Distributed adaptive estimation over multitask networks

Roula Nassif

UNIVERSITE COTE D'AZUR - UFR Sciences

Ecole Doctorale Sciences Fondamentales et Appliquées

THESE

pour obtenir le titre de
Docteur en Sciences
de l'UNIVERSITE COTE D'AZUR

Discipline : Sciences de l'Ingénieur

présentée et soutenue par
Roula NASSIF

Estimation Distribuée Adaptative sur les Réseaux Multitâches
Distributed Adaptive Estimation over Multitask Networks

Thèse dirigée par Cédric RICHARD et André FERRARI
soutenue le 30 novembre 2016

Jury :

Rapporteurs :	Romain COUILLET	Professeur	-	Centrale Supélec, Gif-sur-Yvette, France
	Eric MOULINES	Professeur	-	Ecole Polytechnique, Palaiseau, France
Examinateurs :	Pascal BIANCHI	Professeur	-	Télécom ParisTech, Paris, France
	Pierre BORGNAT	Directeur de recherche	-	Ecole Normale Supérieure de Lyon, France
	Danilo MANDIC	Professeur	-	Imperial College, Londres, U. K.
	Marc MOONEN	Professeur	-	KU Leuven, Belgique
Invité:	Ali H. SAYED	Professeur	-	University of California, Los Angeles, USA
Directeurs :	André FERRARI	Professeur	-	Université de Nice Sophia Antipolis, France
	Cédric RICHARD	Professeur	-	Université de Nice Sophia Antipolis, France

Abstract

Distributed adaptive estimation over multitask networks

Distributed adaptive learning allows for a collection of interconnected agents to perform parameter estimation tasks from streaming data by relying solely on local computations and interactions with immediate neighbors. Most prior literature on distributed inference is concerned with *single-task* problems, where agents with separable objective functions need to agree on a common parameter vector. However, several network applications require more complex models and flexible algorithms than single-task implementations since their agents have to estimate and track multiple objectives simultaneously. Networks of this kind are referred to as *multitask* networks. Although agents may generally have distinct though related tasks to perform, they may still be able to capitalize on inductive transfer between them to improve the estimation accuracy. This thesis aims to bring advances on distributed inference over multitask networks. First, the unsupervised scenario is considered. The well-known diffusion LMS strategy to solve single-task estimation problems is presented and its performance in multitask environments is studied in the presence of noisy communication links. In order to improve the estimation accuracy, an adaptive strategy allowing the agents to limit their cooperation to neighbors with similar objective is presented. Next, we consider the multitask diffusion LMS strategy which has been proposed to solve multitask estimation problems where the network is decomposed into clusters of agents seeking different, but close (in the squared Euclidean norm) parameter vectors. For rigor, a realistic scenario is assumed whereby the synchronous assumption is violated. The performance of the multitask strategy over asynchronous networks is assessed. Noteworthy, we find that the multitask strategy is robust to the various sources of uncertainties occurring in the asynchronous network. In the third part, we consider multitask estimation problems over clustered networks where the parameter vectors at neighboring clusters have a large number of similar entries and a relatively small number of distinct entries. Based on the proximal gradient method, a proximal multitask LMS strategy is derived to solve the problem and its stability is studied using the energy conservation framework. Finally, we consider multitask estimation problems where each agent is interested in estimating its own parameter vector and where the parameter vectors at neighboring agents are related linearly according to a set of constraints. A projection based LMS approach is derived and studied in detail. Several numerical examples and motivating applications are considered throughout the thesis to validate the theoretical models and to show the benefits of learning multiple tasks simultaneously.

Keywords: Distributed estimation, multitask networks, adaptive algorithms, cooperation, asynchronous networks, imperfect information exchange, diffusion strategies, regularizer, smoothness condition, sparsity-inducing coregularizer, forward-backward splitting, gradient projection method, clustering, energy conservation framework.

Résumé

Estimation distribuée adaptative sur les réseaux multitâches

L'apprentissage adaptatif distribué sur les réseaux permet à un ensemble d'agents de résoudre des problèmes d'estimation de paramètres en ligne en se basant simplement sur des calculs locaux et sur des échanges locaux avec les voisins immédiats. La littérature sur l'estimation distribuée considère essentiellement les problèmes à *simple tâche*, où les agents disposant de fonctions objectives séparables doivent converger vers un vecteur de paramètres commun. Cependant, dans de nombreuses applications nécessitant des modèles plus complexes et des algorithmes plus flexibles, les agents ont besoin d'estimer et de suivre plusieurs vecteurs de paramètres simultanément. Nous appelons ce type de réseau, où les agents doivent estimer plusieurs vecteurs de paramètres, réseau *multitâche*. Bien que les agents puissent avoir différentes tâches à résoudre, ils peuvent capitaliser sur le transfert inductif entre eux afin d'améliorer les performances de leurs estimés. Le but de cette thèse est de proposer et d'étudier de nouveaux algorithmes d'estimation distribuée sur les réseaux multitâches. Dans un premier temps, nous présentons l'algorithme *diffusion LMS* qui est une stratégie efficace pour résoudre les problèmes d'estimation à simple-tâche et nous étudions théoriquement ses performances lorsqu'il est mis en œuvre dans un environnement multitâche et que les communications entre les nœuds sont bruitées. Ensuite, nous présentons une stratégie de clustering non-supervisé permettant de regrouper les nœuds réalisant une même tâche en clusters, et de restreindre les échanges d'information aux seuls nœuds d'un même cluster. Dans un second temps, nous considérons l'algorithme *multitask diffusion LMS* qui a été proposé pour résoudre des problèmes d'estimation multitâches où le réseau est décomposé en clusters de nœuds, et où différents clusters cherchent à estimer différents vecteurs de paramètres qui sont proches en norme Euclidienne. Nous considérons le scénario asynchrone et nous étudions les performances de l'algorithme. Nous constatons que la stratégie multitâche est robuste aux événements asynchrones se produisant dans le réseau. Dans la troisième partie, nous considérons les problèmes d'estimation multitâches où les agents sont groupés en clusters, et où les vecteurs de paramètres des clusters voisins ont un grand nombre de composantes similaires et un petit nombre de composantes différentes. En se basant sur la méthode proximale, nous proposons l'algorithme *proximal multitask diffusion LMS* pour résoudre le problème et nous étudions la stabilité de l'algorithme. Finalement, nous considérons les problèmes d'estimation où chaque agent dans le réseau cherche à estimer son propre vecteur de paramètres et où les tâches aux nœuds voisins sont liées linéairement par des contraintes d'égalité. En se basant sur les méthodes de projection, nous proposons un algorithme pour résoudre le problème et nous étudions ses performances. Les simulations numériques montrent la validité des modèles théoriques et les bénéfices de la coopération sur les réseaux multitâches.

Mots-Clés : Estimation distribuée, réseaux multitâches, algorithmes adaptatifs, coopération, réseaux asynchrones, échange d'informations bruitées, stratégies de diffusion, régularisation, condition de régularité, parcimonie, méthode proximale, méthode de projection, clustering, conservation d'énergie.

Dedicated to my parents Michel and Thérèse
Dedicated to my sisters Hanane and Elissa
Dedicated to my partner in life and love Bassam

Acknowledgements

First of all, I would like to thank my supervisors Prof. Cédric Richard and Prof. André Ferrari for providing me with the opportunity to start my scientific career by offering me a PhD position at the J.-L. Lagrange Laboratory and for making my dream come true. During my PhD, I worked closely with Prof. Cédric who was always there to listen to me, to talk about my research directions, to proofread my papers, and to give me valuable suggestions. From the first day I met him, I learnt a lot from him. He thought me to be always persistent and to maintain high morals. I would like to thank him for his continuous encouragement, guidance, support, and optimism throughout my PhD program. He also provided me the opportunity to get in touch with distinguished professors in the field and to participate in both national and international conferences where I presented my work in front of experts. I would like to thank Prof. André for his support and valuable discussions, especially for his expertise in the convex optimization domain and statistical signal processing field.

I would like to express my gratitude to Prof. Ali Sayed from the University of California Loss Angeles (UCLA) for his cooperation during my thesis. His careful readings, useful comments, immense knowledge, careful planning in presenting the results, and high standards in research greatly helped to improve the quality of my work. It is a great honor for me to have worked with him.

I would also like to thank Prof. Romain Couillet, Prof. Eric Moulines, Prof. Pacal Bianchi, Prof. Pierre Borgnat, Prof. Danilo Mandic, and Prof. Marc Moonen for honoring me by accepting to be members of my thesis committee.

During my PhD study, I met various interesting and inspiring minds. To begin with, I am especially grateful for my colleague and compatriot Rita Ammanouil with whom I spent a lot of time discussing interesting problems, work related or not. Cooking with her, hanging around in Nice, watching movies, and sharing stressful moments are those unforgettable memories in the process. Thank you my best friend! I wish to give special thanks to Dr. Vincent Gripon, who I met in a summer school during my first year. His enthusiasm is contagious and the discussions with him meant a lot to me, both professionally and personally. I would like to thank Delphine for her friendship, warm welcome, help, and support during my thesis. Words cannot express my gratitude for her and for her husband Marc. I would like to thank Dr. Jie Chen for the scientific discussions, which were important for me at the beginning of the process. A special thanks to my lab-mates: Raja, Wei, and Khalil for the fun and sincere friendship.

I would like to thank my beloved partner and my rock, Bassam, for his love, patience, continuous support, unwavering kindness, and precious moments together. You are my greatest gift. I know that sometimes I'm a difficult person to deal with but yet, you are still here. Thank you for accepting my flaws, for protecting me, and for making me laugh when I didn't even want to smile. The world is a better place because of you! Let's continue our life journey!

I am forever indebted to my father Michel and my mother Thérèse for supporting me through all my studies. Their endless love, strong faith in God, positive attitude, and strong work ethic were an example to me to follow. I would not have made it to this point without their sacrifice, prayers, and care for my education. I would like to thank my sisters Hanane and Elissa for their

love, hope, friendship, and endless support and encouragements during these years. I also extend my thanks to all my friends, in particular to Perla, Reine, Sarah, Racha, Nancy, and Khalil for the strength that they gave me throughout the years.

Table of Contents

List of Figures

List of Tables

Acronyms

ADMM alternating direction method of multipliers

ANC active noise control

APA affine projection algorithm

ATC adapt-then-combine

BIBO bounded input bounded output

CTA combine-then-adapt

CLMS constrained least-mean-squares

D-NSPE diffusion node specific parameter estimation

EMSE excess-mean-square-error

LASSO least absolute shrinkage and selection operator

LMS least-mean-squares

MSD mean-square-deviation

MSE mean-square-error

NLMS normalized least-mean-squares

PDE partial differential equation

PSD power spectrum density

RLS recursive-least-squares

SNR signal-to-noise ratio

Abbreviations

i.i.d.	independent and identically distributed
RHS	right-hand side
vs.	versus
w.r.t.	with respect to

Notations

a	Normal font lowercase letters denote scalars
$\boldsymbol{a}$	Boldface lowercase letters denote column vectors
$\boldsymbol{A}$	Boldface uppercase letters denote matrices
$\mathcal{A}$	Calligraphy normal font uppercase letters denote sets
$\boldsymbol{\mathcal{A}}$	Calligraphy bold font uppercase letters denote block matrices
$[a_{\ell k}]$	Matrix $\boldsymbol{A}$ with (ℓ, k)-th entry $a_{\ell k}$
$[\boldsymbol{A}_{\ell k}]$	Block matrix $\boldsymbol{\mathcal{A}}$ with (ℓ, k)-th block entry $\boldsymbol{A}_{\ell k}$
$[\boldsymbol{a}]_k$	k-th entry of vector $\boldsymbol{a}$
$[\boldsymbol{\mathcal{A}}]_{k,\bullet}$	k-th block row of block matrix $\boldsymbol{\mathcal{A}}$
$[\boldsymbol{\mathcal{A}}]_{\bullet,k}$	k-th block column of block matrix $\boldsymbol{\mathcal{A}}$
$\boldsymbol{A}^\top$	Transpose of matrix $\boldsymbol{A}$
$\boldsymbol{A}^{-1}$	Inverse of matrix $\boldsymbol{A}$
$\boldsymbol{A}^\dagger$	Pseudo-inverse of full row-rank matrix $\boldsymbol{A}$, namely, $\boldsymbol{A}^\dagger = \boldsymbol{A}^\top \left(\boldsymbol{A}\boldsymbol{A}^\top\right)^{-1}$
$\mathrm{Tr}(\boldsymbol{A})$	Trace of matrix $\boldsymbol{A}$
$\lambda_{\min}(\boldsymbol{A})$	Minimum eigenvalue of matrix $\boldsymbol{A}$
$\lambda_{\max}(\boldsymbol{A})$	Maximum eigenvalue of matrix $\boldsymbol{A}$
$\lambda_i(\boldsymbol{A})$	i-th eigenvalue of matrix $\boldsymbol{A}$
$\rho(\boldsymbol{A})$	Spectral radius of matrix $\boldsymbol{A}$
$\boldsymbol{A} > \boldsymbol{0}$	Matrix $\boldsymbol{A}$ is positive definite
$\mid a \mid$	Absolute value of a
$\|\boldsymbol{a}\|_0$	Pseudo ℓ_0-norm of vector $\boldsymbol{a}$

$\|\boldsymbol{a}\|_1$	ℓ_1-norm of vector $\boldsymbol{a}$
$\|\boldsymbol{a}\|$	Euclidean norm or ℓ_2-norm of vector $\boldsymbol{a}$
$\|\boldsymbol{a}\|_\infty$	ℓ_∞-norm of vector $\boldsymbol{a}$
$\mathrm{vec}(\boldsymbol{A})$	Vector obtained by stacking the columns of $\boldsymbol{A}$ on top of each others
$\mathrm{bvec}(\boldsymbol{\mathcal{A}})$	Vector obtained by vectorizing and stacking blocks of $\boldsymbol{\mathcal{A}}$
$\boldsymbol{A} \otimes \boldsymbol{B}$	Kronecker product of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$
$\boldsymbol{\mathcal{A}} \otimes_b \boldsymbol{\mathcal{B}}$	Block Kronecker product of block matrices $\boldsymbol{\mathcal{A}}$ and $\boldsymbol{\mathcal{B}}$
$\mathrm{diag}\{\boldsymbol{A}, \boldsymbol{B}\}$	Block matrix with block entry $\boldsymbol{B}$ placed immediately below and to the right of its predecessor $\boldsymbol{A}$
$\mathrm{col}\{\boldsymbol{a}, \boldsymbol{b}\}$	Block column vector with entries $\boldsymbol{a}$ and $\boldsymbol{b}$
$\boldsymbol{0}$	Vector or matrix with zero entries
$\mathbb{1}$	Vector with all its entries equal to one
$\boldsymbol{I}_M$	Identity matrix of size $M \times M$
$\varnothing$	Empty set
$\mathrm{card}\{\mathcal{X}\}$	Cardinality of set $\mathcal{X}$, i.e., the number of elements of $\mathcal{X}$
$\mathcal{X} \cap \mathcal{Y}$	Intersection of sets $\mathcal{X}$ and $\mathcal{Y}$
$\mathcal{X} \setminus \mathcal{Y}$	Relative complement of set $\mathcal{Y}$ in set $\mathcal{X}$
$\mathbb{E}$	Expected value operator
$\nabla_x f$	Gradient vector of function f relative to $\boldsymbol{x}$
$\mathrm{prox}_g(\boldsymbol{a})$	Proximal operator of function g at vector $\boldsymbol{a}$
$\mathrm{P}_{\mathcal{X}}(\boldsymbol{a})$	Projection of vector $\boldsymbol{a}$ onto convex set $\mathcal{X}$
$\mathcal{N}(a, b^2)$	Gaussian distribution with mean a and variance b^2
$\mathcal{U}(a, b)$	Uniform distribution with parameters a and b
δ_{ij}	Kronecker delta function of i and j, namely, $\delta_{ij} = 1$ if $i = j$, and zero otherwise

CHAPTER 1

INTRODUCTION

In this chapter, we review the concept of distributed processing over networks by focusing first on the single-task scenario where all agents are seeking a common objective. In particular, we recall state-of-the-art distributed techniques to solve general *single-task* estimation problems in a distributed manner by focusing on diffusion type strategies. Next, we introduce *multitask* networks considered in this work and we provide a literature review describing relevant works in the field of distributed processing over this kind of networks. Finally, we provide an overview of the problems addressed in this dissertation and a brief chapter-by-chapter description.

1.1 Distributed processing over networks

With the rapid development of computer, network, and communication technologies, distributed data processing techniques have received a lot of attention in many areas of engineering, biological, and social sciences. A wide range of problems are network structured [Jackson, 2008, Newman, 2010, Lewis, 2011, Sayed, 2014a]. Sensor networks, power grids, communication networks, biological colonies, economic networks, and internet are some typical examples. Such networked systems consist of a collection of autonomous agents (sensors, processors, actuators, etc.) distributed over some geographic area and connected through a network topology. In a distributed or decentralized mode of operation, the agents that are connected by a topology are allowed to exchange information (raw data, processed data, etc.) only with their intermediate neighbors and to perform local computations in order to achieve some network global task. Nature exhibits many motivating examples where a complex pattern of behavior arises from limited and localized interactions among the individual agents of the network. While each individual agent is not capable of complex behavior, it is the interaction among agents and the distributed processing of dispersed data that enable the agents to solve sophisticated tasks. For example,

in fish schools [Partridge, 1982], fishes are able to swim in the same direction in a coordinated manner. The individual agents tend to have similar speeds, to move in alignment, and to react instantly when predators appear by reconfiguring their topology to evade the predator and then regrouping to continue.

Studying and developing strategies for distributed optimization, inference, and parameter estimation over networks become an important step to handle inference problems in several domains, including signal processing, sensor networks, machine learning, optimization, biology, and control. The objective of the network is to estimate some parameters or phenomena of interest from streaming data collected by the agents, e.g., the position of some targets or the power system state in power grids. Since the parameters of interest may change over time, the algorithm must be able to track these changes. Furthermore, in real applications, the network system is not very reliable and the estimation algorithm must be able to cope with several sources of uncertainties such as changing topologies, packet losses, agents turning on and off, random link failures, quantization errors, and additive noise over the communication links. Cooperation, real-time adaptation, self-healing, and self-organization are particular features possessed by biological systems to be inherited by networked systems tasked with distributed parameters estimation.

1.1.1 Centralized vs. distributed solutions

There are two main strategies for the estimation of parameters or phenomena from noisy streaming measurements collected by a set of spatially distributed agents, namely, centralized solutions and distributed solutions. In a centralized mode of operation, agents send their data to a fusion center for processing. The fusion center performs all the computations required for the estimation and sends the results back to the agents. While centralized solutions may strongly benefit from collecting the data from across the network at the fusion center, there are several reasons for which one may desire distributed or decentralized implementations relying only on in-network processing and local exchange of information. In addition to the fact that the datasets in many modern problems are already distributed over geographically distinct sources, the centralized solution suffers from some limitations related to [Sayed, 2014a]:

- **Communication ressources:** In a centralized solution, exchanging the data back and forth between the agents and the fusion center results in a waste of communication ressources and power;

- **Robustness:** In a centralized solution, any malfunction in the fusion center can lead to a network breakdown;

- **Privacy concerns:** In some cases, there are some privacy issues in sharing data with fusion center. In distributed implementations, agents are allowed to share locally some processed version of the data so that their privacy is preserved.

1.1.2 Distributed adaptive estimation over single-task networks

Distributed adaptive processing allows for a collection of agents to solve optimization problems and parameter estimation problems from streaming data. It is particularly attractive in applications where the underlying signal statistics are unknown or time varying. Adaptation helps the network to track variations in the parameters of interest as new measurements become available. The continuous learning and diffusion of information across the network enable the agents to respond, not only to drifts in the data, but also to changes in the network topology. Distributed adaptive algorithms have also been used to model several biological networks encountered in nature, such as fish schooling, bird flight formations, and bacteria motility [Sayed et al., 2013].

The literature on distributed estimation theory is large. However, we limit the discussion to the class of *single-time* scale distributed adaptive algorithms based on *stochastic (sub-)gradient* techniques that is directly related to the contents of this thesis. A comprehensive overview of other classes of distributed algorithms, including ADMM based methods and two-time scales implementations, is provided in [Sayed, 2014a]. Stochastic approximation techniques allow to iteratively solve convex optimization problems based on noisy (sub-)gradient observations.

Depending on the cooperation protocol allowed among the agents, we distinguish between two main classes of distributed algorithms, namely, incremental type algorithms and diffusion type algorithms. As shown in Figure 1.1 (left), in an incremental mode of cooperation [Bertsekas, 1997, Nedic and Bertsekas, 2001, Rabbat and Nowak, 2005, Lopes and Sayed, 2007, Blatt et al., 2007], each node communicates with only one neighbor, and the information flows on a cyclic path that runs across all agents. Although incremental cooperation tends to require the least amount of communications, it suffers from some limitations including the definition of a cyclic path, which is an NP-hard problem, and sensitivity to links and agents failures. On the other hand, in a diffusion mode of cooperation [Nedic and Ozdaglar, 2009, Dimakis et al., 2010, Kar and Moura, 2011, Lopes and Sayed, 2008, Cattivelli and Sayed, 2010, Chen and Sayed, 2012], agents communicate with their immediate neighbors as dictated by the network topology. Figure 1.1 (right) illustrates such diffusion mode. Although diffusion mode of cooperation requires more communications than the incremental mode, it relaxes the requirement of setting a cyclic path and improves the robustness to links and agents failures. For these reasons, diffusion mode of cooperation is considered throughout this thesis.

The existing literature on distributed estimation over networks is mostly concerned with consensus problems where a collection of agents with separable objective functions need to converge asymptotically toward a common parameter vector. This kind of problem, where all agents in the network are seeking the same objective, is referred to as *single-task* distributed estimation problem. In general, single-task estimation over networks can be expressed as the problem of minimizing the aggregate sum of cost functions, each available at an agent, subject to convex

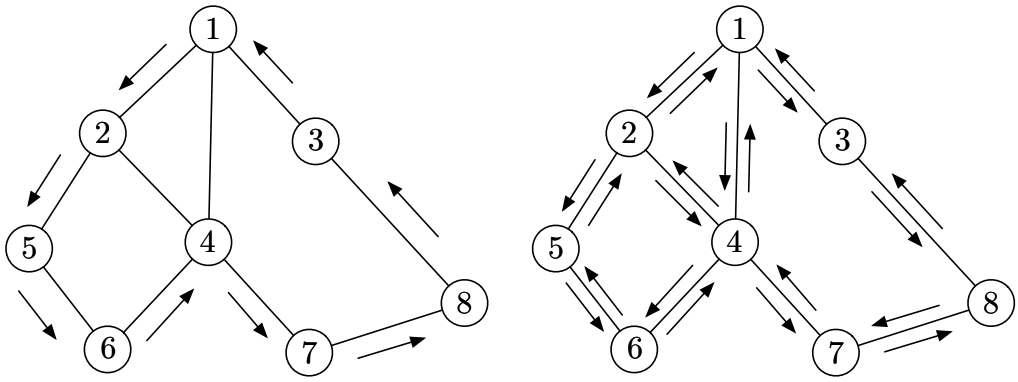


Figure 1.1: Two modes of cooperation. (*Left*) Incremental mode of cooperation. (*Right*) Diffusion mode of cooperation.

constraints that are also distributed across the agents:

$$\begin{aligned} &\underset{\boldsymbol{w}}{\text{minimize}} \;\; J^{\text{glob}}(\boldsymbol{w}) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}), \\ &\text{subject to} \;\; \boldsymbol{w} \in \mathcal{W}_1 \cap \cdots \cap \mathcal{W}_N, \end{aligned} \tag{1.1}$$

where $\boldsymbol{w}$ is the global optimization variable, N is the number of agents in the network, $J_k(\cdot)$ is the cost function at agent k, and $\mathcal{W}_k$ is the constraint set at agent k (see Figure 1.2 for an illustration). Each agent seeks to estimate the minimizer of (1.1) through local computations and communications among neighboring agents without the need to know any of the constraints or costs besides their own. Within the class of single-time scale stochastic gradient algorithms using diffusion implementation, two main strategies have been proposed for solving single-task estimation problems in a fully distributed manner, namely, consensus strategies [Nedic and Ozdaglar, 2009, Dimakis et al., 2010, Kar and Moura, 2011] and diffusion strategies [Lopes and Sayed, 2008, Cattivelli and Sayed, 2010, Chen and Sayed, 2012, Sayed et al., 2013, Chen and Sayed, 2013, Sayed, 2014a,b,c, Towfic and Sayed, 2014]. As shown in [Tu and Sayed, 2012], consensus strategies can lead to unstable network behavior when constant step-sizes enabling adaptation are used. In this dissertation, we shall focus instead on diffusion strategies that are particularly attractive since they are scalable, robust, enable continuous learning and adaptation in response to data drifts, and lead to stable behaviors when constant step-sizes are used.

In a nutshell, diffusion strategies are single-time scale distributed adaptive algorithms that are able to respond to streaming data. Since their introduction, they have been adopted to solve distributed estimation and optimization problems subject to a wide range of conditions [Srivastava and Nedic, 2011, Lorenzo et al., 2012, Liu et al., 2012, Chouvardas et al., 2012, Lorenzo and Sayed, 2013, Chen and Sayed, 2013, Wee and Yamada, 2013, Chouvardas et al., 2013, Sayed, 2014a, Towfic and Sayed, 2014, Vlaski and Sayed, 2015, Xia et al., 2011, Kanna and Mandic, 2016, Bianchi et al., 2011]. They have also been used to model various forms of complex behav-

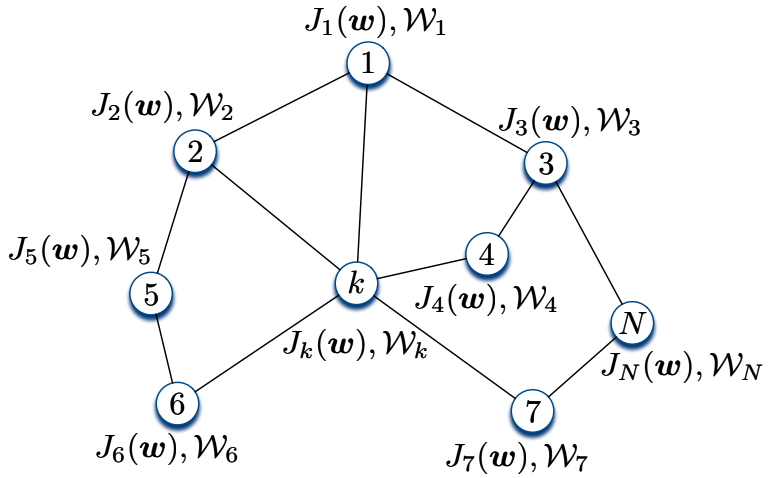


Figure 1.2: Single-task estimation over networks. All agents are seeking the solution of (1.1).

iors encountered in nature [Chen et al., 2010, Cattivelli and Sayed, 2011, Tu and Sayed, 2011b, Sayed et al., 2013].

1.2 Distributed estimation over multitask networks

Recently, we have witnessed a growing interest in distributed inference over *multitask* networks where several parameter vectors, also called *tasks*, need to be estimated simultaneously over a network of agents. When the tasks are related, exploiting the dependencies and the similarities between tasks is often the key to obtain better performance results. Prior to distributed estimation over networks, the problem of estimating simultaneously several related tasks has received considerable attention in several domains, specifically, in the machine learning. Within this community, this problem is the well known *multitask learning* problem or *learning to learn* problem. Depending on the relationships between tasks, several *centralized* multitask approaches have been derived and studied in the machine learning literature, under the *assumption* that all data are available at the starting point at the fusion center. Although this dissertation is concerned with *distributed adaptive* estimation over multitask networks, we devote the next paragraph to briefly describe and review multitask learning in the machine learning community for motivation purposes.

1.2.1 Insights from machine learning

Prior to the machine learning domain, the simultaneous estimation of several statistical models was considered within the econometrics and statistics literature [Greene, 2003, Zellner, 1962, Brown and Zidek, 1980, Breiman and Friedman, 1997]. Within the machine learning community, learning multiple related tasks simultaneously has been shown to improve performance relative to the traditional approach of learning each task independently [Caruana, 1997, Thrun and Pratt,

1998, Baxter, 2000, Bakker and Heskes, 2003, Evgeniou and Pontil, 2004, Evgeniou et al., 2005, Argyriou et al., 2007, Jacob et al., 2009]. Multitask learning is a machine learning approach to inductive transfer (using what is learned for one problem to help another problem) that improves generalization performances by using the domain information contained in the training signals of *related* tasks as an inductive bias. Multitask learning has been applied to several important problems such as, predicting the values of possibly related indicators in finance [Greene, 2003], solving the problem of data sparsity encountered in bioinformatics applications [Xu and Yang, 2011], classifying the web-pages into possibly related categories [Chen et al., 2009a]. Depending on the application, several task relatedness models have been considered. For example, in [Evgeniou et al., 2005, Jacob et al., 2009], it is assumed that the functions to be learned are close to each other in some Hilbert spaces. However, in [Caruana, 1997, Baxter, 2000], it is assumed that the tasks share a common underlying representation. Learning a low-dimensional representation shared across multiple related tasks is considered in the work [Argyriou et al., 2007]. Based on the prior information about how tasks are related, multitask learning algorithms were derived by translating this prior information into constraints on the parameter vectors or functions to be learned. A commonly used approach for designing multitask learning algorithms is to formulate a global regularized optimization problem by adding a regularization term encoding the relationships between tasks to the empirical loss function (squared loss, hinge loss, etc.) that penalizes the error on the training data. A survey of task regularization methods for multitask learning in machine learning is provided in [Xu and Yang, 2011]. Besides the regularization based methods, Bayesian based approaches have also been used for multitask learning, where a probability model capturing the relations between tasks is estimated simultaneously with functions corresponding to each task [Bakker and Heskes, 2003].

1.2.2 Distributed adaptive estimation over multitask networks

There are many multitask oriented network applications where agents are observing distinct phenomena and are sensing data arising from different models. For example, in monitoring applications, different clusters of agents within a network may have to track multiple targets moving along correlated trajectories. Although clusters have distinct though related tasks to perform (e.g., estimating the coordinates of their targets), the agents may still be able to capitalize on inductive transfer between clusters to enhance their estimation accuracy. In distributed power system monitoring, the local state vectors to be estimated at neighboring control centers may overlap partially since the areas in a power system are interconnected [Kekatos and Giannakis, 2013, Abboud et al., 2016]. Likewise, in wireless acoustic sensor networks dealing with distributed active noise control (ANC), agents need to estimate different but overlapping ANC filters [Plata-Chaves et al., 2016b]. In another example, sensor networks deployed to estimate a spatially-varying temperature profile need to exploit more directly the spatio-temporal correlations that exist between measurements at neighboring nodes [Abdolee et al., 2014]. In Big Data

scenarios, learning a dictionary model over a network of agents, where each agent is in charge of a part of the dictionary elements, is often required. This is because large dictionary models may already be available at different locations and it is not feasible to aggregate all dictionaries at one location due to communication and privacy considerations [Chen et al., 2015b]. In cognitive radio networks, secondary users need to estimate the power spectrum transmitted by all primary users, as well as local interference sources. The resulting multitask problem requires cooperation between secondary users in order to alleviate the effect of signal path loss or local shadowing [Bogdanović et al., 2014].

In all the previous examples, it is assumed that agents have some prior knowledge about how tasks are related and about the cluster they belong to. Nevertheless, distributed estimation over multitask networks has also been addressed in an unsupervised way, where it is assumed that there is no prior information about the tasks or clusters. In the following, we provide a literature review describing relevant works in the context of distributed estimation over multitask networks in both supervised and unsupervised scenarios.

1.2.2.1 Unsupervised scenario

In this scenario, no prior information on possible relationships between tasks is assumed and agents do not know which other agents share the same task. In this case, all agents cooperate with each other as dictated by the network topology. Several research efforts have focused on analyzing the performance of diffusion strategies when they are run, intentionally or unintentionally, in a multitask environment.

It is shown in [Chen and Sayed, 2013] that the diffusion iterates converge toward Pareto optimal solution when the optimization problem consists of a sum of individual costs with possibly different minimizers. A detailed convergence analysis is provided in [Sayed, 2014a], where it is shown that the value of the Pareto optimal point is influenced by many factors, including the topology, the combination policies, and the step-sizes. It is further shown in [Chen et al., 2015a] that, when the tasks are sufficiently similar to each other, the single-task diffusion least-mean-squares (LMS) algorithm can still perform better than non-cooperative strategies despite the bias induced by cooperating agents with different objectives.

If, on the other hand, the only available information is that clusters may exist in the network (but their structures are not known), then extended diffusion strategies are proposed in [Zhao and Sayed, 2012, Chen et al., 2015a, Zhao and Sayed, 2015d, Khawatmi et al., 2015, Nassif et al., 2016a] for setting the combination weights in an online manner in order to enable automatic network clustering and, subsequently, to enable agents to identify which neighbors belong to the same cluster and which neighbors should be ignored during cooperation. Such diffusion strategies have been used in [Monajemi et al., 2016] to model the brain connectivity, which is an important task in the neuroscience community. By considering the brain as a multitask network (where each agent consists of an EEG electrode attached to the scalp), it is shown how the adaptive

combination weights can be used to model the relation between tremor intensity of Parkinson's disease patients and the brain connectivity.

1.2.2.2 Supervised scenario

In this scenario, it is assumed that the agents know which cluster they belong to. In this case, existing distributed strategies to address multitask problems mostly depend on how the tasks are related to each other and on exploiting some prior information.

To the best of our knowledge, the works [Bertrand and Moonen, 2010, 2011] were the first to consider explicitly a multitask setting where different but related estimation tasks exist. Specifically, distributed multitask algorithms over fully connected and tree networks were derived to estimate node-specific signals sharing a common latent signal subspace. These properties are exploited to compress information and reduce communication costs. A recent related work dealing with heterogeneous and mixed topology has appeared in [Szurley et al., 2015]. It should be noted that, in this dissertation, we are focusing on *parameter estimation* tasks rather than *signal estimation* tasks. As explained in [Bertrand, 2011], these problems are different and need to be tackled in very different ways.

There have been several useful works dealing with distributed parameter estimation problems over multitask networks. In [Abdolee et al., 2014], a multitask model is obtained by discretizing a second-order partial differential equation (PDE) representing dynamic systems with spatially varying parameters. The multitask problem is transformed into a single-task problem by representing the space-varying parameters of the underlying phenomena as the product of an agent specific matrix of basis functions $\boldsymbol{B}_k$ and a space-invariant parameter vector $\boldsymbol{u}^o$ (see Table 1.1 further ahead). Knowing the matrix $\boldsymbol{B}_k$ at each agent k, the network problem reduces to the estimation of the global parameter vector $\boldsymbol{u}^o$, which can be solved by means of diffusion LMS strategies.

An alternative way to exploit and model relationships among tasks is to formulate optimization problems with appropriate co-regularizers between agents. In this context, it is assumed that different clusters within the network are interested in estimating their own models, and there are some correlations among the models of adjacent clusters. As it is common in the machine learning, one way to capture these correlations is to use appropriate regularization terms. For example, a diffusion based LMS strategy is proposed in [Chen et al., 2014b,c] by adding squared ℓ_2-norm co-regularizers to the MSE criterion in order to promote smoothness of the graph signal. Its convergence behavior is studied over asynchronous networks in [Nassif et al., 2014, 2016c]. Useful extensions of the strategy proposed in [Chen et al., 2014c] dealing with adaptive regularization parameters rather than fixed parameters, partial diffusion rather than standard diffusion, and affine projection algorithm (APA) rather than LMS algorithm appeared recently in [Monajemi et al., 2015, Gogineni and Chakraborty, 2015a,b]. On the other hand, non-smooth co-regularizers (ℓ_1-norm and reweighted ℓ_1-norm) were considered in the works [Nassif

et al., 2015, 2016d] in order to solve multitask problems where the optimal models of adjacent clusters have a large number of similar entries.

In [Bogdanović et al., 2014], a diffusion strategy of the LMS type is developed to address multitask estimation problems where agents are interested in estimating parameters of local interest and parameters of global interest. Particularly, the proposed method consists of running a local LMS at each agent in order to estimate the parameters of local interest and applying a diffusion LMS at all agents in order to estimate the parameters of global interest. An extended strategy allowing the agents to estimate parameters of common interest to a subset of agents simultaneously with parameters of local and global interest is provided in [Plata-Chaves et al., 2015]. Related unsupervised strategies dealing with group of variables rather than variables have been proposed in [Chen et al., 2016, Plata-Chaves et al., 2016a]. In another work [Chen et al., 2014a], a distributed diffusion strategy of the LMS type is developed in order to solve multitask estimation problems where the parameter space is assumed to be decomposed into two orthogonal subspaces, with one of the subspaces being common to all agents.

Multitask estimation problems with linearly related tasks are considered in [Nassif et al., 2016b]. It is assumed that each agent possesses its cost function of its own parameter vector and a set of linear equality constraints involving its parameter vector and the parameter vectors of its neighboring agents. A distributed projection based method is proposed in order to minimize the individual costs subject to all constraints in the network.

Note that, each task relatedness model requires the formulation of an appropriate global optimization problem that is solved in a distributed adaptive manner. In the following, we outline these optimization problems by considering first a general problem. Then we show how this general formulation encompasses the various multitask problems described so far. Let us consider the following problem:

$$\begin{aligned} \underset{\boldsymbol{u},\{\boldsymbol{w}_k,\boldsymbol{\epsilon}_k\}_{k=1}^N}{\text{minimize}} \quad & J^{\text{glob}}\left(\boldsymbol{u},\{\boldsymbol{w}_k,\boldsymbol{\epsilon}_k\}_{k=1}^N\right) \triangleq \sum_{k=1}^N J_k\left(\boldsymbol{w}_k\right) + \eta \sum_{k=1}^N R_k\left(\boldsymbol{w}_k,\{\boldsymbol{w}_\ell\}_{\ell\in\mathcal{I}_k\subseteq\mathcal{N}_k^-}\right), \\ \text{subject to} \quad & \boldsymbol{w}_k = \boldsymbol{X}_k\boldsymbol{u} + \boldsymbol{Y}_k\boldsymbol{\epsilon}_k, \quad k=1,\ldots,N \\ & h_k\left(\boldsymbol{w}_k,\{\boldsymbol{w}_\ell\}_{\ell\in\mathcal{J}_k\subseteq\mathcal{N}_k^-}\right) = \boldsymbol{0}, \quad k=1,\ldots,N. \end{aligned} \tag{1.2}$$

where $\boldsymbol{w}_k$ is the parameter vector to be estimated by agent k, η is a parameter controlling the importance of the regularization, R_k is a regularization function at agent k promoting relationships between its task and the tasks of its neighbors, $\boldsymbol{X}_k, \boldsymbol{Y}_k$ are some known matrices representing the parameter space at agent k, and h_k encodes constraints between the task at agent k and the tasks of its neighbors (see Figure 1.3 for an illustration). The notation $\mathcal{N}_k^-$ refers to the neighborhood of agent k, excluding k. Even though the cost $\sum_{k=1}^N J_k(\boldsymbol{w}_k)$ is separable, the cooperation between the agents is necessary due to the coupling between the tasks through the regularizers and the constraints. We show in Table 1.1 how the proper selection of the quantities $\boldsymbol{X}_k$, $\boldsymbol{Y}_k$,

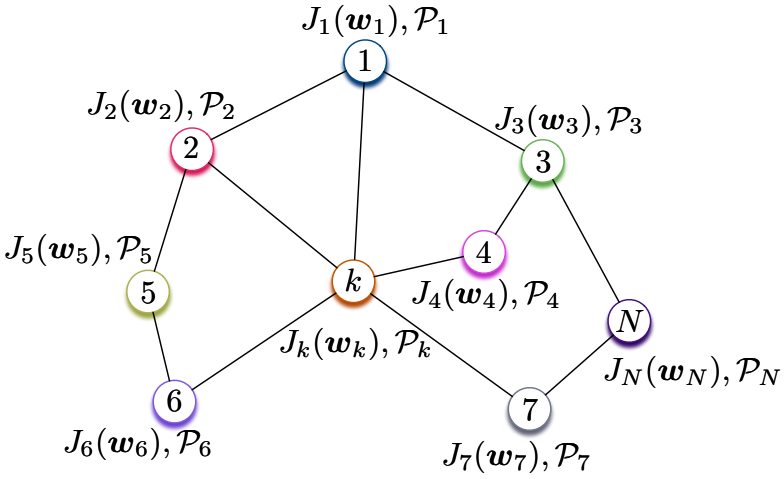


Figure 1.3: Multitask estimation over networks. The quantities $\boldsymbol{X}_k$, $\boldsymbol{Y}_k$, $R_k(\cdot)$, and $h_k(\cdot)$ relating the tasks in problem (1.2) are collected in $\mathcal{P}_k$. Each agent is seeking the sub-vector $\boldsymbol{w}_k$ in (1.2).

R_k, and h_k in problem (1.2) allows us to recover the various supervised multitask estimation problems described so far.

1.3 Objectives and Contributions

Distributed parameter estimation over multitask networks has recently become an active field of research. The objective of this dissertation is to contribute to the development and the investigation of distributed inference algorithms for multitask networks. At the end of this research work, we will be able to answer questions regarding the feasibility of distributed strategies that are able to exploit several forms of dependencies between tasks, and comment on their behavior and steady-state performance under several practical conditions and scenarios. The research work in this dissertation consists of four main parts, as described below. Except for the first part, this dissertation focuses on supervised scenarios where it is assumed that there are some prior knowledge that can be exploited to derive multitask strategies. Throughout this thesis, several multitask network applications (e.g., circular arcs localization, spectrum sensing in cognitive radio networks, and network flows estimation) will be considered to demonstrate the effectiveness of the proposed multitask strategies.

In the first part, we consider the unsupervised scenario. We assume that the single-task diffusion LMS algorithm (which is obtained by applying diffusion strategies to mean-square-error costs) is run, intentionally or unintentionally, in a multitask environment. Furthermore, we consider the realistic case where the data transmitted between agents is corrupted by noises. We conduct a mean and mean-square analysis in order to examine the consequences of violating the single-task hypothesis and the perfect information exchange assumption. The results show that the noises corrupting the regressors reduce the dynamic range of the step-sizes and induce a bias

Tasks relatedness	$\boldsymbol{X}_k$	$\boldsymbol{Y}_k$	$R_k(\cdot), \mathcal{I}_k$	$h_k(\cdot), \mathcal{J}_k$	Reference
Clustered network + Smooth graph signal	$\mathbf{0}$	$\boldsymbol{I}_M$	$\sum_{\ell \in \mathcal{I}_k} \rho_{k\ell} \|\boldsymbol{w}_k - \boldsymbol{w}_\ell\|^2$ with $\mathcal{I}_k = \mathcal{N}_k \setminus \mathcal{C}(k)$	$\boldsymbol{w}_k = \boldsymbol{w}_\ell$ for all $\ell \in \mathcal{J}_k$ with $\mathcal{J}_k = \mathcal{N}_k^- \cap \mathcal{C}(k)$	[Chen et al., 2014b,c, Nassif et al., 2016c], Chapter 3
Clustered network + Piecewise constant transitions	$\mathbf{0}$	$\boldsymbol{I}_M$	$\sum_{\ell \in \mathcal{I}_k} \rho_{k\ell} \|\boldsymbol{w}_k - \boldsymbol{w}_\ell\|_1$ with $\mathcal{I}_k = \mathcal{N}_k \setminus \mathcal{C}(k)$	$\boldsymbol{w}_k = \boldsymbol{w}_\ell$ for all $\ell \in \mathcal{J}_k$ with $\mathcal{J}_k = \mathcal{N}_k^- \cap \mathcal{C}(k)$	[Nassif et al., 2015, 2016d], Chapter 4
Partially overlapping hypothesis spaces	$\boldsymbol{\Theta}$	$\boldsymbol{\Theta}_\perp$	0	$\mathbf{0}$	[Chen et al., 2014a, Bogdanović et al., 2014]
Linear equalities relationships	$\mathbf{0}$	$\boldsymbol{I}_M$	0	$\boldsymbol{D}_{i,k}\boldsymbol{w}_k + \sum_{\ell \in \mathcal{J}_i} \boldsymbol{D}_{i,\ell}\boldsymbol{w}_\ell + \boldsymbol{b}_i$ with $\mathcal{J}_i \subseteq \left(\mathcal{J}_k = \mathcal{N}_k^-\right)$ and $i = 1, \ldots, j_k$	[Nassif et al., 2016b], Chapter 5
Space-varying tasks	$\boldsymbol{B}_k$	$\mathbf{0}$	0	$\mathbf{0}$	[Abdolee et al., 2014]

Table 1.1: Multitask estimation problems over networks: An overview. The notations $\mathcal{N}_k$ and $\mathcal{C}(k)$ denote the neighborhood of agent k and the cluster of agent k, respectively. The notations $\mathcal{N}_k \cap \mathcal{C}(k)$ and $\mathcal{N}_k \setminus \mathcal{C}(k)$ denote the neighborhood of agent k that are inside its cluster and outside its cluster, respectively. In these works, the MSE criterion was chosen as local cost $J_k(\cdot)$. $\rho_{k\ell}$ are regularization factors and j_k is the number of constraints that agent k is involved in.

in the mean. Furthermore, cooperating agents with distinct objectives will also induce a bias. In order to remove the effect of these nuisance factors and to make the algorithm more robust, we propose to adapt the combination coefficients by minimizing the instantaneous mean-square-error at each agent.

In the second part, we examine the behavior of the multitask diffusion LMS strategy proposed in [Chen et al., 2014b,c] under asynchronous conditions where networks are subject to various sources of uncertainties, including changing topology, random link failures, and agents turning on and off randomly for energy conservation. Under these conditions, the adaptive algorithm must be able to work. In order to examine these self-healing and self-organization properties, we first introduce a fairly general model for asynchronous behavior. Under this model, agents may stop updating their solutions, or may stop sending or receiving information in a random manner. Then, we carry out a detailed mean and mean-square-error analysis to examine how the asynchronous events interfere with the learning performance. More precisely, we derive closed form expressions to characterize the convergence behavior and steady-state performance of the agents acting asynchronously. The results show that sufficiently small step-sizes can still ensure both stability and performance.

In the third part, we propose a fully-distributed approach for multitask estimation based on sparsity promoting regularizers. This part addresses multitask learning problems where clusters of agents are interested in estimating their own parameter vector. It is assumed that the optimal models of adjacent clusters have a large number of identical components and a relatively small number of distinct components. Our approach relies on minimizing a global mean-square-error criterion regularized by non-differentiable terms to promote cooperation among neighboring clusters. Building on the proximal gradient methods, a general forward-backward splitting strategy of the LMS type is introduced. Then, it is specialized to the case of sparsity promoting regularizers. In order to achieve higher efficiency, we derive closed-form expression for the proximal operator of a weighted sum of ℓ_1-norms. We also provide conditions on the step-sizes that ensure the convergence of the algorithm in the mean and mean-square error sense. Finally, in order to guarantee an appropriate cooperation between clusters in a changing environment, we introduce a rule to adapt the regularization factors based on local instantaneous estimates.

In the fourth part, we consider multitask estimation problems with linearly related tasks. Each agent in the network is interested in estimating its own parameter vector. It is assumed that the models of neighboring agents are related according to a set of linear equality constraints. Each agent possesses its own MSE cost and the set of constraints involving its parameter vector and the parameter vectors of its neighboring agents. Based on the gradient projection method and diffusion strategies, we propose an adaptive multitask estimation algorithm of the LMS type in order to allow the network to optimize the individual costs subject to all constraints. Although the derivation is carried out for linear equality constraints, the technique can be applied to other forms of convex constraints. We derive conditions on the step-sizes ensuring the stability of the

algorithm and we provide closed form expressions to predict its learning behavior in the mean and mean-square-error sense. We show that for sufficiently small constant step-size, the expected distance between the estimates at each agent and the optimal value can be made arbitrarily small.

1.4 Thesis contents and Publications overview

This thesis is composed of six chapters as shown in Figure 1.4.

We begin the second chapter by reviewing the well-known diffusion LMS strategies for solving distributed single-task estimation problems over MSE networks. We then carry out a mean-square analysis of these strategies when they are run in multitask environment in the presence of noisy communication links. Next, we present a clustering technique that allows the agents to select the neighbors with which they can collaborate to estimate the same objective. Finally, we present simulation results to support the theoretical findings and to test the clustering technique in the presence of noisy links.
The main results presented in this chapter were published in:

- R. Nassif, C. Richard, and A. Ferrari. Estimation distribuée sur les réseaux multitâches en présence d'échanges d'informations bruitées. In *Actes du 25e colloque GRETSI sur le traitement du Signal et Images*, Lyon, France, September 2015.

- R. Nassif, C. Richard, J. Chen, A. Ferrari, and A. H. Sayed. Diffusion LMS over multitask networks with noisy links. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4583–4587, Shanghai, China, March 2016.

We begin the third chapter by recalling the *synchronous* diffusion adaptation strategy developed in [Chen et al., 2014b,c] for performing distributed estimation over multitask networks. Then, we introduce a general model for *asynchronous* behavior with random step-sizes, combination coefficients, and regularization factors. Next, we carry out a convergence analysis of the asynchronous multitask algorithm in the mean and mean-square-error sense, and we derive conditions for convergence. Finally, simulation results are presented to verify the theoretical findings and the framework is applied to network applications involving circular arcs localization.
The work presented in this chapter was published in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Performance analysis of multitask diffusion adaptation over asynchronous networks. In *Proc. Asilomar Conference on Signals, Systems and Computer*s, pages 788–792, Pacific Grove, CA, November 2014.

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion adaptation over asynchronous networks. *IEEE Transactions on Signal Processing*, 64(11):2835–2850, June 2016.

In the fourth chapter, we consider multitask estimation problems where the optimum parameter vectors to be estimated by neighboring clusters have a large number of similar components and a relatively small number of distinct entries. We formulate a global optimization problem and we then propose a diffusion forward-backward splitting approach with ℓ_1-norm and reweighted ℓ_1-norm co-regularizers to address it in a distributed manner. A closed-form expression for the proximal operator is then derived. We analyze the behavior of the proposed strategy and we propose an adaptive rule to guarantee an appropriate cooperation between clusters. Finally, we present simulation results to show the benefit of cooperation and the effectiveness of the proposed adaptive rule. We apply this strategy to cognitive radio networks involving spectrum estimation problems.
The work presented in this chapter was published in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion LMS with sparsity-based regularization. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3516–3520, Brisbane, Australia, April 2015.

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Proximal multitask learning over networks with sparsity inducing coregularization. *IEEE Transactions on Signal Processing*, 64(23):6329–6344, December 2016.

In the fifth chapter, we consider multitask estimation problems where the optimum parameter vectors to be estimated by neighboring agents are related according to a set of linear constraints. We first formulate the optimization problem and derive an adaptive centralized solution. Then, we reformulate the problem in order to obtain a structure amenable to distributed implementations and we derive an adaptive solution based on gradient projection principle and diffusion. Next, we conduct a mean and mean-square analysis. We present simulation results to validate the theoretical findings and show that sufficiently small step-sizes ensure convergence to the optimal solution. Finally, the algorithm is applied to solve network flows estimation problems and space varying field reconstruction problems.
The work presented in this chapter appears in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Diffusion LMS for multitask problems with local linear equality constraints. *Submitted to IEEE Transactions on Signal Processing.* Also available as arXiv:1610.02943, October 2016.

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Distributed learning over multitask networks with linearly related tasks. In *Proc. Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, November 2016.

Finally, in the sixth chapter we conclude the thesis by summarizing the main contributions and suggest some future research directions.

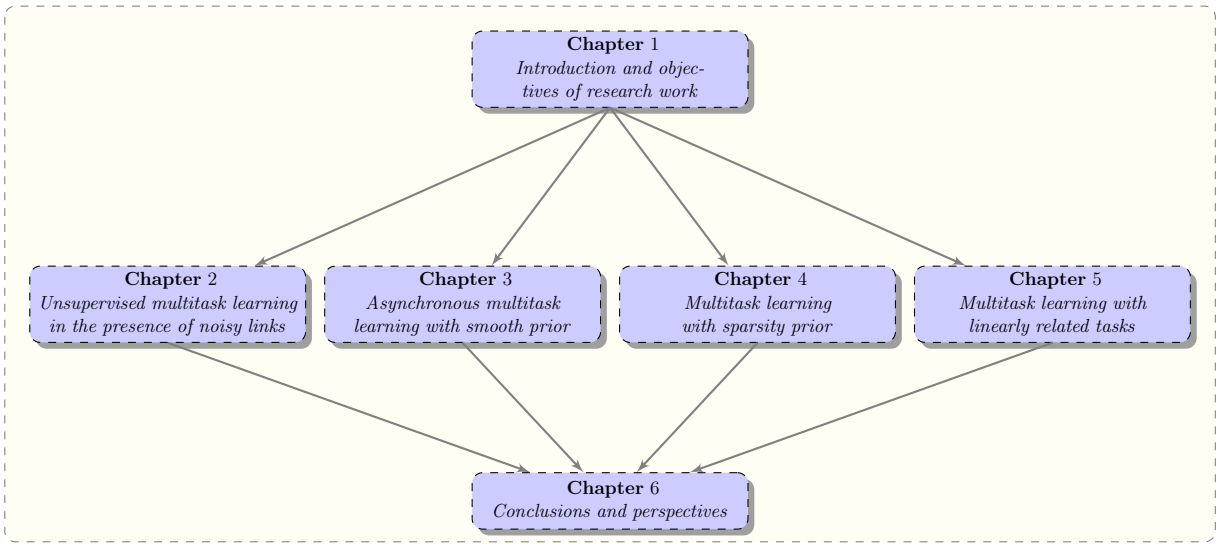


Figure 1.4: Thesis organisation

CHAPTER 2

Diffusion LMS over multitask networks with noisy links

The diffusion LMS is an efficient strategy for solving distributed estimation problems over single-task networks with agents observing data arising from the same optimum model. However, in some applications, the optimum parameter vectors may not be the same for all agents. Moreover, the data exchanged between agents can be subject to quantization errors and additive noise over the communication links. In this chapter, we first present an overview on diffusion LMS strategies over single-task MSE networks. Later, we assume that the single-task hypothesis is violated. We conduct a mean and mean-square analysis in order to quantify the performance of the single-task diffusion LMS when it is run, intentionally or unintentionally, in a multitask environment in the presence of noisy communication links. To reduce the impact of the nuisance factors, namely, the cooperation between agents with distinct objectives and the noisy links, we introduce an improved strategy that allows the agents to promote or reduce exchanges of information with their neighbors.

The main results established in this chapter were published in:

- R. Nassif, C. Richard, and A. Ferrari. Estimation distribuée sur les réseaux multitâches en présence d'échanges d'informations bruitées. In *Actes du 25e colloque GRETSI sur le traitement du Signal et Images*, Lyon, France, September 2015. (Also available at http://www.cedric-richard.fr/Articles/nassif2015estimation.pdf)

- R. Nassif, C. Richard, J. Chen, A. Ferrari, and A. H. Sayed. Diffusion LMS over multitask networks with noisy links. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4583–4587, Shanghai, China, March 2016. (Also available at http://www.cedric-richard.fr/Articles/nassif2016diffusion.pdf)

2.1 Introduction

Most of the existing literature on distributed estimation over networks focuses on consensus problems where a collection of agents need to converge asymptotically toward a common parameter vector. In some cases (e.g., single-task MSE networks described in Section 2.2), the objective at each agent can be estimated by relying solely on local computations without cooperation with neighboring agents. In such non-cooperative strategies, it is well known that the quality of the estimates will depend on the quality of the data at the agents, i.e., agents with noisier data will perform worse than agents with cleaner data. Distributed estimation strategies are attractive in this case since they allow the agents to exchange information with their neighbors in an appropriate manner to improve their own estimates. Among the existing cooperation rules for single-task problems, we are interested in diffusion strategies [Lopes and Sayed, 2008, Cattivelli and Sayed, 2010, Sayed et al., 2013, Sayed, 2014a,b,c] since they are scalable, robust, and enable continuous learning. These strategies estimate a common parameter vector by minimizing, in a distributed manner, a global cost function that aggregates the individual costs. The resulting network is fully distributed and adaptive in the sense that agents perform local computation tasks, exchange information with their neighbors, and are able to adapt continuously in response to concept drifts. The performance of the resulting adaptive network has been studied extensively in the literature (see e.g. [Sayed, 2014a,b,c] and the references therein). In the current chapter, we first recall the diffusion LMS strategies [Lopes and Sayed, 2008, Cattivelli and Sayed, 2010, Sayed et al., 2013, Sayed, 2014c] resulting from applying the diffusion strategies to mean-square-error costs, which are of paramount importance in the context of estimation, adaptation, and learning over networks [Sayed et al., 2013]. Then, we study the behavior of these single-task algorithms in a multitask environment in the presence of noisy communication links.

In multitask networks, agents are sensing data arising from different models, and hence, are interested in estimating several parameter vectors. Several research efforts have been focused on analyzing the performance of diffusion strategies when they are run, intentionally or unintentionally, in a multitask environment. It is shown in [Chen and Sayed, 2013], for example, that the diffusion iterates converge to a Pareto optimal solution when the optimization problem consists of a sum of individual costs with possibly different minimizers. It is further shown in [Chen et al., 2015a] that, when the tasks are sufficiently similar to each other, the single-task diffusion LMS can still perform better (in the mean-square-error sense) than non-cooperative strategies, despite the bias resulting from cooperating agents with different objectives. To avoid poor results resulting from cooperation between neighbors with sufficiently different objectives, extended diffusion strategies with a clustering step are proposed in [Zhao and Sayed, 2012, 2015d, Chen et al., 2015a, Khawatmi et al., 2015] to enable agents to identify which neighbors belong to the same cluster and which neighbors should be ignored.

Usually, the exchange of raw data and local estimates between agents may be corrupted by

Table 2.1: List of the main symbols and notations used in Chapter 2.

M	Length of the parameter vectors
N	Number of agents in the network
$\mathcal{N}_k$	Neighborhood of agent k, i.e., the set of agents that are connected to k by edges, including k
$\mathcal{N}_k^-$	Neighborhood of agent k, excluding k
$\boldsymbol{w}_k$	Parameter vector at agent k
$\boldsymbol{w}_k^o$	Optimum parameter vector at agent k
$\boldsymbol{w}_b$	Network block parameter vector
$\boldsymbol{w}_b^o$	Network block optimum parameter vector

noises over the communication links. Useful results dealing with the consequences of noisy communications on the diffusion LMS behavior are presented in [Abdolee and Champagne, 2011, Khalili et al., 2012, Tu and Sayed, 2011a, Zhao et al., 2012] for single-task environments. In this chapter, we extend these works by considering the more general case of multitask environments. We conduct a mean and mean-square-error analysis in order to reveal the degradation in the performance resulting from running the single-task diffusion LMS algorithm in multitask environments in the presence of noisy communication links. The resulting analytical expressions allow us to identify the influence of each nuisance factor on the dynamics of the network, on the biases in the weight estimates, and on the mean-square-error performance. Since the mean-square-error performance depends on the combination coefficients, we also show how these coefficients can be adjusted efficiently during the learning process in order to enable agents to cooperate only with neighbors sharing the same objective, and to simultaneously reduce the effect of exchanging information through noisy links.

Before starting, we list in Table 2.1 some of the main symbols and notations used in this chapter. Other symbols will be defined in the context where they are used.

2.2 Diffusion LMS over single-task networks

Consider a strongly connected network[1] consisting of N agents, labeled $k = 1, 2, \ldots, N$. At each time instant i, each agent in the network collects a zero-mean scalar measurement $d_k(i)$ and a zero-mean $M \times 1$ regression vector $\boldsymbol{x}_k(i)$ with positive covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\, \boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i) > \boldsymbol{0}$. The data $\{d_k(i), \boldsymbol{x}_k(i)\}$ are assumed to satisfy the linear regression model:

$$d_k(i) = \boldsymbol{x}_k^\top(i)\boldsymbol{w}^o + z_k(i), \quad i \geq 0, \quad k = 1, \ldots, N, \tag{2.1}$$

[1] A network is said to be strongly connected if it is connected, i.e., there exists a path between any pair of vertices, and contains at least one self-loop.

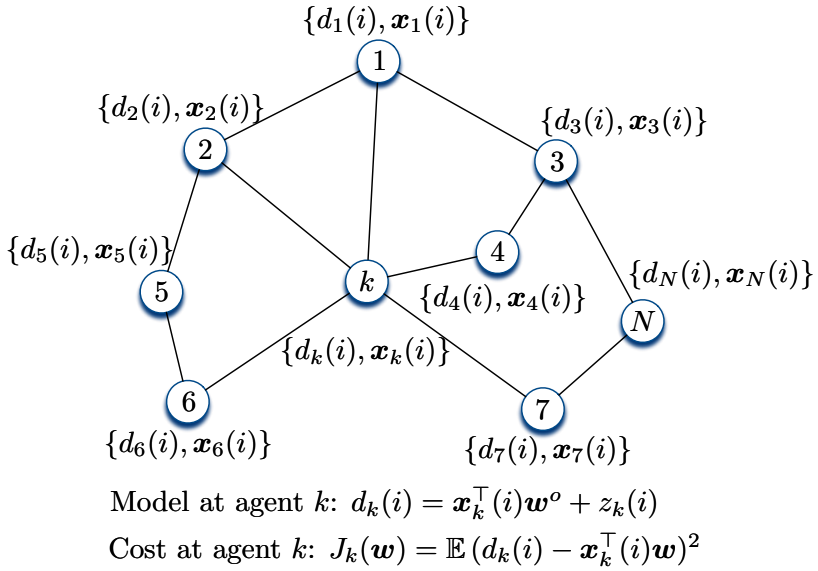


Figure 2.1: Illustration of a single-task MSE network.

for some unknown $M \times 1$ parameter vector $\boldsymbol{w}^o$ that the agents wish to estimate, and where $z_k(i)$ is a measurement noise with zero-mean and variance $\sigma_{z,k}^2$, independent of $\boldsymbol{x}_\ell(j)$ for all ℓ and j, and independent of $z_\ell(j)$ for $\ell \neq k$ or $i \neq j$. The mean-square-error cost is associated with each agent k, namely,

$$J_k(\boldsymbol{w}) \triangleq \mathbb{E}\,(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w})^2. \tag{2.2}$$

Strongly-connected networks with agents receiving streaming data according to (2.1) and seeking to estimate a common parameter vector $\boldsymbol{w}^o$ by adopting mean-square-error costs $J_k(\boldsymbol{w})$ defined by (2.2), are referred to as single-task mean-square-error (MSE) networks [Sayed, 2014a]. An illustrative example is provided in Figure 2.1.

Although this dissertation focuses on mean-square-error costs, most of the arguments and derivations presented here can be extended to more general cost functions. Linear data models and quadratic costs of the form (2.1) and (2.2) are common and useful in many applications, such as target localization and spectral sensing (see e.g. [Sayed, 2014c, Section 2] for further applications).

Since the individual cost $J_k(\boldsymbol{w})$ in (2.2) is strongly convex, it has a unique global minimum $\boldsymbol{w}_k^o$ given by:

$$\boldsymbol{w}_k^o = \boldsymbol{R}_{x,k}^{-1}\boldsymbol{r}_{dx,k}, \tag{2.3}$$

where $\boldsymbol{r}_{dx,k} \triangleq \mathbb{E}\, d_k(i)\boldsymbol{x}_k(i)$. If we multiply both sides of (2.1) by $\boldsymbol{x}_k(i)$, and take expectations, we find that the unknown parameter vector $\boldsymbol{w}^o$ in (2.1) satisfies the same expression as $\boldsymbol{w}_k^o$ in (2.3), so that we must have:

$$\boldsymbol{w}_k^o = \boldsymbol{w}^o, \qquad k = 1, \ldots, N. \tag{2.4}$$

As a consequence, under linear regression models of the form (2.1), the individual MSE cost in (2.2) allows agent k to recover $\boldsymbol{w}^o$ exactly.

2.2.1 Non-cooperative adaptive solution

In a non-cooperative mode of operation, each agent operates individually to estimate $\boldsymbol{w}^o$. Instead of applying (2.3), iterative algorithms can be applied to recover the same solution $\boldsymbol{w}^o$. Let $\boldsymbol{w}_k(i)$ denote the estimate of $\boldsymbol{w}^o$ at agent k at time instant i. Starting with any initial condition $\boldsymbol{w}_k(0)$, the gradient descent recursion takes the following form (by omitting the factor of two in the gradient):

$$\boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k(\boldsymbol{r}_{dx,k} - \boldsymbol{R}_{x,k}\boldsymbol{w}_k(i)), \quad i \geq 0, \tag{2.5}$$

where μ_k is a constant step-size parameter chosen within the interval $(0, 2/\lambda_{\max}(\boldsymbol{R}_{x,k}))$ in order to guarantee convergence [Sayed, 2008].

The closed form expression (2.3) and the recursion (2.5) for determining $\boldsymbol{w}^o$ require knowledge of the second-order moments $\{\boldsymbol{r}_{dx,k}, \boldsymbol{R}_{x,k}\}$. This information is rarely available beforehand; instead agent k senses realizations $\{d_k(i), \boldsymbol{x}_k(i)\}$ of random processes with moments $\boldsymbol{r}_{dx,k}$ and $\boldsymbol{R}_{x,k}$. To address this lack of information, it is necessary to derive a scheme that allows each agent to use these realizations to approximate the unavailable moments $\{\boldsymbol{r}_{dx,k}, \boldsymbol{R}_{x,k}\}$. In this case, an adaptive solution is possible since the approximations are based on realizations, and any changes in the underlying statistical moments end up being reflected in the data. Several constructions for approximating the statistical moments are possible, with different constructions leading to different adaptive algorithms [Sayed, 2008], such as LMS algorithm, normalized least-mean-squares (NLMS) algorithm, and recursive-least-squares (RLS) algorithm. One of the simplest choices is to drop the expectations from the definitions of $\{\boldsymbol{r}_{dx,k}, \boldsymbol{R}_{x,k}\}$ and to use the following instantaneous approximations:

$$\boldsymbol{R}_{x,k} \approx \boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i), \qquad \boldsymbol{r}_{dx,k} \approx \boldsymbol{x}_k(i)d_k(i), \tag{2.6}$$

and the corresponding gradient descent recursion (2.5) becomes:

$$\boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k\boldsymbol{x}_k(i)(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)), \quad i \geq 0. \tag{2.7}$$

The stochastic-gradient algorithm obtained in (2.7) is the well-known least-mean-squares (LMS) adaptive algorithm.

It is known that the estimates generated at agent k by the stand-alone filter (2.7) will converge to $\boldsymbol{w}^o$ in the mean if the step-size μ_k is chosen within the interval $(0, 2/\lambda_{\max}(\boldsymbol{R}_{x,k}))$. Moreover, for sufficiently small step-sizes (ensuring mean-square stability), it holds that the filter mean-square-deviation (MSD), which measures how far $\boldsymbol{w}_k(i)$ is from $\boldsymbol{w}^o$ in the mean-square sense at steady-state, can be approximated as [Sayed, 2008]:

$$\begin{aligned} \text{MSD}_k &\triangleq \lim_{i\to\infty} \mathbb{E}\,\|\boldsymbol{w}^o - \boldsymbol{w}_k(i)\|^2 \\ &\approx \mu_k\sigma_{v,k}^2 M/2. \end{aligned} \tag{2.8}$$

When the step-sizes are equal, i.e., $\mu_k = \mu$ for all k, it can be observed from (2.8) that the quality of the estimators at the agents depends on the quality of their data; agents with noisier data (larger $\sigma_{z,k}^2$) will perform worse than agents with cleaner data. However, since all agents are observing data arising from the same underlying model $\boldsymbol{w}^o$, it is expected that an appropriate cooperation among agents can help enhance their individual performance.

One way to achieve such cooperation is to develop adaptive algorithms that enable the agents to solve, in a distributed manner, the global optimization problem:

$$\underset{\boldsymbol{w}}{\text{minimize}}\ J^{\text{glob}}(\boldsymbol{w}) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}) = \sum_{k=1}^{N} \mathbb{E}\,(d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w})^2, \tag{2.9}$$

for which $\boldsymbol{w}^o$ is a unique global solution. As we mentioned in the previous chapter, there are three well known strategy families to solve this problem: incremental strategies, consensus strategies, and diffusion strategies. In the sequel, we review diffusion strategies.

2.2.2 Cooperative adaptive diffusion strategy

There are several variations of the adaptive diffusion strategy. Let $\mathcal{N}_k$ denote the neighborhood of agent k, i.e., the set of agents that are connected to agent k through an edge, including k. For general optimization problems involving individual costs $\{J_k(\boldsymbol{w})\}$ that are not necessarily quadratic, the adaptive adapt-then-combine (ATC) diffusion strategy for solving

$$\underset{\boldsymbol{w}}{\text{minimize}}\ J^{\text{glob}}(\boldsymbol{w}) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}) \tag{2.10}$$

takes the following form at each agent k [Chen and Sayed, 2012, Sayed, 2014a]:

$$\text{(general ATC)} \qquad \begin{aligned} \boldsymbol{\psi}_k(i+1) &= \boldsymbol{w}_k(i) - \mu_k \sum_{\ell \in \mathcal{N}_k} c_{\ell k}\, \widehat{\nabla_w J_\ell}(\boldsymbol{w}_k(i)) \\ \boldsymbol{w}_k(i+1) &= \sum_{\ell \in \mathcal{N}_k} a_{\ell k}\, \boldsymbol{\psi}_\ell(i+1) \end{aligned} \tag{2.11}$$

where $\widehat{\nabla_w J_\ell}(\cdot)$ is an approximation of the true gradient vector $\nabla_w J_\ell(\cdot)$, μ_k is a small constant step-size parameter, and $\{a_{\ell k}, c_{\ell k}\}$ are non-negative coefficients chosen by the designer to satisfy the following conditions:

$$a_{\ell k} \geq 0, \quad \sum_{\ell=1}^{N} a_{\ell k} = 1, \ \text{and}\ a_{\ell k} = 0 \ \text{if}\ \ell \notin \mathcal{N}_k, \tag{2.12}$$

$$c_{\ell k} \geq 0, \quad \sum_{k=1}^{N} c_{\ell k} = 1, \ \text{and}\ c_{\ell k} = 0 \ \text{if}\ \ell \notin \mathcal{N}_k. \tag{2.13}$$

According to the above conditions, if we collect the coefficients $\{a_{\ell k}, c_{\ell k}\}$ into $N \times N$ matrices $\boldsymbol{A} \triangleq [a_{\ell k}]$ and $\boldsymbol{C} \triangleq [c_{\ell k}]$, we obtain a left-stochastic[2] matrix and a right-stochastic[3] matrix,

[2] A matrix $\boldsymbol{A}$ with non-negative entries is said to be left-stochastic if it satisfies $\boldsymbol{A}^{\top}\mathbb{1} = \mathbb{1}$.
[3] A matrix $\boldsymbol{C}$ with non-negative entries is said to be right-stochastic if it satisfies $\boldsymbol{C}\mathbb{1} = \mathbb{1}$.

respectively. There are several rules to select these coefficients, such as the averaging rule whereby nodes simply average data from their neighbors:

$$a_{\ell k} = \begin{cases} 1/\text{card}\{\mathcal{N}_k\}, & \text{if } \ell \in \mathcal{N}_k \\ 0, & \text{otherwise} \end{cases} \tag{2.14}$$

and the Metropolis rule given by:

$$a_{\ell k} = \begin{cases} 1/\max\left\{\text{card}\{\mathcal{N}_k\}, \text{card}\{\mathcal{N}_\ell\}\right\}, & \text{if } \ell \in \mathcal{N}_k^- \\ 1 - \sum_{\ell \in \mathcal{N}_k^-} a_{\ell k} & \text{if } \ell = k \\ 0, & \text{otherwise.} \end{cases} \tag{2.15}$$

See [Sayed, 2014c, Table 7] for a listing of further common rules. Note that the right-stochastic matrix $\boldsymbol{C}$ can be obtained by transposing the left-stochastic matrix $\boldsymbol{A}$. Note further that, instead of using constant weights, one may add a third step to the diffusion strategy (2.11) in order to set the combination coefficients in an online manner and to enable agents to assign more or less importance to the estimates arriving from their neighbors according to the quality of their data [Sayed, 2014c].

Starting from any initial condition $\boldsymbol{w}_k(0)$, at every time instant $i \geq 0$, the ATC strategy (2.11) performs two steps. In the first step, also called adaptation step, agent k receives from its neighbors their gradient vector approximations, combines this information, and uses it to update its estimate $\boldsymbol{w}_k(i)$ to an intermediate value $\boldsymbol{\psi}_k(i+1)$. All other agents in the network are performing simultaneously a similar information exchange step. The second step is a combination step where agent k combines the intermediate iterates $\boldsymbol{\psi}_\ell(i+1)$ of its neighbors to obtain $\boldsymbol{w}_k(i+1)$. Again, all other agents are performing simultaneously a similar combination step.

A similar implementation can be obtained by switching the order of the adaptation and combination step in (2.11). In the combine-then-adapt (CTA) implementation, agent k first combines the previous estimates of its neighbors to obtain the intermediate estimate $\boldsymbol{\psi}_k(i)$, and then updates this intermediate estimate with a stochastic gradient step:

$$\text{(general CTA)} \qquad \begin{aligned} \boldsymbol{\psi}_k(i) &= \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{w}_\ell(i) \\ \boldsymbol{w}_k(i+1) &= \boldsymbol{\psi}_k(i) - \mu_k \sum_{\ell \in \mathcal{N}_k} c_{\ell k} \widehat{\nabla_w J_\ell}(\boldsymbol{\psi}_k(i)). \end{aligned} \tag{2.16}$$

Other forms of diffusion strategies are also possible by restricting the exchange of information to the combination step; this is the special case when $\boldsymbol{C} = \boldsymbol{I}_N$. The non-cooperative strategy can be also obtained by setting $\boldsymbol{C} = \boldsymbol{A} = \boldsymbol{I}_N$.

Returning to the MSE costs (2.2) and using the simple instantaneous approximations (2.6) for the second-order moments, we obtain an approximate for the gradient vector at agent ℓ:

$$\widehat{\nabla_w J_\ell}(\boldsymbol{w}) = -2\boldsymbol{x}_\ell(i)(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}). \tag{2.17}$$

Substituting into the ATC and CTA strategies (2.11) and (2.16), we arrive at the diffusion LMS strategies. Table 2.2 lists several forms of these strategies [Lopes and Sayed, 2008, Cattivelli and Sayed, 2010, Sayed et al., 2013, Sayed, 2014c]. It is known that the iterates $\{\boldsymbol{w}_k(i)\}$

Table 2.2: Variations of diffusion LMS strategies.

ATC strategy	$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)\big(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)\big)$ $\boldsymbol{w}_k(i+1) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_\ell(i+1)$	(2.18)
ATC strategy (no information exchange)	$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k\boldsymbol{x}_k(i)\big(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\big)$ $\boldsymbol{w}_k(i+1) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_\ell(i+1)$	(2.19)
CTA strategy	$\boldsymbol{\psi}_k(i) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_\ell(i)$ $\boldsymbol{w}_k(i+1) = \boldsymbol{\psi}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)\big(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{\psi}_k(i)\big)$	(2.20)
CTA strategy (no information exchange)	$\boldsymbol{\psi}_k(i) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_\ell(i)$ $\boldsymbol{w}_k(i+1) = \boldsymbol{\psi}_k(i) + \mu_k\boldsymbol{x}_k(i)\big(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{\psi}_k(i)\big)$	(2.21)

converge in the mean to the solution $\boldsymbol{w}^o$ if the step-sizes $\{\mu_k\}$ are chosen within the interval $\Big(0, 2/\lambda_{\max}\Big(\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{R}_{x,\ell}\Big)\Big)$ and that the network MSD defined as $\text{MSD}_{\text{net}} = \frac{1}{N}\sum_{k=1}^N \text{MSD}_k$ improves by N-fold compared to the non-cooperative case. Note that in [Sayed, 2014c, Section 7.1], it is shown that the ATC implementation performs better than the CTA implementation. For this reason, we shall consider in the sequel the ATC implementations.

In Appendix B.1, we present the theoretical foundations of the diffusion LMS strategies listed in Table 2.2. In particular, we show how these strategies can be derived by minimizing the cost function (2.9) based on a completion-of-squares argument, followed by a stochastic approximation step, and an incremental approximation step [Cattivelli and Sayed, 2010, Sayed, 2014c]. The material presented in this Appendix serves for better understanding the context and development in the subsequent chapters. In Appendix B.2, we briefly review the mean and mean-square convergence analysis of the ATC diffusion LMS strategy. The material presented in this Appendix serves as a benchmark allowing the reader to identify the influence of the various models considered throughout the thesis. We refer to [Sayed et al., 2013, Sayed, 2014c] for a detailed discussion on the properties of the diffusion LMS algorithm, its theoretical foundations, its performance, and its applications.

