $\boldsymbol{\eta}_{-k}^i = \mathbf{V}_{-k}^{i\,H}\mathbf{y}_{-k} = \mathbf{W}_{z,-k}^{i\,H}\mathbf{y}_{-k}$. The transmission of $\mathbf{G}_k^{i+1}$ to the other nodes, which then multiply their $\mathbf{V}_q^i$ with $\mathbf{G}_k^{i+1}$, corresponds to the resubstitution defined in (E.10). Therefore, each iteration of D-AO3 corresponds to an iteration of AO3, i.e., steps 2+3 in AO2 where $\mathbf{V}^i$ replaces $\mathbf{W}_z^i$.

Table E.2: Description of the D-AO3 algorithm in a fully connected WSN

1. Initialize $0 \rightarrow i$, $1 \rightarrow k$ (or $k(i) = k(0) = 1$) .
   Initialize $\mathbf{W}_{qq}^0$,$\mathbf{G}_q^0$, and $\mathbf{V}_q^0$ randomly, $\forall q \in \mathcal{K}$.
2. Node k updates its node-specific local parameters, $\mathbf{W}_{kk}$ and $\mathbf{G}_k$, by minimizing its LMMSE criterion based on the B most recent samples observations of its own local sensor signals and the summed received signals from the other $K-1$ neighboring nodes

$$\begin{bmatrix} \mathbf{W}_{kk}^{i+1} \\ \\ \mathbf{G}_k^{i+1} \end{bmatrix} = \underset{\mathbf{W}_{kk},\mathbf{G}_k}{\arg\min}\, \mathcal{E}\left\{ \left|\left| \mathbf{d}_k - \begin{bmatrix} \mathbf{W}_{kk}^H & \mathbf{G}_k^H \end{bmatrix} \breve{\mathbf{y}}_k^i \right|\right|_2^2 \right\} \tag{E.28}$$

   for which the solution is given by (7.4), and updates its compression matrix as

$$\mathbf{V}_k^{i+1} = \mathbf{W}_{kk}^{i+1} \ . \tag{E.29}$$

   Furthermore, node k broadcasts $\mathbf{G}_k^{i+1}$ to the other nodes, who perform the following update on their local compression rule

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{V}_q^{i+1} = \mathbf{V}_q^i \mathbf{G}_k^{i+1} \ . \tag{E.30}$$

   The other nodes do not update their node-specific parameters :

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{W}_{qq}^{i+1} = \mathbf{W}_{qq}^i \text{ and } \mathbf{G}_q^{i+1} = \mathbf{G}_q^i \ . \tag{E.31}$$

3. $i+1 \rightarrow i$.
4. $(i \bmod K) + 1, \rightarrow k$ (or $k(i) = (i \bmod K) + 1$) .
5. return to 2 .

in the D-AO3 algorithm is equivalent to solving the constrained optimization problem in step 2 of AO2. Since (E.32) does not change the constraint set, i.e., the column space of $\mathbf{V}^i$, it will not influence the outcome of this constrained optimization problem, and hence will not influence the resulting $\mathbf{W}_z^{i+1}$, or equivalently $\mathbf{V}^{i+1}$.

The case where $\mathbf{T}$ is chosen as $\mathbf{T} = \left(\mathbf{G}^i_{k(i-1)}\right)^{-1}$, $\forall\, i \in \mathbb{N}$ is now investigated. This is equivalent[2] to replacing (E.29) and (E.30) in Table E.2 with

$$\mathbf{P}^{i+1}_k = \mathbf{W}^{i+1}_{kk} \left(\mathbf{G}^{i+1}_k\right)^{-1} \tag{E.33}$$

and

$$\forall q \in \mathcal{K}\backslash\{k\} : \mathbf{P}^{i+1}_q = \mathbf{P}^i_q \,, \tag{E.34}$$

respectively, where the symbol $\mathbf{V}^i$ has been replaced with $\mathbf{P}^i$ to distinguish between the transformed and the non-transformed case. A key observation is that, after performing the above replacements in Table E.2, then the TI-DANSE algorithm described in Table 7.1 is actually obtained.

Therefore, to prove convergence of the TI-DANSE algorithm, the following sequence must be analyzed

$$\left(\mathbf{P}^i\right)_{i\in\mathbb{N}} \triangleq \left(\mathbf{V}^i \left(\mathbf{G}^i_{k(i)}\right)^{-1}\right)_{i\in\mathbb{N}} . \tag{E.35}$$

From (E.27) and (E.35) it follows that $(\mathbf{P}^i)_{i\in\mathbb{N}}$ will again converge up to a $Q \times Q$-dimensional transformation of its columns, i.e.,

$$\lim_{i\to\infty} \min_{\mathbf{U}} \left|\left|\widehat{\mathbf{W}}_\nu - \mathbf{P}^i\mathbf{U}\right|\right|_F = 0 \,. \tag{E.36}$$

Now setting

$$\mathbf{Q}^i \triangleq \arg\min_{\mathbf{U}} \left|\left|\widehat{\mathbf{W}}_\nu - \mathbf{P}^i\mathbf{U}\right|\right|_F \tag{E.37}$$

for any iteration i (note that $\mathbf{Q}^i$ cannot be computed in practice, since $\widehat{\mathbf{W}}_\nu$ is unknown). From (E.36) and (E.37), it follows that

$$i \to \infty \Rightarrow \mathbf{P}^i_k = \widehat{\mathbf{W}}_{\nu k} \left(\mathbf{Q}^i\right)^{-1} \;\forall k \in \mathcal{K} \,. \tag{E.38}$$

If node k performs an update at iteration i, its estimate of its node-specific desired signal $\mathbf{d}_k$ in TI-DANSE is given as (see (7.12)):

$$\widetilde{\mathbf{d}}^{i+1}_k = \left(\mathbf{W}^{i+1}_{kk}\right)^H \mathbf{y}_k + \left(\mathbf{G}^{i+1}_k\right)^H \boldsymbol{\eta}^i_{-k} \tag{E.39}$$

[2]Note that applying $\mathbf{V}^i_q \leftarrow \mathbf{V}^i_q \left(\mathbf{G}^i_{k(i-1)}\right)^{-1}$ in the beginning of step 2 is equivalent to applying $\mathbf{V}^{i+1}_q \leftarrow \mathbf{V}^{i+1}_q \left(\mathbf{G}^{i+1}_{k(i)}\right)^{-1}$ at the end of step 2.

where

$$\boldsymbol{\eta}_{-k}^{i} \triangleq \sum_{\forall q \in \mathcal{K}\setminus\{k\}} \mathbf{z}_{q}^{i} \quad \text{(see (7.1))} \tag{E.40}$$

$$= \sum_{\forall q \in \mathcal{K}\setminus\{k\}} \mathbf{P}_{q}^{i\,H} \mathbf{y}_{q} \tag{E.41}$$

$$= \mathbf{P}_{-k}^{i\,H} \mathbf{y}_{-k} \tag{E.42}$$

where $\mathbf{y}_{-k}$ is defined as in (2.20), but with $\mathbf{y}_k$ removed.

