Distributed Adaptive Trace Ratio Optimization in Wireless Sensor Networks

Cem Ates Musluoglu, and Alexander Bertrand, *Senior Member, IEEE*

***Abstract*—The trace ratio optimization (TRO) problem consists of finding an orthonormal basis for the discriminative subspace that maximizes the ratio of two trace operators on two covariance matrices corresponding to two distinctive classes or signal components. The TRO problem is encountered in various signal processing problems such as dimensionality reduction, signal enhancement, and discriminative analysis. In this paper, we propose a distributed and adaptive algorithm for solving the TRO problem in the context of wireless sensor networks (WSNs), where the two matrices involved in the trace ratio operators correspond to the (unknown) spatial correlation of the sensor signals across the nodes in the network. We first focus on fully-connected networks where every node can communicate with each other, but only compressed signals observations can be shared to reduce the communication cost. After showing convergence, we modify the algorithm to operate in WSNs with more general topologies. Simulation results are provided to validate and complement the theoretical results.**

***Index Terms*—Trace Ratio Optimization, Distributed Optimization, Linear Discriminant Analysis, SNR Optimization, Dimensionality Reduction, Wireless Sensor Networks**

I. Introduction

THE trace ratio optimization (TRO) problem is the maximization of the ratio between two quadratic forms, where the optimization variable X is required to have orthonormal columns. Mathematically, it can be expressed as:

$$\begin{aligned} &\underset{X}{\text{maximize}} \ \frac{\text{tr}(X^T A X)}{\text{tr}(X^T B X)} \\ &\text{subject to } X^T X = I, \end{aligned} \tag{1}$$

where A and B are positive definite matrices, I is the identity matrix and "tr" and "T" denote the trace and transpose operators respectively. Note that Problem (1) is different from a standard generalized Rayleigh quotient optimization, due to the presence of additional orthogonality constraints on X, and therefore the solution is not provided by a generalized eigenvalue decomposition (GEVD). The TRO problem is commonly encountered in signal processing and machine learning tasks [2]–[7] when the goal is to find an optimal subspace projection for data points belonging to two different classes. The subspace is generated by the columns of X while A and B are chosen in a way that reflects the difference between the classes. For example, in motor imagery brain-computer interfaces based on electroencephalography (EEG), one class could represent EEG activity during imaginary right-hand movement, while the other could represent EEG activity during right-foot movement. In that case, A would correspond to the covariance matrix of the signals resulting from one of these EEG activities while B would be the covariance matrix of the signals from the other one [8].

The TRO problem took its roots from Fisher's linear discriminant [9] and the Foley-Sammon transform (FST) [10], [11]. These are essentially "greedy" formulations of (1), in which X is replaced with a single-column vector. This single-column problem is then solved multiple times (for each column of X), while preserving orthogonality with respect to previously computed columns. However, the optimality does not hold for the space spanned by the whole set of vectors, because of the greedy computation method. This was pointed out in [12], while also providing a method to find a generalized optimal set. It was however mentioned in [2] that this latter technique suffers from separability issues on the projected set of vectors. Therefore, the generalized Foley-Sammon transform was defined in [2] as maximizing the ratio of two trace operators, which led to the TRO problem (1). Several methods to solve the general TRO problem have been described in the literature, using the Grassmann manifold [5], by semi-definite programming [7], and using an iterative method related to the Rayleigh quotient iteration [2]–[4].

In this paper, we study the TRO problem in a distributed context. As a target use case, we consider a wireless sensor network (WSN) in which the sensor signals are spatially correlated across the different nodes. A WSN consists of a multitude of distributed wireless sensor nodes that are equipped with sensing, computing and communication facilities. In general, a commonly encountered objective in WSNs is to use the total information available at all nodes to perform a certain task, such as estimating or detecting a network-wide parameter or signal. Estimation problems of this form are encountered, for example, in acoustic sensor networks [13]–[16], body-sensor and neuro-sensor networks [17]–[19], spectrum sensing for cognitive (radio) sensor networks [20], [21] and many other various applications [22]–[26]. On the other hand, significant results have been obtained on distributed and decentralized detection in the last decades, where detection of both known and unknown sources have been studied [27]. Initial applications were directed towards distributed radar and the problem has been of interest in diverse areas of research since then [28], such as the aerospace field [29].

In our case, we want to exploit the spatial correlation between the sensor signals present in the network. Therefore, the TRO objective is defined by the network-wide spatial correlation matrices of the observed sensor signals across the nodes of the WSN. We assume that these network-wide spatial correlation matrices are unknown and should be implicitly learned from the collected sensor data at run time. We propose a distributed adaptive algorithm to solve the TRO problem in such a context, in which the nodes transmit compressed sensor observations to the other nodes in the network. The distributed TRO (DTRO) algorithm is proven to converge to the solution of the centralized TRO problem, as if each node would have access to all sensor data in the network without the need to explicitly estimate the full network-wide correlation structure. A conference precursor describing the DTRO algorithm for fully-connected WSNs has been published in [1]. In this paper, we provide a more detailed analysis, a proof of convergence, an extension to arbitrary topologies, and more extensive experimental analyses.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 802895). The authors also acknowledge the financial support of the FWO (Research Foundation Flanders) for project G.0A49.18N, and the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

C.A. Musluoglu and A. Bertrand are with KU Leuven, Department of Electrical Engineering (ESAT), Stadius Center for Dynamical Systems, Signal Processing and Data Analytics, Kasteelpark Arenberg 10, box 2446, 3001 Leuven, Belgium and with Leuven.AI - KU Leuven institute for AI. e-mail: cemates.musluoglu, alexander.bertrand @esat.kuleuven.be

A conference precursor of this manuscript has been published in [1].

The outline of the paper is as follows. In Section II, we review the centralized TRO problem and an algorithm to solve it. Then, in Section III, we propose an algorithm to solve the TRO problem in a distributed fashion (DTRO), in the particular context of fully-connected networks (FC-DTRO). Section IV extends the analysis and method to connected networks with any topology, which we denote as topology-independent DTRO (TI-DTRO). The communication and computational aspects of the proposed algorithms are then discussed in Section V. Finally, we provide in Section VI simulation results of the DTRO algorithms in various settings to validate its performance, along with discussions.

Throughout this paper, the notation χ_q^i is used to refer to a certain object χ at node q and iteration i. If χ is a signal, we denote its compressed version by $\widehat{\chi}$. On the other hand, if χ is an object related to the TRO problem (e.g., a variable, signal, set or function), $\widetilde{\chi}$ denotes the analogous object in the DTRO setting (a more formal definition is provided further on in equation (21)). The matrix I_Q denotes the $Q \times Q$ identity matrix. Apart from a few exceptions, scalars, scalar-valued functions and vectors are written in lowercase letters, the latter in bold, while matrices and sets are written in uppercase letters, the latter in calligraphic font.