2.3 Diffusion LMS over multitask networks with noisy links

Again, we consider a connected network of N agents, where at each time instant i, each agent k collects a zero-mean scalar measurement $d_k(i)$ and a zero-mean $M\times 1$ regression vector $\boldsymbol{x}_k(i)$ with

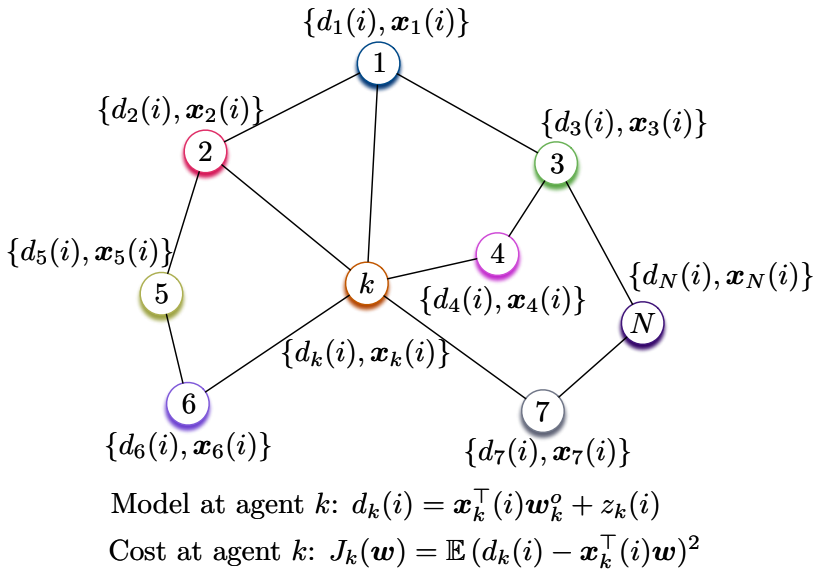


Figure 2.2: Illustration of a multitask MSE network.

positive covariance matrix $\boldsymbol{R}_{x,k}$. In a multitask environment, the data at agent k are assumed to be related to an $M \times 1$ unknown parameter vector $\boldsymbol{w}_k^o$ via the linear data model:

$$d_k(i) = \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k^o + z_k(i), \quad i \geq 0, \tag{2.22}$$

where $z_k(i)$ is a zero-mean measurement noise of variance $\sigma_{z,k}^2$. The noise process is assumed to be temporally white and spatially independent. The problem is to estimate $\boldsymbol{w}_k^o$ at each agent k. To solve this problem, we associate with each agent k a mean-square-error cost of the form (2.2). We refer to networks with agents receiving streaming data related according to (2.22) and seeking to estimate $\{\boldsymbol{w}_k^o\}$ by adopting the mean-square-error costs $J_k(\boldsymbol{w})$ defined by (2.2) as multitask MSE networks. We provide an illustrative example in Figure 2.2.

The single-task environment described in Section 2.2 can be viewed as a particular case of the multitask environment where the optimum parameter vectors are the same across the network, namely, $\boldsymbol{w}_k^o = \boldsymbol{w}^o$ for all k. In this case, it was shown that the use of a diffusion LMS strategy that minimizes, in a fully-distributed manner, the aggregate cost (2.9) allows to recover $\boldsymbol{w}^o$ (without biases) and improves the estimation accuracy relative to the non-cooperative solution by improving the mean-square-error performance of the network [Sayed, 2014c]. In a multitask environment, it happens that the local MSE costs $J_k(\boldsymbol{w})$ are not minimized at the same location and the diffusion iterates lead to a Pareto optimum solution for (2.9) [Chen and Sayed, 2013]. The work [Chen et al., 2015a] assesses the performance limits of diffusion strategies (2.18)–(2.21) when they are run, intentionally or unintentionally, in a multitask environment, and analyzes the critical role of the distance between the tasks $\{\boldsymbol{w}_k^o\}$. Indeed, in some situations, due to inaccurate models, or minor differences between tasks that are neglected intentionally, one may apply the diffusion LMS to a multitask environment. It is shown that, in these situations,

the estimates generated by the algorithm are biased and that the biased solutions may still be beneficial compared to non-cooperative strategies, provided that the tasks are sufficiently close to each other.

In real environment, the data exchanged between agents can be subject to quantization errors and additive noise over the communication links. It becomes necessary to incorporate these aspects in the performance analysis and to make the estimation algorithm more robust to the perturbations. Thus, in the following, we shall study the performance degradation resulting from applying the ATC diffusion LMS (2.18)–(2.19) to multitask environments in the presence of noisy communication links and show how the combination weights can be set in order to reduce the effect of the nuisance factors.

Each step of the ATC algorithm (2.18) involves the transmission of information from node $\ell \in \mathcal{N}_k$ to node k. In the presence of noisy communication links, the ATC diffusion algorithm (2.18) becomes:

$$\begin{aligned} \boldsymbol{\psi}_k(i+1) &= \boldsymbol{w}_k(i) + \mu_k \sum_{\ell \in \mathcal{N}_k} c_{\ell k} \boldsymbol{x}_{\ell k}(i) \left(d_{\ell k}(i) - \boldsymbol{x}_{\ell k}^{\top}(i) \boldsymbol{w}_k(i) \right), \\ \boldsymbol{w}_k(i+1) &= \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell k}(i+1), \end{aligned} \tag{2.23}$$

where $\boldsymbol{x}_{\ell k}(i)$, $d_{\ell k}(i)$, and $\boldsymbol{\psi}_{\ell k}(i+1)$ are the noisy data received by node k from its neighbor ℓ. For modeling noisy communication links, we adopt the model proposed in [Sayed, 2014c, Zhao et al., 2012]:

$$d_{\ell k}(i) = d_{\ell}(i) + z_{d,\ell k}(i), \tag{2.24}$$

$$\boldsymbol{x}_{\ell k}(i) = \boldsymbol{x}_{\ell}(i) + \boldsymbol{z}_{x,\ell k}(i), \tag{2.25}$$

$$\boldsymbol{\psi}_{\ell k}(i) = \boldsymbol{\psi}_{\ell}(i) + \boldsymbol{z}_{\psi,\ell k}(i), \tag{2.26}$$

where $z_{d,\ell k}(i)$ is a scalar noise signal, $\boldsymbol{z}_{x,\ell k}(i)$ and $\boldsymbol{z}_{\psi,\ell k}(i)$ are noise vectors of dimension $M \times 1$. Note that this model is more general than the one adopted in [Abdolee and Champagne, 2011, Khalili et al., 2012] where the diffusion LMS algorithm without exchange of gradient information (2.19) is considered, and which can be obtained from (2.18) by setting $\boldsymbol{C} = \boldsymbol{I}_N$.

2.4 Stochastic performance analysis

Throughout this dissertation, we shall study the performance in the mean and mean-square-error sense of several adaptive distributed algorithms. Studying the performance of such algorithms is a challenging task since the systems are stochastic, nonlinear, and the agents influence each other behavior. In order to make the analysis more tractable, several reasonable and commonly used assumptions need to be introduced whenever necessary. The analysis in the mean-square-error sense relies on the energy conservation framework [Al-Naffouri and Sayed, 2003, Sayed, 2008], which allows us to derive expressions for the MSD by analyzing how the energy (measured in

terms of error variances) flows across the network. In several cases, we end up with closed form expressions allowing us to predict the behavior of the distributed adaptive algorithms. Furthermore, step-size conditions ensuring stability of the algorithms will also be established. Although the energy conservation framework is adopted throughout this dissertation, the assumptions and the technical arguments used in each chapter depend on the optimization approach and the context. For example, the proofs for proximal based approaches differ from the proofs for gradient approaches and the arguments used for asynchronous networks differ from the ones in the synchronous case. Thus, a detailed analysis is provided in each chapter.

Under some reasonable assumptions on the measurement data and noise signals, we assess in the current section the performance of the distributed single-task diffusion LMS algorithm under several conditions, including multitask environments and noisy communication links. In particular, the following assumptions are introduced.

Assumption 2.1. (Independent regressors) *The regressors $\boldsymbol{x}_k(i)$ arise from a zero-mean random process that is temporally white and spatially independent.*

Assumption 2.2. (Properties of the link noises) *The noises $z_{d,\ell k}(i), \boldsymbol{z}_{x,\ell k}(i)$, and $\boldsymbol{z}_{\psi,\ell k}(i)$ are temporally white and spatially independent zero-mean random variables.*

We denote by $\sigma^2_{zd,\ell k}$, $\boldsymbol{R}_{zx,\ell k}$, and $\boldsymbol{R}_{z\psi,\ell k}$ their variances and covariance matrices, respectively. These moments are zero if $\ell \notin \mathcal{N}_k$ or $\ell = k$.

Assumption 2.3. (Mutually independent variables) *The link noises $\{z_{d,mn}(i_1)\}$, $\{\boldsymbol{z}_{x,pq}(i_2)\}$, and $\{\boldsymbol{z}_{\psi,st}(i_3)\}$, the regression data $\{\boldsymbol{x}_k(i_4)\}$, and the measurement noise $\{z_\ell(i_5)\}$ are mutually independent for all $\{k,\ell,m,n,p,q,s,t\}$ and $\{i_1,i_2,i_3,i_4,i_5\}$.*

Let $\widetilde{\boldsymbol{w}}_k(i)$ denote the weight error vector at node k and iteration i, namely,

$$\widetilde{\boldsymbol{w}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_k(i), \tag{2.27}$$

and let $\widetilde{\boldsymbol{w}}_b(i)$ denote the network block weight error vector at iteration i, of size $N \times 1$ with blocks of size $M \times 1$, namely,

$$\widetilde{\boldsymbol{w}}_b(i) \triangleq \operatorname{col}\left\{\widetilde{\boldsymbol{w}}_1(i), \ldots, \widetilde{\boldsymbol{w}}_N(i)\right\}, \tag{2.28}$$

where $\operatorname{col}\{\cdot\}$ stacks the column vector entries on top of each other. In the following, we derive a recursion characterizing the evolution of (2.28). This recursion is the launching point for the analysis in the mean and mean-square-error sense.

2.4.1 Weight error vector recursion

Using data model (2.22), the noisy data $\{d_{\ell k}(i), \boldsymbol{x}_{\ell k}(i)\}$ in (2.24)–(2.25) at node k can be related to the unknown vector $\boldsymbol{w}_\ell^o$ at node ℓ via the relation:

$$d_{\ell k}(i) = \boldsymbol{x}_{\ell k}^\top(i)\boldsymbol{w}_\ell^o + z_{\ell k}(i), \tag{2.29}$$

where we introduced the scalar zero-mean noise signal:

$$z_{\ell k}(i) \triangleq z_\ell(i) + z_{d,\ell k}(i) - \boldsymbol{z}_{x,\ell k}^{\top}(i)\,\boldsymbol{w}_\ell^o, \quad \ell \in \mathcal{N}_k^- \tag{2.30}$$

with variance:

$$\sigma_{z,\ell k}^2 = \sigma_{z,\ell}^2 + \sigma_{zd,\ell k}^2 + (\boldsymbol{w}_\ell^o)^{\top}\boldsymbol{R}_{zx,\ell k}\boldsymbol{w}_\ell^o. \tag{2.31}$$

To unify the notation, we define $z_{kk}(i) = z_k(i)$. Using (2.29), the estimation error that appears in the adaptation step of (2.23) can be written as:

$$d_{\ell k}(i) - \boldsymbol{x}_{\ell k}^{\top}(i)\boldsymbol{w}_k(i) = \boldsymbol{x}_{\ell k}^{\top}(i)\widetilde{\boldsymbol{w}}_k(i) + \boldsymbol{x}_{\ell k}^{\top}(i)\boldsymbol{u}_{\ell k}^o + z_{\ell k}(i), \tag{2.32}$$

where

$$\boldsymbol{u}_{\ell k}^o \triangleq \boldsymbol{w}_\ell^o - \boldsymbol{w}_k^o. \tag{2.33}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the adaptation step in (2.23) and using relation (2.32), we can write:

$$\begin{aligned}\boldsymbol{w}_k^o - \boldsymbol{\psi}_k(i+1) = \left(\boldsymbol{I}_M - \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_{\ell k}(i)\boldsymbol{x}_{\ell k}^{\top}(i)\right)\widetilde{\boldsymbol{w}}_k(i) - \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_{\ell k}(i)\boldsymbol{x}_{\ell k}^{\top}(i)\boldsymbol{u}_{\ell k}^o \\ - \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_{\ell k}(i) z_{\ell k}(i).\end{aligned} \tag{2.34}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the combination step in (2.23) and using (2.26), we obtain:

$$\widetilde{\boldsymbol{w}}_k(i+1) = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\left(\boldsymbol{w}_\ell^o - \boldsymbol{\psi}_\ell(i+1)\right) + \left(\boldsymbol{w}_k^o - \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_\ell^o\right) - \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{z}_{\psi,\ell k}(i+1) \tag{2.35}$$

Relations (2.34)–(2.35) can be described more compactly by collecting the information from across the network into block vectors and matrices. Let $\widetilde{\boldsymbol{w}}_b(i)$ and $\boldsymbol{w}_b^o$ denote the block weight error vector and the block optimum weight vector, all of size $N \times 1$ with blocks of size $M \times 1$, namely,

$$\widetilde{\boldsymbol{w}}_b(i) \;\triangleq\; \operatorname{col}\left\{\widetilde{\boldsymbol{w}}_1(i), \ldots \widetilde{\boldsymbol{w}}_N(i)\right\}, \tag{2.36}$$

$$\boldsymbol{w}_b^o \;\triangleq\; \operatorname{col}\left\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\right\}. \tag{2.37}$$

Collecting the weight error vectors $\{\widetilde{\boldsymbol{w}}_k(i+1)\}$ given in (2.35) into $\widetilde{\boldsymbol{w}}_b(i+1)$ in (2.36) and using (2.34), we find that the network error vector recursion for the diffusion strategy (2.23) can be written as:

$$\boxed{\begin{aligned}\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{A}}^{\top}\left(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)\right)\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xz}(i) - \\ \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xu}(i) + \left(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{A}}^{\top}\right)\boldsymbol{w}_b^o - \boldsymbol{z}_\psi(i+1),\end{aligned}} \tag{2.38}$$

where $\boldsymbol{\mathcal{A}}$, $\boldsymbol{\mathcal{M}}$, and $\boldsymbol{\mathcal{R}}_x(i)$ are $N \times N$ block matrices, with individual blocks of size $M \times M$, defined as:

$$\boldsymbol{\mathcal{M}} \triangleq \operatorname{diag}\{\mu_1 \boldsymbol{I}_M, \ldots, \mu_N \boldsymbol{I}_M\}, \tag{2.39}$$

$$\boldsymbol{\mathcal{A}} \triangleq \boldsymbol{A} \otimes \boldsymbol{I}_M, \tag{2.40}$$

$$\boldsymbol{\mathcal{R}}_x(i) \triangleq \operatorname{diag}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{x}_{\ell 1}(i)\boldsymbol{x}_{\ell 1}^{\top}(i), \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{x}_{\ell N}(i)\boldsymbol{x}_{\ell N}^{\top}(i)\right\}, \tag{2.41}$$

and $\boldsymbol{p}_{xz}(i)$, $\boldsymbol{p}_{xu}(i)$, and $\boldsymbol{z}_{\psi}(i+1)$ are $N \times 1$ block column vectors, with individual entries of size $M \times 1$, given by:

$$\boldsymbol{p}_{xz}(i) \triangleq \operatorname{col}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{x}_{\ell 1}(i)z_{\ell 1}(i), \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{x}_{\ell N}(i)z_{\ell N}(i)\right\}, \tag{2.42}$$

$$\boldsymbol{p}_{xu}(i) \triangleq \operatorname{col}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{x}_{\ell 1}(i)\boldsymbol{x}_{\ell 1}^{\top}(i)\boldsymbol{u}_{\ell 1}^{o}, \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{x}_{\ell N}(i)\boldsymbol{x}_{\ell N}^{\top}(i)\boldsymbol{u}_{\ell N}^{o}\right\}, \tag{2.43}$$

$$\boldsymbol{z}_{\psi}(i+1) \triangleq \operatorname{col}\left\{\sum_{\ell\in\mathcal{N}_1^-} a_{\ell 1}\boldsymbol{z}_{\psi,\ell 1}(i+1), \ldots, \sum_{\ell\in\mathcal{N}_N^-} a_{\ell N}\boldsymbol{z}_{\psi,\ell N}(i+1)\right\}. \tag{2.44}$$

For compactness of notations, we introduce the symbols:

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)), \tag{2.45}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xz}(i), \tag{2.46}$$

$$\boldsymbol{r}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xu}(i) - \left(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{A}}^{\top}\right)\boldsymbol{w}_b^{o}, \tag{2.47}$$

so that the stochastic recursion (2.38) can be written as:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}(i)\,\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{g}(i) - \boldsymbol{r}(i) - \boldsymbol{z}_{\psi}(i+1), \tag{2.48}$$

Compared to recursion (B.26), observe that the term $\boldsymbol{r}(i)$ arises from the multitask environment and the terms $\boldsymbol{g}(i)$ and $\boldsymbol{z}_{\psi}(i+1)$ are inherited from the noises introduced at the adaptation step and the combination step, respectively.

For ease of reference, we list in Table 4.2 the symbols that have been defined in subsection 2.4.1, and others that will be defined in subsections 2.4.2 and 2.4.3.

2.4.2 Mean-error analysis

Note that the evolution of the weight-error vector in (2.48) involves block quantities inherited from the distributed processing. To study the stability of recursions involving block quantities, the *block maximum norm*, defined in Appendix A.3, should be used [Sayed, 2014c].

Taking expectations of both sides of recursion (2.48), we get:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1) = \mathbb{E}\,\{\boldsymbol{\mathcal{B}}(i)\,\widetilde{\boldsymbol{w}}_b(i)\} - \mathbb{E}\,\boldsymbol{g}(i) - \mathbb{E}\,\boldsymbol{r}(i) - \mathbb{E}\,\boldsymbol{z}_{\psi}(i+1), \tag{2.49}$$

Table 2.3: List of symbols defined throughout the performance analysis in Chapter 2.

Symbol	Equation
$\boldsymbol{u}_{\ell k}^o = \boldsymbol{w}_\ell^o - \boldsymbol{w}_k^o$	(2.33)
$\boldsymbol{w}_b^o = \operatorname{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}$	(2.37)
$\boldsymbol{\mathcal{M}} = \operatorname{diag}\{\mu_1 \boldsymbol{I}_M, \ldots, \mu_N \boldsymbol{I}_M\}$	(2.39)
$\boldsymbol{\mathcal{A}} = \boldsymbol{A} \otimes \boldsymbol{I}_M$	(2.40)
$\boldsymbol{\mathcal{C}} = \boldsymbol{C} \otimes \boldsymbol{I}_M$	(2.81)
$\boldsymbol{\mathcal{R}}_x = \operatorname{diag}\left\{ \sum_{\ell\in\mathcal{N}_1} c_{\ell 1}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell 1}), \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell N}) \right\}$	(2.55)
$\boldsymbol{p}_{xz} = -\boldsymbol{\mathcal{R}}_{zx} \boldsymbol{w}_b^o$ with $\boldsymbol{\mathcal{R}}_{zx} = [c_{k\ell} \boldsymbol{R}_{zx,k\ell}]$	(2.56)
$\boldsymbol{p}_{xu} = \operatorname{col}\left\{ \sum_{\ell\in\mathcal{N}_1} c_{\ell 1}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell 1}) \boldsymbol{u}_{\ell 1}^o, \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell N}) \boldsymbol{u}_{\ell N}^o \right\}$	(2.57)
$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}}^\top (\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{R}}_x)$	(2.52)
$\boldsymbol{g} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{p}_{xz}$	(2.53)
$\boldsymbol{r} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{p}_{xu} - \left(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{A}}^\top\right) \boldsymbol{w}_b^o$	(2.54)
$\boldsymbol{D}_k = \sum_{\ell\in\mathcal{N}_k} c_{\ell k}^2 \left(\left(\sigma_{zd,\ell k}^2 + \|\boldsymbol{w}_\ell^o\|_{\boldsymbol{R}_{zx,\ell k}}^2 \right) \boldsymbol{R}_{x,\ell} + \left(\sigma_{zd,\ell k}^2 + \sigma_{z,\ell}^2 \right) \boldsymbol{R}_{zx,\ell k} \right)$	(2.82)
$\boldsymbol{H}_k = \sum_{\ell\in\mathcal{N}_k} c_{\ell k}^2 \Big(\boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{zx,\ell k} + (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{x,\ell} + (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{zx,\ell k} + \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{x,\ell} \Big)$	(2.88)
$\boldsymbol{J}_k = \sum_{\ell\in\mathcal{N}_k} c_{\ell k}^2 \left(\boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o (\boldsymbol{w}_\ell^o)^\top \boldsymbol{R}_{zx,\ell k} + (\boldsymbol{w}_\ell^o)^\top \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{x,\ell} \right)$	(2.91)
$\boldsymbol{\mathcal{S}} \approx \boldsymbol{\mathcal{C}}^\top \operatorname{diag}\left\{ \sigma_{z,1}^2 \boldsymbol{R}_{x,1}, \ldots, \sigma_{z,N}^2 \boldsymbol{R}_{x,N} \right\} \boldsymbol{\mathcal{C}} + \boldsymbol{p}_{xz} \boldsymbol{p}_{xz}^\top + \operatorname{diag}\{\boldsymbol{D}_1, \ldots, \boldsymbol{D}_N\}$	(2.80)
$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{S}} \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{A}}$	(2.76)
$\boldsymbol{\mathcal{R}}_r \approx \boldsymbol{r}\boldsymbol{r}^\top + \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \operatorname{diag}\{\boldsymbol{H}_1, \ldots, \boldsymbol{H}_N\} \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{A}}$	(2.87)
$\boldsymbol{\mathcal{G}}_r \approx \boldsymbol{g}\boldsymbol{r}^\top - \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \operatorname{diag}\{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_N\} \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{A}}$	(2.90)
$\boldsymbol{\mathcal{R}}_{z\psi} = \operatorname{diag}\left\{ \sum_{\ell\in\mathcal{N}_1^-} a_{\ell 1}^2 \boldsymbol{R}_{z\psi,\ell 1}, \ldots, \sum_{\ell\in\mathcal{N}_N^-} a_{\ell N}^2 \boldsymbol{R}_{z\psi,\ell N} \right\}$	(2.93)
$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^\top \otimes \boldsymbol{\mathcal{B}}^\top$	(2.72)
$\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{G}} + \boldsymbol{\mathcal{R}}_r + 2\,\boldsymbol{\mathcal{G}}_r^\top + \boldsymbol{\mathcal{R}}_{z\psi}$	(2.96)
$\boldsymbol{y}(i) = \operatorname{vec}(\boldsymbol{\mathcal{T}}) - 2\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \otimes (\boldsymbol{g} + \boldsymbol{r})$	(2.95)

Under Assumption 2.1 on the regression data, it turns out that $\boldsymbol{x}_k(i)$ is independent of $\widetilde{\boldsymbol{w}}_\ell(j)$ for all ℓ and $j \leq i$. This assumption is commonly used to analyze adaptive constructions since it allows to simplify the derivations without constraining the conclusions. There are several results in the adaptation literature that show that performance results that are obtained under this independence assumption match well the actual performance of the algorithms when the step-sizes are sufficiently small (see, e.g., [Sayed, 2008, App. 24.A] and the many references therein). From (2.25), (2.30) and under Assumptions 2.2, 2.3, we obtain the following expectations:

$$\mathbb{E}\,\{\boldsymbol{x}_{\ell k}(i)\boldsymbol{x}_{\ell k}^{\top}(i)\} = \boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell k}, \quad \text{and} \quad \mathbb{E}\,\{\boldsymbol{x}_{\ell k}(i)z_{\ell k}^{\top}(i)\} = -\boldsymbol{R}_{zx,\ell k}\boldsymbol{w}_{\ell}^{o}. \tag{2.50}$$

Thus, under Assumptions 2.1–2.3, we obtain the following recursion for the network mean error vector:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{g} - \boldsymbol{r}, \tag{2.51}$$

where

$$\boldsymbol{\mathcal{B}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{B}}(i) = \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x), \tag{2.52}$$

$$\boldsymbol{g} \triangleq \mathbb{E}\,\boldsymbol{g}(i) = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xz}, \tag{2.53}$$

$$\boldsymbol{r} \triangleq \mathbb{E}\,\boldsymbol{r}(i) = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xu} - \left(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{A}}^{\top}\right)\boldsymbol{w}_b^{o}, \tag{2.54}$$

with $\boldsymbol{\mathcal{R}}_x$, $\boldsymbol{p}_{xz}$, and $\boldsymbol{p}_{xu}$ defined as:

$$\boldsymbol{\mathcal{R}}_x \triangleq \mathbb{E}\,\boldsymbol{\mathcal{R}}_x(i) = \operatorname{diag}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell 1}), \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell N})\right\}, \tag{2.55}$$

$$\boldsymbol{p}_{xz} \triangleq \mathbb{E}\,\boldsymbol{p}_{xz}(i) = -\boldsymbol{\mathcal{R}}_{zx}\boldsymbol{w}_b^{o}, \tag{2.56}$$

$$\begin{aligned}\boldsymbol{p}_{xu} &\triangleq \mathbb{E}\,\boldsymbol{p}_{xu}(i)\\ &= \operatorname{col}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell 1})\boldsymbol{u}_{\ell 1}^{o}, \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell N})\boldsymbol{u}_{\ell N}^{o}\right\}.\end{aligned} \tag{2.57}$$

The matrix $\boldsymbol{\mathcal{R}}_{zx}$ is an $N \times N$ block matrix with (ℓ, k)-th block given by $c_{k\ell}\boldsymbol{R}_{zx,k\ell}$.

Theorem 2.1. (Mean Stability) *Assume that the data model in* (2.22) *and Assumptions 2.1–2.3 hold. Then, for any initial condition, the diffusion LMS algorithm in* (2.23) *converges in the mean, i.e., recursion* (2.51) *converges as $i \to \infty$, if the step-sizes μ_k satisfy:*

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(\sum_{\ell\in\mathcal{N}_k} c_{\ell k}(\boldsymbol{R}_{x,\ell} + \boldsymbol{R}_{zx,\ell k})\right)}, \qquad k = 1, \ldots, N, \tag{2.58}$$

with the asymptotic mean bias is given by:

$$\lim_{i\to\infty} \mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) = -(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{B}})^{-1}(\boldsymbol{g} + \boldsymbol{r}). \tag{2.59}$$

Proof. The convergence of (2.51) is guaranteed if the coefficient matrix $\boldsymbol{\mathcal{B}}$ is contractive, i.e., all its eigenvalues lie inside the unit disc or equivalently $\rho(\boldsymbol{\mathcal{B}}) < 1$. Since any induced matrix norm is lower bounded by its spectral radius, we can write in terms of the block maximum norm (see Appendix A.3):

$$\begin{aligned}\rho\left(\boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x)\right) &\leq \|\boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x)\|_{b,\infty}\\ &\leq \|\boldsymbol{\mathcal{A}}^{\top}\|_{b,\infty}\cdot\|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x\|_{b,\infty}\\ &= \|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x\|_{b,\infty},\end{aligned} \tag{2.60}$$

where we used the submultiplicative property of the block maximum norm and property (A.23) for the block left-stochastic matrix $\boldsymbol{\mathcal{A}}$. The matrix $\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x$ is block diagonal with individual entries of the form

$$\boldsymbol{I}_M-\mu_k\sum_{\ell\in\mathcal{N}_k}c_{\ell k}(\boldsymbol{R}_{x,\ell}+\boldsymbol{R}_{zx,\ell k}).$$

From property (A.22) we have:

$$\|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x\|_{b,\infty}=\max_{1\leq k\leq N}\rho\left(\boldsymbol{I}_M-\mu_k\sum_{\ell\in\mathcal{N}_k}c_{\ell k}(\boldsymbol{R}_{x,\ell}+\boldsymbol{R}_{zx,\ell k})\right). \tag{2.61}$$

Finally, it can be verified that if the step-sizes are chosen according to (2.58), the block maximum norm of the matrix $\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x$ will be less than one, and hence, $\rho(\boldsymbol{\mathcal{B}})<1$. □

Theorem 2.1 allows us to conclude the following.

Observe that the stability condition (2.58) is the same as the condition obtained in a single-task environment [Zhao et al., 2012], which implies that the dynamic range of the step-sizes is not affected by the multitask environment. The noise corrupting the communication of regressors affects the stability condition through the covariance matrices $\boldsymbol{R}_{zx,\ell k}$. For example, if we assume that $\boldsymbol{C}$ is doubly-stochastic, i.e., $\boldsymbol{C}\mathbb{1}=\boldsymbol{C}^{\top}\mathbb{1}=\mathbb{1}$, and we use the fact that the 2-induced norm of a positive definite matrix is equal to its largest eigenvalue, we obtain by Jensen's inequality[4]:

$$\lambda_{\max}\left(\sum_{\ell\in\mathcal{N}_k}c_{\ell k}(\boldsymbol{R}_{x,\ell}+\boldsymbol{R}_{zx,\ell k})\right)\leq\max_{\ell\in\mathcal{N}_k}\lambda_{\max}(\boldsymbol{R}_{x,\ell}+\boldsymbol{R}_{zx,\ell k}). \tag{2.62}$$

Hence, for a doubly-stochastic matrix $\boldsymbol{C}$, the sufficient condition to ensure stability is:

$$0<\mu_k<\frac{2}{\max_{\ell\in\mathcal{N}_k}\lambda_{\max}(\boldsymbol{R}_{x,\ell}+\boldsymbol{R}_{zx,\ell k})},\qquad k=1,\ldots,N. \tag{2.63}$$

Under perfect communication links, condition (2.63) reduces to:

$$0<\mu_k<\frac{2}{\max_{\ell\in\mathcal{N}_k}\lambda_{\max}(\boldsymbol{R}_{x,\ell})},\qquad k=1,\ldots,N. \tag{2.64}$$

[4]For any convex function f, symmetric matrices $\boldsymbol{X}_i$, and non-negative scalars α_i that add up to one, we have $f(\sum_i\alpha_i\boldsymbol{X}_i)\leq\sum_i\alpha_i f(\boldsymbol{X}_i)$.

Then, we conclude that the noise corrupting the regressors reduces the dynamic range of the step-sizes. Note that the stability is not affected by the noise corrupting the communication of estimates. Furthermore, the stability condition does not depend on the combination matrix $\boldsymbol{A}$.

The bias (2.59) results from two factors, namely, running the single-task algorithm in a multitask environment and exchanging regressors over noisy communication links. If the regressors are not corrupted by noise during their transmission, the vector $\boldsymbol{g}$ in (2.53) is zero. Moreover, if the algorithm is applied in a single-task environment or if there is no cooperation between neighbors with different objectives, the vector $\boldsymbol{r}$ in (2.54) is zero. The magnitude of the bias can be large if the distance between the objectives at cooperating agents is large. Observe that the noise over links transmitting the estimates does not induce a bias in the mean.

Finally, note that the mean behavior of the diffusion LMS algorithm in a single-task environment under perfect information exchange (which is considered in Appendix B.2), in a single-task environment under imperfect exchange, and in a multitask environment under perfect communications can be obtained from Theorem 2.1 by setting $\{\boldsymbol{g}, \boldsymbol{r}, \boldsymbol{R}_{zx,\ell k}\}$ equal to $\boldsymbol{0}$, $\boldsymbol{r} = \boldsymbol{0}$, and $\{\boldsymbol{g}, \boldsymbol{R}_{zx,\ell k}\}$ equal to $\boldsymbol{0}$, respectively.

2.4.3 Mean-square-error analysis

Even if, under condition (2.58), the error vectors $\widetilde{\boldsymbol{w}}_k(i)$ are converging on average, they may have large fluctuations around the steady-state values due to the effect of measurement noises and noises corrupting the links. Therefore, a mean-square analysis is required in order to evaluate how the variances $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2$ evolve with time.

Let $\|\boldsymbol{x}\|^2_{\boldsymbol{\Sigma}}$ denote the weighted square quantity $\boldsymbol{x}^\top\boldsymbol{\Sigma}\boldsymbol{x}$, for any vector $\boldsymbol{x}$ and matrix $\boldsymbol{\Sigma}$. By evaluating the means of weighted square quantities of the form $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$ for any positive semi-definite matrix $\boldsymbol{\Sigma}$ that we are free to choose, we are able to evaluate the evolution of several mean-square-error quantities. For example, the excess-mean-square-error (EMSE) and the mean-square-deviation (MSD) at agent k, defined as [Sayed, 2008]:

$$\text{EMSE}_k(i) \triangleq \mathbb{E}\,(\boldsymbol{x}_k^\top(i)\widetilde{\boldsymbol{w}}_k(i))^2, \qquad \text{MSD}_k(i) \triangleq \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2. \tag{2.65}$$

can be obtained from $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$ by selecting $\boldsymbol{\Sigma}$ as:

$$\text{diag}_k\,\{\boldsymbol{0},\ldots,\boldsymbol{0},\boldsymbol{I}_M,\boldsymbol{0},\ldots,\boldsymbol{0}\} \quad \text{and} \quad \text{diag}_k\{\boldsymbol{0},\ldots,\boldsymbol{0},\boldsymbol{R}_{x,k},\boldsymbol{0},\ldots,\boldsymbol{0}\},$$

respectively, where the notation $\text{diag}_k\,\{\boldsymbol{0},\ldots,\boldsymbol{0},\boldsymbol{X},\boldsymbol{0},\ldots,\boldsymbol{0}\}$ refers to a block diagonal matrix with all blocks zero except the k-th block whose value is $\boldsymbol{X}$. Indeed, under Assumption 2.1, the EMSE at agent k can be written as:

$$\text{EMSE}_k(i) = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2_{\boldsymbol{R}_{x,k}}. \tag{2.66}$$

Observe that, from the linear data model (2.22), it can be verified that:

$$\text{MSE}_k(i) \triangleq J_k(\boldsymbol{w}_k(i)) = \underbrace{\mathbb{E}\,(\boldsymbol{x}_k^\top(i)\widetilde{\boldsymbol{w}}_k(i))^2}_{\text{EMSE}_k(i)} + \underbrace{\sigma^2_{z,k}}_{J_k(\boldsymbol{w}_k^o)}, \tag{2.67}$$

Table 2.4: Mean-square-error performance measures and their corresponding matrix $\boldsymbol{\Sigma}$.

Performance measure	Definition	Matrix $\boldsymbol{\Sigma}$
MSD at agent k	$\text{MSD}_k(i) \triangleq \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2$	$\text{diag}_k\{\boldsymbol{0},\ldots,\boldsymbol{0},\boldsymbol{I}_M,\boldsymbol{0},\ldots,\boldsymbol{0}\}$
EMSE at agent k	$\text{EMSE}_k(i) \triangleq \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2_{\boldsymbol{R}_{x,k}}$	$\text{diag}_k\{\boldsymbol{0},\ldots,\boldsymbol{0},\boldsymbol{R}_{x,k},\boldsymbol{0},\ldots,\boldsymbol{0}\}$
Network MSD	$\text{MSD}_{\text{net}}(i) \triangleq \frac{1}{N}\sum_{k=1}^N \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2$	$\frac{1}{N}\boldsymbol{I}_{MN}$
Network EMSE	$\text{EMSE}_{\text{net}}(i) \triangleq \frac{1}{N}\sum_{k=1}^N \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2_{\boldsymbol{R}_{x,k}}$	$\frac{1}{N}\text{diag}\{\boldsymbol{R}_{x,1},\ldots,\boldsymbol{R}_{x,N}\}$

which implies that the EMSE quantifies the size of the offset in the MSE performance of the adaptive filter. The MSD of each agent k measures how far $\boldsymbol{w}_k(i)$ is from $\boldsymbol{w}_k^o$ in the mean-square-error sense. In the following, we shall analyze the performance in the mean-square by considering the mean-square-error quantity $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$. As shown in Table 2.4, the freedom in selecting the positive semi-definite matrix $\boldsymbol{\Sigma}$ allows us to extract several information about the performance of the network and the agents.

Under Assumptions 2.1–2.3, equating weighted square measures $\|\cdot\|^2_{\boldsymbol{\Sigma}}$ on both sides of recursion (2.48), expanding the RHS, and taking expectations, we find that:

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\Sigma}} =& \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}'} - 2\,\mathbb{E}\,\{\boldsymbol{g}^\top(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\} - 2\,\mathbb{E}\,\{\boldsymbol{r}^\top(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\}+\\ &\mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} + \mathbb{E}\,\|\boldsymbol{r}(i)\|^2_{\boldsymbol{\Sigma}} + 2\,\mathbb{E}\,\{\boldsymbol{r}^\top(i)\boldsymbol{\Sigma}\boldsymbol{g}(i)\} + \mathbb{E}\,\|\boldsymbol{z}_\psi(i+1)\|^2_{\boldsymbol{\Sigma}},\end{aligned} \tag{2.68}$$

where $\boldsymbol{\Sigma}' \triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^\top(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\}$. Now, we proceed to evaluate each of the expectations on the RHS of the above equation.

Let $\boldsymbol{\sigma}$ denote the vectorized version of $\boldsymbol{\Sigma}$ obtained by stacking the columns of $\boldsymbol{\sigma}$ on top of each other, i.e., $\boldsymbol{\sigma} \triangleq \text{vec}(\boldsymbol{\Sigma})$ (Appendix A.1 provides some useful properties on the $\text{vec}(\cdot)$ operator and on the Kronecker product $\otimes$ operator). Let $\boldsymbol{\sigma}' \triangleq \text{vec}(\boldsymbol{\Sigma}')$. Using property (A.6), $\boldsymbol{\sigma}'$ can be related linearly to $\boldsymbol{\sigma}$ according to:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}\boldsymbol{\sigma}, \tag{2.69}$$

where $\boldsymbol{\mathcal{F}}$ is an $(MN)^2 \times (MN)^2$ matrix given by:

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^\top(i) \otimes \boldsymbol{\mathcal{B}}^\top(i)\}. \tag{2.70}$$

From properties (A.3) and (A.4), we obtain:

$$\boldsymbol{\mathcal{F}} = \left(\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN}\otimes\boldsymbol{\mathcal{R}}_x\boldsymbol{\mathcal{M}} - \boldsymbol{\mathcal{R}}_x\boldsymbol{\mathcal{M}}\otimes\boldsymbol{I}_{MN} + \mathbb{E}\,\{\boldsymbol{\mathcal{R}}_x(i)\boldsymbol{\mathcal{M}}\otimes\boldsymbol{\mathcal{R}}_x(i)\boldsymbol{\mathcal{M}}\}\right)(\boldsymbol{\mathcal{A}}\otimes\boldsymbol{\mathcal{A}}). \tag{2.71}$$

Evaluating the expectation term in the above expression requires the knowledge of higher order moments of the regression data and link noises, which are not available under the current assumptions. To proceed, we introduce the *small step-sizes* condition, namely, we assume that the step-sizes μ_k are sufficiently small such that terms depending on higher order powers of

the step-sizes can be ignored. This condition is prevalent in the stochastic gradient approximation literature and diffusion strategies [Sayed, 2008, Benveniste et al., 1987, Haykin, 2002, Cattivelli and Sayed, 2010, Sayed, 2014c, Abdolee et al., 2014, Lorenzo, 2014, Chen et al., 2015a, Plata-Chaves et al., 2015, Zhao et al., 2012]. Under this condition, we can either ignore terms depending on higher order power of the step-sizes or call upon a separation principle to approximate these terms (see [Sayed, 2014c, Section 6.5]). The last term on the RHS of (2.71) is one such term and for small step-sizes, we approximate the expectation of the product by the product of expectations. Therefore, we continue our discussion by approximating $\boldsymbol{\mathcal{F}}$ as:

$$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^{\top} \otimes \boldsymbol{\mathcal{B}}^{\top}, \qquad \text{(small step-sizes)}. \tag{2.72}$$

Note that throughout the mean-square-error analyzes, the notation $\boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{Y}}$ is used when $\boldsymbol{\mathcal{X}}$ is equal to $\boldsymbol{\mathcal{Y}} + \mathcal{O}(\boldsymbol{\mathcal{M}}^2)$ where $\mathcal{O}(\boldsymbol{\mathcal{M}}^2)$ is a factor depending on the square of the step-sizes, whose influence can be neglected for small step-sizes.

By Assumption 2.1, the error vector $\widetilde{\boldsymbol{w}}_b(i)$ is independent of $\boldsymbol{\mathcal{B}}(i)$ and $\boldsymbol{g}(i)$. Thus, the expectation in the second term on the RHS of (2.68) can be expressed as:

$$\begin{aligned}
\mathbb{E}\,\{\boldsymbol{g}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\} &= \mathbb{E}\,\{\boldsymbol{p}_{xz}^{\top}(i)\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i))\}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \\
&\approx \boldsymbol{g}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \\
&= \mathrm{vec}(\boldsymbol{g}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)) \\
&\overset{(A.6),(A.7)}{=} (\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \otimes \boldsymbol{g})^{\top}\boldsymbol{\sigma},
\end{aligned} \tag{2.73}$$

where the approximation follows from the small step-sizes condition. Likewise, the expectation in the third term on the RHS of (2.68) can be approximated as:

$$\mathbb{E}\,\{\boldsymbol{r}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\} \approx (\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \otimes \boldsymbol{r})^{\top}\boldsymbol{\sigma}, \qquad \text{(small step-sizes)}. \tag{2.74}$$

The fourth term on the RHS of (2.68) can be written as:

$$\begin{aligned}
\mathbb{E}\,\|\boldsymbol{g}(i)\|_{\boldsymbol{\Sigma}}^2 = \mathbb{E}\,\left\{\mathrm{Tr}(\boldsymbol{g}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{g}(i))\right\} &= \mathrm{Tr}(\boldsymbol{\Sigma}\boldsymbol{\mathcal{G}}) \\
&\overset{(A.5)}{=} [\mathrm{vec}\,(\boldsymbol{\mathcal{G}})]^{\top}\boldsymbol{\sigma},
\end{aligned} \tag{2.75}$$

where we introduced the $MN \times MN$ matrix $\boldsymbol{\mathcal{G}}$ defined as:

$$\boldsymbol{\mathcal{G}} \triangleq \mathbb{E}\,\{\boldsymbol{g}(i)\boldsymbol{g}^{\top}(i)\} = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{S}}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{A}}, \tag{2.76}$$

with $\boldsymbol{\mathcal{S}}$ an $N \times N$ block matrix given by:

$$\boldsymbol{\mathcal{S}} \triangleq \mathbb{E}\,\{\boldsymbol{p}_{xz}(i)\boldsymbol{p}_{xz}^{\top}(i)\}, \tag{2.77}$$

for which the (ℓ, k)-th block is of size $M \times M$ and is equal to:

$$\sum_{m\in\mathcal{N}_\ell}\sum_{n\in\mathcal{N}_k} c_{m\ell}c_{nk}\mathbb{E}\,\{\boldsymbol{x}_{m\ell}(i)z_{m\ell}(i)z_{nk}(i)\boldsymbol{x}_{nk}^{\top}(i)\}. \tag{2.78}$$

Note that the term $\boldsymbol{\mathcal{G}}$ in (2.76) requiring the evaluation of fourth order moments of the form (2.78) involves higher order powers of the step-sizes. Under Assumptions 2.2, 2.3, using the definitions (2.25), (2.30), and approximating the unknown 4-th order moments by products of 2-nd order moments for small step-sizes, we obtain the following approximation for the expectation appearing in the above expression:

$$\begin{aligned}\mathbb{E}\left\{\boldsymbol{x}_{m\ell}(i)z_{m\ell}(i)z_{nk}(i)\boldsymbol{x}_{nk}^{\top}(i)\right\} \approx\ & \boldsymbol{R}_{zx,m\ell}\boldsymbol{w}_m^o(\boldsymbol{w}_n^o)^{\top}\boldsymbol{R}_{zx,nk} + \delta_{mn}\sigma_{z,m}^2\boldsymbol{R}_{x,m} + \\ & \delta_{mn}\delta_{\ell k}\left[\left(\sigma_{zd,m\ell}^2 + (\boldsymbol{w}_m^o)^{\top}\boldsymbol{R}_{zx,m\ell}\boldsymbol{w}_m^o\right)\boldsymbol{R}_{x,m} + \left(\sigma_{z,m}^2 + \sigma_{zd,m\ell}^2\right)\boldsymbol{R}_{zx,m\ell}\right],\end{aligned} \tag{2.79}$$

where δ_{ij} refers to the Kronecker delta function of i and j. Thus, the matrix $\boldsymbol{\mathcal{S}}$ in (2.77) can be approximated as:

$$\boldsymbol{\mathcal{S}} \approx \boldsymbol{p}_{xz}\boldsymbol{p}_{xz}^{\top} + \boldsymbol{\mathcal{C}}^{\top}\mathrm{diag}\left\{\sigma_{z,1}^2\boldsymbol{R}_{x,1},\ldots,\sigma_{z,N}^2\boldsymbol{R}_{x,N}\right\}\boldsymbol{\mathcal{C}} + \mathrm{diag}\left\{\boldsymbol{D}_1,\ldots,\boldsymbol{D}_N\right\}, \quad \text{(small step-sizes)} \tag{2.80}$$

with

$$\boldsymbol{\mathcal{C}} \triangleq \boldsymbol{C}\otimes\boldsymbol{I}_M, \tag{2.81}$$

$$\boldsymbol{D}_k \triangleq \sum_{\ell\in\mathcal{N}_k} c_{\ell k}^2\left(\left(\sigma_{zd,\ell k}^2 + \|\boldsymbol{w}_{\ell}^o\|_{\boldsymbol{R}_{zx,\ell k}}^2\right)\boldsymbol{R}_{x,\ell} + \left(\sigma_{z,\ell}^2 + \sigma_{zd,\ell k}^2\right)\boldsymbol{R}_{zx,\ell k}\right). \tag{2.82}$$

Likewise, the fifth term on the RHS of (2.68) can be written as:

$$\mathbb{E}\,\|\boldsymbol{r}(i)\|_{\boldsymbol{\Sigma}}^2 = \left[\mathrm{vec}\left(\boldsymbol{\mathcal{R}}_r\right)\right]^{\top}\boldsymbol{\sigma}, \tag{2.83}$$

where we introduced the $MN\times MN$ matrix $\boldsymbol{\mathcal{R}}_r$ defined as:

$$\boldsymbol{\mathcal{R}}_r \triangleq \mathbb{E}\left\{\boldsymbol{r}(i)\boldsymbol{r}^{\top}(i)\right\}. \tag{2.84}$$

The evaluation of $\boldsymbol{\mathcal{R}}_r$ in (2.84) requires the evaluation of $\mathbb{E}\left\{\boldsymbol{p}_{xu}(i)\boldsymbol{p}_{xu}^{\top}(i)\right\}$, which is an $N\times N$ block matrix with (ℓ,k)-th block given by:

$$\sum_{m\in\mathcal{N}_\ell}\sum_{n\in\mathcal{N}_k} c_{m\ell}c_{nk}\mathbb{E}\left\{\boldsymbol{x}_{m\ell}(i)\boldsymbol{x}_{m\ell}^{\top}(i)\boldsymbol{u}_{m\ell}^o(\boldsymbol{u}_{nk}^o)^{\top}\boldsymbol{x}_{nk}(i)\boldsymbol{x}_{nk}^{\top}(i)\right\}. \tag{2.85}$$

Under Assumptions 2.2, 2.3, using (2.25), and approximating 4-th order moments by products of 2-nd order moments under the small step-sizes condition, we obtain the following approximation:

$$\begin{aligned}&\mathbb{E}\left\{\boldsymbol{x}_{m\ell}(i)\boldsymbol{x}_{m\ell}^{\top}(i)\boldsymbol{u}_{m\ell}^o(\boldsymbol{u}_{nk}^o)^{\top}\boldsymbol{x}_{nk}(i)\boldsymbol{x}_{nk}^{\top}(i)\right\} \\ &\approx \left(\boldsymbol{R}_{x,m}+\boldsymbol{R}_{zx,m\ell}\right)\boldsymbol{u}_{m\ell}^o(\boldsymbol{u}_{nk}^o)^{\top}\left(\boldsymbol{R}_{x,n}+\boldsymbol{R}_{zx,nk}\right) + \delta_{mn}\delta_{\ell k}\Big(\boldsymbol{R}_{x,m}\boldsymbol{u}_{m\ell}^o(\boldsymbol{u}_{m\ell}^o)^{\top}\boldsymbol{R}_{zx,m\ell} \\ &\quad + (\boldsymbol{u}_{m\ell}^o)^{\top}\boldsymbol{R}_{zx,m\ell}\boldsymbol{u}_{m\ell}^o\boldsymbol{R}_{x,m} + (\boldsymbol{u}_{m\ell}^o)^{\top}\boldsymbol{R}_{x,m}\boldsymbol{u}_{m\ell}^o\boldsymbol{R}_{zx,m\ell} + \boldsymbol{R}_{zx,m\ell}\boldsymbol{u}_{m\ell}^o(\boldsymbol{u}_{m\ell}^o)^{\top}\boldsymbol{R}_{x,m}\Big).\end{aligned} \tag{2.86}$$

Thus, the matrix $\boldsymbol{\mathcal{R}}_r$ in (2.84) can be approximated as:

$$\boldsymbol{\mathcal{R}}_r \approx \boldsymbol{r}\boldsymbol{r}^{\top} + \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\,\mathrm{diag}\left\{\boldsymbol{H}_1,\ldots,\boldsymbol{H}_N\right\}\boldsymbol{\mathcal{M}}\,\boldsymbol{\mathcal{A}}, \qquad \text{(small step-sizes)} \tag{2.87}$$

where $\boldsymbol{H}_k$ is an $M \times M$ matrix given by:

$$\begin{aligned}\boldsymbol{H}_k \triangleq \sum_{\ell \in \mathcal{N}_k} c_{\ell k}^2 \Big(& \boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{zx,\ell k} + (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{x,\ell} + \\ & (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{zx,\ell k} + \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o (\boldsymbol{u}_{\ell k}^o)^\top \boldsymbol{R}_{x,\ell} \Big).\end{aligned} \tag{2.88}$$

Identically, the evaluation of the expectation appearing in the sixth term on the RHS of (2.68) leads to the following expression:

$$\mathbb{E}\,\{\boldsymbol{r}^\top(i)\boldsymbol{\Sigma}\boldsymbol{g}(i)\} = \left[\mathrm{vec}\left(\boldsymbol{\mathcal{G}}_r^\top\right)\right]^\top \boldsymbol{\sigma}. \tag{2.89}$$

where $\boldsymbol{\mathcal{G}}_r$ is defined as:

$$\begin{aligned}\boldsymbol{\mathcal{G}}_r &\triangleq \mathbb{E}\,\{\boldsymbol{g}(i)\boldsymbol{r}^\top(i)\} \\ &\approx \boldsymbol{g}\boldsymbol{r}^\top - \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}}\,\mathrm{diag}\{\boldsymbol{J}_1,\ldots,\boldsymbol{J}_N\}\boldsymbol{\mathcal{M}}\,\boldsymbol{\mathcal{A}}, \qquad \text{(small step-sizes)}\end{aligned} \tag{2.90}$$

with $\boldsymbol{J}_k$ an $M \times M$ matrix given by:

$$\boldsymbol{J}_k \triangleq \sum_{\ell \in \mathcal{N}_k} c_{\ell k}^2 \left(\boldsymbol{R}_{x,\ell} \boldsymbol{u}_{\ell k}^o (\boldsymbol{w}_\ell^o)^\top \boldsymbol{R}_{zx,\ell k} + (\boldsymbol{w}_\ell^o)^\top \boldsymbol{R}_{zx,\ell k} \boldsymbol{u}_{\ell k}^o \boldsymbol{R}_{x,\ell} \right). \tag{2.91}$$

Finally, the last term on the RHS of (2.68) can be expressed as:

$$\mathbb{E}\,\|\boldsymbol{z}_\psi(i+1)\|_{\boldsymbol{\Sigma}}^2 = [\mathrm{vec}(\boldsymbol{\mathcal{R}}_{z\psi})]^\top \boldsymbol{\sigma}, \tag{2.92}$$

where

$$\boldsymbol{\mathcal{R}}_{z\psi} \triangleq \mathbb{E}\,\{\boldsymbol{z}_\psi(i+1)\boldsymbol{z}_\psi^\top(i+1)\} = \mathrm{diag}\left\{ \sum_{\ell \in \mathcal{N}_1^-} a_{\ell 1}^2\, \boldsymbol{R}_{z\psi,\ell 1}, \ldots, \sum_{\ell \in \mathcal{N}_N^-} a_{\ell N}^2\, \boldsymbol{R}_{z\psi,\ell N} \right\}. \tag{2.93}$$

Replacing the expectations appearing on the RHS of (2.68) by (2.73), (2.74), (2.75), (2.83), (2.89), and (2.92), we find that relation (2.68) becomes:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\Sigma}}^2 = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\Sigma}'}^2 + \boldsymbol{y}^\top(i)\boldsymbol{\sigma}, \tag{2.94}$$

where $\boldsymbol{y}(i)$ is a short-hand representation for:

$$\boldsymbol{y}(i) \triangleq \mathrm{vec}(\boldsymbol{\mathcal{T}}) - 2\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) \otimes (\boldsymbol{g} + \boldsymbol{r}) \tag{2.95}$$

and

$$\boldsymbol{\mathcal{T}} \triangleq \boldsymbol{\mathcal{G}} + \boldsymbol{\mathcal{R}}_r + 2\,\boldsymbol{\mathcal{G}}_r^\top + \boldsymbol{\mathcal{R}}_{z\psi}. \tag{2.96}$$

It is convenient to introduce the alternative notation $\|\boldsymbol{x}\|_{\boldsymbol{\sigma}}^2$ to refer to the weighted square quantity $\|\boldsymbol{x}\|_{\boldsymbol{\Sigma}}^2$, where $\boldsymbol{\sigma} = \mathrm{vec}(\boldsymbol{\Sigma})$. These two notations will be used interchangeably. The vector notation allows us to exploit the linear relation (2.69) between $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}'$ and to rewrite (2.94) as:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2 = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2 + \boldsymbol{y}^\top(i)\boldsymbol{\sigma}, \tag{2.97}$$

with the same vector $\boldsymbol{\sigma}$ appearing on both sides. Relation (2.97) shall be used to characterize the transient and steady-state behavior of the diffusion LMS over multitask networks in the presence of noisy communication links.

Theorem 2.2. (Mean-square stability) *Assume that the data model in* (2.22) *and Assumptions 2.1–2.3 hold. Then, for any initial conditions, the diffusion LMS algorithm in* (2.23) *is stable in the mean-square-error sense, i.e., the quadratic quantity* $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$ *converges as* $i \to \infty$ *for any positive semi-definite matrix* $\boldsymbol{\Sigma}$*, if the algorithm is mean stable and if the matrix* $\boldsymbol{\mathcal{F}}$ *in* (2.71) *is contractive. Sufficiently small step-sizes that satisfy* (2.58) *will also ensure mean-square stability.*

Proof. Provided that $\boldsymbol{\mathcal{F}}$ is contractive (i.e., all its eigenvalues lie inside the unit disc), recursion (2.97) is stable if the second term on its RHS is bounded. Since $\boldsymbol{\mathcal{T}}$, $\boldsymbol{\mathcal{B}}$, $\boldsymbol{g}$, $\boldsymbol{r}$, and $\boldsymbol{\sigma}$ are finite and constant terms, the boundedness of $\boldsymbol{y}^{\top}(i)\boldsymbol{\sigma}$ depends on $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ being bounded. From Theorem 2.1 and recursion (2.51), we know that $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ is uniformly bounded since (2.51) is a bounded input bounded output (BIBO) stable recursion with a bounded driving term $-(\boldsymbol{g}+\boldsymbol{r})$. It follows that the second term on the RHS of (2.97) is uniformly bounded. As a result, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$ converges to a bounded value as $i \to \infty$, and the algorithm is mean-square stable.

For small step-sizes, from property (A.8) and approximation (2.72), we have $\rho(\boldsymbol{\mathcal{F}}) \approx \rho^2(\boldsymbol{\mathcal{B}})$. Hence, the matrix $\boldsymbol{\mathcal{F}}$ is contractive if the matrix $\boldsymbol{\mathcal{B}}$ is contractive. Therefore, sufficiently small step-sizes which ensure stability in the mean will also ensure stability in the mean-square. □

Observe that the mean-square-error stability is affected by the noises corrupting the regressors communications and that the multitask environment and the noises $\{z_k(i), z_{\ell k}(i)\}$ do not affect the condition on the step-sizes.

Theorem 2.3. (Transient performance) *Assume the same settings as in Theorem 2.2. Then, the learning curve defined by* $\zeta(i) \triangleq \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$ *evolves according to the following recursion:*

$$\zeta(i+1) = \zeta(i) + \|\widetilde{\boldsymbol{w}}_b(0)\|^2_{\left(\boldsymbol{\mathcal{F}}-\boldsymbol{I}_{(MN)^2}\right)\boldsymbol{\mathcal{F}}^i\boldsymbol{\sigma}} + \Big(\boldsymbol{y}^{\top}(i) + \boldsymbol{\Upsilon}(i)\Big)\,\boldsymbol{\sigma}, \tag{2.98}$$

with $\widetilde{\boldsymbol{w}}_b(0)$ *the initial condition and* $\boldsymbol{\Upsilon}(i)$ *a row vector updated according to the following recursion:*

$$\boldsymbol{\Upsilon}(i+1) = \boldsymbol{\Upsilon}(i)\boldsymbol{\mathcal{F}} + \boldsymbol{y}^{\top}(i)(\boldsymbol{\mathcal{F}} - \boldsymbol{I}_{(MN)^2}), \tag{2.99}$$

with $\boldsymbol{\Upsilon}(0) = \boldsymbol{0}_{1\times(MN)^2}$.

Proof. Iterating (2.97) starting from $i = 0$, we find:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(0)\|^2_{\boldsymbol{\mathcal{F}}^{i+1}\boldsymbol{\sigma}} + \sum_{j=0}^{i} \boldsymbol{y}^{\top}(i-j)\boldsymbol{\mathcal{F}}^j\boldsymbol{\sigma}. \tag{2.100}$$

Comparing (2.100) at time instant i and $i+1$, we obtain (2.98) with:

$$\boldsymbol{\Upsilon}(i) = \sum_{j=1}^{i} \boldsymbol{y}^{\top}(i-j)\boldsymbol{\mathcal{F}}^{j} - \sum_{j=0}^{i-1} \boldsymbol{y}^{\top}(i-1-j)\boldsymbol{\mathcal{F}}^{j}. \tag{2.101}$$

Evaluating (2.101) at time instant $i+1$, we obtain:

$$\begin{aligned}
\boldsymbol{\Upsilon}(i+1) &= \sum_{j=1}^{i+1} \boldsymbol{y}^{\top}(i+1-j)\boldsymbol{\mathcal{F}}^{j} - \sum_{j=0}^{i} \boldsymbol{y}^{\top}(i-j)\boldsymbol{\mathcal{F}}^{j} \\
&= \sum_{j=0}^{i} \boldsymbol{y}^{\top}(i-j)\boldsymbol{\mathcal{F}}^{j+1} - \sum_{j=-1}^{i-1} \boldsymbol{y}^{\top}(i-1-j)\boldsymbol{\mathcal{F}}^{j+1} \\
&= \boldsymbol{y}^{\top}(i)\boldsymbol{\mathcal{F}} + \sum_{j=1}^{i} \boldsymbol{y}^{\top}(i-j)\boldsymbol{\mathcal{F}}^{j+1} - \boldsymbol{y}^{\top}(i) - \sum_{j=0}^{i-1} \boldsymbol{y}^{\top}(i-1-j)\boldsymbol{\mathcal{F}}^{j+1},
\end{aligned} \tag{2.102}$$

and we recover recursion (2.99). □

Theorem 2.4. (Steady-state performance) *Consider the same settings as in Theorem 2.2 and assume mean and mean-square-error stability. Then, the steady-state performance of the diffusion LMS algorithm in* (2.23) *defined as* $\zeta^{\star} \triangleq \lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$ *is given by:*

$$\zeta^{\star} = \boldsymbol{y}^{\top}(\infty)\left(\boldsymbol{I}_{(MN)^2} - \boldsymbol{\mathcal{F}}\right)^{-1}\boldsymbol{\sigma}. \tag{2.103}$$

where $\boldsymbol{y}(\infty)$ *is given by:*

$$\boldsymbol{y}(\infty) \triangleq \mathrm{vec}(\boldsymbol{\mathcal{T}}) - 2\left(\boldsymbol{\mathcal{B}} \lim_{i\to\infty} \mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\right) \otimes (\boldsymbol{g}+\boldsymbol{r}), \tag{2.104}$$

and where $\lim_{i\to\infty} \mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ *is the asymptotic mean bias obtained from* (2.59).

Proof. Consider sufficiently small step-sizes which ensure mean and mean-square stability, and take the limit of (2.97) as $i \to \infty$ and groupe the terms, we obtain:

$$\lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{(\boldsymbol{I}_{(MN)^2}-\boldsymbol{\mathcal{F}})\boldsymbol{\sigma}} = \boldsymbol{y}^{\top}(\infty)\boldsymbol{\sigma} \tag{2.105}$$

In order to recover $\zeta^{\star} = \lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$, the vector $\boldsymbol{\sigma}$ in (2.105) should be replaced by $\left(\boldsymbol{I}_{(MN)^2} - \boldsymbol{\mathcal{F}}\right)^{-1}\boldsymbol{\sigma}$. □

Theorem 2.4 allows us to obtain several useful performance metrics. For example, the steady-state network MSD that averages the MSDs of the agents, is obtained by replacing $\boldsymbol{\sigma}$ in (2.103) by $\frac{1}{N}\mathrm{vec}\,(\boldsymbol{I}_{MN})$:

$$\mathrm{MSD}_{\mathrm{net}} = \frac{1}{N}\lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2 = \frac{1}{N}\boldsymbol{y}^{\top}(\infty)\left(\boldsymbol{I}_{(MN)^2} - \boldsymbol{\mathcal{F}}\right)^{-1}\mathrm{vec}(\boldsymbol{I}_{MN}). \tag{2.106}$$

We observe that, at steady-state, the mean-square-error performance is affected by the noises corrupting the communication links transmitting the regressors and the estimates, and by the multitask environment. Theorems 2.2, 2.3, and 2.4 allow us to recover the performances of the diffusion LMS algorithm under several scenarios (e.g., single-task environment, perfect information exchange) by canceling some terms, such as, $\boldsymbol{u}^{o}_{\ell k}$, $\boldsymbol{R}_{zx,\ell k}$, $\boldsymbol{R}_{z\psi,\ell k}$, etc.

2.5 Optimizing the combination weights

In order to suppress the negative effects of running (2.23) in a multitask environment in the presence of noisy links, we suggest to follow the strategy used in [Zhao et al., 2012, Zhao and Sayed, 2012, Chen et al., 2015a]. It consists of adjusting the combination weights $\{a_{\ell k}\}$ by minimizing the instantaneous MSD at each node k:

$$\text{MSD}_k(i+1) = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i+1)\|^2 = \mathbb{E}\left\|\boldsymbol{w}_k^o - \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell k}(i+1)\right\|^2. \tag{2.107}$$

Using the fact that $\boldsymbol{A}$ is left-stochastic and relaxing the problem in (2.107) by omitting the expectation operator and the cross terms, we arrive at:

$$a_{\ell k}a_{mk}(\boldsymbol{w}_k^o - \boldsymbol{\psi}_{\ell k}(i+1))^\top(\boldsymbol{w}_k^o - \boldsymbol{\psi}_{mk}(i+1)), \quad \text{for } \ell \neq m.$$

As suggested in [Zhao et al., 2012, Zhao and Sayed, 2012], we then obtain the following optimization problem at agent k:

$$\begin{aligned} &\underset{\{a_{\ell k}\}}{\text{minimize}} \quad \sum_{\ell=1}^{N} a_{\ell k}^2\|\widehat{\boldsymbol{w}}_k^o - \boldsymbol{\psi}_{\ell k}(i+1)\|^2, \\ &\text{subject to} \quad \sum_{\ell=1}^{N} a_{\ell k} = 1, \quad a_{\ell k} \geq 0, \quad \text{and} \quad a_{\ell k} = 0 \text{ if } \ell \notin \mathcal{N}_k, \end{aligned} \tag{2.108}$$

where $\widehat{\boldsymbol{w}}_k^o$ is some approximation for $\boldsymbol{w}_k^o$ since, in general, $\boldsymbol{w}_k^o$ is unknown. One useful approximation is the local one-step approximation used in [Chen et al., 2015a]:

$$\widehat{\boldsymbol{w}}_k^o(i+1) = \boldsymbol{\psi}_k(i+1) + \mu_k\frac{\boldsymbol{q}_k(i)}{\|\boldsymbol{q}_k(i)\| + \epsilon}, \tag{2.109}$$

where $\boldsymbol{q}_k(i)$ is an approximation of the opposite of the true gradient vector of the cost $J_k(\boldsymbol{w})$ at the estimate $\boldsymbol{\psi}_k(i+1)$ given by:

$$\boldsymbol{q}_k(i) \triangleq -\widehat{\nabla_w J_k}(\boldsymbol{\psi}_k(i+1)) = \boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{\psi}_k(i+1)\right), \tag{2.110}$$

and ϵ is a small positive value which is added to obtain $\frac{\boldsymbol{q}_k(i)}{\|\boldsymbol{q}_k(i)\|+\epsilon} = \boldsymbol{0}$ when $\boldsymbol{q}_k(i)$ is zero. Introducing the notation $\gamma_{\ell k}^2(i+1) \triangleq \|\widehat{\boldsymbol{w}}_k^o(i+1) - \boldsymbol{\psi}_{\ell k}(i+1)\|^2$, the solution of problem (2.108) is given by:

$$a_{\ell k}(i+1) = \frac{\gamma_{\ell k}^{-2}(i+1)}{\sum_{n\in\mathcal{N}_k}\gamma_{nk}^{-2}(i+1)}, \qquad \ell \in \mathcal{N}_k. \tag{2.111}$$

This rule for adjusting the combination coefficients takes into account the closeness of the local estimate $\boldsymbol{\psi}_k(i)$ to the noisy neighboring estimate $\boldsymbol{\psi}_{\ell k}(i)$ for $\ell \in \mathcal{N}_k$, and the local slope of the cost function $J_k(\cdot)$ [Chen et al., 2015a]. In particular, agent k tends to decrease the weight $a_{\ell k}(i+1)$ if the distance $\|\boldsymbol{\psi}_k(i+1) - \boldsymbol{\psi}_{\ell k}(i+1)\|^2$ is large and if $J_k(\boldsymbol{\psi}_{\ell k}(i+1)) > J_k(\boldsymbol{\psi}_k(i+1))$.

This observation follows from the following expansion of the inverse of the numerator in (2.111):

$$\begin{aligned}
&\|\widehat{\boldsymbol{w}}_k^o(i+1) - \boldsymbol{\psi}_{\ell k}(i+1)\|^2 \\
&= \left\| \boldsymbol{\psi}_k(i+1) + \mu_k \frac{\boldsymbol{q}_k(i)}{\|\boldsymbol{q}_k(i)\| + \epsilon} - \boldsymbol{\psi}_{\ell k}(i+1) \right\|^2 \\
&= \|\boldsymbol{\psi}_k(i+1) - \boldsymbol{\psi}_{\ell k}(i+1)\|^2 + \mu_k^2 \left\| \frac{\boldsymbol{q}_k(i)}{\|\boldsymbol{q}_k(i)\| + \epsilon} \right\|^2 - \frac{2\mu_k}{\|\boldsymbol{q}_k(i)\| + \epsilon} \left(\boldsymbol{\psi}_{\ell k}(i+1) - \boldsymbol{\psi}_k(i+1)\right)^\top \boldsymbol{q}_k(i)
\end{aligned} \tag{2.112}$$

The first term on the RHS of the previous relation is equal to the distance between the local estimate at node k and the estimate arriving from neighboring node ℓ. The second term on the RHS is the same for all $\ell \in \mathcal{N}_k$. The third term is proportional to $J_k(\boldsymbol{\psi}_{\ell k}(i+1)) - J_k(\boldsymbol{\psi}_k(i+1))$ since the first order Taylor expansion of $J_k(\boldsymbol{w})$ at the estimate $\boldsymbol{\psi}_k(i+1)$ gives:

$$J_k(\boldsymbol{w}) \approx J_k(\boldsymbol{\psi}_k(i+1)) - (\boldsymbol{w} - \boldsymbol{\psi}_k(i+1))^\top \boldsymbol{q}_k(i). \tag{2.113}$$

Thus, this rule leads to a network clustering by assigning large weights to agents sharing the same objective and negligible weights to agents with distinct objectives.

It is noticed in [Khawatmi et al., 2015] that the clustering strategy (2.111) (under perfect information exchange conditions) may suffer from a larger probability of false alarm, that is, $a_{\ell k}$ may tend to zero even in situations where nodes k and ℓ share the same task. To overcome this problem, we propose to smooth $\gamma_{\ell k}^2(i+1)$ as follows:

$$\boxed{\gamma_{\ell k}^2(i+1) = (1-\nu_k)\gamma_{\ell k}^2(i) + \nu_k \|\widehat{\boldsymbol{w}}_k^o(i+1) - \boldsymbol{\psi}_{\ell k}(i+1)\|^2} \tag{2.114}$$

where $\nu_k \in [0,1]$ is a forgetting factor.

The protocol for adjusting the combination weights in [Zhao and Sayed, 2012, Zhao et al., 2012] differs from (2.111) and (2.114) by using the estimate $\boldsymbol{w}_k(i)$ as an approximation for $\boldsymbol{w}_k^o$ in (2.109). Moreover, the clustering strategy proposed in [Chen et al., 2015a] does not include the smoothing step (2.114). As shown by simulations, this step reduces the probability of erroneous clustering especially in the presence of noisy links.

2.6 Simulation results

Consider a connected network the topology of which is shown in Figure 2.3 (left plot), consisting of 20 agents grouped into 4 clusters: $\mathcal{C}_1 = \{1,\ldots,6\}$, $\mathcal{C}_2 = \{7,\ldots,12\}$, $\mathcal{C}_3 = \{13,\ldots,16\}$, and $\mathcal{C}_4 = \{17,\ldots,20\}$. We assume that agents belonging to the same cluster are interested in estimating the same parameter vector, i.e., $\boldsymbol{w}_k^o = \boldsymbol{w}_{\mathcal{C}_q}^o$ if $k \in \mathcal{C}_q$, with $\boldsymbol{w}_{\mathcal{C}_q}^o$ denoting the optimum parameter vector at cluster $\mathcal{C}_q$. Regressors are 2×1 zero-mean Gaussian random vectors with covariance matrices $\boldsymbol{R}_{x,k} = \sigma_{x,k}^2 \boldsymbol{I}_2$. Noises $z_k(i)$ are zero-mean i.i.d. Gaussian with variance $\sigma_{z,k}^2$. Variances $\sigma_{x,k}^2$ and $\sigma_{z,k}^2$ are shown in Figure 2.3 (right plot). Noises over links $z_{d,\ell k}(i)$, $\boldsymbol{z}_{x,\ell k}(i)$, and $\boldsymbol{z}_{\psi,\ell k}(i)$ are also zero-mean i.i.d. Gaussian of variances σ_z^2 and covariances $\sigma_z^2 \boldsymbol{I}_2$ for

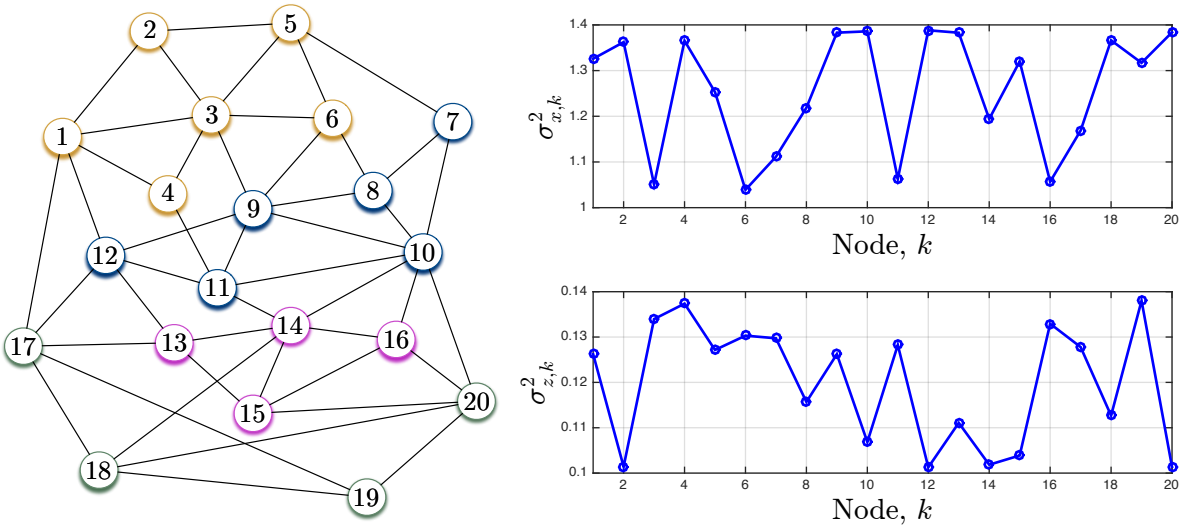


Figure 2.3: Experimental setup. (*Left*) Network topology. (*Right*) Regression and noise variances.

all $\ell \in \mathcal{N}_k^-$. The optimum parameter vectors are uniformly distributed on a circle of radius r centered at $\boldsymbol{w}_o = [0.5, -0.5]^\top$, namely,

$$\boldsymbol{w}_{\mathcal{C}_q}^o = \boldsymbol{w}_o + r \begin{pmatrix} \cos\left(\dfrac{2\pi(q-1)}{4} + \dfrac{\pi}{10}\right) \\ \sin\left(\dfrac{2\pi(q-1)}{4} + \dfrac{\pi}{10}\right) \end{pmatrix}, \quad q = 1, \ldots, 4. \tag{2.115}$$

We use a constant step-size $\mu_k = 0.01$ for all k. The results are averaged over 100 runs. Let $\boldsymbol{A}_0$ and $\boldsymbol{C}_0$ be uniform combination matrices (2.14), namely, $a_{\ell k} = 1/\text{card}\{\mathcal{N}_k\}$ for $\ell \in \mathcal{N}_k$ and $c_{\ell k} = 1/\text{card}\{\mathcal{N}_\ell\}$ for $k \in \mathcal{N}_\ell$, respectively.

First, we considered the case where the tasks are close to each other by setting $r = 0.02$. We run the ATC algorithm (2.23) with $\boldsymbol{C}_0$ and $\boldsymbol{A}_0$ for 4 levels of noise over links: $\text{L}_0 : \sigma_z^2 = 0$, $\text{L}_1 : \sigma_z^2 = 10^{-4}$, $\text{L}_2 : \sigma_z^2 = 10^{-3}$, and $\text{L}_3 : \sigma_z^2 = 10^{-2}$. The non-cooperative LMS (2.7) is also considered by setting $\boldsymbol{A} = \boldsymbol{C} = \boldsymbol{I}_N$. The network MSD learning curves are reported in Figure 2.4. The theoretical transient MSD is obtained from Theorem 2.3 by setting $\boldsymbol{\sigma} = \frac{1}{N}\text{vec}\,(\boldsymbol{I}_{MN})$ and the theoretical steady-state MSD is given in (2.106). It can be observed that the theoretical findings match well the simulated curves. Furthermore, for a certain degree of similarity between tasks, diffusion LMS with perfect information exchange can still deliver superior performance compared to non-cooperative strategies despite the bias introduced by the multitask scenario. The performance decreases when the level of noise over links increases.

In the following, we use $\boldsymbol{A}(0) = \boldsymbol{A}_0$ and $\boldsymbol{C}(0) = \boldsymbol{C}_0$. The coefficients $c_{\ell k}(i)$ were set such that $\boldsymbol{C}(i+1) = \boldsymbol{A}^\top(i)$. Three different protocols for adjusting the combination coefficients $a_{\ell k}$ are considered: the rule (2.109)–(2.114) with $\nu_k = 0.05$ and $\epsilon = 0.01$, the rule in [Chen et al., 2015a] with $\epsilon = 0.01$, and the rule in [Zhao et al., 2012, Zhao and Sayed, 2012] with $\nu_k = 0.05$. We run algorithm (2.23) for $r = 0.02$ and $\sigma_z^2 = 10^{-2}$ with the adaptive combination rules mentioned

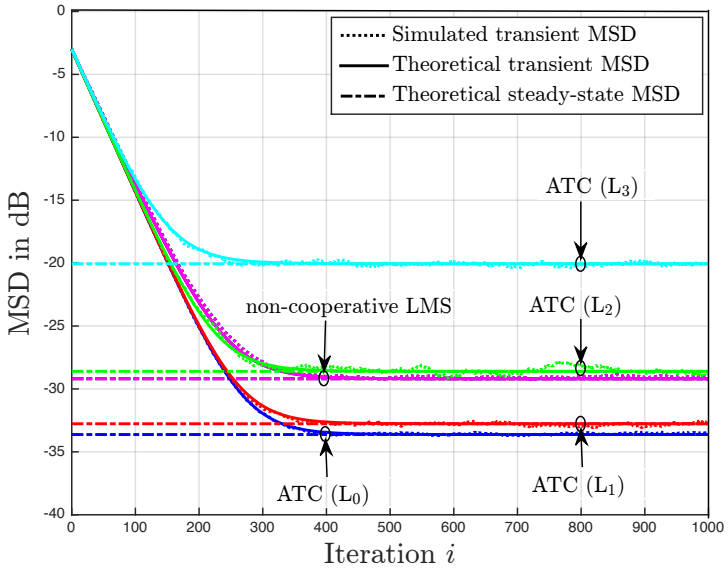


Figure 2.4: Network MSD behavior of the ATC diffusion algorithm (2.23) for different levels of noise over the communication links. Non-cooperative LMS is given in (2.7).

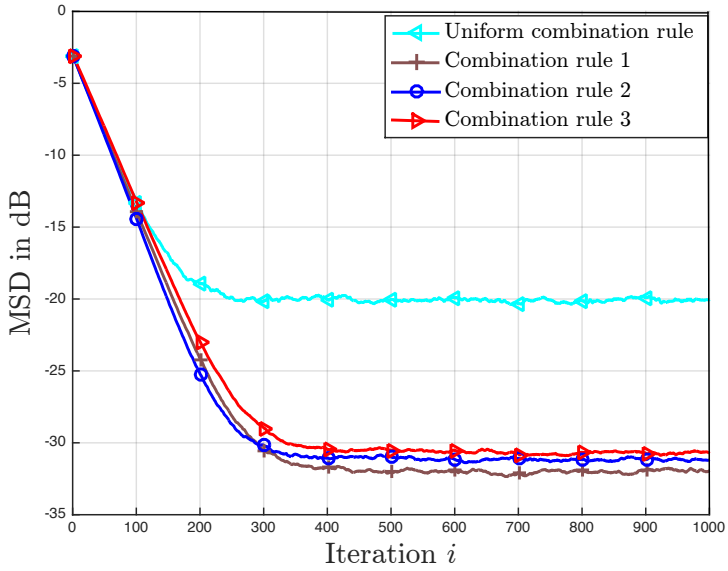


Figure 2.5: Network MSD of the ATC diffusion algorithm (2.23) for different combination rules (close tasks). Combination rule 1 refers to the rule (2.109)–(2.114). Combination rule 2 refers to the rule in [Chen et al., 2015a]. Combination rule 3 refers to the rule in [Zhao et al., 2012, Zhao and Sayed, 2012].

earlier. Figure 2.5 illustrates the network MSD behavior for these algorithms. It appears that all these rules allow us to reduce the negative effects of noise over communication links. Our rule (2.109)–(2.114) achieves the best performance.