Using (E.42) and (E.38), and for $i \to \infty$, (E.39) becomes

$$i \to \infty : \widetilde{\mathbf{d}}_{k}^{i+1} = \left(\mathbf{W}_{kk}^{i+1}\right)^{H} \mathbf{y}_{k} + \left(\mathbf{G}_{k}^{i+1}\right)^{H} \left(\mathbf{Q}^{i}\right)^{-H} \left(\widehat{\mathbf{W}}_{\nu,-k}\right)^{H} \mathbf{y}_{-k} \,. \tag{E.43}$$

If node k would choose the following values

$$i \to \infty : \mathbf{W}_{kk}^{i+1} = \widehat{\mathbf{W}}_{\nu k} \left(\overline{\mathbf{A}}_{\nu}\right)^{-H} \overline{\mathbf{A}}_{k}^{H} \tag{E.44}$$

$$\mathbf{G}_{k}^{i+1} = \mathbf{Q}^{i} \left(\overline{\mathbf{A}}_{\nu}\right)^{-H} \overline{\mathbf{A}}_{k}^{H} \tag{E.45}$$

then it is easy to see that, when using (2.29), (E.43) becomes

$$i \to \infty : \widetilde{\mathbf{d}}_{k}^{i+1} = \left(\overline{\mathbf{A}}_{k}\right) \overline{\mathbf{A}}_{\nu}^{-1} \widehat{\mathbf{W}}_{\nu}^{H} \mathbf{y} = \widehat{\mathbf{W}}_{k}^{H} \mathbf{y} \tag{E.46}$$

i.e., node k is able to achieve the network-wide LMMSE solution by making a proper choice for its local parameters, which proves convergence and optimality of the node-specific signal estimates.

In principle, this concludes the proof. Nevertheless, the convergence of the internal parameters $\mathbf{W}_{kk}$ and $\mathbf{G}_{k}$ can be elaborated on a bit more. Due to the strict convexity of the local LMMSE estimation problem at node k, the optimality in (E.46) can only be achieved if node k indeed exactly performs the update given in (E.44) and (E.45). The right-hand side of (E.44) is not dependent on the iteration index i, and hence this implies convergence for $\mathbf{W}_{kk}$, $\forall k \in \mathcal{K}$. However, this does not hold for the right-hand side of (E.45). To prove convergence of the $\mathbf{G}_{k}$'s, $\mathbf{Q}^{i}$ must also converge to ensure that the right-hand side of (E.45) also becomes independent of the iteration index.

Using (E.44) and (E.45) when $i \to \infty$ yields

$$i \to \infty : \mathbf{P}_{k}^{i+1} \triangleq \mathbf{W}_{kk}^{i+1} \left(\mathbf{G}_{k}^{i+1}\right)^{-1} \tag{E.47}$$

$$= \widehat{\mathbf{W}}_{\nu k} \overline{\mathbf{A}}_{\nu}^{-H} \overline{\mathbf{A}}_{k}^{H} \overline{\mathbf{A}}_{k}^{-H} \overline{\mathbf{A}}_{\nu}^{H} \left(\mathbf{Q}^{i}\right)^{-1} \tag{E.48}$$

$$= \widehat{\mathbf{W}}_{\nu k} \left(\mathbf{Q}^{i}\right)^{-1} \tag{E.49}$$

which with (E.38), yields

$$i \to \infty : \mathbf{P}_k^{i+1} = \mathbf{P}_k^i \ . \tag{E.50}$$

This together with (E.37), shows that

$$i \to \infty : \mathbf{Q}^{i+1} = \mathbf{Q}^i \tag{E.51}$$

and with (E.45)

$$i \to \infty : \mathbf{G}_k^{i+1} = \mathbf{G}_k^i \tag{E.52}$$

which proves that subsequent values of $\mathbf{G}_k^i$, $\forall k$ become infinitely close when $i \to \infty$.

Remark 4. *It is noted that there is a slight abuse of the '=' notation in (E.51) and (E.52), as well as the expressions that were used to derive these. A correct way would be :*

$$\lim_{i \to \infty} \|\Delta^i\| = 0 \tag{E.53}$$

$$\Delta^{i+1} \triangleq \mathbf{G}_k^{i+1} - \mathbf{G}_k^i \ . \tag{E.54}$$

Note that (E.53) does not truly imply convergence of $\mathbf{G}_k^i$ in the strict mathematical sense, since $\sum_{i=0}^{\infty} \Delta^i$ can still be infinitely large. In principle, the coefficients of $\mathbf{G}_k$ can keep "floating" with step sizes that becomes infinitesimally small. Nevertheless, the optimality and convergence of the signal estimates, as stated in (E.46), will still hold, even if this floating behavior occurs.

□

Bibliography

[1] J. Yick, M. Biswanath, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[2] Y. Li, M. T. Thai, and W. Wu, *Wireless Sensor Networks and Applications*. Signals and Communication Technology, Springer, 2008.

[3] S. Haykin and K. J. R. Liu, *Handbook on Array Processing and Sensor Networks*. Adaptive and Cognitive Dynamic Systems: Signal Processing, Learning, Communications and Control, Wiley, 2010.

[4] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *IEEE Circuits and Syst. Mag.*, vol. 5, pp. 19–31, 2005.

[5] S. Movassaghi, M. Abolhasan, J. Lipman, D. Smith, and A. Jamalipour, "Wireless body area networks: A survey," *IEEE Communi. Surveys Tutorials*, vol. 16, pp. 1658–1686, Mar. 2014.

[6] J. Liang and Q. Liang, "Design and analysis of distributed radar sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 22, pp. 1926–1933, Nov. 2011.

[7] J. Wang, F. J. Tilmann, R. S. White, H. Soosalu, and P. Bordoni, "Application of multi-channel Wiener filters to the suppression of ambient seismic noise in passive seismic arrays," *Leading Edge*, vol. 27, no. 2, pp. 232–238, 2008.

[8] R. Lopes Pereira, J. Trindade, F. Gonçalves, L. Suresh, D. Barbosa, and T. Vazão, "A wireless sensor network for monitoring volcano-seismic signals," *Natural Hazards and Earth System Sciences*, vol. 14, no. 12, pp. 3123–3142, 2014.