II. The Trace Ratio Optimization Problem

A. Problem Definition and Relationship to Other Problems

For convenience, and to introduce some new notation, we repeat the formal definition of the TRO problem. Given two positive definite matrices $A, B \in \mathbb{R}^{M\times M}$, the TRO problem aims at finding a matrix $X \in \mathbb{R}^{M\times Q}$ solving the following problem:

$$\begin{aligned} \underset{X}{\text{maximize}}\ & \varrho(X) \triangleq \frac{\text{tr}(X^T A X)}{\text{tr}(X^T B X)} \\ \text{subject to}\ & \quad X^T X = I_Q. \end{aligned} \tag{2}$$

The optimization variable X therefore contains Q orthonormal vectors in its columns spanning a subspace maximizing the objective ϱ, typically with $Q \ll M$. It is noted that the solution of (2), is unique up to a unitary transformation [30].

The matrix pencil (A, B) of Problem (2) can have various interpretations depending on the application. For example, in linear discriminant analysis, the aim is to tightly group points of a same class while separating each class from another in the best way possible [31], yielding A to correspond to the between-class scatter matrix of the data points, whereas B would be the within-class scatter matrix. In a multi-channel signal processing context, X can be interpreted as a collection of orthogonal filter banks applied to different states or components of a multi-channel signal, e.g., to discriminate between signal and noise or between two underlying states that alter the statistics of the signal. A and B would in that case represent the two covariance matrices corresponding to these two signal states or components (see also Section III-A).

In various applications, the optimal discriminant vectors are often taken to be the generalized eigenvectors (GEVCs) of the ordered matrix pencil (A, B) for easier computation of the optimal vectors, as in [12]. Mathematically, both problems are similarly formulated, the only difference being the constraint set. The optimization problem related to the GEVD problem can be written as:

$$\begin{aligned} \underset{X}{\text{maximize}}\ & \varrho(X) = \frac{\text{tr}(X^T A X)}{\text{tr}(X^T B X)} \\ \text{subject to}\ & \quad X^T B X = I_Q. \end{aligned} \tag{3}$$

It can be shown [32] that the maximal value of ϱ is the scaled sum of the largest generalized eigenvalues (GEVLs) of the matrix pencil (A, B). A solution of (3) is then given by the GEVCs corresponding to those GEVLs. These GEVCs are defined as the columns of the $M \times Q$ matrix $X^\dagger$ satisfying:

$$AX^\dagger = BX^\dagger \Lambda^\dagger, \tag{4}$$

where $\Lambda^\dagger \in \mathbb{R}^{Q\times Q}$ is a diagonal matrix with the GEVLs on its diagonal. It has been pointed out in previous studies ([2], [3], [30]) that although related, problems (2) and (3) differ in the general case, the only exception being the setting $Q = 1$. In that particular case, if ρ^* is the optimal value of (2) and $\rho^\dagger$ the optimal value of (3), then $\rho^* = \rho^\dagger$ and if $\mathbf{x}^\dagger \in \mathbb{R}^M$ is a solution of (3), then $\mathbf{x}^* = ||\mathbf{x}^\dagger||^{-1}\mathbf{x}^\dagger \in \mathbb{R}^M$ is a solution of (2).

Despite the similarity of the solutions for $Q = 1$, the links between the solutions of both problems are less trivial for higher projection dimensions. It has been discussed in [2]–[4], [30] that both problems are not interchangeable, resulting in optimal projection matrices with different properties. In [4], it is pointed out that the GEVD problem can be described as a greedy way to solve the TRO problem. Interesting comparison arguments between both methods have been given in [3], while [5] explains that the GEVD problem will not necessarily give a larger maximal value for ϱ. Even though the GEVD problem is easier to compute, it is noted in [33] that the solution of the TRO problem has the advantage that it does not distort the metric structure of the signals, by enforcing orthogonality on the filters.

We also note that the TRO problem should not be confused with the problem referred to as the "ratio trace" [3], which can be considered as a relaxation of (3). This problem is obtained by replacing the trace operators in the objective by the determinant operator, while removing the constraints. All possible GEVCs corresponding to the Q largest GEVLs of the matrix pencil (A, B) are a subset of the solution set of the ratio trace problem, the latter also including matrices obtained by a scaling followed by an orthogonal transformation of the columns of $X^\dagger$ [3], [34].

B. The Trace Ratio Optimization Algorithm

Although there exist several ways to solve (2), we will only review the one presented in [3], [30], as it serves as the basis for the distributed algorithm presented in Section III. The method shares similar steps with the Rayleigh quotient iteration method [35]–[38], used for computing eigenvalues.

The iterative algorithm to solve the TRO problem is constructed by transforming the optimization problem to a scalar one (i.e., in a single variable), by defining the auxiliary functions $h : \mathcal{S} \times \mathbb{R} \to \mathbb{R}$:

$$h(X, \rho) \triangleq \text{tr}\left(X^T(A - \rho B)X\right), \tag{5}$$

where $\mathcal{S} \triangleq \{X \in \mathbb{R}^{M\times Q} : X^T X = I_Q\}$ and $f : \mathbb{R} \to \mathbb{R}$ as:

$$f(\rho) \triangleq \max_{X\in\mathcal{S}} h(X, \rho). \tag{6}$$

As shown in [2], the following theorem relates f to the optimum ρ^* of (2).

Theorem 1. *[2] We have the following relationships between f in (6) and the optimum ρ^* of (2):*

- $f(\rho) = 0 \iff \rho = \rho^*$,
- $f(\rho) > 0 \iff \rho < \rho^*$,
- $f(\rho) < 0 \iff \rho > \rho^*$.

Furthermore, it is shown in [2] that an X^* satisfying

$$X^* \in \arg\max_{X\in\mathcal{S}} h(X, \rho^*), \tag{7}$$

solves the TRO problem (2), and the converse is also true, i.e., a solution of the TRO problem also maximizes $h(X, \rho^*)$ under the constraint $X \in \mathcal{S}$. The initial problem is therefore transformed into finding the root of the function f over the

Algorithm 1: Trace Ratio Maximization Algorithm [3]

input : $A, B \in \mathbb{R}^{M\times M}$
output: X^*, ρ^*

X^0 initialized randomly, $\rho^0 \leftarrow \varrho(X^0)$, $i \leftarrow 0$
repeat
1) $X^{i+1} \leftarrow \text{EVC}_Q(A - \rho^i B)$, where $\text{EVC}_Q(Z)$ extracts Q orthonormal eigenvectors corresponding to the Q largest eigenvalues of Z
2) $X^{i+1} \leftarrow X^{i+1}U^{i+1}$, where $U^{i+1} = \text{argmin}_{U\in\mathcal{D}}||X^{i+1}U - X^i||_F$, with $\mathcal{D}$ the set of signature matrices, i.e., diagonal matrices containing either 1 or -1 on their diagonal
3) $\rho^{i+1} \leftarrow \varrho(X^{i+1})$ where ϱ is defined in (2)
$i \leftarrow i+1$
until *Convergence*

variable ρ. It is noted that the evaluation of $f(\rho)$ defined in (6) is equivalent to taking the sum of the Q largest eigenvalues (EVLs) of $(A - \rho B)$ [39]. Since the matrix $A - \rho B$ is symmetric, the variable X maximizing the function $h(\cdot, \rho)$ within the orthogonal constraint set $\mathcal{S}$ therefore corresponds to the Q eigenvectors (EVCs) corresponding to those EVLs, i.e., the X's that satisfy:

$$(A - \rho B)X = X\Lambda, \tag{8}$$

where Λ is a diagonal matrix containing the Q largest EVLs.