To test the clustering ability of the ATC algorithm (2.23) with adaptive combiners in the presence of noisy links, we increased the distance between tasks by setting $r = 1$. In Figure 2.6,

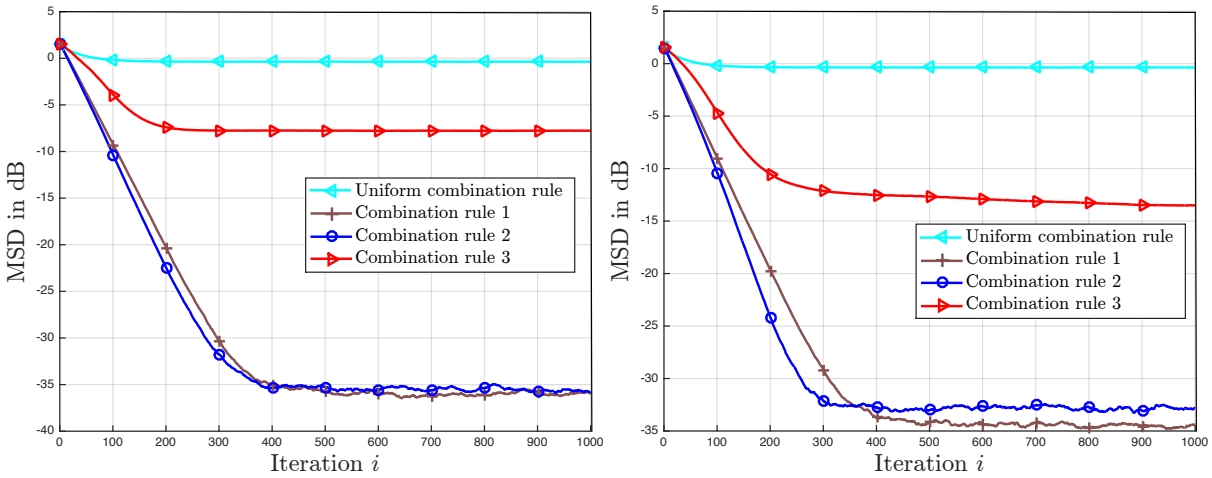


Figure 2.6: Network MSD of the ATC diffusion algorithm (2.23) for different combination rules (distant tasks) with perfect (*left*) and imperfect (*right*) information exchange. Combination rule 1 refers to the rule (2.109)–(2.114). Combination rule 2 refers to the rule in [Chen et al., 2015a]. Combination rule 3 refers to the rule in [Zhao et al., 2012, Zhao and Sayed, 2012].

we compare the network MSD of the algorithm under perfect (left plot) and imperfect (right plot) information exchange, by setting $\sigma_z^2 = 0$ and $\sigma_z^2 = 10^{-4}$, respectively. In each case, we considered fixed combiners $\{a_{\ell k}, c_{\ell k}\}$ and adaptive combiners using the 3 different protocols mentioned earlier. As shown by the experiments, the use of adaptive combiners is necessary when the tasks are not close enough. Furthermore, our rule (2.109)-(2.114) provides the best performance especially in the presence of noisy information exchange. To better analyze this behavior, we report in Figure 2.7 the probabilities of erroneous clustering decisions of types I (left plot) and II (right plot). Consider the link $\mathcal{L}_{\ell,k}$ connecting k to its neighbor ℓ. The probability of type I for node k is the probability that $\mathcal{L}_{\ell,k}$ is erroneously dropped while $\boldsymbol{w}_k^o = \boldsymbol{w}_\ell^o$. The probability of type II is the probability that $\mathcal{L}_{\ell,k}$ is erroneously connected while $\boldsymbol{w}_k^o \neq \boldsymbol{w}_\ell^o$. We considered that the link is dropped off if $a_{\ell k}(i) < 0.05$. The experiments show that the rule in [Zhao et al., 2012, Zhao and Sayed, 2012] suffers when the distance between tasks is large. The rule in [Chen et al., 2015a] tends to drop off links between agents of the same clusters, notably in the presence of noisy links. Our rule (2.109)-(2.114) is able to perform a perfect clustering in the presence and absence of noisy links since both types of probabilities are decaying to zero.

2.7 Conclusion

In this chapter, we have reviewed the diffusion LMS strategies over single-task MSE networks. Although other forms of adaptive implementations, such as RLS and NLMS, have also been considered in the context of diffusion strategies, we shall focus on LMS implementations because of their simplicity. Next, we have considered a scenario where the diffusion LMS algorithm is applied to multitask networks in the presence of noisy links and have studied its performance in

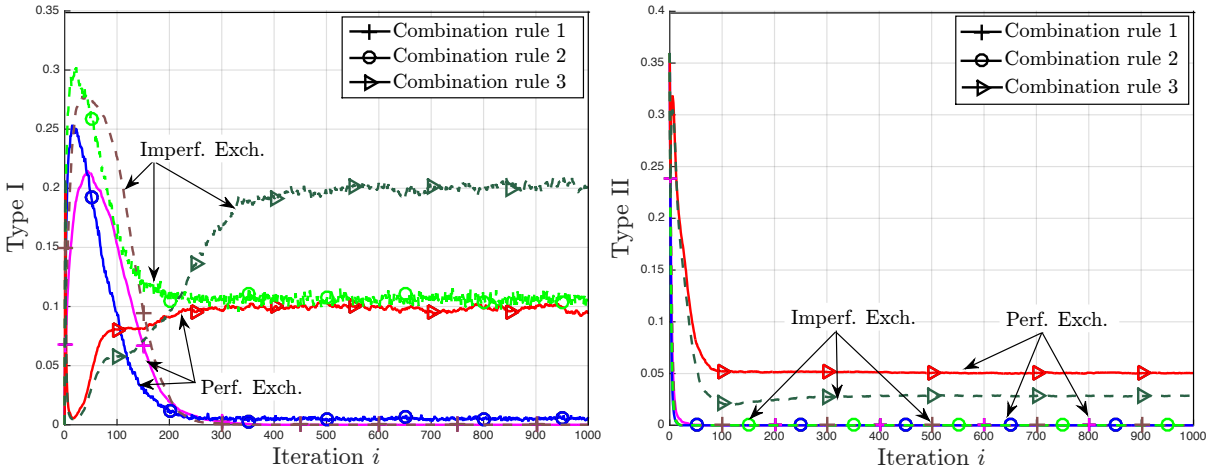


Figure 2.7: Erroneous clustering decisions of type I (*left*) and II (*right*) with perfect (solid) and imperfect (dashed) information exchange. Combination rule 1 refers to the rule (2.109)–(2.114). Combination rule 2 refers to the rule in [Chen et al., 2015a]. Combination rule 3 refers to the rule in [Zhao et al., 2012, Zhao and Sayed, 2012].

the mean and mean-square error sense. An online strategy for adapting the combination coefficients has been provided to reduce the impact of communicating agents with distinct objectives in the presence of noisy links. Finally, simulation results have validated the theoretical findings and have tested the efficiency of the clustering strategy. We observed that by smoothing $\gamma_{\ell k}^2(i)$ through a first order filter, the clustering decision at agent k is more robust.

In the following chapters, we consider strategies that generalize the consensus requirement to settings where different related models need to be estimated simultaneously by the agents collecting noisy measurements. It is assumed that agents have some prior knowledge about the clustering and the relationships between the models. The objective is then to derive and study distributed strategies that exploit the dependencies between the tasks in an appropriate manner in order to improve the estimation performances.

For compactness and concision purposes, in chapters 3, 4, and 5, we focus on analyzing multitask networks where the exchange of information between agents is not corrupted by noise. However, note that, most of the analyzes in these chapters can be extended to handle the more general scenario of noisy links using a similar additive noise model as in Section 2.3.

CHAPTER 3

Multitask diffusion LMS over asynchronous networks

The multitask diffusion LMS proposed in [Chen et al., 2014b,c] is a distributed strategy to simultaneously infer, in a collaborative manner, multiple models that are close to each other in the Euclidean norm sense. However, this work assumes that all agents can process data synchronously. In several applications, agents may not be able to act synchronously because networks can be subject to several sources of uncertainties such as changing topology, random link failures, or agents turning on and off for energy conservation. In this chapter, we describe a multitask model over *asynchronous* networks and examine its performance by carrying out a detailed mean and mean-square error analysis. The analysis reveals how the asynchronous events interfere with the learning behavior of the multitask algorithm.
The work presented in this chapter was published in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Performance analysis of multitask diffusion adaptation over asynchronous networks. In *Proc. Asilomar Conference on Signals, Systems and Computers*, pages 788–792, Pacific Grove, CA, November 2014. (Also available at http://www.cedric-richard.fr/Articles/nassif2014performance.pdf)

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion adaptation over asynchronous networks. *IEEE Transactions on Signal Processing*, 64(11):2835–2850, June 2016. (Also available at http://arxiv.org/abs/1412.1798)

3.1 Introduction

Most of the prior literature on distributed adaptive processing focuses primarily on *single-task* estimation problems where agents estimate a single parameter vector collaboratively. As described in Section 1.2, some applications need more complex models and flexible algorithms than

single-task implementations since their agents may need to estimate and track multiple targets simultaneously. For instance, sensor networks deployed to estimate a spatially-varying temperature profile need to exploit more directly the spatio-temporal correlations that exist between measurements at neighboring nodes [Abdolee et al., 2014]. Likewise, monitoring applications where agents need to track the movement of multiple correlated targets need to exploit the correlation profile in the data for enhanced accuracy. In this chapter, we consider *multitask* estimation problems where agents need to infer simultaneously multiple parameter vectors from noisy measurements.

Existing strategies to address multitask problems mostly depend on how the tasks relate to each other and on exploiting some prior information. There have been some useful works dealing with such problems over distributed networks. For example, in [Plata-Chaves et al., 2015], it is assumed that there are three types of parameters: parameters of global interest to the whole network, parameters of common interest to a subset of agents, and a collection of parameters of local interest. A diffusion strategy of the LMS type was developed to perform estimation under these conditions. In comparison, the parameter space is decomposed into two orthogonal subspaces in [Chen et al., 2014a], with one of the subspaces being common to all agents. Multitask estimation algorithms over fully connected networks [Bertrand and Moonen, 2010], tree networks [Bertrand and Moonen, 2011], and combinations thereof [Szurley et al., 2015] are also considered. These works assume that the node-specific signals lie in a common latent signal subspace and exploit this property to compress information and to reduce communication costs. An alternative way to exploit and model relationships among tasks is to formulate optimization problems with appropriate co-regularizers between agents [Chen et al., 2014a,c, Nassif et al., 2015, 2016d]. The multitask diffusion LMS algorithm derived in [Chen et al., 2014b,c] relies on this principle, and we build on this construction in this chapter. In this context, the network is not assumed to be fully connected and agents do not need to be interested in some common parameters. It is sufficient to assume that different clusters within the network are interested in estimating their own models, and that there are some correlations among the models of adjacent clusters. These correlations are captured by means of regularization parameters.

The aforementioned works on multitask problems assume that all agents respond to data synchronously. In several applications, agents may not be able to act synchronously because networks can be subject to several sources of uncertainties such as changing topology, random link failures, or agents turning on and off randomly for energy conservation. There exist several useful studies in the literature on the performance of consensus and gossip strategies in the presence of asynchronous events [Kar and Moura, 2009, Srivastava and Nedic, 2011, Tsitsiklis et al., 1986, Boyd et al., 2006] or changing topologies [Kar and Moura, 2008, 2009, 2010, 2011, Boyd et al., 2006, Srivastava and Nedic, 2011, Aysal et al., 2009, Jakovetic et al., 2010, 2011]. In most parts, these works investigate pure averaging algorithms that cannot process streaming data, or assume

noise-free data, or make use of decreasing step-size sequences. There are also studies in the context of diffusion strategies. In particular, the works [Zhao and Sayed, 2015a,b,c] introduced a rather general framework for asynchronous networks that includes many prior models as special cases. These works examined how asynchronous events interfere with the behavior of adaptive networks in the presence of streaming noisy data and under constant step-size adaptation. Several interesting conclusions are reported in [Zhao and Sayed, 2015c] where comparisons are carried out between synchronous and asynchronous behavior, as well as with centralized solutions. In this thesis, we would like to examine similar effects to [Zhao and Sayed, 2015a,b] albeit in the context of *multitask* networks as opposed to *single-task* networks. In this case, a new dimension arises in that asynchronous events can interfere with the exchange of information among clusters. We examine in some details the mean and mean-square stability of the multitask network and show that sufficiently small step-sizes can still ensure convergence and performance. Various simulation results illustrate the theoretical findings and the benefit from learning multiple tasks simultaneously. The framework is applied to a particular application involving the localization and the tracking of circular targets.

Before starting our presentation, we list in Table 3.1 some of the main symbols and notations used in this chapter. Other symbols will be defined in the context where they are used.

3.2 Multitask diffusion LMS over asynchronous networks

We now briefly recall the *synchronous* diffusion adaptation strategy developed in [Chen et al., 2014c] for solving distributed estimation problems over multitask networks.

3.2.1 Synchronous multitask diffusion adaptation

We consider a connected network consisting of N nodes grouped into Q clusters, as illustrated in Figure 3.1. The problem is to estimate an $M \times 1$ unknown parameter vector $\boldsymbol{w}_k^o$ at each node k from observed data. Node k has access to temporal measurement sequences $\{d_k(i), \boldsymbol{x}_k(i)\}$, where $d_k(i)$ is a scalar zero-mean reference signal, and $\boldsymbol{x}_k(i)$ is an $M \times 1$ regression vector with a positive-definite covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\,\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)\} > \boldsymbol{0}$. The data at node k are assumed to be related via the linear regression model:

$$d_k(i) = \boldsymbol{x}_k^\top(i)\,\boldsymbol{w}_k^o + z_k(i), \qquad i \geq 0 \tag{3.1}$$

where $z_k(i)$ is a zero-mean i.i.d. noise of variance $\sigma_{z,k}^2$ that is independent of any other signal in the network. We assume that nodes belonging to the same cluster have the same parameter vector to estimate, namely,

$$\boldsymbol{w}_k^o = \boldsymbol{w}_{\mathcal{C}_q}^o, \quad \text{whenever} \quad k \in \mathcal{C}_q. \tag{3.2}$$

We say that two clusters are connected if there exists at least one edge linking a node from one cluster to a node in the other cluster. We also assume that relationships between connected

Table 3.1: List of the main symbols and notations used in Chapter 3.

M	Length of the parameter vectors
N	Number of agents in the network
Q	Number of clusters in the network
$\mathcal{N}_k$	Neighborhood of agent k, i.e., the set of agents that are connected to k by edges
$\mathcal{N}_k^-$	Neighborhood of agent k, excluding k
$\mathcal{C}_q$	Cluster q, i.e., the set of agents in the q-th cluster
$\mathcal{C}(k)$	The cluster of agents to which agent k belongs, including k
$\mathcal{C}(k)^-$	The cluster of agents to which agent k belongs, excluding k
$\mathcal{N}_k \cap \mathcal{C}(k)$	Neighbors of agent k that are inside its cluster
$\mathcal{N}_k \setminus \mathcal{C}(k)$	Neighbors of agent k that are outside its cluster
$J(\cdot),\ \overline{J}(\cdot)$	Cost function without/ with regularization
$\boldsymbol{w}_k$	Parameter vector at agent k
$\boldsymbol{w}_k^o,\ \boldsymbol{w}_{\mathcal{C}_q}^o,\ \boldsymbol{w}_{\mathcal{C}(k)}^o$	Optimum parameter vectors at agent k, at cluster $\mathcal{C}_q$, and at cluster $\mathcal{C}(k)$
$\boldsymbol{w}_b$	Network block parameter vector, also called graph signal
$\boldsymbol{w}_b^o$	Network block optimum parameter vector
$\overline{\boldsymbol{M}},\ \overline{\boldsymbol{A}},\ \overline{\boldsymbol{P}}_\rho$	Means of the random processes $\boldsymbol{M}(i)$, $\boldsymbol{A}(i)$, and $\boldsymbol{P}_\rho(i)$
$\boldsymbol{C}_M,\ \boldsymbol{C}_A,\ \boldsymbol{C}_P$	Kronecker covariance matrices of the random processes $\boldsymbol{M}(i)$, $\boldsymbol{A}(i)$, and $\boldsymbol{P}_\rho(i)$

clusters exist so that cooperation among adjacent clusters is beneficial. In particular, we suppose that the parameter vectors corresponding to two connected clusters $\mathcal{C}_p$ and $\mathcal{C}_q$ satisfy certain properties, such as being close to each other [Chen et al., 2014c]. Cooperation across these clusters can therefore be beneficial to infer $\boldsymbol{w}_{\mathcal{C}_p}^o$ and $\boldsymbol{w}_{\mathcal{C}_q}^o$.

Consider the cluster $\mathcal{C}(k)$ to which node k belongs. A local cost function, $J_k(\boldsymbol{w}_{\mathcal{C}(k)})$, is associated with node k. It is assumed to be strongly convex and second-order differentiable. An example of which is the mean-square-error (MSE) criterion considered throughout this dissertation and defined by:

$$J_k\big(\boldsymbol{w}_{\mathcal{C}(k)}\big) = \mathbb{E}\left(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{\mathcal{C}(k)}\right)^2. \tag{3.3}$$

Depending on the application, there may be certain properties among the optimal vectors $\{\boldsymbol{w}_{\mathcal{C}_1}^o, \ldots, \boldsymbol{w}_{\mathcal{C}_Q}^o\}$ that deserve to be promoted in order to enhance estimation accuracy. Among other possible options, a smoothness condition was enforced in [Chen et al., 2014c]. Specifically, the local variation of the graph signal (which refers to the $N \times 1$ block vector $\boldsymbol{w}_b$ for which the k-th block is the parameter vector at node k) at node k was defined as the squared ℓ_2-norm of

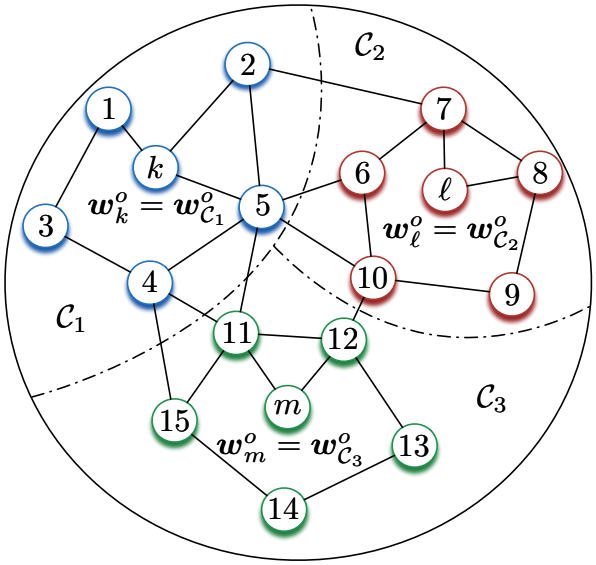


Figure 3.1: Clustered multitask network consisting of 3 clusters. Two clusters are connected if they share at least one edge.

the graph gradient at this node [Grady and Polimeni, 2010], namely,

$$\|\nabla_k \boldsymbol{w}_b\|^2 = \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)} \rho_{k\ell} \|\boldsymbol{w}_k - \boldsymbol{w}_\ell\|^2 \tag{3.4}$$

where $\rho_{k\ell}$ is a nonnegative weight assigned to the edge between nodes k and ℓ. As an alternative to (3.4) and in order to promote piecewise constant transitions in the entries of the parameter vectors, the use of the ℓ_1-norm of the graph gradient at each node is proposed and studied in the next chapter. In this chapter, we will focus on (3.4).

To estimate the unknown parameter vectors $\boldsymbol{w}_{\mathcal{C}_1}^o, \ldots, \boldsymbol{w}_{\mathcal{C}_Q}^o$, it was shown in [Chen et al., 2014c] that the local cost (3.3) and the regularizer (3.4) can be combined at the level of each cluster. This formulation led to the following estimation problem defined in terms of Q Nash equilibrium problems [Basar and Olsder, 1995], where each cluster $\mathcal{C}_q$ estimates $\boldsymbol{w}_{\mathcal{C}_q}^o$ by minimizing the regularized cost function $\overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q}, \boldsymbol{w}_{-\mathcal{C}_q})$:

$$\begin{cases} \underset{\boldsymbol{w}_{\mathcal{C}_q}}{\text{minimize}}\, \overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q}, \boldsymbol{w}_{-\mathcal{C}_q}) \\ \text{with } \overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q}, \boldsymbol{w}_{-\mathcal{C}_q}) \triangleq \displaystyle\sum_{k \in \mathcal{C}_q} \mathbb{E}\left(d_k(i) - \boldsymbol{x}_k^\top(i) \boldsymbol{w}_{\mathcal{C}(k)}\right)^2 + \eta \sum_{k \in \mathcal{C}_q} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}_q} \rho_{k\ell} \|\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}\|^2, \end{cases} \tag{3.5}$$

for $q = 1, \ldots, Q$. Note that we have kept the notation $\boldsymbol{w}_{\mathcal{C}(k)}$ in (3.5) to make the role of the regularization term clearer, even though we have $\boldsymbol{w}_{\mathcal{C}(k)} = \boldsymbol{w}_{\mathcal{C}_q}$ for all k in $\mathcal{C}_q$. The notation $\boldsymbol{w}_{-\mathcal{C}_q}$ denotes the collection of weight vectors estimated by the other clusters, that is, $\boldsymbol{w}_{-\mathcal{C}_q} = \{\boldsymbol{w}_{\mathcal{C}_p} : p = 1, \ldots, Q\} \setminus \{\boldsymbol{w}_{\mathcal{C}_q}\}$. The first term on the RHS of expression (3.5) penalizes the error on the streaming data. However, the second term enforces the prior information on the smoothness

of the graph signal $\{\boldsymbol{w}_{\mathcal{C}_1}, \ldots, \boldsymbol{w}_{\mathcal{C}_Q}\}$, with strength parameter $\eta \geq 0$. In [Chen et al., 2014c], the coefficients $\{\rho_{k\ell}\}$ used to adjust the regularization strength among neighbors were chosen to satisfy the conditions:

$$\sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)^-} \rho_{k\ell} = 1, \text{ and } \begin{cases} \rho_{k\ell} > 0, \text{ if } \ell \in \mathcal{N}_k \setminus \mathcal{C}(k), \\ \rho_{kk} \geq 0, \\ \rho_{k\ell} = 0, \text{ otherwise.} \end{cases} \tag{3.6}$$

In this way, it is clear that η controls the regularization strength while $\rho_{k\ell}$ adjusts the regularization strength between the neighbors. We impose $\rho_{k\ell} = 0$ for all $\ell \notin \mathcal{N}_k \setminus \mathcal{C}(k)^-$ since nodes belonging to the same cluster estimate the same parameter vector. Observe from (3.5) that the regularization strength between two clusters is directly related to the number of edges that connect them.

Following the same line of reasoning from [Cattivelli and Sayed, 2010, Sayed, 2014c] in the single-task case, and extending the argument to problem (3.5) by using Nash-equilibrium properties [Basar and Olsder, 1995, Rosen, 1965], the following diffusion strategy of the adapt-then-combine (ATC) form was derived in [Chen et al., 2014c] for solving the multitask learning problem (3.5) in a fully distributed manner:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \, \boldsymbol{x}_k(i) \left(d_k(i) - \boldsymbol{x}_k^\top(i) \boldsymbol{w}_k(i) \right) + \eta \, \mu_k \left(\displaystyle\sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)} \rho_{k\ell} (\boldsymbol{w}_\ell(i) - \boldsymbol{w}_k(i)) \right), \\ \boldsymbol{w}_k(i+1) = \displaystyle\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k} \, \boldsymbol{\psi}_\ell(i+1). \end{cases} \tag{3.7}$$

where $\boldsymbol{w}_k(i)$ denotes the estimate of the unknown parameter vector $\boldsymbol{w}_k^o$ at node k and iteration i, and μ_k is a positive step-size parameter. The combination coefficients $\{a_{\ell k}\}$ are nonnegative scalars that are chosen to satisfy the conditions:

$$\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k} = 1, \text{ and } \begin{cases} a_{\ell k} > 0, \text{ if } \ell \in \mathcal{N}_k \cap \mathcal{C}(k), \\ a_{\ell k} = 0, \text{ otherwise.} \end{cases} \tag{3.8}$$

As explained in Section 2.2, there are several ways to select these coefficients such as using the averaging rule or the Metropolis rule.

Starting from an initial condition $\boldsymbol{w}_k(0)$, at every time instant $i \geq 0$, the ATC strategy (3.7) performs two steps. In the first step, also called adaptation step, agent k receives from its neighbors that are outside its cluster their estimates $\boldsymbol{w}_\ell(i)$ and uses these information, along with its instantaneous noisy data, to update the estimate $\boldsymbol{w}_k(i)$ to an intermediate value $\boldsymbol{\psi}_k(i+1)$. This step involves a regularization term promoting relations between neighboring clusters. All other agents in the network are performing simultaneously a similar adaptation step. The second step is a combination step where agent k combines the intermediate iterates $\boldsymbol{\psi}_\ell(i+1)$ of its cluster neighbors to obtain $\boldsymbol{w}_k(i+1)$. Again, all other agents are performing simultaneously a similar combination step.

Observe that, if η is set to 0, algorithm (3.7) degenerates into the ATC diffusion LMS algorithm within each cluster and without information exchange between the clusters. The larger this factor is, the more the signal over the graph is smooth.

3.2.2 Asynchronous multitask diffusion adaptation

To model the asynchronous behavior over networks, we follow the same procedure developed in [Zhao and Sayed, 2015a] since the model presented in that work allows us to cover many situations of practical interest. Specifically, we replace each deterministic step-size μ_k by a random process $\mu_k(i)$, and model uncertainties in the links by using *random* combination coefficients $\{a_{\ell k}(i)\}$ and *random* regularization factors $\{\rho_{k\ell}(i)\}$. In other words, we modify the multitask diffusion strategy (3.7) to the following form:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k(i)\,\boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^{\top}(i)\,\boldsymbol{w}_k(i)\right) \\ \qquad\qquad + \eta\,\mu_k(i)\left(\displaystyle\sum_{\ell\in\mathcal{N}_k(i)\setminus\mathcal{C}(k)} \rho_{k\ell}(i)(\boldsymbol{w}_\ell(i) - \boldsymbol{w}_k(i))\right), \\ \boldsymbol{w}_k(i+1) = \displaystyle\sum_{\ell\in\mathcal{N}_k(i)\cap\mathcal{C}(k)} a_{\ell k}(i)\,\boldsymbol{\psi}_\ell(i+1) \end{cases} \tag{3.9}$$

where $\mathcal{N}_k(i)$ is also now random and denotes the random neighborhood of agent k at time instant i. The composition of each cluster is assumed to be known a priori and does not change over time. When dealing with multitask networks, compared to single-task networks [Zhao and Sayed, 2015a], a second source of uncertainty comes from links transmitting data between clusters. Indeed, data transmitted over intra-cluster links are used to reach a consensus while data transmitted over inter-cluster links are used to promote relationships between tasks.

The asynchronous network model is assumed to satisfy the following conditions:

- *Conditions on the step-size parameters:* At each time instant i, the step-size at node k is a bounded nonnegative random variable $\mu_k(i) \in [0, \mu_{\max,k}]$. These step-sizes are collected into the random matrix $\boldsymbol{M}(i) \triangleq \mathrm{diag}\{\mu_1(i), \ldots, \mu_N(i)\}$. We assume that $\{\boldsymbol{M}(i), i \geq 0\}$ is a weakly stationary random process with mean $\overline{\boldsymbol{M}}$ and Kronecker-covariance matrix $\boldsymbol{C}_M$ of size $N^2 \times N^2$ defined as

$$\boldsymbol{C}_M \triangleq \mathbb{E}\left\{(\boldsymbol{M}(i) - \overline{\boldsymbol{M}}) \otimes (\boldsymbol{M}(i) - \overline{\boldsymbol{M}})\right\}. \tag{3.10}$$

- *Conditions on the combination coefficients:* The random coefficients $\{a_{\ell k}(i)\}$ used to scale the estimates $\{\boldsymbol{\psi}_\ell(i+1)\}$ that are being received by node k from its cluster neighbors $\ell \in \mathcal{N}_k(i) \cap \mathcal{C}(k)$ satisfy the following constraints at each iteration i:

$$\sum_{\ell\in\mathcal{N}_k(i)\cap\mathcal{C}(k)} a_{\ell k}(i) = 1, \text{ and } \begin{cases} a_{\ell k}(i) > 0, \text{ if } \ell \in \mathcal{N}_k(i) \cap \mathcal{C}(k), \\ a_{\ell k}(i) = 0, \text{ otherwise.} \end{cases} \tag{3.11}$$

We collect these coefficients into the random $N \times N$ left-stochastic matrix $\boldsymbol{A}(i)$. We again assume that $\{\boldsymbol{A}(i), i \geq 0\}$ is a weakly stationary random process. Let $\overline{\boldsymbol{A}}$ be its mean and $\boldsymbol{C}_A$ its Kronecker-covariance matrix of size $N^2 \times N^2$ defined as

$$\boldsymbol{C}_A \triangleq \mathbb{E}\left\{(\boldsymbol{A}(i) - \overline{\boldsymbol{A}}) \otimes (\boldsymbol{A}(i) - \overline{\boldsymbol{A}})\right\}. \tag{3.12}$$

- *Conditions on the regularization factors:* The random factors $\{\rho_{k\ell}(i)\}$, which adjust the regularization strength between the parameter vectors at neighboring nodes of distinct clusters, satisfy the following constraints at each iteration i:

$$\sum_{\ell \in \mathcal{N}_k(i) \setminus \mathcal{C}(k)^-} \rho_{k\ell}(i) = 1, \text{ and } \begin{cases} \rho_{k\ell}(i) > 0, \text{ if } \ell \in \mathcal{N}_k(i) \setminus \mathcal{C}(k), \\ \rho_{kk}(i) \geq 0, \\ \rho_{k\ell}(i) = 0, \text{ otherwise.} \end{cases} \tag{3.13}$$

  We collect these coefficients into the random $N \times N$ right-stochastic matrix $\boldsymbol{P}_\rho(i)$. We assume that $\{\boldsymbol{P}_\rho(i), i \geq 0\}$ is a weakly stationary random process with mean $\overline{\boldsymbol{P}}_\rho$ and Kronecker-covariance matrix $\boldsymbol{C}_P$ of size $N^2 \times N^2$ defined as

$$\boldsymbol{C}_P \triangleq \mathbb{E}\left\{(\boldsymbol{P}_\rho(i) - \overline{\boldsymbol{P}}_\rho) \otimes (\boldsymbol{P}_\rho(i) - \overline{\boldsymbol{P}}_\rho)\right\}. \tag{3.14}$$

- *Independence assumptions:* To enable tractable analysis, we shall assume that the random matrices $\boldsymbol{M}(i)$, $\boldsymbol{A}(i)$, and $\boldsymbol{P}_\rho(i)$ at iteration i are mutually-independent and independent of any other random variables. These matrices are related to node, intra-cluster, and inter-cluster link failures, respectively.

- *Mean graph:* The mean matrices $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{P}}_\rho$ define the intra-cluster and inter-cluster neighborhoods, namely, $\mathcal{N}_k \cap \mathcal{C}(k)$ and $\mathcal{N}_k \setminus \mathcal{C}(k)$ for all k, respectively. We refer to the neighborhoods $\mathcal{N}_k = \big(\mathcal{N}_k \cap \mathcal{C}(k)\big) \cup \big(\mathcal{N}_k \setminus \mathcal{C}(k)\big)$ for all k, defined by $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{P}}_\rho$, as the mean graph.

  In view of the above conditions, the mean combination coefficients $\bar{a}_{\ell k} \triangleq \mathbb{E}\, a_{\ell k}(i)$ and regularization factors $\bar{\rho}_{k\ell} \triangleq \mathbb{E}\, \rho_{k\ell}(i)$ are nonnegative and satisfy the following constraints.

$$\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} \bar{a}_{\ell k} = 1, \text{ and } \begin{cases} \bar{a}_{\ell k} > 0, \text{ if } \ell \in \mathcal{N}_k \cap \mathcal{C}(k), \\ \bar{a}_{\ell k} = 0, \text{ otherwise,} \end{cases} \tag{3.15}$$

$$\sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)^-} \bar{\rho}_{k\ell} = 1, \text{ and } \begin{cases} \bar{\rho}_{k\ell} > 0, \text{ if } \ell \in \mathcal{N}_k \setminus \mathcal{C}(k), \\ \bar{\rho}_{kk} \geq 0, \\ \bar{\rho}_{k\ell} = 0, \text{ otherwise.} \end{cases} \tag{3.16}$$

At each time instant i, the matrix $\boldsymbol{P}_\rho(i) \otimes \boldsymbol{P}_\rho(i)$ has nonnegative entries since $\boldsymbol{P}_\rho(i)$ has nonnegative entries. It follows that $\overline{\boldsymbol{P}}_\rho$ and $\mathbb{E}\,\{\boldsymbol{P}_\rho(i) \otimes \boldsymbol{P}_\rho(i)\} \overset{(3.14)}{=} \overline{\boldsymbol{P}}_\rho \otimes \overline{\boldsymbol{P}}_\rho + \boldsymbol{C}_P$ have also nonnegative entries, and are right-stochastic since

$$\overline{\boldsymbol{P}}_\rho \mathbb{1}_N = \mathbb{E}\, \boldsymbol{P}_\rho(i)\, \mathbb{1}_N = \mathbb{1}_N, \tag{3.17}$$

and

$$(\overline{\boldsymbol{P}}_\rho \otimes \overline{\boldsymbol{P}}_\rho + \boldsymbol{C}_P)\mathbb{1}_{N^2} = \mathbb{E}\,\{(\boldsymbol{P}_\rho(i) \otimes \boldsymbol{P}_\rho(i))(\mathbb{1}_N \otimes \mathbb{1}_N)\} = \mathbb{E}\,\{(\boldsymbol{P}_\rho(i)\,\mathbb{1}_N) \otimes (\boldsymbol{P}_\rho(i)\,\mathbb{1}_N)\} = \mathbb{1}_{N^2}. \tag{3.18}$$

In the same token, the matrices $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{A}} \otimes \overline{\boldsymbol{A}} + \boldsymbol{C}_A$ are left-stochastic. Thus, we can state the following properties for the asynchronous model (3.9):

Property 3.1. *The $N \times N$ matrix $\overline{\boldsymbol{A}}$ and the $N^2 \times N^2$ matrix $\overline{\boldsymbol{A}} \otimes \overline{\boldsymbol{A}} + \boldsymbol{C}_A$ are left-stochastic matrices.*

Property 3.2. *The $N \times N$ matrix $\overline{\boldsymbol{P}}_\rho$ and the $N^2 \times N^2$ matrix $\overline{\boldsymbol{P}}_\rho \otimes \overline{\boldsymbol{P}}_\rho + \boldsymbol{C}_P$ are right-stochastic matrices.*

Property 3.3. *For every node k, the neighborhood $\mathcal{N}_k$ that is defined by the mean graph of the asynchronous model (3.9) is equal to the union of all possible realizations for the random neighborhood $\mathcal{N}_k(i) = \big(\mathcal{N}_k(i) \cap \mathcal{C}(k)\big) \cup \big(\mathcal{N}_k(i) \setminus \mathcal{C}(k)\big)$.*

Property 3.3 is established in Appendix C.1. We provide in Appendix C.2 one example for a common asynchronous network referred to as the Bernoulli network. The Bernoulli model proposed in [Zhao and Sayed, 2015a] is more general than the one used for modeling random link failures in consensus networks [Kar and Moura, 2008, 2009] since it also allows to consider random "on-off" behavior for agents. When dealing with multitask problems over asynchronous network, additional sources of uncertainties must be considered. The network provided in Appendix C.2 allows us to jointly model intra-cluster link failures, inter-cluster link failures, and random "on-off" behaviors for agents.

3.3 Stochastic performance analysis

The performance of the multitask diffusion algorithm (3.9) is affected by various random perturbations due to the asynchronous events. We now examine the stochastic behavior of this strategy in the mean and mean-square error sense by relying on the energy conservation framework.

3.3.1 Weight error vector recursion

For each agent k, we introduce the weight error vectors:

$$\widetilde{\boldsymbol{w}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_k(i), \qquad \widetilde{\boldsymbol{\psi}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\psi}_k(i) \tag{3.19}$$

where $\boldsymbol{w}_k^o$ is the optimum parameter vector at node k. We denote by $\widetilde{\boldsymbol{w}}_b(i)$, $\widetilde{\boldsymbol{\psi}}_b(i)$, and $\boldsymbol{w}_b^o$ the block weight error vector, the block intermediate weight error vector, and the block optimum

weight vector, all of size $N \times 1$ with blocks of size $M \times 1$, namely,

$$\widetilde{\boldsymbol{w}}_b(i) \triangleq \mathrm{col}\{\widetilde{\boldsymbol{w}}_1(i), \ldots, \widetilde{\boldsymbol{w}}_N(i)\}, \tag{3.20}$$

$$\widetilde{\boldsymbol{\psi}}_b(i) \triangleq \mathrm{col}\{\widetilde{\boldsymbol{\psi}}_1(i), \ldots, \widetilde{\boldsymbol{\psi}}_N(i)\}, \tag{3.21}$$

$$\boldsymbol{w}_b^o \triangleq \mathrm{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}. \tag{3.22}$$

We also introduce the following $N \times N$ block matrices with individual entries of size $M \times M$:

$$\boldsymbol{\mathcal{M}}(i) \triangleq \boldsymbol{M}(i) \otimes \boldsymbol{I}_M, \tag{3.23}$$

$$\boldsymbol{\mathcal{A}}(i) \triangleq \boldsymbol{A}(i) \otimes \boldsymbol{I}_M, \tag{3.24}$$

$$\boldsymbol{\mathcal{P}}_\rho(i) \triangleq \boldsymbol{P}_\rho(i) \otimes \boldsymbol{I}_M. \tag{3.25}$$

To perform the theoretical analysis, we introduce the following independence assumption.

Assumption 3.1. (Independent regressors) *The regression vectors $\boldsymbol{x}_k(i)$ arise from stationary random processes that are temporally stationary, temporally white, and independent over space with $\boldsymbol{R}_{x,k} = \mathbb{E}\,\{\boldsymbol{x}_k(i)\,\boldsymbol{x}_k^\top(i)\} > \boldsymbol{0}$.*

A direct consequence is that $\boldsymbol{x}_k(i)$ is independent of $\widetilde{\boldsymbol{w}}_\ell(j)$ for all ℓ and $j \leq i$. As explained in Chapter 2, this assumption is commonly used in adaptive filtering and helps to simplify the analysis. Furthermore, for sufficiently small step-sizes, performance results obtained under this assumption match well the actual performance of stand alone filters [Sayed, 2008, App. 24.A].

The estimation error in the first step of the asynchronous strategy (3.9) can be rewritten as:

$$d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i) = \boldsymbol{x}_k^\top(i)\widetilde{\boldsymbol{w}}_k(i) + z_k(i). \tag{3.26}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the adaptation step in (3.9) and using the above relation, we can express the update equation for $\widetilde{\boldsymbol{\psi}}_b(i+1)$ as:

$$\widetilde{\boldsymbol{\psi}}_b(i+1) = \big[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\,\boldsymbol{\mathcal{Q}}(i))\big]\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{\mathcal{M}}(i)\boldsymbol{p}_{xz}(i) + \eta\,\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\boldsymbol{w}_b^o \tag{3.27}$$

where

$$\boldsymbol{\mathcal{Q}}(i) \triangleq \boldsymbol{I}_{MN} - \boldsymbol{\mathcal{P}}_\rho(i), \tag{3.28}$$

while $\boldsymbol{\mathcal{R}}_x(i)$ is an $N \times N$ block matrix with individual entries of size $M \times M$ given by:

$$\boldsymbol{\mathcal{R}}_x(i) \triangleq \mathrm{diag}\big\{\boldsymbol{x}_1(i)\boldsymbol{x}_1^\top(i), \ldots, \boldsymbol{x}_N(i)\boldsymbol{x}_N^\top(i)\big\}, \tag{3.29}$$

and $\boldsymbol{p}_{xz}(i)$ is the $N \times 1$ block column vector with blocks of size $M \times 1$ defined as:

$$\boldsymbol{p}_{xz}(i) \triangleq \mathrm{col}\big\{\boldsymbol{x}_1(i)z_1(i), \ldots, \boldsymbol{x}_N(i)z_N(i)\big\}. \tag{3.30}$$

Upon subtracting $\boldsymbol{w}_k^o$ from both sides of the combination step in (3.9), we obtain the block weight error vector:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{A}}^\top(i)\,\widetilde{\boldsymbol{\psi}}_b(i+1). \tag{3.31}$$

Substituting (3.27) into (3.31), we find that the error dynamics of the asynchronous multitask diffusion strategy (3.9) evolves according to the following recursion:

$$\boxed{\begin{aligned}\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{A}}^{\top}(i)\big[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\,\boldsymbol{\mathcal{Q}}(i))\big]\widetilde{\boldsymbol{w}}_b(i) - \\ \boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\boldsymbol{p}_{xz}(i) + \eta\,\boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\boldsymbol{w}_b^o.\end{aligned}} \tag{3.32}$$

For compactness of notation, we introduce the symbols:

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}(i)\big[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\,\boldsymbol{\mathcal{Q}}(i))\big], \tag{3.33}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\,\boldsymbol{p}_{xz}(i), \tag{3.34}$$

$$\boldsymbol{r}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o, \tag{3.35}$$

so that (3.32) can be written as

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}(i)\,\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{g}(i) + \eta\,\boldsymbol{r}(i). \tag{3.36}$$

For ease of reference, we list in Table 3.2 the various symbols that will be defined throughout the analysis.

3.3.2 Mean-error analysis

Upon taking the expectation of both sides of the above recursion, and using Assumption 3.1, and the independence of $\boldsymbol{A}(i)$, $\boldsymbol{M}(i)$, and $\boldsymbol{P}_\rho(i)$, the evolution of the mean error vector of the network is governed by the following dynamics:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) + \eta\,\boldsymbol{r} \tag{3.37}$$

with

$$\boldsymbol{\mathcal{B}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{B}}(i) = \boldsymbol{\mathcal{A}}^{\top}\big[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\,\boldsymbol{\mathcal{Q}})\big] \tag{3.38}$$

$$\boldsymbol{r} \triangleq \mathbb{E}\,\boldsymbol{r}(i) = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{Q}}\,\boldsymbol{w}_b^o, \tag{3.39}$$

where $\boldsymbol{\mathcal{A}}$, $\boldsymbol{\mathcal{M}}$, $\boldsymbol{\mathcal{R}}_x$, and $\boldsymbol{\mathcal{Q}}$ denote the expectations of $\boldsymbol{\mathcal{A}}(i)$, $\boldsymbol{\mathcal{M}}(i)$, $\boldsymbol{\mathcal{R}}_x(i)$, and $\boldsymbol{\mathcal{Q}}(i)$, respectively, and are given by:

$$\boldsymbol{\mathcal{A}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{A}}(i) = \overline{\boldsymbol{A}} \otimes \boldsymbol{I}_M, \tag{3.40}$$

$$\boldsymbol{\mathcal{M}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{M}}(i) = \overline{\boldsymbol{M}} \otimes \boldsymbol{I}_M, \tag{3.41}$$

$$\boldsymbol{\mathcal{P}}_\rho \triangleq \mathbb{E}\,\boldsymbol{\mathcal{P}}_\rho(i) = \overline{\boldsymbol{P}}_\rho \otimes \boldsymbol{I}_M, \tag{3.42}$$

$$\boldsymbol{\mathcal{R}}_x \triangleq \mathbb{E}\,\boldsymbol{\mathcal{R}}_x(i) = \mathrm{diag}\{\boldsymbol{R}_{x,1}, \ldots, \boldsymbol{R}_{x,N}\} \tag{3.43}$$

$$\boldsymbol{\mathcal{Q}} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{Q}}(i) = \boldsymbol{I}_{MN} - \mathbb{E}\,\boldsymbol{\mathcal{P}}_\rho(i) = \boldsymbol{I}_{MN} - \boldsymbol{\mathcal{P}}_\rho. \tag{3.44}$$

Note that $\mathbb{E}\,\boldsymbol{g}(i) = 0$ since $z_k(i)$ is zero-mean and independent of any other signal.

Table 3.2: List of symbols defined throughout the performance analysis in Chapter 3.

Symbol	Equation
$\boldsymbol{w}_b^o = \operatorname{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}$	(3.22)
$\boldsymbol{\mathcal{A}} = \overline{\boldsymbol{A}} \otimes \boldsymbol{I}_M$	(3.40)
$\boldsymbol{\mathcal{M}} = \overline{\boldsymbol{M}} \otimes \boldsymbol{I}_M$	(3.41)
$\boldsymbol{\mathcal{P}}_\rho = \overline{\boldsymbol{P}}_\rho \otimes \boldsymbol{I}_M$	(3.42)
$\boldsymbol{\mathcal{Q}} = \boldsymbol{I}_{MN} - \boldsymbol{\mathcal{P}}_\rho$	(3.44)
$\boldsymbol{\mathcal{R}}_x = \operatorname{diag}\{\boldsymbol{R}_{x,1}, \ldots, \boldsymbol{R}_{x,N}\}$	(3.43)
$\boldsymbol{\mathcal{S}} = \operatorname{diag}\left\{\sigma_{z,1}^2 \boldsymbol{R}_{x,1}, \ldots, \sigma_{z,N}^2 \boldsymbol{R}_{x,N}\right\}$	(3.65)
$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}}^\top \left[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\, \boldsymbol{\mathcal{Q}})\right]$	(3.38)
$\boldsymbol{r} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{Q}}\, \boldsymbol{w}_b^o$	(3.39)
$\boldsymbol{\mathcal{M}}_{\mathrm{I}} = \left(\overline{\boldsymbol{M}} \otimes \overline{\boldsymbol{M}} + \boldsymbol{C}_M\right) \otimes \boldsymbol{I}_{M^2}$	(3.54)
$\boldsymbol{\mathcal{A}}_{\mathrm{I}} = \left(\overline{\boldsymbol{A}} \otimes \overline{\boldsymbol{A}} + \boldsymbol{C}_A\right) \otimes \boldsymbol{I}_{M^2}$	(3.55)
$\boldsymbol{\mathcal{P}}_{\mathrm{I}} = \left(\overline{\boldsymbol{P}}_\rho \otimes \overline{\boldsymbol{P}}_\rho + \boldsymbol{C}_P\right) \otimes \boldsymbol{I}_{M^2}$	(3.56)
$\boldsymbol{\mathcal{Q}}_{\mathrm{I}} = \left(\boldsymbol{I}_{N^2} - \boldsymbol{I}_N \otimes \overline{\boldsymbol{P}}_\rho - \overline{\boldsymbol{P}}_\rho \otimes \boldsymbol{I}_N + \overline{\boldsymbol{P}}_\rho \otimes \overline{\boldsymbol{P}}_\rho + \boldsymbol{C}_P\right) \otimes \boldsymbol{I}_{M^2}$	(3.57)
$\boldsymbol{g}_b = \boldsymbol{\mathcal{A}}_{\mathrm{I}}^\top \boldsymbol{\mathcal{M}}_{\mathrm{I}} \operatorname{bvec}(\boldsymbol{\mathcal{S}})$	(3.64)
$\boldsymbol{r}_b = \boldsymbol{\mathcal{A}}_{\mathrm{I}}^\top \boldsymbol{\mathcal{M}}_{\mathrm{I}} \boldsymbol{\mathcal{Q}}_{\mathrm{I}} \operatorname{bvec}\left(\boldsymbol{w}_b^o (\boldsymbol{w}_b^o)^\top\right)$	(3.67)
$\boldsymbol{\mathcal{K}} = \boldsymbol{\mathcal{A}}_{\mathrm{I}}^\top \Big[\big(\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}} \boldsymbol{\mathcal{Q}} \boldsymbol{w}_b^o\big) - \boldsymbol{\mathcal{M}}_{\mathrm{I}}\big((\boldsymbol{\mathcal{R}}_x \otimes_b \boldsymbol{\mathcal{Q}} \boldsymbol{w}_b^o) + \eta \boldsymbol{\mathcal{Q}}_{\mathrm{I}} (\boldsymbol{I}_{MN} \otimes_b \boldsymbol{w}_b^o)\big)\Big]$	(3.71)
$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{A}}_{\mathrm{I}}^\top \left[\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) \otimes_b \boldsymbol{I}_{MN}\right]$	(3.62)
$\boldsymbol{y}_b(i) = \boldsymbol{g}_b + \eta^2\, \boldsymbol{r}_b + 2\, \eta\, \boldsymbol{\mathcal{K}}\, \mathbb{E}\, \widetilde{\boldsymbol{w}}_b(i)$	(3.73)

Theorem 3.1. (Stability in the mean) *Assume that the data model in* (3.1) *and Assumption 3.1 hold. Then, for any initial condition, the multitask diffusion LMS strategy in* (3.9) *applied to asynchronous networks converges asymptotically in the mean if, and only if, the step-sizes in* $\boldsymbol{\mathcal{M}}$ *satisfy:*

$$\rho\Big(\boldsymbol{\mathcal{A}}^{\top}\big[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\,\boldsymbol{\mathcal{Q}})\big]\Big)<1. \tag{3.45}$$

In that case, the asymptotic mean bias is given by:

$$\lim_{i\to\infty}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)=\eta\,(\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{B}})^{-1}\boldsymbol{r}. \tag{3.46}$$

Assume that the expected values for all step-sizes are equal, namely, $\mathbb{E}\,\mu_k(i)=\bar{\mu}$ *for all* k*. Then, a sufficient condition for* (3.45) *to hold is*

$$0<\bar{\mu}<\frac{2}{\max\limits_{1\le k\le N}\lambda_{\max}(\boldsymbol{R}_{x,k})+2\eta}. \tag{3.47}$$

Proof. Convergence in the mean requires the matrix $\boldsymbol{\mathcal{B}}$ in (3.37) to be contractive, i.e., $\rho(\boldsymbol{\mathcal{B}})<1$. Since any induced matrix norm is lower bounded by its spectral radius, we can write:

$$\begin{aligned}\rho\Big(\boldsymbol{\mathcal{A}}^{\top}[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\,\boldsymbol{\mathcal{Q}})]\Big) &\le \|\boldsymbol{\mathcal{A}}^{\top}[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\,\boldsymbol{\mathcal{Q}})]\|_{b,\infty}\\ &\le \|\boldsymbol{\mathcal{A}}^{\top}\|_{b,\infty}\cdot\|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\,\boldsymbol{\mathcal{Q}})\|_{b,\infty},\end{aligned} \tag{3.48}$$

where we used the submultiplicative property of the block maximum norm. Using property (A.23) and the triangle inequality property, we obtain:

$$\begin{aligned}\rho\big(\boldsymbol{\mathcal{A}}^{\top}[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\,\boldsymbol{\mathcal{Q}})]\big) &\le \|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta\boldsymbol{\mathcal{Q}})\|_{b,\infty}\\ &= \|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x+\eta(\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{P}}_{\rho}))\|_{b,\infty}\\ &\le \|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x-\eta\boldsymbol{\mathcal{M}}\|_{b,\infty}+\eta\|\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_{\rho}\|_{b,\infty}.\end{aligned} \tag{3.49}$$

Consider the first term on the RHS of (3.49). Since the matrices $\boldsymbol{\mathcal{M}}$ and $\boldsymbol{\mathcal{R}}_x$ are block diagonal, it holds from property (A.22) that:

$$\begin{aligned}\|\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x-\eta\boldsymbol{\mathcal{M}}\|_{b,\infty} &= \max_{1\le k\le N}\rho\big((1-\eta\,\bar{\mu}_k)\boldsymbol{I}_M-\bar{\mu}_k\boldsymbol{R}_{x,k}\big)\\ &= \max_{1\le k\le N}\max_{1\le m\le M}|(1-\eta\,\bar{\mu}_k)-\bar{\mu}_k\lambda_m(\boldsymbol{R}_{x,k})|\end{aligned} \tag{3.50}$$

where $\bar{\mu}_k\triangleq\mathbb{E}\,\mu_k(i)$, and $\lambda_m(\cdot)$ denotes the m-th eigenvalue of its matrix argument. Consider now the second term on the RHS of (3.49). Using the submultiplicative property of the block maximum norm, and property (A.24) for the block right-stochastic matrix $\boldsymbol{\mathcal{P}}_{\rho}$, we get:

$$\eta\|\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_{\rho}\|_{b,\infty}\le\eta\|\boldsymbol{\mathcal{M}}\|_{b,\infty}. \tag{3.51}$$

Because $\boldsymbol{\mathcal{M}}$ is a block diagonal matrix, we further have that

$$\|\boldsymbol{\mathcal{M}}\|_{b,\infty}=\max_{1\le k\le N}\bar{\mu}_k. \tag{3.52}$$

Combining (3.50) and (3.52), we conclude that the algorithm is stable in the mean if

$$\max_{1\leq k\leq N}\max_{1\leq m\leq M}|1-\eta\,\bar{\mu}_k-\bar{\mu}_k\lambda_m(\boldsymbol{R}_{x,k})|+\eta\max_{1\leq k\leq N}\bar{\mu}_k<1. \tag{3.53}$$

In order to simplify this condition, assume that $\bar{\mu}_k=\bar{\mu}$ for all k. Condition (3.53) then reduces to (3.47). □

Note that the randomness in the topology does not affect the condition for stability in the mean of the algorithm. Furthermore, observe that the mean recursion of the *synchronous* multitask algorithm over the *mean-graph* topology has the same form as the mean recursion (3.37) with the same coefficient matrix $\boldsymbol{\mathcal{B}}$ in (3.38) and the same vector $\boldsymbol{r}$ in (3.39). Hence, we conclude that the mean behavior of the multitask algorithm (3.9) is immune to the effect of asynchronous events.

3.3.3 Mean-square-error analysis

To perform the mean-square-error analysis over *asynchronous* networks, compared to *synchronous* networks [Chen et al., 2014c], new operators with additional properties must be introduced. We shall use the block Kronecker product operator $\otimes_b$ instead of the Kronecker product $\otimes$, and the block vectorization operator $\mathrm{bvec}(\cdot)$ instead of the vectorization operator $\mathrm{vec}(\cdot)$. This is because, as explained in [Zhao and Sayed, 2015b, Sayed, 2014a], these block operators preserve the locality of the blocks in the original matrix arguments. Overviews on the block Kronecker product operator is provided in Appendix A.2. Recall that if $\boldsymbol{\mathcal{X}}$ is an $N\times N$ block matrix with blocks of size $M\times M$, $\mathrm{bvec}(\boldsymbol{\mathcal{X}})$ vectorizes each block of $\boldsymbol{\mathcal{X}}$ and stacks the vectors on top of each other.

We now use the properties (A.13)–(A.19) to evaluate the expectation of some block Kronecker matrix products that will be useful in the sequel:

$$\begin{aligned}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\triangleq\mathbb{E}\,\{\boldsymbol{\mathcal{M}}(i)\otimes_b\boldsymbol{\mathcal{M}}(i)\}&=\mathbb{E}\,\big\{(\boldsymbol{M}(i)\otimes\boldsymbol{I}_M)\otimes_b(\boldsymbol{M}(i)\otimes\boldsymbol{I}_M)\big\}\\&\overset{(A.15)}{=}\mathbb{E}\,\big\{(\boldsymbol{M}(i)\otimes\boldsymbol{M}(i))\otimes(\boldsymbol{I}_M\otimes\boldsymbol{I}_M)\big\}\\&\overset{(3.10)}{=}(\overline{\boldsymbol{M}}\otimes\overline{\boldsymbol{M}}+\boldsymbol{C}_M)\otimes\boldsymbol{I}_{M^2}.\end{aligned} \tag{3.54}$$

In the same way, we get the following expectations:

$$\boldsymbol{\mathcal{A}}_{\mathrm{I}}\triangleq\mathbb{E}\,\{\boldsymbol{\mathcal{A}}(i)\otimes_b\boldsymbol{\mathcal{A}}(i)\}=(\overline{\boldsymbol{A}}\otimes\overline{\boldsymbol{A}}+\boldsymbol{C}_A)\otimes\boldsymbol{I}_{M^2}, \tag{3.55}$$

$$\boldsymbol{\mathcal{P}}_{\mathrm{I}}\triangleq\mathbb{E}\,\{\boldsymbol{\mathcal{P}}_\rho(i)\otimes_b\boldsymbol{\mathcal{P}}_\rho(i)\}=(\overline{\boldsymbol{P}}_\rho\otimes\overline{\boldsymbol{P}}_\rho+\boldsymbol{C}_P)\otimes\boldsymbol{I}_{M^2}. \tag{3.56}$$

Since $\boldsymbol{\mathcal{Q}}(i)=\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{P}}_\rho(i)$, we also obtain:

$$\boldsymbol{\mathcal{Q}}_{\mathrm{I}}\triangleq\mathbb{E}\,\{\boldsymbol{\mathcal{Q}}(i)\otimes_b\boldsymbol{\mathcal{Q}}(i)\}=\big(\boldsymbol{I}_{N^2}-\boldsymbol{I}_N\otimes\overline{\boldsymbol{P}}_\rho-\overline{\boldsymbol{P}}_\rho\otimes\boldsymbol{I}_N+\overline{\boldsymbol{P}}_\rho\otimes\overline{\boldsymbol{P}}_\rho+\boldsymbol{C}_P\big)\otimes\boldsymbol{I}_{M^2}. \tag{3.57}$$

To analyze the convergence in mean-square-error sense of the multitask diffusion LMS algorithm (3.9) over asynchronous networks, we consider the variance of the weight error vector

$\widetilde{\boldsymbol{w}}_b(i)$ weighted by any positive semi-definite matrix $\boldsymbol{\Sigma}$, that is, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$, where $\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}} \triangleq \widetilde{\boldsymbol{w}}_b^{\,\top}(i)\,\boldsymbol{\Sigma}\,\widetilde{\boldsymbol{w}}_b(i)$. As explained in Section 2.4, the freedom in selecting $\boldsymbol{\Sigma}$ will allow us to extract various types of information about the network and the nodes. By Assumption 3.1 and using (3.36), we get:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\Sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}'} + \mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} + \eta^2\mathbb{E}\,\|\boldsymbol{r}(i)\|^2_{\boldsymbol{\Sigma}} + 2\eta\mathbb{E}\,\big\{\boldsymbol{r}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\big\}, \quad (3.58)$$

where $\boldsymbol{\Sigma}' = \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\}$. Let $\boldsymbol{\sigma}$ denotes the $(MN)^2 \times 1$ vector representation of $\boldsymbol{\Sigma}$ that is obtained by the block vectorization operator, namely, $\boldsymbol{\sigma} \triangleq \mathrm{bvec}(\boldsymbol{\Sigma})$. In the sequel, it will be more convenient to work with $\boldsymbol{\sigma}$ than with $\boldsymbol{\Sigma}$ itself and we shall use the alternative notation $\|\boldsymbol{x}\|^2_{\boldsymbol{\sigma}}$ to refer to the weighted square quantity $\|\boldsymbol{x}\|^2_{\boldsymbol{\Sigma}} = \boldsymbol{x}^{\top}\boldsymbol{\Sigma}\boldsymbol{x}$. Let $\boldsymbol{\sigma}' \triangleq \mathrm{bvec}(\boldsymbol{\Sigma}')$. Using property (A.17), we can verify that:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}^{\top}\boldsymbol{\sigma} \quad (3.59)$$

where $\boldsymbol{\mathcal{F}}$ is the $(MN)^2 \times (MN)^2$ matrix given by:

$$\begin{aligned}
\boldsymbol{\mathcal{F}} &\triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}(i) \otimes_b \boldsymbol{\mathcal{B}}(i)\} \\
&\overset{(A.14)}{=} \mathbb{E}\,\{\boldsymbol{\mathcal{A}}^{\top}(i) \otimes_b \boldsymbol{\mathcal{A}}^{\top}(i)\} \\
&\qquad \mathbb{E}\,\big\{[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\boldsymbol{\mathcal{Q}}(i))] \otimes_b [\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\boldsymbol{\mathcal{Q}}(i))]\big\} \\
&\overset{(3.55),(A.13)}{=} \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\Big[\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\boldsymbol{\mathcal{Q}}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\boldsymbol{\mathcal{Q}}) \otimes_b \boldsymbol{I}_{MN} + \\
&\qquad \mathbb{E}\,\big\{\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\boldsymbol{\mathcal{Q}}(i)) \otimes_b \boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i) + \eta\boldsymbol{\mathcal{Q}}(i))\big\}\Big] \quad (3.60)
\end{aligned}$$

where using property (A.14) and the definition of $\boldsymbol{\mathcal{M}}_{\mathrm{I}}$ in (3.54), we have

$$\mathbb{E}\,\big\{\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\boldsymbol{\mathcal{Q}}(i))\otimes_b\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\boldsymbol{\mathcal{Q}}(i))\big\} = \boldsymbol{\mathcal{M}}_{\mathrm{I}}\,\mathbb{E}\,\big\{(\boldsymbol{\mathcal{R}}_x(i)+\eta\boldsymbol{\mathcal{Q}}(i))\otimes_b(\boldsymbol{\mathcal{R}}_x(i)+\eta\boldsymbol{\mathcal{Q}}(i))\big\}. \quad (3.61)$$

The term on the RHS of equation (3.61) is proportional to $\boldsymbol{\mathcal{M}}_{\mathrm{I}} = \mathbb{E}\,\{\boldsymbol{M}(i) \otimes \boldsymbol{M}(i)\} \otimes \boldsymbol{I}_{M^2}$, where $\mathbb{E}\,\{\boldsymbol{M}(i) \otimes \boldsymbol{M}(i)\}$ is an $N \times N$ block diagonal matrix for which the k-th block is an $N \times N$ diagonal matrix with ℓ-th entry given by $\mathbb{E}\,\{\mu_k(i)\mu_\ell(i)\}$. It is sufficient for the exposition in this work to focus on the case of sufficiently small step-sizes (sufficiently small upper bounds $\mu_{\max,k}$) so that terms that depend on higher order powers of the step-sizes can be ignored. Such approximations are common when analyzing stochastic gradient algorithms and diffusion strategies in the mean-square-error sense (see Subsection 2.4.3 and [Sayed, 2014c, Section 6.5]). Accordingly, the last term in (3.60) can be neglected and we continue our discussion by letting:

$$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\big[\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\boldsymbol{\mathcal{Q}}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta\boldsymbol{\mathcal{Q}}) \otimes_b \boldsymbol{I}_{MN}\big], \qquad \text{(small step-sizes)}. \quad (3.62)$$

Consider next the second term on the RHS of (3.58). We can write:

$$\mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} = \mathrm{Tr}\big(\mathbb{E}\,\{\boldsymbol{g}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{g}(i)\}\big) = \mathrm{Tr}\big(\boldsymbol{\Sigma}\,\mathbb{E}\,\{\boldsymbol{g}(i)\,\boldsymbol{g}^{\top}(i)\}\big) \overset{(A.16)}{=} \boldsymbol{g}_b^{\top}\boldsymbol{\sigma}, \quad (3.63)$$

where we used the fact that $\mathrm{Tr}(\boldsymbol{X}\boldsymbol{Y}) = \mathrm{Tr}(\boldsymbol{Y}\boldsymbol{X})$ for any two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ of compatible dimensions and where $\boldsymbol{g}_b \triangleq \mathrm{bvec}\big(\mathbb{E}\,\{\boldsymbol{g}(i)\,\boldsymbol{g}^{\top}(i)\}\big)$. Using expression (3.34) and the definitions of $\boldsymbol{\mathcal{M}}_{\mathrm{I}}$ and $\boldsymbol{\mathcal{A}}_{\mathrm{I}}$ in (3.54) and (3.55), we have

$$\begin{aligned}
\boldsymbol{g}_b &= \mathrm{bvec}\Big(\mathbb{E}\,\big\{\boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\,\boldsymbol{p}_{xz}(i)\,\boldsymbol{p}_{xz}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{A}}(i)\big\}\Big) \\
&\overset{(A.17)}{=} \mathbb{E}\,\Big\{\big(\boldsymbol{\mathcal{A}}^{\top}(i)\otimes_b\boldsymbol{\mathcal{A}}^{\top}(i)\big)\,\mathrm{bvec}\,\big(\boldsymbol{\mathcal{M}}(i)\,\boldsymbol{p}_{xz}(i)\,\boldsymbol{p}_{xz}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\big)\Big\} \\
&\overset{(A.17),(A.18)}{=} \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\,\mathbb{E}\,\Big\{\big(\boldsymbol{\mathcal{M}}(i)\otimes_b\boldsymbol{\mathcal{M}}(i)\big)\,\mathrm{bvec}\,\big(\boldsymbol{p}_{xz}(i)\,\boldsymbol{p}_{xz}^{\top}(i)\big)\Big\} \\
&= \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\,\mathrm{bvec}(\boldsymbol{\mathcal{S}}),
\end{aligned} \tag{3.64}$$

where

$$\boldsymbol{\mathcal{S}} \triangleq \mathbb{E}\,\{\boldsymbol{p}_{xz}(i)\boldsymbol{p}_{xz}^{\top}(i)\} = \mathrm{diag}\,\big\{\sigma_{z,1}^2\boldsymbol{R}_{x,1},\ldots,\sigma_{z,N}^2\boldsymbol{R}_{x,N}\big\}. \tag{3.65}$$

Let us examine now the third term on the RHS of (3.58):

$$\mathbb{E}\,\|\boldsymbol{r}(i)\|_{\boldsymbol{\Sigma}}^2 = \mathrm{Tr}\big(\boldsymbol{\Sigma}\,\mathbb{E}\,\{\boldsymbol{r}(i)\,\boldsymbol{r}^{\top}(i)\}\big) \overset{(A.16)}{=} \boldsymbol{r}_b^{\top}\boldsymbol{\sigma} \tag{3.66}$$

where $\boldsymbol{r}_b \triangleq \mathrm{bvec}\big(\mathbb{E}\,\{\boldsymbol{r}(i)\,\boldsymbol{r}^{\top}(i)\}\big)$. Using expression (3.35), property (A.17), and the definitions of $\boldsymbol{\mathcal{M}}_{\mathrm{I}}$, $\boldsymbol{\mathcal{A}}_{\mathrm{I}}$, and $\boldsymbol{\mathcal{Q}}_{\mathrm{I}}$ in (3.54), (3.55), and (3.57), and proceeding as in (3.64), we obtain the following expression:

$$\boldsymbol{r}_b = \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\boldsymbol{\mathcal{Q}}_{\mathrm{I}}\,\mathrm{bvec}\,\big(\boldsymbol{w}_b^o(\boldsymbol{w}_b^o)^{\top}\big). \tag{3.67}$$

Consider now the fourth term $\mathbb{E}\,\big\{\boldsymbol{r}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\big\}$. We have:

$$\begin{aligned}
\mathbb{E}\,\big\{\boldsymbol{r}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\big\} &= \mathbb{E}\,\Big\{\mathrm{bvec}\big(\boldsymbol{r}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\big)\Big\} \\
&\overset{(A.17)}{=} \mathbb{E}\,\Big\{\big(\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\big)^{\top}\otimes_b\boldsymbol{r}^{\top}(i)\Big\}\,\boldsymbol{\sigma} \\
&\overset{(A.18)}{=} \mathbb{E}\,\big\{\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\otimes_b\boldsymbol{r}(i)\big\}^{\top}\boldsymbol{\sigma} \\
&\overset{(A.14)}{=} \mathbb{E}\,\big\{\widetilde{\boldsymbol{w}}_b(i)\otimes_b 1\big\}^{\top}\mathbb{E}\,\big\{\boldsymbol{\mathcal{B}}(i)\otimes_b\boldsymbol{r}(i)\big\}^{\top}\boldsymbol{\sigma} \\
&= \big(\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\big)^{\top}\mathbb{E}\,\big\{\boldsymbol{\mathcal{B}}(i)\otimes_b\boldsymbol{r}(i)\big\}^{\top}\boldsymbol{\sigma}
\end{aligned} \tag{3.68}$$

with

$$\begin{aligned}
\mathbb{E}\,\{\boldsymbol{\mathcal{B}}(i)\otimes_b\boldsymbol{r}(i)\} &= \mathbb{E}\,\Big\{\boldsymbol{\mathcal{A}}^{\top}(i)\big[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\,\boldsymbol{\mathcal{Q}}(i))\big]\otimes_b\boldsymbol{\mathcal{A}}^{\top}(i)\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o\Big\} \\
&\overset{(A.14)}{=} \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\,\mathbb{E}\,\Big\{\big[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\,\boldsymbol{\mathcal{Q}}(i))\big]\otimes_b\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o\Big\} \\
&\overset{(A.13)}{=} \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\Big[\big(\boldsymbol{I}_{MN}\otimes_b\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{Q}}\boldsymbol{w}_b^o\big) - \\
&\qquad\qquad \mathbb{E}\,\big\{\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\,\boldsymbol{\mathcal{Q}}(i))\otimes_b\boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o\big\}\Big],
\end{aligned} \tag{3.69}$$

where

$$\begin{aligned}
&\mathbb{E}\left\{\boldsymbol{\mathcal{M}}(i)(\boldsymbol{\mathcal{R}}_x(i)+\eta\,\boldsymbol{\mathcal{Q}}(i))\otimes_b \boldsymbol{\mathcal{M}}(i)\boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o\right\}\\
&\overset{(A.14)}{=}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\,\mathbb{E}\left\{(\boldsymbol{\mathcal{R}}_x(i)+\eta\,\boldsymbol{\mathcal{Q}}(i))\otimes_b \boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o\right\}\\
&\overset{(A.13)}{=}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\,((\boldsymbol{\mathcal{R}}_x\otimes_b \boldsymbol{\mathcal{Q}}\,\boldsymbol{w}_b^o)+\eta\,\mathbb{E}\,\boldsymbol{\mathcal{Q}}(i)\otimes_b \boldsymbol{\mathcal{Q}}(i)\,\boldsymbol{w}_b^o)\\
&\overset{(A.14)}{=}\boldsymbol{\mathcal{M}}_{\mathrm{I}}\left((\boldsymbol{\mathcal{R}}_x\otimes_b \boldsymbol{\mathcal{Q}}\,\boldsymbol{w}_b^o)+\eta\,\boldsymbol{\mathcal{Q}}_{\mathrm{I}}(\boldsymbol{I}_{MN}\otimes_b \boldsymbol{w}_b^o)\right).
\end{aligned}\tag{3.70}$$

Finally, combining (3.69) and (3.70) and introducing the notation $\boldsymbol{\mathcal{K}}$, we get:

$$\boldsymbol{\mathcal{K}}\triangleq\mathbb{E}\,\{\boldsymbol{\mathcal{B}}(i)\otimes_b\boldsymbol{r}(i)\}=\boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\Big[\big(\boldsymbol{I}_{MN}\otimes_b\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{Q}}\boldsymbol{w}_b^o\big)-\boldsymbol{\mathcal{M}}_{\mathrm{I}}\big((\boldsymbol{\mathcal{R}}_x\otimes_b\boldsymbol{\mathcal{Q}}\boldsymbol{w}_b^o)+\eta\boldsymbol{\mathcal{Q}}_{\mathrm{I}}(\boldsymbol{I}_{MN}\otimes_b\boldsymbol{w}_b^o)\big)\Big].\tag{3.71}$$

Relation (3.58) can be written in a more compact form as:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2=\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\mathcal{F}}^{\top}\boldsymbol{\sigma}}^2+\boldsymbol{y}_b^{\top}(i)\boldsymbol{\sigma},\tag{3.72}$$

where $\boldsymbol{y}_b(i)$ is the $(MN)^2\times 1$ vector given by:

$$\boldsymbol{y}_b(i)\triangleq\boldsymbol{g}_b+\eta^2\,\boldsymbol{r}_b+2\,\eta\,\boldsymbol{\mathcal{K}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i).\tag{3.73}$$

Theorem 3.2. (Mean-square stability) *Assume data model* (3.1) *and Assumption 3.1 hold. Then, the asynchronous diffusion multitask algorithm* (3.9) *is mean-square stable if the algorithm is mean stable and if the matrix* $\boldsymbol{\mathcal{F}}$ *defined by* (3.60) *is contractive.*

Proof. Provided that $\boldsymbol{\mathcal{F}}$ is contractive, recursion (3.72) is stable if $\boldsymbol{y}_b^{\top}(i)\boldsymbol{\sigma}$ is bounded. Since $\eta,\boldsymbol{g}_b,\boldsymbol{r}_b,\boldsymbol{\mathcal{K}}$, and $\boldsymbol{\sigma}$ are finite and constant terms, the boundedness of $\boldsymbol{y}_b^{\top}(i)\boldsymbol{\sigma}$ depends on $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ being bounded. We know from (3.37) that $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ is uniformly bounded because (3.37) is a BIBO stable recursion with a bounded driving term $\eta\,\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{Q}}\,\boldsymbol{w}_b^o$. Then, $\boldsymbol{y}_b^{\top}(i)\boldsymbol{\sigma}$ is uniformly bounded, and $\mathbb{E}\,\{\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2\}$ converges to a bounded value as $i\to\infty$.

Assume sufficiently small step-sizes where the approximation (3.62) is justified by ignoring higher-order powers of the step-sizes. The condition ensuring that the matrix in (3.62) is contractive is derived in Appendix C.3. It is worth noting that, due to the Kronecker covariance matrix $\boldsymbol{C}_A$ (arising from the asynchronous events), the matrix $\boldsymbol{\mathcal{F}}$ cannot be approximated by $\boldsymbol{\mathcal{B}}\otimes\boldsymbol{\mathcal{B}}$ as in the previous chapter or as in the synchronous case [Chen et al., 2014c]. Moreover, deriving a condition that ensures that $\boldsymbol{\mathcal{F}}$ is contractive in a multitask setting is more challenging than in the single-task setting [Zhao and Sayed, 2015b] due to the presence of the non-block diagonal matrix $\boldsymbol{\mathcal{Q}}$ (arising from the cooperation between clusters) on the RHS of (3.62). □

Theorem 3.3. (Transient performance) *Assume the same settings as Theorem 3.2. The variance curve defined by* $\zeta(i)\triangleq\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\sigma}}^2$ *evolves according to the following recursion for* $i\geq 0$*:*

$$\zeta(i+1)=\zeta(i)+\|\widetilde{\boldsymbol{w}}(0)\|_{(\boldsymbol{\mathcal{F}}^{\top}-\boldsymbol{I}_{(MN)^2})(\boldsymbol{\mathcal{F}}^{\top})^i\boldsymbol{\sigma}}^2+\big(\boldsymbol{y}_b^{\top}(i)+\boldsymbol{\Upsilon}(i)\big)\boldsymbol{\sigma}\tag{3.74}$$

where $\boldsymbol{\Upsilon}(i+1)$ *is updated as follows:*

$$\boldsymbol{\Upsilon}(i+1) = \boldsymbol{\Upsilon}(i)\,\boldsymbol{\mathcal{F}}^{\top} + \boldsymbol{y}_b^{\top}(i)(\boldsymbol{\mathcal{F}}^{\top} - \boldsymbol{I}_{(MN)^2}), \tag{3.75}$$

with the initial conditions $\zeta(0) = \|\widetilde{\boldsymbol{w}}(0)\|^2_{\boldsymbol{\sigma}}$ *and* $\boldsymbol{\Upsilon}(0) = \boldsymbol{0}_{1\times(MN)^2}$.

Proof. Iterating (3.72) starting from $i = 0$, we find:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(0)\|^2_{(\boldsymbol{\mathcal{F}}^{\top})^{i+1}\boldsymbol{\sigma}} + \sum_{j=0}^{i} \boldsymbol{y}_b^{\top}(i-j)(\boldsymbol{\mathcal{F}}^{\top})^j\boldsymbol{\sigma}. \tag{3.76}$$

Comparing the above recursion at instants i and $i+1$, we obtain recursion (3.74) with

$$\boldsymbol{\Upsilon}(i) = \sum_{j=1}^{i-1} \boldsymbol{y}_b^{\top}(i-j)(\boldsymbol{\mathcal{F}}^{\top})^j - \sum_{j=0}^{i-1} \boldsymbol{y}_b^{\top}(i-1-j)(\boldsymbol{\mathcal{F}}^{\top})^j \tag{3.77}$$

which verifies recursion (3.75). □

Theorem 3.3 allows us to derive several performance metrics through the proper selection of $\boldsymbol{\Sigma}$. For instance, the network mean-square-deviation (MSD) value at time instant i, defined by

$$\mathrm{MSD}_{\mathrm{net}}(i) \triangleq \frac{1}{N}\sum_{k=1}^{N} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2 = \frac{1}{N}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2, \tag{3.78}$$

is obtained for $\boldsymbol{\Sigma} = \frac{1}{N}\boldsymbol{I}_{MN}$. The MSD of cluster $\mathcal{C}_q$ at time instant i is defined as:

$$\mathrm{MSD}_{\mathcal{C}_q}(i) \triangleq \frac{1}{\mathrm{card}\{\mathcal{C}_q\}}\sum_{k\in\mathcal{C}_q} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_k(i)\|^2. \tag{3.79}$$

This quantity can be obtained by computing $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}_{\mathcal{C}_q}}$ with a block diagonal weighting matrix $\boldsymbol{\Sigma}_{\mathcal{C}_q}$ that has the block $\frac{1}{\mathrm{card}\{\mathcal{C}_q\}}\boldsymbol{I}_M$ as k-th entry, for all $k \in \mathcal{C}_q$, and zeros elsewhere.