[9] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, "Wireless multimedia sensor networks: A survey," *IEEE Wireless Commun.*, vol. 14, pp. 32–39, Dec. 2007.

[10] B. Harjito and S. Han, "Wireless multimedia sensor networks applications and security challenges," in *the Proc. of the Int. Conf. on Broadband, Wireless Computing, Communication and Applications (BWCCA)*, pp. 842–846, Nov. 2010.

[11] 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. on Audio, Speech, and Language Process.*, vol. 17, pp. 38–51, Jan. 2009.

[12] A. Bertrand and M. Moonen, "Robust distributed noise reduction in hearing aids with external acoustic sensor nodes," *EURASIP J. Adv. Sig. Process.*, vol. 58, pp. 1–14, Oct. 2009.

[13] W. Hu, N. Bulusu, C. T. Chou, S. Jha, A. Taylor, and V. N. Tran, "Design and evaluation of a hybrid sensor network for cane toad monitoring," *ACM Trans. Sen. Netw.*, vol. 5, no. 1, pp. 4–28, 2009.

[14] S. Markovich-Golan, S. Gannot, and I. Cohen, "A reduced bandwidth binaural MVDR beamformer," in *the Proc. of the Int. Workshop on Acoust. Echo and Noise Control (IWAENC)*, 2010.

[15] B. C. J. Moore, "Binaural sharing of audio signals: Prospective benefits and limitations," *The Hearing Journal*, vol. 60, no. 11, pp. 46–48, 2007.

[16] B. Cornelis, S. Doclo, T. Van dan Bogaert, M. Moonen, and J. Wouters, "Theoretical analysis of binaural multimicrophone noise reduction techniques," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 18, pp. 342–355, Feb. 2010.

[17] S. Markovich-Golan, A. Bertrand, M. Moonen, and S. Gannot, "Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks," *Signal Processing*, vol. 107, pp. 4–20, 2015.

[18] A. Bertrand, "Applications and trends in wireless acoustic sensor networks: a signal processing perspective," in *the Proc. of the IEEE Symp. on Communications and Vehicular Technology (SCVT)*, Nov. 2011.

[19] A. Flammini, P. Ferrari, D. Marioli, E. Sisinni, and A. Taroni, "Wired and wireless sensor networks for industrial applications," *Microelectronics Journal*, vol. 40, no. 9, pp. 1322–1336, 2009.

[20] A. Buades, B. Coll, and J. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[21] H. Faraji and W. J. MacLean, "CCD noise removal in digital images," *IEEE Transactions on Image Process.*, vol. 15, pp. 2676–2685, Sep. 2006.

[22] J. Benesty, J. Chen, Y. Huang, and I. Cohen, *Noise Reduction in Speech Processing*, vol. 2 of *Springer Topics in Signal Processing*. Springer-Verlag, 2009.

[23] S. Devuyst, T. Dutoit, P. Stenuit, M. Kerkhofs, and E. Stanus, "Cancelling ECG artifacts in EEG using a modified independent component analysis approach," *EURASIP J. Adv. Sig. Process.*, 2008.

[24] S. P. Cho, M. H. Song, Y. C. Park, H. S. Choi, and K. J. Lee, "Adaptive Noise Canceling of Electrocardiogram Artifacts in Single Channel Electroencephalogram," in *the Proc. of the IEEE Annu. Int. Conf. on Eng. in Medicine and Biology Soc. (IEMBS)*, pp. 3278–3281, 2007.

[25] M. J. Berryman, S. Messer, A. Allison, and D. Abbott, "Techniques for noise removal from EEG, ECG, and airflow signals in sleep patients," in *the Proc. of the Conf. on Society of Photo-Optical Instrumentation Engineers (SPIE)*, vol. 5467, pp. 89–97, May 2004.

[26] E. A. Clancy, E. L. Morin, and R. Merletti, "Sampling, noise-reduction and amplitude estimation issues in surface electromyography," *Journal of Electromyography and Kinesiology*, vol. 12, no. 1, pp. 1–16, 2002.

[27] S. V. Vaseghi, *Advanced Digital Signal Processing and Noise Reduction*. John Wiley & Sons, 2006.

[28] J. F. Cardoso, "Blind signal separation: statistical principles," *in the Proc. of the IEEE*, vol. 86, pp. 2009–2025, Oct. 1998.

[29] Y. Zhu, A. Vikram, and H. Fu, "On tracking multiple indistinguishable targets," in *the Proc. of the IEEE Int. Conf. on Mobile Adhoc and Sensor Systems (MASS)*, pp. 1–9, Oct. 2012.

[30] J. S. Lim and A. V. Oppenheim, "Enhancement and bandwidth compression of noisy speech," *in the Proc. of the IEEE*, vol. 67, pp. 1586–1604, Dec. 1979.

[31] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. on Acoustics, Speech and Signal Process.*, vol. 27, pp. 113–120, Apr. 1979.

[32] J. Chen, J. Benesty, Y. Huang, and S. Doclo, "New insights into the noise reduction wiener filter," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 14, pp. 1218–1234, Jul. 2006.

[33] J. Chen, J. Benesty, and Y. A. Huang, "On the optimal linear filtering techniques for noise reduction," *Speech Communication*, vol. 49, no. 4, pp. 305–316, 2007.

[34] P. C. Hansen and S. H. Jensen, "Subspace-based noise reduction for speech signals via diagonal and triangular matrix decompositions: Survey and analysis," *EURASIP J. Adv. Sig. Process.*, vol. 2007, no. 1, pp. 92–95, 2007.

[35] P. C. Loizou, *Speech Enhancement: Theory and Practice*. Signal Processing and Communications, Taylor & Francis, 2007.

[36] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE Trans. on Acoustics, Speech and Signal Processs.*, vol. 32, pp. 1109–1121, Dec. 1984.

[37] P. Vary, "Noise suppression by spectral magnitude estimation mechanism and theoretical limits," *Signal Processing*, vol. 8, no. 4, pp. 387–400, 1985.

[38] O. Cappe, "Elimination of the musical noise phenomenon with the Ephraim and Malah noise suppressor," *IEEE Trans. on Speech and Audio Process.*, vol. 2, pp. 345–349, Apr. 1994.

[39] Y. A. Huang, J. Benesty, and J. Chen, "Analysis and comparison of multichannel noise reduction methods in a common framework," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 16, pp. 957–968, Jul. 2008.

[40] T. Lotter and P. Vary, "Dual-channel speech enhancement by superdirective beamforming," *EURASIP J. Adv. Sig. Process.*, 2006.

[41] B. D. Van Veen and K. M. Buckley, "Beamforming: a versatile approach to spatial filtering," *IEEE ASSP Mag.*, vol. 5, pp. 4–24, Apr. 1988.