The iterative step of the TRO algorithm in [3] is based on computing a new ρ using X from (8), substituting it in $\varrho(X)$ defined in (2) and repeating this process. Algorithm 1 summarizes the iterative TRO solver from [3], which has a proven convergence to the optimal value ρ^* and an optimal argument X^*. In the general case, the convergence is quadratic [4].

Since the EVCs obtained in Step 1 are unique up to a sign, the signs are chosen so as to minimize the norm of the difference between two iterations, which avoids "oscillations" in the sign of the columns when approaching convergence, as described in Step 2. We note that, at convergence, equation (8) becomes

$$(A - \rho^* B)X^* = X^*\Lambda^*, \tag{9}$$

but the sign of the columns of X^* can be arbitrarily chosen, hence the solution of Algorithm 1 is only defined up to an arbitrary sign of the columns of X^*, and all outputs solve the TRO problem (2). Note that all solutions of Algorithm 1 are TRO solutions, but not all TRO solutions are a solution of Algorithm 1, i.e., the solution set of the latter is more restricted (with only a sign ambiguity instead of a unitary transformation ambiguity). This is an important fact which will be exploited later on in the convergence proof of the distributed TRO algorithm. In the remaining of this paper, we denote by X^* a solution of (2) which is an output of Algorithm 1, i.e., X^* is a TRO solution that also satisfies (9).

III. Distributed TRO Algorithm in Fully-Connected Networks (FC-DTRO)

We start this section by describing the TRO problem in the context of signal processing before defining the problem in a distributed WSN setting. For the sake of an easy exposition, we initially present a distributed approach for solving the TRO problem in a fully-connected broadcast network in which a signal transmitted by any node is observable by all other nodes in the network. This will be generalized to other topologies in Section IV.

A. Adaptive TRO in Multi-Channel Signal Processing

In a multi-channel signal processing context, we consider two M-channel time signals $\mathbf{y}(t)$ and $\mathbf{v}(t) \in \mathbb{R}^M$, where t is a sample time index. Supposing the signals are zero-mean, their covariance matrix $R_{\mathbf{yy}} = \mathbb{E}[\mathbf{y}(t)\mathbf{y}(t)^T]$ and $R_{\mathbf{vv}} = \mathbb{E}[\mathbf{v}(t)\mathbf{v}(t)^T]$ would replace A and B, where $\mathbb{E}[\cdot]$ denotes the expectation operator. The solution of (2) could be interpreted as a discriminative spatial filter bank with M inputs and Q outputs, for which the total (summed) energy of the signals at the output can be used for discrimination between these two classes. Another example can be given in signal denoising [40], where $\mathbf{y}$ would represent "signal-plus-noise" segments and $\mathbf{v}$ would represent "noise-only" segments. In that case, the solution of (2) would act as an orthogonal spatial filter bank for joint dimensionality reduction and denoising purposes. In the remaining of this paper, we will adopt this signal processing context and therefore define $A = R_{\mathbf{yy}}$ and $B = R_{\mathbf{vv}}$, which leads to:

$$\begin{aligned} \underset{X}{\text{maximize}}\ & \varrho(X) \triangleq \frac{\text{tr}(X^T R_{\mathbf{yy}} X)}{\text{tr}(X^T R_{\mathbf{vv}} X)} \\ \text{subject to}\ & X \in \mathcal{S}. \end{aligned} \tag{10}$$

We assume that the signals are short-term stationary and ergodic, so that given a sufficiently large number of samples N of the signals $\mathbf{y}$ and $\mathbf{v}$, we can estimate the covariance matrices using an estimation well-suited for the application at hand, such as $R_{\mathbf{yy}} = \mathbb{E}[\mathbf{y}(t)\mathbf{y}(t)^T] \approx \frac{1}{N}\sum_{t=0}^{N-1}\mathbf{y}(t)\mathbf{y}(t)^T$, and similarly for $\mathbf{v}$. These assumptions would also imply that Algorithm 1 is able to track the changing statistical properties of the signals over time thereby making the iterative process adaptive. The different iterations of Algorithm 1 can then be spread out over different time windows of N samples. As will be discussed in Section III-C, the distributed method we propose to solve the TRO problem should also be viewed in such an adaptive context, where the iterations are spread out over time. The objective ϱ can then be approximated as:

$$\varrho(X) \approx \frac{\frac{1}{N}\sum_{t=0}^{N-1}||\sum_k X_k^T\mathbf{y}_k(t)||^2}{\frac{1}{N}\sum_{t=0}^{N-1}||\sum_k X_k^T\mathbf{v}_k(t)||^2}. \tag{11}$$

In the remaining parts of this paper, we will generally use the expectation operator $\mathbb{E}[\cdot]$ for ease of notation and mathematical tractability. However, in practice this operator will typically be replaced with a finite-window sample average as in (11).

B. WSN Formulation

We consider a WSN with K nodes belonging to a set $\mathcal{K} = \{1, \ldots, K\}$, where each node k measures two[1] M_k-dimensional signals $\mathbf{y}_k(t)$ and $\mathbf{v}_k(t)$. For notational convenience, we will omit the time index t unless referring to a specific time sample. Stacking every node's signals together, we obtain the $M-$dimensional signals $\mathbf{y}$ and $\mathbf{v}$, with $M = \sum_{k\in\mathcal{K}} M_k$ and $\mathbf{y} = \left[\mathbf{y}_1^T, \ldots, \mathbf{y}_K^T\right]^T$, $\mathbf{v} = \left[\mathbf{v}_1^T, \ldots, \mathbf{v}_K^T\right]^T$. Each node k has access to its own observations $\mathbf{y}_k$ and $\mathbf{v}_k$, but not to $\mathbf{y}_l$ and $\mathbf{v}_l$, gathered at nodes $l \in \mathcal{K}\backslash\{k\}$. This immediately eliminates using Algorithm 1 as is, because in Step 1, we require the network-wide signals' covariance matrices $R_{\mathbf{yy}}$ and $R_{\mathbf{vv}}$, which cannot be estimated in any single node, unless all the sensor observations are transmitted to a fusion center. This option is not a solution we will consider because it creates a bandwidth bottleneck. The approach we propose will solve the TRO problem in a distributed and adaptive fashion, by allowing transmission of compressed local signals.

[1]Note that $\mathbf{y}_k$ and $\mathbf{v}_k$ could also denote the same sensor signal recorded in two different conditions as explained in Section II-A.