Theorem 3.4. (Steady-state performance) *Assume the same settings as Theorem 3.2. Assume mean and mean-square stability. Then, the steady-state performance* $\zeta^{\star}$ *defined as* $\zeta^{\star} \triangleq \lim_{i\to\infty}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$ *for the multitask diffusion LMS* (3.9) *applied to asynchronous network is given by:*

$$\zeta^{\star} = \Big(\boldsymbol{g}_b + \eta^2\,\boldsymbol{r}_b + 2\eta\,\boldsymbol{\mathcal{K}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}(\infty)\Big)^{\top}\Big(\boldsymbol{I}_{(MN)^2} - \boldsymbol{\mathcal{F}}^{\top}\Big)^{-1}\boldsymbol{\sigma}. \tag{3.80}$$

where $\mathbb{E}\,\{\widetilde{\boldsymbol{w}}(\infty)\}$ *is given by* (3.46).

Proof. From the recursive expression (3.72), we obtain as $i \to \infty$:

$$\lim_{i\to\infty}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{(\boldsymbol{I}_{(MN)^2}-\boldsymbol{\mathcal{F}}^{\top})\boldsymbol{\sigma}} = \big(\boldsymbol{g}_b + \eta^2\,\boldsymbol{r}_b + 2\eta\,\boldsymbol{\mathcal{K}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}(\infty)\big)^{\top}\boldsymbol{\sigma}. \tag{3.81}$$

To obtain $\zeta^{\star}$, we replace $\boldsymbol{\sigma}$ in (3.81) by $\big(\boldsymbol{I}_{(MN)^2} - \boldsymbol{\mathcal{F}}^{\top}\big)^{-1}\boldsymbol{\sigma}$. □

The steady-state network MSD is obtained from Theorem 3.4 by setting $\boldsymbol{\sigma} = \mathrm{bvec}(\boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma} = \frac{1}{N}\boldsymbol{I}_{MN}$ and the steady-state MSD of cluster $\mathcal{C}_q$ is obtained by setting $\boldsymbol{\sigma} = \mathrm{bvec}(\boldsymbol{\Sigma}_{\mathcal{C}_q})$.

Before moving on to the presentation of experimental results, note that the performance of the *synchronous* multitask algorithm over the *mean-graph* topology can be obtained by setting $\boldsymbol{C}_A$, $\boldsymbol{C}_M$, and $\boldsymbol{C}_P$ to zero in (3.54)–(3.56).

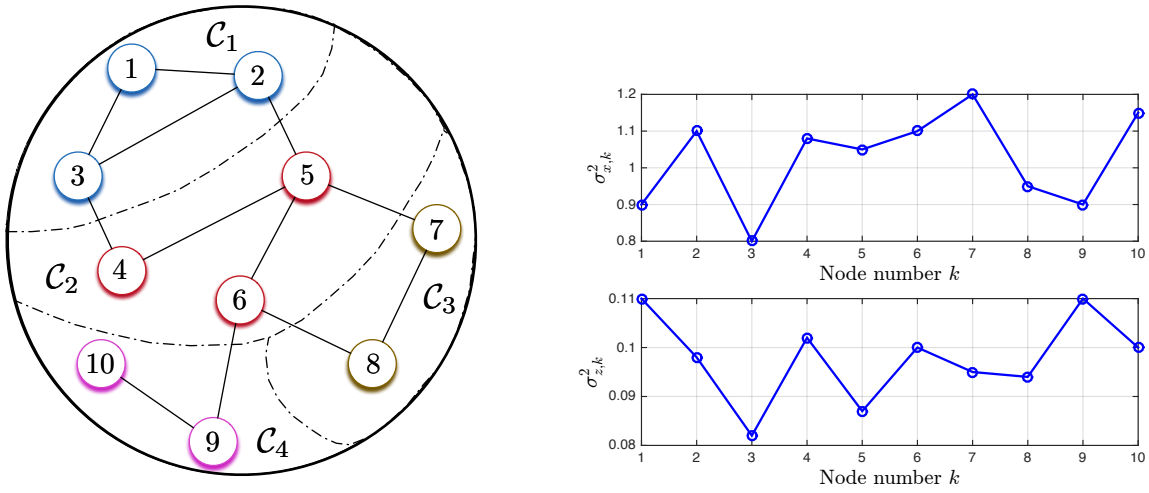


Figure 3.2: Experimental setup. (*Left*) Network topology. (*Right*) Regression and noise variances.

3.4 Simulation results

3.4.1 Illustrative example

We adopt the same clustered multitask network as [Chen et al., 2014c] in our simulations. As shown in Figure 3.2, the network consists of 10 nodes divided into 4 clusters: $\mathcal{C}_1 = \{1, 2, 3\}$, $\mathcal{C}_2 = \{4, 5, 6\}$, $\mathcal{C}_3 = \{7, 8\}$, and $\mathcal{C}_4 = \{9, 10\}$. The unknown parameter vector $\boldsymbol{w}^o_{\mathcal{C}_q}$ of each cluster is of size 2×1, and has the following form: $\boldsymbol{w}^o_{\mathcal{C}_q} = \boldsymbol{w}_0 + \delta\boldsymbol{w}_{\mathcal{C}_q}$ with $\boldsymbol{w}_0 = [0.5, -0.4]^\top$, $\delta\boldsymbol{w}_{\mathcal{C}_1} = [0.0287, -0.005]^\top$, $\delta\boldsymbol{w}_{\mathcal{C}_2} = [0.0234, 0.005]^\top$, $\delta\boldsymbol{w}_{\mathcal{C}_3} = [-0.0335, 0.0029]^\top$, and $\delta\boldsymbol{w}_{\mathcal{C}_4} = [0.0224, 0.00347]^\top$. The input and output data at each node k are related via the linear regression model: $d_k(i) = \boldsymbol{x}_k^\top(i)\boldsymbol{w}^o_k + z_k(i)$ where $\boldsymbol{w}^o_k = \boldsymbol{w}^o_{\mathcal{C}(k)}$. The regressors are zero-mean 2×1 random vectors governed by a Gaussian distribution with covariance matrices $\boldsymbol{R}_{x,k} = \sigma^2_{x,k}\boldsymbol{I}_M$. The variances $\sigma^2_{x,k}$ are shown in Figure 3.2. The background noises $z_k(i)$ are i.i.d. zero-mean Gaussian random variables, independent of any other signals. The corresponding variances are given in Figure 3.2.

We considered the Bernoulli asynchronous model described in Appendix C.2. We set the coefficient $a_{\ell k}$ in (C.6) such that $a_{\ell k} = 1/\text{card}\{\mathcal{N}_k \cap \mathcal{C}(k)\}$ for all $\ell \in (\mathcal{N}_k \cap \mathcal{C}(k))$. Then we set the regularization factors $\rho_{k\ell}$ in (C.10) as follows. If $\mathcal{N}_k \setminus \mathcal{C}(k) \neq \varnothing$, $\rho_{k\ell}$ was set to $\rho_{k\ell} = 1/\text{card}\{\mathcal{N}_k \setminus \mathcal{C}(k)\}$ for all $\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)$, and to $\rho_{k\ell} = 0$ for any other ℓ. If $\mathcal{N}_k \setminus \mathcal{C}(k) = \varnothing$, these factors were set to $\rho_{kk} = 1$ and to $\rho_{k\ell} = 0$ for all $\ell \neq k$. This usually leads to asymmetrical regularization factors. The parameters of the Bernoulli distribution governing the step-sizes $\mu_k(i)$ were the same over the network, that is, we set μ_k in (C.4) to 0.03 for all k. The regularization strength η was set to 1. The MSD learning curves were averaged over 100 Monte-Carlo runs. The transient MSD curves were obtained with Theorem 3.3, and the steady-state MSD was estimated with Theorem 3.4.

In Figure 3.3 (left), we report the network MSD learning curves for 3 different cases:

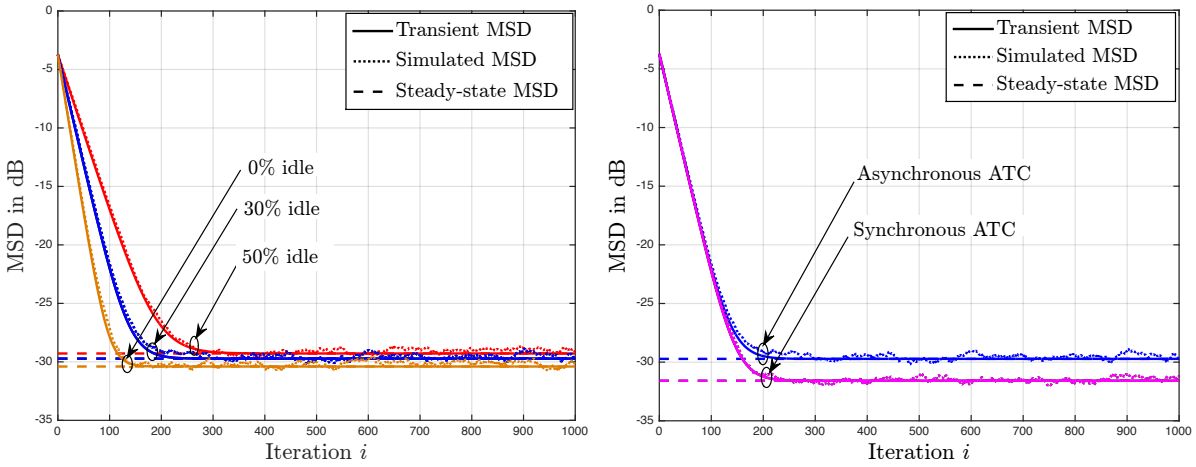


Figure 3.3: (*Left*) Comparison of asynchronous network MSD under 50% idle, 30% idle, and 0% idle. (*Right*) Network MSD comparison of synchronous network under 30% idle and the corresponding synchronous network.

Case 1: 50% idle: $p_{\mu,k} = p_{a,\ell k} = p_{\rho,k\ell} = 0.5$;

Case 2: 30% idle: $p_{\mu,k} = p_{a,\ell k} = p_{\rho,k\ell} = 0.7$;

Case 3: 0% idle: $p_{\mu,k} = p_{a,\ell k} = p_{\rho,k\ell} = 1$.

We observe that the simulation results match well the theoretical results. Furthermore, the performance of the network is influenced by the probability of occurrence of random events. In Figure 3.3 (right), the asynchronous algorithm in Case 2 is compared with its synchronous version obtained from (3.7) by setting μ_k, $a_{\ell k}$, and $\rho_{k\ell}$ to the expected values $\bar{\mu}_k = \mathbb{E}\,\mu_k(i)$, $\bar{a}_{\ell k} = \mathbb{E}\,a_{\ell k}(i)$, and $\bar{\rho}_{k\ell} = \mathbb{E}\,\rho_{k\ell}(i)$, respectively. Although both algorithms show the same convergence rate, the asynchronous algorithm suffers from degradation in its MSD performance caused by the additional randomness throughout the adaptation process.

3.4.2 Multitask learning benefit

In this section, we provide an example to show the benefit of multitask learning. We consider a network consisting of $N = 100$ nodes grouped into $Q = 3$ clusters such that $\mathcal{C}_1 = \{1, \ldots, 70\}$, $\mathcal{C}_2 = \{71, \ldots, 90\}$, and $\mathcal{C}_3 = \{91, \ldots, 100\}$. The physical connections are defined by the connectivity matrix represented in Figure 3.4. The inputs $\boldsymbol{x}_k(i)$ were zero-mean 21×1 random vectors governed by a Gaussian distribution with covariance matrix $\boldsymbol{R}_{x,k} = \sigma_{x,k}^2 \boldsymbol{I}_{21}$, where $\sigma_{x,k}^2$ were randomly chosen in the interval $[1, 1.4]$. The noises $z_k(i)$ were i.i.d. zero-mean Gaussian random variables, independent of any other signal with variances $\sigma_{z,k}^2$ randomly chosen in the interval $[0.1, 0.15]$. The 21×1 unknown parameter vectors were chosen as: $\boldsymbol{w}_{\mathcal{C}_1}^o = \boldsymbol{w}_0 = [\mathbb{1}_{1\times 3}, \boldsymbol{0}_{1\times 3}, 2 \cdot \mathbb{1}_{1\times 3}, \boldsymbol{0}_{1\times 3}, -\mathbb{1}_{1\times 3}, \boldsymbol{0}_{1\times 3}, -2 \cdot \mathbb{1}_{1\times 3}]^\top$, $\boldsymbol{w}_{\mathcal{C}_2}^o = \boldsymbol{w}_0 + \delta\boldsymbol{w}$, $\boldsymbol{w}_{\mathcal{C}_3}^o = \boldsymbol{w}_0 - \delta\boldsymbol{w}$ where $\delta\boldsymbol{w}$ was randomly generated such that $\|\delta\boldsymbol{w}\|_\infty = \max_m |[\delta\boldsymbol{w}]_m| = 0.03$.

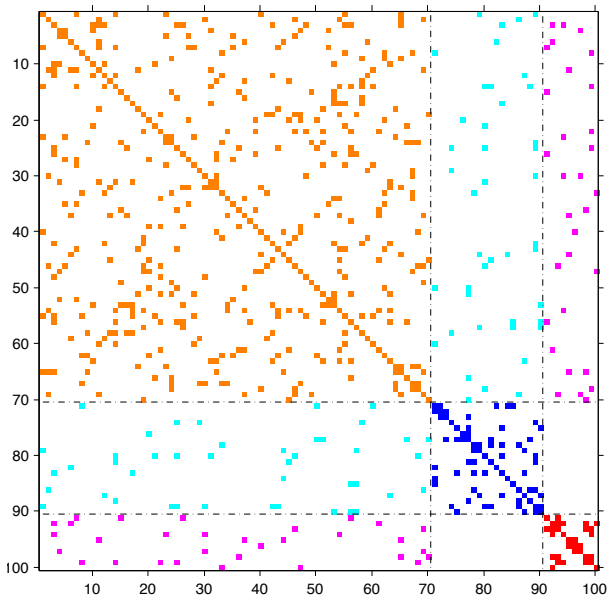


Figure 3.4: Connectivity matrix of the network. The orange, blue, and red elements correspond to links within $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, respectively. The cyan elements correspond to links between $\mathcal{C}_1$ and $\mathcal{C}_2$ and the magenta elements correspond to links between $\mathcal{C}_1$ and $\mathcal{C}_3$. No links between $\mathcal{C}_2$ and $\mathcal{C}_3$.

We considered the Bernoulli asynchronous model. The coefficients $\{a_{\ell k}\}$ and $\{\rho_{k\ell}\}$ in (C.6) and (C.10), respectively, were generated in the same manner as in 3.4.1. Parameters μ_k and $p_{\mu,k}$ in (C.4) were set to $\mu_k = 1/30$, $p_{\mu,k} = 0.8$ for nodes in the first cluster, $\mu_k = 2/45$, $p_{\mu,k} = 0.6$ for nodes in the second cluster, and $\mu_k = 1/15$, $p_{\mu,k} = 0.4$ for nodes in the third cluster. The probabilities $\{p_{a,\ell k}\}$ in (C.6) were $p_{a,\ell k} = 0.8$ for links in the first cluster, $p_{a,\ell k} = 0.6$ for links in the second cluster, and $p_{a,\ell k} = 0.4$ for links in the third cluster. The probability that a link connecting two nodes belonging to neighboring clusters drops was $1-p_{\rho,k\ell} = 0.25$. The simulated curves were obtained by averaging over 150 Monte-Carlo runs.

In Figure 3.5 (left), we compare two algorithms: the asynchronous diffusion strategy without regularization (obtained from (3.9) by setting $\eta = 0$) and its synchronous counterpart (obtained from (3.9) by setting $\eta = 0$ and replacing $\mu_k(i), a_{\ell k}(i)$ by $\bar{\mu}_k, \bar{a}_{\ell k}$). As shown in this figure, the performance is highly deteriorated in the third cluster and slightly deteriorated in the first cluster because $\mathcal{C}_3$ is more susceptible to random events. In Figure 3.5 (right), we compare two algorithms: the asynchronous diffusion strategy with regularization (obtained from (3.9) by setting $\eta = 2$) and the same synchronous algorithm as in the left plot. As shown in this figure, the cooperation between clusters improves the performance of each cluster so that gaps appearing in the left plot are reduced. In other words, $\mathcal{C}_2$ and $\mathcal{C}_3$ benefit from the high performance levels achieved by $\mathcal{C}_1$. This can be justified by two arguments: a large number of nodes is employed to collectively estimate $\boldsymbol{w}^o_{\mathcal{C}_1}$ and the probabilities associated with random events in $\mathcal{C}_1$ are small. As a conclusion, when tasks between neighboring clusters are similar, cooperation among clusters

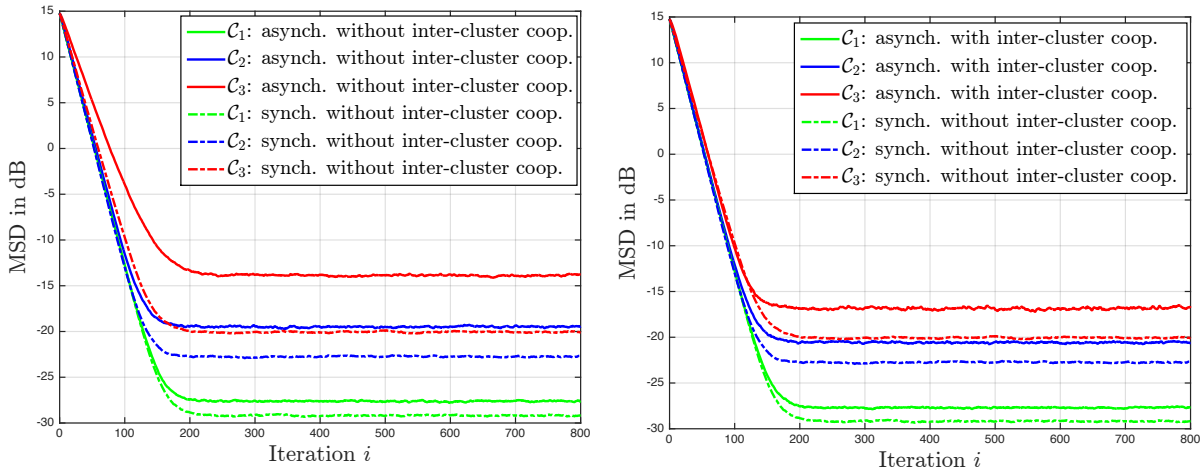


Figure 3.5: Cluster learning curves. (*Left*) Comparison of the asynchronous multitask diffusion LMS (3.9) without inter-cluster cooperation ($\eta = 0$) and its synchronous counterpart. (*Right*) Comparison of the asynchronous multitask diffusion LMS (3.9) with inter-cluster cooperation ($\eta \neq 0$) and the multitask diffusion LMS (3.9) without inter-cluster cooperation ($\eta = 0$).

improves the learning especially for clusters where asynchronous events occur frequently.

3.4.3 Circular arcs localization

In this section, we consider the problem of adaptive surface localization over asynchronous networks. When dealing with a smooth target surface, we can expect that promoting the smoothness of the graph signal will improve the performance of the network [Chen et al., 2014c]. Note that most of the existing works on diffusion strategies over adaptive (or biological) networks focus on localizing and tracking a point target (e.g., a projectile, predator, etc.) by the whole network [Sayed, 2014c, Tu and Sayed, 2011b, Cattivelli and Sayed, 2011]. In this section, we consider the case where the target cannot be reduced to a point [Chen et al., 2014c, Section VI-B]. Particularly, we consider an arc localization application where the radius of the arc is changing over time, and we illustrate the influence of the random events on the learning behavior and tracking ability of the network.

Let us denote by $\mathcal{L} = [\theta_0, \theta_1]$ an arc of circle with radius R and subtending an angle $\theta = \theta_1 - \theta_0$ with the circle center $\boldsymbol{w}_o$. Let us decompose $\mathcal{L}$ into Q sub-arcs $\mathcal{L}_q$ with radius R and subtending an angle $\delta \ll \theta$ with $\boldsymbol{w}_o$. In order to estimate the location of $\mathcal{L}$, and for sufficiently small δ, it is sufficient to estimate the location of each of these Q sub-arcs by solving a point target localization problem. This can be done by employing a network of N nodes, composed of Q clusters, where nodes of each cluster $\mathcal{C}_q$ are interested in locating $\mathcal{L}_q$ by estimating a parameter

vector $\boldsymbol{w}_{\mathcal{C}_q}^o$, which can be expressed as:

$$\boldsymbol{w}_{\mathcal{C}_q}^o = \boldsymbol{w}_o + R \begin{pmatrix} \cos\left(\theta_0 + \dfrac{\theta}{Q}(q - \dfrac{1}{2})\right) \\ \sin\left(\theta_0 + \dfrac{\theta}{Q}(q - \dfrac{1}{2})\right) \end{pmatrix}, \quad \forall q = 1, \ldots, Q, \tag{3.82}$$

where $\theta = \theta_1 - \theta_0$. Let us consider node k belonging to cluster $\mathcal{C}_q$. At each time instant i, node k gets noisy measurements $\{d_k(i), \boldsymbol{x}_k(i)\}$ that are related via the linear data model [Sayed, 2014c]:

$$d_k(i) = \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{\mathcal{C}_q}^o + z_k(i), \tag{3.83}$$

where $z_k(i)$ is a zero-mean temporally and spatially independent Gaussian noise with variance $\sigma_{z,k}^2$, $\boldsymbol{x}_k(i)$ is a noisy measurement of the unit-norm direction vector of $\boldsymbol{x}_k$ pointing from agent k to the target $\boldsymbol{w}_{\mathcal{C}_q}^o$ given by:

$$\boldsymbol{x}_k(i) = \boldsymbol{x}_k + \alpha_k(i)\boldsymbol{x}_k^\perp + \beta_k(i)\boldsymbol{x}_k, \tag{3.84}$$

with $\boldsymbol{x}_k$ given by $\boldsymbol{x}_k = (\boldsymbol{w}_{\mathcal{C}_q}^o - \boldsymbol{n}_k)/\|\boldsymbol{w}_{\mathcal{C}_q}^o - \boldsymbol{n}_k\|$ where $\boldsymbol{n}_k$ is the location vector of node k, $\boldsymbol{x}_k^\perp$ denoting a unit norm vector that lies in the same space as $\boldsymbol{x}_k$ and whose direction is perpendicular to $\boldsymbol{x}_k$. The variables $\alpha_k(i)$ and $\beta_k(i)$ are zero-mean independent Gaussian random variables of variances $\sigma_{\alpha,k}^2$ and $\sigma_{\beta,k}^2$, respectively. The amount of perturbation along the parallel direction is assumed to be small compared to the amount of perturbation along the perpendicular direction, that is, $\sigma_{\beta,k}^2 \ll \sigma_{\alpha,k}^2$.

To show the effects of randomness at the level of nodes and links, we considered a network of 100 nodes grouped into $Q = 10$ clusters, located over arcs of radiuses uniformly distributed between $3R_0$ and $5R_0$ given R_0. Angular parameters θ_0 and θ_1 were set to $13\pi/8$ and $15\pi/8$, respectively. The network topology is shown in Figure 3.6. The noise variances were set to $\sigma_{z,k}^2 = 0.2$, $\sigma_{\alpha,k}^2 = 0.05$, and $\sigma_{\beta,k}^2 = 0.005$, for all k. We considered a Bernoulli asynchronous model. The coefficients $a_{\ell k}$ in (C.6) were set to $1/\text{card}\{\mathcal{N}_k \cap \mathcal{C}(k)\}$ for intra-cluster links, and to zero for inter-cluster links. The regularization factors $\rho_{k\ell}$ in (C.10) were set to $1/\text{card}\{\mathcal{N}_k \backslash \mathcal{C}(k)\}$. The probabilities of success $p_{\mu,k}$, $p_{a,\ell k}$, and $p_{\rho,k\ell}$ were identically set to 0.5.

The MSD learning curves were averaged over 200 Monte-Carlo runs. We ran the synchronous and asynchronous multitask algorithms in two different situations. For the first one, we set the regularization strength η to zero, that is, we did not allow any cooperation between neighboring clusters. In the second one, we set the regularization strength η to 0.5. For comparison purposes, we also ran the non-cooperative LMS given by:

$$\boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k(i)\boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\right), \quad k = 1, \ldots, N, \tag{3.85}$$

which was obtained from (3.9) by setting $\boldsymbol{A}(i) = \boldsymbol{P}_\rho(i) = \boldsymbol{I}_N$ for all i, and the standard single-task ATC diffusion LMS:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k(i)\,\boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^\top(i)\,\boldsymbol{w}_k(i)\right) \\ \boldsymbol{w}_k(i+1) = \displaystyle\sum_{\ell \in \mathcal{N}_k(i)} a_{\ell k}(i)\,\boldsymbol{\psi}_\ell(i+1) \end{cases} \tag{3.86}$$

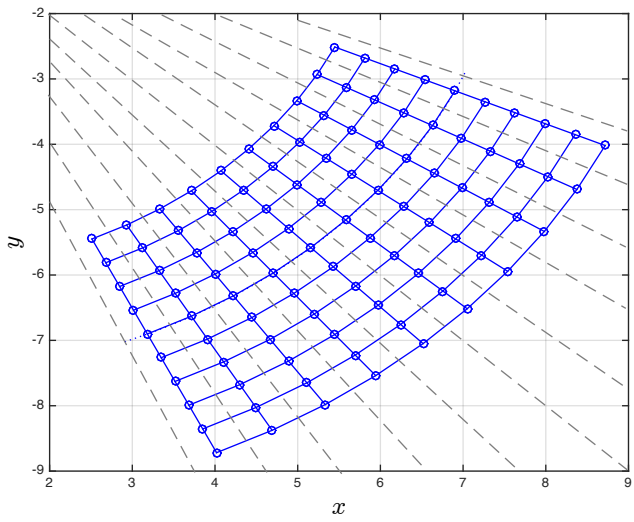


Figure 3.6: Network topology consisting of 10 clusters: circles for nodes, solid lines for links, and dashed lines for cluster boundaries.

which was obtained by setting $a_{\ell k}$ in (C.6) to $1/\text{card}\{\mathcal{N}_k\}$ for all $\ell \in \mathcal{N}_k$ and $\rho_{k\ell} = 0$. In both cases, synchronous and asynchronous algorithms were also considered. Each synchronous algorithm was derived from its asynchronous counterpart by making $\mu_k(i)$, $a_{\ell k}(i)$, and $\rho_{k\ell}(i)$ deterministic quantities equal to $\bar{\mu}_k$, $\bar{a}_{\ell k}$, and $\bar{\rho}_{k\ell}$, respectively. In order to illustrate the tracking ability of the algorithms, we modified the radius R of $\mathcal{L}$ every 500 iterations such that: for $i \in [0, 500]$, $R = 0.5R_0$, for $i \in]500, 1000]$, $R = R_0$, for $i \in]1000, 1500]$, $R = 1.5R_0$, and for $i \in]1500, 2000]$, $R = 2R_0$. Note that varying R has an effect on the level of similarity between neighboring tasks when characterized by $\|\boldsymbol{w}^o_{\mathcal{C}_p} - \boldsymbol{w}^o_{\mathcal{C}_q}\|^2$, where $\mathcal{C}_p$ and $\mathcal{C}_q$ denote two neighboring clusters. Indeed, with the topology shown in Figure 3.6, we obtain from (3.82):

$$\|\boldsymbol{w}^o_{\mathcal{C}_p} - \boldsymbol{w}^o_{\mathcal{C}_q}\|^2 = R^2\big(2 - 2\cos(\theta/Q)\big). \tag{3.87}$$

Figure 3.7 shows that cooperation among clusters improved the network MSD performance and endowed the network with robustness towards asynchronous events. We also observe that the performance of the standard diffusion LMS algorithm deteriorates when the level of similarity between tasks decreases. Figure 3.8 depicts the estimated arc when $R = R_0$ for the following algorithms in an asynchronous setting: non-cooperative LMS (3.85), standard diffusion LMS (3.86), and multitask diffusion LMS (3.9). In each case, the results were averaged over 150 Monte-Carlo runs and over 50 samples after convergence. The multitask diffusion algorithm outperformed the non-cooperative LMS and the standard diffusion. The standard diffusion was not able to estimate the location of the target since it is a single-task algorithm. It is shown in [Chen and Sayed, 2013] that standard diffusion LMS converges to a Pareto optimal solution when it is applied to multitask problems.

Finally, in order to show the effects of the number of clusters (or tasks) on the performance of the network, we considered 2 additional experimental setups. In the first one represented

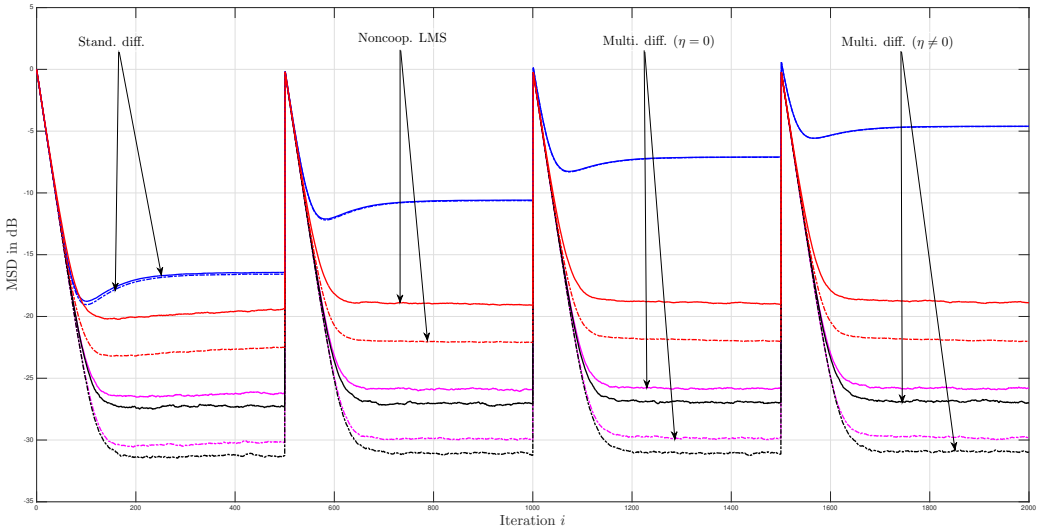


Figure 3.7: Network topology consisting of 10 clusters. Network MSD learning curves in a non-stationary environment: comparison of the multitask diffusion LMS with (namely, $\eta > 0$) and without (namely, $\eta = 0$) inter-cluster cooperation, the standard diffusion LMS (3.86) and the non-cooperative LMS (3.85). The dotted lines are for synchronous networks and the solid lines are for asynchronous networks.

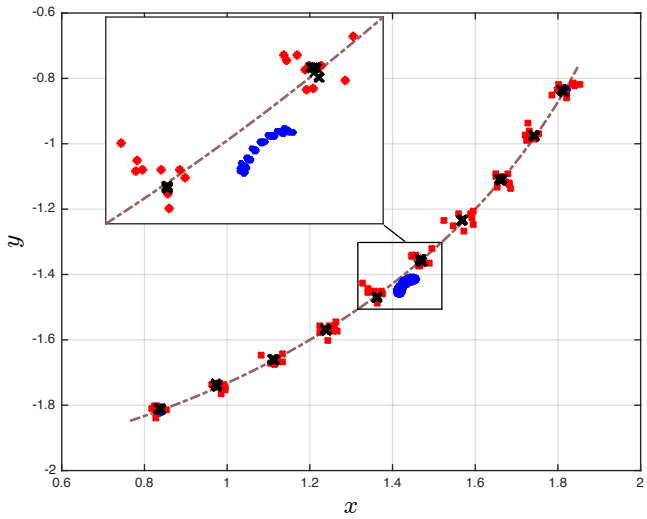


Figure 3.8: Target estimation results ($R = R_0 = 2$) over asynchronous network: black cross sign for multitask diffusion (3.9), red asterisk sign for non-cooperative (3.85), and blue circle sign for standard diffusion (3.86).

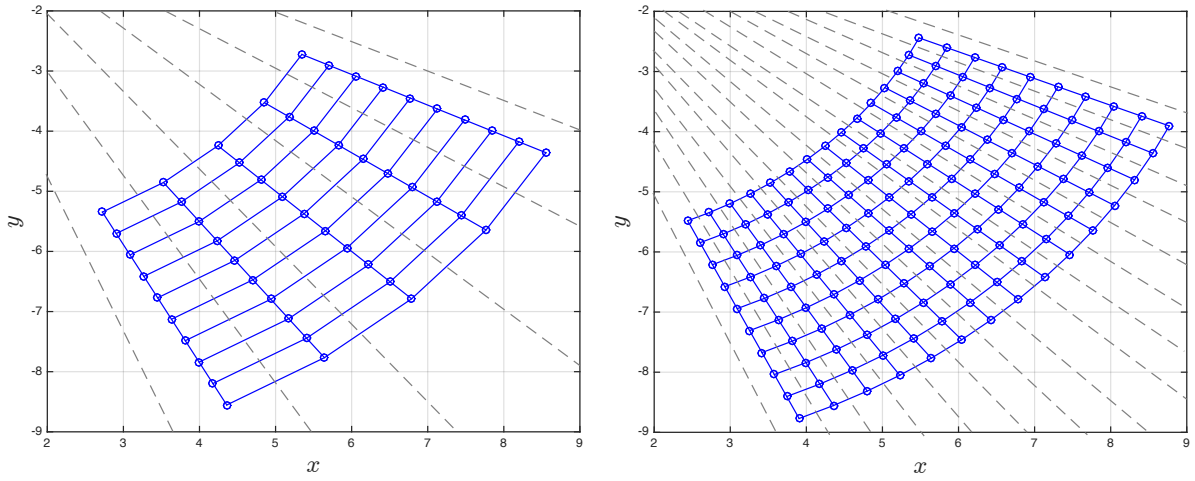


Figure 3.9: Network topology: circles for nodes, solid lines for links, and dashed lines for cluster boundaries. (*Left*) Network consisting of 5 clusters. (*Right*) Network consisting of 15 clusters.

in Figure 3.9 (left), the number of tasks was set to 5, that is, the arc $\mathcal{L}$ was decomposed into 5 sub-arcs. In the second one depicted in Figure 3.9 (right), the number of clusters was set to 15. Except for these changes, we considered the same experimental setup as before. Every 500 time steps, the radius R of the arc was modified as before in order to decrease the similarity level between tasks. The learning curves of the algorithms considered in Figure 3.7 are reported in Figure 3.10. As expected, it can be observed that the larger the number of clusters is, the more efficient the collaboration between clusters becomes. The benefits of inter-cluster cooperation decreases when the number of clusters becomes small.

3.5 Conclusion

In this chapter, we examined the behavior of the multitask diffusion LMS algorithm over asynchronous networks. This algorithm was originally proposed to solve multitask estimation problems by adding squared ℓ_2-norm co-regularizers to the MSE criterion in order to promote smoothness of the graph signal. We introduced a general model for asynchronous behavior with random step-sizes, combination coefficients, and co-regularization factors. We then carried out a convergence analysis of the asynchronous multitask algorithm in the mean and mean-square-error sense, and we derived conditions for convergence. Closed form expressions quantifying the transient and steady-state performances were also derived. Finally, we presented simulation results to show the effectiveness of the multitask strategy over asynchronous networks. We found that the mean behavior of the algorithm is immune to the effect of asynchronous events and we observed through simulations that the mean-square convergence rate in the asynchronous case is the same as in the synchronous case and that the steady-state MSD is affected by the asynchronous events.

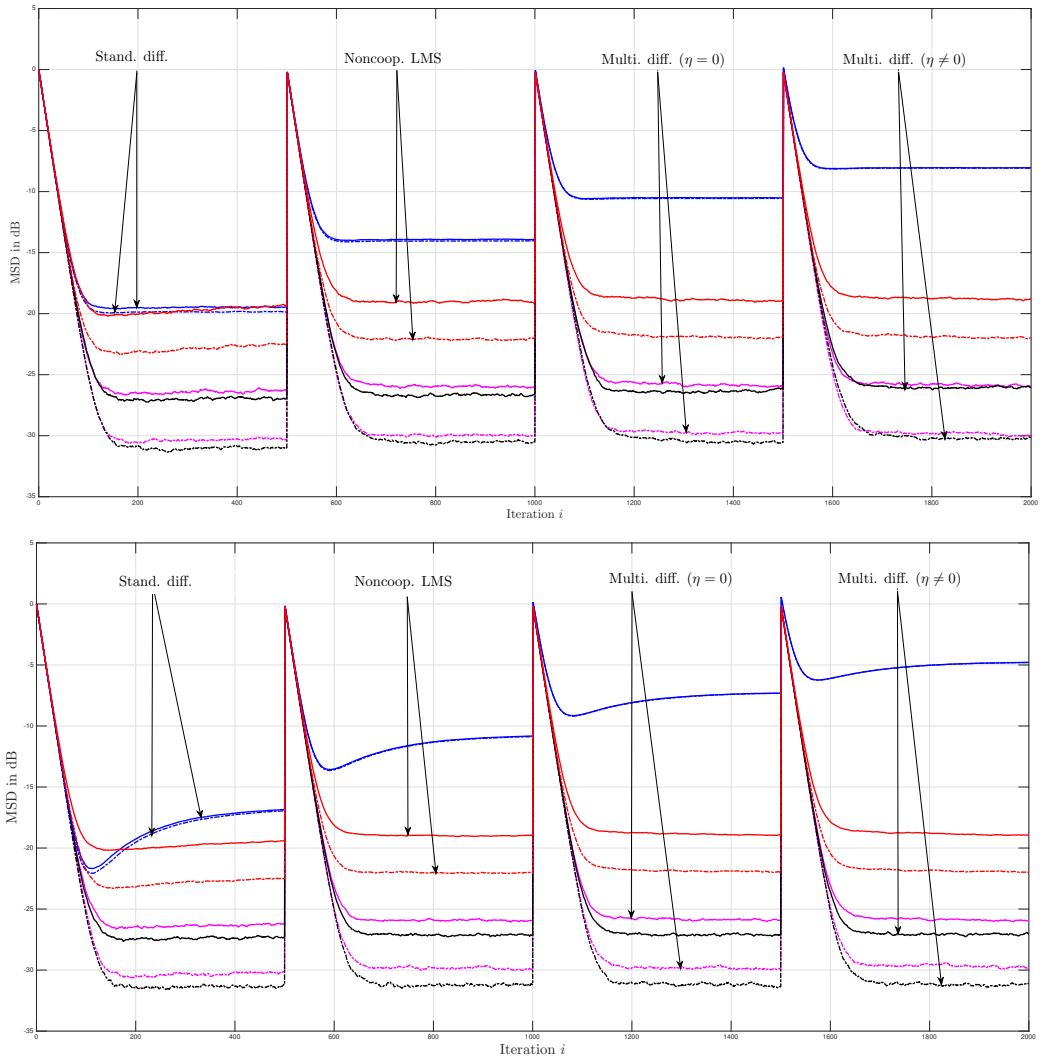


Figure 3.10: Network MSD learning curves in a non-stationary environment: comparison of the same algorithms considered in Figure 3.7. The dotted lines are for synchronous networks and the solid lines are for asynchronous networks. Top: network consisting of 5 clusters. Down: Network consisting of 15 clusters.

Several open problems still have to be solved for specific applications. For instance, it would be advantageous to consider alternative co-regularizers in order to promote other properties such as piecewise constant transitions in the entries of the parameter vectors, and to analyze the convergence of the resulting algorithms. This problem is addressed in the next chapter. Although the arguments and derivations in the next chapter can be carried out for asynchronous networks, we focus on the synchronous behavior for the sake of simplicity and in order to highlight the new contributions.

CHAPTER 4

Proximal multitask diffusion LMS over networks

As in the previous chapter, we consider multitask learning problems where clusters of agents are interested in estimating their own parameter vector. Cooperation among clusters is beneficial when the optimal models of adjacent clusters have a large number of similar entries and a relatively small number of distinct components. We propose a fully distributed algorithm for solving this problem. The approach relies on minimizing a global mean-square-error criterion regularized by non-differentiable terms to promote cooperation among neighboring clusters. A general diffusion forward-backward splitting strategy is introduced. Then, it is specialized to the case of sparsity promoting regularizers. A closed-form expression for the proximal operator of a weighted sum of ℓ_1-norms is derived to achieve higher efficiency. We also provide conditions on the step-sizes that ensure convergence of the algorithm in the mean and mean-square error sense. Finally, we propose a rule to adapt the regularization factors based on instantaneous estimates in order to guarantee an appropriate cooperation between clusters.

The work presented in this chapter was published in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion LMS with sparsity-based regularization. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3516–3520, Brisbane, Australia, April 2015. (Also available at http://www.cedric-richard.fr/Articles/nassif2015multitask.pdf)

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Proximal multitask learning over networks with sparsity inducing coregularization. *IEEE Transactions on Signal Processing*, 64(23):6329–6344, December 2016. (Also available at https://arxiv.org/abs/1509.01360)

4.1 Introduction

This chapter considers the problem of distributed adaptive estimation over multitask networks where agents are grouped into clusters, and each cluster is interested in estimating its own parameter vector (i.e., each cluster has its own task). Although clusters may generally have distinct though related tasks to perform, the agents may still be able to capitalize on inductive transfer between clusters to improve their estimation accuracy. Such situations occur when the tasks of nearby clusters are correlated, which happens, for instance, in monitoring applications where agents in a network need to track multiple targets moving along correlated trajectories.

A supervised scenario is considered where it is assumed that agents know which clusters they belong to. In this case, multitask diffusion strategies can be derived by exploiting this information on the relationships between tasks. As already stated in Subsection 1.2.2, several recent works have addressed variations of this scenario. For example, in [Abdolee et al., 2014], a diffusion LMS strategy estimates spatially-varying parameters by exploiting the spatio-temporal correlations of the measurements at neighboring nodes. In [Plata-Chaves et al., 2015], a diffusion LMS strategy was developed to perform estimation under the assumption that there are three types of parameters: parameters of global interest, parameters of common interest to a subset of agents, and parameters of local interest. In another work [Chen et al., 2014a], the parameter space is decomposed into two orthogonal subspaces, with one of the subspaces being common to all agents. Another useful way to exploit and model relationships among tasks is to formulate optimization problems with appropriate co-regularizers between agents. The strategy developed in [Chen et al., 2014c], whose convergence behavior over asynchronous networks was studied in the previous chapter, adds squared ℓ_2-norm co-regularizers to the mean-square-error criterion in order to promote smoothness of the graph signal.

In some applications, however, such as cognitive radio [Plata-Chaves et al., 2015] (considered in Subsection 4.4.2) and remote sensing [Chen et al., 2014a, Section VI-C], it may happen that the optimum parameter vectors of neighboring clusters have a large number of similar entries and a relatively small number of distinct components. It is then advantageous to develop distributed strategies that involve cooperation among adjacent clusters in order to promote and exploit such similarity. Although the current problem seems to be related to the problem studied in [Chen et al., 2014c], it should be noted that the squared ℓ_2-norm regularizers used in [Chen et al., 2014c] are not effective when sparsity promoting regularization is required. Moreover, when neighboring agents belonging to different clusters are aware of the indices of common and distinct entries, and when these indices are fixed over time, one may appeal to the multitask diffusion strategy developed in [Plata-Chaves et al., 2015, Chen et al., 2014a]. However, in the current chapter, we are interested in solutions that are able to handle situations where the only available information is that the optimum parameter vectors of neighboring clusters have a large number of similar entries and a relatively small number of distinct components. A multitask diffusion algorithm

with ℓ_1-norm co-regularizers is proposed in [Nassif et al., 2015] to address this problem leading to a subgradient descent method distributed among the agents. In this chapter, we present a more general approach for solving such convex but *non-differentiable* problems by employing instead a diffusion forward-backward splitting strategy based on the proximal projection operator. Before proceeding, we recall the forward-backward splitting approach in a single-agent deterministic environment [Combettes and Pesquet, 2011, Combettes et al., 2011, Parikh and Boyd, 2013].

Consider the problem

$$\underset{\boldsymbol{w}\in\mathbb{R}^M}{\text{minimize}}\; f(\boldsymbol{w}) + g(\boldsymbol{w}) \tag{4.1}$$

with f a real-valued differentiable convex function with β-Lipschitz continuous gradient, and g a real-valued convex function. The *proximal gradient* method or the *forward-backward splitting* approach for solving (4.1) is given by the iteration:

$$\boldsymbol{w}(i+1) = \text{prox}_{\mu g}\big(\boldsymbol{w}(i) - \mu\nabla f(\boldsymbol{w}(i))\big), \tag{4.2}$$

where μ is a constant step-size chosen such that $\mu \in (0, 2\beta^{-1}]$ to ensure convergence to the minimizer of (4.1). The gradient-descent step is the forward step (explicit step) and the proximal step is the backward step (implicit step). The proximal operator of $\mu g(\boldsymbol{w})$ at a given point $\boldsymbol{w}_o \in \mathbb{R}^M$ is a real-valued map given by [Parikh and Boyd, 2013]:

$$\text{prox}_{\mu g}(\boldsymbol{w}_o) = \underset{\boldsymbol{w}\in\mathbb{R}^M}{\text{argmin}}\; g(\boldsymbol{w}) + \frac{1}{2\mu}\|\boldsymbol{w} - \boldsymbol{w}_o\|^2. \tag{4.3}$$

Since the proximal operator needs to be calculated at each iteration in (4.2), it is important to have a closed form expression for evaluating it. In this chapter, we derive a multitask diffusion adaptation strategy where each agent employs this approach for minimizing a cost function with sparsity based co-regularizers. Instead of using iterative algorithms for evaluating the proximal operator of a weighted sum of ℓ_1-norms at each iteration [Combettes et al., 2011], we shall instead derive a closed form expression that allows us to compute it exactly. We shall also examine under which conditions on the step-sizes the proposed multitask diffusion strategy is mean and mean-square stable. An adaptive rule for setting the regularization weights is also introduced to guarantee an appropriate cooperation between clusters. Finally, simulations are conducted to show the effectiveness of the proposed strategies. The algorithm is applied to a particular application involving spectrum sensing in cognitive radio networks.

Before starting, we list in Table 4.1 some of the main symbols and notations used in this chapter. Other symbols will be defined in the context where they are used.

4.2 Multitask diffusion LMS with Forward-Backward splitting

4.2.1 Network model and problem formulation

We consider a network of N nodes grouped into Q connected clusters in a predefined topology, as illustrated in Figure 4.1. Clusters are assumed to be connected, i.e., there exists a path between

Table 4.1: List of the main symbols and notations used in Chapter 4.

M	Length of the parameter vectors
N	Number of agents in the network
Q	Number of clusters in the network
$\mathcal{N}_k$	Neighborhood of agent k, i.e., the set of agents that are connected to k by edges
$\mathcal{N}_k^-$	Neighborhood of agent k, excluding k
$\mathcal{C}_q$	Cluster q, i.e., the set of agents in the q-th cluster
$\mathcal{C}(k)$	The cluster of agents to which agent k belongs
$\mathcal{N}_k \cap \mathcal{C}(k)$	Neighbors of agent k that are inside its cluster
$\mathcal{N}_k \setminus \mathcal{C}(k)$	Neighbors of agent k that are outside its cluster
$J(\cdot),\ \overline{J}(\cdot)$	Cost function without/ with regularization
$\boldsymbol{w}_k, \boldsymbol{w}_{\mathcal{C}_q}, \boldsymbol{w}_{\mathcal{C}(k)}$	Parameter vectors at agent k, at cluster $\mathcal{C}_q$, and at cluster $\mathcal{C}(k)$
$\boldsymbol{w}_k^o, \boldsymbol{w}_{\mathcal{C}_q}^o, \boldsymbol{w}_{\mathcal{C}(k)}^o$	Optimum parameter vectors at agent k, at cluster $\mathcal{C}_q$, and at cluster $\mathcal{C}(k)$
$\boldsymbol{w}_b$	Network block parameter vector
$\boldsymbol{w}_b^o$	Network block optimum parameter vector
$\boldsymbol{\delta}_{k,\ell}$	Difference vector $\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}$

any pair of nodes in the cluster. At every time instant i, every node k has access to a zero-mean measurement $d_k(i)$ and a zero-mean $M \times 1$ regression vector $\boldsymbol{x}_k(i)$ with positive covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\,\{\boldsymbol{x}_k(i)\,\boldsymbol{x}_k^\top(i)\} > \boldsymbol{0}$. We assume the data to be related via the linear regression model:

$$d_k(i) = \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k^o + z_k(i), \qquad i \geq 0, \tag{4.4}$$

where $\boldsymbol{w}_k^o$ is the $M \times 1$ unknown parameter vector, also called task, we wish to estimate at node k, and $z_k(i)$ is a zero-mean measurement noise of variance $\sigma_{z,k}^2$, independent of $\boldsymbol{x}_\ell(j)$ for all ℓ and j, and independent of $z_\ell(j)$ for $\ell \neq k$ or $i \neq j$. We assume that all nodes in a cluster are interested in estimating the same parameter vector, namely, $\boldsymbol{w}_k^o = \boldsymbol{w}_{\mathcal{C}_q}^o$ whenever k belongs to cluster $\mathcal{C}_q$. However, if cluster $\mathcal{C}_p$ is connected to cluster $\mathcal{C}_q$, that is, there exists at least one link connecting a node from $\mathcal{C}_p$ to a node from $\mathcal{C}_q$, vectors $\boldsymbol{w}_{\mathcal{C}_p}^o$ and $\boldsymbol{w}_{\mathcal{C}_q}^o$ are assumed to have a large number of similar entries and only a relatively small number of distinct entries. Cooperation across these clusters can therefore be beneficial to infer $\boldsymbol{w}_{\mathcal{C}_p}^o$ and $\boldsymbol{w}_{\mathcal{C}_q}^o$.

Considerable interest has been shown in the literature about estimating an optimum parameter vector $\boldsymbol{w}^o$ subject to the property of being sparse. Motivated by the well-known least absolute shrinkage and selection operator (LASSO) problem [Tibshirani, 1996] and compressed sensing framework [Baraniuk, 2007], different techniques for sparse adaptation have been proposed. For example, the authors in [Chen et al., 2009b, Gu et al., 2009] promote sparsity within an LMS framework by considering regularizers based on the ℓ_1-norm, reweighed ℓ_1-norm, and

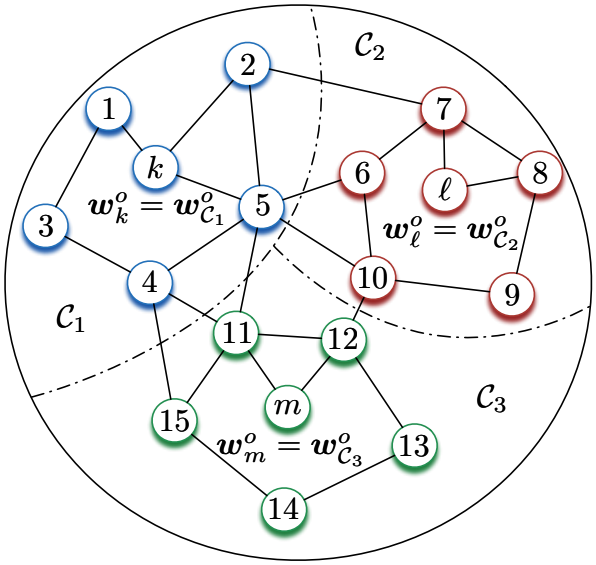


Figure 4.1: Multitask clustered network consisting of 3 clusters. Two clusters are connected if they share at least one edge.

convex approximation of ℓ_0-norm. In [Kopsinis et al., 2011], projections of streaming data onto hyperslabs and weighted ℓ_1-balls are used instead of minimizing regularized costs recursively. Proximal forward-backward splitting is considered in an adaptive scenario in [Murakami et al., 2010]. In the context of distributed learning over *single-task* networks, diffusion LMS methods promoting sparsity have been proposed. Sparse diffusion LMS strategies using subgradient methods are proposed in [Lorenzo et al., 2012, Lorenzo and Sayed, 2013, Liu et al., 2012] and using proximal methods are proposed in [Wee and Yamada, 2013, Lorenzo, 2014, Vlaski and Sayed, 2015]. In [Chouvardas et al., 2012], the authors employ projection-based techniques [Kopsinis et al., 2011] to derive distributed diffusion algorithms promoting sparsity, and in [Chouvardas et al., 2013] a diffusion LMS algorithm for estimating an s-sparse vector is proposed based on adaptive greedy techniques similar to [Mileounis et al., 2010]. These techniques estimate the positions of non-zero entries in the target vector, and then perform computations on this subset. More generally, diffusion strategies based on proximal gradient for minimizing general costs (not necessarily mean-square error costs) and subject to a broader class of constraints on the parameter vector to be estimated (including sparsity) are derived in [Vlaski and Sayed, 2015].

Our purpose in this chapter is to derive an adaptive learning algorithm over *multitask* networks where optimum parameter vectors of neighboring clusters share a large number of similar entries and a relatively small number of distinct entries. Consider nodes k and ℓ of neighboring clusters $\mathcal{C}(k)$ and $\mathcal{C}(\ell)$, and let $\boldsymbol{\delta}_{k,\ell}$ denote the vector difference $\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}$. Promoting the sparsity of $\boldsymbol{\delta}_{k,\ell}$ can be performed by considering the pseudo ℓ_0-norm of $\boldsymbol{\delta}_{k,\ell}$ as it denotes the number of nonzero entries. Nevertheless, $\|\boldsymbol{\delta}_{k,\ell}\|_0$ is a non-convex co-regularizer that leads to computational challenges. A common alternative is to use the ℓ_1-norm regularization function

defined as

$$f_1(\boldsymbol{\delta}_{k,\ell}) = \|\boldsymbol{\delta}_{k,\ell}\|_1 = \sum_{m=1}^{M} \big|[\boldsymbol{\delta}_{k,\ell}]_m\big|. \tag{4.5}$$

Since the ℓ_1-norm uniformly shrinks all the components of a vector and does not distinguish between zero and non-zero entries [Candes et al., 2008], it is common in the sparse adaptive filtering framework [Chen et al., 2009b, Kopsinis et al., 2011, Murakami et al., 2010, Lorenzo et al., 2012, Lorenzo and Sayed, 2013, Wee and Yamada, 2013, Lorenzo, 2014, Chouvardas et al., 2012, Gao et al., 2013] to consider a weighted formulation of the ℓ_1-norm. Weighted ℓ_1-norm was designed to reduce the bias induced by the ℓ_1-norm and enhance the penalization of the non-zero entries of a vector [Candes et al., 2008, Kopsinis et al., 2011, Zou, 2006]. Given the weight vector $\boldsymbol{\alpha}_{k\ell} = [\alpha_{k\ell}^1, \ldots, \alpha_{k\ell}^M]^\top$, with $\alpha_{k\ell}^m > 0$ for all m, the weighted ℓ_1-norm is defined as:

$$f_2(\boldsymbol{\delta}_{k,\ell}) = \sum_{m=1}^{M} \alpha_{k\ell}^m \big|[\boldsymbol{\delta}_{k,\ell}]_m\big|. \tag{4.6}$$

The weights are usually chosen as:

$$\alpha_{k\ell}^m = \frac{1}{\epsilon + \big|[\boldsymbol{\delta}_{k,\ell}^o]_m\big|}, \qquad m = 1, \ldots, M, \tag{4.7}$$

where $\boldsymbol{\delta}_{k,\ell}^o \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_\ell^o$ and where ϵ is a small constant to prevent the denominator from vanishing. Since the optimum parameter vectors are not available beforehand, we set

$$\alpha_{k\ell}^m(i) = \frac{1}{\epsilon + \big|[\boldsymbol{\delta}_{k,\ell}(i-1)]_m\big|}, \qquad m = 1, \ldots, M, \tag{4.8}$$

at each iteration i, where $\boldsymbol{\delta}_{k,\ell}(i)$ is the estimate of $\boldsymbol{\delta}_{k,\ell}^o$ at nodes k and ℓ and iteration i. This technique, also known as reweighted ℓ_1 minimization [Candes et al., 2008], is performed at each iteration of the stochastic optimization process. It has been shown in [Candes et al., 2008] that, by minimizing (4.6) with the weights (4.8), one minimizes the log-sum penalty function, $\sum_{m=1}^{M} \log\big(\epsilon + \big|[\boldsymbol{\delta}_{k,\ell}]_m\big|\big)$, which acts like the ℓ_0 pseudo norm by allowing a relatively large penalty to be placed on small nonzero coefficients and more strongly encourages them to be set to zero. In the sequel, we shall use $f\big(\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big)$ to refer to the unweighted or reweighted ℓ_1-norm promoting the sparsity of $\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}$.

Although most of the derivations presented in this chapter can be extended to more general cost functions, we shall focuses on the mean-square-error criterion. That is, we shall assume that the local cost function $J_k(\boldsymbol{w}_{\mathcal{C}(k)})$ at node k is given by:

$$J_k(\boldsymbol{w}_{\mathcal{C}(k)}) = \mathbb{E}\,\big(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{\mathcal{C}(k)}\big)^2. \tag{4.9}$$

Combining local mean-square-error cost functions and regularization functions, the cooperative multitask estimation problem is formulated as the problem of seeking a fully distributed solution for solving:

$$\underset{\boldsymbol{w}_{\mathcal{C}_1},\ldots,\boldsymbol{w}_{\mathcal{C}_Q}}{\text{minimize}}\ \overline{J}^{\,\text{glob}}\big(\boldsymbol{w}_{\mathcal{C}_1}, \ldots, \boldsymbol{w}_{\mathcal{C}_Q}\big) \triangleq \sum_{k=1}^{N} J_k\big(\boldsymbol{w}_{\mathcal{C}(k)}\big) + \eta \sum_{k=1}^{N} \sum_{\ell \in \mathcal{N}_k \backslash \mathcal{C}(k)} \rho_{k\ell}\, f\big(\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big), \tag{4.10}$$

where $\eta > 0$ is the regularization strength used to enforce sparsity. It ensures a trade-off between fidelity to the measurements and prior information on the relationships between tasks. The weights $\rho_{k\ell} \geq 0$ aim at locally adjusting the regularization strength. We recall that the notation $\mathcal{N}_k \setminus \mathcal{C}(k)$ denotes the set of neighboring nodes of k that are outside its cluster.

Note that the regularization terms (4.5) and (4.6) are symmetric with respect to the weight vectors $\boldsymbol{w}_{\mathcal{C}(k)}$ and $\boldsymbol{w}_{\mathcal{C}(\ell)}$, that is, $f\big(\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big) = f\big(\boldsymbol{w}_{\mathcal{C}(\ell)} - \boldsymbol{w}_{\mathcal{C}(k)}\big)$. Due to the summation over the N nodes, each term $f\big(\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big)$ can be viewed as weighted by $\frac{(\rho_{k\ell}+\rho_{\ell k})}{2}$ in (4.10). Problem (4.10) can therefore be written in an alternative way as:

$$\underset{\boldsymbol{w}_{\mathcal{C}_1},\ldots,\boldsymbol{w}_{\mathcal{C}_Q}}{\text{minimize}}\ \overline{J}^{\,\text{glob}}\big(\boldsymbol{w}_{\mathcal{C}_1},\ldots,\boldsymbol{w}_{\mathcal{C}_Q}\big) = \sum_{k=1}^{N} J_k(\boldsymbol{w}_{\mathcal{C}(k)}) + \eta \sum_{k=1}^{N} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)} p_{k\ell} f(\boldsymbol{w}_{\mathcal{C}(k)} - \boldsymbol{w}_{\mathcal{C}(\ell)}) \tag{4.11}$$

where the factors $\{p_{k\ell}\}$ are symmetric, i.e., $p_{k\ell} = p_{\ell k}$, and are given by:

$$p_{k\ell} \triangleq \frac{(\rho_{k\ell} + \rho_{\ell k})}{2}. \tag{4.12}$$

One way to avoid symmetrical regularization is to consider an alternative problem formulation defined in terms of Q Nash equilibrium problems as done in [Chen et al., 2014c] with ℓ_2-norm co-regularizers (considered in the previous chapter). In this chapter, we shall focus on problem (4.10).

Let us consider the variable $\boldsymbol{w}_{\mathcal{C}_q}$ of the q-th cluster. Given $\boldsymbol{w}_{\mathcal{C}(\ell)}$ with $\ell \in \mathcal{N}_k \setminus \mathcal{C}_q$ and $k \in \mathcal{C}_q$, the subdifferential of $\overline{J}^{\,\text{glob}}\big(\boldsymbol{w}_{\mathcal{C}_1},\ldots,\boldsymbol{w}_{\mathcal{C}_Q}\big)$ in (4.11) with respect to $\boldsymbol{w}_{\mathcal{C}_q}$ is given by:

$$\begin{aligned}
\partial_{\boldsymbol{w}_{\mathcal{C}_q}} \overline{J}^{\,\text{glob}}\big(\boldsymbol{w}_{\mathcal{C}_1},\ldots,\boldsymbol{w}_{\mathcal{C}_Q}\big) &= \sum_{k \in \mathcal{C}_q} \nabla_{\boldsymbol{w}_{\mathcal{C}_q}} J_k(\boldsymbol{w}_{\mathcal{C}_q}) + \\
&\quad \eta \sum_{k \in \mathcal{C}_q} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}_q} \Big[p_{k\ell} \partial_{\boldsymbol{w}_{\mathcal{C}_q}} f\big(\boldsymbol{w}_{\mathcal{C}_q} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big) + p_{\ell k} \partial_{\boldsymbol{w}_{\mathcal{C}_q}} f\big(\boldsymbol{w}_{\mathcal{C}(\ell)} - \boldsymbol{w}_{\mathcal{C}_q}\big) \Big] \\
&= \sum_{k \in \mathcal{C}_q} \nabla_{\boldsymbol{w}_{\mathcal{C}_q}} J_k(\boldsymbol{w}_{\mathcal{C}_q}) + 2\,\eta \sum_{k \in \mathcal{C}_j} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}_q} p_{k\ell} \partial_{\boldsymbol{w}_{\mathcal{C}_q}} f\big(\boldsymbol{w}_{\mathcal{C}_q} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big),
\end{aligned} \tag{4.13}$$

where we used the fact that the regularization terms (4.5), (4.6), and the regularization factors $\{p_{k\ell}\}$ are symmetric. Since we are interested in a distributed strategy for solving (4.10) that relies only on in-network processing, we associate the following regularized problem with each cluster $\mathcal{C}_q$:

$$\underset{\boldsymbol{w}_{\mathcal{C}_q}}{\text{minimize}}\ \overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q}) \triangleq \sum_{k \in \mathcal{C}_q} \mathbb{E}\,\big(d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_{\mathcal{C}_q}\big)^2 + 2\,\eta \sum_{k \in \mathcal{C}_q} \sum_{\ell \in \mathcal{N}_k \setminus \mathcal{C}_q} p_{k\ell} f\big(\boldsymbol{w}_{\mathcal{C}_q} - \boldsymbol{w}_{\mathcal{C}(\ell)}\big). \tag{4.14}$$

Given $\boldsymbol{w}_{\mathcal{C}(\ell)}$ with $\ell \in \mathcal{N}_k \setminus \mathcal{C}_q$, note that the costs in problems (4.10) and (4.14) have the same subdifferential relative to $\boldsymbol{w}_{\mathcal{C}_q}$. In order that each node can solve the problem in an autonomous and adaptive manner using only local interactions, we shall derive a distributed iterative algorithm for solving (4.10) by considering (4.14) since both costs have the same subdifferential information.

4.2.2 Problem relaxation

We shall now extend the derivations in [Cattivelli and Sayed, 2010, Sayed, 2014c, Chen et al., 2014b] to handle multitask estimation problems with nondifferentiable functions. In the sequel, we write $\boldsymbol{w}_k$ instead of $\boldsymbol{w}_{\mathcal{C}(k)}$ for simplicity of notation. First, we associate with each node k an unregularized local cost function $J_k^{\rm loc}(\cdot)$ and a regularized local cost function $\overline{J}_k^{\rm loc}(\cdot)$ of the form:

$$J_k^{\rm loc}(\boldsymbol{w}_k) \triangleq \sum_{\ell\in\mathcal{N}_k\cap\mathcal{C}(k)} c_{\ell k}\mathbb{E}\left(d_\ell(i)-\boldsymbol{x}_\ell^{\top}(i)\boldsymbol{w}_k\right)^2, \tag{4.15}$$

$$\overline{J}_k^{\rm loc}(\boldsymbol{w}_k) \triangleq \sum_{\ell\in\mathcal{N}_k\cap\mathcal{C}(k)} c_{\ell k}\mathbb{E}\left(d_\ell(i)-\boldsymbol{x}_\ell^{\top}(i)\boldsymbol{w}_k\right)^2 + 2\,\eta\sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell}f\big(\boldsymbol{w}_k-\boldsymbol{w}_\ell\big), \tag{4.16}$$

where the set of nodes in the neighborhood of node k that belongs to its cluster is denoted by $\mathcal{N}_k\cap\mathcal{C}(k)$, and $\{c_{\ell k}\}$ are non-negative weights satisfying

$$\sum_{k=1}^{N} c_{\ell k} = 1, \quad \text{and} \quad c_{\ell k} = 0 \quad \text{if} \quad k\notin\mathcal{N}_\ell\cap\mathcal{C}(\ell). \tag{4.17}$$

Note that $\boldsymbol{w}_k=\boldsymbol{w}_\ell$ whenever $\ell\in\mathcal{N}_k\cap\mathcal{C}(k)$. Both costs (4.15) and (4.16) consist of a combination of mean-square error costs in the neighborhood of node k but limited to its cluster. In addition, expression (4.16) takes interactions among neighboring clusters into account. Let us consider node k belonging to cluster $\mathcal{C}_q$, i.e., $\mathcal{C}_q=\mathcal{C}(k)$. It can be checked that $\overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q})$ in (4.14) can be written as:

$$\overline{J}_{\mathcal{C}_q}(\boldsymbol{w}_{\mathcal{C}_q}) = \sum_{\ell\in\mathcal{C}_q}\overline{J}_\ell^{\rm loc}(\boldsymbol{w}_\ell) = \overline{J}_k^{\rm loc}(\boldsymbol{w}_k) + \sum_{\ell\in\mathcal{C}_q\setminus\{k\}}\overline{J}_\ell^{\rm loc}(\boldsymbol{w}_\ell), \tag{4.18}$$

The term $\sum_{\ell\in\mathcal{C}_q\setminus\{k\}}\overline{J}_\ell^{\rm loc}(\boldsymbol{w}_\ell)$ contains terms promoting relationships between nodes $\ell\in\mathcal{C}_q\setminus\{k\}$ and their neighbors that are outside $\mathcal{C}_q$ but not necessarily in the neighborhood of node k. To limit these inter-cluster information exchanges to node k and its extra-cluster neighbors, we relax $\sum_{\ell\in\mathcal{C}_q\setminus\{k\}}\overline{J}_\ell^{\rm loc}(\boldsymbol{w}_\ell)$ to $\sum_{\ell\in\mathcal{C}_q\setminus\{k\}}J_\ell^{\rm loc}(\boldsymbol{w}_\ell)$. Since (4.15) is second-order differentiable, a completion-of-squares argument, or equivalently, a second order Taylor expansion, shows that each $J_\ell^{\rm loc}(\boldsymbol{w}_\ell)$ can be expressed as [Sayed, 2014c]:

$$J_\ell^{\rm loc}(\boldsymbol{w}_\ell) = J_\ell^{\rm loc}\big(\boldsymbol{w}_\ell^{\rm loc}\big) + \|\boldsymbol{w}_\ell-\boldsymbol{w}_\ell^{\rm loc}\|^2_{\boldsymbol{R}_\ell}, \tag{4.19}$$

where the notation $\|\boldsymbol{x}\|^2_{\boldsymbol{\Sigma}}$ denotes $\boldsymbol{x}^{\top}\boldsymbol{\Sigma}\,\boldsymbol{x}$ for any nonnegative definite matrix $\boldsymbol{\Sigma}$, $\boldsymbol{w}_\ell^{\rm loc}$ is the minimizer of $J_\ell^{\rm loc}(\boldsymbol{w}_\ell)$ (which coincides with $\boldsymbol{w}_\ell^o$), and $\boldsymbol{R}_\ell$ is given by:

$$\boldsymbol{R}_\ell = \sum_{k\in\mathcal{N}_\ell\cap\mathcal{C}(\ell)} c_{k\ell}\boldsymbol{R}_{x,k}. \tag{4.20}$$

Thus, using (4.16), (4.18), and (4.19) and dropping the constant term $J_\ell^{\rm loc}\big(\boldsymbol{w}_\ell^{\rm loc}\big)$, we can replace the original cluster cost (4.14) by the following cost function for cluster $\mathcal{C}(k)$ at node k:

$$\begin{aligned}\overline{J}'_{\mathcal{C}(k)}(\boldsymbol{w}_k) = &\sum_{\ell\in\mathcal{N}_k\cap\mathcal{C}(k)} c_{\ell k}\mathbb{E}\left(d_\ell(i)-\boldsymbol{x}_\ell^{\top}(i)\boldsymbol{w}_k\right)^2 + 2\,\eta\sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell}f(\boldsymbol{w}_k-\boldsymbol{w}_\ell)\\ &+\sum_{\ell\in\mathcal{C}(k)\setminus\{k\}}\|\boldsymbol{w}_\ell-\boldsymbol{w}_\ell^{\rm loc}\|^2_{\boldsymbol{R}_\ell}.\end{aligned} \tag{4.21}$$

Equation (4.21) is an approximation relating the local cost function $\overline{J}_k^{\,\mathrm{loc}}(\boldsymbol{w}_k)$ at node k to the global cost function (4.14) associated with the cluster $\mathcal{C}(k)$. Node k cannot minimize (4.21) directly since this cost still requires global information that may not be available in its neighborhood. To avoid access to information via multihop, we relax $\overline{J}'_{\mathcal{C}(k)}(\boldsymbol{w}_k)$ by limiting the sum in the third term on the RHS of (4.21) over the neighbors of node k. In addition, since the covariance matrices $\boldsymbol{R}_{x,\ell}$ may not be known beforehand within the context of online learning, a useful strategy proposed in [Sayed, 2014c] is to substitute the covariance matrices $\boldsymbol{R}_\ell$ by diagonal matrices of the form $b_{\ell k}\boldsymbol{I}_M$[1], where $b_{\ell k}$ are nonnegative coefficients that allow to assign different weights to different neighbors. Later, these coefficients will be incorporated into a left-stochastic matrix and the designer does not need to worry about their selection. Based on the arguments presented so far, the cluster cost function at each node k can be relaxed as follows:

$$\begin{aligned}\overline{J}''_{\mathcal{C}(k)}(\boldsymbol{w}_k) = \sum_{\ell\in\mathcal{N}_k\cap\mathcal{C}(k)} c_{\ell k}\mathbb{E}\left(d_\ell(i)-\boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k\right)^2 + 2\eta \sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell} f(\boldsymbol{w}_k-\boldsymbol{w}_\ell) \\ + \sum_{\ell\in\mathcal{N}_k^-\cap\mathcal{C}(k)} b_{\ell k}\|\boldsymbol{w}_k-\boldsymbol{w}_\ell^{\mathrm{loc}}\|^2.\end{aligned} \tag{4.22}$$

Since this cost function only relies on data available in the neighborhood of each node k, we can now proceed to derive distributed strategies.

The first and third terms on the RHS of (4.22) are second-order differentiable and strictly convex. The second term is convex but not continuously differentiable. Among other possible techniques [Nassif et al., 2015], we illustrate in this chapter how to obtain a multitask adapt-then-combine (ATC) algorithm for solving the convex minimization problem (4.22) using a forward-backward splitting approach.

4.2.3 Multitask diffusion with forward-backward splitting approach

Let $\boldsymbol{w}_k(i)$ denote the estimate of $\boldsymbol{w}_k^o$ at node k and iteration i. Considering a forward-backward splitting strategy for solving (4.22), we have:

$$\boldsymbol{w}_k(i+1) = \mathrm{prox}_{2\eta\nu_k\tilde{g}_{k,i}}\Big(\boldsymbol{w}_k(i) - \nu_k\nabla_{\boldsymbol{w}_k}J''_{\mathcal{C}(k)}\big(\boldsymbol{w}_k(i)\big)\Big), \tag{4.23}$$

with ν_k a positive step-size parameter,

$$\tilde{g}_{k,i}(\boldsymbol{w}_k) \triangleq \sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell} f\big(\boldsymbol{w}_k-\boldsymbol{w}_\ell(i)\big), \tag{4.24}$$

and $J''_{\mathcal{C}(k)}(\boldsymbol{w}_k)$ denoting the unregularized part of $\overline{J}''_{\mathcal{C}(k)}(\boldsymbol{w}_k)$ limited to the first and third terms on the RHS of (4.22). Let

$$\boldsymbol{\phi}_k(i+1) \triangleq \boldsymbol{w}_k(i) - \nu_k\nabla_{\boldsymbol{w}_k}J''_{\mathcal{C}(k)}(\boldsymbol{w}_k(i)). \tag{4.25}$$

[1]The approximation follows from the Rayleigh-Ritz characterization of eigenvalues where, using the fact that the matrices $\boldsymbol{R}_\ell$ are positive definite, we obtain $\lambda_{\min}(\boldsymbol{R}_\ell)\boldsymbol{I}_M \le \|\boldsymbol{w}_\ell-\boldsymbol{w}_\ell^{\mathrm{loc}}\|^2_{\boldsymbol{R}_\ell} \le \lambda_{\max}(\boldsymbol{R}_\ell)\boldsymbol{I}_M$.

Using (4.22), we arrive at:

$$\boldsymbol{\phi}_k(i+1) = \boldsymbol{w}_k(i) + 2\,\nu_k \left(\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} (\boldsymbol{r}_{dx,\ell} - \boldsymbol{R}_{x,\ell} \boldsymbol{w}_k(i)) \right) + 2\,\nu_k \sum_{\ell \in \mathcal{N}_k^- \cap \mathcal{C}(k)} b_{\ell k} \left(\boldsymbol{w}_\ell^{\rm loc} - \boldsymbol{w}_k(i) \right). \tag{4.26}$$

where $\boldsymbol{r}_{dx,k} \triangleq \mathbb{E}\,\{d_k(i)\boldsymbol{x}_k(i)\}$. The update in the previous equation can be implemented in two successive steps as follows [Sayed, 2014c]:

$$\begin{aligned} \boldsymbol{\psi}_k(i+1) &= \boldsymbol{w}_k(i) + 2\,\nu_k \left(\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} (\boldsymbol{r}_{dx,\ell} - \boldsymbol{R}_{x,\ell} \boldsymbol{w}_k(i)) \right) \\ \boldsymbol{\phi}_k(i+1) &= \boldsymbol{\psi}_k(i+1) + 2\,\nu_k \sum_{\ell \in \mathcal{N}_k^- \cap \mathcal{C}(k)} b_{\ell k} \left(\boldsymbol{w}_\ell^{\rm loc} - \boldsymbol{w}_k(i) \right). \end{aligned} \tag{4.27}$$

Since $\boldsymbol{w}_\ell^{\rm loc}$, or equivalently $\boldsymbol{w}_\ell^o$, is not available at node ℓ and node ℓ is trying to estimate it, we replace it by the available approximation at node ℓ, which is the intermediate estimate $\boldsymbol{\psi}_\ell(i+1)$. Furthermore, since $\boldsymbol{\psi}_k(i+1)$ at node k is a better estimate for $\boldsymbol{w}_k^o$ than $\boldsymbol{w}_k(i)$, we replace $\boldsymbol{w}_k(i)$ by $\boldsymbol{\psi}_k(i+1)$ in the second step of (4.27). With these substitutions, the second step in (4.27) can be written as:

$$\boldsymbol{\phi}_k(i+1) = \sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k} \boldsymbol{\psi}_\ell(i+1), \tag{4.28}$$

where we introduced the coefficients $\{a_{\ell k}\}$ given by:

$$a_{kk} \triangleq 1 - 2\,\nu_k \sum_{\ell \in \mathcal{N}_k^- \cap \mathcal{C}(k)} b_{\ell k}, \qquad a_{\ell k} \triangleq 2\,\nu_k b_{\ell k},\ \text{if}\ \ell \in \mathcal{N}_k^- \cap \mathcal{C}(k), \qquad a_{\ell k} \triangleq 0,\ \text{if}\ \ell \notin \mathcal{N}_k \cap \mathcal{C}(k). \tag{4.29}$$

For sufficiently small step-sizes, these coefficients are non-negative and satisfy:

$$\sum_{\ell=1}^{N} a_{\ell k} = 1, \quad \text{and} \quad a_{\ell k} = 0 \quad \text{if} \quad \ell \notin \mathcal{N}_k \cap \mathcal{C}(k). \tag{4.30}$$

Since the moments $\{\boldsymbol{r}_{dx,k}, \boldsymbol{R}_{x,k}\}$ are rarely available beforehand, we use the following instantaneous approximations:

$$\boldsymbol{R}_{x,k} \approx \boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i), \qquad \boldsymbol{r}_{dx,k} \approx d_k(i)\boldsymbol{x}_k(i). \tag{4.31}$$

Thus, we arrive at the following adapt-then-combine (ATC) diffusion strategy with forward-backward splitting for solving problem (4.10) in a fully distributed adaptive manner:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \sum\limits_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} c_{\ell k}\, \boldsymbol{x}_\ell(i) \big[d_\ell(i) - \boldsymbol{x}_\ell^\top(i) \boldsymbol{w}_k(i) \big], \\ \boldsymbol{\phi}_k(i+1) = \sum\limits_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k} \boldsymbol{\psi}_\ell(i+1), \\ \boldsymbol{w}_k(i+1) = \mathrm{prox}_{\eta \mu_k g_{k,i+1}}(\boldsymbol{\phi}_k(i+1)), \end{cases} \tag{4.32}$$

where $\mu_k = 2\nu_k$ is introduced to avoid an extra factor of 2 multiplying ν_k and coming from evaluating the gradient of squared quantities in $J''_{\mathcal{C}(k)}(\boldsymbol{w}_k)$ and

$$g_{k,i+1}(\boldsymbol{w}_k) \triangleq \sum_{\ell \in \mathcal{N}_k \backslash \mathcal{C}(k)} p_{k\ell} f\big(\boldsymbol{w}_k - \boldsymbol{\phi}_\ell(i+1)\big). \tag{4.33}$$

Functions $\tilde{g}_{k,i}(\cdot)$ in (4.24) and $g_{k,i+1}(\cdot)$ in (4.33) are iteration dependent through $\boldsymbol{w}_\ell(i)$ and $\boldsymbol{\phi}_\ell(i+1)$. Note that we have substituted $\boldsymbol{w}_\ell(i)$ in (4.24) by $\boldsymbol{\phi}_\ell(i+1)$ in (4.33) since $\boldsymbol{\phi}_\ell(i+1)$ is an updated estimate of $\boldsymbol{w}_\ell(i)$ at node ℓ. The proximal operator of $\eta\mu_k g_{k,i+1}(\cdot)$ in the third step of (4.32) needs to be evaluated at each iteration $i+1$ and for all nodes k in the network. A closed-form expression is recommended to achieve higher computational efficiency. We shall derive such closed-form expression when f in (4.33) is selected either as the ℓ_1-norm or the reweighted ℓ_1-norm — see Section 4.2.4 for details.