[42] D. B. Ward, R. A. Kennedy, and R. C. Williamson, "Theory and design of broadband sensor arrays with frequency invariant far-field beam patterns," *The Journal of the Acoustical Society of America*, vol. 97, no. 2, pp. 1023–1034, 1995.

[43] J. Bitzer and K. U. Simmer, "Superdirective microphone arrays," in *Microphone Arrays* (M. Brandstein and D. Ward, eds.), Digital Signal Processing, pp. 19–38, Springer Berlin Heidelberg, 2001.

[44] O. L. Frost III, "An algorithm for linearly constrained adaptive array processing," *in the Proc. of the IEEE*, vol. 60, pp. 926–935, Aug. 1972.

[45] L. J. Griffiths and C. W. Jim, "An alternative approach to linearly constrained adaptive beamforming," *IEEE Trans. on Antennas and Propagation*, vol. 30, pp. 27–34, Jan. 1982.

[46] B. Widrow, J. R. Glover Jr., J. M. McCool, J. Kaunitz, C. S. Williams, R. H. Hearn, J. R. Zeidler, J. Dong, Eugene, and R. C. Goodlin, "Adaptive noise cancelling: Principles and applications," *in the Proc. of the IEEE*, vol. 63, pp. 1692–1716, Dec. 1975.

[47] S. Gannot, D. Burshtein, and E. Weinstein, "Signal enhancement using beamforming and nonstationarity with applications to speech," *IEEE Trans. on Signal Process.*, vol. 49, pp. 1614–1626, Aug. 2001.

[48] J. Benesty and Y. Huang, *Adaptive Signal Processing: Applications to Real-World Problems*. Engineering Online Library, Springer, 2003.

[49] M. Souden, J. Benesty, and S. Affes, "On optimal frequency-domain multichannel linear filtering for noise reduction," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 18, pp. 260–276, Feb. 2010.

[50] T. C. Lawin-Ore and S. Doclo, "Analysis of rate constraints for MWF-based noise reduction in acoustic sensor networks," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, May 2011.

[51] M. Gastpar, P. L. Dragotti, and M. Vetterli, "The distributed Karhunen - Loeve transform," *IEEE Trans. on Inform. Theory*, vol. 52, pp. 5177–5196, Dec. 2006.

[52] I. D. Schizas, G. B. Giannakis, and Z.-Q. Luo, "Distributed estimation using reduced-dimensionality sensor observations," *IEEE Trans. on Signal Process.*, vol. 55, pp. 4284–4299, Aug. 2007.

[53] 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.*, vol. 58, pp. 5277–5291, Oct. 2010.

[54] A. Bertrand and M. Moonen, "Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology," *IEEE Trans. on Signal Process.*, vol. 59, pp. 2196–2210, May 2011.

[55] J. Szurley, A. Bertrand, and M. Moonen, "Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks," *Accepted for Publication in Signal Processing*, 2015.

[56] J. Szurley, A. Bertrand, and M. Moonen, "Topology-independent distributed adaptive node-specific signal estimation in wireless sensor networks." *submitted for publication*, 2015.

[57] A. Hassani, A. Bertrand, and M. Moonen, "Cooperative integrated noise reduction and node-specific direction-of-arrival estimation in a fully connected wireless acoustic sensor network," *Signal Processing*, vol. 107, pp. 68–81, 2015.

[58] J. Szurley, A. Bertrand, and M. Moonen, "Network topology selection for distributed speech enhancement in wireless acoustic sensor networks," in *the Proc. of the Eur. Signal Process. Conf. (EUSIPCO)*, pp. 1–5, Sep. 2013.

[59] A. Bertrand and M. Moonen, "Distributed eye blink artifact removal in a wireless EEG sensor network," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 5890–5894, May 2014.

[60] J. Chen and A. H. Sayed, "Diffusion adaptation strategies for distributed optimization and learning over networks," *IEEE Trans. on Signal Process.*, vol. 60, pp. 4289–4305, Aug. 2012.

[61] A. K. Das and M. Mesbahi, "Distributed linear parameter estimation over wireless sensor networks," *IEEE Trans. on Aerospace and Electron. Syst.*, vol. 45, pp. 1293–1306, Oct. 2009.

[62] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, pp. 155–171, May 2013.

[63] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. on Signal Process.*, vol. 55, pp. 4064–4077, Aug. 2007.

[64] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *in the Proc. of the IEEE*, vol. 95, pp. 215–233, Jan. 2007.

[65] J. Chen, C. Richard, and A. H. Sayed, "Diffusion LMS for clustered multitask networks," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 5487–5491, May 2014.

[66] J. Plata-Chaves, N. Bogdanovic, and K. Berberidis, "Distributed diffusion-based LMS for node-specific adaptive parameter estimation," in *the Proc. IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 7223–7227, 2014.

[67] N. Bogdanovic, J. Plata-Chaves, and K. Berberidis, "Distributed incremental-based LMS for node-specific adaptive parameter estimation," *IEEE Trans. on Signal Proces.*, vol. 62, pp. 5382–5397, Oct. 2014.

[68] N. Kimura and S. Latifi, "A survey on data compression in wireless sensor networks," in *the Proc. of the Int. Conf. on Inform. Technology: Coding and Computing (ITCC)*, pp. 8–13, IEEE Comput. Soc., 2005.

[69] Z.-Q. Luo, G. B. Giannakis, and S. Zhang, "Optimal linear decentralized estimation in a bandwidth constrained sensor network," in *the Proc. of the IEEE Symp. on Inform. Theory (ISIT)*, pp. 1441–1445, Sep. 2005.

[70] G. Yu, Chaoand Sharma, "Distributed estimation using reduced dimensionality sensor observations: A separation perspective," in *the Proc. of the Annu. Conf. on Inform. Sciences and Syst. (CISS)*, pp. 150–154, Mar. 2008.

[71] M. Shamaiah, S. Banerjee, and H. Vikalo, "Greedy sensor selection: Leveraging submodularity," in *the Proc. of the IEEE Int. Conf. Decision and Control (CDC)*, pp. 2572–2577, Dec. 2010.

[72] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. on Signal Process.*, vol. 57, pp. 451–462, Feb. 2009.

[73] G. Thatte and U. Mitra, "Sensor selection and power allocation for distributed estimation in sensor networks: Beyond the star topology," *IEEE Trans. Signal Process.*, vol. 56, pp. 2649–2661, Jul. 2008.

[74] J. Deng, Y. S. Han, P.-N. Chen, and P. K. Varshney, "Optimal transmission range for wireless ad-hoc networks based on energy efficiency," *IEEE Trans. on Commun.*, vol. 55, no. 9, pp. 1772–1782, 2007.