In this section, we assume that the network is fully-connected, i.e., each node broadcasts its (compressed) observations to all other nodes in the network. Denoting the current iteration as i, we consider that all nodes k linearly compress their observations using an iteration-dependent compression matrix $F_k^i \in \mathbb{R}^{M_k \times P}$, with $P \leq M_k$:

$$\widehat{\mathbf{y}}_k^i = F_k^{iT}\mathbf{y}_k, \widehat{\mathbf{v}}_k^i = F_k^{iT}\mathbf{v}_k, \tag{12}$$

where $\widehat{\mathbf{y}}_k^i$ and $\widehat{\mathbf{v}}_k^i$ are the $P-$dimensional compressed signals of node k at iteration i, which will be broadcast to every other node in the network. This results in a compression ratio of M_k/P at every node $k \in \mathcal{K}$. We will later show that if P is set equal to Q, i.e., the number of columns of X in the TRO problem (10), we can still compute the centralized TRO solution in a distributed fashion, even though each node sends compressed sensor data at only Q/M_k of the original rate.

C. Algorithm Description

A naive approach for solving the TRO problem in a distributed context would be to compute the EVD in Step 1 of Algorithm 1 using a distributed (G)EVD algorithm such as, e.g., [40] [41]. The reason why we do not consider this approach is because these distributed algorithms are iterative themselves. Therefore, before arriving at Step 2 in Algorithm 1, we would need to wait for the distributed (G)EVD procedure to converge, thereby creating nested iterations inside the outer-loop iterations of Algorithm 1. This would considerably slow down the convergence speed and it would generate an algorithm that operates at two different time scales (with fast iterations inside slow iterations). The method we propose solves (10) without nested iterations, but instead *interleaves* the iterations of Algorithm 1 with those of the distributed GEVD algorithm in [40] and therefore runs at a single time scale. We refer to this algorithm as the (Fully-Connected) Distributed Trace Ratio Optimization or (FC)-DTRO algorithm.

To derive the FC-DTRO algorithm's steps, we will start by noting that the global problem (10) can be reformulated by partitioning the variables and signals of interest into blocks, each corresponding to a node. Then, we will observe that with well-chosen compression matrices F_k^i from (12), the nodes are able to partially recreate the global problem (10), albeit with extra constraints due to the reduced information available at each node (this will result in equation (20) further on). Solving this problem at node q will lead to a local estimation of the block of the global variable X which corresponds to node q and an updating rule for the other nodes. This procedure is sequentially repeated at every node in the network. In the remaining of this section, we will formalize these steps in a rigorous fashion.

Let us partition the network-wide variable $X \in \mathbb{R}^{M \times Q}$ as:

$$X = \left[X_1^T, \ldots, X_K^T\right]^T, \tag{13}$$

where $X_k \in \mathbb{R}^{M_k \times Q}$, corresponds to the part of X that is applied to the sensor signals of node k. Then, the objective in (10) can be rewritten as a sum of the elements in the partitioning (13):

$$\varrho(X) = \frac{\mathrm{tr}\Big(\sum\limits_{k,l} X_k^T R_{\mathbf{y}_k\mathbf{y}_l} X_l\Big)}{\mathrm{tr}\Big(\sum\limits_{k,l} X_k^T R_{\mathbf{v}_k\mathbf{v}_l} X_l\Big)} = \frac{\mathbb{E}\Big[||\sum_k X_k^T\mathbf{y}_k||^2\Big]}{\mathbb{E}\Big[||\sum_k X_k^T\mathbf{v}_k||^2\Big]}, \tag{14}$$

with $k, l \in \mathcal{K}$, $R_{\mathbf{y}_k\mathbf{y}_l} = \mathbb{E}\big[\mathbf{y}_k\mathbf{y}_l^T\big]$ and $R_{\mathbf{v}_k\mathbf{v}_l} = \mathbb{E}\big[\mathbf{v}_k\mathbf{v}_l^T\big]$. This allows to express the network-wide objective using local signals $\mathbf{y}_k, \mathbf{v}_k$. Node k is responsible for updating X_k, which can be viewed as its local variable. In the FC-DTRO algorithm, we set the compression matrices as:

$$F_k^i = X_k^i, \tag{15}$$

with $P = Q$. Therefore, combining (12) and (15), we obtain the compressed version of signal $\mathbf{y}_k$ at node k and iteration i:

$$\widehat{\mathbf{y}}_k^i = X_k^{iT}\mathbf{y}_k. \tag{16}$$

The projection of the network-wide signal $\mathbf{y}$ onto the subspace spanned by the columns of X^i is then given by:

$$\widehat{\mathbf{y}}^i \triangleq X^{iT}\mathbf{y} = \sum_{k\in\mathcal{K}} X_k^{iT}\mathbf{y}_k = \sum_{k\in\mathcal{K}} \widehat{\mathbf{y}}_k^i. \tag{17}$$

Similar arguments hold for $\widehat{\mathbf{v}}_k^i$ and $\widehat{\mathbf{v}}^i$. We see that the FC-DTRO algorithm uses X_k^i both as a compression matrix and as the part of the estimation of the optimal solution X^*. As a result, (14) becomes:

$$\varrho(X^i) = \frac{\mathbb{E}\Big[||X^{iT}\mathbf{y}||^2\Big]}{\mathbb{E}\Big[||X^{iT}\mathbf{v}||^2\Big]} = \frac{\mathbb{E}\Big[||\sum_{k\in\mathcal{K}} \widehat{\mathbf{y}}_k^i||^2\Big]}{\mathbb{E}\Big[||\sum_{k\in\mathcal{K}} \widehat{\mathbf{v}}_k^i||^2\Big]}, \tag{18}$$

such that each node is able to evaluate the network-wide objective from the compressed data only.

The updates on the network will be done in a sequential round-robin fashion, with only one updating node at a time. Suppose the updating node at iteration i is node q, which receives compressed observations from other nodes $\widehat{\mathbf{y}}_k^i$, $\widehat{\mathbf{v}}_k^i$, $k \in \mathcal{K}\backslash\{q\}$, as defined in (16). The total information available at the updating node q, namely node q's own sensor signals $\mathbf{y}_q$ and the compressed signals $\widehat{\mathbf{y}}_k^i$ received from the other nodes $k \in \mathcal{K}\backslash\{q\}$, can be stacked to form the $\widetilde{M}_q-$dimensional vector:

$$\widetilde{\mathbf{y}}_q^i = \Big[\mathbf{y}_q^T, \widehat{\mathbf{y}}_1^{iT}, \ldots, \widehat{\mathbf{y}}_{q-1}^{iT}, \widehat{\mathbf{y}}_{q+1}^{iT}, \ldots, \widehat{\mathbf{y}}_K^{iT}\Big]^T, \tag{19}$$

where $\widetilde{M}_q = M_q + Q(K-1)$, and similarly for $\widetilde{\mathbf{v}}_q^i$. Our aim by defining this variable is to express an equivalent problem to (10) at node q, with the available data.