The multitask diffusion LMS (4.32) with forward-backward splitting starts with an initial estimate $\boldsymbol{w}_k(0)$ for all k, and repeats (4.32) at each instant $i \geq 0$ and for all k. In the first step of (4.32), which corresponds to the adaptation step, node k receives from its intra-cluster neighbors their raw data $\{d_\ell(i), \boldsymbol{x}_\ell(i)\}$, combines this information through the coefficients $\{c_{\ell k}\}$, and uses it to update its estimate $\boldsymbol{w}_k(i)$ to an intermediate estimate $\boldsymbol{\psi}_k(i+1)$. The second step in (4.32) is a combination step where node k receives the intermediate estimates $\{\boldsymbol{\psi}_\ell(i+1)\}$ from its intra-cluster neighbors and combines them through the coefficients $\{a_{\ell k}\}$ to obtain the intermediate value $\boldsymbol{\phi}_k(i+1)$. Finally, in the third step in (4.32), node k receives the intermediate estimates $\{\boldsymbol{\phi}_\ell(i+1)\}$ from its neighbors that are outside its cluster and evaluates the proximal operator of the function in (4.33) at $\boldsymbol{\phi}_k(i+1)$ to obtain $\boldsymbol{w}_k(i+1)$.

To run the algorithm, each node k only needs to know the step-size μ_k, the regularization strength η, the regularization weights $\{p_{k\ell}\}_{\ell \in \mathcal{N}_k \backslash \mathcal{C}(k)}$, and the coefficients $\{a_{\ell k}, c_{\ell k}\}_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)}$ satisfying conditions (4.17) and (4.30). The scalars $\{a_{\ell k}, c_{\ell k}\}$ and $\{\rho_{k\ell}\}$ correspond to weighting coefficients over the edges linking node k to its neighbors ℓ according to whether these neighbors lie inside or outside its cluster. There are several ways to select these coefficients [Sayed, 2014a,b,c, Chen et al., 2014c]. In Section 4.4, we propose an adaptive rule for selecting each regularization weight $p_{k\ell}$ based on a measure of the sparsity level of $\boldsymbol{w}_k^o - \boldsymbol{w}_\ell^o$ at node k. Finally, note that alternative implementations of (4.32) may be considered. In particular, the adaptation step can be followed by the proximal step, before or after aggregation as in the possible Adapt-then-Combine and Combine-then-Adapt diffusion strategies.

Algorithm (4.32) may be applied to multitask problems involving any type of coregularizers $f(\cdot)$ provided that the proximal operator of a weighted sum of these regularizers can be assessed in closed form. In the next section, we shall focus on the particular case of sparsity promoting regularizers.

4.2.4 Proximal operator of weighted sum of ℓ_1-norms

We shall now derive a closed form expression for the proximal operator of the convex function $g_{k,i+1}(\boldsymbol{w}_k)$ in (4.33). Considering both regularizations addressed in this work, that is, the ℓ_1-norm (4.5) and the reweighted ℓ_1-norm (4.6), we write:

$$\begin{aligned} g_{k,i+1}(\boldsymbol{w}_k) &= \sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell} \sum_{m=1}^{M} \alpha_{k\ell}^{m}(i)\big|[\boldsymbol{w}_k]_m - [\boldsymbol{\phi}_\ell(i+1)]_m\big| \\ &= \sum_{m=1}^{M} \Phi_{k,m,i+1}\big([\boldsymbol{w}_k]_m\big), \end{aligned} \tag{4.34}$$

where $\Phi_{k,m,i+1}([\boldsymbol{w}_k]_m)$ is the iteration-dependent function given by:

$$\Phi_{k,m,i+1}\big([\boldsymbol{w}_k]_m\big) \triangleq \sum_{\ell\in\mathcal{N}_k\setminus\mathcal{C}(k)} p_{k\ell}\,\alpha_{k\ell}^{m}(i)\big|[\boldsymbol{w}_k]_m - [\boldsymbol{\phi}_\ell(i+1)]_m\big|. \tag{4.35}$$

Since $g_{k,i+1}(\boldsymbol{w}_k)$ is fully separable, its proximal operator can be evaluated component-wise [Parikh and Boyd, 2013]:

$$\Big[\mathrm{prox}_{\eta\mu_k g_{k,i+1}}\big(\boldsymbol{\phi}_k(i+1)\big)\Big]_m = \mathrm{prox}_{\eta\mu_k\Phi_{k,m,i+1}}\Big(\big[\boldsymbol{\phi}_k(i+1)\big]_m\Big), \quad \forall m = 1,\ldots,M. \tag{4.36}$$

For clarity of presentation, we shall now derive the proximal operator of a function $h(\cdot)$ similar to $\Phi_{k,m,i+1}(\cdot)$. Next, we shall establish the closed-form expression for $\mathrm{prox}_{\eta\mu_k\Phi_{k,m,i+1}}(\cdot)$ by identification.

Let $h:\mathbb{R}\to\mathbb{R}$ be a combination of absolute value functions defined as:

$$h(x) \triangleq \sum_{j=1}^{J} c_j\, h_j(x) = \sum_{j=1}^{J} c_j\,|x - b_j|, \tag{4.37}$$

with $c_j > 0$ for all j and $b_1 < b_2 < \ldots < b_J$. Note that this ordering is assumed for convenience of derivation and does not affect the final result. Iterative algorithms have been proposed in the literature for evaluating the proximal operator of sums of composite functions [Combettes and Pesquet, 2011, Combettes et al., 2011]. We are, however, able to derive a closed-form expression for (4.37) as detailed in the sequel. From the optimality condition for (4.3), namely that zero belongs to the subgradient set at the minimizer $\mathrm{prox}_{\lambda h}(v)$, we have,

$$0 \in \partial h\big(\mathrm{prox}_{\lambda h}(v)\big) + \frac{1}{\lambda}\big(\mathrm{prox}_{\lambda h}(v) - v\big) \Rightarrow v - \mathrm{prox}_{\lambda h}(v) \in \lambda\partial h\big(\mathrm{prox}_{\lambda h}(v)\big). \tag{4.38}$$

Since $x\in\mathbb{R}$ and c_j are non-negative, we have [Polyak, 1987, Chapter 5: Lemma 10]:

$$\partial\left(\sum_{j=1}^{J} c_j h_j(x)\right) = \sum_{j=1}^{J} c_j\partial h_j(x) = \sum_{j=1}^{J} c_j\partial|x - b_j|. \tag{4.39}$$

Hence, the subdifferential of the real valued convex function $h(x)$ in (4.37) is:

$$\partial h(x) = \begin{cases} -\sum_{j=1}^{J} c_j, & \text{if} \quad x < b_1, \\ c_1 \cdot [-1,1] - \sum_{j=2}^{J} c_j, & \text{if} \quad x = b_1, \\ c_1 - \sum_{j=2}^{J} c_j, & \text{if} \quad b_1 < x < b_2, \\ \quad \vdots & \\ \sum_{j=1}^{J-1} c_j + c_J \cdot [-1,1], & \text{if} \quad x = b_J, \\ \sum_{j=1}^{J} c_j, & \text{if} \quad x > b_J. \end{cases} \tag{4.40}$$

From (4.38) and (4.40), extensive but routine calculations lead to the following implementation for evaluating the proximal operator of h in (4.37). Let us decompose $\mathbb{R}$ into $J+1$ intervals such that $\mathbb{R} = \bigcup_{n=0}^{J} \mathcal{I}_n$ where, as illustrated in Figure 4.2:

$$\mathcal{I}_0 \triangleq \left]-\infty\,,\, b_1 - \lambda \sum_{j=1}^{J} c_j \right[, \tag{4.41}$$

$$\mathcal{I}_n \triangleq \mathcal{I}_{n,1} \cup \mathcal{I}_{n,2}, \qquad n = 1, \ldots, J, \tag{4.42}$$

with

$$\mathcal{I}_{n,1} \triangleq \left[b_n - \lambda \left(\sum_{j=n}^{J} c_j - \sum_{j=1}^{n-1} c_j \right) \,,\, b_n - \lambda \left(\sum_{j=n+1}^{J} c_j - \sum_{j=1}^{n} c_j \right) \right[, \qquad n = 1, \ldots, J, \tag{4.43}$$

$$\mathcal{I}_{n,2} \triangleq \left[b_n - \lambda \left(\sum_{j=n+1}^{J} c_j - \sum_{j=1}^{n} c_j \right) \,,\, b_{n+1} - \lambda \left(\sum_{j=n+1}^{J} c_j - \sum_{j=1}^{n} c_j \right) \right[, \quad n = 1, \ldots, J-1, \tag{4.44}$$

$$\mathcal{I}_{J,2} \triangleq \left[b_J + \lambda \sum_{j=1}^{J} c_j \,,\, +\infty \right[. \tag{4.45}$$

Depending on the interval to which v belongs, we evaluate the proximal operator according to:

$$\text{prox}_{\lambda h}(v) = \begin{cases} v + \lambda \sum_{j=1}^{J} c_j, & \text{if } v \in \mathcal{I}_0 \\ b_n, & \text{if } v \in \mathcal{I}_{n,1} \\ v + \lambda \left(\sum_{j=n+1}^{J} c_j - \sum_{j=1}^{n} c_j \right), & \text{if } v \in \mathcal{I}_{n,2}. \end{cases} \tag{4.46}$$

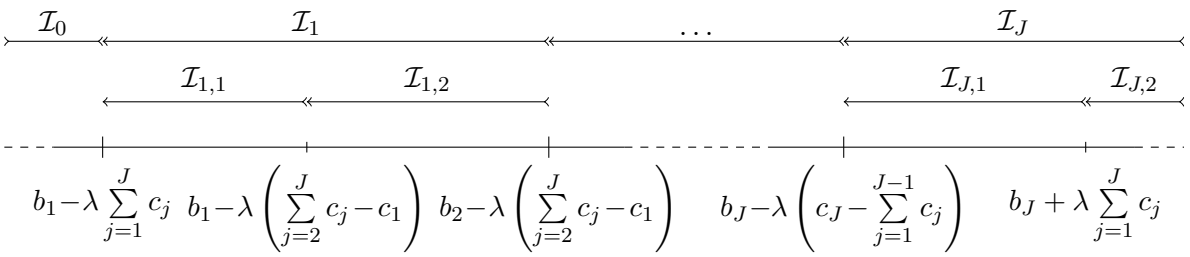
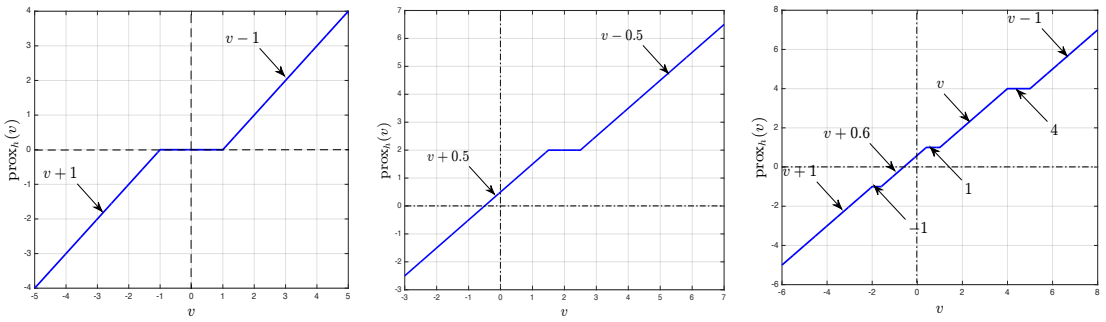


Figure 4.2: Decomposition of $\mathbb{R}$ into $J+1$ intervals given by (4.41)–(4.45). The width of the intervals depends on the weights $\{c_j\}_{j=1}^J$ and on the coefficients $\{b_j\}_{j=1}^J$.

In order to make clearer how the operator in (4.46) works, we plot $\text{prox}_h(v)$ for three expressions of h in Figure 4.3.



(a) $h(x)=|x|$ (b) $h(x)=\frac{1}{2}|x-2|$ (c) $h(x)=\frac{1}{5}|x+1|+\frac{3}{10}|x-1|+\frac{1}{2}|x-4|$

Figure 4.3: Proximal operator $\text{prox}_{\lambda h}(v)$ versus $v\in\mathbb{R}$ with $\lambda=1$ and $h:\mathbb{R}\to\mathbb{R}$, $h(x)=\sum_{j=1}^{J} c_j|x-b_j|$.

It can be checked that the proximal operator in (4.46) can be written more compactly as:

$$\text{prox}_{\lambda h}(v)=v-\lambda\Gamma(v), \tag{4.47}$$

where

$$\Gamma(v)=\frac{1}{2}\sum_{n=1}^{J}\left\{\left|\frac{v-b_n}{\lambda}-\sum_{j=1}^{n-1}c_j+\sum_{j=n}^{J}c_j\right|-\left|\frac{v-b_n}{\lambda}-\sum_{j=1}^{n}c_j+\sum_{j=n+1}^{J}c_j\right|\right\}. \tag{4.48}$$

Comparing (4.38) and (4.47), we remark that $\Gamma(v)$ is a subgradient of h at $\text{prox}_{\lambda h}(v)$. Based on equation (4.46), $\Gamma(v)$ is bounded as follows:

$$|\Gamma(v)|\leq\sum_{j=1}^{J}c_j \tag{4.49}$$

for all v. In fact, equality holds when v belongs to $\mathcal{I}_0$ in (4.41) or $\mathcal{I}_{J,2}$ in (4.45). When v belongs to an interval of the form of $\mathcal{I}_{n,1}$ in (4.43), we have:

$$\Gamma(v) = \frac{v - b_n}{\lambda} \in \left[\sum_{j=1}^{n-1} c_j - \sum_{j=n}^{J} c_j, \sum_{j=1}^{n} c_j - \sum_{j=n+1}^{J} c_j\right] \subset \left[-\sum_{j=1}^{J} c_j, \sum_{j=1}^{J} c_j\right], \tag{4.50}$$

and when it belongs to an interval of the form of $\mathcal{I}_{n,2}$ in (4.44), we have:

$$\Gamma(v) = \sum_{j=1}^{n} c_j - \sum_{j=n+1}^{J} c_j \in \left[-\sum_{j=1}^{J} c_j, \sum_{j=1}^{J} c_j\right]. \tag{4.51}$$

We note that the upper bound in (4.49) is independent of λ.

Using (4.47), the m-th entry of $\text{prox}_{\eta\mu_k g_{k,i+1}}\big(\boldsymbol{\phi}_k(i+1)\big)$ in (4.36) can be written as:

$$\Big[\text{prox}_{\eta\mu_k g_{k,i+1}}\left(\boldsymbol{\phi}_k(i+1)\right)\Big]_m = [\boldsymbol{\phi}_k(i+1)]_m - \eta\,\mu_k \Gamma_{k,m,i+1}\left([\boldsymbol{\phi}_k(i+1)]_m\right). \tag{4.52}$$

Note that $\Gamma_{k,m,i+1}\big(\big[\boldsymbol{\phi}_k(i+1)\big]_m\big)$ is a function of the form (4.48) where, based on (4.35), the coefficients b_j and c_j are given by $\big[\boldsymbol{\phi}_\ell(i+1)\big]_m$ and $p_{k\ell}\,\alpha_{k\ell}^m(i)$, respectively, and the scalar v corresponds to the m-th component of the vector $\boldsymbol{\phi}_k(i+1)$.

Using the boundedness of $\Gamma_{k,m,i+1}(\cdot)$ in (4.49), we obtain:

$$|\Gamma_{k,m,i+1}([\boldsymbol{\phi}_k(i+1)]_m)| \leq \sum_{\ell\in\mathcal{N}_k\backslash\mathcal{C}(k)} p_{k\ell}\,\alpha_{k\ell}^m(i) \;\triangleq s_k^m(i) \tag{4.53}$$

for all $[\boldsymbol{\phi}_k(i+1)]_m$. For the ℓ_1-norm (4.5), we have:

$$s_k^m(i) = s_k \triangleq \sum_{\ell\in\mathcal{N}_k\backslash\mathcal{C}(k)} p_{k\ell}, \tag{4.54}$$

for all i and $m = 1, \ldots, M$. For the reweighted ℓ_1-norm (4.6), we have:

$$\begin{aligned} s_k^m(i) &= \sum_{\ell\in\mathcal{N}_k\backslash\mathcal{C}(k)} \frac{p_{k\ell}}{\epsilon + |[\boldsymbol{\delta}_{k,\ell}(i-1)]_m|} \\ &= \frac{1}{\epsilon} \sum_{\ell\in\mathcal{N}_k\backslash\mathcal{C}(k)} \frac{p_{k\ell}}{1 + \frac{|[\boldsymbol{\delta}_{k,\ell}(i-1)]_m|}{\epsilon}} \\ &\leq \frac{s_k}{\epsilon} \end{aligned} \tag{4.55}$$

for all i and $m = 1, \ldots, M$. Using (4.52), the proximal operator of $\eta\mu_k g_{k,i+1}$ can be written as:

$$\text{prox}_{\eta\mu_k g_{k,i+1}}(\boldsymbol{\phi}_k(i+1)) = \boldsymbol{\phi}_k(i+1) - \eta\mu_k \boldsymbol{\Gamma}_{k,i+1}(\boldsymbol{\phi}_k(i+1)), \tag{4.56}$$

where $\boldsymbol{\Gamma}_{k,i+1}(\boldsymbol{\phi}_k(i+1))$ is the $M \times 1$ vector given by:

$$\boldsymbol{\Gamma}_{k,i+1}(\boldsymbol{\phi}_k(i+1)) = \text{col}\Big\{\Gamma_{k,1,i+1}([\boldsymbol{\phi}_k(i+1)]_1, \ldots, \Gamma_{k,M,i+1}([\boldsymbol{\phi}_k(i+1)]_M)\Big\}. \tag{4.57}$$

As a consequence, the ℓ_2-norm of the vector $\boldsymbol{\Gamma}_{k,i+1}(\cdot)$ can be bounded as:

$$\|\boldsymbol{\Gamma}_{k,i+1}(\cdot)\|_2 \;\leq\; s_k\sqrt{M}, \qquad \text{for the } \ell_1\text{-norm}, \tag{4.58}$$

$$\|\boldsymbol{\Gamma}_{k,i+1}(\cdot)\|_2 \;\leq\; \frac{s_k\sqrt{M}}{\epsilon}, \qquad \text{for the reweighted } \ell_1\text{-norm}. \tag{4.59}$$

4.3 Stability analysis

4.3.1 Weight error vector recursion

We shall now analyze the stability of the multitask diffusion algorithm (4.32) in the mean and mean-square-error sense. We first define at node k and iteration i the weight error vector $\widetilde{\boldsymbol{w}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_k(i)$ and the intermediate error vector $\widetilde{\boldsymbol{\phi}}_k(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\phi}_k(i)$. Furthermore, we introduce the network block vectors:

$$\widetilde{\boldsymbol{w}}_b(i) \triangleq \operatorname{col}\{\widetilde{\boldsymbol{w}}_1(i), \ldots, \widetilde{\boldsymbol{w}}_N(i)\} \tag{4.60}$$

$$\boldsymbol{\phi}_b(i) \triangleq \operatorname{col}\{\boldsymbol{\phi}_1(i), \ldots, \boldsymbol{\phi}_N(i)\} \tag{4.61}$$

$$\widetilde{\boldsymbol{\phi}}_b(i) \triangleq \operatorname{col}\left\{\widetilde{\boldsymbol{\phi}}_1(i), \ldots, \widetilde{\boldsymbol{\phi}}_N(i)\right\}. \tag{4.62}$$

Let $\boldsymbol{\mathcal{M}}$ and $\boldsymbol{\mathcal{R}}_x(i)$ be the $N \times N$ block diagonal matrices defined as:

$$\boldsymbol{\mathcal{M}} \triangleq \operatorname{diag}\{\mu_1 \boldsymbol{I}_M, \ldots, \mu_N \boldsymbol{I}_M\}, \tag{4.63}$$

$$\boldsymbol{\mathcal{R}}_x(i) \triangleq \operatorname{diag}\left\{\sum_{\ell \in \mathcal{N}_1 \cap \mathcal{C}(1)} c_{\ell 1}\, \boldsymbol{x}_\ell(i)\, \boldsymbol{x}_\ell^\top(i), \ldots, \sum_{\ell \in \mathcal{N}_N \cap \mathcal{C}(N)} c_{\ell N}\, \boldsymbol{x}_\ell(i)\, \boldsymbol{x}_\ell^\top(i)\right\}, \tag{4.64}$$

where each block is of dimension $M \times M$ and $\boldsymbol{p}_{xz}(i)$ be the $N \times 1$ block vector defined as:

$$\boldsymbol{p}_{xz}(i) \triangleq \boldsymbol{\mathcal{C}}^\top \operatorname{col}\{\boldsymbol{x}_1(i)\, z_1(i), \ldots, \boldsymbol{x}_N(i)\, z_N(i)\}, \tag{4.65}$$

where

$$\boldsymbol{\mathcal{C}} \triangleq \boldsymbol{C} \otimes \boldsymbol{I}_M, \tag{4.66}$$

with $\boldsymbol{C}$ an $N \times N$ right-stochastic matrix with ℓk-th entry $c_{\ell k}$, namely, $\boldsymbol{C} \triangleq [c_{\ell k}]$. Let

$$\boldsymbol{\mathcal{A}} \triangleq \boldsymbol{A} \otimes \boldsymbol{I}_M \tag{4.67}$$

where $\boldsymbol{A}$ is the $N \times N$ left-stochastic matrix with ℓk-th entry $a_{\ell k}$, namely, $\boldsymbol{A} \triangleq [a_{\ell k}]$. Subtracting $\boldsymbol{w}_k^o$ from both sides of the first and second step in (4.32), and using the linear data model (4.4), we obtain:

$$\widetilde{\boldsymbol{\phi}}_b(i+1) = \boldsymbol{\mathcal{A}}^\top \left[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)\right] \widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{p}_{xz}(i). \tag{4.68}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the third step in (4.32), and using result (4.56), we get:

$$\widetilde{\boldsymbol{w}}_k(i+1) = \widetilde{\boldsymbol{\phi}}_k(i+1) + \eta \mu_k\, \boldsymbol{\Gamma}_{k,i+1}(\boldsymbol{\phi}_k(i+1)). \tag{4.69}$$

Hence, the network error vector for the diffusion strategy (4.32) evolves according to the following recursion:

$$\boxed{\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{A}}^\top \left[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)\right] \widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{p}_{xz}(i) + \eta\, \boldsymbol{\mathcal{M}} \boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right),} \tag{4.70}$$

where $\boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right)$ is the $N \times 1$ block vector with k-th block given by (4.57), namely,

$$\boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right) \triangleq \operatorname{col}\left\{\boldsymbol{\Gamma}_{1,i+1}\left(\boldsymbol{\phi}_1(i+1)\right), \ldots, \boldsymbol{\Gamma}_{N,i+1}\left(\boldsymbol{\phi}_N(i+1)\right)\right\}. \tag{4.71}$$

Table 4.2: List of symbols defined throughout the performance analysis in Chapter 4

Symbol	Equation
$\boldsymbol{\mathcal{M}} = \text{diag}\{\mu_1 \boldsymbol{I}_M, \ldots, \mu_N \boldsymbol{I}_M\}$	(4.63)
$\boldsymbol{\mathcal{C}} = \boldsymbol{C} \otimes \boldsymbol{I}_M$	(4.66)
$\boldsymbol{\mathcal{A}} = \boldsymbol{A} \otimes \boldsymbol{I}_M$	(4.67)
$\boldsymbol{\mathcal{R}}_x = \text{diag}\left\{\sum_{\ell \in \mathcal{N}_1 \cap \mathcal{C}(1)} c_{\ell 1} \boldsymbol{R}_{x,\ell}, \ldots, \sum_{\ell \in \mathcal{N}_N \cap \mathcal{C}(N)} c_{\ell N} \boldsymbol{R}_{x,\ell}\right\}$	(4.78)
$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}}^\top \left[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x\right]$	(4.77)
$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}}\, \boldsymbol{\mathcal{C}}^\top \text{diag}\left\{\boldsymbol{R}_{x,1}\sigma_{z,1}^2, \ldots, \boldsymbol{R}_{x,N}\sigma_{z,N}^2\right\} \boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{A}}$	(4.99)
$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^\top \otimes \boldsymbol{\mathcal{B}}^\top$	(4.98)

In order to make the presentation clearer, we shall use the following notation for terms in recursion (4.70):

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^\top \left[\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)\right], \tag{4.72}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{p}_{xz}(i), \tag{4.73}$$

$$\boldsymbol{r}(i+1) \triangleq \boldsymbol{\mathcal{M}}\boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right). \tag{4.74}$$

Hence, recursion (4.70) can be rewritten as follows:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{g}(i) + \eta\, \boldsymbol{r}(i+1). \tag{4.75}$$

Before proceeding, let us introduce the following assumptions on the regression data and step-sizes.

Assumption 4.1. (Independent regressors) *The regression vectors $\boldsymbol{x}_k(i)$ arise from a zero-mean random process that is temporally white and spatially independent.*

It follows that $\boldsymbol{x}_k(i)$ is independent of $\boldsymbol{w}_\ell(j)$ for $i \geq j$ and for all ℓ.

For ease of reference, we list in Table 4.2 the symbols that have been defined in subsection 4.3.1, and others that will be defined in subsections 4.3.2 and 4.3.3.

4.3.2 Mean-error analysis

Taking the expectation of both sides of (4.75), using Assumption 4.1, and $\mathbb{E}\,\boldsymbol{p}_{xz}(i) = \boldsymbol{0}$, we obtain that the mean error vector evolves according to the following recursion:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}\,\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) + \eta\,\mathbb{E}\,\boldsymbol{r}(i+1), \tag{4.76}$$

where

$$\boldsymbol{\mathcal{B}} \triangleq \boldsymbol{\mathcal{A}}^{\top}\left[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_{x}\right], \tag{4.77}$$

$$\boldsymbol{\mathcal{R}}_{x} \triangleq \mathbb{E}\,\boldsymbol{\mathcal{R}}_{x}(i)=\operatorname{diag}\left\{\sum_{\ell\in\mathcal{N}_1\cap\mathcal{C}(1)} c_{\ell 1}\boldsymbol{R}_{x,\ell},\ldots,\sum_{\ell\in\mathcal{N}_N\cap\mathcal{C}(N)} c_{\ell N}\boldsymbol{R}_{x,\ell}\right\}, \tag{4.78}$$

$$\mathbb{E}\,\boldsymbol{r}(i+1) \triangleq \boldsymbol{\mathcal{M}}\,\mathbb{E}\left\{\boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right)\right\}. \tag{4.79}$$

The following theorem guarantees the mean stability of the multitask diffusion LMS (4.32) with forward-backward splitting.

Recall from Appendix A.3 that the block maximum norm of an $N\times 1$ block vector $\boldsymbol{x}=\operatorname{col}\{\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N\}$ and the induced block maximum norm of an $N\times N$ block matrix $\boldsymbol{\mathcal{X}}$ are defined as:

$$\begin{aligned}\|\boldsymbol{x}\|_{b,\infty} &= \max_{1\le k\le N}\|\boldsymbol{x}_k\|,\\ \|\boldsymbol{\mathcal{X}}\|_{b,\infty} &= \max_{\boldsymbol{x}\neq\boldsymbol{0}}\frac{\|\boldsymbol{\mathcal{X}}\boldsymbol{x}\|_{b,\infty}}{\|\boldsymbol{x}\|_{b,\infty}},\end{aligned} \tag{4.80}$$

Theorem 4.1. (Stability in the mean) *Assume the data model in* (4.4) *and Assumption 4.1 hold. Then, for any initial conditions, the multitask diffusion strategy* (4.32) *converges in the mean to a small bounded region of the order of* $\mu_{\max}$*, i.e.,* $\lim_{i\to\infty}\mathbb{E}\left\{\|\widetilde{\boldsymbol{w}}_b(i)\|_{b,\infty}\right\}=\mathcal{O}(\mu_{\max})$*, if the step-sizes are chosen such that:*

$$0<\mu_k<\frac{2}{\lambda_{\max}\left(\sum_{\ell\in\mathcal{N}_k\cap\mathcal{C}(k)} c_{\ell k}\boldsymbol{R}_{x,\ell}\right)},\quad k=1,\ldots,N, \tag{4.81}$$

where $\mu_{\max}\triangleq\max_{1\le k\le N}\mu_k$*. The block maximum norm of the bias can be upper bounded as:*

$$\lim_{i\to\infty}\|\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\|_{b,\infty} \le \frac{\eta\,\mu_{\max}\,s_{\max}\sqrt{M}}{1-\|\boldsymbol{\mathcal{B}}\|_{b,\infty}}, \tag{4.82}$$

$$\lim_{i\to\infty}\|\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\|_{b,\infty} \le \frac{1}{\epsilon}\cdot\frac{\eta\,\mu_{\max}\,s_{\max}\sqrt{M}}{1-\|\boldsymbol{\mathcal{B}}\|_{b,\infty}}, \tag{4.83}$$

for the ℓ_1*-norm and the reweighted* ℓ_1*-norm, respectively.*

Proof. Iterating (4.76) starting from $i=0$, we arrive to the following expression:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1)=\boldsymbol{\mathcal{B}}^{i+1}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(0)+\eta\sum_{j=0}^{i}\boldsymbol{\mathcal{B}}^{j}\,\mathbb{E}\,\boldsymbol{r}(i+1-j), \tag{4.84}$$

where $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(0)$ is the initial condition. $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1)$ converges when $i\to\infty$ if, and only if, both terms on the RHS of (4.84) converges to finite values.

The first term converges to zero as $i\to\infty$ if the matrix $\boldsymbol{\mathcal{B}}$ is contractive, that is, $\rho(\boldsymbol{\mathcal{B}})<1$. Since any induced matrix norm is lower bounded by its spectral radius, we can write in terms of the block maximum norm:

$$\rho\left(\boldsymbol{\mathcal{A}}^{\top}\left[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_{x}\right]\right)\le\|\boldsymbol{\mathcal{A}}^{\top}\left[\boldsymbol{I}_{MN}-\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_{x}\right]\|_{b,\infty}. \tag{4.85}$$

Using the submultiplicative property of the block maximum norm, property (A.23), and property (A.22), we find that condition (4.81) is a sufficient condition to ensure a contractive matrix $\boldsymbol{\mathcal{B}}$.

We shall now prove the convergence of the second term on the RHS of (4.84). To prove the convergence of the series $\sum_{j=0}^{+\infty} \boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)$, it is sufficient to prove that the series $\sum_{j=0}^{+\infty} \left[\boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\right]_m$ converges for $m = 1, \ldots, MN$. According to the comparison test [Whittaker and Watson, 1996], a series is absolutely convergent if each term of the series can be bounded by a term of an absolutely convergent series. Since the block maximum norm of a block vector is greater than or equal to the largest absolute value of its entries, each term $\left|\left[\boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\right]_m\right|$ can be bounded as:

$$\begin{aligned} \left|\left[\boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\right]_m\right| &\leq \|\boldsymbol{\mathcal{B}}\|_{b,\infty}^j \cdot \|\mathbb{E}\, \boldsymbol{r}(i+1-j)\|_{b,\infty} \\ &\leq \|\boldsymbol{\mathcal{B}}\|_{b,\infty}^j r_{\max}. \end{aligned} \tag{4.86}$$

The quantity $\|\mathbb{E}\, \boldsymbol{r}(i+1-j)\|_{b,\infty}$ is finite for all i and j and bounded by some constant $r_{\max} = \mathcal{O}(\mu_{\max})$. In fact, from (4.79), we have:

$$\|\mathbb{E}\, \boldsymbol{r}(i+1)\|_{b,\infty} \leq \mu_{\max} \, \|\mathbb{E}\, \{\boldsymbol{\Gamma}_{b,i+1}\left(\boldsymbol{\phi}_b(i+1)\right)\}\|_{b,\infty} \tag{4.87}$$

since $\|\boldsymbol{\mathcal{M}}\|_{b,\infty} = \mu_{\max}$. Using (4.58)–(4.59), the block maximum norm of $\boldsymbol{\Gamma}_{b,i+1}(\boldsymbol{\phi}_b(i+1))$ in (4.71) can be bounded as:

$$\|\boldsymbol{\Gamma}_{b,i+1}(\boldsymbol{\phi}_b(i+1))\|_{b,\infty} \leq s_{\max}\sqrt{M}, \quad (\ell_1\text{-norm}) \tag{4.88}$$

$$\|\boldsymbol{\Gamma}_{b,i+1}(\boldsymbol{\phi}_b(i+1))\|_{b,\infty} \leq \frac{s_{\max}\sqrt{M}}{\epsilon}, \quad (\text{rew. } \ell_1\text{-norm}) \tag{4.89}$$

for all i, where $s_{\max} = \max\limits_{1\leq k\leq N} s_k$. If the step-sizes are chosen according to (4.81), the series $\sum_{j=0}^{+\infty} \|\boldsymbol{\mathcal{B}}\|_{b,\infty}^j r_{\max}$ is absolutely convergent. Therefore, the series $\sum_{j=0}^{+\infty} \left[\boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\right]_m$ is an absolutely convergent series.

Note that when $i \to \infty$, the block maximum norm of the bias can be bounded as

$$\begin{aligned} \lim_{i\to\infty} \|\mathbb{E}\, \widetilde{\boldsymbol{w}}_b(i)\|_{b,\infty} &= \lim_{i\to\infty} \Big\|\eta \sum_{j=0}^{i} \boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\Big\|_{b,\infty} \\ &\leq \lim_{i\to\infty} \eta \sum_{j=0}^{\infty} \big\|\boldsymbol{\mathcal{B}}^j \, \mathbb{E}\, \boldsymbol{r}(i+1-j)\big\|_{b,\infty} \\ &\leq \lim_{i\to\infty} \eta \sum_{j=0}^{\infty} \|\boldsymbol{\mathcal{B}}\|_{b,\infty}^j \, r_{\max} = \frac{\eta \, r_{\max}}{1 - \|\boldsymbol{\mathcal{B}}\|_{b,\infty}}. \end{aligned} \tag{4.90}$$

□

Observe that the mean stability (4.81) does not depend on the combination coefficients $a_{\ell k}$ and the regularization factors $\rho_{k\ell}$. Only the matrix $\boldsymbol{C}$ influences the condition on the step-sizes.

When $\boldsymbol{C} = \boldsymbol{I}_N$, i.e., the agents do not share the raw data $\{d_\ell(i), \boldsymbol{x}_\ell(i)\}$, condition (4.81) becomes:

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(\boldsymbol{R}_{x,k}\right)}, \quad k = 1, \ldots, N, \tag{4.91}$$

which is the same as the condition obtained in the non-cooperative solution. For uniform step-sizes, i.e., $\mu_k = \mu$ for all k, and doubly stochastic matrix $\boldsymbol{C}$, using similar arguments as in (2.63), condition (4.81) on the step-sizes becomes:

$$0 < \mu < \min_{1 \le k \le N} \left\{ \frac{2}{\lambda_{\max}\left(\boldsymbol{R}_{x,k}\right)} \right\}. \tag{4.92}$$

4.3.3 Mean-square-error stability

We examine the mean-square-error stability by studying the convergence of the weighted square value $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$, where $\boldsymbol{\Sigma}$ is a positive semi-definite matrix that we are free to choose. Evaluating the variance, we obtain:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\Sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}'} + \mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} + \varphi(\boldsymbol{r}(i+1), \boldsymbol{\Sigma}, \boldsymbol{\mathcal{B}}(i), \widetilde{\boldsymbol{w}}_b(i), \boldsymbol{g}(i)), \tag{4.93}$$

where $\boldsymbol{\Sigma}' \triangleq \mathbb{E}\left\{\boldsymbol{\mathcal{B}}^\top(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\right\}$ and

$$\begin{aligned}\varphi\left(\boldsymbol{r}(i+1), \boldsymbol{\Sigma}, \boldsymbol{\mathcal{B}}(i), \widetilde{\boldsymbol{w}}_b(i), \boldsymbol{g}(i)\right) =& \eta^2 \mathbb{E}\,\|\boldsymbol{r}(i+1)\|^2_{\boldsymbol{\Sigma}} + 2\,\eta\,\mathbb{E}\left\{\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i)\right\} \\ & - 2\,\eta\,\mathbb{E}\left\{\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\boldsymbol{g}(i)\right\}\end{aligned} \tag{4.94}$$

is a term coming from promoting relationships between clusters. The last two terms on the RHS of (4.94) contain higher-order powers of the step-sizes. For *sufficiently small step-sizes*, using a separation principle (see Subsection 2.4.3), we get the following approximation:

$$\varphi(\boldsymbol{r}(i+1), \boldsymbol{\Sigma}, \widetilde{\boldsymbol{w}}_b(i)) \approx \eta^2 \mathbb{E}\,\|\boldsymbol{r}(i+1)\|^2_{\boldsymbol{\Sigma}} + 2\,\eta\,\mathbb{E}\left\{\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\widetilde{\boldsymbol{w}}_b(i)\right\}, \qquad \text{(small step-sizes)}. \tag{4.95}$$

Let $\boldsymbol{\sigma} \triangleq \mathrm{vec}(\boldsymbol{\Sigma})$ and $\boldsymbol{\sigma}' \triangleq \mathrm{vec}(\boldsymbol{\Sigma}')$ where the $\mathrm{vec}(\cdot)$ operator stacks the columns of a matrix on top of each other. We will use the notation $\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}}$ and $\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}}$ interchangeably to denote the same quantity $\widetilde{\boldsymbol{w}}_b(i)^\top \boldsymbol{\Sigma} \widetilde{\boldsymbol{w}}_b(i)$. Using property (A.6), the relation between $\boldsymbol{\sigma}'$ and $\boldsymbol{\sigma}$ can be expressed in the following form:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}\boldsymbol{\sigma}, \tag{4.96}$$

where $\boldsymbol{\mathcal{F}}$ is the $(MN)^2 \times (MN)^2$ matrix given by:

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\left\{\boldsymbol{\mathcal{B}}^\top(i) \otimes \boldsymbol{\mathcal{B}}^\top(i)\right\} \tag{4.97}$$

$$\approx \boldsymbol{\mathcal{B}}^\top \otimes \boldsymbol{\mathcal{B}}^\top, \qquad \text{(small step-sizes)}. \tag{4.98}$$

The approximation in (4.98) is reasonable under the small step-sizes condition, where the effect of terms involving higher order powers of step-sizes is ignored. Introducing the matrix $\boldsymbol{\mathcal{G}}$:

$$\boldsymbol{\mathcal{G}} \triangleq \mathbb{E}\left\{\boldsymbol{g}(i)\boldsymbol{g}^\top(i)\right\} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{M}}\, \boldsymbol{\mathcal{C}}^\top \mathrm{diag}\left\{\boldsymbol{R}_{x,1}\sigma^2_{z,1}, \ldots, \boldsymbol{R}_{x,N}\sigma^2_{z,N}\right\} \boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{A}}, \tag{4.99}$$

and using property (A.5), the second term on the RHS of (4.93) can be written as:

$$\mathbb{E}\,\|\boldsymbol{g}(i)\|_{\boldsymbol{\Sigma}}^2 = \Big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\Big]^\top \boldsymbol{\sigma}. \tag{4.100}$$

Hence, the variance recursion (4.93) can be expressed as

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2 = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2 + \Big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\Big]^\top \boldsymbol{\sigma} + \varphi(\boldsymbol{r}(i+1), \boldsymbol{\sigma}, \widetilde{\boldsymbol{w}}_b(i)). \tag{4.101}$$

Theorem 4.2. (Mean-square-error stability) *Assume the data model in* (4.4) *and Assumptions 4.1 hold. Assume further that the step-sizes are sufficiently small. Then, for any initial conditions, the multitask diffusion strategy* (4.32) *is mean-square stable if the error recursion* (4.70) *is mean stable and the matrix* $\boldsymbol{\mathcal{F}}$ *in* (4.97) *is contractive. Using the approximation* (4.98) *and property* (A.8)*, we find that small step-sizes satisfying* (4.81) *will also ensure mean-square stability.*

Proof. Since $\boldsymbol{\Sigma}$ is a positive semi-definite matrix and the vector $\boldsymbol{r}(i+1)$ is uniformly bounded for all i, $\mathbb{E}\,\|\boldsymbol{r}(i+1)\|_{\boldsymbol{\Sigma}}^2$ can be bounded as

$$0 \le \eta^2 \mathbb{E}\,\|\boldsymbol{r}(i+1)\|_{\boldsymbol{\Sigma}}^2 \le \kappa_1 \tag{4.102}$$

for all i, where κ_1 is a positive constant. Since $\boldsymbol{r}(i+1)$ is uniformly bounded for all i, the vector $2\,\eta\,\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\,\boldsymbol{\mathcal{B}}$ is also bounded for all i. Let $\gamma_{\max}$ be a bound on the largest component of $2\,\eta\,\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\,\boldsymbol{\mathcal{B}}$ in absolute value for all i. We obtain

$$\begin{aligned} 2\,\eta\,\Big|\mathbb{E}\left\{\boldsymbol{r}^\top(i+1)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\widetilde{\boldsymbol{w}}_b(i)\right\}\Big| &\le \gamma_{\max}\sum_{m=1}^{MN} |\,[\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)]_m\,| \\ &= \gamma_{\max}\cdot\|\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\|_1\,. \end{aligned} \tag{4.103}$$

Under condition (4.81) on the step-sizes, the mean error vector $\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)$ converges to a small bounded region as $i \to \infty$. Hence, $\|\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i)\|_1$ can be upper bounded by some positive constant scalar κ_2 for all i, and using the approximation (4.95), $|\varphi(\boldsymbol{r}(i+1), \boldsymbol{\sigma}, \widetilde{\boldsymbol{w}}_b(i))|$ satisfies:

$$|\varphi(\boldsymbol{r}(i+1), \boldsymbol{\sigma}, \widetilde{\boldsymbol{w}}_b(i))| \le \kappa_1 + \gamma_{\max}\kappa_2 \tag{4.104}$$

for all i. The positive constant $\kappa_3 \triangleq \kappa_1 + \gamma_{\max}\kappa_2$ can be written as a scaled multiple of the positive quantity $\big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\big]^\top \boldsymbol{\sigma}$ as $\kappa_3 = t\,\big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\big]^\top \boldsymbol{\sigma}$ where $t \ge 0$ [Lorenzo and Sayed, 2013]. We arrive at the following inequality for (4.101):

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2 \le \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2 + (1+t)\cdot\Big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\Big]^\top \boldsymbol{\sigma}. \tag{4.105}$$

Iterating (4.105) starting from $i = 0$, we obtain

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|_{\boldsymbol{\sigma}}^2 \le \mathbb{E}\,\left\{\|\widetilde{\boldsymbol{w}}_b(0)\|_{\boldsymbol{\mathcal{F}}^{i+1}\boldsymbol{\sigma}}^2\right\} + (1+t)\,\Big[\mathrm{vec}(\boldsymbol{\mathcal{G}}^\top)\Big]^\top \sum_{j=0}^{i} \boldsymbol{\mathcal{F}}^j\boldsymbol{\sigma}, \tag{4.106}$$

where $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(0)\|^2$ is the initial condition. If we show that the RHS of (4.106) converges, then $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\sigma}}$ is stable. The first term on the RHS of (4.106) vanishes as $i \to \infty$ if the matrix $\boldsymbol{\mathcal{F}}$ is contractive. Consider now the second term on the RHS of (4.106). The series $\sum_{j=0}^{\infty} \boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}$ converges if $\sum_{j=0}^{\infty} \left[\boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}\right]_m$ converges for $m = 1, \ldots, (MN)^2$. Each term of the series can be bounded as:

$$\left[\boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}\right]_m \leq \left|\left[\boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}\right]_m\right| \leq \left\|\boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}\right\|_{b,\infty} \leq \left\|\boldsymbol{\mathcal{F}}^j\right\|_{b,\infty} \cdot \|\boldsymbol{\sigma}\|_{b,\infty}\,. \tag{4.107}$$

Since $\boldsymbol{\mathcal{F}}$ is contractive, there exists a submultiplicative norm[2] $\|\cdot\|_\rho$ such that $\|\boldsymbol{\mathcal{F}}\|_\rho = \upsilon < 1$. All norms are equivalent in finite dimensional vector spaces. Thus, we have:

$$\|\boldsymbol{\mathcal{F}}^j\|_{b,\infty} \leq \tau \|\boldsymbol{\mathcal{F}}^j\|_\rho \leq \tau \|\boldsymbol{\mathcal{F}}\|^j_\rho = \tau \upsilon^j, \tag{4.108}$$

for some positive constant τ. Considering this bound with (4.107) yields:

$$\sum_{j=0}^{\infty} \left|[\boldsymbol{\mathcal{F}}^j \boldsymbol{\sigma}]_m\right| \leq \sum_{j=0}^{\infty} \|\boldsymbol{\mathcal{F}}^j\|_{b,\infty} \cdot \|\boldsymbol{\sigma}\|_{b,\infty} \leq \tau \sum_{j=0}^{\infty} \upsilon^j \|\boldsymbol{\sigma}\|_{b,\infty} = \frac{\tau \cdot \|\boldsymbol{\sigma}\|_{b,\infty}}{1-\upsilon}. \tag{4.109}$$

As a consequence, since the second term on the RHS of (4.106) converges to a bounded region when $\boldsymbol{\mathcal{F}}$ is contractive, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\sigma}}$ also converges. □

4.4 Simulation results

Before proceeding, we present a new rule for selecting the regularization weight $p_{k\ell}$ based on a measure of sparsity of the vector $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$. The intuition behind this rule is to employ a large weight $p_{k\ell}$ when the objectives at nodes k and ℓ have few distinct entries, i.e., $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$ is sparse, and a small weight $p_{k\ell}$ when the objectives have few similar entries, i.e., $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$ is not sparse. Among other possible choices for the sparsity measure, we select a popular one based on a relationship between the ℓ_1-norm and ℓ_2-norm [Hoyer, 2004]:

$$\xi\left(\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell\right) = \frac{M}{M-\sqrt{M}}\left(1 - \frac{\|\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell\|_1}{\sqrt{M} \cdot \|\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell\|}\right) \in [0,1]. \tag{4.110}$$

The quantity $\xi(\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell)$ is equal to one when only a single component of $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$ is non-zero, and zero when all elements of $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$ are relatively large [Hoyer, 2004]. Since the nodes do not know the true objectives $\boldsymbol{w}^o_k$ and $\boldsymbol{w}^o_\ell$, we propose to replace these quantities by the available estimates at each time instant i and allow the regularization factors to vary with time according to:

$$p_{k\ell}(i) \propto \begin{cases} \frac{M}{M-\sqrt{M}}\left(1 - \frac{\|\boldsymbol{\phi}_k(i+1) - \boldsymbol{\phi}_\ell(i+1)\|_1}{\sqrt{M}\cdot\|\boldsymbol{\phi}_k(i+1) - \boldsymbol{\phi}_\ell(i+1)\|}\right), & \text{if } \ell \in \mathcal{N}_k \setminus \mathcal{C}(k) \\ 0, \quad \text{otherwise} \end{cases} \tag{4.111}$$

where the symbol $\propto$ denotes proportionality. As we shall see in the simulations, this rule improves the performance of the algorithm and allows agent k to adapt the regularization strength $p_{k\ell}$ with respect to the sparsity level of the vector $\boldsymbol{w}^o_k - \boldsymbol{w}^o_\ell$ at time instant i.

[2] The norm $\|\cdot\|_\rho$ is called submultiplicative if for any square matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ of compatible dimensions we have: $\|\boldsymbol{XY}\|_\rho \leq \|\boldsymbol{X}\|_\rho \cdot \|\boldsymbol{Y}\|_\rho$.

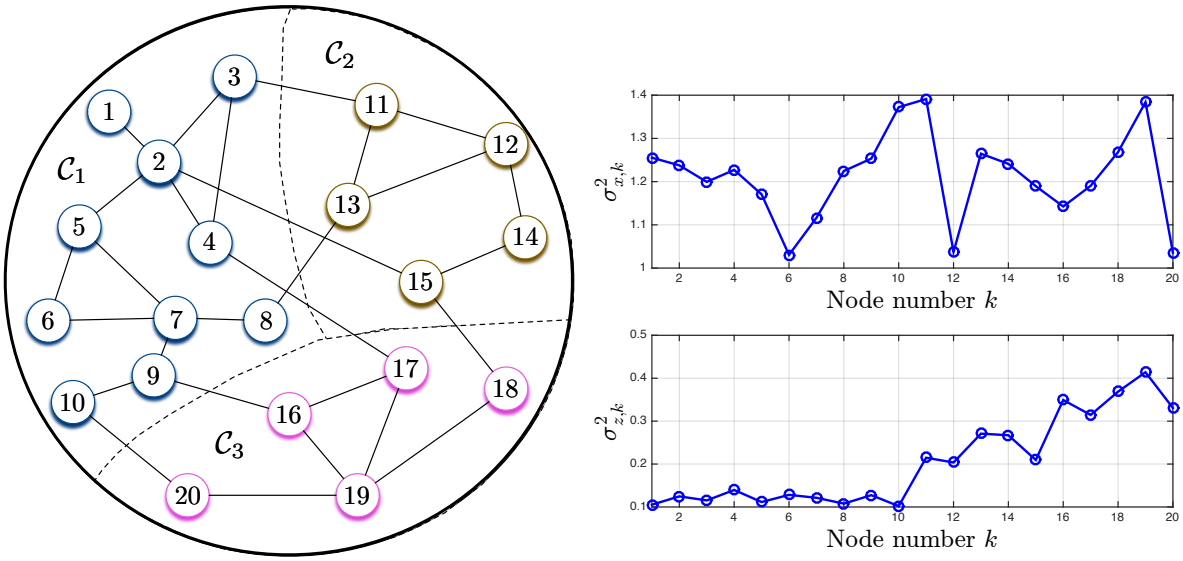


Figure 4.4: Experimental setup. (*Left*) Network topology. (*Right*) Regression and noise variances.

4.4.1 Illustrative example

We consider a clustered network with the topology shown in Figure 4.4 (left), consisting of 20 nodes divided into 3 clusters: $\mathcal{C}_1 = \{1, \ldots, 10\}$, $\mathcal{C}_2 = \{11, \ldots, 15\}$, and $\mathcal{C}_3 = \{16, \ldots, 20\}$. The regression vectors $\boldsymbol{x}_k(i)$ are 18×1 zero-mean Gaussian distributed vectors with covariance matrices $\boldsymbol{R}_{x,k} = \sigma^2_{x,k}\boldsymbol{I}_{18}$. The variances $\sigma^2_{x,k}$ are shown in Figure 4.4 (right). The noises $z_k(i)$ are zero-mean i.i.d. Gaussian random variables, independent of any other signal, with variances $\sigma^2_{z,k}$ shown in Figure 4.4 (right). We run the diffusion algorithm (4.32) by setting $c_{\ell k} = \frac{1}{\text{card}\{\mathcal{N}_\ell \cap \mathcal{C}(\ell)\}}$ for $k \in \mathcal{N}_\ell \cap \mathcal{C}(\ell)$ and $a_{\ell k} = \frac{1}{\text{card}\{\mathcal{N}_k \cap \mathcal{C}(k)\}}$ for $\ell \in \mathcal{N}_k \cap \mathcal{C}(k)$. The regularization weights are set to $\rho_{k\ell} = \frac{1}{\text{card}\{\mathcal{N}_k \setminus \mathcal{C}(k)\}}$ for $\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)$. We use a constant step-size $\mu = 0.02$ for all nodes, a sparsity strength $\eta = 0.06$ for the ℓ_1-norm regularizer, and $\eta = 0.04$ for the reweighted ℓ_1-norm regularizer with $\epsilon = 0.1$. The results are averaged over 200 Monte-Carlo runs.

The optimum vectors are set to $\boldsymbol{w}^o_{\mathcal{C}_q} = \boldsymbol{w}_0 + \delta\boldsymbol{w}_{\mathcal{C}_q}$ at each cluster with $\boldsymbol{w}_0$ an 18×1 vector whose entries are generated from the Gaussian distribution $\mathcal{N}(0, 1)$. First, we set $\delta\boldsymbol{w}_{\mathcal{C}_1}$ to $\boldsymbol{0}^\top_{1\times 18}$, $\delta\boldsymbol{w}_{\mathcal{C}_2}$ to $[-1\ \boldsymbol{0}_{1\times 17}]^\top$, and $\delta\boldsymbol{w}_{\mathcal{C}_3}$ to $[\boldsymbol{0}_{1\times 6}\ -1\ \boldsymbol{0}_{1\times 11}]^\top$. Observe that at most two entries differ between clusters. After 500 iterations, we set $\delta\boldsymbol{w}_{\mathcal{C}_2}$ to $[-\mathbb{1}_{1\times 3}\ 1\ \boldsymbol{0}_{1\times 14}]^\top$ and $\delta\boldsymbol{w}_{\mathcal{C}_3}$ to $[\boldsymbol{0}_{1\times 12}\ -\mathbb{1}_{1\times 3}\ \boldsymbol{0}_{1\times 3}]^\top$. In this way, at most 7 entries differ between clusters. After 1000 iterations, we set $\delta\boldsymbol{w}_{\mathcal{C}_2}$ to $[-\mathbb{1}_{1\times 3}\ \mathbb{1}_{1\times 3}\ -\mathbb{1}_{1\times 3}\ \boldsymbol{0}_{1\times 9}]^\top$ and $\delta\boldsymbol{w}_{\mathcal{C}_3}$ to $[\boldsymbol{0}_{1\times 9}\ \mathbb{1}_{1\times 3}\ -\mathbb{1}_{1\times 3}\ \mathbb{1}_{1\times 3}]^\top$. Thus, at most 18 entries now differ between clusters.

In Figure 4.5, we compare 6 algorithms: the non-cooperative LMS (algorithm (4.32) with $\boldsymbol{A} = \boldsymbol{C} = \boldsymbol{I}_N$ and $\eta = 0$):

$$\boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\right), \tag{4.112}$$

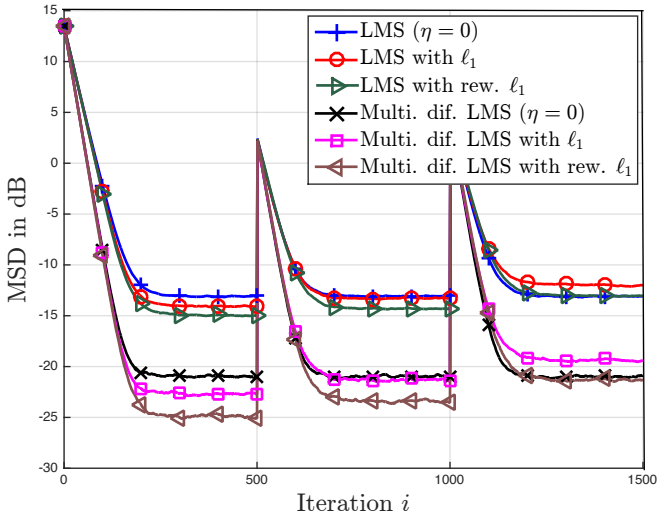


Figure 4.5: Network MSD comparison for 6 different strategies: non-cooperative LMS (4.112), spatially regularized LMS (4.113) with ℓ_1-norm and reweighted ℓ_1-norm, standard diffusion without cooperation between clusters (4.114), and our proximal diffusion (4.32) with ℓ_1-norm and reweighted ℓ_1-norm.

the regularized LMS (algorithm (4.32) with $\boldsymbol{A} = \boldsymbol{C} = \boldsymbol{I}_N$) with ℓ_1-norm and reweighted ℓ_1-norm:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \boldsymbol{x}_k(i) \left(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)\right), \\ \boldsymbol{w}_k(i+1) = \text{prox}_{\eta\mu_k g_{k,i+1}}(\boldsymbol{\psi}_k(i+1)), \end{cases} \tag{4.113}$$

the multitask diffusion LMS without regularization (algorithm (4.32) with $\eta = 0$):

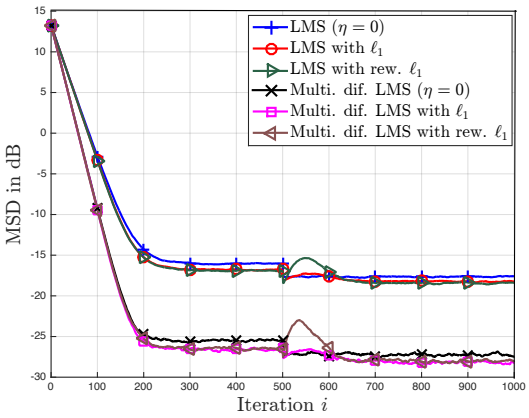
$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \displaystyle\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} c_{\ell k}\, \boldsymbol{x}_\ell(i) \big[d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)\big], \\ \boldsymbol{w}_k(i+1) = \displaystyle\sum_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k} \boldsymbol{\psi}_\ell(i+1), \end{cases} \tag{4.114}$$

and the multitask diffusion LMS (4.32) with ℓ_1-norm and reweighted ℓ_1-norm regularization. As observed in this figure, when the tasks share a sufficient number of components, cooperation between clusters enhances the network MSD performance given in (3.78). When the number of common entries decreases, the cooperation between clusters becomes less effective. The use of the ℓ_1-norm can lead to a degradation of the network MSD relative to the absence of cooperation among clusters. However, the use of the reweighted ℓ_1-norm allows to improve the performance.

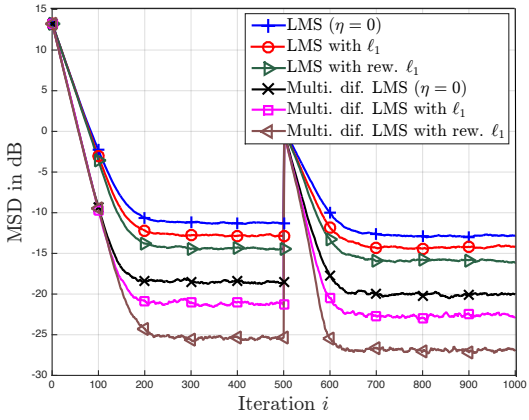
In order to better understand the behavior of the algorithm (4.32) in the clusters, we report in Figure 4.6 the learning curves for $i \in [0, 1000]$ of the common and distinct entries among clusters given by

$$\frac{1}{\text{card}\{\mathcal{C}_q\}} \sum_{k \in \mathcal{C}_q} \mathbb{E} \left\{ \sum_{m \in \mathcal{E}(i)} \left([\boldsymbol{w}_k^o(i) - \boldsymbol{w}_k(i)]_m\right)^2 \right\}, \tag{4.115}$$

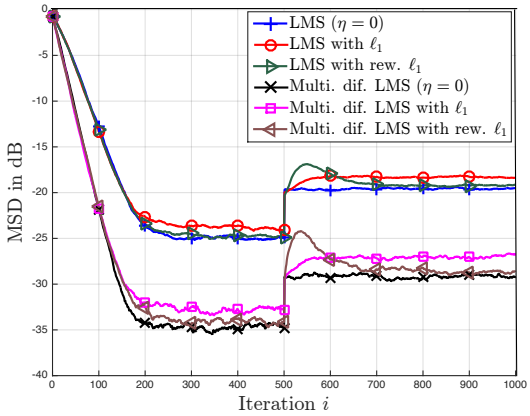
for $q = 1,\ 3$, where $\mathcal{E}(i)$ is the set of identical (or distinct) components among all clusters at iteration i and $\boldsymbol{w}_k^o(i)$ is the optimum parameter vector at node k and iteration i. For example,



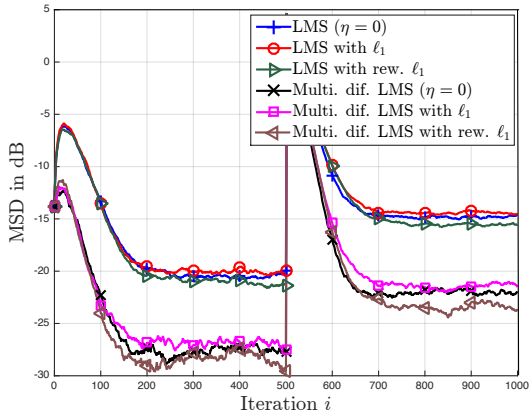
(a) Cluster 1 MSD over identical entries.



(b) Cluster 3 MSD over identical entries.



(c) Cluster 1 MSD over distinct entries.



(d) Cluster 3 MSD over distinct entries.

Figure 4.6: Clusters MSD over identical and distinct components. Comparison for the same 6 different strategies considered in Figure 4.5.

for $i \in [0, 500]$, the set of distinct components is $\{1, 7\}$. As shown in this figure, cluster $\mathcal{C}_3$ benefits considerably from cooperation with other clusters in the estimation of the common entries. Nevertheless, cluster $\mathcal{C}_1$ benefits slightly from cooperation. This is due to the fact that the performance of $\mathcal{C}_3$ is low relatively to that of $\mathcal{C}_1$ since the signal-to-noise ratio (SNR) in $\mathcal{C}_3$ is small and the number of nodes employed in this cluster is 5.

We shall now illustrate the effect of the regularization strength η over the performance of the algorithm for different numbers of common entries between the optimum vectors $\{\boldsymbol{w}_k^o\}$. We consider the same settings as above, which means that the number of common entries among clusters is successively set to 16, 11, and 0 over 18. Parameter η is uniformly sampled over $[0, 0.14]$. Figure 4.7 shows the gain in steady-state MSD versus the unregularized algorithm obtained for $\eta = 0$, as a function of η. For each η, the results are averaged over 50 Monte-Carlo runs and over 50 samples after convergence of the algorithm. It can be observed in Figure 4.7 that the interval for η over which the network benefits from cooperation between clusters becomes smaller as the number of common entries decreases. In addition, the reweighted

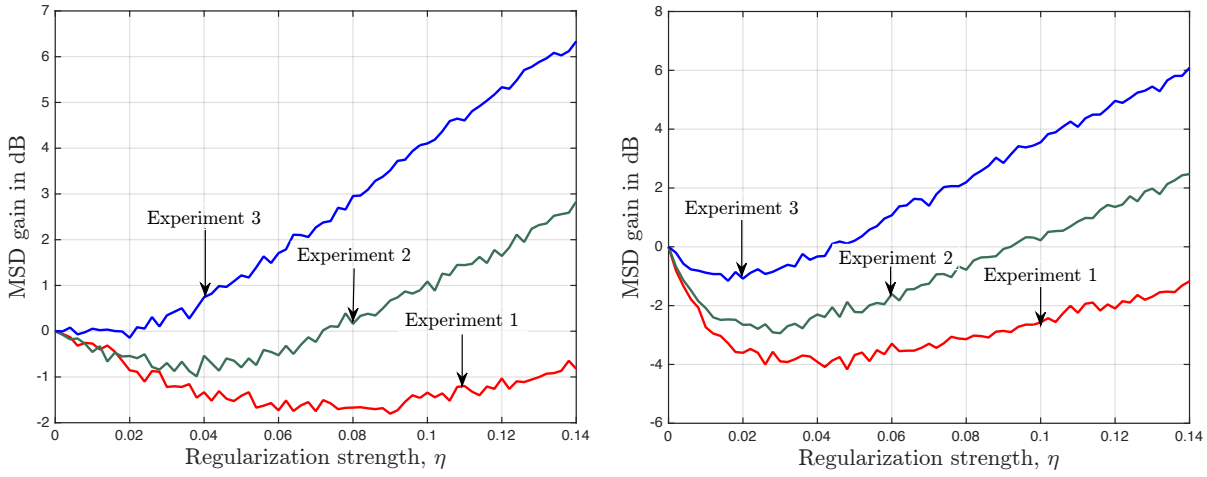


Figure 4.7: Differential network MSD ($\text{MSD}(\eta) - \text{MSD}(\eta = 0)$) in dB with respect to the regularization strength η for the multitask diffusion LMS (4.32) with ℓ_1-norm (*left*) and reweighted ℓ_1-norm (*right*) for 3 different degrees of similarity between tasks. Experiment 1: at most 2 entries differ between clusters. Experiment 2: at most 7 entries differ between clusters. Experiment 3: at most 18 entries differ between clusters.

ℓ_1-norm regularizer provides better performance than the ℓ_1-norm regularizer.

In order to guarantee a correct cooperation among clusters, we repeat the same experiment as Figure 4.5 using the adaptive rule in (4.111) for adjusting the regularization factors $p_{k\ell}$. The proportionality coefficient in (4.111) is set equal to one. As shown in Figure 4.8, when the number of distinct components is small, both ℓ_1 and reweighted ℓ_1-norms yield better performance than the diffusion LMS with $\eta = 0$. When the number of distinct components increases ($i \in (1000, 1500]$), the performance of strategy (4.32) with ℓ_1-norm gets closer to diffusion LMS with $\eta = 0$, while the reweighted ℓ_1-norm still guarantees a gain. Thus, the mechanism proposed in (4.111) for the selection of the regularization factors improves the cooperation between nodes belonging to distinct clusters.

Finally, we compare the current multitask diffusion strategy (4.32) with two other useful strategies existing in the literature [Plata-Chaves et al., 2015, Chen et al., 2014c]. We consider a stationary environment where the optimum parameter vectors $\{\boldsymbol{w}_{\mathcal{C}_q}^o\}_{q=1}^3$ consist of a sub-vector $\boldsymbol{\xi}^o$ of 16 parameters of global interest to the whole network and a 2×1 sub-vector $\{\boldsymbol{\varsigma}_{\mathcal{C}_q}^o\}$ of common interest to nodes belonging to cluster $\mathcal{C}_q$, namely, $\boldsymbol{w}_{\mathcal{C}_q}^o = \text{col}\{\boldsymbol{\xi}^o, \boldsymbol{\varsigma}_{\mathcal{C}_q}^o\}$. The entries of $\boldsymbol{\xi}^o$, $\boldsymbol{\varsigma}_{\mathcal{C}_1}^o$, $\boldsymbol{\varsigma}_{\mathcal{C}_2}^o$, and $\boldsymbol{\varsigma}_{\mathcal{C}_3}^o$ are uniformly sampled from a uniform distribution $\mathcal{U}(-3, 3)$. Except for these changes, we consider the same experimental setup described in the first paragraph of the current section. When applying the strategy developed in [Plata-Chaves et al., 2015], we assume that node k belonging to cluster $\mathcal{C}_q$ is aware that the first 16 parameters of $\boldsymbol{w}_{\mathcal{C}_q}^o$ are of global interest to the whole network while the remaining parameters are of common interest to nodes in cluster $\mathcal{C}_q$. However, the current method (4.32) and the algorithm in [Chen et al., 2014c] do not require such

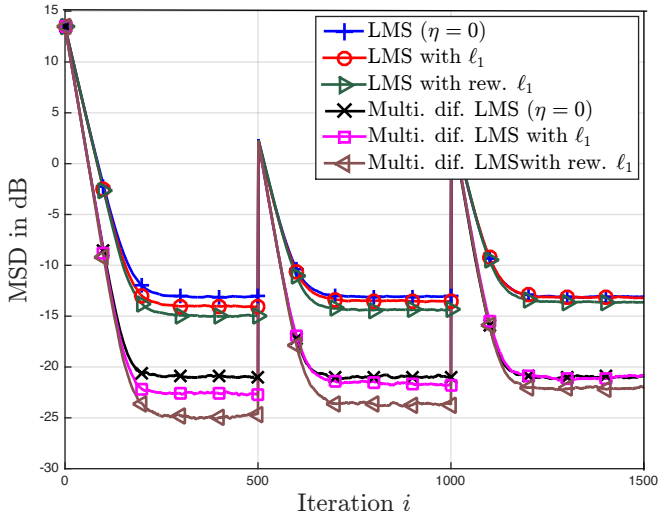


Figure 4.8: Network MSD comparison for the same 6 different strategies considered in Figure 4.5 using adaptive regularization factors $p_{k\ell}(i)$.

assumption. We run the ATC diffusion node specific parameter estimation (D-NSPE) strategy developed in [Plata-Chaves et al., 2015], and which is given by:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu\, \boldsymbol{x}_k(i)(d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_k(i)), \\ \boldsymbol{\xi}_k(i+1) = \sum\limits_{\ell \in \mathcal{N}_k} a_{\ell k}^{\xi} \boldsymbol{\psi}_k^{\xi}(i+1), \\ \boldsymbol{\varsigma}_k(i+1) = \sum\limits_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k}^{\varsigma_{\mathcal{C}(k)}} \boldsymbol{\psi}_k^{\varsigma}(i+1), \end{cases} \tag{4.116}$$

where $\boldsymbol{\psi}_k(i+1) = \text{col}\{\boldsymbol{\psi}_k^{\xi}(i+1), \boldsymbol{\psi}_k^{\varsigma}(i+1)\}$, and $\boldsymbol{\xi}_k(i+1)$, $\boldsymbol{\varsigma}_k(i+1)$ are the estimates of $\boldsymbol{\xi}^o$ and $\boldsymbol{\varsigma}_{\mathcal{C}(k)}^o$ at node k, respectively. We use uniform combination weights $a_{\ell k}^{\xi} = 1/\text{card}\{\mathcal{N}_k\}$ for $\ell \in \mathcal{N}_k$ and $a_{\ell k}^{\varsigma_{\mathcal{C}(k)}} = 1/\text{card}\{\mathcal{N}_k \cap \mathcal{C}(k)\}$ for $\ell \in \mathcal{N}_k \cap \mathcal{C}(k)$, and uniform step-sizes $\mu_k = 0.02\ \forall k$. We run the multitask diffusion strategy developed in [Chen et al., 2014c] which is given by:

$$\begin{cases} \boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \sum\limits_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} c_{\ell k} \boldsymbol{x}_\ell(i) \left(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)\right) \\ \qquad\qquad + \eta\, \mu_k \left(\sum\limits_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)} \rho_{k\ell}(\boldsymbol{w}_\ell(i) - \boldsymbol{w}_k(i)) \right), \\ \boldsymbol{w}_k(i+1) = \sum\limits_{\ell \in \mathcal{N}_k \cap \mathcal{C}(k)} a_{\ell k}\, \boldsymbol{\psi}_\ell(i+1). \end{cases} \tag{4.117}$$

by setting $\{c_{\ell k}, a_{\ell k}, \rho_{k\ell}\}$ in the same manner described in the first paragraph of the current section, $\mu_k = 0.02\ \forall k$, and $\eta = 0.06$. The learning curves of the algorithms are reported in Figure 4.9. As expected, it can be observed that the cooperation between clusters based on the ℓ_2-norm [Chen et al., 2014c] degrades the performance relative to the case of non-cooperative clusters, i.e., $\eta = 0$. Indeed, the multitask diffusion strategy (4.117) considers squared ℓ_2-norm co-regularizers to promote the smoothness of the graph signal, whereas, in the current simulation

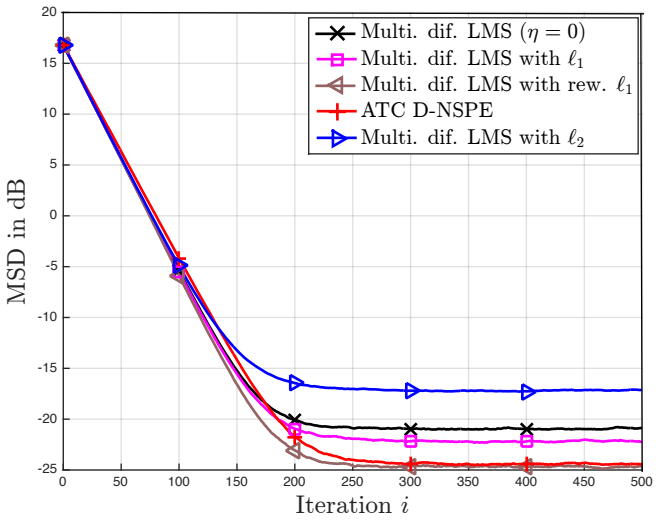


Figure 4.9: Network MSD comparison for 5 different strategies: standard diffusion without cooperation between clusters (4.114), our proximal diffusion (4.32) with ℓ_1-norm and reweighted ℓ_1-norm, the ATC D-NSPE algorithm (4.116), and the multitask diffusion strategy with squared ℓ_2-norm coregularizers (4.117).

we need to promote the sparsity of the vector $\boldsymbol{w}_k^o - \boldsymbol{w}_\ell^o$. Furthermore, when the reweighted ℓ_1-norm is used, our method is able to perform well compared to the strategy (4.116) that requires the knowledge of the indices of common and distinct entries in the parameter vectors. We note that recent unsupervised strategies [Chen et al., 2016, Plata-Chaves et al., 2016a] dealing with group of variables rather than variables propose to add a step in order to adapt the cooperation between neighboring nodes based on the group at hand. However, it is shown in [Chen et al., 2016] that the performance depends heavily on the group decomposition of the parameter vectors considered at the origin.

4.4.2 Distributed spectrum sensing

Consider a cognitive radio network composed of N_P primary users (PU) and N_S secondary users (SU). To avoid causing harmful interference to the primary users, each secondary user has to detect the frequency bands used by all primary users, even under low SNR conditions [Lorenzo et al., 2013, Sayed, 2014c, Plata-Chaves et al., 2015]. We assume that the secondary users are grouped into Q clusters and that there exists within each cluster a low power interference source (IS). The goal of each secondary user is to estimate the aggregated spectrum transmitted by all active primary users, as well as the spectrum of the interference source present in its cluster.

In order to facilitate the estimation task of the secondary users, we assume that the power spectrum of the signal transmitted by the primary user p and the interference source q can be

represented by a linear combination of N_B basis functions $\phi_m(f)$:

$$S_p(f) = \sum_{m=1}^{N_B} \alpha_{pm}\phi_m(f), \quad p = 1, \ldots, N_P, \tag{4.118}$$

$$S_q(f) = \sum_{m=1}^{N_B} \beta_{qm}\phi_m(f), \quad q = 1, \ldots, Q, \tag{4.119}$$

where α_{pm}, β_{qm} are the combination weights, and f is the normalized frequency. Each secondary user $k \in \mathcal{C}_q$ has to estimate the $N_B \times (N_P + 1)$ vector $\boldsymbol{w}_k^o = \text{col}\{\boldsymbol{\alpha}_1^o, \ldots, \boldsymbol{\alpha}_{N_P}^o, \boldsymbol{\beta}_q^o\}$ where $\boldsymbol{\alpha}_p^o = [\alpha_{p1}, \ldots, \alpha_{pN_B}]^\top$ and $\boldsymbol{\beta}_q^o = [\beta_{q1}, \ldots, \beta_{qN_B}]^\top$. Let $\ell_{p,k}(i)$ denote the path loss factor between the primary user p and the secondary user k at time i. Let also $\ell'_{q,k}(i)$ denote the path loss factor between the interference source q and the secondary user k at time i. Then, the power spectrum sensed by node $k \in \mathcal{C}_q$ at time i and frequency f_j can be expressed as follows:

$$r_{k,j}(i) = \sum_{p=1}^{N_P} \ell_{p,k}(i)S_p(f_j) + \ell'_{q,k}(i)S_q(f_j) + z_{k,j}(i), \tag{4.120}$$

where $z_{k,j}(i)$ is the sampling noise at the j-th frequency assumed to be zero-mean Gaussian with variance $\sigma^2_{z_{k,j}}$. At each time instant i, node k observes the power spectrum over N_F frequency samples. Let $\boldsymbol{r}_k(i)$ and $\boldsymbol{z}_k(i)$ be the $N_F \times 1$ vectors with j-th entries equal to $r_{k,j}(i)$ and $z_{k,j}(i)$, respectively. Using (4.120), we can establish the following linear data model for node $k \in \mathcal{C}_q$:

$$\boldsymbol{r}_k(i) = \boldsymbol{\Phi}_k(i)\boldsymbol{w}_k^o + \boldsymbol{z}_k(i), \tag{4.121}$$

where $\boldsymbol{\Phi}_k(i) \triangleq [\ell_{1,k}(i), \ldots, \ell_{N_P,k}(i), \ell'_{q,k}(i)] \otimes \boldsymbol{\Phi}$ with $\boldsymbol{\Phi}$ the $N_F \times N_B$ matrix whose j-th row contains the magnitudes of the N_B basis functions at the frequency sample f_j.