[75] S. Mahfoudh and P. Minet, "Survey of energy efficient strategies in wireless ad-hoc and sensor networks," in *the Proc. of the IEEE Int. Conf. on Networking (ICN)*, pp. 1–7, 2008.

[76] Q. Guan, F. R. Yu, S. Jiang, and V. C. M. Leung, "Capacity-optimized topology control for MANETs with cooperative communications," *IEEE Trans. on Wireless Commun.*, vol. 10, no. 7, pp. 2162–2170, 2011.

[77] Z. Fang and B. Bensaou, "Fair bandwidth sharing algorithms based on game theory frameworks for wireless ad-hoc networks," in *the Proc. of the IEEE Int. Conf. on Comput. and Commun. Societies (INFOCOM)*, vol. 2, pp. 1284–1295, 2004.

[78] P. Galiotos, "Security-aware topology control for wireless ad-hoc networks," in *the Proc. of the IEEE Int. Conf. on Global Telecommun. (GLOBECOM)*, pp. 1–6, 2008.

[79] L. Yu, N. Wang, W. Zhang, and C. Zheng, "Deploying a heterogeneous wireless sensor network," in *the Proc. of the Int. Conf. on Wireless Commun., Networking and Mobile Computing (WiCom)*, pp. 2588–2591, 2007.

[80] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, "Exploiting heterogeneity in sensor networks," in *the Proc. of the IEEE Int. Conf. on Comput. and Commun. Societies (INFOCOM)*, vol. 2, pp. 878–890, 2005.

[81] B.-J. Kwak, N.-O. Song, and M. E. Miller, "On the scalability of ad-hoc networks: a traffic analysis at the center of a network," in *the Proc. of the IEEE Int. Conf. on Wireless Commun. and Networking (WCNC)*, vol. 2, pp. 1030–1033, 2004.

[82] J. Eriksson, M. Faloutsos, and S. Krishnamurthy, "Scalable ad-hoc routing: the case for dynamic addressing," in *the Proc. of the IEEE Int. Conf. on Comput. and Commun. Societies (INFOCOM)*, vol. 2, pp. 1108–1119, 2004.

[83] G. Călinescu and S. Tongngam, "Relay nodes in wireless sensor networks," in *Proc. IEEE Int. Conf. on Wireless Algorithms, Syst., and Applicat. (WASA)*, pp. 286–297, 2008.

[84] X. Cheng, D. Z. Du, L. Wang, and B. Xu, "Relay sensor placement in wireless sensor networks," *Wireless Networks*, vol. 14, no. 3, pp. 347–355, 2008.

[85] S. Doclo and M. Moonen, "GSVD-based optimal filtering for single and multimicrophone speech enhancement," *IEEE Trans. on Signal Proces.*, vol. 50, pp. 2230–2244, Sep. 2002.

[86] J. Benesty, S. Makino, and J. Chen, *Speech Enhancement*. Signals and Communication Technology, Springer-Verlag New York, Inc., 2005.

[87] D. Florencio and H. Malvar, "Multichannel filtering for optimum noise reduction in microphone arrays," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, May 2001.

[88] A. Spriet, M. Moonen, and J. Wouters, "Spatially pre-processed speech distortion weighted multi-channel Wiener filtering for noise reduction," *Signal Processing*, vol. 84, no. 12, pp. 2367–2387, 2004.

[89] A. Spriet, S. Doclo, M. Moonen, and J. Wouters, "A unification of adaptive multi-microphone noise reduction systems," in *the Proc. of the Int. Workshop on Acoust. Echo and Noise Control (IWAENC)*, Sep. 2006.

[90] 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. on Audio, Speech, and Language Process.*, vol. 19, pp. 1368–1381, Jul. 2011.

[91] S. Doclo, T. J. Klasen, T. Van den Bogaert, J. Wouters, and M. Moonen, "Theoretical analysis of binaural cue preservation using multi-channel Wiener filtering and interaural transfer functions," in *the Proc. of the Int. Workshop on Acoust. Echo and Noise Control (IWAENC)*, Sep. 2006.

[92] 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*, vol. 49, no. 7-8, pp. 636–656, 2007.

[93] A. P. Liavas, "On the robustness of the finite-length MMSE-DFE with respect to channel and second-order statistics estimation errors," *IEEE Trans. Signal Process.*, vol. 50, pp. 2866–2874, Nov. 2002.

[94] E. Ollila, "On the circularity of a complex random variable," *IEEE Signal Process. Lett.*, vol. 15, pp. 841–844, 2008.

[95] B. Picinbono and P. Chevalier, "Widely linear estimation with complex data," *IEEE Trans. on Signal Proces.*, vol. 43, pp. 2030–2033, Aug. 1995.

[96] P. J. Schreier, L. L. Scharf, and C. T. Mullis, "A unified approach to performance comparisons between linear and widely linear processing," in *the Proc. of the IEEE Workshop on Statistical Signal Process.*, pp. 114–117, Sep. 2003.

[97] T. Adali, H. Li, and R. Aloysius, "On properties of the widely linear MSE filter and its LMS implementation," in *the Proc. of the Annu. Conf. on Inform. Sciences and Systems (CISS)*, pp. 876–881, Mar. 2009.

[98] J. Chen and J. Benesty, "A time-domain widely linear MVDR filter for binaural noise reduction," in *the Proc. of the IEEE Workshop on Applicat. of Signal Proces. to Audio and Acoust. (WASPAA)*, pp. 105–108, Oct. 2011.

[99] J. Benesty and J. Chen, "A multichannel widely linear approach to binaural noise reduction using an array of microphones," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 313–316, Mar. 2012.

[100] J. Benesty, J. Chen, and Y. Huang, "Binaural noise reduction in the time domain with a stereo setup," *IEEE Trans. Audio, Speech, and Language Process.*, vol. 19, pp. 2260–2272, Nov. 2011.

[101] J. Chen and J. Benesty, "On the time-domain widely linear LCMV filter for noise reduction with a stereo system," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 21, no. 7, pp. 1343–1354, 2013.

[102] C. Stanciu, J. Benesty, C. Paleologu, T. Gänsler, and S. Ciochina, "A widely linear model for stereophonic acoustic echo cancellation," *Signal Processing*, vol. 93, no. 2, pp. 511–516, 2013.

[103] Y. Zeng, R. C. Hendriks, and R. Heusdens, "Clique-based distributed beamforming for speech enhancement in wireless sensor networks," in *the Proc. of the Eur. Signal Process. Conf. (EUSIPCO)*, Sep. 2013.