At the beginning of iteration i, node q estimates the covariance matrices $R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} = \mathbb{E}\big[\widetilde{\mathbf{y}}_q^i\widetilde{\mathbf{y}}_q^{iT}\big], R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q} = \mathbb{E}\big[\widetilde{\mathbf{v}}_q^i\widetilde{\mathbf{v}}_q^{iT}\big] \in \mathbb{R}^{\widetilde{M}_q\times\widetilde{M}_q}$, by collecting a contiguous stream of N time samples of $\widetilde{\mathbf{y}}_q^i$ and $\widetilde{\mathbf{v}}_q^i$. Then, defining the variable $\widetilde{X}_q \in \mathbb{R}^{\widetilde{M}_q\times Q}$ for node q, we create the compressed version of the original TRO problem (10), based solely on the data available at node q:

$$\begin{aligned} \underset{\widetilde{X}_q}{\text{maximize}}\ & \tilde{\varrho}_q^i(\widetilde{X}_q) \triangleq \frac{\mathrm{tr}(\widetilde{X}_q^T R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q}\widetilde{X}_q)}{\mathrm{tr}(\widetilde{X}_q^T R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}\widetilde{X}_q)} \\ \text{subject to}\ & \widetilde{X}_q \in \widetilde{\mathcal{S}}_q^i, \end{aligned} \tag{20}$$

where $\widetilde{\mathcal{S}}_q^i \triangleq \{\widetilde{X}_q \in \mathbb{R}^{\widetilde{M}_q\times Q} : \widetilde{X}_q^T K_q^i \widetilde{X}_q = I_Q\}$ and $K_q^i \in \mathbb{R}^{\widetilde{M}_q\times\widetilde{M}_q}$ is an orthonormalizing matrix which we will define in the next paragraph, in (27). Note that the local problem (20) at node q involves $\widetilde{M}_q \times \widetilde{M}_q$ matrices instead of $M \times M$ matrices, and is therefore much smaller in size than (10).

A question that now arises is how the network-wide problem (10) can be linked to the local problem (20) at iteration i. To this end, we introduce the matrix C_q^i, which links the network-wide signal $\mathbf{y}$ to the local signals $\widetilde{\mathbf{y}}_q^i$:

$$\widetilde{\mathbf{y}}_q^i = C_q^{iT}\mathbf{y}. \tag{21}$$

Looking at the definition of both variables, we see that:

$$C_q^i = \left[\begin{array}{c|c|c} & B^i_{<q} & \\ \hline A_q^T & 0 & 0 \\ \hline & 0 & B^i_{>q} \end{array}\right] \in \mathbb{R}^{M\times\widetilde{M}_q}, \tag{22}$$

where:

$$A_q = [0_{M_q \times \sum_{j<q} M_j} | I_{M_q} | 0_{M_q \times \sum_{j>q} M_j}], \tag{23}$$

with $0_{m\times n}$ denoting an $m \times n$ all-zero matrix and $B^i_{<q}$ is a block-diagonal matrix such that $B^i_{<q} = \text{BlkDiag}(X^i_1, \ldots, X^i_{q-1})$, i.e. containing $X^i_1, \ldots, X^i_{q-1}$ on its (block-)diagonal and similarly, we have $B^i_{>q} = \text{BlkDiag}(X^i_{q+1}, \ldots, X^i_{q+1})$. We note that, based on (21), the local and network-wide covariance matrices are related by:

$$R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} = C^{iT}_q R_{\mathbf{yy}} C^i_q. \tag{24}$$

Plugging (24) into (20) implies that the network-wide variable X in (10) is now defined by the following parameterization at node q:

$$X = C^i_q \widetilde{X}_q = \begin{bmatrix} X^i_1 \boxed{G_1} \\ \vdots \\ X^i_{q-1} \boxed{G_{q-1}} \\ \boxed{X_q} \\ X^i_{q+1} \boxed{G_{q+1}} \\ \vdots \\ X^i_K \boxed{G_K} \end{bmatrix}, \tag{25}$$

where X_q is $M_q \times Q$, G_k's are $Q \times Q$ and $\widetilde{X}_q$ is partitioned as:

$$\widetilde{X}_q = \left[X^T_q, G^T_1, \ldots, G^T_{q-1}, G^T_{q+1}, \ldots, G^T_K\right]^T. \tag{26}$$

The orthogonality constraint $X^T X = I_Q$ can then be written as $\widetilde{X}^T_q K^i_q \widetilde{X}_q = I_Q$, obtained by the substitution in (25), with:

$$\begin{aligned} K^i_q &= C^{iT}_q C^i_q \\ &= \text{BlkDiag}(I_{M_q}, L^i_1, \ldots, L^i_{q-1}, L^i_{q+1}, \ldots, L^i_K), \end{aligned} \tag{27}$$

and:

$$L^i_k = X^{iT}_k X^i_k. \tag{28}$$

Proposition 1. *At each iteration i, the local objective value computed at the updating node q based on (20) is the same as the global objective value, i.e.,*

$$\varrho(X) = \tilde{\varrho}^i_q(\widetilde{X}_q), \tag{29}$$

and $\widetilde{X}_q \in \widetilde{\mathcal{S}}^i_q$ implies $X \in \mathcal{S}$.

Proof. Using the relationships $R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} = C^{iT}_q R_{\mathbf{yy}} C^i_q$ and $X = C^i_q \widetilde{X}_q$ from (24) and (25) respectively, we have $\tilde{\varrho}^i_q(\widetilde{X}_q) = \varrho(C^i_q \widetilde{X}_q) = \varrho(X)$. On the other hand, by the definition of K^i_q in (27), $I_Q = \widetilde{X}^T_q K^i_q \widetilde{X}_q = \widetilde{X}^T_q C^{iT}_q C^i_q \widetilde{X}_q = X^T X$. □

This result shows that the local objective value computed at node q is the same as the global objective value and that a local variable $\widetilde{X}_q$ which belongs to the constraint set of Problem (20) leads to a global variable satisfying the constraints of the global problem (10). Therefore, if node q optimizes the local problem (20), it optimizes a parameterized version of the global problem (10). In (25), the variables that are computed by node q are highlighted by a square around them, thus, the parameterization in (25) implies that only X_q can be optimized freely by node q, whereas the variables X_k, $k \neq q$, are constrained to maintain the same column space as X^i_k, i.e., the estimation of X_k at the beginning of iteration i (this can be seen from the $G-$matrices in (25)). This additional constraint makes the FC-DTRO method different from Algorithm 1, and makes it necessary to change the updating node between iterations, to be able to freely update the corresponding local variables.