To show the effect of multitask learning with sparsity-based regularization, we consider a cognitive radio network consisting of $N_P = 2$ primary users and $N_S = 13$ secondary users decomposed into 4 clusters as shown in Figure 4.10. The power spectrum is represented by a combination of $N_B = 20$ Gaussian basis functions centered at the normalized frequency f_m with variance $\sigma_m^2 = 0.001$ for all m:

$$\phi_m(f) = \exp^{-\frac{(f-f_m)^2}{2\sigma_m^2}}, \tag{4.122}$$

where the central frequencies f_m are uniformly distributed. The combination vectors are set to:

$$\begin{aligned}
\boldsymbol{w}_{\mathcal{C}_1}^o &= [\mathbf{0}_{1\times4}\ 1\ 1\ \mathbf{0}_{1\times14},\ \mathbf{0}_{1\times14}\ 1\ 1\ \mathbf{0}_{1\times4},\ 0\ 0.3\ 0.3\ \mathbf{0}_{1\times17}]^\top \\
\boldsymbol{w}_{\mathcal{C}_2}^o &= [\mathbf{0}_{1\times4}\ 1\ 1\ \mathbf{0}_{1\times14},\ \mathbf{0}_{1\times14}\ 1\ 1\ \mathbf{0}_{1\times4},\ \mathbf{0}_{1\times20}]^\top \\
\boldsymbol{w}_{\mathcal{C}_3}^o &= [\mathbf{0}_{1\times4}\ 1\ 1\ \mathbf{0}_{1\times14},\ \mathbf{0}_{1\times14}\ 1\ 1\ \mathbf{0}_{1\times4},\ 0\ 0.3\ \mathbf{0}_{1\times16}\ 0.3\ 0]^\top \\
\boldsymbol{w}_{\mathcal{C}_4}^o &= [\mathbf{0}_{1\times4}\ 1\ 1\ \mathbf{0}_{1\times14},\ \mathbf{0}_{1\times14}\ 1\ 1\ \mathbf{0}_{1\times4},\ \mathbf{0}_{1\times17}\ 0.3\ 0.3\ 0]^\top.
\end{aligned} \tag{4.123}$$

We consider $N_F = 80$ frequency samples. Based on the free propagation theory, we set the deterministic path loss factor $\overline{\ell}_{p,k}$ to the inverse of the squared distance between the transmitter

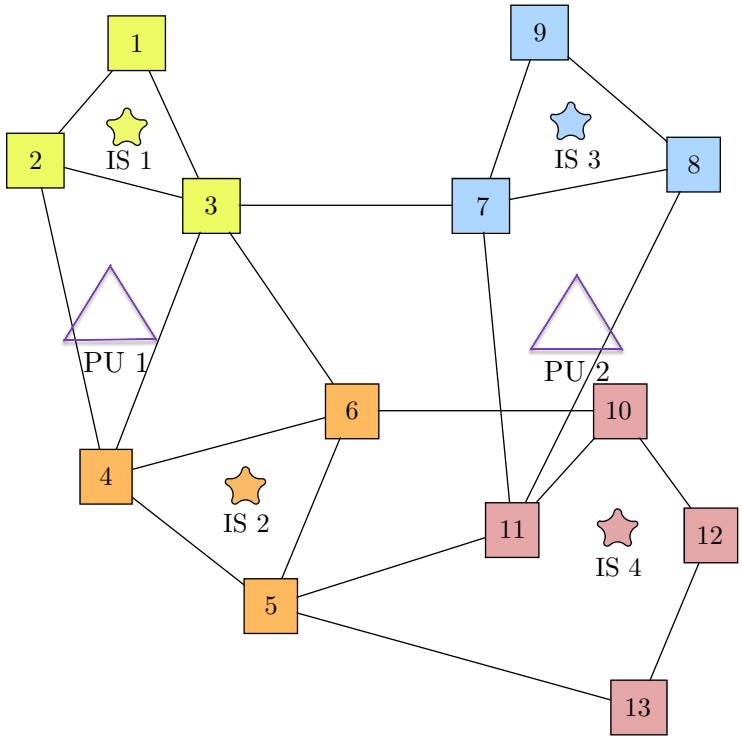


Figure 4.10: A cognitive radio network consisting of 2 primary users and 13 secondary users grouped into 4 clusters containing each an interference source IS.

p and the receiver k. At time instant i, we set $\ell_{p,k}(i) = \overline{\ell}_{p,k} + \delta\ell_{p,k}(i)$ with $\delta\ell_{p,k}(i)$ a zero-mean random Gaussian variable with standard deviation $0.1\overline{\ell}_{p,k}$. The secondary user k estimates $\ell_{p,k}(i)$ according to the following model:

$$\hat{\ell}_{p,k}(i) = \begin{cases} \overline{\ell}_{p,k}, & \text{if} \quad \ell_{p,k}(i) > \ell_0, \\ 0, & \text{otherwise} \end{cases} \tag{4.124}$$

with ℓ_0 a threshold value. The same rule is used to set the path loss factor between the interference sources and the secondary users. We run the ATC diffusion algorithm (4.32) with the following adaptation step:

$$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k \hat{\boldsymbol{\Phi}}_k^{\top}(i)[\boldsymbol{r}_k(i) - \hat{\boldsymbol{\Phi}}_k(i)\boldsymbol{w}_k(i)]. \tag{4.125}$$

The sampling noise $z_{k\ell,j}(i)$ is assumed to be a zero-mean random Gaussian variable with standard deviation 0.01. The combination coefficients $\{a_{\ell k}\}$ and regularization factors $\{\rho_{k\ell}\}$ are set in the same way as in the previous experimentation. The MSD learning curves are averaged over 50 Monte-Carlo runs.

We run the multitask diffusion LMS (4.32) in two different situations. In the first scenario, we do not allow any cooperation between clusters by setting $\eta = 0$. In the second scenario, we set the regularization strength η to 0.01 and we use the ℓ_1-norm as co-regularizing function. As it can be seen in Figure 4.11, the network MSD performance is significantly improved by cooperation

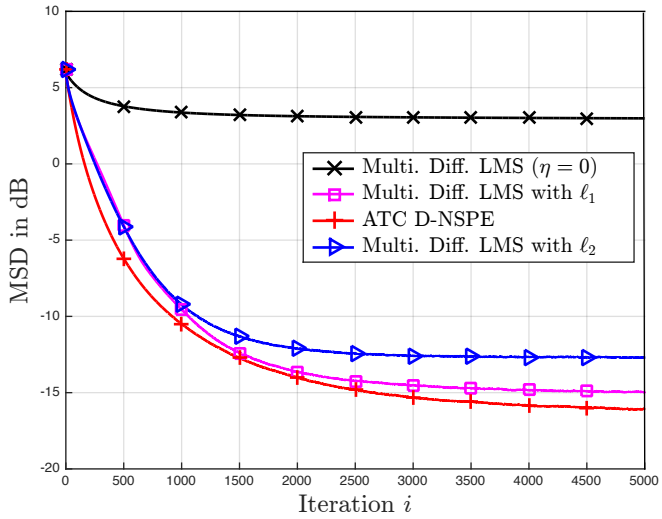


Figure 4.11: Network MSD comparison for 4 different algorithms: standard diffusion LMS without cooperation between clusters (4.114), our proximal diffusion (4.32) with ℓ_1-norm regularizer, the ATC D-NSPE algorithm (4.116), and the multitask diffusion strategy (4.117).

among clusters. For comparison purposes, we also run the ATC D-NSPE strategy (4.116) and the multitask diffusion strategy with ℓ_2-norm given by (4.117). For the ATC D-NSPE strategy we assume that nodes are aware that the first $N_P \times N_B$ components of the vector $\boldsymbol{w}_k^o$ are of global interest to the whole network and that the remaining components are of common interest to the cluster $\mathcal{C}(k)$. The link weights $\{a_{\ell k}, c_{\ell k}, \rho_{k\ell}, a_{\ell k}^{\xi}, a_{\ell k}^{\mathcal{C}(k)}\}$ are set in the same manner as the experiment in Figure 4.9. It can be observed from Figure 4.11 that our strategy performs well without the need to know the parameters of global interest and the parameters of common interest during the learning process.

Figure 4.12 shows the estimated power spectrum density (PSD) for nodes 2, 4, 7, and 13 when running the multitask diffusion strategy (4.32) with $\eta = 0$ (left) and $\eta = 0.01$ (right). In the left plot, we observe that the clusters are able to estimate their interference source. However, depending on the distance to the primary users, the secondary users do not always succeed in estimating the power spectrum transmitted by all active primary users. For example, clusters 1 and 2 are not able to estimate the power spectrum transmitted by PU2. As shown in the right plot, regardless of the distance between primary and secondary users, each secondary user is able to estimate the aggregated power spectrum transmitted by all the primary users and its own interference source by cooperating with nodes belonging to neighboring clusters.

4.5 Conclusion

In this chapter, we considered multitask learning problems over networks where the optimum parameter vectors to be estimated by neighboring clusters have a large number of similar entries and a relatively small number of distinct entries. It then becomes advantageous to develop

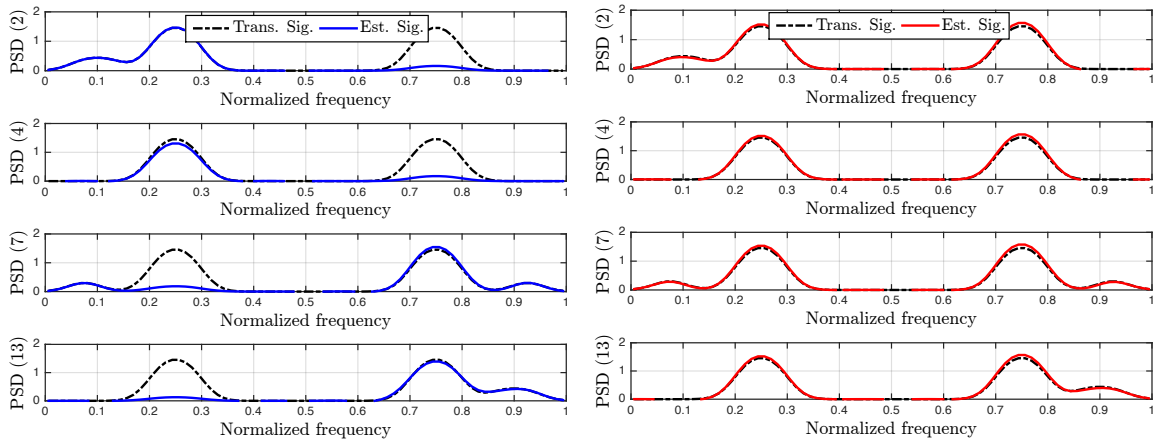


Figure 4.12: PSD estimation for nodes 2 ($\mathcal{C}_1$), 4 ($\mathcal{C}_2$), 7 ($\mathcal{C}_3$), and 13 ($\mathcal{C}_4$). (*Left*) Noncooperating clusters (4.114). (*Right*) Cooperating clusters (4.32).

distributed strategies that involve cooperation among adjacent clusters in order to exploit these similarities. A diffusion forward-backward splitting algorithm with ℓ_1-norm and reweighed ℓ_1-norm co-regularizers was proposed to address this problem. A closed-form expression for the proximal operator was derived to achieve higher efficiency. Conditions on the step-sizes to ensure convergence of the algorithm in the mean and mean-square sense were derived. Finally, simulation results were presented to illustrate the benefit of cooperating to promote similarities between estimates and an adaptive rule was established to ensure appropriate cooperation among clusters.

In Chapters 3 and 4, we considered multitask estimation problems where the agents are grouped into clusters and, within each cluster, the agents are seeking the same optimum model. We assumed that the optimum models to be estimated by neighboring clusters are close to each others in the sense of some norm. The next chapter deals with multitask estimation problems where agents are interested in estimating different but linearly related models.

CHAPTER 5

Distributed constrained LMS for multitask problems

This chapter considers distributed multitask learning problems over a network of agents where each agent is interested in estimating its own parameter vector, also called task, and where the tasks are related locally according to a set of linear equality constraints. Each agent possesses its own convex cost function of its parameter vector, and a set of linear equality constraints relating its own parameter vector to the parameter vectors of its neighboring agents. We propose an adaptive stochastic algorithm based on the projection gradient method and diffusion strategies in order to allow the network to optimize the individual costs subject to all constraints. Although the derivation is carried out for linear equality constraints, the technique can be applied to other forms of convex constraints. A detailed mean-square-error analysis of the proposed algorithm is conducted and closed-form expressions to predict its learning behavior are derived. Simulations are provided to illustrate the theoretical findings. Finally, the algorithm is employed to solve two problems in a distributed manner: a minimum-cost flow problem over a network and a space-time varying field reconstruction problem.

The work presented in this chapter appears in:

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Diffusion LMS for multitask problems with local linear equality constraints. *Submitted to IEEE Transactions on Signal Processing*, October 2016. (Also available at https://arxiv.org/abs/1610.02943)

- R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Distributed learning over multitask networks with linearly related tasks. In *Proc. Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, November 2016.

5.1 Introduction

As it became clear, there exists two main types of distributed estimation algorithms, namely, single-task algorithms and multitask algorithms. Single-task distributed estimation over networks allows to minimize the aggregate sum of convex cost functions, each available at an agent, subject to convex constraints that are also distributed across the agents. Each learner seeks to estimate the minimizer through local computations and communications among neighboring agents without the need to know any of the constraints or costs besides their own [Bertsekas, 1997, Lopes and Sayed, 2008, Chen and Sayed, 2012, Ram et al., 2010, Mota et al., 2013, Srivastava and Nedic, 2011, Lee and Nedić, 2013, Towfic and Sayed, 2014]. Multitask distributed estimation over networks is particularly well-suited for applications where several parameter vectors need to be estimated simultaneously from successive noisy measurements using in-network processing [Kekatos and Giannakis, 2013, Abdolee et al., 2014, Chen et al., 2015b, Bertrand and Moonen, 2012, Plata-Chaves et al., 2015, 2016b, Chen et al., 2014c, Nassif et al., 2016c,d, Chen et al., 2014a]. In important applications, such as network flow problems [Ahuja et al., 1993] and monitoring applications [Bertsekas and Tsitsiklis, 1989, Kekatos and Giannakis, 2013], it happens that the optimum parameter vectors to be estimated at neighboring agents are related according to a set of constraints. This observation motivates us to consider in this chapter multitask estimation problems subject to linear equality constraints of the form:

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N}{\text{minimize}} \quad J^{\text{glob}}(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}_k), \tag{5.1a}$$

$$\text{subject to} \quad \sum_{\ell\in\mathcal{I}_p} \boldsymbol{D}_{p\ell}\,\boldsymbol{w}_\ell + \boldsymbol{b}_p = \boldsymbol{0}, \quad p=1,\ldots,P. \tag{5.1b}$$

Each agent k in the network seeks to estimate its own $M_k \times 1$ parameter vector $\boldsymbol{w}_k$, and has knowledge of its cost function $J_k(\cdot)$ and the set of linear equality constraints that agent k is involved in. Each constraint is indexed by p, and defined by the $L_p \times M_\ell$ matrices $\boldsymbol{D}_{p\ell}$, the $L_p \times 1$ vector $\boldsymbol{b}_p$, and the set $\mathcal{I}_p$ of agent indices involved in this constraint. It is assumed that each agent k in $\mathcal{I}_p$ can collect estimates from all agents in $\mathcal{I}_p$ in order to satisfy the p-th constraint, i.e., $\mathcal{I}_p \subseteq \mathcal{N}_k$ if $k \in \mathcal{I}_p$ where $\mathcal{N}_k$ denotes the neighborhood of agent k. This assumption is reasonable in many applications, for instance, in remote monitoring of physical phenomena involving discretization of spatial differential equations [Bertsekas and Tsitsiklis, 1989], and in network monitoring involving conservation laws at each junction [Ahuja et al., 1993].

For illustration purposes, consider a minimum-cost flow problem over the network shown in Figure 5.1. This network consists of 10 nodes, 1 destination sink D, and 15 communication links. With each link j, we associate a directed arc and we let f_j denote the flow or traffic on this link, with $f_j > 0$ meaning that the flow is in the direction of the arc, and $f_j < 0$ otherwise. At each node k, an external source flow s_k enters and flows through the network to the destination

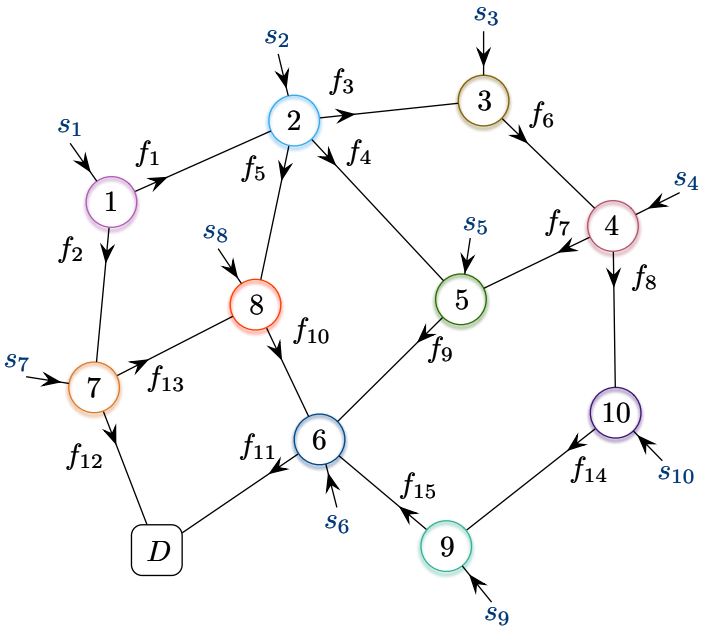


Figure 5.1: Flow network topology with 10 nodes, 1 destination sink D, and 15 communication links.

sink. The flow must satisfy a conservation equation, which states that at each node k, the sum of flows entering the node, plus the external source s_k, is equal to the sum of flows leaving node k. Given the external sources s_k and the network topology, a number of studies have been devoted to finding the optimal flows $f_j^\star$ that minimize a total flow transmission cost and satisfy the conservation equations [Ahuja et al., 1993, Boyd and Vandenberghe, 2004, Ventura, 1991]. Problems of this type arise in applications such as electrical networks, telecommunication networks, pipeline networks [Ahuja et al., 1993]. In some of these applications, it happens that node k has only access to noisy measurements $s_k(i)$ of the external source at each time instant i. Denoting by $\boldsymbol{w}_k$ the $M_k \times 1$ vector containing the flows f_j entering and leaving node k, we are interested in distributed online learning settings where each node k seeks to estimate $\boldsymbol{w}_k$ from noisy measurements $s_k(i)$ by relying only on local computations and communications with its neighbors. This problem can be recast in the form (5.1a)–(5.1b) and addressed with the multitask strategy proposed in this paper. This example will be considered further in the numerical experiments section.

We shall propose a primal technique (based on propagating and estimating the primal variable) for solving such distributed multitask estimation problems. The technique relies on combining diffusion adaptation with a stochastic gradient projection step, and on the use of constant step-sizes to enable continuous adaptation and learning from streaming data. Since we are learning from streaming data, the dual function cannot be computed exactly and the use of primal-dual methods may result in some cases in stability problems as already shown in [Towfic and Sayed, 2015]. For this reason, we shall focus on primal techniques. Our current work is able

Table 5.1: List of the main symbols and notations used in Chapter 5.

M_k	Length of the parameter vector at agent k
M_b, M_e	Length of the network block parameter vector without/with auxiliary variables
N	Number of agents in the network
N_e	Number of sub-nodes in the network
P	Number of constraints in the network
L_p	Number of linear equalities in the p-th constraint
$\mathcal{N}_k$	Neighborhood of agent k, i.e., the set of agents that are connected to k by edges
$\mathcal{I}_p,\ \mathcal{I}_{e,p}$	Set of agents/sub-nodes involved in the p-th constraint
$\boldsymbol{w}_k$	Parameter vector at agent k
$\boldsymbol{w}_{k_m}$	Parameter vector at sub-node k_m
$\boldsymbol{w}_k^o, \boldsymbol{w}_k^\star$	Optimum parameter vectors at agent k without/with constraints
$\boldsymbol{w}_b, \boldsymbol{w}_e$	Network block parameter vectors without/with auxiliary variables
$\boldsymbol{w}_b^o, \boldsymbol{w}_e^o$	Network block optimum parameter vectors without constraints and without/with auxiliary variables
$\boldsymbol{w}_b^\star, \boldsymbol{w}_e^\star$	Network block optimum parameter vectors with constraints and without/with auxiliary variables

to cope with the following two scenarios: 1) multitask problems with prior information on linear relationships between tasks, and 2) constrained multitask problems with distributed information access. We analyze the behavior of our algorithm in the mean and mean-square-error sense (w.r.t. the minimizers of the local costs and w.r.t. the solution of the constrained multitask problem) and we derive expressions to predict its transient and steady-state behavior. We show that for small constant step-sizes, the expected distance between the estimates at each agent and the optimal value can be made arbitrarily small.

Before starting our presentation, we list in Table 5.1 some of the main symbols and notations used in this chapter. Other symbols will be defined in the context where they are used.

5.2 Problem formulation and centralized solution

5.2.1 Problem formulation and assumptions

Consider a network of N agents, labeled $k = 1, \ldots, N$. At each time instant i, each agent k has access to a zero-mean real-valued observation $d_k(i)$, and a zero-mean real-valued $M_k \times 1$ regression vector $\boldsymbol{x}_k(i)$, with positive covariance matrix $\boldsymbol{R}_{x,k} = \mathbb{E}\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)\} > \boldsymbol{0}$. We assume

the data to be related via the linear data model:

$$d_k(i) = \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k^o + z_k(i), \qquad i \geq 0, \tag{5.2}$$

where $\boldsymbol{w}_k^o$ is an $M_k \times 1$ unknown parameter vector, and $z_k(i)$ is a zero-mean measurement noise of variance $\sigma_{z,k}^2$, independent of $\boldsymbol{x}_\ell(j)$ for all ℓ and j, and independent of $z_\ell(j)$ for $\ell \neq k$ or $i \neq j$. We let $\boldsymbol{r}_{dx,k} \triangleq \mathbb{E}\,\{d_k(i)\boldsymbol{x}_k(i)\}$ and $\sigma_{d,k}^2 \triangleq \mathbb{E}\,(d_k(i))^2$.

Let $\boldsymbol{w}_k$ denote some generic $M_k \times 1$ vector that is associated with agent k. The objective of agent k is to find an estimate for $\boldsymbol{w}_k^o$, and we associate with this agent the mean-square-error criterion:

$$J_k(\boldsymbol{w}_k) = \mathbb{E}\,\left(d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k\right)^2, \tag{5.3}$$

which is strongly convex, second-order differentiable, and minimized at $\boldsymbol{w}_k^o$. In addition, P linear equality constraints of the form (5.1b) are imposed on the parameter vectors $\{\boldsymbol{w}_k\}$ at each time instant i. Let us collect the parameter vectors $\{\boldsymbol{w}_k\}$ and $\{\boldsymbol{w}_k^o\}$ from across the network into $N \times 1$ block column vectors $\boldsymbol{w}_b$ and $\boldsymbol{w}_b^o$, respectively:

$$\boldsymbol{w}_b \triangleq \mathrm{col}\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N\}, \qquad \boldsymbol{w}_b^o \triangleq \mathrm{col}\{\boldsymbol{w}_1^o, \ldots, \boldsymbol{w}_N^o\}, \tag{5.4}$$

and let us write the P linear equality constraints in (5.1b) more compactly as:

$$\boldsymbol{\mathcal{D}}\boldsymbol{w}_b + \boldsymbol{b} = \boldsymbol{0}, \tag{5.5}$$

where $\boldsymbol{\mathcal{D}}$ is a $P \times N$ block matrix, with each block $\boldsymbol{D}_{p\ell}$ having dimensions $L_p \times M_\ell$, and $\boldsymbol{b}$ is a $P \times 1$ block column vector where each block $\boldsymbol{b}_p$ has dimensions $L_p \times 1$. Combining (5.5) and (5.3), the network optimization problem becomes:

$$\begin{aligned} \underset{\boldsymbol{w}_b}{\text{minimize}} \quad & \sum_{k=1}^{N} \mathbb{E}\,\left(d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k\right)^2, \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w}_b + \boldsymbol{b} = \boldsymbol{0}, \end{aligned} \tag{5.6}$$

where each agent k is in charge of estimating the k-th sub-vector $\boldsymbol{w}_k$ of $\boldsymbol{w}_b$. Since the mean-square-error criterion in (5.6) is separable, we shall assume without loss of generality that each parameter vector $\boldsymbol{w}_k$ is involved in at least one constraint so that cooperation is justified. We shall also assume that $\boldsymbol{\mathcal{D}}$ is full row-rank to ensure that equation $\boldsymbol{\mathcal{D}}\boldsymbol{w}_b + \boldsymbol{b} = \boldsymbol{0}$ has at least one solution. We also introduce an assumption on the availability of the constraints. Let $\mathcal{I}_p$ be the set of agent indices involved in the p-th constraint. We shall assume that every agent k in $\mathcal{I}_p$ is aware of the p-th constraint, and that the network topology permits this agent to collect estimates from all agents in $\mathcal{I}_p$, that is, $\mathcal{I}_p \subseteq \mathcal{N}_k$, so it can apply this constraint to its own estimate. This assumption is reasonable in many applications, for instance, in remote monitoring of physical phenomena [Bertsekas and Tsitsiklis, 1989], and in network distribution system monitoring (as described in the introduction). These examples will be considered in numerical experiments section.

Before proceeding, note that problem (5.6) can be recast as a quadratic program (QP) [Boyd and Vandenberghe, 2004], and any algorithm that solves QPs can solve it. We are interested instead in distributed adaptive solutions that can operate in real-time on streaming data. As we will see later, the traditional constrained LMS algorithm [Frost III, 1972] can solve (5.6) in a centralized manner. In this centralized solution, each agent at each iteration sends its data to a fusion center, which in turn processes the data and sends the estimates back to the agents. The entire matrix $\boldsymbol{\mathcal{D}}$ and the entire vector $\boldsymbol{b}$ then need to be available at the fusion center. While centralized solutions can be powerful, decentralized solutions are more attractive since they are more robust, require less communication costs, and respect the privacy policy of each agent.

5.2.2 Centralized solution

Let us first describe the centralized solution. We assume that the agents transmit the collected data $\{d_k(i), \boldsymbol{x}_k(i)\}$ to a fusion center for processing. Problem (5.6) can be written equivalently as:

$$\begin{aligned} \underset{\boldsymbol{w}_b}{\text{minimize}} \quad & \boldsymbol{w}_b^\top \boldsymbol{\mathcal{R}}_x \boldsymbol{w}_b - 2\boldsymbol{r}_{dx}^\top \boldsymbol{w}_b + \boldsymbol{r}_d^\top \mathbb{1}, \\ \text{subject to} \quad & \boldsymbol{\mathcal{D}}\boldsymbol{w}_b + \boldsymbol{b} = \boldsymbol{0}, \end{aligned} \tag{5.7}$$

where the $N \times N$ block diagonal matrix $\boldsymbol{\mathcal{R}}_x$, the $N \times 1$ block column vector $\boldsymbol{r}_{dx}$, and the $N \times 1$ column vector $\boldsymbol{r}_d$ are given by:

$$\boldsymbol{\mathcal{R}}_x \triangleq \operatorname{diag}\{\boldsymbol{R}_{x,1}, \ldots, \boldsymbol{R}_{x,N}\}, \tag{5.8}$$

$$\boldsymbol{r}_{dx} \triangleq \operatorname{col}\{\boldsymbol{r}_{dx,1}, \ldots, \boldsymbol{r}_{dx,N}\}, \tag{5.9}$$

$$\boldsymbol{r}_d \triangleq \operatorname{col}\{\sigma_{d,1}^2, \ldots, \sigma_{d,N}^2\}. \tag{5.10}$$

Since $\boldsymbol{\mathcal{R}}_x$ is positive definite, problem (5.7) is a positive definite quadratic program with equality constraints. It has a unique global minimum given by:

$$\boldsymbol{w}_b^\star = \boldsymbol{w}_b^o - \boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^\top \left(\boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{R}}_x^{-1}\boldsymbol{\mathcal{D}}^\top\right)^{-1} \left(\boldsymbol{\mathcal{D}}\boldsymbol{w}_b^o + \boldsymbol{b}\right). \tag{5.11}$$

Let Ω denote the linear manifold:

$$\Omega \triangleq \{\boldsymbol{w}_b \colon \boldsymbol{\mathcal{D}}\boldsymbol{w}_b + \boldsymbol{b} = \boldsymbol{0}\}. \tag{5.12}$$

If $\boldsymbol{w}_b^o \in \Omega$, the optimum $\boldsymbol{w}_b^\star$ coincides with $\boldsymbol{w}_b^o$. In this case, the constrained optimization problem (5.6) can be thought as estimating the unknown parameter vectors $\boldsymbol{w}_k^o$ given prior information about relationships between tasks of the form (5.1b). Exploiting such prior information may improve the estimation as we will see in the experiments. Let M_b denote the dimension of the network block parameter vector $\boldsymbol{w}_b$, i.e., $M_b = \sum_{k=1}^N M_k$. The projection of any vector $\boldsymbol{z} \in \mathbb{R}^{M_b}$ onto Ω is given by:

$$\mathrm{P}_\Omega(\boldsymbol{z}) = \boldsymbol{\mathcal{P}}\boldsymbol{z} - \boldsymbol{f}, \tag{5.13}$$

where

$$\boldsymbol{\mathcal{P}} \triangleq \boldsymbol{I}_{M_b} - \boldsymbol{\mathcal{D}}^{\dagger}\boldsymbol{\mathcal{D}}, \qquad \boldsymbol{f} \triangleq \boldsymbol{\mathcal{D}}^{\dagger}\boldsymbol{b}, \tag{5.14}$$

with $\boldsymbol{\mathcal{D}}^{\dagger}$ denoting the pseudo-inverse of the matrix $\boldsymbol{\mathcal{D}}$ given by $\boldsymbol{\mathcal{D}}^{\top}\left(\boldsymbol{\mathcal{D}}\boldsymbol{\mathcal{D}}^{\top}\right)^{-1}$. Let $\boldsymbol{w}_b(i)$ denotes the estimate of $\boldsymbol{w}_b^{\star}$ at iteration i. In order to solve (5.7) iteratively, the gradient projection method [Bertsekas, 1999] can be applied on top of a gradient-descent iteration:

$$\boldsymbol{w}_b(i+1) = \mathrm{P}_{\Omega}\left(\boldsymbol{w}_b(i) + \mu\left(\boldsymbol{r}_{dx} - \boldsymbol{\mathcal{R}}_x\boldsymbol{w}_b(i)\right)\right). \tag{5.15}$$

In order to run recursion (5.15), we need to have access to the second-order moments $\{\boldsymbol{R}_{x,k}, \boldsymbol{r}_{dx,k}\}$. Since these moments are rarely available beforehand, the agents use their instantaneous data $\{d_k(i), \boldsymbol{x}_k(i)\}$ to approximate these moments, namely, $\boldsymbol{R}_{x,k} \approx \boldsymbol{x}_k(i)\boldsymbol{x}_k^{\top}(i)$ and $\boldsymbol{r}_{dx,k} \approx d_k(i)\boldsymbol{x}_k(i)$. Doing so and replacing $\mathrm{P}_{\Omega}(\cdot)$ by (5.13), we obtain the following stochastic-gradient algorithm in lieu of (5.15):

$$\boldsymbol{w}_b(i+1) = \boldsymbol{\mathcal{P}} \cdot \mathrm{col}\left\{\boldsymbol{w}_k(i) + \mu\boldsymbol{x}_k(i)\left(d_k(i) - \boldsymbol{x}_k^{\top}(i)\boldsymbol{w}_k(i)\right)\right\}_{k=1}^{N} - \boldsymbol{f}, \tag{5.16}$$

where $\mathrm{col}\{\boldsymbol{a}_k\}_{k=1}^{N}$ refers to the block column vector $\mathrm{col}\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N\}$. Collecting the regression vectors into the $M_b \times N$ matrix $\boldsymbol{X}(i) \triangleq \mathrm{diag}\{\boldsymbol{x}_1(i), \ldots, \boldsymbol{x}_N(i)\}$ and the observations into the $N \times 1$ vector $\boldsymbol{d}(i) \triangleq \mathrm{col}\{d_1(i), \ldots, d_N(i)\}$, algorithm (5.16) becomes the constrained least-mean-squares (CLMS) algorithm:

$$\boldsymbol{w}_b(i+1) = \boldsymbol{\mathcal{P}}\left(\boldsymbol{w}_b(i) + \mu\boldsymbol{X}(i)\left(\boldsymbol{d}(i) - \boldsymbol{X}^{\top}(i)\boldsymbol{w}_b(i)\right)\right) - \boldsymbol{f}. \tag{5.17}$$

This procedure was originally proposed in [Frost III, 1972] as an online linearly constrained minimum variance (LCMV) filter for solving mean-square-error estimation problems subject to linear constraints; the motivation there was not concerned with multi-task problems. In this section, we showed that the centralized multi-task constrained problem reduces to a similar problem, for which algorithm (5.17) can be applied. The performance of such stand-alone centralized solutions was studied in [Frost III, 1972, Sayed, 2008].

5.3 Problem reformulation and Distributed solution

5.3.1 Problem reformulation

We move on to develop a distributed solution with a continuous adaptation mechanism. First, note that several works for solving problems of the form (5.6) with possible distributed information access already exist in the literature [Ram et al., 2010, Mota et al., 2012, 2013, Chen et al., 2015b, Towfic and Sayed, 2014, Lee and Nedić, 2013, Yuan et al., 2016]. However, except for the work dealing with distributed dictionary learning [Chen et al., 2015b], these other works solve single-task estimation problems where the entire network is employed to estimate the minimizer of (5.6). Furthermore, in comparison to [Mota et al., 2012, 2013, Chen et al., 2015b, Yuan et al., 2016], we shall assume stochastic errors in the evaluation of the gradients of local cost functions.

To proceed with the analysis, one of the challenges we now face is that any given agent k may be involved in several constraints. Our strategy is to transform (5.6) into an equivalent optimization problem exhibiting structure amenable to distributed optimization with separable constraints. Let j_k denote the number of constraints that agent k is involved in. We expand each node k into a cluster $\mathcal{C}_k$ of j_k *virtual* sub-nodes, namely, $\mathcal{C}_k \triangleq \{k_m\}_{m=1}^{j_k}$. Each one of these sub-nodes is involved in a single constraint. Let $\boldsymbol{w}_{k_m}$ denote the $M_k \times 1$ auxiliary vector associated with sub-node k_m. In order to ensure that agent k satisfies simultaneously all the constraints at convergence, we will allow all sub-nodes at agent k to run diffusion learning to reach agreement on their estimates $\{\boldsymbol{w}_{k_m}\}$ asymptotically. We denote by $\mathcal{I}_{e,p}$ the set of sub-nodes which are involved in the p-th constraint.

In order to clarify the presentation, an illustrative example is provided in Figure 5.2. On the left of this panel is the original network topology with $N = 6$ agents and $P = 3$ constraints. On the right is the network topology model with clusters of sub-nodes shown in grey color. Observe that $\mathcal{I}_2 = \{1, 3, k\}$ and $\mathcal{I}_3 = \{4, k, \ell\}$, which means that agent k is involved in constraints 2 and 3. Agent k is thus expanded into a cluster $\mathcal{C}_k = \{k_1, k_2\}$ of 2 sub-nodes. Sub-nodes k_1 and k_2 are assigned to constraints 2 and 3, respectively. Each other agent, say ℓ, involved in a single constraint is renamed ℓ_1 and assigned to a single-node cluster $\mathcal{C}_\ell = \{\ell_1\}$ for consistency of notation. This leads to the sets $\mathcal{I}_{e,2} = \{1_2, 3_1, k_1\}$ and $\mathcal{I}_{e,3} = \{4_1, \ell_1, k_2\}$ where all sub-nodes are involved in a single constraint. Finally, as handled by the algorithm introduced in the sequel, note that in the network topology model, only sub-nodes involved in a common constraint will share data (see the communication link connecting agents 2 and k). However, all sub-nodes k_m in a cluster $\mathcal{C}_k$ are not affected by this rule, i.e., they can share data, since they refer to the same agent k.

Accordingly, we can now reformulate problem (5.6). We start by collecting the vectors $\boldsymbol{w}_{k_m}$ into the $N_e \times 1$ network block column vector:

$$\boldsymbol{w}_e \triangleq \operatorname{col}\left\{\operatorname{col}\{\boldsymbol{w}_{k_m}\}_{m=1}^{j_k}\right\}_{k=1}^{N}, \tag{5.18}$$

where $N_e \triangleq \sum_{k=1}^{N} j_k$. We introduce for each agent k a set of j_k coefficients $\{c_{k_m}\}$ that satisfy two conditions:

$$c_{k_m} > 0, \text{ for } m = 1, \ldots, j_k, \quad \text{and} \quad \sum_{m=1}^{j_k} c_{k_m} = 1. \tag{5.19}$$

The coefficients $\{c_{k_m}\}$ are free parameters that are chosen by the user. A natural choice is $c_{k_m} = \frac{1}{j_k}$ for all m. The global cost in (5.1a) can be written as:

$$J^{\text{glob}}(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} J_k(\boldsymbol{w}_k) = \sum_{k=1}^{N} \sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_k). \tag{5.20}$$

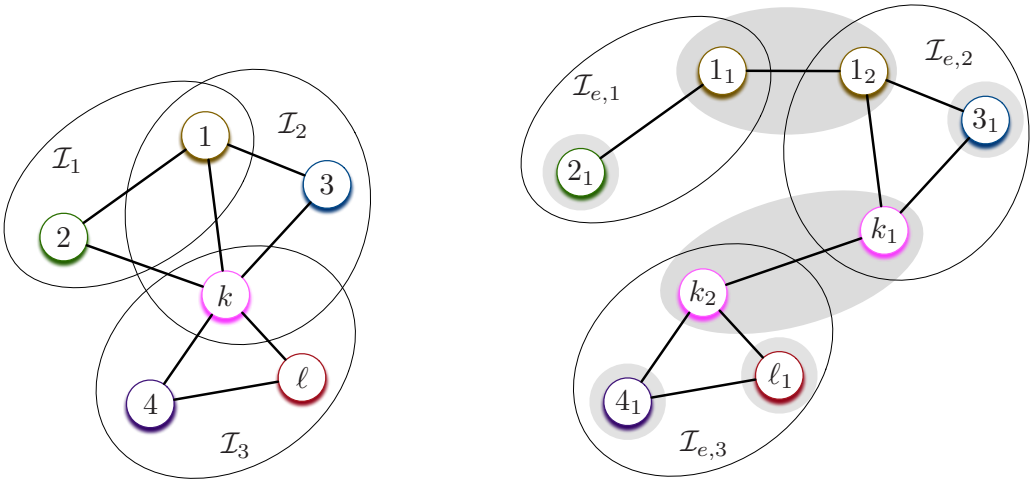


Figure 5.2: (*Left*) Network topology with constraints identified by the subsets of nodes $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{I}_3$. (*Right*) Network topology model with clusters shown in grey color and constraints now identified by the subsets of sub-nodes $\mathcal{I}_{e,1}$, $\mathcal{I}_{e,2}$, and $\mathcal{I}_{e,3}$. All sub-nodes in this model are involved in a single constraint. Diffusion learning is run in clusters with more than one sub-node to reach agreement on local estimates while satisfying their respective constraints.

We reformulate problem (5.1) in the following equivalent form by introducing the auxiliary variables $\{\boldsymbol{w}_{k_m}\}$:

$$\underset{\boldsymbol{w}_e}{\text{minimize}} \quad \sum_{k=1}^{N}\sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_{k_m}) \tag{5.21a}$$

$$\text{subject to} \quad \sum_{\ell_n \in \mathcal{I}_{e,p}} \boldsymbol{D}_{p\ell_n}\boldsymbol{w}_{\ell_n} + \boldsymbol{b}_p = \boldsymbol{0}, \quad p = 1,\ldots,P, \tag{5.21b}$$

$$\boldsymbol{w}_{k_1} = \ldots = \boldsymbol{w}_{k_{j_k}}, \qquad k = 1\ldots,N. \tag{5.21c}$$

In the following, we shall address the equality constraints (5.21c) with a diffusion algorithm within each cluster of sub-nodes with the objective of reaching an agreement within each cluster (all sub-nodes converge to the same estimate). Since the diffusion strategy in a single-task network allows the agents to converge to the same limit point asymptotically for sufficiently small constant step-sizes when the network is strongly connected [Sayed, 2014b], we allow the sub-nodes in cluster $\mathcal{C}_k$ to be connected such that the resultant cluster $\mathcal{C}_k$ is strongly connected. This does not lead to a change in the network topology since each sub-node in a cluster refers to the same agent. We refer to the *virtual* set of neighboring sub-nodes of k_m in $\mathcal{C}_k$ by $\mathcal{N}_{k_m} \cap \mathcal{C}_k$.

The cost function in (5.21a) can be written as:

$$\sum_{k=1}^{N}\sum_{m=1}^{j_k} c_{k_m} J_k(\boldsymbol{w}_{k_m}) = \boldsymbol{w}_e^{\top}\boldsymbol{\mathcal{R}}_{x,e}\boldsymbol{w}_e - 2\boldsymbol{r}_{dx,e}^{\top}\boldsymbol{w}_e + \boldsymbol{r}_{d,e}^{\top}\mathbb{1}, \tag{5.22}$$

where the $N_e \times N_e$ block diagonal matrix $\boldsymbol{\mathcal{R}}_{x,e}$, the $N_e \times 1$ block column vector $\boldsymbol{r}_{dx,e}$, and the

$N_e \times 1$ column vector $\boldsymbol{r}_{d,e}$ are given by:

$$\boldsymbol{\mathcal{R}}_{x,e} \triangleq \operatorname{diag}\{\boldsymbol{C}_1 \otimes \boldsymbol{R}_{x,1}, \ldots, \boldsymbol{C}_N \otimes \boldsymbol{R}_{x,N}\}, \tag{5.23}$$

$$\boldsymbol{r}_{dx,e} \triangleq \operatorname{col}\{\boldsymbol{c}_1 \otimes \boldsymbol{r}_{dx,1}, \ldots, \boldsymbol{c}_N \otimes \boldsymbol{r}_{dx,N}\}, \tag{5.24}$$

$$\boldsymbol{r}_{d,e} \triangleq \operatorname{col}\left\{\sigma_{d,1}^2 \boldsymbol{c}_1, \ldots, \sigma_{d,N}^2 \boldsymbol{c}_N\right\}, \tag{5.25}$$

with

$$\boldsymbol{C}_k \triangleq \operatorname{diag}\left\{c_{k_1}, \ldots, c_{k_{j_k}}\right\} \quad \text{and} \quad \boldsymbol{c}_k \triangleq \operatorname{col}\left\{c_{k_1}, \ldots, c_{k_{j_k}}\right\}. \tag{5.26}$$

The equality constraints in (5.21b)–(5.21c) can be written more compactly as:

$$\boldsymbol{\mathcal{D}}_e' \boldsymbol{w}_e + \boldsymbol{b}' = \boldsymbol{0} \tag{5.27}$$

with

$$\boldsymbol{\mathcal{D}}_e' = \begin{bmatrix} \boldsymbol{\mathcal{D}}_e \\ \boldsymbol{\mathcal{H}} \end{bmatrix}, \quad \boldsymbol{b}' = \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{0} \end{bmatrix}, \tag{5.28}$$

where $\boldsymbol{\mathcal{D}}_e$ is a $P \times N_e$ block matrix constructed according to (5.21b) which can be viewed as an expanded form of the $P \times N$ block matrix $\boldsymbol{\mathcal{D}}$, and $\boldsymbol{\mathcal{H}}$ is a $\sum_{k=1}^{N}(j_k - 1) \times N_e$ block matrix constructed according to (5.21c).

Using similar arguments as in Subsection 5.2.2, we find that the solution of (5.21) is given by:

$$\boldsymbol{w}_e^\star = \boldsymbol{w}_e^o - \boldsymbol{\mathcal{R}}_{x,e}^{-1} \boldsymbol{\mathcal{D}}_e'^\top (\boldsymbol{\mathcal{D}}_e' \boldsymbol{\mathcal{R}}_{x,e}^{-1} \boldsymbol{\mathcal{D}}_e'^\top)^{-1} (\boldsymbol{\mathcal{D}}_e' \boldsymbol{w}_e^o + \boldsymbol{b}'), \tag{5.29}$$

where the $N_e \times 1$ block column vector $\boldsymbol{w}_e^o$ is given by:

$$\boldsymbol{w}_e^o \triangleq \operatorname{col}\{\mathbb{1}_{j_1 \times 1} \otimes \boldsymbol{w}_1^o, \ldots, \mathbb{1}_{j_N \times 1} \otimes \boldsymbol{w}_N^o\}. \tag{5.30}$$

Let $\boldsymbol{w}_k^\star$ denote the k-th block of $\boldsymbol{w}_b^\star$ in (5.11). The optimum vector $\boldsymbol{w}_e^\star$ can be written alternatively as:

$$\boldsymbol{w}_e^\star = \operatorname{col}\{\mathbb{1}_{j_1 \times 1} \otimes \boldsymbol{w}_1^\star, \ldots, \mathbb{1}_{j_N \times 1} \otimes \boldsymbol{w}_N^\star\}. \tag{5.31}$$

5.3.2 Distributed solution

To solve problem (5.21) with distributed information access, we propose an iterative algorithm based on diffusion strategies and gradient-projection principle. First, we present the algorithm when the second order moments of the observations are assumed to be known by each sub-node. Although cluster $\mathcal{C}_k$ and agent k refer to the same entity, we shall use the notion of cluster and sub-nodes in order to simplify the presentation.

Let $\boldsymbol{w}_{e,p}$ denote the $i_p \times 1$ block column vector given by $\boldsymbol{w}_{e,p} = \operatorname{col}\{\boldsymbol{w}_{\ell_n}\}_{\ell_n \in \mathcal{I}_{e,p}}$ where i_p is the number of nodes involved in the p-th constraint. Also, note that i_p is the cardinality of $\mathcal{I}_p$ and $\mathcal{I}_{e,p}$. Let Ω_p denote the linear manifold corresponding to the p-th constraint in (5.21b), namely, $\Omega_p \triangleq \{\boldsymbol{\mathcal{D}}_p \boldsymbol{w}_{e,p} + \boldsymbol{b}_p = \boldsymbol{0}\}$ where $\boldsymbol{\mathcal{D}}_p$ is a $1 \times i_p$ block matrix.

Let $\boldsymbol{w}_{k_m}(i)$ be the estimate of $\boldsymbol{w}_k^\star$ at sub-node k_m and time instant i. We assume that $k_m \in \mathcal{I}_{e,p}$. Based on the gradient projection algorithm, following the same line of reasoning as [Sayed, 2014c] in the single-task case, and extending the argument to our multitask problem as in the previous chapter, we arrive at the following diffusion algorithm consisting of three steps:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m}\left(\boldsymbol{r}_{dx,k} - \boldsymbol{R}_{x,k}\boldsymbol{w}_{k_m}(i)\right) \tag{5.32a}$$

$$\boldsymbol{\phi}_{k_m}(i+1) = \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m}\boldsymbol{\psi}_{k_n}(i+1) \tag{5.32b}$$

$$\boldsymbol{w}_{k_m}(i+1) = \left[\mathrm{P}_{\Omega_p}\left(\mathrm{col}\{\boldsymbol{\phi}_{\ell_n}(i+1)\}_{\ell_n \in \mathcal{I}_{e,p}}\right)\right]_{k_m} \tag{5.32c}$$

where $\mu > 0$ is a constant step-size parameter, $[\boldsymbol{x}]_{k_m}$ is the block of $\boldsymbol{x}$ corresponding to sub-node k_m, and $\boldsymbol{w}_{k_m}(0) = \boldsymbol{w}_k(0)$ for all m. In the first step (5.32a), also called adaptation step, each sub-node k_m in the network adapts its estimate $\boldsymbol{w}_{k_m}(i)$ via gradient descent on $c_{k_m}J_k(\cdot)$. This step results in the intermediate estimate $\boldsymbol{\psi}_{k_m}(i+1)$.

In the combination step (5.32b), each sub-node k_m combines its estimate $\boldsymbol{\psi}_{k_m}(i+1)$ with the estimates $\boldsymbol{\psi}_{k_n}(i+1)$ of its intra-cluster neighbors $\mathcal{N}_{k_m} \cap \mathcal{C}_k$. This step results in the intermediate estimate $\boldsymbol{\phi}_{k_m}(i+1)$. The nonnegative coefficients $\{a_{k_n,k_m}\}$ are chosen to satisfy the following conditions:

$$a_{k_n,k_m} \geq 0, \quad \sum_{k_m \in \mathcal{N}_{k_n} \cap \mathcal{C}_k} a_{k_n,k_m} = 1, \quad \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m} = 1, \quad \text{and } a_{k_n,k_m} = 0 \text{ if } k_n \notin \mathcal{N}_{k_m} \cap \mathcal{C}_k. \tag{5.33}$$

Collecting these coefficients into a $j_k \times j_k$ matrix $\boldsymbol{A}_k$ for each cluster $\mathcal{C}_k$, it follows that $\boldsymbol{A}_k$ is doubly stochastic.

Let M_p denote the dimension of the vector $\boldsymbol{w}_{e,p}$, i.e., $M_p = \sum_{\ell_n \in \mathcal{I}_{e,p}} M_\ell$. Before describing the third step, we recall that the projection of any point $\boldsymbol{z}$ onto Ω_p has the form:

$$\mathrm{P}_{\Omega_p}(\boldsymbol{z}) = \boldsymbol{\mathcal{P}}_p\,\boldsymbol{z} - \boldsymbol{f}_p, \tag{5.34}$$

where

$$\boldsymbol{\mathcal{P}}_p \triangleq \boldsymbol{I}_{M_p} - \boldsymbol{\mathcal{D}}_p^\dagger\boldsymbol{\mathcal{D}}_p \qquad \text{and} \qquad \boldsymbol{f}_p \triangleq \boldsymbol{\mathcal{D}}_p^\dagger\boldsymbol{b}_p. \tag{5.35}$$

To evaluate the block $\left[\mathrm{P}_{\Omega_p}(\boldsymbol{z})\right]_{k_m}$, even if sub-node k_m is only in charge of estimating $\boldsymbol{w}_{k_m}$, it needs the entire vector $\boldsymbol{z}$, the $M_k \times M_p$ matrix $[\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet}$, and the $M_k \times 1$ vector $[\boldsymbol{f}_p]_{k_m}$. In the projection step (5.32c), each sub-node $k_m \in \mathcal{I}_{e,p}$ collects the intermediate estimates $\boldsymbol{\phi}_{\ell_n}(i+1)$ from all sub-nodes $\ell_n \in \mathcal{I}_{e,p}$ and combines them according to (5.32c). This step results in the estimate $\boldsymbol{w}_{k_m}(i+1)$ of $\boldsymbol{w}_k^\star$ at sub-node k_m and iteration $i+1$.

The adaptation step (5.32a) requires knowledge of the second-order moments of data. Proceeding as in the centralized case, and replacing the moments by instantaneous approximations, we obtain algorithm (5.36) for solving problem (5.21) in a distributed manner.

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m} \boldsymbol{x}_k(i) \Big(d_k(i) - \boldsymbol{x}_k^{\top}(i) \boldsymbol{w}_{k_m}(i) \Big) \tag{5.36a}$$

$$\boldsymbol{\phi}_{k_m}(i+1) = [\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet} \cdot \operatorname{col}\big\{ \boldsymbol{\psi}_{\ell_n}(i+1) \big\}_{\ell_n \in \mathcal{I}_{e,p}} - [\boldsymbol{f}_p]_{k_m} \tag{5.36b}$$

$$\boldsymbol{w}_{k_m}(i+1) = \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m} \boldsymbol{\phi}_{k_n}(i+1) \tag{5.36c}$$

Compared to (5.32), observe in algorithm (5.36) that each sub-node k_m projects its intermediate estimate before combining it. We recommend this permutation since it allows, with the appropriate parameter settings described below, to reduce the algorithm computational complexity without compromising its convergence, as confirmed in the sequel. Consider any agent k. By setting factors c_{k_m} to $\frac{1}{j_k}$ for all $m = 1, \ldots, j_k$, and combining the intermediate estimate $\boldsymbol{\phi}_{k_m}(i+1)$ at each sub-node k_m with the estimates of all other sub-nodes available at node k using uniform combination coefficients, i.e., $\mathcal{N}_{k_m} \cap \mathcal{C}_k = \mathcal{C}_k$ and $a_{k_n,k_m} = \frac{1}{j_k}$ for $n = 1, \ldots, j_k$, steps (5.36a) and (5.36c) reduce to:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{\psi}_k(i+1), \qquad \text{for } m = 1, \ldots, j_k, \tag{5.37}$$

$$\boldsymbol{w}_{k_m}(i+1) = \boldsymbol{w}_k(i+1), \qquad \text{for } m = 1, \ldots, j_k, \tag{5.38}$$

where $\boldsymbol{\psi}_k(i+1)$ and $\boldsymbol{w}_k(i+1)$ are given by:

$$\boldsymbol{\psi}_k(i+1) = \boldsymbol{w}_k(i) + \frac{\mu}{j_k} \boldsymbol{x}_k(i) \Big(d_k(i) - \boldsymbol{x}_k^{\top}(i) \boldsymbol{w}_k(i) \Big), \tag{5.39}$$

$$\boldsymbol{w}_k(i+1) = \frac{1}{j_k} \sum_{n=1}^{j_k} \boldsymbol{\phi}_{k_n}(i+1). \tag{5.40}$$

5.4 Stochastic performance analysis

5.4.1 Weight error vector recursion

We shall study the stochastic behavior of algorithm (5.36) with respect to the optimal parameter vector $\boldsymbol{w}_e^o$ and with respect to the solution $\boldsymbol{w}_e^\star$ of the optimization problem with constraints (5.21). To this end, we introduce for each sub-node k_m the weight error vectors:

$$\widetilde{\boldsymbol{w}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{w}_{k_m}(i), \qquad \widetilde{\boldsymbol{w}}'_{k_m}(i) \triangleq \boldsymbol{w}_k^\star - \boldsymbol{w}_{k_m}(i), \tag{5.41}$$

and the intermediate error vectors:

$$\widetilde{\boldsymbol{\psi}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\psi}_{k_m}(i), \qquad \widetilde{\boldsymbol{\phi}}_{k_m}(i) \triangleq \boldsymbol{w}_k^o - \boldsymbol{\phi}_{k_m}(i). \tag{5.42}$$

We further introduce the $N_e \times 1$ block network error vectors:

$$\widetilde{\boldsymbol{w}}_e(i) \triangleq \operatorname{col}\Big\{ \operatorname{col}\big\{ \widetilde{\boldsymbol{w}}_{k_m}(i) \big\}_{m=1}^{j_k} \Big\}_{k=1}^{N}, \tag{5.43}$$

$$\widetilde{\boldsymbol{w}}'_e(i) \triangleq \operatorname{col}\Big\{ \operatorname{col}\big\{ \widetilde{\boldsymbol{w}}'_{k_m}(i) \big\}_{m=1}^{j_k} \Big\}_{k=1}^{N}. \tag{5.44}$$

Note that the behavior of algorithm (5.36) with respect to $\boldsymbol{w}_e^\star$ can be deduced from its behavior with respect to $\boldsymbol{w}_e^o$ using the following relation:

$$\widetilde{\boldsymbol{w}}_e'(i+1) = \widetilde{\boldsymbol{w}}_e(i+1) - \boldsymbol{w}_e^\delta, \tag{5.45}$$

where

$$\boldsymbol{w}_e^\delta \triangleq \boldsymbol{w}_e^o - \boldsymbol{w}_e^\star. \tag{5.46}$$

Thus, in the sequel, we first study the behavior of algorithm (5.36) with respect to $\boldsymbol{w}_e^o$ and, next, we characterize its behavior with respect to $\boldsymbol{w}_e^\star$ using relation (5.45).

Let M_e denote the length of the network error vector $\widetilde{\boldsymbol{w}}_e(i)$, that is, $M_e \triangleq \sum_{k=1}^N j_k M_k$. Using the linear model (5.2), the estimation error in the adaptation step (5.36a) can be written as:

$$d_k(i) - \boldsymbol{x}_k^\top(i)\boldsymbol{w}_{k_m}(i) = \boldsymbol{x}_k^\top(i)\widetilde{\boldsymbol{w}}_{k_m}(i) + z_k(i). \tag{5.47}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the adaptation step (5.36a), using (5.47), and collecting the error vectors $\widetilde{\boldsymbol{\psi}}_{k_m}(i)$ into the $N_e \times 1$ block vector $\widetilde{\boldsymbol{\psi}}_e(i) \triangleq \operatorname{col}\left\{\operatorname{col}\{\widetilde{\boldsymbol{\psi}}_{k_m}(i)\}_{m=1}^{j_k}\right\}_{k=1}^N$, we obtain:

$$\widetilde{\boldsymbol{\psi}}_e(i+1) = \left[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\right]\widetilde{\boldsymbol{w}}_e(i) - \mu\,\boldsymbol{p}_{xz,e}(i), \tag{5.48}$$

where

$$\boldsymbol{\mathcal{R}}_{x,e}(i) \triangleq \operatorname{diag}\left\{\boldsymbol{C}_1 \otimes \boldsymbol{x}_1(i)\boldsymbol{x}_1^\top(i), \ldots, \boldsymbol{C}_N \otimes \boldsymbol{x}_N(i)\boldsymbol{x}_N^\top(i)\right\}, \tag{5.49}$$

$$\boldsymbol{p}_{xz,e}(i) \triangleq \operatorname{col}\left\{\boldsymbol{c}_1 \otimes \boldsymbol{x}_1(i)z_1(i), \ldots, \boldsymbol{c}_N \otimes \boldsymbol{x}_N(i)z_N(i)\right\}. \tag{5.50}$$

Projecting $\boldsymbol{\psi}_e(i+1)$ onto the sets Ω_p in (5.34), we obtain from (5.36b):

$$\boldsymbol{\phi}_e(i+1) = \boldsymbol{\mathcal{P}}_e\boldsymbol{\psi}_e(i+1) - \boldsymbol{f}_e, \tag{5.51}$$

where

$$\boldsymbol{\psi}_e(i) \triangleq \operatorname{col}\left\{\operatorname{col}\left\{\boldsymbol{\psi}_{k_m}(i)\right\}_{m=1}^{j_k}\right\}_{k=1}^N, \quad \boldsymbol{\phi}_e(i) \triangleq \operatorname{col}\left\{\operatorname{col}\left\{\boldsymbol{\phi}_{k_m}(i)\right\}_{m=1}^{j_k}\right\}_{k=1}^N, \tag{5.52}$$

$\boldsymbol{\mathcal{P}}_e$ is an $M_e \times M_e$ orthogonal projection matrix, and $\boldsymbol{f}_e$ is an $M_e \times 1$ vector given by (see Appendix D.1):

$$\boldsymbol{\mathcal{P}}_e \triangleq \boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{D}}_e^\dagger\boldsymbol{\mathcal{D}}_e = \boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{D}}_e^\top(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top)^{-1}\boldsymbol{\mathcal{D}}_e, \tag{5.53}$$

$$\boldsymbol{f}_e \triangleq \boldsymbol{\mathcal{D}}_e^\dagger\boldsymbol{b} = \boldsymbol{\mathcal{D}}_e^\top(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top)^{-1}\boldsymbol{b}. \tag{5.54}$$

Subtracting $\boldsymbol{w}_e^o$ in (5.30) from both sides of recursion (5.51), we obtain:

$$\widetilde{\boldsymbol{\phi}}_e(i+1) \triangleq \operatorname{col}\left\{\operatorname{col}\left\{\widetilde{\boldsymbol{\phi}}_{k_m}(i+1)\right\}_{m=1}^{j_k}\right\}_{k=1}^N = \boldsymbol{\mathcal{P}}_e\widetilde{\boldsymbol{\psi}}_e(i+1) + \left(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\right)\boldsymbol{w}_e^o + \boldsymbol{f}_e. \tag{5.55}$$

Subtracting $\boldsymbol{w}_k^o$ from both sides of the combination step (5.36c) and using (5.33), we obtain that the network error vector $\widetilde{\boldsymbol{w}}_e(i+1)$ for the diffusion strategy (5.36) evolves according to the following recursion:

$$\boxed{\widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left[\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}(i)\right] \widetilde{\boldsymbol{w}}_e(i) - \mu\, \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e\, \boldsymbol{p}_{xz,e}(i) + \boldsymbol{\mathcal{A}}^\top \left(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\right) \boldsymbol{w}_e^o + \boldsymbol{\mathcal{A}}^\top \boldsymbol{f}_e,} \tag{5.56}$$

where

$$\boldsymbol{\mathcal{A}} \triangleq \operatorname{diag}\left\{\boldsymbol{A}_1 \otimes \boldsymbol{I}_{M_1}, \ldots, \boldsymbol{A}_N \otimes \boldsymbol{I}_{M_N}\right\}. \tag{5.57}$$

Using (5.45) with (5.56), the fact that $\boldsymbol{w}_e^\star$ verifies the constraints $\{\boldsymbol{\mathcal{D}}_e \boldsymbol{w}_e + \boldsymbol{b} = \boldsymbol{0}\}$, namely,

$$\boldsymbol{\mathcal{P}}_e \boldsymbol{w}_e^\star - \boldsymbol{f}_e = \boldsymbol{w}_e^\star, \tag{5.58}$$

and the fact that $\boldsymbol{\mathcal{A}}^\top \mathbb{1} = \mathbb{1}$, we obtain that $\widetilde{\boldsymbol{w}}'_e(i+1)$ evolves according to the following recursion:

$$\boxed{\widetilde{\boldsymbol{w}}'_e(i+1) = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left[\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}(i)\right] \widetilde{\boldsymbol{w}}'_e(i) - \mu\, \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e\, \boldsymbol{p}_{xz,e}(i) - \mu\, \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{R}}_{x,e}(i) \boldsymbol{w}_e^\delta.} \tag{5.59}$$

Recursions (5.56) and (5.59) can be rewritten in a more compact form:

$$\widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{B}}(i) \widetilde{\boldsymbol{w}}_e(i) - \mu \boldsymbol{g}(i) + \boldsymbol{r}, \tag{5.60}$$

$$\widetilde{\boldsymbol{w}}'_e(i+1) = \boldsymbol{\mathcal{B}}(i) \widetilde{\boldsymbol{w}}'_e(i) - \mu \boldsymbol{g}(i) - \mu \boldsymbol{r}'(i), \tag{5.61}$$

where we introduced the following notations:

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left[\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}(i)\right], \tag{5.62}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e\, \boldsymbol{p}_{xz,e}(i), \tag{5.63}$$

$$\boldsymbol{r} \triangleq \boldsymbol{\mathcal{A}}^\top \left(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\right) \boldsymbol{w}_e^o + \boldsymbol{\mathcal{A}}^\top \boldsymbol{f}_e, \tag{5.64}$$

$$\boldsymbol{r}'(i) \triangleq \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{R}}_{x,e}(i) \boldsymbol{w}_e^\delta. \tag{5.65}$$

For ease of reference, we list in Table 5.2 the various analysis symbols that have been defined so far and others that will be defined in the sequel. Before proceeding, let us introduce the following assumption on the regression data.

Assumption 5.1. (Independent regressors) *The regression vectors $\boldsymbol{x}_k(i)$ arise from a zero-mean random process that is temporally white and spatially independent.*

As already explained in the previous chapters, this assumption is commonly used in the adaptive filtering literature. Under this assumption, $\boldsymbol{x}_k(i)$ is independent of $\boldsymbol{w}_{\ell_m}(j)$ for $i \geq j$ and for all ℓ_m.

Table 5.2: List of symbols defined throughout the performance analysis in Chapter 5

Symbol	Equation
$\boldsymbol{C}_k = \operatorname{diag}\left\{c_{k_1}, \ldots, c_{k_{j_k}}\right\}$	(5.26)
$\boldsymbol{c}_k = \operatorname{col}\left\{c_{k_1}, \ldots, c_{k_{j_k}}\right\}$	(5.26)
$\boldsymbol{w}_e^o = \operatorname{col}\{\mathbb{1}_{j_1\times 1}\otimes \boldsymbol{w}_1^o, \ldots, \mathbb{1}_{j_N\times 1}\otimes \boldsymbol{w}_N^o\}$	(5.30)
$\boldsymbol{w}_e^\star = \operatorname{col}\{\mathbb{1}_{j_1\times 1}\otimes \boldsymbol{w}_1^\star, \ldots, \mathbb{1}_{j_N\times 1}\otimes \boldsymbol{w}_N^\star\}$	(5.31)
$\boldsymbol{w}_e^\delta = \boldsymbol{w}_e^o - \boldsymbol{w}_e^\star$	(5.46)
$\boldsymbol{\mathcal{A}} = \operatorname{diag}\{\boldsymbol{A}_1\otimes \boldsymbol{I}_{M_1}, \ldots, \boldsymbol{A}_N\otimes \boldsymbol{I}_{M_N}\}$	(5.57)
$\boldsymbol{\mathcal{P}}_e = \boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{D}}_e^\dagger \boldsymbol{\mathcal{D}}_e$	(5.53)
$\boldsymbol{f}_e = \boldsymbol{\mathcal{D}}_e^\dagger \boldsymbol{b}$	(5.54)
$\boldsymbol{\mathcal{R}}_{x,e} = \operatorname{diag}\{\boldsymbol{C}_1\otimes \boldsymbol{R}_{x,1}, \ldots, \boldsymbol{C}_N\otimes \boldsymbol{R}_{x,N}\}$	(5.23)
$\boldsymbol{r} = \boldsymbol{\mathcal{A}}^\top (\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e)\, \boldsymbol{w}_e^o + \boldsymbol{\mathcal{A}}^\top \boldsymbol{f}_e$	(5.64)
$\boldsymbol{r}' = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{R}}_{x,e} \boldsymbol{w}_e^\delta$	(5.69)
$\boldsymbol{\mathcal{B}}(i) = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left[\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}(i)\right]$	(5.62)
$\boldsymbol{\mathcal{B}} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left(\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}\right)$	(5.67)
$\boldsymbol{\mathcal{G}} = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \operatorname{diag}\left\{\boldsymbol{c}_1\boldsymbol{c}_1^\top \otimes \sigma_{z,1}^2 \boldsymbol{R}_{x,1}, \ldots, \boldsymbol{c}_N\boldsymbol{c}_N^\top \otimes \sigma_{z,N}^2 \boldsymbol{R}_{x,N}\right\} \boldsymbol{\mathcal{P}}_e \boldsymbol{\mathcal{A}}$	(5.81)
$\boldsymbol{\mathcal{F}} = \mathbb{E}\{\boldsymbol{\mathcal{B}}(i) \otimes_b \boldsymbol{\mathcal{B}}(i)\}$	(5.79), (D.16)
$\boldsymbol{\mathcal{Y}}(i) = \mu^2 \boldsymbol{\mathcal{G}}^\top + \boldsymbol{r}\boldsymbol{r}^\top + 2\boldsymbol{r}\mathbb{E}\{\widetilde{\boldsymbol{w}}_e^\top(i)\}\boldsymbol{\mathcal{B}}^\top$	(5.84)

5.4.2 Mean behavior analysis

Taking the expectation of both sides of recursion (5.60), using Assumption 5.1, and $\mathbb{E}\, \boldsymbol{g}(i) = \boldsymbol{0}$, we find that the mean error vector $\mathbb{E}\, \widetilde{\boldsymbol{w}}_e(i+1)$ evolves according to the recursion:

$$\mathbb{E}\, \widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{B}}\, \mathbb{E}\, \widetilde{\boldsymbol{w}}_e(i) + \boldsymbol{r}, \tag{5.66}$$

where

$$\boldsymbol{\mathcal{B}} \triangleq \mathbb{E}\, \boldsymbol{\mathcal{B}}(i) = \boldsymbol{\mathcal{A}}^\top \boldsymbol{\mathcal{P}}_e \left(\boldsymbol{I}_{M_e} - \mu \boldsymbol{\mathcal{R}}_{x,e}\right). \tag{5.67}$$

Likewise, taking the expectation of both sides of recursion (5.61), using Assumption 5.1, and $\mathbb{E}\, \boldsymbol{p}_{xz,e}(i) = \boldsymbol{0}$, we find that the mean error vector $\mathbb{E}\, \widetilde{\boldsymbol{w}}_e'(i+1)$ evolves according to:

$$\mathbb{E}\, \widetilde{\boldsymbol{w}}_e'(i+1) = \boldsymbol{\mathcal{B}}\, \mathbb{E}\, \widetilde{\boldsymbol{w}}_e'(i) - \mu \boldsymbol{r}', \tag{5.68}$$

where $\boldsymbol{\mathcal{B}}$ is given by (5.67) and

$$\boldsymbol{r}' \triangleq \mathbb{E}\,\boldsymbol{r}'(i) = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{R}}_{x,e}\boldsymbol{w}_e^{\delta}. \tag{5.69}$$

Theorem 5.1. (Stability in the mean) *Assume the data model in* (5.2) *and Assumption 5.1 hold. Then, for any initial condition, the multitask diffusion LMS* (5.36) *converges in the mean, i.e., recursions* (5.66) *and* (5.68) *converge as $i \to \infty$, if the step-size μ is chosen such that the matrix $\boldsymbol{\mathcal{B}}$ is contractive. A sufficient condition is:*

$$0 < \mu < \frac{2}{c_{k,\max}\lambda_{\max}(\boldsymbol{R}_{x,k})}, \quad \forall k = 1,\ldots,N, \tag{5.70}$$

where $c_{k,\max} \triangleq \max\limits_{1\le m\le j_k} c_{k_m}$. In this case, the asymptotic mean biases are given by:

$$\lim_{i\to\infty}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i) = (\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{B}})^{-1}\,\boldsymbol{r}, \tag{5.71}$$

$$\lim_{i\to\infty}\mathbb{E}\,\widetilde{\boldsymbol{w}}'_e(i) = -\mu\,(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{B}})^{-1}\,\boldsymbol{r}'. \tag{5.72}$$

Proof. The convergence of recursions (5.66) and (5.68) is guaranteed if the matrix $\boldsymbol{\mathcal{B}}$ is contractive, i.e., $\rho(\boldsymbol{\mathcal{B}}) < 1$. Since any induced matrix norm is lower bounded by spectral radius, we can write in terms of the 2-induced matrix norm:

$$\begin{aligned}\rho(\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e(\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e})) &\le \|\boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\big(\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\big)\|_2,\\ &\le \|\boldsymbol{\mathcal{A}}^{\top}\|_2\cdot\|\boldsymbol{\mathcal{P}}_e\|_2\cdot\|\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\|_2.\end{aligned} \tag{5.73}$$

Since $\boldsymbol{\mathcal{P}}_e$ is an orthogonal projection matrix, its 2-induced matrix norm agrees with its spectral radius which is equal to one. The 2-induced norm of the doubly-stochastic matrix $\boldsymbol{\mathcal{A}}^{\top}$ is equal to one. Since the matrix $\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}$ is a symmetric block diagonal matrix, its 2-induced norm agrees with its spectral radius:

$$\begin{aligned}\|\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\|_2 = \rho(\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}) &= \max_{1\le k\le N}\rho(\boldsymbol{I}_{j_k\cdot M_k} - \mu\,\boldsymbol{C}_k\otimes\boldsymbol{R}_{x,k})\\ &= \max_{1\le k\le N}\max_{1\le m\le j_k}\rho(\boldsymbol{I}_{M_k} - \mu\,c_{k_m}\boldsymbol{R}_{x,k})\end{aligned} \tag{5.74}$$

In order to ensure that $\rho(\boldsymbol{\mathcal{B}}) < 1$, it is sufficient to choose the step-size μ such that:

$$0 < \mu < \frac{2}{c_{k_m}\cdot\lambda_{\max}(\boldsymbol{R}_{x,k})}, \quad \text{for all } k_m \in \mathcal{C}_k \text{ and } k = 1,\ldots,N. \tag{5.75}$$

or alternatively as in (5.70).

If we let $i \to \infty$ on both sides of recursions (5.66) and (5.68), we find the asymptotic mean biases (5.71) and (5.72). □

We observe that when $\boldsymbol{w}_e^{\star} = \boldsymbol{w}_e^{o}$, i.e., perfect model scenario where $\boldsymbol{w}^o$ satisfies the linear equality constraints, the bias with respect to $\boldsymbol{w}_e^o$ in (5.71) reduces to zero.

Consider now the bias with respect to $\boldsymbol{w}_e^{\star}$ in (5.72). Note that this bias depends on the step-size μ and the vector $\boldsymbol{w}_e^{\delta} = \boldsymbol{w}_e^o - \boldsymbol{w}_e^{\star}$. In the next section, we shall illustrate with simulation

results that $\lim_{i\to\infty} \|\mathbb{E}\,\widetilde{\boldsymbol{w}}'_e(i)\|^2$ is on the order of μ^2. The bias (5.72) is zero in two cases: 1) in the perfect model scenario where $\boldsymbol{w}_k^o$ satisfy the constraints ($\boldsymbol{w}_e^\delta = \boldsymbol{0}$); 2) if each agent is involved in at most one constraint ($\boldsymbol{\mathcal{D}}_e = \boldsymbol{\mathcal{D}}'_e = \boldsymbol{\mathcal{D}}$). In this second case, consider (5.69) and observe that $\boldsymbol{\mathcal{A}} = \boldsymbol{I}_{M_e}$. Replacing $\boldsymbol{w}_e^\delta$ by its expression obtained from (5.29), and $\boldsymbol{\mathcal{P}}_e$ by (5.53), yields $\boldsymbol{r}' = \boldsymbol{0}$.

5.4.3 Mean-square-error analysis

To perform the mean-square-error analysis, we shall use the block Kronecker product operator [Koning et al., 1991] instead of the Kronecker product, and the block vectorization operator bvec$(\cdot)$ instead of the vectorization operator vec$(\cdot)$. As explained in [Sayed, 2014a], these block operators preserve the locality of the blocks in the original matrix arguments. These operators will allow us to evaluate the exact expressions of quantities involving fourth-order moments of the regression data (see Appendix D.2). To analyze the convergence in mean-square-error sense, we consider the variance of the weight error vector $\widetilde{\boldsymbol{w}}_e(i)$, weighted by any positive-definite matrix $\boldsymbol{\Sigma}$, that is, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|^2_{\boldsymbol{\Sigma}}$, where $\|\widetilde{\boldsymbol{w}}_e(i)\|^2_{\boldsymbol{\Sigma}} \triangleq \widetilde{\boldsymbol{w}}_e^{\top}(i)\boldsymbol{\Sigma}\widetilde{\boldsymbol{w}}_e(i)$. The freedom in selecting $\boldsymbol{\Sigma}$ allows us to extract various types of information about the network and the sub-nodes.

From (5.60) and Assumption 5.1, we obtain:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|^2_{\boldsymbol{\Sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|^2_{\boldsymbol{\Sigma}'} + \mu^2\mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} + \|\boldsymbol{r}\|^2_{\boldsymbol{\Sigma}} + 2\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i), \tag{5.76}$$

where matrix $\boldsymbol{\Sigma}'$ is given by:

$$\boldsymbol{\Sigma}' = \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\}. \tag{5.77}$$

Let $\boldsymbol{\sigma}$ denotes the $M_e^2\times 1$ vector representation of $\boldsymbol{\Sigma}$ that is obtained by the block vectorization operator, namely, $\boldsymbol{\sigma} \triangleq \text{bvec}(\boldsymbol{\Sigma})$. In the sequel, it will be more convenient to work with $\boldsymbol{\sigma}$ than with $\boldsymbol{\Sigma}$ itself. We will use the notations $\|\boldsymbol{x}\|^2_{\boldsymbol{\Sigma}}$ and $\|\boldsymbol{x}\|^2_{\boldsymbol{\sigma}}$ to denote the same quantity $\boldsymbol{x}^{\top}\boldsymbol{\Sigma}\boldsymbol{x}$. Let $\boldsymbol{\sigma}' = \text{bvec}(\boldsymbol{\Sigma}')$. Using property (A.17), the vector $\boldsymbol{\sigma}'$ can be related to $\boldsymbol{\sigma}$:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}\boldsymbol{\sigma}, \tag{5.78}$$

where $\boldsymbol{\mathcal{F}}$ is an $M_e^2 \times M_e^2$ matrix given by:

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^{\top}(i) \otimes_b \boldsymbol{\mathcal{B}}^{\top}(i)\}. \tag{5.79}$$

The evaluation of the matrix $\boldsymbol{\mathcal{F}}$ requires knowledge of the fourth-order moments of the regression vectors. In Appendix D.2, we provide a closed form expression for the matrix $\boldsymbol{\mathcal{F}}$ in the case where the regressors are real-valued zero-mean Gaussian random vectors. A common alternative is to use the approximation $\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^{\top} \otimes_b \boldsymbol{\mathcal{B}}^{\top}$ for sufficiently small step-sizes (see Chapters 2 and 4 for more details). Under this approximation, and using property (A.19), the matrix $\boldsymbol{\mathcal{F}}$ will be contractive if the step-size is chosen according to condition (5.70).