[104] Y. Zeng and R. C. Hendriks, "Distributed delay and sum beamformer for speech enhancement via randomized gossip," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 22, pp. 260–273, Jan. 2014.

[105] M. C. Killion, "SNR loss : I can hear what people say, but I can't understand them," *Hearing Review*, vol. 12, pp. 8–14, Apr. 1997.

[106] H. Dillon, *Hearing Aids.* Thieme, 2012.

[107] J. Wouters and J. Vanden Berghe, "Speech recognition in noise for cochlear implantees with a two-microphone monaural adaptive noise reduction system," *Ear and Hearing*, vol. 22, pp. 420–430, Oct. 2001.

[108] J. G. Desloge, W. M. Rabinowitz, and P. M. Zurek, "Microphone-array hearing aids with binaural output .I. fixed-processing systems," *IEEE Trans. on Audio, Speech, and Language Process.*, vol. 5, pp. 529–542, Nov. 1997.

[109] J. Wouters, S. Doclo, R. Koning, and T. Francart, "Sound processing for better coding of monaural and binaural cues in auditory prostheses," *in the Proc. of the IEEE*, vol. 101, pp. 1986–1997, Sep. 2013.

[110] R. Gilkey and T. R. Anderson, *Binaural and Spatial Hearing in Real and Virtual Environments.* Lawrence Erlbaum, 1997.

[111] R. H. Gifford and L. J. Revit, "Speech perception for adult cochlear implant recipients in a realistic background noise: Effectiveness of preprocessing strategies and external options for improving speech recognition in noise," *Journal of the American Academy of Audiology*, vol. 21.7, p. 441–488, Jul.-Aug. 2010.

[112] E. C. Schafer and L. M. Thibodeau, "Electroacoustic verification of FM benefits in advanced hearing aid circuitry," *The Journal of the Acoustical Society of America*, vol. 112, no. 5, pp. 2271–2272, 2002.

[113] E. C. Schafer, K. Sanders, D. Bryant, K. Keeney, and N. Baldus, "Effects of voice priority in FM systems for children with hearing aids," *Journal of Educational Audiology*, vol. 19, pp. 12–24, 2013.

[114] G. Grimm, G. Guilmin, F. Poppen, M. Vlaming, and V. Hohmann, "The personal hearing system–a software hearing aid for a personal communication system," *EURASIP J. Adv. Sig. Process.*, vol. 591921, no. 1, pp. 1–9, 2009.

[115] T. Francart, A. Lenssen, and J. Wouters, "Enhancement of interaural level differences improves sound localization in bimodal hearing," *The Journal of the Acoustical Society of America*, vol. 130, no. 5, pp. 2817–2826, 2011.

[116] T. Van den Bogaert, J. Wouters, S. Doclo, and M. Moonen, "Binaural cue preservation for hearing aids using an interaural transfer function multichannel wiener filter," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, vol. 4, pp. 565–568, Apr. 2007.

[117] T. J. Klasen, T. Van den Bogaert, M. Moonen, and J. Wouters, "Binaural noise reduction algorithms for hearing aids that preserve interaural time delay cues," *IEEE Trans. on Signal Process.*, vol. 55, pp. 1579–1585, Apr. 2007.

[118] T. Van den Bogaert, S. Doclo, J. Wouters, and M. Moonen, "The effect of multimicrophone noise reduction systems on sound source localization by users of binaural hearing aids," *The Journal of the Acoustical Society of America*, vol. 124, no. 1, pp. 484–497, 2008.

[119] T. Van den Bogaert, S. Doclo, J. Wouters, and M. Moonen, "Speech enhancement with multichannel Wiener filter techniques in multimicrophone binaural hearing aids," *The Journal of the Acoustical Society of America*, vol. 125, no. 1, pp. 360–371, 2009.

[120] D. Wang and G. J. Brown, *Computational Auditory Scene Analysis: Principles, Algorithms, and Applications*. Wiley-IEEE Press, 2006.

[121] H. Kayser, S. D. Ewert, J. Anemüller, T. Rohdenburg, V. Hohmann, and B. Kollmeier, "Database of multichannel in-ear and behind-the-ear head-related and binaural room impulse responses," *EURASIP J. Adv. Sig. Process.*, vol. 2009, pp. 1–10, 2009.

[122] H. Rowaihy, S. Eswaran, M. Johnson, D. Verma, A. Bar-noy, and T. Brown, "A survey of sensor selection schemes in wireless sensor networks," in *the Proc. of the SPIE Defense and Security Symp. Conf. on Unattended Ground, Sea, and Air Sensor Technologies and Applicat.*, 2007.

[123] D. Golovin, M. Faulkner, and A. Krause, "Online distributed sensor selection," in *the Proc. of the ACM/IEEE Int. Conf. on Inform. Process. in Sensor Networks*, pp. 220–231, 2010.

[124] A. Bertrand and M. Moonen, "Efficient sensor subset selection and link failure response for linear MMSE signal estimation in wireless sensor networks," in *the Proc. of the Eur. Signal Process. Conf. (EUSIPCO)*, pp. 1092–1096, Aug. 2010.

[125] A. Bertrand, J. Szurley, P. Ruckebusch, I. Moerman, and M. Moonen, "Efficient calculation of sensor utility and sensor removal in wireless sensor networks for adaptive signal estimation and beamforming," *IEEE Trans. Signal Process.*, vol. 60, pp. 5857–5869, Nov. 2012.

[126] Z. Quan, W. J. Kaiser, and A. H. Sayed, "Innovations diffusion: A spatial sampling scheme for distributed estimation and detection," *IEEE Trans. Signal Process.*, vol. 57, pp. 738–751, Feb. 2009.

[127] Y.-H. Chang and X. Yu, "Reduced-rank antenna selection for MIMO DS-CDMA channels," in *the Proc. of the IEEE Int. Symp. on Circuits and Systems (ISCAS)*, vol. 2, pp. 1730–1733, May 2005.

[128] H. Godrich, A. P. Petropulu, and H. V. Poor, "Sensor selection in distributed multiple-radar architectures for localization: A knapsack problem formulation," *IEEE Trans. Signal Process.*, vol. 60, pp. 247–260, Jan. 2012.

[129] H. Godrich, A. Petropulu, and H. V. Poor, "A combinatorial optimization framework for subset selection in distributed multiple-radar architectures," in *the Proc. of the IEEE Int. Conf. on Acoust., Speech and Signal Process. (ICASSP)*, pp. 2796–2799, May 2011.

[130] F. Bian, D. Kempe, and R. Govindan, "Utility-based sensor selection," in *the Proc. of the Int. Conf. on Information Process. in Sensor Networks (IPSN)*, pp. 11–18, 2006.