Nevertheless, we still follow similar steps as in Algorithm 1, creating the analogous local auxiliary problem at node q, similar to (5) and (7):

$$\widetilde{X}^{i+1}_q \triangleq \underset{\widetilde{X}_q \in \widetilde{\mathcal{S}}^i_q}{\text{argmax}} \ \text{tr}\left(\widetilde{X}^T_q (R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}) \widetilde{X}_q\right), \tag{30}$$

where ρ^i can be locally computed using node q's own observations and the ones received from other nodes, following the relationship given in (18):

$$\rho^i = \frac{\mathbb{E}\left[||X^{iT}_q \mathbf{y}_q + \sum_{k\in\mathcal{K}\setminus\{q\}} \widehat{\mathbf{y}}^i_k||^2\right]}{\mathbb{E}\left[||X^{iT}_q \mathbf{v}_q + \sum_{k\in\mathcal{K}\setminus\{q\}} \widehat{\mathbf{v}}^i_k||^2\right]}. \tag{31}$$

Due to the constraint set $\widetilde{\mathcal{S}}^i_q$, (30) is a GEVD problem (as opposed to an EVD problem at the equivalent stage of Algorithm 1) for the matrix pencil $(R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}, K^i_q)$. To be able to solve this problem, we note that node q also needs L^i_k's, which are therefore sent by their corresponding node to q at the beginning of the iteration[2].

At iteration i, node q is therefore solving for the variable $\widetilde{X}_q$ the following GEVD problem:

$$(R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}) \widetilde{X}_q = K^i_q \widetilde{X}_q \widetilde{\Lambda}_q, \tag{32}$$

where $\widetilde{\Lambda}_q$ is a $Q \times Q$ diagonal matrix containing the GEVLs of the matrix pencil $(R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}, K^i_q)$. In the sequel, we define $X = \text{GEVC}_Q(A, B)$ as the $Q-$column matrix which contains in its columns the generalized eigenvectors belonging to the Q largest GEVLs of the matrix pencil (A, B), scaled such that $X^T B X = I_Q$. The solution of (32) can then be described as $\widetilde{X}^{i+1}_q = \text{GEVC}_Q(R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}, K^i_q)$.

Remark 1. *We assume in the remaining of this paper that both matrices in the matrix pencil $(R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}, K^i_q)$ are full rank and their largest $Q+1$ GEVLs are all distinct, so that the solution $\widetilde{X}_q$ in (32) is well-defined. In contrived cases where this assumption does not hold, some technical modifications to the algorithm are necessary to make the problem well-defined. For the sake of an easy exposition, we make abstraction of this problem for the time being and refer the reader to Appendix C for precisions.*

The final step of the FC-DTRO algorithm is to communicate the updates to the other nodes. For that, we return to the parameterization $X = C^i_q \widetilde{X}_q$. As discussed previously, node q has the full freedom to optimize its local variable X_q, therefore, it can take the first M_q rows of $\widetilde{X}^{i+1}_q$ as the estimate X^{i+1}_q, whereas the other nodes $k \neq q$ are constrained to preserve their original column space. Let us partition $\widetilde{X}^{i+1}_q$ as in (26). Following the parameterization defined in (25) (based on (13) and (22)), we obtain:

$$X^{i+1}_k = \begin{cases} X^{i+1}_q & \text{if } k = q \\ X^i_k G^{i+1}_k & \text{if } k \neq q \end{cases}. \tag{33}$$

After solving (32), node q partitions the solution as in (26) and communicates the G_k's to all other nodes $k \neq q$, so that they can update their local variable X_k using (33). We note that the global variable X is thus never fully constructed

[2]Note that this can be assumed to be a negligible extra communication cost compared to the transmission of N time samples of $\widehat{\mathbf{y}}^i_k$ / $\widehat{\mathbf{v}}^i_k$ in each iteration to estimate $R^i_{\widetilde{\mathbf{y}}_q \widetilde{\mathbf{y}}_q}$ and $R^i_{\widetilde{\mathbf{v}}_q \widetilde{\mathbf{v}}_q}$.

Algorithm 2: Fully-Connected Distributed Trace Ratio Optimization (FC-DTRO)

output: X^*, ρ^*

X^0 initialized randomly, $i \leftarrow 0$

repeat

Choose the main updating node as $q \leftarrow (i \mod K) + 1$

1) Node q receives $L_k^i = X_k^{iT} X_k^i$ and $\widehat{\mathbf{y}}_k^i(t)$, $\widehat{\mathbf{v}}_k^i(t)$ for $t = iN, \ldots, (i+1)N-1$ from all other nodes $k \neq q$

2) Node q estimates $R^i_{\widehat{\mathbf{y}}_q \widehat{\mathbf{y}}_q}, R^i_{\widehat{\mathbf{v}}_q \widehat{\mathbf{v}}_q}$ based on the stacking defined in (19)

3) ρ^i is updated as in (31)

4) $\widetilde{X}_q^{i+1} \leftarrow \text{GEVC}_Q(R^i_{\widehat{\mathbf{y}}_q \widehat{\mathbf{y}}_q} - \rho^i R^i_{\widehat{\mathbf{v}}_q \widehat{\mathbf{v}}_q}, K_q^i)$, where K_q^i is given in (27)

5) $\widetilde{X}_q^{i+1} \leftarrow \widetilde{X}_q^{i+1} U^{i+1}$, where $U^{i+1} = \text{argmin}_{U \in \mathcal{D}} ||X_q^{i+1} U - X_q^i||_F$ and $\mathcal{D}$ is the set of $Q \times Q$ signature matrices

6) Partition $\widetilde{X}_q^{i+1}$ as in (26), broadcast $G_k^{i+1}, \forall k \neq q$

7) Every node updates X_k^{i+1} according to (33)

$i \leftarrow i + 1$

until *Convergence*

Remark: For any iteration-updating node pair i, q, node q can locally compute the projected data $\{X^{iT}\mathbf{y}(t)\}_{t=iN,\ldots,(i+1)N-1}$ as $\{\sum_k \widehat{\mathbf{y}}_k^i(t)\}_{t=iN,\ldots,(i+1)N-1}$

in the algorithm, i.e., none of the nodes need access to the network-wide variable. In Section III-E we will see that under mild assumptions, communicating the matrices G_k is not even necessary.

The steps of the FC-DTRO algorithm are summarized in Algorithm 2. As in the case of Algorithm 1, Step 5 is added to resolve the sign ambiguity in the GEVCs found in (32) to avoid arbitrarily chosen signs by the algorithm. In Step 1, it is important to note that the same block of samples is never communicated twice, i.e., the next updating node will use a stream of N new samples to perform the update. This inherently makes the algorithm adaptive and also allows to track slow changes in the signal statistics as long as the rate of change is slower than the convergence rate of the algorithm. Figure 1 provides a block diagram representation of the tasks completed at node q (Steps 3 and 5 omitted).

D. Convergence of the FC-DTRO Algorithm

From Algorithm 2, we note that if we were to remove the updating of ρ (Step 3) in the FC-DTRO algorithm, we would obtain an instance of the distributed adaptive covariance matrix generalized eigenvector estimation (DACGEE) algorithm from [40], applied to the GEVD problem for the matrix pencil $(R_{\mathbf{yy}} - \rho R_{\mathbf{vv}}, I_M)$ for an arbitrary (fixed) value ρ. This demonstrates that the FC-DTRO algorithm interleaves iterations of Algorithm 1 with the iterations of a distributed GEVD algorithm. However, we cannot rely on the convergence of Algorithm 1 to justify convergence of the FC-DTRO algorithm because in each iteration, the latter solves partially, and not fully, the network-wide GEVD problem. On the other hand, the convergence of the DACGEE algorithm in [40] does not imply convergence of the FC-DTRO algorithm either, as ρ changes at each iteration, changing the eigenvalue problem to solve at each iteration.