The second term on the RHS of relation (5.76) can be written as:

$$\mu^2\mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} = \mu^2\mathbb{E}\,\{\boldsymbol{g}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{g}(i)\} = \mu^2\text{Tr}\,(\boldsymbol{\Sigma}\boldsymbol{\mathcal{G}}) \overset{\text{(A.16)}}{=} \mu^2\left[\text{bvec}\left(\boldsymbol{\mathcal{G}}^{\top}\right)\right]^{\top}\boldsymbol{\sigma}, \tag{5.80}$$

where $\boldsymbol{\mathcal{G}}$ is the $M_e \times M_e$ matrix given by:

$$\boldsymbol{\mathcal{G}} \triangleq \mathbb{E}\,\{\boldsymbol{g}(i)\boldsymbol{g}^{\top}(i)\} = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{P}}_e\,\mathrm{diag}\left\{\boldsymbol{c}_1\boldsymbol{c}_1^{\top}\otimes\sigma_{z,1}^2\boldsymbol{R}_{x,1},\ldots,\boldsymbol{c}_N\boldsymbol{c}_N^{\top}\otimes\sigma_{z,N}^2\boldsymbol{R}_{x,N}\right\}\boldsymbol{\mathcal{P}}_e\,\boldsymbol{\mathcal{A}}. \tag{5.81}$$

Similarly, the third term on the RHS of relation (5.76) can be written as:

$$\|\boldsymbol{r}\|_{\boldsymbol{\Sigma}}^2 = \left[\mathrm{bvec}\left(\boldsymbol{r}\boldsymbol{r}^{\top}\right)\right]^{\top}\boldsymbol{\sigma}, \tag{5.82}$$

and the fourth term can be written as:

$$2\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i) = 2\mathrm{Tr}\left(\boldsymbol{r}^{\top}\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)\right) = 2\left[\mathrm{bvec}\left(\boldsymbol{r}\mathbb{E}\,\{\widetilde{\boldsymbol{w}}_e^{\top}(i)\}\boldsymbol{\mathcal{B}}^{\top}\right)\right]^{\top}\boldsymbol{\sigma} \tag{5.83}$$

Let us define the $M_e \times M_e$ time dependent matrix $\boldsymbol{\mathcal{Y}}(i)$ given by:

$$\boldsymbol{\mathcal{Y}}(i) \triangleq \mu^2\boldsymbol{\mathcal{G}}^{\top} + \boldsymbol{r}\boldsymbol{r}^{\top} + 2\boldsymbol{r}\mathbb{E}\,\{\widetilde{\boldsymbol{w}}_e^{\top}(i)\}\boldsymbol{\mathcal{B}}^{\top}. \tag{5.84}$$

Then, the variance relation (5.76) can be expressed as:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2 = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}}^2 + [\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top}\boldsymbol{\sigma}. \tag{5.85}$$

Using relation (5.45), the convergence behavior of algorithm (5.36) toward $\boldsymbol{w}_e^{\star}$ in the mean-square sense can be obtained from $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\Sigma}}^2$ and $\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i+1)$ according to:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e'(i+1)\|_{\boldsymbol{\Sigma}}^2 = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\Sigma}}^2 - 2\,\mathbb{E}\,\{\widetilde{\boldsymbol{w}}_e^{\top}(i+1)\}\boldsymbol{\Sigma}\boldsymbol{w}_e^{\delta} + \|\boldsymbol{w}_e^{\delta}\|_{\boldsymbol{\Sigma}}^2. \tag{5.86}$$

Theorem 5.2. (Mean-square stability) *Assume the data model in* (5.2) *and Assumption 5.1 hold. Then, for any initial condition, the multitask diffusion LMS* (5.36) *converges w.r.t.* $\boldsymbol{w}_e^o$ *in the mean-square-error sense, i.e., the quadratic quantity* $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\Sigma}}^2$ *converges as* $i \to \infty$ *for any positive semi-definite matrix* $\boldsymbol{\Sigma}$*, if the algorithm is mean stable w.r.t.* $\boldsymbol{w}_e^o$ *and if the matrix* $\boldsymbol{\mathcal{F}}$ *in* (5.79) *is contractive.*

Proof. Provided that $\boldsymbol{\mathcal{F}}$ is contractive, recursion (5.85) is stable if $[\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top}\boldsymbol{\sigma}$ is bounded. Since $\boldsymbol{\mathcal{G}}$, $\boldsymbol{r}$, $\boldsymbol{\mathcal{B}}$, $\boldsymbol{\sigma}$, and μ are constant and finite terms, the boundedness of $[\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top}\boldsymbol{\sigma}$ depends on $\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)$ being bounded. We know from (5.66) that $\mathbb{E}\,\widetilde{\boldsymbol{w}}_e(i)$ is bounded because (5.66) is a BIBO stable recursion with a bounded driving term $\boldsymbol{r}$. It follows that $[\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top}\boldsymbol{\sigma}$ is uniformly bounded. As a result, $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2$ converges to a bounded value as $i \to \infty$, and the algorithm is mean-square-error stable. □

Following similar arguments as in Theorems 3.3 and 2.3 and doing the required adjustments, we arrive at the following theorem characterizing the transient behavior of the weighted variance $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i+1)\|_{\boldsymbol{\sigma}}^2$.

Theorem 5.3. (Transient performance) *Assume the same settings as Theorem 5.2. The learning curve defined by* $\zeta(i) \triangleq \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|_{\boldsymbol{\sigma}}^2$ *ends up evolving according to the following recursion:*

$$\zeta(i+1) = \zeta(i) + \|\widetilde{\boldsymbol{w}}_e(0)\|_{\left(\boldsymbol{\mathcal{F}}-\boldsymbol{I}_{M_e^2}\right)\boldsymbol{\mathcal{F}}^i\boldsymbol{\sigma}}^2 + \left([\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top} + \boldsymbol{\Upsilon}(i)\right)\boldsymbol{\sigma}, \tag{5.87}$$

where $\widetilde{\boldsymbol{w}}_e(0)$ *is the initial condition and* $\boldsymbol{\Upsilon}(i+1)$ *is a* $1 \times M_e^2$ *vector that can be evaluated from* $\boldsymbol{\Upsilon}(i)$ *according to:*

$$\boldsymbol{\Upsilon}(i+1) = \boldsymbol{\Upsilon}(i)\boldsymbol{\mathcal{F}} + [\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(i))]^{\top}\left(\boldsymbol{\mathcal{F}} - \boldsymbol{I}_{M_e^2}\right), \tag{5.88}$$

with $\boldsymbol{\Upsilon}(0) = \boldsymbol{0}$.

Proof. The argument is similar to the one used in Theorem 3.3. □

Theorem 5.4. (Steady-state performance) *Consider the same settings as Theorem 5.2. Assume mean and mean-square stability. Then, the steady-state performance defined as* $\zeta^{\star} \triangleq \lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|^2_{\boldsymbol{\sigma}_{ss}}$ *is given by:*

$$\zeta^{\star} = [\mathrm{bvec}(\boldsymbol{\mathcal{Y}}(\infty))]^{\top}\,(\boldsymbol{I}_{M_e^2} - \boldsymbol{\mathcal{F}})^{-1}\boldsymbol{\sigma}, \tag{5.89}$$

where

$$\boldsymbol{\mathcal{Y}}(\infty) \triangleq \mu^2\boldsymbol{\mathcal{G}}^{\top} + \boldsymbol{r}\boldsymbol{r}^{\top} + 2\boldsymbol{r}\lim_{i\to\infty}\mathbb{E}\,\{\widetilde{\boldsymbol{w}}_e^{\top}(i)\}\boldsymbol{\mathcal{B}}^{\top}. \tag{5.90}$$

Proof. The argument is similar to the one used in Theorem 3.4. □

The theoretical findings (5.66), (5.71), (5.87), and (5.89) allow us to predict the behavior in the mean and in the mean-square-error sense of the stochastic algorithm (5.36) w.r.t. the parameter vector $\boldsymbol{w}_e^o$. The transient and steady-state behaviors of $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}'_e(i)\|^2_{\boldsymbol{\Sigma}}$ can be derived from the models derived for $\widetilde{\boldsymbol{w}}_e(i)$ in the mean and mean-square sense according to relation (5.86). We shall show with simulation results that the steady-state $\lim_{i\to\infty}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}'_e(i)\|^2$ is on the order of μ. We observed experimentally that modeling the behavior of $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}'_e(i)\|^2_{\boldsymbol{\Sigma}}$ accurately needs the exact expression of $\boldsymbol{\mathcal{F}}$. In Appendix D.2, we determine $\boldsymbol{\mathcal{F}}$ in the case of Gaussian zero-mean real-valued regressors.

The network MSD w.r.t. $\boldsymbol{w}_e^o$ defined as:

$$\mathrm{MSD}_{\mathrm{net}}(i) \triangleq \frac{1}{N}\sum_{k=1}^{N}\mathrm{MSD}_k(i) = \frac{1}{N}\sum_{k=1}^{N}\left(\frac{1}{j_k}\sum_{m=1}^{j_k}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{km}(i)\|^2\right), \tag{5.91}$$

can be obtained from $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_e(i)\|^2_{\boldsymbol{\Sigma}}$ by setting $\boldsymbol{\Sigma}$ as:

$$\boldsymbol{\Sigma} = \frac{1}{N}\,\mathrm{diag}\left\{\frac{1}{j_1}\boldsymbol{I}_{j_1\cdot M_1}, \ldots, \frac{1}{j_N}\boldsymbol{I}_{j_N\cdot M_N}\right\}. \tag{5.92}$$

Similarly, the network MSD w.r.t. $\boldsymbol{w}_e^{\star}$ can be obtained from $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}'_e(i)\|^2_{\boldsymbol{\Sigma}}$ by setting $\boldsymbol{\Sigma}$ as in (5.92).

5.5 Simulation results

Throughout this section, the factors c_{k_m} are set equal to $\frac{1}{j_k}$ and the sets $\mathcal{N}_{k_m} \cap \mathcal{C}_k = \mathcal{C}_k$ for $m = 1, \ldots, j_k$. We run algorithm (5.36) with $a_{k_n,k_m} = 1/j_k$ for $n = 1, \ldots, j_k$. In the following, we shall compare our algorithm (5.36) with the non-cooperative LMS algorithm (obtained from (5.17)

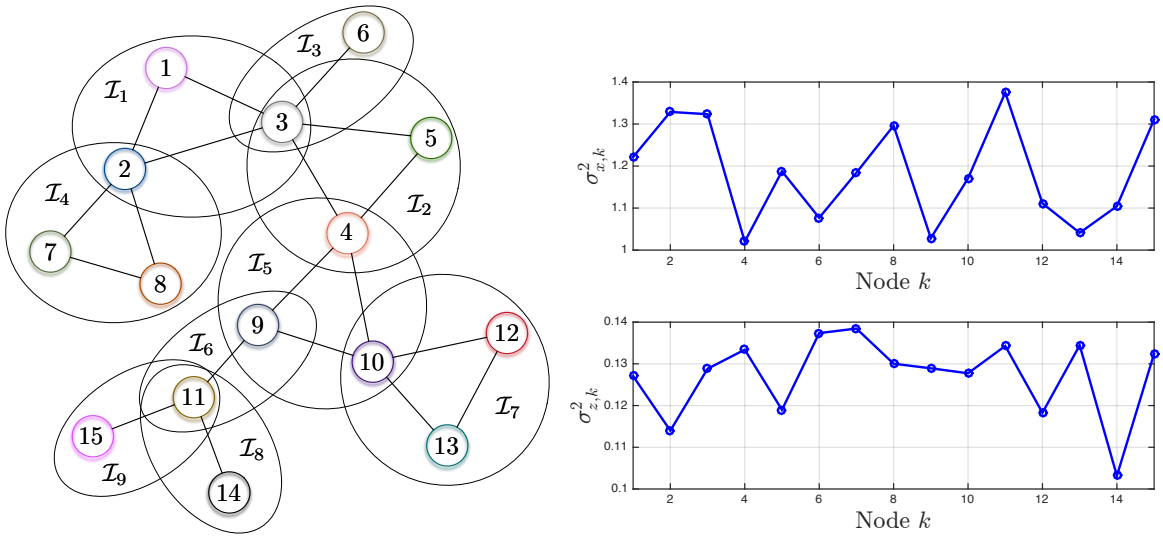


Figure 5.3: Experimental setup. (*Left*) Network topology with constraints. (*Right*) Regression and noise variances.

by setting $\boldsymbol{\mathcal{P}} = \boldsymbol{I}_{M_b}$ and $\boldsymbol{f} = \boldsymbol{0}$), the centralized CLMS algorithm (5.17), and the following algorithm:

$$\begin{cases} \boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m} \boldsymbol{x}_k(i) \left(d_k(i) - \boldsymbol{x}_k^{\top}(i) \boldsymbol{w}_{k_m}(i)\right) \\ \boldsymbol{\phi}_{k_m}(i+1) = \sum_{k_n \in \mathcal{N}_{k_m} \cap \mathcal{C}_k} a_{k_n,k_m} \boldsymbol{\psi}_{k_n}(i+1) \\ \boldsymbol{w}_{k_m}(i+1) = [\boldsymbol{\mathcal{P}}_p]_{k_m,\bullet} \cdot \operatorname{col}\left\{\boldsymbol{\phi}_{\ell_n}(i+1)\right\}_{\ell_n \in \mathcal{I}_{e,p}} - [\boldsymbol{f}_p]_{k_m} \end{cases} \tag{5.93}$$

where the sub-nodes "combine-then-project" instead of "project-then-combine". In Appendix D.3 we show how the theoretical learning curves of these algorithms can be obtained from the analysis in Section 5.4.

5.5.1 Theoretical model validation

We shall now provide an example to illustrate the behavior of algorithm (5.36). We considered a network consisting of 15 agents with the topology shown in Figure 5.3. The regression vectors $\boldsymbol{x}_k(i)$ were 2×1 zero-mean Gaussian distributed with covariance matrices $\boldsymbol{R}_{x,k} = \sigma^2_{x,k} \boldsymbol{I}_2$. The noises $z_k(i)$ were zero-mean i.i.d. Gaussian random variables, independent of any other signal with variances $\sigma^2_{z,k}$. The variances $\sigma^2_{x,k}$ and $\sigma^2_{z,k}$ are shown in Figure 5.3. We randomly sampled 9 linear equality constraints of the form:

$$\sum_{\ell \in \mathcal{I}_p} d_{p\ell} \boldsymbol{w}_\ell = b_p \cdot \mathbb{1}_{2\times 1}, \tag{5.94}$$

where the scalars $d_{p\ell}$ and b_p were randomly chosen from the set $\{-3, -2, -1, 1, 2, 3\}$. We used a constant step-size $\mu = 0.025$ for all agents. The results were averaged over 200 Monte-Carlo runs.

First, we considered the case of a perfect model scenario where the observation parameter vector $\boldsymbol{w}_b^o$ satisfies the equality constraints, i.e., $\boldsymbol{w}_b^\star = \boldsymbol{w}_b^o$. In Figure 5.4, we compare three

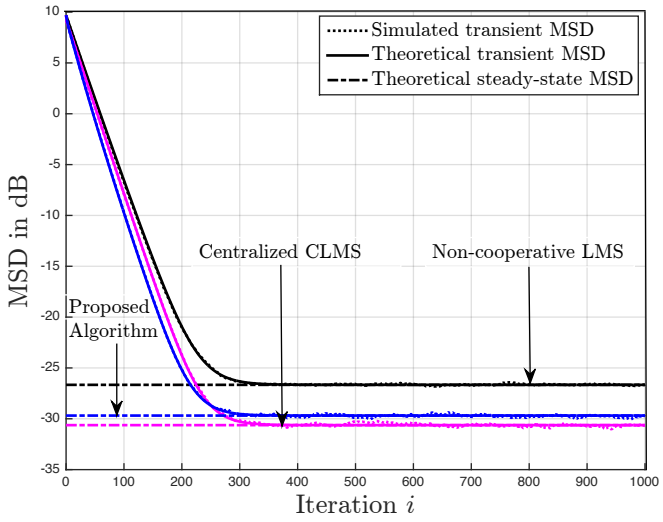


Figure 5.4: Network MSD comparison of the non-cooperative LMS, the centralized CLMS (5.17), and our multitask algorithm (5.36) for the perfect model scenario.

algorithms: the non-cooperative LMS algorithm, the centralized CLMS algorithm (5.17) which assumes that the constraints are available at the fusion center, and our algorithm (5.36). For each algorithm, we report the theoretical transient network MSD, the theoretical steady-state network MSD, and the simulated network MSD. We observe that the simulation results match well the actual performance. Furthermore, the network MSD is improved by promoting relationships between tasks. Finally, our algorithm performs well compared to the centralized solution.

Next, we perturbed the optimum parameter vector $\boldsymbol{w}_b^o$ as follows:

$$\boldsymbol{w}_{\text{pert}}^o = \boldsymbol{w}_b^o + \boldsymbol{u}^o, \tag{5.95}$$

so $\boldsymbol{w}_{\text{pert}}^o$ does not satisfy the constraints (5.94). The entries of $\boldsymbol{u}^o$ were sampled from Gaussian distribution $\mathcal{N}(0, \sigma^2)$. We evaluated our algorithm on 6 different setups characterized by $\sigma \in \{0, 0.01, 0.05, 0.1, 0.2, 0.5, 1\}$. The theoretical and simulated learning curves with respect to $\boldsymbol{w}_e^o$ and $\boldsymbol{w}_e^\star$ are reported in Figure 5.5. Observe that the performance with respect to $\boldsymbol{w}_e^o$ highly deteriorates when σ increases. However, even for the largest values of $\sigma = 1$, our algorithm still perform well with respect to the solution $\boldsymbol{w}_e^\star$ of the optimization problem with constraints.

For comparison purposes, we illustrate in Figure 5.6 the theoretical and simulated learning curves with respect to $\boldsymbol{w}_b^\star$ for the settings where $\sigma = 0.5$ (left) and $\sigma = 1$ (right) of the centralized CLMS algorithm (5.17), the proposed algorithm (5.36) where the sub-nodes "project-then-combine", and algorithm (5.93) where the sub-nodes "combine-then-project". Observe that both algorithms (5.36) and (5.93) have approximately the same performance. However, with the settings considered in this section, algorithm (5.36) is less complex than algorithm (5.93) as explained in subsection 5.3.2. Furthermore, we observe that the larger the vector $\boldsymbol{w}_e^\delta$ is, the larger the performance gap between the centralized solution and the distributed solutions is. This is

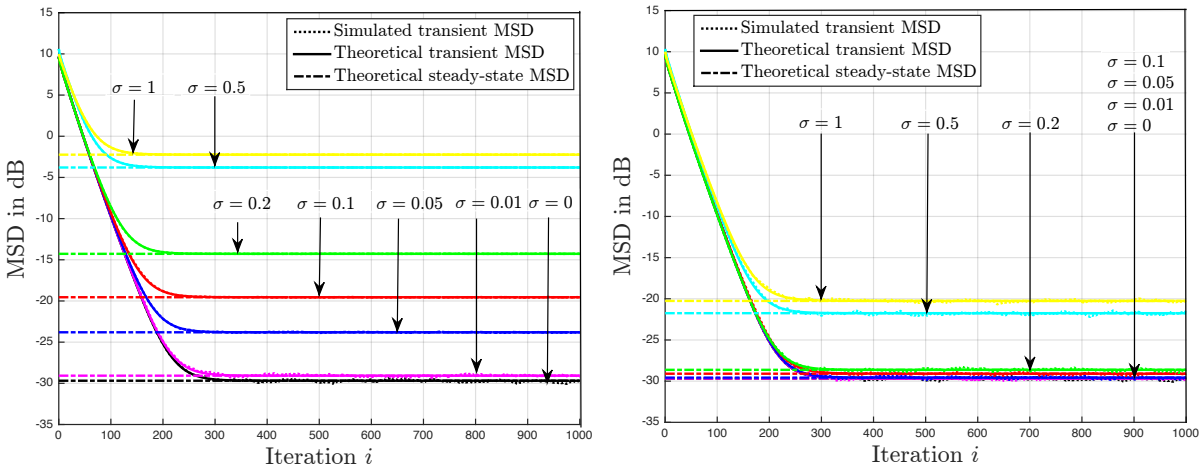


Figure 5.5: Learning curves of our algorithm (5.36) with respect to $\boldsymbol{w}_e^o$ (*left*) and $\boldsymbol{w}_e^\star$ (*right*) for 6 different values of σ.

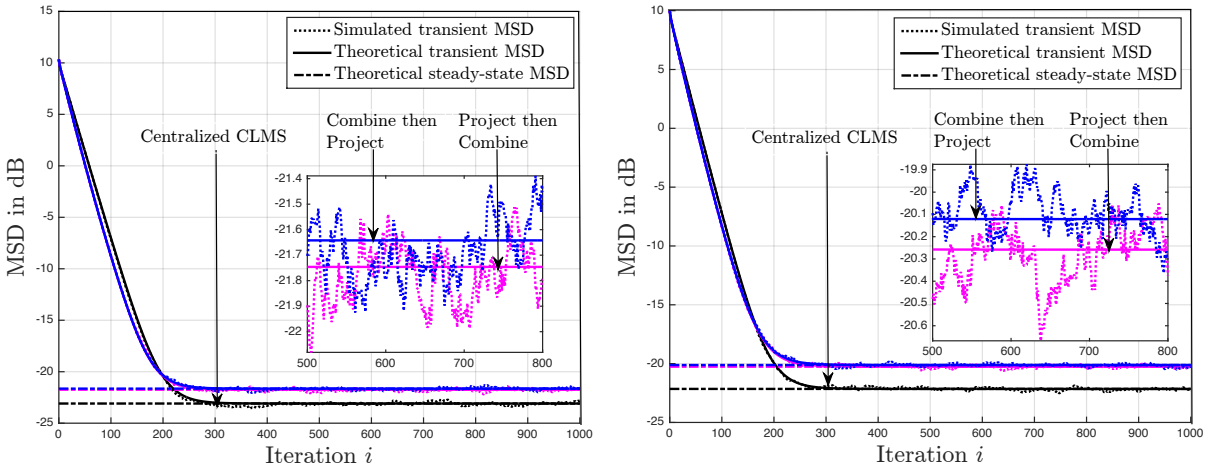


Figure 5.6: Network MSD comparison with respect to $\boldsymbol{w}_b^\star$, $\boldsymbol{w}_e^\star$ of the centralized CLMS (5.17), algorithm (5.36), and algorithm (5.93) for $\sigma = 0.5$ (*left*) and $\sigma = 1$ (*right*).

due to the bias (5.72) induced in the distributed solution which does not exist in the centralized CLMS algorithm.

In order to characterize the influence of the step-size μ on the performance of our algorithm, Figure 5.7 (left) reports the theoretical steady-state MSD with respect to $\boldsymbol{w}_e^\star$ for different values of μ (when $\sigma = 0.5$). We observe that the network MSD increases 10 dB per decade (when the step-size goes from μ_1 to $10\mu_1$). This means that the steady-state MSD is on the order of μ. Figure 5.7 (right) reports the squared norm of the bias (5.72) for different values of μ. We note that it increases approximately 20 dB per decade. This shows that, as expected, this quantity is on the order of μ^2.

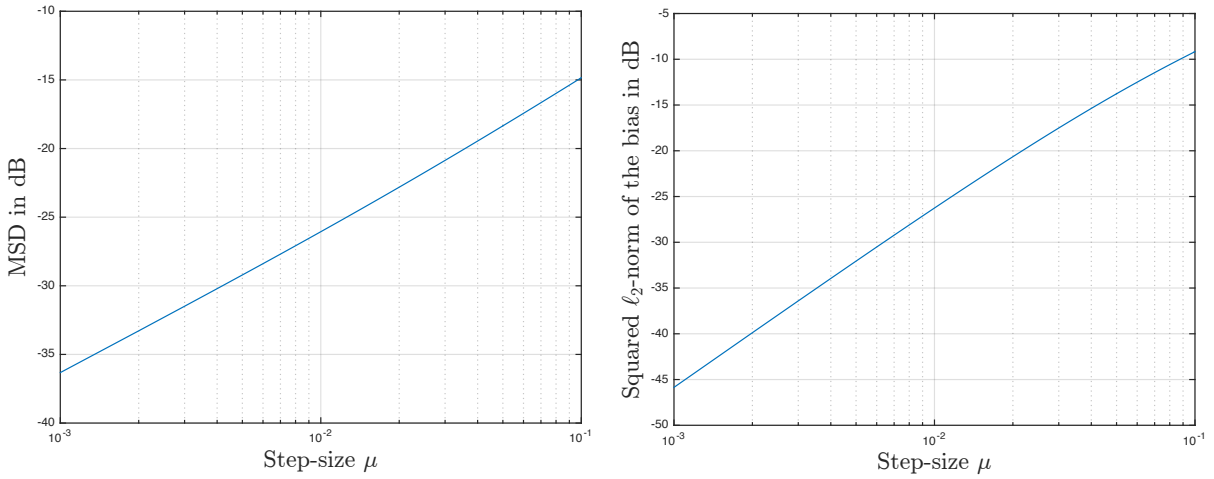


Figure 5.7: Influence of the step-size μ on the performance of the algorithm. (*Left*) Network steady-state MSD for different values of μ. (*Right*) Squared norm of the bias, i.e, $\lim_{i\to\infty} \|\mathbb{E}\, \widetilde{\boldsymbol{w}}'_e(i)\|^2$, for different values of μ.

Next, we considered the case of non-diagonal matrices $\boldsymbol{D}_{p\ell}$ defined as:

$$\boldsymbol{D}_{p\ell} = d_{p\ell}\boldsymbol{I}_2 + \boldsymbol{\Delta}_{p\ell}. \tag{5.96}$$

Parameters $d_{p\ell}$ were randomly selected as in (5.94). The entries of the 2×2 matrix $\boldsymbol{\Delta}_{p\ell}$ were sampled from Gaussian distribution $\mathcal{N}(0, \sigma_D^2)$. As shown in Figure 5.8, the variance σ_D^2 was set to 0.01 (left) and 1 (right). To test the tracking ability of our algorithm, we also perturbed the parameter vector $\boldsymbol{w}_b^o$ as in (5.95) by increasing σ^2 every 500 iterations. In both cases, i.e., $\sigma_D^2 = 0.01$ and $\sigma_D^2 = 1$, $\boldsymbol{w}_b^o$ in (5.95) was set to satisfy the equality constraints defined by $\boldsymbol{D}_{p\ell}$. We observe that the theoretical models match well the actual performance whatever the constraints are. Furthermore, our algorithm adapts its response to drifts in the location of $\boldsymbol{w}_b^\star$ when $\boldsymbol{w}_b^o$ changes over time.

5.5.2 Optimal network flow

As briefly discussed in Section 5.1, we shall now consider the minimum-cost flow problem over the network with topology shown in Figure 5.1. We are interested in online distributed learning where each node k seeks to estimate the entering and leaving flows f_j from noisy measurement $s_k(i)$ of the external source, by relying only on local computations and communications with its neighbors.

Let M_k be the number of flows to be estimated at node k. We denote by $\boldsymbol{w}_k$ the $M_k \times 1$ parameter vector containing the flows f_j entering and leaving node k, negatively and positively signed, respectively. For instance, for nodes 1 and 2, we have:

$$\boldsymbol{w}_1 \triangleq [f_1 \;\; f_2]^\top \quad \boldsymbol{w}_2 \triangleq [-f_1 \;\; f_3 \;\; f_4 \;\; f_5]^\top. \tag{5.97}$$

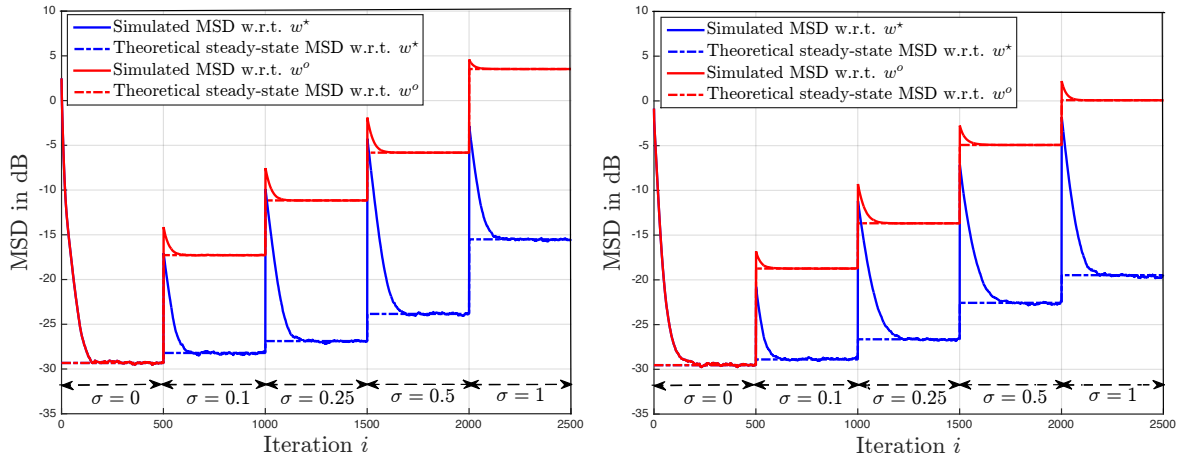


Figure 5.8: Tracking ability of the algorithm for two sets of linear equality constraints. (*Left*) $\sigma_D^2 = 0.01$. (*Right*) $\sigma_D^2 = 1$.

From the flow conservation principle, the noisy measurement $s_k(i)$ can be related to $\boldsymbol{w}_k(i)$ as follows:

$$s_k(i) = \mathbb{1}_{M_k \times 1}^{\top} \boldsymbol{w}_k + z_k(i), \tag{5.98}$$

with $z_k(i)$ a zero-mean measurement noise, and $\mathbb{1}_{M_k \times 1}$ an $M_k \times 1$ vector of ones. We consider the bi-objective problem consisting of minimizing $\mathbb{E}\,|z_k(i)|^2$ and the cost network flow. We shall assume that the cost for flow through an arc is quadratic in the flow, as in applications such as electrical network monitoring and urban traffic control [Ahuja et al., 1993, Ventura, 1991]. We formulate the estimation problem as follows:

$$\underset{\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N}{\text{minimize}} \quad J^{\text{glob}}(\boldsymbol{w}_1,\ldots,\boldsymbol{w}_N) \triangleq \sum_{k=1}^{N} \left(\mathbb{E}\ |s_k(i) - \mathbb{1}_{M_k \times 1}^{\top} \boldsymbol{w}_k|^2 + \frac{\eta}{2} \|\boldsymbol{w}_k\|^2 \right), \tag{5.99a}$$

$$\text{subject to} \quad [\boldsymbol{w}_k]_{f(k,\ell)} + [\boldsymbol{w}_\ell]_{f(\ell,k)} = 0, \quad \ell \in \mathcal{N}_k, \quad k = 1,\ldots,N \tag{5.99b}$$

where $[\boldsymbol{w}_p]_{f(p,q)}$ returns the flow entry in $\boldsymbol{w}_p$ that node p has in common with node q, and η is a tuning parameter to trade off between both objectives.

For each agent k, the external flow s_k and the variance $\sigma_{z,k}^2$ of the Gaussian noise $z_k(i)$ were randomly generated from the uniform distributions $\mathcal{U}(0,3)$ and $\mathcal{U}(0.1,0.14)$, respectively. In order to solve the multitask problem (5.99) in a fully distributed manner, we applied algorithm (5.36) by modifying the adaptation step according to:

$$\boldsymbol{\psi}_{k_m}(i+1) = \boldsymbol{w}_{k_m}(i) + \mu\, c_{k_m} \mathbb{1}_{M_k \times 1} \left(s_k(i) - \mathbb{1}_{M_k \times 1}^{\top} \boldsymbol{w}_{k_m}(i) \right) - \frac{\mu}{2} c_{km} \eta\, \boldsymbol{w}_{k_m}(i), \tag{5.100}$$

and setting $\mu = 0.2$ and $\eta = 0.002$. In order to test the tracking ability of the algorithm, the external flow s_k at each node k was re-generated from $\mathcal{U}(0,3)$ after 45000 iterations. The MSD learning curve with respect to the solution of problem (5.99) is reported in Figure 5.9. This result was obtained by averaging over 150 Monte-Carlo runs. This figure shows that our strategy

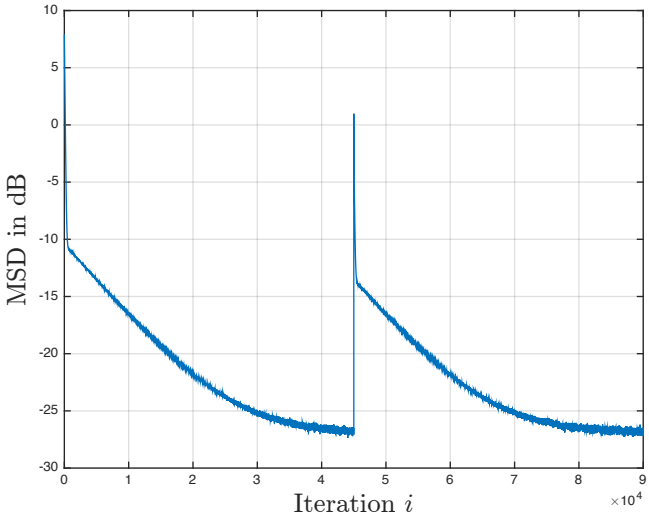


Figure 5.9: MSD performance and tracking ability of our algorithm for the minimum cost network flow problem.

was able to solve the minimum-cost flow problem in a fully distributed manner. The estimated flows over the network for both settings considered in the tracking experiment are showed in Figure 5.10 (a)–(b). Note that the direction of the estimated flow between nodes 3 and 4 is reversed. The true and estimated flows are reported in Figure 5.10 (c) for both settings.

5.5.3 Numerical solution of a two-dimensional process

Consider now the problem of estimating a two-dimensional process driven by a partial differential equation (PDE) with a sensor network. To see how our distributed algorithm can be tuned to address this issue, we shall focus on the Poisson's PDE defined by:

$$\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2} = g(x,y), \qquad (x,y) \in [0,1]^2, \tag{5.101}$$

with $g : [0,1]^2 \to \mathbb{R}$ an input function, and on a two dimensional network of $(n-2)^2$ sensor nodes and $4(n-1)$ boundary points equally spaced over the unit square $(x,y) \in [0,1]^2$ with $\Delta_x = \Delta_y = \Delta = \frac{1}{n-1}$, as illustrated in Figure 5.11 (a).

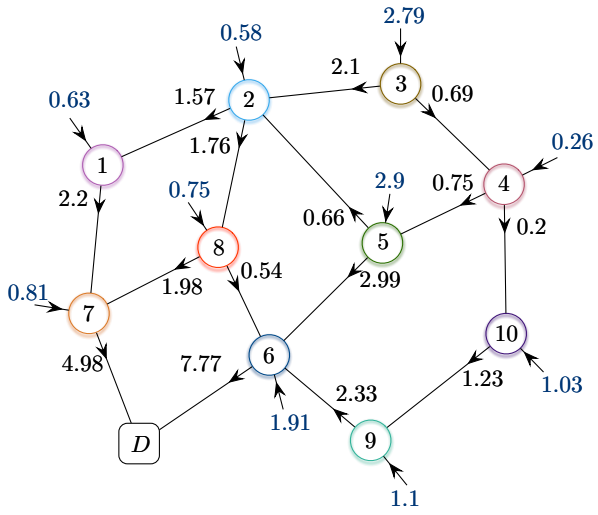
We introduce the grid point $(x_k, y_\ell) \triangleq (k\Delta, \ell\Delta)$ and the sampled values at this point $f_{k,\ell} \triangleq f(k\Delta, \ell\Delta)$ and $g_{k,\ell} \triangleq g(k\Delta, \ell\Delta)$ with $0 \leq k, \ell \leq n-1$. We use the central difference approximation for the second derivative [Bertsekas and Tsitsiklis, 1989]:

$$\frac{\partial^2 f(k\Delta,\ell\Delta)}{\partial x^2} \approx \frac{1}{\Delta^2}\left(f_{k+1,\ell} - 2f_{k,\ell} + f_{k-1,\ell}\right) \tag{5.102}$$

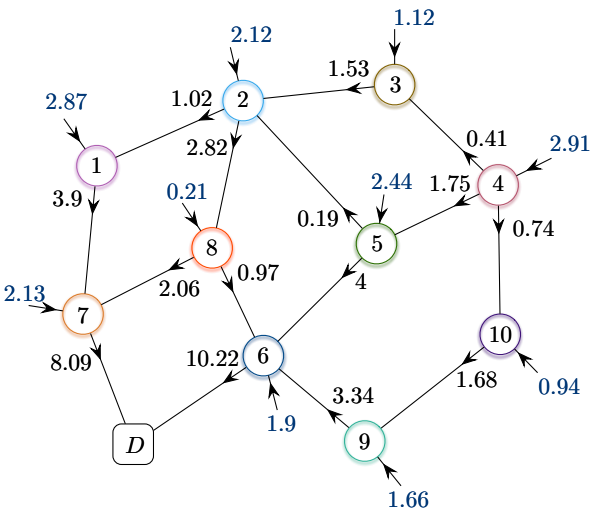
$$\frac{\partial^2 f(k\Delta,\ell\Delta)}{\partial y^2} \approx \frac{1}{\Delta^2}\left(f_{k,\ell+1} - 2f_{k,\ell} + f_{k,\ell-1}\right) \tag{5.103}$$

which leads to:

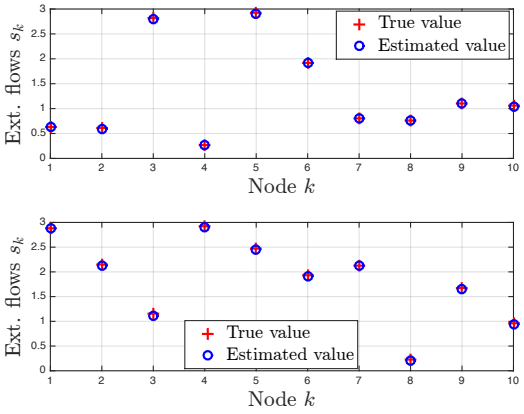
$$\frac{1}{\Delta^2}\left(-4f_{k,\ell} + f_{k-1,\ell} + f_{k,\ell-1} + f_{k,\ell+1} + f_{k+1,\ell}\right) = g_{k,\ell}. \tag{5.104}$$



(a) First experiment



(b) Second experiment



(c) Comparison of the true and estimated flows s_k (top: first experiment, bottom: second experiment)

Figure 5.10: Estimated network flows. A rounding to 2 decimal places is adopted when visualizing the estimated flows.

In this experiment, we shall consider the unknown physical process f and the input function g given by:

$$f(x,y) = (1-x^2)(2y^3-3y^2+1), \qquad (x,y)\in[0,1]^2, \qquad (5.105)$$

$$g(x,y) = -2(2y^3-3y^2+1)+6(1-x^2)(2y-1), \qquad (x,y)\in[0,1]^2, \qquad (5.106)$$

with boundary conditions $f(0,y)=2y^3-3y^2+1$, $f(x,0)=1-x^2$, and $f(1,y)=f(x,1)=0$. These functions are illustrated in Figures 5.11 (b), (c).

The objective is to estimate $f(x,y)$ at the interior grid points (x_k,y_ℓ) with $0<k,\ell<n-1$, given noisy measurements $g_{k\ell}(i)=g_{k\ell}+z_{k\ell}(i)$ of $g(x,y)$ collected by the sensors located at these interior grid points. The noise process $z_{k\ell}(i)$ is assumed to be zero mean, temporally white, and spatially independent. The values of $f(x,y)$ at the boundary points are known a priori as they

Table 5.3: Parameter vector $\boldsymbol{w}_{k\ell}$ (first row of each cell), regression vector $\Delta^2\boldsymbol{x}_{k\ell}$ (second row of each cell), and scalar value $\Delta^2 v^o_{k\ell}$ (last row of each cell) at each node (k,ℓ).

ℓ \ k	1	$2,\ldots,n-3$	$n-2$
1	$[f_{k,\ell}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1]^\top$ $f^o_{1,0}+f^o_{0,1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{k,0}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell+1}]^\top$ $[-4,1,1]^\top$ $f^o_{n-2,0}+f^o_{n-1,1}$
2 $\vdots$ $n-3$	$[f_{k,\ell}, f_{k,\ell-1}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{0,\ell}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k,\ell+1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1,1]^\top$ 0	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k,\ell+1}]^\top$ $[-4,1,1,1]^\top$ $f^o_{n-1,\ell}$
$n-2$	$[f_{k,\ell}, f_{k,\ell-1}, f_{k+1,\ell}]^\top$ $[-4,1,1]^\top$ $f^o_{0,n-2}+f^o_{1,n-1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}, f_{k+1,\ell}]^\top$ $[-4,1,1,1]^\top$ $f^o_{k,n-1}$	$[f_{k,\ell}, f_{k-1,\ell}, f_{k,\ell-1}]^\top$ $[-4,1,1]^\top$ $f^o_{n-2,n-1}+f^o_{n-1,n-2}$

correspond to boundary conditions. We denote by $f^o_{k\ell}$ the value at (x_k, y_ℓ) of the function $f(x,y)$ that satisfies (5.101), and by $f_{k\ell}$ the estimated value of $f^o_{k\ell}$. To each node (k,ℓ) we associate an $M_{k\ell}\times 1$ parameter vector $\boldsymbol{w}_{k\ell}$ to estimate, an $M_{k\ell}\times 1$ regression vector $\boldsymbol{x}_{k\ell}$ and a scalar $v^o_{k\ell}$, defined in Table 5.3 depending on the node location on the grid.

According to (5.104), the linear regression model can be written as follows:

$$g_{k\ell}(i) = \boldsymbol{x}_{k\ell}^\top \boldsymbol{w}_{k\ell} + v^o_{k\ell} + z_{k\ell}(i). \tag{5.107}$$

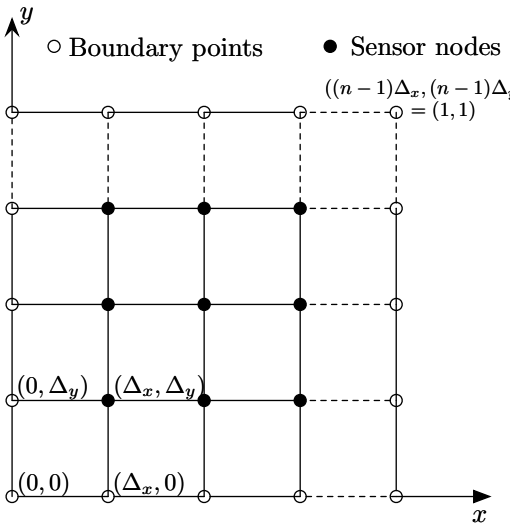
As can be seen in Table 5.3, equality constraints of the form (5.1b) need to be imposed on the parameter vectors of neighboring sensor nodes in order to achieve equality between common entries. For instance, let us consider neighboring nodes (k,ℓ) and $(k+1,\ell)$ with $2\le k\le n-4$ and $2\le \ell\le n-3$. Since these nodes are jointly estimating $f_{k,\ell}$ and $f_{k+1,\ell}$, the following equality constraint is required:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{w}_{k\ell} + \begin{bmatrix} 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{w}_{(k+1)\ell} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{5.108}$$

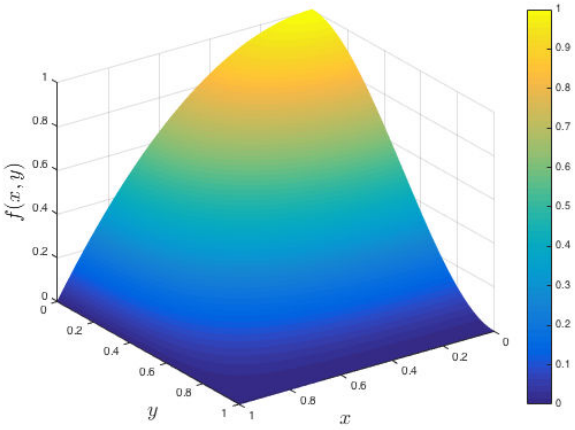
Algorithm (5.36) can be used to address this problem by replacing the adaptation step (5.36a) by:

$$\boldsymbol{\psi}_{k\ell_m}(i+1) = \boldsymbol{w}_{k\ell_m}(i) + \mu\, c_{k\ell_m}\boldsymbol{x}_{k\ell}\left(g_{k\ell}(i) - \boldsymbol{x}_{k\ell}^\top \boldsymbol{w}_{k\ell_m}(i) - v^o_{k\ell}\right), \tag{5.109}$$

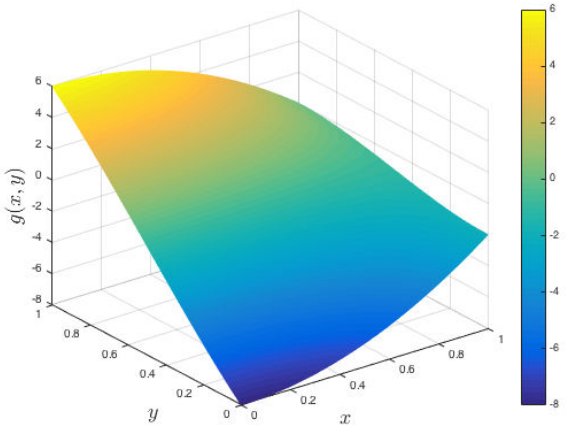
where $\boldsymbol{w}_{k\ell_m}(i)$ denotes the estimate of $\boldsymbol{w}_{k\ell}$ at the m-th sub-node of (k,ℓ). The noises $z_{k,\ell}(i)$ were zero-mean i.i.d. Gaussian distributed with variances $\sigma^2_{z,k\ell}$ randomly generated from the uniform distribution $\mathcal{U}(0.1, 0.14)$. We used a constant step-size $\mu = 7\cdot 10^{-5}$ for all nodes. Figure 5.12 shows the network MSD learning curves for $n=9$. The simulated curves were obtained by averaging over 100 independent runs. Figure 5.13 shows the true (left) and estimated (right) process after convergence of our algorithm.



(a) An $n \times n$ grid network for the solution of Poisson's equation



(b) $f(x,y) = (1-x^2)(2y^3 - 3y^2 + 1)$



(c) $g(x,y) = -2(2y^3 - 3y^2 + 1) + 6(1-x^2)(2y-1)$

Figure 5.11: Network topology, function $f(x,y)$ to estimate over the interior grid points, and input function $g(x,y)$.

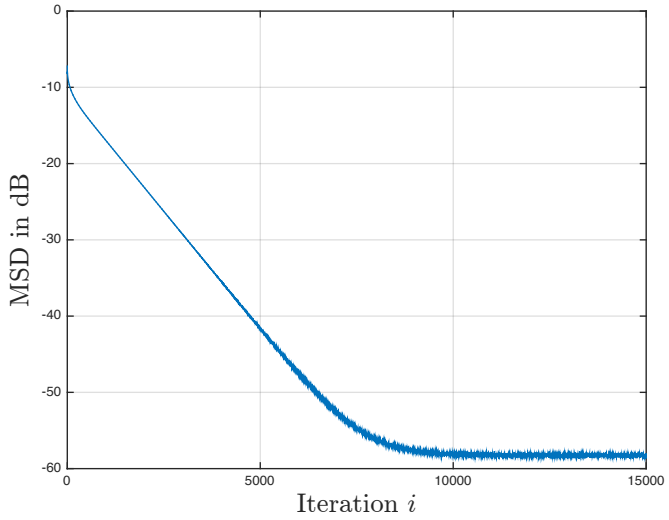


Figure 5.12: Network MSD performance for $n = 9$.

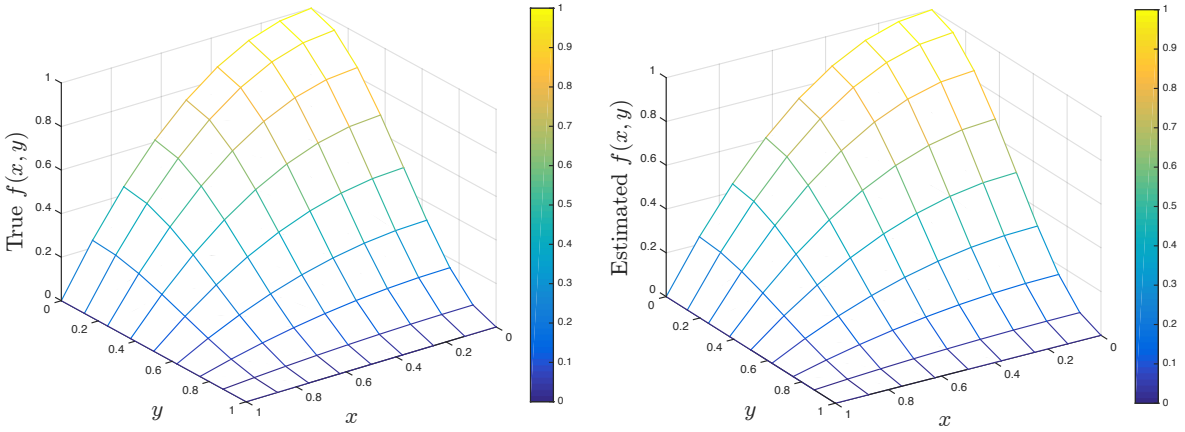


Figure 5.13: Poisson process $f(x, y)$ over the network grid. (*Left*) True process. (*Right*) Estimated process.

5.6 Conclusion

In this chapter, we proposed a multitask LMS algorithm for solving problems that require the simultaneous estimation of multiple parameter vectors that are related locally via linear equality constraints. Our primal technique was based on the stochastic gradient projection algorithm with constant step-sizes. The behavior of the algorithm in the mean and mean-square-error sense was studied. We showed how the agents are able to reach the optimal solution with arbitrarily good precision.

In the next chapter, which is the last, we close this dissertation by summarizing the main contributions and presenting several research directions that can be pursued in the field of distributed adaptive estimation over multitask networks.

Chapter 6

Conclusion and Future works

Motivated by the ubiquity of networked systems and distributed data acquisition nowadays, we focused in this thesis on distributed processing over adaptive networks where agents process streaming data. In particular, we were concerned with multitask problems involving the estimation of several parameter vectors simultaneously. This is in contrast to the traditional single-task situation where all agents are seeking the same objective, or are sensing data arising from the same model. The main motivation behind formulating multitask estimation problems is that there exists a wide range of applications where it cannot be assumed that there is only one phenomenon influencing the whole network. Throughout the dissertation, several task relatedness models were considered, including, smooth graph signal, piecewise-constant transitions, and linear relationships. Newly distributed multitask algorithms were derived and their performances were studied using the energy conservation framework. Furthermore, we considered practical conditions such as exchanging information over noisy communication links, agents turning on and off randomly for energy conservation, and random links failures. Since these conditions directly influence the behavior of the algorithms and play critical role in the convergence, we showed how the adaptive strategies can be modified in order to enhance the real-time adaptation, self-healing, and self-organization features. In the following, we summarize the main contributions of this thesis and discuss the presented results. Then, we present several future works that may be conducted following the studies of this dissertation and others research directions that can be pursued in the field.

6.1 Summary of main results

Both chapters 2 and 3 were related to analyzing the performance of distributed adaptive strategies under limiting aspects and practical conditions, including, imperfect information exchange and

asynchronous events. In chapter 2, the well-known single-task diffusion strategies over MSE networks were reviewed and applied to multitask scenarios where the agents are sensing data arising from different models. We assumed that the regressors and the estimates are corrupted by additive noise during their transmission. Under these conditions, we first showed that if the single-task algorithm is implemented to estimate the underlying systems parameters without considering the multitask environment and the noise corrupting the regressors, the estimates at each agent will be biased. We then showed how this bias can be eliminated by minimizing the instantaneous mean-square error at each agent w.r.t. the combination coefficients and how a clustered network with zero probability of erroneous clustering can be obtained. Interestingly, the results revealed that the measurement noise and the noise corrupting the estimates affect the MSD performance without inducing a bias, and that the dynamic range of the step-sizes is only affected by the noise corrupting the regressors. Chapter 3 reviewed the useful multitask diffusion LMS approach proposed in the literature for solving multitask estimation problems over clustered networks where it is assumed that the optimal models of adjacent clusters are close to each others in the sense of ℓ_2-norm. We considered the practical scenario where synchronization is a very restrictive assumption to impose and hard to satisfy. Indeed, in addition to the fact that the synchronization of clocks is difficult in distributed implementations, the networks can be subject to several sources of uncertainty, including random links, agents failures, and packet losses. We explored the asynchronous implementation of the multitask algorithm which allows the agents to handle several asynchronous events. We studied its performance and convergence properties as a function of the random events occurring in the clustered asynchronous network. We checked the theoretical findings and the robustness of the multitask strategy with simulation results, and showed how the asynchronous network could benefit from the cooperation between clusters. Finally, the asynchronous algorithm was applied to the problem of circular arcs localization involving a smoothness condition.

Chapter 4 was dedicated to multitask estimation problems over clustered networks where the optimum parameter vectors at neighboring clusters have a large number of similar components and a relatively small number of distinct components. Based on the proximal gradient method and diffusion strategies, we proposed a distributed adaptive algorithm consisting of three steps: an adaptation step (involving intra-cluster cooperation), a combination step (involving intra-cluster cooperation), and a proximal step (involving inter-cluster cooperation). Instead of applying iterative algorithms to evaluate the proximal operator of a weighted sum of ℓ_1-norms at each iteration, a closed form expression was derived. We theoretically analyzed the convergence of the algorithm in the mean and mean-square error sense and we provided conditions on the step-sizes ensuring stability. Interestingly, we found that the stability condition is not affected by the combination coefficients and the regularization factors. In order to guarantee an appropriate cooperation between clusters, especially in non-stationary environments, we proposed a rule to adapt the regularization factors based on a well-known measure of sparsity. The performance

of the algorithm and the adaptive regularizers were evaluated through numerical simulations, showing comparisons w.r.t. other state-of-the art techniques. Finally, the algorithm was applied to the problem of cooperative spectrum sensing in cognitive radio networks.

In Chapter 5 we were concerned with multitask estimation problems where each agent is interested in estimating its own parameter vector, and where the parameter vectors of neighboring agents are related according to a set of linear equality constraints. Since each agent can be involved in several constraints, we proposed to associate with each agent a cluster of virtual sub-nodes, each one being responsible of one constraint. Based on the gradient projection method and diffusion strategies, we devised a new adaptive algorithm consisting of three steps: an adaptation step, a projection step, and a local combination step. Detailed theoretical analyses in the mean and mean-square sense were carried out. The theoretical results were validated through numerical simulations, showing that for small step-sizes, the expected distance between the estimates and the optimal solution can be made arbitrarily small. The algorithm was applied to two multitask applications: the cooperative estimation of network flows and the field reconstruction problems over sensor networks.

6.2 Discussion and Future directions

In Chapter 2, it is assumed that the regression vectors and the estimates exchanged between neighbors are corrupted by communication noise, and that the regression data at the agents themselves are noise-free. In practical conditions, the regression data collected at agents are corrupted by noise. It is well known that this noise will induce a bias in the estimates and that a bias-elimination technique (different from that considered in the chapter) is required at each agent [Abdolee and Champagne, 2016, Bertrand, 2011]. Studying the performance of the diffusion LMS in multitask environment and deriving clustering techniques in the presence of noisy regressors at the agents themselves is an interesting research path.

In this thesis, it is assumed that there is some prior knowledge on the relationships between tasks. Regularization terms were used to promote these relationships. Within the machine learning context, Bayesian based approaches allowing to estimate a probability distribution capturing the relations between tasks simultaneously with the tasks have been introduced in the literature [Bakker and Heskes, 2003]. It would be interesting to extend Bayesian modeling to multitask adaptive networks, especially because, in the recent years, there has been growing interest in introducing Bayesian techniques to adaptive filtering. These extensions could help, for example, in the determination of the combination weights and the regularization factors.

In Chapter 3 and 4, it is assumed that the tasks are close to each others in the sense of squared ℓ_2-norm and ℓ_1-norm, respectively. In some applications, e.g., state estimation in power grids [Kekatos and Giannakis, 2013], the parameter vectors at neighboring agents partially overlap. Then, we have to consider group of variables instead of variables individually. In this case,

group-sparsity inducing regularizers, such as Group Lasso based penalty ($\ell_{2,1}$-norm), should be used instead of the Lasso based penalty (ℓ_1-norm) or the Tikhonov based penalty (squared ℓ_2-norm). Thus, it would be advantageous to consider alternative co-regularizers in order to promote other properties such as block sparsity, and to analyze the convergence behavior of the resulting algorithms.

In Chapter 4, we used the ℓ_1-norm and reweighted ℓ_1-norm to promote sparsity in the multitask setting. In the single-task scenario, other thresholding operators, e.g., Garotte and approximate ℓ_0-norm, are typically used to promote the sparsity of the parameter vector, and can lead to better performance when compared to the ℓ_1-norm and reweighted ℓ_1-norm [Lorenzo, 2014]. Using the Garotte and the approximate ℓ_0-norm in our multitask setting would be interesting but also challenging since the proximal operators corresponding to weighted sums of these functions must be elaborated. Future work in this direction could be beneficial.

In Chapter 5, we assumed linear equality constraints relating the parameter vectors of neighboring clusters. In principle, the extension of our strategy to other forms of constraints, whose projection operators can be evaluated in closed form, is straightforward. However, for more general convex constraints, whose projection operators cannot be evaluated analytically, our strategy cannot be applied. In the single-task scenario (recall problem (1.1) in the introduction), the authors in [Towfic and Sayed, 2014] proposed to use appropriate penalty functions and replaced the projection steps by stochastic approximation updates that run concurrently with the optimization step. A future research direction can be defined as deriving and studying penalty based adaptive algorithms for solving multitask estimation problems where the parameter vectors at neighboring agents are related according to a set of convex constraints.

Throughout the thesis, we considered linear regression models where each agent is collecting linearly related measurements. However, there exist several phenomena of interest that cannot be modeled by linear functions. Therefore, generalization of the multitask strategies considered in this thesis to estimate parameters of non-linear regression models or combinations of linear and nonlinear models [Ammanouil et al., 2016] is important. Note that, recently, there has been a growing interest in deriving and studying distributed kernel-based algorithms for non-linear adaptive filtering over single-task networks where all agents are observing data arising from the same non-linear model [Gao et al., 2015, Chouvardas and Draief, 2016]. The extension to multitask networks can be considered a significant contribution in the field.

Most of the existing tools for the analysis of static signals over graphs focus on centralized implementations and, in many modern applications, the data are already distributed over a network of agents [Shuman et al., 2013, Lorenzo et al., 2016]. Furthermore, the agents may be subject to streaming data. Hence, it would be advantageous to process the information in a distributed and adaptive manner and, as explained in [Lorenzo et al., 2016], the development of distributed adaptive processing tools for signals defined over graphs would benefit from the field of distributed estimation over networks. For example, in [Lorenzo et al., 2016], the development of distributed

sampling theories for the distributed adaptive reconstruction of signals defined over graphs has been considered. Another perspective would be to study the performance of diffusion learning over dynamic networks or random graphs (e.g., stochastic block models used in [Tiomoko Ali and Couillet, 2016]), which are useful to model many real-time applications [Jackson, 2008, Zhang et al., 2016].

Therefore, future works should consider the points raised above as well as other new applications involving multitask learning with various forms of relationships.

APPENDIX A

MATRIX PROPERTIES

In this appendix, we recall several matrix properties that are used in the text.

A.1 Kronecker product

Let $\boldsymbol{X} = [x_{ij}]$ and $\boldsymbol{Y} = [y_{ij}]$ be two matrices of dimensions $N \times N$ and $M \times M$, respectively. Their Kronecker product is denoted by $\boldsymbol{X} \otimes \boldsymbol{Y}$ and is defined as the $MN \times MN$ matrix given by [Bernstein, 2005]:

$$\boldsymbol{X} \otimes \boldsymbol{Y} = \begin{bmatrix} x_{11}\boldsymbol{Y} & x_{12}\boldsymbol{Y} & \dots & x_{1N}\boldsymbol{Y} \\ x_{21}\boldsymbol{Y} & x_{22}\boldsymbol{Y} & \dots & x_{2N}\boldsymbol{Y} \\ \vdots & \vdots & \vdots & \vdots \\ x_{N1}\boldsymbol{Y} & x_{N2}\boldsymbol{Y} & \dots & x_{NN}\boldsymbol{Y} \end{bmatrix} \tag{A.1}$$

For any two vectors $\{\boldsymbol{x}, \boldsymbol{y}\}$, we have:

$$\mathrm{vec}(\boldsymbol{x}\boldsymbol{y}^\top) = \boldsymbol{y} \otimes \boldsymbol{x}, \tag{A.2}$$

where vec$(\cdot)$ operator trasforms a matrix into a vector by stacking the columns of the matrix on top of each other. For any matrices $\{\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{W}\}$, we have:

$$(\boldsymbol{X} + \boldsymbol{Y}) \otimes (\boldsymbol{Z} + \boldsymbol{W}) = \boldsymbol{X} \otimes \boldsymbol{Z} + \boldsymbol{X} \otimes \boldsymbol{W} + \boldsymbol{Y} \otimes \boldsymbol{Z} + \boldsymbol{Y} \otimes \boldsymbol{W}, \tag{A.3}$$

$$(\boldsymbol{X}\boldsymbol{Z}) \otimes (\boldsymbol{Y}\boldsymbol{W}) = (\boldsymbol{X} \otimes \boldsymbol{Y})(\boldsymbol{Z} \otimes \boldsymbol{W}), \tag{A.4}$$

$$\mathrm{Tr}(\boldsymbol{X}\boldsymbol{Y}) = [\,\mathrm{vec}(\boldsymbol{Y}^\top)]^\top \mathrm{vec}(\boldsymbol{X}), \tag{A.5}$$

$$\mathrm{vec}(\boldsymbol{X}\boldsymbol{Y}\boldsymbol{Z}) = (\boldsymbol{Z}^\top \otimes \boldsymbol{X})\,\mathrm{vec}(\boldsymbol{Y}), \tag{A.6}$$

$$(\boldsymbol{X} \otimes \boldsymbol{Y})^\top = \boldsymbol{X}^\top \otimes \boldsymbol{Y}^\top, \tag{A.7}$$

$$\{\lambda(\boldsymbol{X} \otimes \boldsymbol{Y})\} = \left\{\lambda_i(\boldsymbol{X})\lambda_j(\boldsymbol{Y})\right\}_{i=1,j=1}^{N,M}. \tag{A.8}$$

A.2 Block Kronecker product

Let $\boldsymbol{\mathcal{X}}$ denote an $N \times N$ block matrix with blocks $\{\boldsymbol{X}_{ij}\}$ of size $M \times M$. Likewise, let $\boldsymbol{\mathcal{Y}}$ denote a second $N' \times N'$ block matrix with each block $\boldsymbol{Y}_{ij}$ of size $M \times M$. The block Kronecker product of these two matrices is denoted by $\boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Y}}$ and is defined as the $NN'M^2 \times NN'M^2$ matrix $\boldsymbol{\mathcal{Z}}$ [Koning et al., 1991, Sayed, 2014a]:

$$\boldsymbol{\mathcal{Z}} \triangleq \boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Y}} = \begin{bmatrix} \boldsymbol{Z}_{11} & \boldsymbol{Z}_{12} & \ldots & \boldsymbol{Z}_{1N} \\ \boldsymbol{Z}_{21} & \boldsymbol{Z}_{22} & \ldots & \boldsymbol{Z}_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{Z}_{N1} & \boldsymbol{Z}_{N2} & \ldots & \boldsymbol{Z}_{NN} \end{bmatrix} \tag{A.9}$$

where each block $\boldsymbol{Z}_{ij}$ is of dimensions $N'M^2 \times N'M^2$ given by:

$$\boldsymbol{Z}_{ij} = \begin{bmatrix} \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{11} & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{12} & \ldots & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{1N'} \\ \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{21} & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{22} & \ldots & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{2N'} \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{N'1} & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{N'2} & \ldots & \boldsymbol{X}_{ij} \otimes \boldsymbol{Y}_{N'N'} \end{bmatrix} \tag{A.10}$$

For any two block vectors $\{\boldsymbol{x}, \boldsymbol{y}\}$ we have:

$$\text{bvec}(\boldsymbol{x}\boldsymbol{y}^\top) = \boldsymbol{y} \otimes_b \boldsymbol{x}. \tag{A.11}$$

where the bvec$(\cdot)$ operator vectorizes each block of the matrix and then stacks the resulting columns on top of each other, i.e.,

$$\text{bvec}(\boldsymbol{\mathcal{X}}) \triangleq \text{col}\big\{\text{vec}(\boldsymbol{X}_{11}), \text{vec}(\boldsymbol{X}_{21}), \ldots, \text{vec}(\boldsymbol{X}_{N1}), \text{vec}(\boldsymbol{X}_{12}), \ldots\big\}. \tag{A.12}$$

For any block-matrices $\{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}, \boldsymbol{\mathcal{Z}}, \boldsymbol{\mathcal{W}}\}$ with blocks of size $M \times M$, we have:

$$(\boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{Y}}) \otimes_b (\boldsymbol{\mathcal{Z}} + \boldsymbol{\mathcal{W}}) = \boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Z}} + \boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{W}} + \boldsymbol{\mathcal{Y}} \otimes_b \boldsymbol{\mathcal{Z}} + \boldsymbol{\mathcal{Y}} \otimes_b \boldsymbol{\mathcal{W}}, \tag{A.13}$$

$$(\boldsymbol{\mathcal{X}}\boldsymbol{\mathcal{Z}}) \otimes_b (\boldsymbol{\mathcal{Y}}\boldsymbol{\mathcal{W}}) = (\boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Y}})(\boldsymbol{\mathcal{Z}} \otimes_b \boldsymbol{\mathcal{W}}), \tag{A.14}$$

$$(\boldsymbol{X} \otimes \boldsymbol{Y}) \otimes_b (\boldsymbol{Z} \otimes \boldsymbol{W}) = (\boldsymbol{X} \otimes \boldsymbol{Z}) \otimes (\boldsymbol{Y} \otimes \boldsymbol{W}), \tag{A.15}$$

$$\text{Tr}(\boldsymbol{\mathcal{X}}\boldsymbol{\mathcal{Y}}) = [\,\text{bvec}(\boldsymbol{\mathcal{Y}}^\top)]^\top \text{bvec}(\boldsymbol{\mathcal{X}}), \tag{A.16}$$

$$\text{bvec}(\boldsymbol{\mathcal{X}}\boldsymbol{\mathcal{Y}}\boldsymbol{\mathcal{Z}}) = (\boldsymbol{\mathcal{Z}}^\top \otimes_b \boldsymbol{\mathcal{X}})\,\text{bvec}(\boldsymbol{\mathcal{Y}}), \tag{A.17}$$

$$(\boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Y}})^\top = \boldsymbol{\mathcal{X}}^\top \otimes_b \boldsymbol{\mathcal{Y}}^\top, \tag{A.18}$$

$$\{\lambda(\boldsymbol{\mathcal{X}} \otimes_b \boldsymbol{\mathcal{Y}})\} = \big\{\lambda_i(\boldsymbol{\mathcal{X}})\lambda_j(\boldsymbol{\mathcal{Y}})\big\}_{i=1,j=1}^{MN,MN'}. \tag{A.19}$$

A.3 Block maximum norm

Let $\boldsymbol{x} = \text{col}\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ denote an $N \times 1$ block column vector with each block $\boldsymbol{x}_k$ of size $M \times 1$. The block maximum norm of $\boldsymbol{x}$ is denoted by $\|\boldsymbol{x}\|_{b,\infty}$ and is defined as [Sayed, 2014c]:

$$\|\boldsymbol{x}\|_{b,\infty} \triangleq \max_{1 \le k \le N} \|\boldsymbol{x}_k\|, \tag{A.20}$$

where $\|\boldsymbol{x}_k\|$ is the Euclidean norm of $\boldsymbol{x}_k$. This vector norm induces a block maximum matrix norm. Let $\boldsymbol{\mathcal{X}}$ denote an $N \times N$ block matrix with individual blocks of size $M \times M$. The block maximum norm of $\boldsymbol{\mathcal{X}}$ is defined as:

$$\|\boldsymbol{\mathcal{X}}\|_{b,\infty} \triangleq \max_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\|\boldsymbol{\mathcal{X}}\boldsymbol{x}\|_{b,\infty}}{\|\boldsymbol{x}\|_{b,\infty}}. \tag{A.21}$$

Let $\boldsymbol{\mathcal{Y}} = \text{diag}\{\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_N\}$ denote an $N \times N$ block diagonal matrix where each block $\boldsymbol{Y}_k$ is $M \times M$ symmetric. Then, we have [Sayed, 2014c]:

$$\|\boldsymbol{\mathcal{Y}}\|_{b,\infty} = \rho(\boldsymbol{\mathcal{Y}}) = \max_{1 \leq k \leq N} \rho(\boldsymbol{Y}_k). \tag{A.22}$$

Let $\boldsymbol{X}$ be an $N \times N$ left-stochastic matrix and $\boldsymbol{Y}$ be an $N \times N$ right-stochastic matrix. Let $\boldsymbol{\mathcal{X}} \triangleq \boldsymbol{X} \otimes \boldsymbol{I}_M$ and $\boldsymbol{\mathcal{Y}} \triangleq \boldsymbol{Y} \otimes \boldsymbol{I}_M$. It holds that [Sayed, 2014c]:

$$\|\boldsymbol{\mathcal{X}}^\top\|_{b,\infty} = 1, \tag{A.23}$$

$$\|\boldsymbol{\mathcal{Y}}\|_{b,\infty} = 1. \tag{A.24}$$

APPENDIX B

Background study: Distributed diffusion LMS strategies

In this appendix, we review the well-studied diffusion LMS strategies over a single-task MSE network by demonstrating first how diffusion algorithms can be derived by minimizing a mean-square error function and then briefly review the mean and mean-square convergence. A useful overview of diffusion LMS strategies appears in [Sayed, 2014c].

B.1 Diffusion LMS strategies over single-task MSE network

Consider the single-task MSE network described in Section 2.2 and illustrated in Figure 2.1. In this section, we review the derivation of the distributed diffusion strategies for estimating $\boldsymbol{w}^o$ at each agent k by seeking to minimize the global cost function (2.9). The derivation is based on a completion-of-squares argument, followed by a stochastic approximation step, and an incremental approximation step [Cattivelli and Sayed, 2010].

In the following, we first show how the global cost (2.9) can be approximated by an alternative local cost that is amenable to distributed optimization. Then, each agent will optimize the alternative cost via a stochastic gradient method. We start by introducing a set of non-negative coefficients $\{c_{\ell k}\}$ that satisfy the following conditions:

$$\sum_{k=1}^{N} c_{\ell k} = 1, \quad \text{and} \quad c_{\ell k} = 0 \quad \text{if} \quad k \notin \mathcal{N}_\ell. \tag{B.1}$$

These coefficients are free parameters that are chosen by the designer and if we collect these coefficients into a matrix $\boldsymbol{C} = [c_{\ell k}]$, we obtain a right-stochastic matrix $\boldsymbol{C}$. Using these coefficients, we associate with each agent k, a local cost function of the following form:

$$J_k^{\text{loc}}(\boldsymbol{w}) \triangleq \sum_{\ell \in \mathcal{N}_k} c_{\ell k} J_\ell(\boldsymbol{w}). \tag{B.2}$$

This cost consists of a weighted combination of the individual costs at the neighbors of agent k. Since the $\{c_{\ell k}\}$ are nonnegative and $J_\ell(\boldsymbol{w})$ are strictly convex, then $J_k^{\rm loc}(\boldsymbol{w})$ is strictly convex and minimized at $\boldsymbol{w}^o$. Note that each individual cost $J_k(\boldsymbol{w})$ in (2.2) can be factored via a completion-of-squares argument and written as [Sayed, 2014c]:

$$J_k(\boldsymbol{w}) = J_k(\boldsymbol{w}^o) + \|\boldsymbol{w} - \boldsymbol{w}^o\|^2_{\boldsymbol{R}_{x,k}}. \tag{B.3}$$

Using the alternative representation (B.3), we can re-express the local cost $J_k^{\rm loc}(\boldsymbol{w})$ as:

$$J_k^{\rm loc}(\boldsymbol{w}) = J_{k,\min}^{\rm loc} + \|\boldsymbol{w} - \boldsymbol{w}^o\|^2_{\boldsymbol{R}_k}, \tag{B.4}$$

where

$$J_{k,\min}^{\rm loc} = \sum_{\ell\in\mathcal{N}_k} c_{\ell k} J_\ell(\boldsymbol{w}^o), \tag{B.5}$$

$$\boldsymbol{R}_k = \sum_{\ell\in\mathcal{N}_k} c_{\ell k} \boldsymbol{R}_{x,\ell}. \tag{B.6}$$

The global cost (2.9) can be expressed as follows:

$$J^{\rm glob}(\boldsymbol{w}) = \sum_{\ell=1}^N J_\ell(\boldsymbol{w}) = \sum_{\ell=1}^N \left(\sum_{k=1}^N c_{\ell k}\right) J_\ell(\boldsymbol{w}) = \sum_{k=1}^N \left(\sum_{\ell=1}^N c_{\ell k} J_\ell(\boldsymbol{w})\right) = \sum_{k=1}^N J_k^{\rm loc}(\boldsymbol{w}), \tag{B.7}$$

or equivalently as:

$$J^{\rm glob}(\boldsymbol{w}) = J_k^{\rm loc}(\boldsymbol{w}) + \sum_{\ell\neq k} J_\ell^{\rm loc}(\boldsymbol{w}). \tag{B.8}$$

Using the alternative representation (B.4), we can write:

$$J^{\rm glob}(\boldsymbol{w}) = J_k^{\rm loc}(\boldsymbol{w}) + \sum_{\ell\neq k} \|\boldsymbol{w} - \boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell} + \sum_{\ell\neq k} J_{\ell,\min}^{\rm loc}. \tag{B.9}$$

Since the last term in the above equation is independent of $\boldsymbol{w}$, minimizing $J^{\rm glob}(\boldsymbol{w})$ over $\boldsymbol{w}$ is equivalent to minimizing the following alternative global cost:

$$J^{\rm glob'}(\boldsymbol{w}) = J_k^{\rm loc}(\boldsymbol{w}) + \sum_{\ell\neq k} \|\boldsymbol{w} - \boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell}. \tag{B.10}$$

The second term on the RHS of the above equation implies that how by incorporating the quadratic parts, the newly-introduced local cost function $J_k^{\rm loc}(\boldsymbol{w})$ can be corrected to the global cost $J^{\rm glob}(\boldsymbol{w})$. However, the minimizer $\boldsymbol{w}^o$ appearing in the quadratic part is unknown since the agents wish to determine its value. Likewise, not all the weighting matrices $\boldsymbol{R}_\ell$ are available to agent k; only those from its neighbors can be assumed available. Expression (B.10) motivates us to introduce a newly localized cost function at agent k that is close enough to the desired $J^{\rm glob}(\boldsymbol{w})$ and which can be minimized through local cooperation. We denote this localized cost by $J_k^{\rm dist}(\boldsymbol{w})$; it is obtained by limiting the summation on the RHS of (B.10) to the neighbors of agent k, namely,

$$J_k^{\rm dist}(\boldsymbol{w}) = J_k^{\rm loc}(\boldsymbol{w}) + \sum_{\ell\in\mathcal{N}_k\backslash\{k\}} \|\boldsymbol{w} - \boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell}. \tag{B.11}$$

The cost functions $J_k^{\rm loc}(\boldsymbol{w})$ and $J_k^{\rm dist}(\boldsymbol{w})$ are both associated with agent k; the difference between them is that expression for the latter is closer to the global cost (B.10) that we want to optimize.

The weighting matrices $\boldsymbol{R}_\ell$ in (B.14) may not be available in practice. Usually, agents can only observe realizations $\boldsymbol{x}_\ell(i)$ of regressors arising from distributions whose covariances are $\boldsymbol{R}_{x,\ell}$. One way to address this issue is to replace each of the weighted norms $\|\boldsymbol{w}-\boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell}$ by a scaled multiple of the unweighted norm:

$$\|\boldsymbol{w}-\boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell} \approx b_{\ell k}\|\boldsymbol{w}-\boldsymbol{w}^o\|^2, \tag{B.12}$$

where $b_{\ell k}$ is some nonnegative coefficient. This substitution amounts to having each agent k approximates the $\{\boldsymbol{R}_\ell\}$ from its neighbors by multiples of the identity matrix, i.e., $\boldsymbol{R}_\ell \approx b_{\ell k}\boldsymbol{I}_M$. This approximation is reasonable because using the Rayleigh-Ritz characterization of eigenvalues, it holds that [Sayed, 2014c]:

$$\lambda_{\min}(\boldsymbol{R}_\ell)\cdot\|\boldsymbol{w}-\boldsymbol{w}^o\|^2 \le \|\boldsymbol{w}-\boldsymbol{w}^o\|^2_{\boldsymbol{R}_\ell} \le \lambda_{\max}(\boldsymbol{R}_\ell)\cdot\|\boldsymbol{w}-\boldsymbol{w}^o\|^2. \tag{B.13}$$

As the derivations will show, the scalars $b_{\ell k}$ will end up being embedded into another set of coefficients $a_{\ell k}$ that will be selected by the designer. Thus, we replace (B.14) by:

$$J_k^{\rm dist'}(\boldsymbol{w}) = J_k^{\rm loc}(\boldsymbol{w}) + \sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}\|\boldsymbol{w}-\boldsymbol{w}^o\|^2. \tag{B.14}$$

With the exception of the minimizer $\boldsymbol{w}^o$, this alternative cost at agent relies solely on information that is available to agent k from its neighborhood. Now, agent k can apply a steepest-descent iteration to minimize its localized cost $J_k^{\rm dist'}(\boldsymbol{w})$. The iteration would take the following form:

$$\begin{aligned}\boldsymbol{w}_k(i+1) &= \boldsymbol{w}_k(i) - \mu_k\nabla_{\boldsymbol{w}}J_k^{\rm dist'}(\boldsymbol{w}_k(i))\\ &= \boldsymbol{w}_k(i) + \mu_k\sum_{\ell\in\mathcal{N}_k} c_{\ell k}(\boldsymbol{r}_{dx,\ell}-\boldsymbol{R}_{x,\ell}\boldsymbol{w}_k(i)) + \mu_k\sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}(\boldsymbol{w}^o-\boldsymbol{w}_k(i)).\end{aligned} \tag{B.15}$$

The step-size μ_k can be constant or time variant. Constant step-sizes allow the resulting strategies to learn and adapt continuously, while time variant step-sizes that decay to zero turn off the learning abilities of the algorithm. An adaptive implementation of (B.15) can be obtained by replacing the moments $\{\boldsymbol{r}_{dx,\ell},\boldsymbol{R}_{x,\ell}\}$ by the instantaneous approximations (2.6). Doing so leads to the following recursion:

$$\boldsymbol{w}_k(i+1) = \boldsymbol{w}_k(i) + \mu_k\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)(d_\ell(i)-\boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)) + \mu_k\sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}(\boldsymbol{w}^o-\boldsymbol{w}_k(i)). \tag{B.16}$$

According to (B.16), the update from $\boldsymbol{w}_k(i)$ to $\boldsymbol{w}_k(i+1)$ involves two correction terms. However, the last correction term still depends on the unknown minimizer $\boldsymbol{w}^o$. We can now use incremental-type arguments to replace $\boldsymbol{w}^o$ in (B.16) by suitable approximations for it.