[131] A. Das and D. Kempe, "Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection," in *the Proc. of the Int. Conf. on Machine Learning (ICML)*, pp. 1057–1064, 2011.

[132] G. H. Golub and C. F. Van Loan, *Matrix Computations*. The Johns Hopkins University Press, 3rd ed., Oct. 1996.

[133] D. H. Bailey and H. R. P. Gerguson, "A Strassen-Newton algorithm for high-speed parallelizable matrix inversion," in *the Proc. of the ACM/IEEE Conf. on Supercomputing*, pp. 419–424, 1988.

[134] D. S. Zois, M. Levorato, and U. Mitra, "Energy-efficient, heterogeneous sensor selection for physical activity detection in wireless body area networks," *IEEE Trans. on Signal Process.*, vol. 61, no. 7, pp. 1581–1594, 2013.

[135] P. Kuryloski, A. Giani, R. Giannantonio, K. Gilani, R. Gravina, V. P. Seppa, E. Seto, V. Shia, C. Wang, P. Yan, A. Y. Yang, J. Hyttinen, S. Sastry, S. Wicker, and R. Bajcsy, "Dexternet: An open platform for heterogeneous body sensor networks and its applications," in *the Proc. of the Int. Workshop on Wearable and Implantable Body Sensor Networks (BSN)*, pp. 92–97, 2009.

[136] J. Matamoros and C. Antón-Haro, "Hierarchical organizations of sensors for decentralized parameter estimation," in *the Proc. of the IEEE Int. Symp. on Signal Process. and Inform. Technology (ISSPIT)*, pp. 513–518, 2007.

[137] F. S. Cattivelli and A. H. Sayed, "Hierarchical diffusion algorithms for distributed estimation," in *the Proc. of the IEEE Workshop on Statistical Signal Process. (SSP)*, pp. 537–540, 2009.

[138] S. Zhu, C. Chen, X. Guan, and C. Long, "An estimator model for distributed estimation in heterogenous wireless sensor networks," in *the Proc. of the IEEE Int. Symp. on a World of Wireless Mobile and Multimedia Networks (WoWMoM)*, pp. 1–6, 2010.

[139] M. H. Chaudhary and L. Vandendorpe, "Performance of power-constrained estimation in hierarchical wireless sensor networks," *IEEE Trans. on Signal Process.*, vol. 61, no. 3, pp. 724–739, 2013.

[140] M. H. Chaudhary and L. Vandendorpe, "Power constrained linear estimation of correlated sources in hierarchical wireless sensor networks," in *the Proc. of the IEEE Int. Workshop on Signal Process. Advances in Wireless Commun. (SPAWC)*, pp. 146–150, 2011.

[141] J. M. Ng and Y. Zhang, "Reference region group mobility model for ad-hoc networks," in *the Proc. of the IFIP Int. Conf. on Wireless and Optical Communications Networks (WOCN)*, pp. 290–294, 2005.

[142] L. Ritchie, H. S. Yang, A. W. Richa, and M. Reisslein, "Cluster overlay broadcast (COB): MANET routing with complexity polynomial in source-destination distance," *IEEE Trans. on Mobile Computing*, vol. 5, no. 6, pp. 653–667, 2006.

[143] Y. P. Chen, A. L. Liestman, and J. Liu, "Clustering algorithms for ad-hoc wireless networks," in *Ad-Hoc and Sensor Networks*, 2004.

[144] B. Han, "Zone-based virtual backbone formation in wireless ad-hoc networks.," *Ad-Hoc Networks*, no. 1, pp. 183–200, 2008.

[145] X. Chen and P. Yu, "Research on hierarchical mobile wireless sensor network architecture with mobile sensor nodes," in *the Proc. of the Int. Conf. on Biomedical Eng. and Informatics (BMEI)*, vol. 7, pp. 2863–2867, 2010.

[146] P. Santi, "Topology control in wireless ad-hoc and sensor networks," *ACM Comput. Surv.*, vol. 37, no. 2, pp. 164–194, 2005.

[147] K. Erciyes, *Distributed Graph Algorithms for Computer Networks*. Springer London, 2013.

[148] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, pp. 814–818, Jun. 2005.

[149] K. Saleh, N. Nasser, and A. Alyatama, "Hierarchical and heterogeneous mobile agent-based wireless sensor networks," in *the Proc. of the Int. Conf. on Computing Technology and Inform. Management (ICCM)*, vol. 2, pp. 688–691, 2012.

[150] C. Bron and J. Kerbosch, "Algorithm 457: Finding all cliques of an undirected graph," *Commun. ACM*, vol. 16, no. 9, pp. 575–577, 1973.

[151] D. Ciullo, G. D. Celik, and E. Modiano, "Minimizing transmission energy in sensor networks via trajectory control," in *the Proc. of the Symp. on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, pp. 132–141, Jun. 2010.

[152] G. Mao, B. D. O. Anderson, and B. Fidan, "Path loss exponent estimation for wireless sensor network localization," *Comput. Netw.*, vol. 51, no. 10, pp. 2467–2483, 2007.

[153] I. Kang and R. Poovendran, "Broadcast with heterogeneous node capability [wireless ad hoc or sensor networks]," in *the Proc. of the IEEE Int. Conf. on Global Telecommun. (GLOBECOM)*, vol. 6, pp. 4114–4119, 2004.

[154] Y.-H. Song, C. Suh, and T.-G. Kwon, "Design and implementation of efficient base station for wireless sensor networks," in *the Proc. of the IEEE Int. Conf. on Sensors (ICSENS)*, pp. 726–729, 2006.

[155] T. Lee, Y. Yang, K. Shin, and M. Park, "An energy-efficient uni-scheduling based on S-MAC in wireless sensor network," in *the Proc. of the IEEE Int. Conf. on High Performance Computing and Commun. (HPCC)*, pp. 293–304, 2005.

[156] A. K. Das, R. J. Marks II, M. El-Sharkawi, P. Arabshahi, and A. Gray, "Minimum power broadcast trees for wireless networks: integer programming formulations," in *the Proc. of the IEEE Int. Conf. on Comput. and Commun. Societies (INFOCOM)*, vol. 2, pp. 1001–1010, 2003.

[157] X. Chen and N. C. Rowe, "Saving energy by adjusting transmission power in wireless sensor networks," in *the Proc. of the IEEE Int. Conf. on Global Telecommun. (GLOBECOM)*, pp. 1–5, 2011.

[158] A. Bertrand and M. Moonen, "Seeing the bigger picture: How nodes can learn their place within a complex ad-hoc network topology," *IEEE Signal Process. Mag.*, 2013.