Nevertheless, it can be shown that the FC-DTRO algorithm converges, as formalized in Theorem 2.

Definition 1. *We define the equality up to a sign of the columns as $\overset{*}{=}$, i.e., $X \overset{*}{=} Y$ if $X = YD$ with $D \in \mathcal{D}$, where $\mathcal{D}$ is the set of $Q \times Q$ signature matrix, i.e., diagonal matrices containing either 1 or -1 on their diagonal.*

Theorem 2. *For any initialization $X^0 \in \mathbb{R}^{M \times Q}$, the updates of Algorithm 2 satisfy $\lim_{i \to +\infty} X^i \overset{*}{=} X^*$ where X^* is a solution of Algorithm 1, i.e., Algorithm 2 converges to a solution of the TRO problem (10).*

Proof. See Appendix A. □

To arrive to this conclusion, we require two intermediate results. The global strategy is to first show that for iterations $i > 0$, the sequence $\{\rho^i\}_{i>0}$ is monotonic non-decreasing. Since the sequence has an upper bound ρ^*, we will be able to show convergence in the objective. Then, we will show that any equilibrium point of Algorithm 2 can only be a solution of (10). This will guarantee that the algorithm does not "get stuck" in non-optimal points. These two results, summarized in Lemmas 1 and 2 of Appendix A, will then allow us to conclude that the FC-DTRO algorithm converges to the optimal argument X^*, up to a sign ambiguity in the columns.

E. Reduced Communication Overhead

As mentioned previously, the updating rule (33) allows the updating node q to fully choose a new estimate X_q, but for $k \neq q$, X_k must preserve its original column space, described by the relationship $X_k^{i+1} = X_k^i G_k^{i+1}$. In this section, we elaborate on this latter update, which enforces to communicate the matrices G_k from node q to the other nodes $k \neq q$. We will see that in the case of fully-connected networks, the information carried by these G_k matrices is not crucial for Algorithm (2) to converge.

There are two major contributions of the matrices G_k to Algorithm 2. The first one is the guarantee of orthogonality, which can be seen through the relationship $X = C_q^i \widetilde{X}_q$. Indeed, we have a guarantee that at any iteration $i+1$, the network-wide variable $X^{i+1} \in \mathcal{S}$, because $\widetilde{X}_q^{i+1} \in \widetilde{\mathcal{S}}_q^i$ as detailed in Proposition 1. The second use is to preserve correspondence between the local variables X_k, i.e., to avoid cases where the individual X_k's converge to submatrices of *different* optimal solutions X^* (remember that a TRO solution X^* obtained using Algorithms 1 and 2 is only defined up to an arbitrary change in the signs of the columns). In this case, the stacked matrix of all X_k's will not be a valid solution in itself.

The most important information content of the matrices G_k is actually found in the sign of the entries on their diagonal. This information is collected in the diagonal, $Q \times Q$ matrix D_k, i.e.:

$$D_k = \text{sgn}(G_k \circ I_Q), \tag{34}$$

where $\circ$ denotes the element-wise multiplication (Hadamard product); it can be shown that using the following alternative updating rule instead of (33):

$$X_k^{i+1} = \begin{cases} X_q^{i+1} & \text{if } k = q \\ X_k^i D_k^{i+1} & \text{if } k \neq q \end{cases}, \tag{35}$$

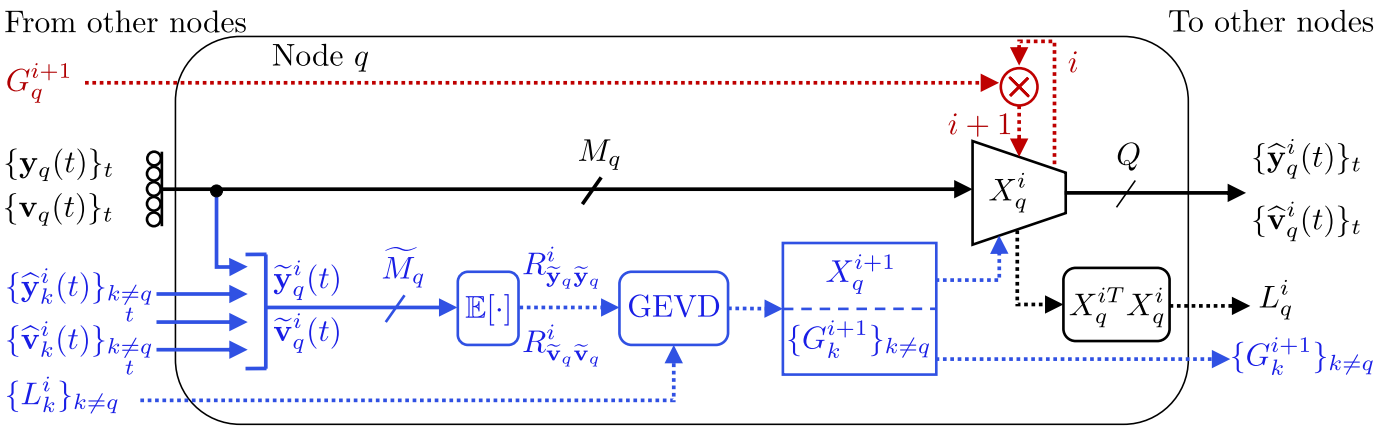

Fig. 1: Block diagram representation of the steps followed by a given node q. The black part is executed at any time t in each node. The colored parts are executed at each iteration increment $i \to i+1$. The blocks in blue are executed when node q is the main updating node. Otherwise, the part in red is carried out. Dashed lines correspond to a transmission of parameters (independent of the sample time t), full lines correspond to a transmission of (compressed) signal observations (one for each sample time). At any given iteration i, the time index t belongs to $\{iN, \dots, (i+1)N-1\}$. In Section III-E, we will show that the red part can actually be omitted without affecting convergence, such that no matrix G_k have to be shared. For visual simplicity, we do not represent the computation of ρ^i, nor Step 5 in the figure.

does not affect convergence. This means that only the signs of the diagonal of D_k have to be transmitted (rather than the full G_k's). We note that using the update rule (35) requires node q to compute ρ^{i+1} at the end of iteration i by plugging $\widetilde{X}_q^{i+1}$ in (20) and communicate it to the next updating node, i.e., ρ cannot be estimated as in Step 3 of Algorithm 2 if we use the updating rule (35) instead of (33). This change is valid because of the stationarity assumptions of the signals $\mathbf{y}$ and $\mathbf{v}$, meaning that the evaluation of ρ at the end of iteration i and node q is equivalent to computing it at iteration $i+1$ and node $q+1$ in Step 3 of Algorithm 2. Theorem 3 recapitulates the results of using updating rule (35) instead of (33).