Thus, note first that there are two correction terms on the RHS of (B.16) and these terms can be added one at a time. For example, we can achieve (B.16) by splitting the update into the

following two steps involving an intermediate estimate $\boldsymbol{\psi}_k(i+1)$:

$$\begin{aligned}\boldsymbol{\psi}_k(i+1) &= \boldsymbol{w}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)) \\ \boldsymbol{w}_k(i+1) &= \boldsymbol{w}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}(\boldsymbol{w}^o - \boldsymbol{w}_k(i)).\end{aligned} \tag{B.17}$$

Since each agent ℓ has a readily available approximation for $\boldsymbol{w}^o$, which is its local intermediate estimate $\boldsymbol{\psi}_\ell(i+1)$, we replace $\boldsymbol{w}^o$ in the second step of (B.17) by $\boldsymbol{\psi}_\ell(i+1)$. Second, since $\boldsymbol{\psi}_k(i+1)$ at agent k is generally a better estimate for $\boldsymbol{w}^o$ than $\boldsymbol{w}_k(i)$, we further replace $\boldsymbol{w}_k(i)$ in the second step of (B.17) by $\boldsymbol{\psi}_k(i+1)$. With these replacements, the second step of (B.17) becomes:

$$\boldsymbol{w}_k(i+1) = \boldsymbol{\psi}_k(i+1) - \mu_k \sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}(\boldsymbol{\psi}_k(i+1) - \boldsymbol{\psi}_\ell(i+1)). \tag{B.18}$$

We can rewrite the previous step in a more compact and revealing form by introducing the following coefficients:

$$a_{\ell k} = \begin{cases} 1 - \mu_k \sum_{\ell\in\mathcal{N}_k\setminus\{k\}} b_{\ell k}, & \text{if } \ell = k \\ \mu_k b_{\ell k}, & \text{if } \ell \in \mathcal{N}_k \setminus \{k\} \\ 0, & \text{otherwise} \end{cases} \tag{B.19}$$

Observe that, for sufficiently small step-sizes, these coefficients are non negative and satisfy the following conditions:

$$a_{\ell k} \geq 0, \qquad \sum_{\ell=1}^{N} a_{\ell k} = 1, \quad \text{and} \quad a_{\ell k} = 0 \text{ if } \ell \notin \mathcal{N}_k. \tag{B.20}$$

If we collect the combination coefficients $a_{\ell k}$ into an $N \times N$ matrix $\boldsymbol{A} \triangleq [a_{\ell k}]$, which we call the network combination matrix, then it follows from (B.20) that this matrix is a left-stochastic matrix. Using the coefficients $\{a_{\ell k}\}$ so defined, we arrive at the following alternative compact form for (B.17), known as the adapt-then-combine (ATC) diffusion LMS strategy:

$$\begin{aligned}\boldsymbol{\psi}_k(i+1) &= \boldsymbol{w}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{w}_k(i)) \\ \boldsymbol{w}_k(i+1) &= \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_\ell(i+1).\end{aligned} \tag{B.21}$$

The ATC diffusion consists of two steps. The first step is an adaptation step where agent k uses its own data and the data received from its neighbors to update its weight $\boldsymbol{w}_k(i)$ to an intermediate value $\boldsymbol{\psi}_k(i+1)$. The second step is a combination step where the intermediate estimate $\{\boldsymbol{\psi}_\ell(i)\}$ from the neighborhood of agent k are combined through the combination coefficients $\{a_{\ell k}\}$ to obtain the updated weight estimate $\boldsymbol{w}_k(i+1)$.

If we return back to (B.16) and add the second correction term first, and following a similar procedure as in the ATC diffusion, we arrive at the other variant of diffusion algorithm, known

as combine-then-adapt (CTA) diffusion LMS strategy:

$$\begin{aligned}\boldsymbol{\psi}_k(i) &= \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{w}_\ell(i)\\ \boldsymbol{w}_k(i+1) &= \boldsymbol{\psi}_k(i) + \mu_k \sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{x}_\ell(i)(d_\ell(i) - \boldsymbol{x}_\ell^\top(i)\boldsymbol{\psi}_k(i)).\end{aligned} \tag{B.22}$$

By comparing the ATC and CTA strategy, we note that the order of the combination and adaptation steps is reversed.

B.2 Performance analysis

The mean and mean-square performance of the diffusion algorithms over single-task MSE networks have been studied in details in [Cattivelli and Sayed, 2010, Sayed, 2014c]. In this section, we briefly review the performance analysis of the ATC diffusion LMS strategies.

B.2.1 Weight error vector recursion

We define the local weight error vector as:

$$\widetilde{\boldsymbol{w}}_k(i) \triangleq \boldsymbol{w}^o - \boldsymbol{w}_k(i). \tag{B.23}$$

We collect all error vectors and step-sizes across the network into a block vector and block diagonal matrix:

$$\widetilde{\boldsymbol{w}}_b(i) \triangleq \mathrm{col}\{\widetilde{\boldsymbol{w}}_1(i),\ldots,\widetilde{\boldsymbol{w}}_N(i)\}, \tag{B.24}$$

$$\boldsymbol{\mathcal{M}} \triangleq \mathrm{diag}\{\mu_1\boldsymbol{I}_M,\ldots,\mu_N\boldsymbol{I}_M\}. \tag{B.25}$$

We further introduce the extended combination matrix $\boldsymbol{\mathcal{A}} \triangleq \boldsymbol{A}\otimes\boldsymbol{I}_M$. Starting from (B.21), using the model (2.1), the network weight error vector $\widetilde{\boldsymbol{w}}(i+1)$ can be found to evolve according to the following recursion:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{A}}^\top(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i))\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xz}(i), \tag{B.26}$$

where $\boldsymbol{\mathcal{R}}_x(i)$ is a block diagonal matrix and $\boldsymbol{p}_{xz}(i)$ is a block column vector:

$$\boldsymbol{\mathcal{R}}_x(i) \triangleq \mathrm{diag}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{x}_\ell(i)\boldsymbol{x}_\ell^\top(i),\ldots,\sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{x}_\ell(i)\boldsymbol{x}_\ell^\top(i)\right\} \tag{B.27}$$

$$\boldsymbol{p}_{xz}(i) \triangleq \mathrm{col}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{x}_\ell(i)z_\ell(i),\ldots,\sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{x}_\ell(i)z_\ell(i)\right\}. \tag{B.28}$$

Recursion (B.26) can be described by a more compact recursion of the form:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}(i)\widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{g}(i), \tag{B.29}$$

where

$$\boldsymbol{\mathcal{B}}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x(i)) \tag{B.30}$$

$$\boldsymbol{g}(i) \triangleq \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{p}_{xz}(i). \tag{B.31}$$

The matrix $\boldsymbol{\mathcal{B}}(i)$ controls the evolution dynamics of the network error vector $\widetilde{\boldsymbol{w}}_b(i)$.

B.2.2 Mean error behavior

Taking the expectation of both sides of recursion (B.29) and using Assumption 2.1, we find:

$$\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{B}}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i), \tag{B.32}$$

where

$$\boldsymbol{\mathcal{B}} \triangleq \boldsymbol{\mathcal{A}}^{\top}(\boldsymbol{I}_{MN} - \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{R}}_x), \tag{B.33}$$

with

$$\boldsymbol{\mathcal{R}}_x \triangleq \operatorname{diag}\left\{\sum_{\ell\in\mathcal{N}_1} c_{\ell 1}\boldsymbol{R}_{x,\ell}, \ldots, \sum_{\ell\in\mathcal{N}_N} c_{\ell N}\boldsymbol{R}_{x,\ell}\right\}. \tag{B.34}$$

The necessary and sufficient condition to ensure $\lim_{i\to\infty}\mathbb{E}\,\widetilde{\boldsymbol{w}}_b(i) = \boldsymbol{0}$ is therefore to select the step-sizes $\{\mu_k\}$ that ensure that the matrix $\boldsymbol{\mathcal{B}}$ is contractive, i.e., $\rho(\boldsymbol{\mathcal{B}}) < 1$. From (A.22) and (A.23), it follows that choosing the step-sizes according to:

$$0 < \mu_k < \frac{2}{\lambda_{\max}\left(\sum_{\ell\in\mathcal{N}_k} c_{\ell k}\boldsymbol{R}_{x,\ell}\right)}, \quad k = 1, \ldots, N, \tag{B.35}$$

will guarantee $\rho(\boldsymbol{\mathcal{B}}) < 1$. An interesting observation that follows from (B.35) is that the stability range of diffusion does not depend on the matrix $\boldsymbol{A}$.

B.2.3 Mean-square-error behavior

From (B.29) and under Assumption 2.1, we obtain the following variance relation:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\Sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\Sigma}'} + \mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}}, \tag{B.36}$$

where $\boldsymbol{\Sigma} \geq \boldsymbol{0}$ is a weighting matrix that we are free to choose, and

$$\boldsymbol{\Sigma}' \triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^{\top}(i)\boldsymbol{\Sigma}\boldsymbol{\mathcal{B}}(i)\}. \tag{B.37}$$

The second term on the RHS of (B.36) can be evaluated as follows:

$$\mathbb{E}\,\|\boldsymbol{g}(i)\|^2_{\boldsymbol{\Sigma}} = \operatorname{Tr}(\boldsymbol{\Sigma}\boldsymbol{\mathcal{G}}), \tag{B.38}$$

where $\boldsymbol{\mathcal{G}}$ is given by:

$$\boldsymbol{\mathcal{G}} \triangleq \mathbb{E}\,\{\boldsymbol{g}(i)\boldsymbol{g}^{\top}(i)\} = \boldsymbol{\mathcal{A}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{C}}^{\top}\operatorname{diag}\{\sigma^2_{z,1}\boldsymbol{R}_{x,1}, \ldots, \sigma^2_{z,N}\boldsymbol{R}_{x,N}\}\boldsymbol{\mathcal{C}}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{A}}. \tag{B.39}$$

Introducing $\boldsymbol{\sigma} \triangleq \text{vec}(\boldsymbol{\Sigma})$ and $\boldsymbol{\sigma}' \triangleq \text{vec}(\boldsymbol{\Sigma}')$, and using property (A.6), we can write:

$$\boldsymbol{\sigma}' = \boldsymbol{\mathcal{F}}\boldsymbol{\sigma}, \tag{B.40}$$

where

$$\boldsymbol{\mathcal{F}} \triangleq \mathbb{E}\,\{\boldsymbol{\mathcal{B}}^{\top}(i) \otimes \boldsymbol{\mathcal{B}}^{\top}(i)\}. \tag{B.41}$$

Using property (A.5), the variance relation (B.36) can be written more compactly as:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i+1)\|^2_{\boldsymbol{\sigma}} = \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\mathcal{F}}\boldsymbol{\sigma}} + \boldsymbol{y}^{\top}\boldsymbol{\sigma}, \tag{B.42}$$

where we used the notation $\|\boldsymbol{x}\|^2_{\boldsymbol{\sigma}}$ as a short form for $\|\boldsymbol{x}\|^2_{\boldsymbol{\Sigma}}$ and where

$$\boldsymbol{y} \triangleq \text{vec}(\boldsymbol{\mathcal{G}}). \tag{B.43}$$

A necessary and sufficient condition for mean-square stability of the network[1] is to select the step-sizes $\{\mu_k\}$ such that the matrix $\boldsymbol{\mathcal{F}}$ is contractive. A simpler condition for mean-square stability can be obtained by assuming sufficiently small step-sizes where the matrix $\boldsymbol{\mathcal{F}}$ in (B.41) can be approximated as follows [Sayed, 2014c]:

$$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{B}}^{\top} \otimes \boldsymbol{\mathcal{B}}^{\top}, \qquad \text{(small step-sizes)}. \tag{B.44}$$

Hence, sufficiently small step-sizes that satisfy (B.35) will also ensure mean-square stability.

Assume that the step-sizes are sufficiently small so that condition (B.35) holds and the network is stable in the mean and mean-square sense. Under this condition, the network achieves its steady-state operation. The convergence rate of the network determines the rate at which the quantity $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}(i)\|^2_{\boldsymbol{\Sigma}}$ converges towards its steady state value. This convergence rate is determined by the spectral radius of the matrix $\boldsymbol{\mathcal{F}}$, i.e., $\rho(\boldsymbol{\mathcal{F}})$, or approximately, $[\rho(\boldsymbol{\mathcal{B}})]^2$ for sufficiently small step-sizes. The smaller the spectral radius is, the faster the convergence is. Under certain simplifying assumptions, it can be shown that the spectral radius of the matrix $\boldsymbol{\mathcal{B}}$ in diffusion is smaller than the non-cooperative one (the matrix $\boldsymbol{\mathcal{B}}$ in the non-cooperative case is obtained by setting $\boldsymbol{A} = \boldsymbol{C} = \boldsymbol{I}_N$) (see [Sayed, 2014c, Sections 5.3 and 6.4]).

To obtain the steady-state mean-square error $\zeta^{\star} \triangleq \lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}(i)\|^2_{\boldsymbol{\sigma}_{ss}}$, we let $i \to \infty$ and use expression (B.42) to write:

$$\lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{(\boldsymbol{I}-\boldsymbol{\mathcal{F}})\boldsymbol{\sigma}} = \boldsymbol{y}^{\top}\boldsymbol{\sigma}. \tag{B.45}$$

In order to recover $\zeta^{\star}$, it is sufficient to replace $\boldsymbol{\sigma}$ in the above expression by $(\boldsymbol{I} - \boldsymbol{\mathcal{F}})^{-1}\boldsymbol{\sigma}_{ss}$.

[1] An adaptive algorithm is mean-square stable if $\lim_{i\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_b(i)\|^2_{\boldsymbol{\sigma}} \to a_o$, where a_o is a positive real number.

APPENDIX C

Bernoulli model and derivations in Chapter 3

C.1 Proof of Property 3.3

Consider node k and its fixed cluster $\mathcal{C}(k)$. Let Ω_s denote the sample spaces of the intra-cluster neighborhood $\mathcal{N}_k(i) \cap \mathcal{C}(k)$. In the following, we prove that the intra-cluster neighborhood $\mathcal{N}_k \cap \mathcal{C}(k)$ defined by the mean matrix $\overline{\boldsymbol{A}}$ is equal to the union of all possible realizations of the random neighborhood $\mathcal{N}_k(i) \cap \mathcal{C}(k)$, namely,

$$\mathcal{N}_k \cap \mathcal{C}(k) = \bigcup_{\omega \in \Omega_s} \mathcal{N}_k(i, \omega) \cap \mathcal{C}(k). \tag{C.1}$$

First we establish that:

$$\left(\bigcup_{\omega \in \Omega} \mathcal{N}_k(i, \omega) \cap \mathcal{C}(k) \right) \subseteq \mathcal{N}_k \cap \mathcal{C}(k). \tag{C.2}$$

For all $\ell \in \left(\bigcup_{\omega \in \Omega} \mathcal{N}_k(i, \omega) \cap \mathcal{C}(k) \right)$, by condition (3.11), we have $a_{\ell k}(i) > 0$. If the probability of the event $a_{\ell k}(i) > 0$ is non-zero, then $\overline{a}_{\ell k} > 0$. This implies that $\ell \in \mathcal{N}_k \cap \mathcal{C}(k)$ for all ℓ, and $\mathcal{N}_k(i) \cap \mathcal{C}(k) \subseteq \mathcal{N}_k \cap \mathcal{C}(k)$. This relation holds for any realization of $\mathcal{N}_k(i) \cap \mathcal{C}(k)$, so we have (C.2).

Now we have to establish that:

$$\mathcal{N}_k \cap \mathcal{C}(k) \subseteq \left(\bigcup_{\omega \in \Omega} \mathcal{N}_k(i, \omega) \cap \mathcal{C}(k) \right) \tag{C.3}$$

For any $\ell \in \mathcal{N}_k \cap \mathcal{C}(k)$, we have $\overline{a}_{\ell k} > 0$. This assertion is true if, and only if, there exists at least one realization $\mathcal{N}_k(i, \omega) \cap \mathcal{C}(k)$ where $a_{\ell k}(i) > 0$, that is, $\ell \in \mathcal{N}_k(i, \omega) \cap \mathcal{C}(k)$. This leads to (C.3).

In a similar manner, we establish that the inter-cluster neighborhood $\mathcal{N}_k \setminus \mathcal{C}(k)$ defined by the mean matrix $\overline{\boldsymbol{P}}_\rho$ is equal to the union of all possible realizations of the random neighborhood $\mathcal{N}_k(i) \setminus \mathcal{C}(k)$.

C.2 The Bernoulli model

In this model, the step-sizes $\{\mu_k(i)\}$ are distributed as follows:

$$\mu_k(i) = \begin{cases} \mu_k, & \text{with probability } p_{\mu,k} \\ 0, & \text{with probability } 1 - p_{\mu,k} \end{cases} \tag{C.4}$$

where μ_k is a fixed value. This probability distribution allows us to model random "on-off" behavior by each agent k due to power saving strategies or random agent failures. We assume that the step-sizes $\mu_k(i)$ are spatially uncorrelated for different k. At each iteration i, the mean of the step-size $\mu_k(i)$ is $\bar{\mu}_k = \mu_k p_{\mu,k}$, and the covariance between $\mu_k(i)$ and $\mu_\ell(i)$ is:

$$c_{\mu,k,\ell} \triangleq \mathbb{E}\{(\mu_k(i) - \bar{\mu}_k)(\mu_\ell(i) - \bar{\mu}_\ell)\} = \begin{cases} \mu_k^2 p_{\mu,k}(1 - p_{\mu,k}), & \text{if } \ell = k \\ 0, & \text{otherwise.} \end{cases} \tag{C.5}$$

Furthermore, combination weights $\{a_{\ell k}(i)\}$ are distributed as follows:

$$a_{\ell k}(i) = \begin{cases} a_{\ell k}, & \text{with probability } p_{a,\ell k} \\ 0, & \text{with probability } 1 - p_{a,\ell k} \end{cases} \tag{C.6}$$

for any $\ell \in \mathcal{N}_k^-(i) \cap \mathcal{C}(k)$, where $0 < a_{\ell k} < 1$ is a fixed coefficient. The coefficients $\{a_{\ell k}(i)\}$ are spatially uncorrelated for different ℓ and k. Node k adjusts its own combination coefficient to ensure that the sum of its neighboring coefficients is equal to one as follows:

$$a_{kk}(i) = 1 - \sum_{\ell \in \mathcal{N}_k^-(i) \cap \mathcal{C}(k)} a_{\ell k}(i) \geq 0. \tag{C.7}$$

The probability distribution (C.6) allows us to model a random "on-off" status for links within clusters at time i due to communication cost saving strategies or random link failures. With this model, we are giving the opportunity to each agent k to randomly choose a subset of neighbors that belong to its cluster to perform the combination step. At each iteration i, the mean of the coefficient $a_{\ell k}(i)$ is given by:

$$\bar{a}_{\ell k} = \begin{cases} a_{\ell k} p_{a,\ell k}, & \text{if} \quad \ell \in \mathcal{N}_k^- \cap \mathcal{C}(k) \\ 1 - \sum\limits_{\ell \in \mathcal{N}_k^- \cap \mathcal{C}(k)} a_{\ell k} p_{a,\ell k}, & \text{if} \quad \ell = k \\ 0, & \text{otherwise.} \end{cases} \tag{C.8}$$

and the covariance between $a_{\ell k}(i)$ and $a_{nm}(i)$ equals:

$$\begin{aligned} c_{a,\ell k,nm} &= \mathbb{E}\{(a_{\ell k}(i) - \overline{a}_{\ell k})(a_{nm}(i) - \overline{a}_{nm})\} \\ &= \begin{cases} c_{a,\ell k,\ell k}, & \text{if } k = m, \ell = n, \ell \in \mathcal{N}_k^- \cap \mathcal{C}(k) \\ -c_{a,\ell k,\ell k}, & \text{if } k = m = n, \ell \in \mathcal{N}_k^- \cap \mathcal{C}(k) \\ -c_{a,nk,nk}, & \text{if } k = m = \ell, n \in \mathcal{N}_k^- \cap \mathcal{C}(k) \\ \sum\limits_{j \in \mathcal{N}_k^- \cap \mathcal{C}(k)} c_{a,jk,jk}, & \text{if } k = m = \ell = n \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{C.9}$$

where $c_{a,\ell k,\ell k} = a_{\ell k}^2 p_{a,\ell k}(1-p_{a,\ell k})$.

Finally, the regularization factors $\{\rho_{k\ell}(i)\}$ are distributed as follows:

$$\rho_{k\ell}(i) = \begin{cases} \rho_{k\ell}, & \text{with probability } p_{\rho,k\ell} \\ 0, & \text{with probability } 1-p_{\rho,k\ell} \end{cases} \tag{C.10}$$

for any $\ell \in \mathcal{N}_k(i) \setminus \mathcal{C}(k)$, where $0 < \rho_{k\ell} < 1$ is a fixed regularization factor. The factors $\{\rho_{k\ell}(i)\}$ are spatially uncorrelated for $k \neq \ell$. At each iteration i, in order to get a right stochastic matrix $\boldsymbol{P}_\rho(i)$, node k adjusts its regularization factor as follows:

$$\rho_{kk}(i) = 1 - \sum_{\ell \in \mathcal{N}_k(i) \setminus \mathcal{C}(k)} \rho_{k\ell}(i) \; \geq 0. \tag{C.11}$$

The probability distribution (C.10) allows each agent k to randomly select a subset of neighbors that do not belong to its cluster and introduce co-regularization in the estimation process. This behavior can also be interpreted as resulting from link random failures between neighboring clusters: at every time instant i, the communication link from agent ℓ to agent k drops with probability $1-p_{\rho,k\ell}$. The mean of $\rho_{k\ell}(i)$ is given by:

$$\overline{\rho}_{k\ell} = \begin{cases} \rho_{k\ell} p_{\rho,k\ell}, & \text{if } \ell \in \mathcal{N}_k \setminus \mathcal{C}(k) \\ 1 - \sum\limits_{\ell \in \mathcal{N}_k \setminus \mathcal{C}(k)} \rho_{k\ell} p_{\rho,k\ell}, & \text{if } \ell = k \\ 0, & \text{otherwise,} \end{cases} \tag{C.12}$$

and the covariance between $\rho_{k\ell}(i)$ and $\rho_{mn}(i)$ is:

$$\begin{aligned} c_{\rho,k\ell,mn} &= \mathbb{E}\{(\rho_{k\ell}(i) - \overline{\rho}_{k\ell})(\rho_{mn}(i) - \overline{\rho}_{mn})\} \\ &= \begin{cases} c_{\rho,k\ell,k\ell}, & \text{if } k=m, \ell=n, \ell \in \mathcal{N}_k \setminus \mathcal{C}(k) \\ -c_{\rho,k\ell,k\ell}, & \text{if } k=m=n, \ell \in \mathcal{N}_k \setminus \mathcal{C}(k) \\ -c_{\rho,kn,kn}, & \text{if } k=m=\ell, n \in \mathcal{N}_k \setminus \mathcal{C}(k) \\ \sum\limits_{j \in \mathcal{N}_k \setminus \mathcal{C}(k)} c_{\rho,kj,kj}, & \text{if } k=m=\ell=n \\ 0, & \text{otherwise} \end{cases} \end{aligned} \tag{C.13}$$

where $c_{\rho,k\ell,k\ell} = \rho_{k\ell}^2 p_{\rho,k\ell}(1-p_{\rho,k\ell})$.

C.3 Stability of the matrix $\boldsymbol{\mathcal{F}}$

Recall from (3.62) that

$$\boldsymbol{\mathcal{F}} \approx \boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top} \big[\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) \otimes_b \boldsymbol{I}_{MN} \big]. \tag{C.14}$$

We now upper-bound the spectral radius of $\boldsymbol{\mathcal{F}}$ in order to derive a sufficient condition for mean-square stability of the algorithm. Since any induced matrix norm is lower bounded by its spectral radius, we can write in terms of the block maximum norm defined in Appendix A.3:

$$\rho(\boldsymbol{\mathcal{F}}) \leq \|\boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\|_{b,\infty} \cdot \|\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{\mathcal{Q}}) \otimes_b \boldsymbol{I}_{MN}\|_{b,\infty}. \tag{C.15}$$

Since the matrix $\boldsymbol{\mathcal{A}}_{\mathrm{I}}$ is a block left-stochastic matrix, we know from (A.23) that $\|\boldsymbol{\mathcal{A}}_{\mathrm{I}}^{\top}\|_{b,\infty} = 1$. Using (3.44) and the triangular inequality, we have:

$$\begin{aligned}\rho(\boldsymbol{\mathcal{F}}) \;\leq\; & \|\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) \otimes_b \boldsymbol{I}_{MN}\|_{b,\infty} + \\ & \eta \|\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_\rho\|_{b,\infty} + \eta \|\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_\rho \otimes_b \boldsymbol{I}_{MN}\|_{b,\infty}. \end{aligned} \tag{C.16}$$

Consider the second term on the RHS of (C.16). We know that

$$\begin{aligned}\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_\rho \;&\overset{(A.14)}{=}\; (\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}})(\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{P}}_\rho) \\ &\overset{(A.15)}{=}\; \Big((\boldsymbol{I}_N \otimes \overline{\boldsymbol{M}}) \otimes \boldsymbol{I}_{M^2}\Big)\Big((\boldsymbol{I}_N \otimes \overline{\boldsymbol{P}}_\rho) \otimes \boldsymbol{I}_{M^2}\Big). \end{aligned} \tag{C.17}$$

Since $\big((\boldsymbol{I}_N \otimes \overline{\boldsymbol{P}}_\rho) \otimes \boldsymbol{I}_{M^2}\big)$ is a block right-stochastic matrix and $\big((\boldsymbol{I}_N \otimes \overline{\boldsymbol{M}}) \otimes \boldsymbol{I}_{M^2}\big)$ is an $N^2 \times N^2$ block diagonal matrix with each block of the form $\bar{\mu}_k \boldsymbol{I}_{M^2}$ $(k = 1, \ldots, N)$, we obtain from relations (A.24) and (A.22):

$$\begin{aligned}\|\boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_\rho\|_{b,\infty} \;&\leq\; \|(\boldsymbol{I}_N \otimes \overline{\boldsymbol{M}}) \otimes \boldsymbol{I}_{M^2}\|_{b,\infty} \cdot \|(\boldsymbol{I}_N \otimes \overline{\boldsymbol{P}}_\rho) \otimes \boldsymbol{I}_{M^2}\|_{b,\infty} \\ &=\; \max_{1 \leq k \leq N} \overline{\mu}_k \end{aligned} \tag{C.18}$$

Following the same steps for the third term on the RHS of (C.16), we have:

$$\|\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{P}}_\rho \otimes_b \boldsymbol{I}_{MN}\|_{b,\infty} \leq \max_{1 \leq k \leq N} \overline{\mu}_k. \tag{C.19}$$

The matrix $\big[\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) \otimes_b \boldsymbol{I}_{MN}\big]$ in the first term on the RHS of (C.16) is an $N^2 \times N^2$ block diagonal matrix. The m-th block on the diagonal (where $m = (\ell - 1)N + k$ for $k, \ell = 1, \ldots, N$) is of size $M^2 \times M^2$, symmetric, and has the following form:

$$\begin{aligned}&\boldsymbol{I}_{M^2} - \boldsymbol{I}_M \otimes \bar{\mu}_k(\boldsymbol{R}_{x,k} + \eta \boldsymbol{I}_M) - \bar{\mu}_\ell(\boldsymbol{R}_{x,\ell} + \eta \boldsymbol{I}_M) \otimes \boldsymbol{I}_M \\ &= (-\bar{\mu}_\ell \boldsymbol{R}_{x,\ell} - \eta \bar{\mu}_\ell \boldsymbol{I}_M) \otimes \boldsymbol{I}_M + \boldsymbol{I}_M \otimes (\boldsymbol{I}_M - \bar{\mu}_k \boldsymbol{R}_{x,k} - \eta \bar{\mu}_k \boldsymbol{I}_M) \end{aligned} \tag{C.20}$$

Before proceeding, let us recall the Kronecker sum operator, denoted by $\oplus$. If $\boldsymbol{X}$ and $\boldsymbol{Y}$ are two matrices of dimension $M \times M$ each, then

$$\boldsymbol{X} \oplus \boldsymbol{Y} \triangleq \boldsymbol{X} \otimes \boldsymbol{I}_M + \boldsymbol{I}_M \otimes \boldsymbol{Y}. \tag{C.21}$$

Let $\lambda_k(\cdot)$ denote the k-th eigenvalue of its matrix argument. Then, the eigenvalues of $\boldsymbol{X} \oplus \boldsymbol{Y}$ are of the form $\lambda_i(\boldsymbol{X}) + \lambda_j(\boldsymbol{Y})$ for $i,\ j = 1, \ldots, M$ [Bernstein, 2005]. Note that the RHS of equation (C.20) can be written as:

$$(-\bar{\mu}_\ell \boldsymbol{R}_{x,\ell} - \eta \bar{\mu}_\ell \boldsymbol{I}_M) \oplus (\boldsymbol{I}_M - \bar{\mu}_k \boldsymbol{R}_{x,k} - \eta \bar{\mu}_k \boldsymbol{I}_M) \tag{C.22}$$

and its eigenvalues are therefore of the form:

$$1 - \eta \bar{\mu}_k - \bar{\mu}_k \lambda_j(\boldsymbol{R}_{x,k}) - \eta \bar{\mu}_\ell - \bar{\mu}_\ell \lambda_i(\boldsymbol{R}_{x,\ell}) \tag{C.23}$$

for i, $j = 1, \ldots, M$ and k, $\ell = 1, \ldots, N$. In order to simplify the mean-square stability condition, we assume that the first order moment of the step-sizes is the same for all nodes. Using the fact that the block maximum norm of a block diagonal symmetric matrix is equal to the largest spectral radius of its block entries, namely, relation (A.22), we get:

$$
\begin{aligned}
&\|\boldsymbol{I}_{(MN)^2} - \boldsymbol{I}_{MN} \otimes_b \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) - \boldsymbol{\mathcal{M}}(\boldsymbol{\mathcal{R}}_x + \eta \boldsymbol{I}_{MN}) \otimes_b \boldsymbol{I}_{MN}\|_{b,\infty} \\
&= \max_{1 \leq k,\ell \leq N} \Big(\max_{1 \leq i,j \leq M} |1 - 2\eta\bar{\mu} - \bar{\mu}(\lambda_j(\boldsymbol{R}_{x,k}) + \lambda_i(\boldsymbol{R}_{x,\ell}))| \Big) \\
&= \max_{1 \leq k,\ell \leq N} \Big(\max_{1 \leq i,j \leq M} \left\{1 - 2\eta\bar{\mu} - \bar{\mu}(\lambda_j(\boldsymbol{R}_{x,k}) + \lambda_i(\boldsymbol{R}_{x,\ell})) \, , -1 + 2\eta\bar{\mu} + \bar{\mu}(\lambda_j(\boldsymbol{R}_{x,k}) + \lambda_i(\boldsymbol{R}_{x,\ell}))\right\} \Big) \\
&= \max \Big\{ 1 - 2\eta\bar{\mu} - \bar{\mu} \min_{k,\ell} \left(\lambda_{\min}(\boldsymbol{R}_{x,k}) + \lambda_{\min}(\boldsymbol{R}_{x,\ell})\right), \\
&\qquad\qquad - 1 + 2\eta\bar{\mu} + \bar{\mu} \max_{k,\ell} \left(\lambda_{\max}(\boldsymbol{R}_{x,k}) + \lambda_{\max}(\boldsymbol{R}_{x,\ell})\right) \Big\}. \qquad \text{(C.24)}
\end{aligned}
$$

The minimum (identically the maximum) on k and ℓ that appears in the last equality of (C.24) is reached for $k = \ell$. Thus, a sufficient condition for mean-square stability is given by:

$$
\max_{1 \leq k \leq N} \left(\max_{1 \leq i \leq M} |1 - 2\eta\bar{\mu} - 2\bar{\mu}\lambda_i(\boldsymbol{R}_{x,k})| + 2\eta\bar{\mu} \right) < 1, \qquad \text{(C.25)}
$$

which is verified if the first order moment of the step-sizes satisfies:

$$
0 < \bar{\mu} < \frac{1}{\max\limits_{1 \leq k \leq N} \lambda_{\max}(\boldsymbol{R}_{x,k}) + 2\eta}. \qquad \text{(C.26)}
$$

APPENDIX D

Derivations in Chapter 5

D.1 Projection matrix structure

We denote by $\boldsymbol{\mathcal{D}}_{e,p}$ the p-th block row in $\boldsymbol{\mathcal{D}}_e$ and by $[\boldsymbol{\mathcal{D}}_e]_{p,k_m}$ the $L_p \times M_k$ block of $\boldsymbol{\mathcal{D}}_{e,p}$ corresponding to the k_m-th sub-node. First, we show that the $M_k \times M_\ell$ (k_m, ℓ_n)-th block of the $N_e \times N_e$ block matrix $\boldsymbol{\mathcal{P}}_e$ in (5.53) is equal to:

$$[\boldsymbol{\mathcal{P}}_e]_{k_m,\ell_n} = \begin{cases} \boldsymbol{I}_{M_k} - [\boldsymbol{\mathcal{D}}_e]_{p,k_m}^\top \left(\boldsymbol{\mathcal{D}}_{e,p}\boldsymbol{\mathcal{D}}_{e,p}^\top\right)^{-1} [\boldsymbol{\mathcal{D}}_e]_{p,k_m}, & \text{if } k_m = \ell_n \text{ and } k_m \in \mathcal{I}_{e,p}, \\ -[\boldsymbol{\mathcal{D}}_e]_{p,k_m}^\top \left(\boldsymbol{\mathcal{D}}_{e,p}\boldsymbol{\mathcal{D}}_{e,p}^\top\right)^{-1} [\boldsymbol{\mathcal{D}}_e]_{p,\ell_n}, & \text{if } k_m \neq \ell_n \text{ and } k_m, \ell_n \in \mathcal{I}_{e,p}, \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{D.1}$$

Furthermore, we show that the k_m-th block of the $N_e \times 1$ block column vector $\boldsymbol{f}_e$ in (5.54) is equal to:

$$[\boldsymbol{f}_e]_{k_m} = \begin{cases} [\boldsymbol{\mathcal{D}}_e]_{p,k_m}^\top \left(\boldsymbol{\mathcal{D}}_{e,p}\boldsymbol{\mathcal{D}}_{e,p}^\top\right)^{-1} \boldsymbol{b}_p, & \text{if } k_m \in \mathcal{I}_{e,p}, \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{D.2}$$

It can be verified that $\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top$ is a $P \times P$ block diagonal matrix with (p,p)-th block of dimension $L_p \times L_p$ and given by:

$$[\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top]_{p,p} = \boldsymbol{\mathcal{D}}_{e,p}\boldsymbol{\mathcal{D}}_{e,p}^\top = \boldsymbol{\mathcal{D}}_p\boldsymbol{\mathcal{D}}_p^\top. \tag{D.3}$$

The inverse of the block diagonal matrix $\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top$ is:

$$\left(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top\right)^{-1} = \operatorname{diag}\left\{\left(\boldsymbol{\mathcal{D}}_{e,1}\boldsymbol{\mathcal{D}}_{e,1}^\top\right)^{-1}, \ldots, \left(\boldsymbol{\mathcal{D}}_{e,P}\boldsymbol{\mathcal{D}}_{e,P}^\top\right)^{-1}\right\}. \tag{D.4}$$

By multiplying the matrix $(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top)^{-1}$ from the left by $\boldsymbol{\mathcal{D}}_e^\top$ we obtain an $N_e \times P$ block matrix with (k_m, p)-th block of dimension $M_k \times L_p$ given by:

$$\left[\boldsymbol{\mathcal{D}}_e^\top \left(\boldsymbol{\mathcal{D}}_e\boldsymbol{\mathcal{D}}_e^\top\right)^{-1}\right]_{k_m,p} = \begin{cases} [\boldsymbol{\mathcal{D}}_e]_{p,k_m}^\top \left(\boldsymbol{\mathcal{D}}_{e,p}\boldsymbol{\mathcal{D}}_{e,p}^\top\right)^{-1}, & \text{if } k_m \in \mathcal{I}_{e,p} \\ \boldsymbol{0}, & \text{otherwise.} \end{cases} \tag{D.5}$$

When we multiply the matrix $\boldsymbol{\mathcal{D}}_e^\top \left(\boldsymbol{\mathcal{D}}_e \boldsymbol{\mathcal{D}}_e^\top\right)^{-1}$ from the right by $\boldsymbol{\mathcal{D}}_e$, we obtain an $N_e \times N_e$ block matrix with (k_m, ℓ_n)-th block corresponding to sub-nodes k_m, ℓ_n of dimension $M_k \times M_\ell$ and given by:

$$\left[\boldsymbol{\mathcal{D}}_e^\top \left(\boldsymbol{\mathcal{D}}_e \boldsymbol{\mathcal{D}}_e^\top\right)^{-1} \boldsymbol{\mathcal{D}}_e\right]_{k_m,\ell_n} = \begin{cases} [\boldsymbol{\mathcal{D}}_e]_{p,k_m}^\top \left(\boldsymbol{\mathcal{D}}_{e,p} \boldsymbol{\mathcal{D}}_{e,p}^\top\right)^{-1} [\boldsymbol{\mathcal{D}}_e]_{p,\ell_n}, & \text{if } k_m, \ell_n \in \mathcal{I}_{e,p}, \\ \mathbf{0}, & \text{otherwise.} \end{cases} \tag{D.6}$$

From (5.54) and (D.5), we obtain (D.2).

D.2 Evaluation of matrix $\boldsymbol{\mathcal{F}}$ for zero-mean real Gaussian regressors

Without loss of generality, we assume in the following that M_k is uniform across the network, i.e., $M_k = M_0$ for all k. We note that for any symmetric matrix $\boldsymbol{U}$, we have [Isserlis, 1918]:

$$\mathbb{E}\left\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)\boldsymbol{U}\boldsymbol{x}_\ell(i)\boldsymbol{x}_\ell^\top(i)\right\} = \boldsymbol{R}_{x,k}\boldsymbol{U}\boldsymbol{R}_{x,\ell} + \delta_{k,\ell}\big(\boldsymbol{R}_{x,k}\boldsymbol{U}\boldsymbol{R}_{x,k} + \boldsymbol{R}_{x,k}\mathrm{Tr}(\boldsymbol{R}_{x,k}\boldsymbol{U})\big). \tag{D.7}$$

From (5.62) and (5.77), we obtain:

$$\begin{aligned} \boldsymbol{\Sigma}' = \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e - \mu\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{R}}_{x,e} - \mu\boldsymbol{\mathcal{R}}_{x,e}\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e + \\ \mu^2\mathbb{E}\left\{\boldsymbol{\mathcal{R}}_{x,e}(i)\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{R}}_{x,e}(i)\right\}. \end{aligned} \tag{D.8}$$

In order to evaluate $\boldsymbol{\Sigma}'$ we need to evaluate the fourth term on the RHS of the above equation. Let:

$$\boldsymbol{\mathcal{V}} \triangleq \mathbb{E}\left\{\boldsymbol{\mathcal{R}}_{x,e}(i)\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{R}}_{x,e}(i)\right\}, \tag{D.9}$$

$$\boldsymbol{\mathcal{U}} \triangleq \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e. \tag{D.10}$$

It can be verified that the (k_m, ℓ_n)-th block of the matrix $\boldsymbol{\mathcal{V}}$ corresponding to the (k_m, ℓ_n)-th sub-node is given by:

$$\begin{aligned} [\boldsymbol{\mathcal{V}}]_{k_m,\ell_n} &= c_{k_m}c_{\ell_n}\mathbb{E}\left\{\boldsymbol{x}_k(i)\boldsymbol{x}_k^\top(i)[\boldsymbol{\mathcal{U}}]_{k_m,\ell_n}\boldsymbol{x}_\ell(i)\boldsymbol{x}_\ell^\top(i)\right\} \\ &= c_{k_m}c_{\ell_n}\boldsymbol{R}_{x,k}[\boldsymbol{\mathcal{U}}]_{k_m,\ell_n}\boldsymbol{R}_{x,\ell} + \delta_{k,\ell}c_{k_m}c_{\ell_n}\left(\boldsymbol{R}_{x,k}[\boldsymbol{\mathcal{U}}]_{k_m,\ell_n}\boldsymbol{R}_{x,k} + \boldsymbol{R}_{x,k}\mathrm{Tr}(\boldsymbol{R}_{x,k}[\boldsymbol{\mathcal{U}}]_{k_m,\ell_n})\right), \end{aligned} \tag{D.11}$$

where the $M_0 \times M_0$ matrix $[\boldsymbol{\mathcal{U}}]_{k_m,\ell_n}$ is the (k_m, ℓ_n)-th block of the matrix $\boldsymbol{\mathcal{U}}$. The matrix $\boldsymbol{\mathcal{V}}$ in (D.9) can be written as:

$$\boldsymbol{\mathcal{V}} = \boldsymbol{\mathcal{R}}_{x,e}\boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{R}}_{x,e} + \sum_{k=1}^{N}\left(\boldsymbol{\mathcal{S}}_k(\boldsymbol{I}_{N_e}\otimes\boldsymbol{R}_{x,k})\boldsymbol{\mathcal{U}}(\boldsymbol{I}_{N_e}\otimes\boldsymbol{R}_{x,k})\boldsymbol{\mathcal{S}}_k + \boldsymbol{\mathcal{S}}_k(\boldsymbol{I}_{N_e}\otimes\boldsymbol{R}_{x,k})\boldsymbol{\mathcal{Z}}_k\boldsymbol{\mathcal{S}}_k\right), \tag{D.12}$$

where $\boldsymbol{\mathcal{S}}_k$ is the $N \times N$ block diagonal matrix with (k,k)-th block equal to $\boldsymbol{C}_k \otimes \boldsymbol{I}_{M_0}$, and $\boldsymbol{\mathcal{Z}}_k$ is the $N_e \times N_e$ block matrix with (k_m, ℓ_n)-th block given by:

$$[\boldsymbol{\mathcal{Z}}_k]_{h_m,\ell_n} = \boldsymbol{I}_{M_0}[\mathrm{vec}(\boldsymbol{R}_{x,k})]^\top \mathrm{vec}\left([\boldsymbol{\mathcal{U}}]_{k_m,\ell_n}\right). \tag{D.13}$$

Applying the block-vectorization operator to $\boldsymbol{\mathcal{V}}$ and using property (A.17), we obtain:

$$\begin{aligned}\text{bvec}(\boldsymbol{\mathcal{V}}) = (\boldsymbol{\mathcal{R}}_{x,e} \otimes_b \boldsymbol{\mathcal{R}}_{x,e})\text{bvec}(\boldsymbol{\mathcal{U}}) + \sum_{k=1}^{N} \Big(\boldsymbol{\mathcal{S}}_k^{\top}(\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}) \otimes_b \boldsymbol{\mathcal{S}}_k(\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k})\Big)\,\text{bvec}(\boldsymbol{\mathcal{U}}) + \\ \sum_{k=1}^{N} \Big(\boldsymbol{\mathcal{S}}_k^{\top} \otimes_b (\boldsymbol{\mathcal{S}}_k[\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}])\Big)\,\text{bvec}(\boldsymbol{\mathcal{Z}}_k),\end{aligned} \tag{D.14}$$

where $\text{bvec}(\boldsymbol{\mathcal{Z}}_k)$ can be expressed as:

$$\text{bvec}(\boldsymbol{\mathcal{Z}}_k) = \Big(\boldsymbol{I}_{N_e^2} \otimes \text{vec}(\boldsymbol{I}_{M_0}) \otimes [\text{vec}(\boldsymbol{R}_{x,k})]^{\top}\Big)\,\text{bvec}(\boldsymbol{\mathcal{U}}), \tag{D.15}$$

where $\text{bvec}(\boldsymbol{\mathcal{U}}) = (\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}} \otimes_b \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}})\boldsymbol{\sigma}$. Finally, we conclude that the matrix $\boldsymbol{\mathcal{F}}$ in (5.79) can be written as:

$$\begin{aligned}\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{B}}^{\top} \otimes_b \boldsymbol{\mathcal{B}}^{\top} + \mu^2 \sum_{k=1}^{N} \Big(\boldsymbol{\mathcal{S}}_k^{\top}(\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}) \otimes_b \boldsymbol{\mathcal{S}}_k(\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k})\Big)\,(\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}} \otimes_b \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}) + \\ \mu^2 \sum_{k=1}^{N} \Big(\boldsymbol{\mathcal{S}}_k^{\top} \otimes_b (\boldsymbol{\mathcal{S}}_k(\boldsymbol{I}_{N_e} \otimes \boldsymbol{R}_{x,k}))\Big)\,\Big(\boldsymbol{I}_{N_e^2} \otimes \text{vec}(\boldsymbol{I}_{M_0}) \otimes [\text{vec}(\boldsymbol{R}_{x,k})]^{\top}\Big)\,(\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}} \otimes_b \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}})\,.\end{aligned} \tag{D.16}$$

D.3 Performance of competing algorithms

For compactness purposes, we explain in the following how the theoretical curves of the centralized CLMS algorithm (5.17) and the distributed algorithm (5.93) can be obtained from the analysis in Section 5.4 without showing the final theoretical expressions.

Consider the centralized CLMS algorithm (5.17). Let $\widetilde{\boldsymbol{w}}_b(i)$ and $\widetilde{\boldsymbol{w}}'_b(i)$ denote the $N \times 1$ block error vectors at the fusion center given by:

$$\widetilde{\boldsymbol{w}}_b(i) \triangleq \boldsymbol{w}_b^o - \boldsymbol{w}_b(i), \tag{D.17}$$

$$\widetilde{\boldsymbol{w}}'_b(i) \triangleq \boldsymbol{w}_b^{\star} - \boldsymbol{w}_b(i). \tag{D.18}$$

Note that the evolution of $\widetilde{\boldsymbol{w}}'_b(i)$ can be deduced from the evolution of $\widetilde{\boldsymbol{w}}_b(i)$ using the following relation:

$$\widetilde{\boldsymbol{w}}'_b(i) = \widetilde{\boldsymbol{w}}_b(i) - \boldsymbol{w}_b^{\delta}, \tag{D.19}$$

where $\boldsymbol{w}_b^{\delta} = \boldsymbol{w}_b^o - \boldsymbol{w}_b^{\star}$. Subtracting $\boldsymbol{w}_b^o$ from both sides of recursion (5.17) and using the linear data model (5.2), we arrive at the following recursion for the error vector $\widetilde{\boldsymbol{w}}_b(i)$:

$$\widetilde{\boldsymbol{w}}_b(i+1) = \boldsymbol{\mathcal{P}}\,(\boldsymbol{I}_{M_b} - \mu\boldsymbol{\mathcal{R}}_x(i))\,\widetilde{\boldsymbol{w}}_b(i) - \mu\boldsymbol{\mathcal{P}}\boldsymbol{p}_{xz}(i) + (\boldsymbol{I}_{M_b} - \boldsymbol{\mathcal{P}})\,\boldsymbol{w}_b^o + \boldsymbol{f}, \tag{D.20}$$

where $\boldsymbol{\mathcal{R}}_x(i)$ and $\boldsymbol{p}_{xz}(i)$ are given by:

$$\boldsymbol{\mathcal{R}}_x(i) \triangleq \text{diag}\left\{\boldsymbol{x}_1(i)\boldsymbol{x}_1^{\top}(i), \ldots, \boldsymbol{x}_N(i)\boldsymbol{x}_N^{\top}(i)\right\}, \tag{D.21}$$

$$\boldsymbol{p}_{xz}(i) \triangleq \text{col}\left\{d_1(i)\boldsymbol{x}_1(i), \ldots, d_N(i)\boldsymbol{x}_N(i)\right\}, \tag{D.22}$$

Using (D.19) with (D.20) and the fact that $\boldsymbol{w}_b^\star$ satisfies $\boldsymbol{\mathcal{P}}\boldsymbol{w}_b^\star - \boldsymbol{f} = \boldsymbol{w}_b^\star$, we obtain that $\widetilde{\boldsymbol{w}}_b'(i+1)$ evolves according to the following recursion:

$$\widetilde{\boldsymbol{w}}_b'(i+1) = \boldsymbol{\mathcal{P}}\left(\boldsymbol{I}_{M_b} - \mu\boldsymbol{\mathcal{R}}_x(i)\right)\widetilde{\boldsymbol{w}}_b'(i) - \mu\boldsymbol{\mathcal{P}}\boldsymbol{p}_{xz}(i) - \mu\boldsymbol{\mathcal{P}}\boldsymbol{\mathcal{R}}_x(i)\boldsymbol{w}_b^\delta. \tag{D.23}$$

Comparing recursions (D.20) and (D.23) with recursions (5.56) and (5.59), we observe that the learning curves of the centralized solution (5.17) can be deduced from the theoretical curves of the decentralized solution (5.36) by properly modifying the coefficient matrices and vectors. Note that, unlike the decentralized algorithm (5.17), the centralized solution is unbiased with respect to $\boldsymbol{w}_b^\star$ since $\mu\boldsymbol{\mathcal{P}}\mathbb{E}\,\boldsymbol{\mathcal{R}}_x(i)\boldsymbol{w}_b^\delta = \boldsymbol{0}$.

Next, consider the distributed solution (5.93). Following the same line of reasoning as in Subsection 5.4.1, we obtain the following recursions for the network block error vectors $\widetilde{\boldsymbol{w}}_e(i)$ in (5.43) and $\widetilde{\boldsymbol{w}}_e'(i)$ in (5.44):

$$\widetilde{\boldsymbol{w}}_e(i+1) = \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}^\top\left[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\right]\widetilde{\boldsymbol{w}}_e(i) - \mu\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}^\top\boldsymbol{p}_{xz,e}(i) + \left(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\right)\boldsymbol{w}_e^o + \boldsymbol{f}_e, \tag{D.24}$$

$$\widetilde{\boldsymbol{w}}_e'(i+1) = \boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}^\top\left[\boldsymbol{I}_{M_e} - \mu\boldsymbol{\mathcal{R}}_{x,e}(i)\right]\widetilde{\boldsymbol{w}}_e'(i) - \mu\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}^\top\boldsymbol{p}_{xz,e}(i) - \mu\boldsymbol{\mathcal{P}}_e\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{R}}_{x,e}(i)\boldsymbol{w}_e^\delta. \tag{D.25}$$

Comparing recursions (D.24) and (D.25) with recursions (5.56) and (5.59), we observe that the learning curves of the distributed solution (5.93) can be deduced from the theoretical curves of the decentralized solution (5.36) by properly replacing the product $\boldsymbol{\mathcal{A}}^\top\boldsymbol{\mathcal{P}}_e$ in the analysis of Section 5.4 by the product $\boldsymbol{\mathcal{P}}_e\,\boldsymbol{\mathcal{A}}^\top$ and the definition of vector $\boldsymbol{r}$ in (5.64) by $\boldsymbol{r} \triangleq \left(\boldsymbol{I}_{M_e} - \boldsymbol{\mathcal{P}}_e\right)\boldsymbol{w}_e^o + \boldsymbol{f}_e$.

Bibliography

A. Abboud, F. Iutzeler, R. Couillet, M. Debbah, and H. Siguerdidjane. Distributed production-sharing optimization and application to power grid networks. *IEEE Transactions on Signal and Information Processing over Networks*, 2(1):16–28, March 2016. (Cited on page 6.)

R. Abdolee and B. Champagne. Diffusion LMS algorithms for sensor networks over non-ideal inter-sensor wireless channels. In *Proc. IEEE International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, pages 1–6, Barcelona, Spain, June 2011. (Cited on pages 19 and 26.)

R. Abdolee and B. Champagne. Diffusion LMS strategies in sensor networks with noisy input data. *IEEE/ACM Transactions on Networking*, 24(1):3–14, February 2016. (Cited on page 139.)

R. Abdolee, B. Champagne, and A. H. Sayed. Estimation of space-time varying parameters using a diffusion LMS algorithm. *IEEE Transactions on Signal Processing*, 62(2):403–418, January 2014. (Cited on pages 6, 8, 11, 35, 48, 76, and 108.)

R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Inc. Upper Saddle River, NJ, USA, 1993. (Cited on pages 108, 109, and 130.)

T. Y. Al-Naffouri and A. H. Sayed. Transient analysis of data-normalized adaptive filters. *IEEE Transactions on Signal Processing*, 51(3):639–652, March 2003. (Cited on page 26.)

R. Ammanouil, A. Ferrari, C. Richard, and S. Mathieu. Nonlinear unmixing of hyperspectral data with vector-valued kernel functions. *Submitted for publication.* Also available at http://www.cedric-richard.fr/Articles/ammanouil2016nonlinear.pdf, 2016. (Cited on page 140.)

A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *Advances in Neural Information Processing Systems 19 (NIPS)*, pages 41–48. 2007. (Cited on page 6.)

T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione. Broadcast gossip algorithms for consensus. *IEEE Transactions on Signal Processing*, 57(7):2748–2761, 2009. (Cited on page 48.)

B. Bakker and T. Heskes. Task clustering and gating for Bayesian multi-task learning. *Journal of Machine Learning Research*, 4:83–99, December 2003. (Cited on pages 6 and 139.)

R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 25:21–30, March 2007. (Cited on page 78.)

T. Basar and G. J. Olsder. *Dynamic Noncooperative Game Theory*. London, Academic Press, 2nd edition edition, 1995. (Cited on pages 51 and 52.)

J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12(1): 149–198, March 2000. (Cited on page 6.)

A. Benveniste, M. Métivier, and P. Priouret. *Adaptive Algorithms and Stochastic Approximations*, volume 22. Springer-Verlag, NY, 1987. (Cited on page 35.)

D. S. Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas with Application to Linear Systems Theory*. Princeton University Press, 2005. (Cited on pages 143 and 158.)

A. Bertrand. *Signal processing algorithms for wireless acoustic sensor networks*. PhD thesis, Katholieke Universiteit Leuven, Belgium, May 2011. (Cited on pages 8 and 139.)

A. Bertrand and M. Moonen. Distributed adaptive node-specific signal estimation in fully connected sensor networks - Part I: Sequential node updating. *IEEE Transactions on Signal Processing*, 58(10):5277–5291, October 2010. (Cited on pages 8 and 48.)

A. Bertrand and M. Moonen. Distributed adaptive estimation of node-specific signals in wireless sensor networks with a tree topology. *IEEE Transactions on Signal Processing*, 59(5):2196–2210, May 2011. (Cited on pages 8 and 48.)

A. Bertrand and M. Moonen. Distributed node-specific LCMV beamforming in wireless sensor networks. *IEEE Transactions on Signal Processing*, 60(1):233–246, January 2012. (Cited on page 108.)

D. P. Bertsekas. A new class of incremental gradient methods for least squares problems. *SIAM Journal on Optimization*, 7(4):913–926, November 1997. (Cited on pages 3 and 108.)

D. P. Bertsekas. *Nonlinear Programming*. Athena scientific, 1999. (Cited on page 113.)

D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989. (Cited on pages 108, 111, and 131.)

P. Bianchi, G. Fort, W. Hachem, and J. Jakubowicz. Convergence of a distributed parameter estimator for sensor networks with local averaging of the estimates. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3764–3767, May 2011. (Cited on page 4.)

D. Blatt, A. O. Hero, and H. Gauchman. A convergent incremental gradient method with a constant step size. *SIAM Journal on Optimization*, 18(1):29–51, February 2007. (Cited on page 3.)

N. Bogdanović, J. Plata-Chaves, and K. Berberidis. Distributed diffusion-based LMS for node-specific parameter estimation over adaptive networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7223–7227, Florence, Italy, May 2014. (Cited on pages 7, 9, and 11.)

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. (Cited on pages 109 and 112.)

S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006. (Cited on page 48.)

L. Breiman and J. H. Friedman. Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 59(1):3–54, 1997. (Cited on page 5.)

P. J. Brown and J. V. Zidek. Adaptive multivariate ridge regression. *The Annals of Statistics*, 8 (1):64–74, January 1980. (Cited on page 5.)

E. J. Candes, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14:877–905, 2008. (Cited on page 80.)

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, July 1997. (Cited on pages 5 and 6.)

F. S. Cattivelli and A. H. Sayed. Diffusion LMS strategies for distributed estimation. *IEEE Transactions on Signal Processing*, 58(3):1035–1048, March 2010. (Cited on pages 3, 4, 18, 23, 24, 35, 52, 82, 147, and 151.)

F. S. Cattivelli and A. H. Sayed. Modeling bird flight formations using diffusion adaptation. *IEEE Transactions on Signal Processing*, 59(5):2038–2051, May 2011. (Cited on pages 5 and 68.)

J. Chen and A. H. Sayed. Diffusion adaptation strategies for distributed optimization and learning over networks. *IEEE Transactions on Signal Processing*, 60(8):4289–4305, August 2012. (Cited on pages 3, 4, 22, and 108.)

J. Chen and A. H. Sayed. Distributed Pareto optimization via diffusion strategies. *IEEE Journal of Selected Topics in Signal Processing*, 7(2):205–220, April 2013. (Cited on pages 4, 7, 18, 25, and 70.)

J. Chen, L. Tang, J. Liu, and J. Ye. A convex formulation for learning shared structures from multiple tasks. In *Proc. of the 26th Annual International Conference on Machine Learning (ICML)*, pages 137–144, 2009a. (Cited on page 6.)

J. Chen, X. Zhao, and A. H. Sayed. Bacterial motility via diffusion adaptation. In *Proc. Asilomar Conference on Signals, Systems and Computers*, pages 1930–1934, Pacific Grove, CA, November 2010. (Cited on page 5.)

J. Chen, C. Richard, A. O. Hero, and A. H. Sayed. Diffusion LMS for multitask problems with overlapping hypothesis subspaces. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Reims, France, September 2014a. (Cited on pages 9, 11, 48, 76, and 108.)

J. Chen, C. Richard, and A. H. Sayed. Diffusion LMS for clustered multitask networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5487–5491, Florence, Italy, May 2014b. (Cited on pages 8, 11, 12, 13, 47, 48, and 82.)

J. Chen, C. Richard, and A. H. Sayed. Multitask diffusion adaptation over networks. *IEEE Transactions on Signal Processing*, 62(16):4129–4144, August 2014c. (Cited on pages 8, 11, 12, 13, 47, 48, 49, 50, 51, 52, 60, 63, 65, 68, 76, 81, 85, 100, 101, and 108.)

J. Chen, C. Richard, and A. H. Sayed. Diffusion LMS over multitask networks. *IEEE Transactions on Signal Processing*, 63(11):2733–2748, June 2015a. (Cited on pages xiii, 7, 18, 25, 35, 40, 41, 42, 43, 44, and 45.)

J. Chen, Z. J. Towfic, and A. H. Sayed. Dictionary learning over distributed models. *IEEE Transactions on Signal Processing*, 63(4):1001–1016, February 2015b. (Cited on pages 7, 108, and 113.)

J. Chen, S. K. Ting, C. Richard, and A. H. Sayed. Group diffusion LMS. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4925–4929, Shanghai, China, March 2016. (Cited on pages 9 and 102.)

Y. Chen, Y. Gu, and A. O. Hero. Sparse LMS for system identification. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3125–3128, Taipei, Taiwan, April 2009b. (Cited on pages 78 and 80.)

S. Chouvardas and M. Draief. A diffusion kernel LMS algorithm for nonlinear adaptive networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4164–4168, Shanghai, China, March 2016. (Cited on page 140.)

S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis. Sparsity-promoting adaptive algorithm for distributed learning in diffusion networks. In *Proc. 20th European Signal Process-*

ing Conference (EUSIPCO), pages 1084–1088, Bucharest, Romania, August 2012. (Cited on pages 4, 79, and 80.)

S. Chouvardas, G. Mileounis, N. Kalouptsidis, and S. Theodoridis. A greedy sparsity-promoting LMS for distributed adaptive learning in diffusion networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5415–5419, Vancouver, Canada, May 2013. (Cited on pages 4 and 79.)

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. H. Bauschke *et al.*, editor, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, volume 49 of *Springer Optimization and Its Applications*, pages 185–212. Springer New York, 2011. (Cited on pages 77 and 86.)

P. L. Combettes, D. Dũng, and B. C. Vũ. Proximity for sums of composite functions. *Journal of Mathematical Analysis and Applications*, 380(2):680–688, 2011. (Cited on pages 77 and 86.)

A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, November 2010. (Cited on pages 3 and 4.)

T. Evgeniou and M. Pontil. Regularized multi–task learning. In *Proc. of the tenth ACM SIGKDD International Conference on Knowledge discovery and data mining*, pages 109–117, 2004. (Cited on page 6.)

T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, December 2005. (Cited on page 6.)

O. L. Frost III. An algorithm for linearly constrained adaptive array processing. *Proceedings of the IEEE*, 60(8):926–935, August 1972. (Cited on pages 112 and 113.)

W. Gao, J. Chen, C. Richard, J. Huang, and R. Flamary. Kernel LMS algorithm with forward-backward splitting for dictionary learning. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 5735–5739, Vancouver, Canada, May 2013. (Cited on page 80.)

W. Gao, J. Chen, C. Richard, and J. Huang. Diffusion adaptation over networks with kernel least-mean-square. In *IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 217–220, Cancun, Mexico, December 2015. (Cited on page 140.)

V. C. Gogineni and M. Chakraborty. Diffusion adaptation over clustered multitask networks based on the affine projection algorithm. *Submitted for publication.* Also available as arXiv:1507.08566, July 2015a. (Cited on page 8.)

V. C. Gogineni and M. Chakraborty. Distributed multi-task APA over adaptive networks based on partial diffusion. *Submitted for publication.* Also available as arXiv:1509.09157, July 2015b. (Cited on page 8.)

L. Grady and J. R. Polimeni. *Discrete Calculus: Applied Analysis on Graphs for Computational Science*. Springer, 2010. (Cited on page 51.)

W. H. Greene. *Econometric Analysis*, volume 5. Prentice Hall, 2003. (Cited on pages 5 and 6.)

Y. Gu, J. Jin, and S. Mei. ℓ_0-norm constraint LMS algorithm for sparse system identification. *IEEE Signal Processing Letters*, 16(9):774–777, September 2009. (Cited on page 78.)

S. Haykin. *Adaptive Filter Theory*. Prentice Hall, 2002. (Cited on page 35.)

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469, 2004. (Cited on page 96.)

L. Isserlis. On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika*, 12(1/2):134–139, November 1918. (Cited on page 162.)

M. O. Jackson. *Social and Economic networks*. Princeton University Press, Princeton, NJ, 2008. (Cited on pages 1 and 141.)

L. Jacob, J.-P. Vert, and F. Bach. Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems 21 (NIPS)*, pages 745–752. 2009. (Cited on page 6.)

D. Jakovetic, J. Xavier, and J. M. F. Moura. Weight optimization for consensus algorithms with correlated switching topology. *IEEE Transactions on Signal Processing*, 58(7):3788–3801, 2010. (Cited on page 48.)

D. Jakovetic, J. Xavier, and J. M. F. Moura. Cooperative convex optimization in networked systems: Augmented Lagrangian algorithms with directed gossip communication. *IEEE Transactions on Signal Processing*, 59(8):3889–3902, 2011. (Cited on page 48.)

S. Kanna and D. P. Mandic. Steady-state behavior of general complex-valued diffusion LMS strategies. *IEEE Signal Processing Letters*, 23(5):722–726, May 2016. (Cited on page 4.)

S. Kar and J. M. F. Moura. Sensor networks with random links: Topology design for distributed consensus. *IEEE Transactions on Signal Processing*, 56(7):3315–3326, 2008. (Cited on pages 48 and 55.)

S. Kar and J. M. F. Moura. Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise. *IEEE Transactions on Signal Processing*, 57 (1):355–369, 2009. (Cited on pages 48 and 55.)

S. Kar and J. M. F. Moura. Distributed consensus algorithms in sensor networks: Quantized data and random link failures. *IEEE Transactions on Signal Processing*, 58(3):1383–1400, 2010. (Cited on page 48.)

S. Kar and J. M. F. Moura. Convergence rate analysis of distributed gossip (linear parameter) estimation: Fundamental limits and tradeoffs. *IEEE Journal of Selected Topics in Signal Processing*, 5(4):674–690, August 2011. (Cited on pages 3, 4, and 48.)

V. Kekatos and G. B. Giannakis. Distributed robust power system state estimation. *IEEE Transactions on Power Systems*, 28(2):1617–1626, May 2013. (Cited on pages 6, 108, and 139.)

A. Khalili, M. A. Tinati, A. Rastegarnia, and J. A. Chambers. Steady-state analysis of diffusion LMS adaptive networks with noisy links. *IEEE Transactions on Signal Processing*, 60(2): 974–979, February 2012. (Cited on pages 19 and 26.)

S. Khawatmi, A. M. Zoubir, and A. H. Sayed. Decentralized clustering over adaptive networks. In *Proc. 23rd European Signal Processing Conference (EUSIPCO)*, pages 2696–2700, Nice, France, August 2015. (Cited on pages 7, 18, and 41.)

R. H. Koning, H. Neudecker, and T. Wansbeek. Block Kronecker products and the vecb operator. *Linear Algebra and its Applications*, 149:165–184, April 1991. (Cited on pages 123 and 144.)

Y. Kopsinis, K. Slavakis, and S. Theodoridis. Online sparse system identification and signal reconstruction using projections onto weighted ℓ_1-balls. *IEEE Transactions on Signal Processing*, 59(3):936–952, March 2011. (Cited on pages 79 and 80.)

S. Lee and A. Nedić. Distributed random projection algorithm for convex optimization. *IEEE Journal of Selected Topics in Signal Processing*, 7(2):221–229, 2013. (Cited on pages 108 and 113.)

T. G. Lewis. *Network Science: Theory and Applications*. Wiley, NJ, 2011. (Cited on page 1.)

Y. Liu, C. Li, and Z. Zhang. Diffusion sparse least-mean squares over networks. *IEEE Transactions on Signal Processing*, 60(8):4480–4485, August 2012. (Cited on pages 4 and 79.)

C. G. Lopes and A. H. Sayed. Incremental adaptive strategies over distributed networks. *IEEE Transactions on Signal Processing*, 55(8):4064–4077, August 2007. (Cited on page 3.)

C. G. Lopes and A. H. Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, July 2008. (Cited on pages 3, 4, 18, 23, and 108.)

P. Di Lorenzo. Diffusion adaptation strategies for distributed estimation over Gaussian Markov random fields. *IEEE Transactions on Signal Processing*, 62(21):5748–5760, November 2014. (Cited on pages 35, 79, 80, and 140.)

P. Di Lorenzo and A. H. Sayed. Sparse distributed learning based on diffusion adaptation. *IEEE Transactions on Signal Processing*, 61(6):1419–1433, March 2013. (Cited on pages 4, 79, 80, and 95.)

P. Di Lorenzo, S. Barbarossa, and A. H. Sayed. Sparse diffusion LMS for distributed adaptive estimation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3281–3284, Kyoto, Japan, March 2012. (Cited on pages 4, 79, and 80.)

P. Di Lorenzo, S. Barbarossa, and A. H. Sayed. Bio-inspired decentralized radio access based on swarming mechanisms over adaptive networks. *IEEE Transactions on Signal Processing*, 61 (12):3183–3197, June 2013. (Cited on page 102.)

P. Di Lorenzo, P. Banelli, S. Barbarossa, and S. Sardellitti. Distributed adaptive learning of graph signals. Available as arXiv:1609.06100, September 2016. (Cited on page 140.)

G. Mileounis, B. Babadi, N. Kalouptsidis, and V. Tarokh. An adaptive greedy algorithm with application to nonlinear communications. *IEEE Transactions on Signal Processing*, 58(6): 2998–3007, June 2010. (Cited on page 79.)

S. Monajemi, S. Sanei, S. H. Ong, and A. H. Sayed. Adaptive regularized diffusion adaptation over multitask networks. In *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–5, Boston, USA, September 2015. (Cited on page 8.)

S. Monajemi, K. Eftaxias, S. Sanei, and S. H. Ong. An informed multitask diffusion adaptation approach to study tremor in Parkinson's disease. *IEEE Journal of Selected Topics in Signal Processing*, PP(99):1–1, 2016. (Cited on page 7.)

J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Puschel. Distributed basis pursuit. *IEEE Transactions on Signal Processing*, 60(4):1942–1956, April 2012. (Cited on page 113.)

J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel. D-ADMM: A communication-efficient distributed algorithm for separable optimization. *IEEE Transactions Signal Processing*, 61(10):2718–2723, May 2013. (Cited on pages 108 and 113.)

Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada. A sparse adaptive filtering using time-varying soft-thresholding techniques. In *Proc. International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 3734–3737, Dallas, USA, March 2010. (Cited on pages 79 and 80.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Performance analysis of multitask diffusion adaptation over asynchronous networks. In *Proc. Asilomar Conference on Signals, Systems and Computers*, pages 788–792, Pacific Grove, CA, November 2014. (Cited on page 8.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion LMS with sparsity-based regularization. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3516–3520, Brisbane, Australia, April 2015. (Cited on pages 8, 11, 48, 77, and 83.)

R. Nassif, C. Richard, J. Chen, A. Ferrari, and A. H. Sayed. Diffusion LMS over multitask networks with noisy links. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 4583–4587, Shanghai, China, March 2016a. (Cited on page 7.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Diffusion LMS for multitask problems with local linear equality constraints. *Submitted for publication.* Also available as arXiv:1610.02943, October 2016b. (Cited on pages 9 and 11.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion adaptation over asynchronous networks. *IEEE Transactions on Signal Processing*, 64(11):2835–2850, June 2016c. (Cited on pages 8, 11, and 108.)

R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Proximal multitask learning over networks with sparsity inducing coregularization. *IEEE Transactions on Signal Processing*, 64(23):6329–6344, December 2016d. (Cited on pages 9, 11, 48, and 108.)

A. Nedic and D. P. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. *SIAM Journal on Optimization*, 12(1):109–138, July 2001. (Cited on page 3.)

A. Nedic and A. Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, January 2009. (Cited on pages 3 and 4.)

M. Newman. *Networks: An introduction.* Oxford University Press, 2010. (Cited on page 1.)

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3): 123–231, 2013. (Cited on pages 77 and 86.)

B. L. Partridge. The structure and function of fish schools. *Scientific American*, 246(6):114–123, June 1982. (Cited on page 2.)

J. Plata-Chaves, N. Bogdanović, and K. Berberidis. Distributed diffusion-based LMS for node-specific adaptive parameter estimation. *IEEE Transactions on Signal Processing*, 63(13):3448–3460, July 2015. (Cited on pages 9, 35, 48, 76, 100, 101, 102, and 108.)

J. Plata-Chaves, M. H. Bahari, M. Moonen, and A. Bertrand. Unsupervised diffusion-based LMS for node-specific parameter estimation over wireless sensor networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4159–4163, Shanghai, China, March 2016a. (Cited on pages 9 and 102.)

J. Plata-Chaves, A. Bertrand, and M. Moonen. Incremental multiple error filtered-X LMS for node-specific active noise control over wireless acoustic sensor networks. In *IEEE Sensor Array and Multichannel Signal Processing Workshop*, pages 1–5, Rio de Janeiro, Brazil, July 2016b. (Cited on pages 6 and 108.)

B. T. Polyak. *Introduction to Optimization*. Optimization Software, 1987. (Cited on page 86.)

M. G. Rabbat and R. D. Nowak. Quantized incremental algorithms for distributed optimization. *IEEE Journal on Selected Areas in Communications*, 23(4):798–808, April 2005. (Cited on page 3.)

S. S. Ram, A. Nedić, and V. V. Veeravalli. Distributed stochastic subgradient projection algorithms for convex optimization. *Journal of Optimization Theory and Applications*, 147(3): 516–545, 2010. (Cited on pages 108 and 113.)

J. B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica: Journal of the Econometric Society*, 33(3):520–534, 1965. (Cited on page 52.)

A. H. Sayed. *Adaptive Filters*. John Wiley & Sons, NJ, 2008. (Cited on pages 21, 26, 31, 33, 35, 56, and 113.)

A. H. Sayed. Adaptation, learning, and optimization over networks. *Foundations and Trends in Machine Learning*, 7(4-5):311–801, July 2014a. (Cited on pages 1, 2, 3, 4, 7, 18, 20, 22, 60, 85, 123, and 144.)

A. H. Sayed. Adaptive networks. *Proceedings of the IEEE*, 102(4):460–497, April 2014b. (Cited on pages 4, 18, 85, and 115.)

A. H. Sayed. Diffusion adaptation over networks. In R. Chellapa and S. Theodoridis, editors, *E-Reference Signal Processing*, volume 3, pages 323–454. Elsevier, 2014c. (Cited on pages 4, 18, 20, 23, 24, 25, 26, 29, 35, 52, 61, 68, 69, 82, 83, 84, 85, 102, 117, 144, 145, 147, 148, 149, 151, and 153.)

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 Processing Magazine*, 30(3):155–171, May 2013. (Cited on pages 3, 4, 5, 18, 23, and 24.)

D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, May 2013. (Cited on page 140.)

K. Srivastava and A. Nedic. Distributed asynchronous constrained stochastic optimization. *IEEE Journal of Selected Topics in Signal Processing*, 5(4):772–790, August 2011. (Cited on pages 4, 48, and 108.)

J. Szurley, A. Bertrand, and M. Moonen. Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks. *Signal Processing*, 117:44–60, December 2015. (Cited on pages 8 and 48.)

S. Thrun and L. Pratt. *Learning to Learn*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. (Cited on page 5.)

R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, pages 267–288, 1996. (Cited on page 78.)

H. Tiomoko Ali and R. Couillet. Performance analysis of spectral community detection in realistic graph models. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4548–4552, Shanghai, China, March 2016. (Cited on page 141.)

Z. J. Towfic and A. H. Sayed. Adaptive penalty-based distributed stochastic convex optimization. *IEEE Transactions on Signal Processing*, 62(15):3924–3938, August 2014. (Cited on pages 4, 108, 113, and 140.)

Z. J. Towfic and A. H. Sayed. Stability and performance limits of adaptive primal-dual networks. *IEEE Transactions on Signal Processing*, 63(11):2888–2903, June 2015. (Cited on page 109.)

J. N. Tsitsiklis, D. P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9): 803–812, 1986. (Cited on page 48.)

S.-Y. Tu and A. H. Sayed. Adaptive networks with noisy links. In *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1–5, Houston, TX, December 2011a. (Cited on page 19.)

S. Y. Tu and A. H. Sayed. Mobile adaptive networks. *IEEE Journal of Selected Topics in Signal Processing*, 5(4):649–664, August 2011b. (Cited on pages 5 and 68.)

S. Y. Tu and A. H. Sayed. Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks. *IEEE Transactions on Signal Processing*, 60(12):6217–6234, December 2012. (Cited on page 4.)

J. A. Ventura. Computational development of a Lagrangian dual approach for quadratic networks. *Networks*, 21(4):469–485, 1991. (Cited on pages 109 and 130.)

S. Vlaski and A. H. Sayed. Proximal diffusion for stochastic costs with non-differentiable regularizers. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3352–3356, Brisbane, Australia, April 2015. (Cited on pages 4 and 79.)

W. M. Wee and I. Yamada. A proximal splitting approach to regularized distributed adaptive estimation in diffusion networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5420–5424, Vancouver, Canada, May 2013. (Cited on pages 4, 79, and 80.)

E. T. Whittaker and G. N. Watson. *A Course of Modern Analysis*. Cambridge University Press, 1996. (Cited on page 93.)

Y. Xia, D. P. Mandic, and A. H. Sayed. An adaptive diffusion augmented CLMS algorithm for distributed filtering of noncircular complex signals. *IEEE Signal Processing Letters*, 18(11): 659–662, November 2011. (Cited on page 4.)

Q. Xu and Q. Yang. A survey of transfer and multitask learning in bioinformatics. *Journal of Computing Science and Engineering*, 5(3):257–268, September 2011. (Cited on page 6.)

K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016. (Cited on page 113.)

A. Zellner. An efficient method of estimating seemingly unrelated regressions and tests for aggregation bias. *Journal of the American Statistical Association*, 57(298):348–368, June 1962. (Cited on page 5.)

X. Zhang, C. Moore, and M. E. J. Newman. Random graph models for dynamic networks. Also available as arXiv:1607.07570, 2016. (Cited on page 141.)

X. Zhao and A. H. Sayed. Asynchronous adaptation and learning over networks-Part I: Modeling and stability analysis. *IEEE Transactions on Signal Processing*, 63(4):811–826, February 2015a. (Cited on pages 49, 53, and 55.)

X. Zhao and A. H. Sayed. Asynchronous adaptation and learning over networks-Part II: Performance analysis. *IEEE Transactions on Signal Processing*, 63(4):827–842, February 2015b. (Cited on pages 49, 60, and 63.)

X. Zhao and A. H. Sayed. Asynchronous adaptation and learning over networks-Part III: Comparison analysis. *IEEE Transactions on Signal Processing*, 63(4):843–858, February 2015c. (Cited on page 49.)

X. Zhao and A. H. Sayed. Distributed clustering and learning over networks. *IEEE Transactions on Signal Processing*, 63(13):3285–3300, July 2015d. (Cited on pages 7 and 18.)

X. Zhao, S.-Y. Tu, and A. H. Sayed. Diffusion adaptation over networks under imperfect information exchange and non-stationary data. *IEEE Transactions on Signal Processing*, 60(7): 3460–3475, July 2012. (Cited on pages xiii, 19, 26, 32, 35, 40, 41, 42, 43, 44, and 45.)

Z. Zhao and A. H. Sayed. Clustering via diffusion adaptation over networks. In *Proc. International Workshop on Cognitive Information Processing (CIP)*, pages 1–6, Parador de Baiona, Spain, May 2012. (Cited on pages xiii, 7, 18, 40, 41, 42, 43, 44, and 45.)

H. Zou. The adaptive LASSO and its oracle properties. *Journal of the American Statistical Association*, 101(476):1418–1429, 2006. (Cited on page 80.)