[159] T. Opsahl, F. Agneessens, and J. Skvoretz, "Node centrality in weighted networks: Generalizing degree and shortest paths," *Social Networks*, vol. 32, no. 3, pp. 245–251, 2010.

[160] S. Kar and J. M. F. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Trans. on Signal Process.*, vol. 56, pp. 3315–3326, Jul. 2008.

[161] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd ed., 2001.

[162] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, 1979.

[163] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *in the Proc. of the IEEE*, vol. 98, pp. 1847–1864, Nov. 2010.

[164] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: A tiny aggregation service for ad-hoc sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, 2002.

[165] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *the Proc. of the ACM/IEEE Annu. Conf. on Mobile Computing and Networking (MobiCom)*, pp. 174–185, 1999.

[166] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Trans. Program. Lang. Syst.*, vol. 5, no. 1, pp. 66–77, 1983.

[167] H. Chen, A. M. Campbell, B. W. Thomas, and A. Tamir, "Minimax flow tree problems," *Networks*, vol. 54, no. 3, pp. 117–129, 2009.

[168] Y. Choi, M. Khan, V. S. A. Kumar, and G. Pandurangan, "Energy-optimal distributed algorithms for minimum spanning trees," *IEEE Journal on Selected Areas in Commun.*, vol. 27, pp. 1297–1304, Sep. 2009.

[169] R. L. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," *IEEE Signal Processing Mag.*, vol. 30, pp. 82–105, Jan. 2013.

[170] R. C. Hendriks, Z. Erkin, and T. Gerkmann, "Privacy preserving distributed beamforming based on homomorphic encryption," in *the Proc. of the Eur. Signal Process. Conf. (EUSIPCO)*, pp. 1–5, Sep. 2013.

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

[172] E. A. P. Habets, J. Benesty, S. Gannot, and I. Cohen, "The MVDR beamformer for speech enhancement," in *Speech Processing in Modern Communication* (I. Cohen, J. Benesty, and S. Gannot, eds.), vol. 3 of *Springer Topics in Signal Processing*, pp. 225–254, Springer Berlin Heidelberg, 2010.

[173] J. Benesty, J. Chen, and Y. Huang, "On widely linear Wiener and tradeoff filters for noise reduction," *Speech Commun.*, vol. 52, no. 5, pp. 427–439, 2010.

[174] R.-C. Li, "On eigenvalue variations of Rayleigh quotient matrix pencils of a definite pencil," *Linear Algebra and its Applications*, vol. 208–209, pp. 471–483, 1994.

[175] J. C. Bezdek and R. J. Hathaway, "Some notes on alternating optimization," in *Advances in Soft Computing*, pp. 187–195, Springer Berlin Heidelberg, 2002.

Curriculum Vitae



Joseph Szurley received his B.A. degree in physics from the University of California at Berkeley in 2005 and his M.Sc. in electrical engineering (with honors) from California State University Long Beach in 2010. Since 2011 he has been a Ph.D. candidate at KU Leuven, Belgium.

In 2003 he worked at Lawrence Berkeley National Laboratory in the Atomic Molecular and Optical Sciences (AMSO) division. From 2004-2007 he was a research assistant for Takeda Pharmaceuticals. From 2009-2010 he worked at Conexant Systems, Inc. in the product engineering group. His research interests include audio signal processing, wireless (acoustic) sensor networks and distributed signal and speech enhancement.

Publication List

International Journal Papers

1. Szurley J., Bertrand A., and Moonen M., "Topology-independent distributed adaptive node-specific signal estimation in wireless sensor networks", *submitted for publication*, 2015.

2. Szurley J., Bertrand A., and Moonen M., "Binaural noise cue preservation in a binaural noise reduction system with a remote microphone signal", *submitted for publication*, 2015.

3. Szurley J., Bertrand A., and Moonen M., "Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks", *accepted for publication in* Signal Processing, 2015.

4. Szurley J., Bertrand A., Ruckebusch P., Moerman I., and Moonen M., "Greedy distributed node selection for node-specific signal estimation in wireless sensor networks", Signal Processing, vol. 94, Jan. 2014, pp. 57-73.

5. Szurley J., Bertrand A., and Moonen M., "On the use of time-domain widely linear filtering for binaural speech enhancement", IEEE Signal Processing Letters, vol. 20, Jul. 2013, pp. 649-652.

6. Bertrand A., Szurley J., Ruckebusch P., Moerman I., and Moonen M., "Efficient calculation of sensor utility and sensor removal in wireless sensor networks for adaptive signal estimation and beamforming", IEEE Transactions on Signal Processing, vol. 60, no. 11, Nov. 2012, pp. 5857-5869.

International Conference Papers

1. Szurley J., Bertrand A., Moerman I., and Moonen M., "A modified broadcast strategy for distributed signal estimation in a wireless sensor

network with a tree topology", in the Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, May 2014, pp. 7620-7624.

2. Szurley J., Bertrand A., Moerman I., and Moonen M., "Network topology selection for distributed speech enhancement in wireless acoustic sensor networks", in the Proc. of the European Signal Processing Conference (EUSIPCO), Marrakech, Morocco, Sep. 2013, pp. 1-5.

3. Szurley J., Bertrand A., and Moonen M., "Improved tracking performance for distributed node-specific signal enhancement in wireless acoustic sensor networks", in the Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, Canada, May 2013, pp. 336-340.

4. Szurley J., Bertrand A., Ruckebusch P., Moerman I., and Moonen M., "Energy aware greedy subset selection for speech enhancement in wireless acoustic sensor networks", in the Proc. of the European Signal Processing Conference (EUSIPCO), Bucharest, Romania, Aug. 2012, pp. 789-793.

5. Bertrand A., Szurley J., and Moonen M., "Distributed distortionless signal estimation in wireless acoustic sensor networks", in the Proc. of the European Signal Processing Conference (EUSIPCO), Bucharest, Romania, Aug. 2012, pp. 1254-1258.

6. Szurley J., Bertrand A., and Moonen M., "Efficient computation of microphone utility in a wireless acoustic sensor network with multi-channel Wiener filter based noise reduction", in the Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan, Mar. 2012, pp. 2657-2660.

7. Szurley J., Bertrand A., Ruckebusch P., Moerman I., and Moonen M., "Utility based cross-layer collaboration for speech enhancement in wireless acoustic sensor networks", in the Proc. of the European Signal Processing Conference (EUSIPCO), Barcelona, Spain, Aug. 2011, pp. 235-239.



FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
Stadius Center for Dynamical Systems,
Signal Processing and Data Analytics
Kasteelpark Arenberg 10, bus 2446
B-3001 Leuven
Belgium
joseph.szurley@esat.kuleuven.be