Theorem 3. *Suppose X^0 is initialized randomly and $\underline{X}^0 = X^0$. We denote by $\{X^i\}_i$ the sequence generated by Algorithm 2 using the updating rule (33) and $\{\underline{X}^i\}_i$ the one generated by Algorithm 2 using the updating rule (35) instead of (33). Then, we have the following properties:*

P1) $\underline{X}_q^{i+1} \overset{*}{=} X_q^{i+1}$, *for the updating node q at iteration i.*
P2) $\lim_{i\to+\infty} \underline{X}^i \overset{*}{=} \lim_{i\to+\infty} X^i \overset{*}{=} X^*$

Proof. See Appendix B. □

Remark 2. *Keeping the same notation as in Theorem 3, let us additionally underline the matrices obtained when running Algorithm 2 using the updating rule (35). Then, we have $\underline{\widetilde{X}}_q^{(i+1)T}\underline{C}_q^{iT}\underline{C}_q^i\underline{\widetilde{X}}_q^{i+1} = I_Q$ since $\underline{\widetilde{X}}_q^{i+1}$ solves (30). However, with this updating scheme, we note that $\underline{X}^{i+1} \neq \underline{C}_q^i\underline{\widetilde{X}}_q^{i+1}$. Therefore, $\underline{X}^{iT}\underline{X}^i \neq I_Q$ in general, but $\lim_{i\to+\infty}\underline{X}^{iT}\underline{X}^i = I_Q$ from the convergence result of Theorem 3. In other words, the orthogonality constraint is not satisfied in each iteration, yet it will be satisfied in the limit when the solution converges.*

The communication overhead can be further reduced by not sending matrices D_k either, i.e.:

$$X_k^{i+1} = \begin{cases} X_q^{i+1} & \text{if } k = q \\ X_k^i & \text{if } k \neq q \end{cases}. \tag{36}$$

However, this updating scheme comes at the cost of not guaranteeing global convergence, but local convergence is preserved, as discussed in the following result.

Corollary 1. *We denote by $\{X^i\}_i$ the sequence generated by Algorithm 2 using the updating rule (36) and define the partitioning X^* as:*

$$X^* = \left[(X_1^*)^T, \dots, (X_K^*)^T\right]^T, \tag{37}$$

where X^* is an $M \times Q$ matrix, solution of Algorithm 1. Then, $\lim_{i\to+\infty} X_k^i \overset{*}{=} X_k^*$ for all $k \in \mathcal{K}$, but P2) is not necessarily satisfied, i.e., we achieve convergence locally but not necessarily globally.

Proof. See Appendix B. □

In practice, one can use (36) for the majority of the time to avoid sharing G_k or D_k matrices, while using (33) or (35) once in a while to "match" the local solutions to each other. In theory, this should only happen once after convergence of the algorithm, yet for adaptive/dynamic scenarios, this matching may have to be performed several times.

IV. Topology-Independent DTRO Algorithm (TI-DTRO)

In practical applications, a WSN is often not fully-connected. In this section, we study how Algorithm 2 can be adapted to any connected topology (i.e., not necessarily fully-connected). We will first describe the data flow in these networks starting with the star network because of the way this particular topology will naturally lead to the others. Similar discussions have been made in [15], [41]–[44] for other distributed algorithms with different objective functions. In the following parts, we denote by $\mathcal{N}_k$ the set containing the neighboring nodes of node k, i.e., all the nodes that can communicate with node k (excluding node k itself).

A. Data Flow

In this subsection, we describe how the data flow of fused/compressed signal observations is arranged in networks that are not fully-connected. The transmission of the G_k and L_k parameter matrices will be discussed in the next subsection, where we will go into the details of how to modify Algorithm 2 to operate in such networks.

1) Star Networks: Suppose we have K nodes arranged in a star fashion, i.e., all nodes are leaf nodes (nodes with only a single neighbor) except one center node which is connected to all others. In Figure 2, the graph labeled "Star" is an example of such networks. As in the case of fully-connected networks, each node k collects its M_k-dimensional observations $\mathbf{y}_k$ and $\mathbf{v}_k$, has an estimation X_k^i at iteration i of its local projection matrix, and can communicate the compressed observations $\widehat{\mathbf{y}}_k^i = X_k^{iT}\mathbf{y}_k$ and $\widehat{\mathbf{v}}_k^i = X_k^{iT}\mathbf{v}_k$. While the center node c can broadcast its compressed sensor signals to all leaf nodes, the leaf nodes $k \in \mathcal{K}\backslash\{c\}$ can only share these vectors with the central node c. Referring to (31), since the global objective depends on the sum of the $\widehat{\mathbf{y}}_k^i$'s, the strategy we propose is for the central node c to sum the received compressed signals and to add its own compressed signal observations to the sum before forwarding the result to the main updating node. Assume $q \neq c$ is the updating node, then the center node c first receives the compressed observations $\widehat{\mathbf{y}}_k^i$ from all nodes $k \in \mathcal{N}_c\backslash\{q\}$, and computes:

$$\widehat{\mathbf{y}}_{c\to q}^i = X_c^{iT}\mathbf{y}_c + \sum_{k\in\mathcal{N}_c\backslash\{q\}} \widehat{\mathbf{y}}_k^i, \tag{38}$$

which is then communicated to node q, $\widehat{\mathbf{v}}_{c\to q}^i$ is similarly defined. From the perspective of node q, the network is therefore perceived as having only two nodes, namely itself and the central node c. Analogously to (19), q stacks all the available data in:

$$\widetilde{\mathbf{y}}_q^i = \left[\mathbf{y}_q^T, \widehat{\mathbf{y}}_{c\to q}^{iT}\right]^T, \tag{39}$$

and similarly for $\widetilde{\mathbf{v}}_q^i$ which are then both used for estimating $\widetilde{X}_q^{i+1}$ as described in Algorithm 2. On the other hand, when the central node updates, the discussion is the same as in the fully-connected case, since every node can reach c.

2) Tree Topologies: Trees are networks without cyclic paths, such as the graphs labeled "Tree", "Regular Tree" and "Path" in Figure 2. In these networks, we can apply a similar strategy as in the star topology case. The idea behind gathering information at the updating node q comes from the "sum-and-forward" strategy. When a node $k \neq q$ receives data from all its neighbors except one, it sums the signals, adds its own compressed signals $\widehat{\mathbf{y}}_k^i$ and $\widehat{\mathbf{v}}_k^i$ and forwards the summed signal to the remaining neighbor.

Mathematically, consider a tree with K nodes, and within this network, suppose node k has received the summed compressed signals from all its neighbors except one (here denoted as node n). This is the trigger for node k to generate the fused signal:

$$\widehat{\mathbf{y}}_{k\to n}^i = X_k^{iT}\mathbf{y}_k + \sum_{l\in\mathcal{N}_k\backslash\{n\}} \widehat{\mathbf{y}}_{l\to k}^i, \tag{40}$$

and transmit the results to node n. If each node follows this simple "triggering" rule, the data will naturally start flowing towards node q (being fused along the way). This is illustrated in Figure 3. Note that the data flow will automatically initiate at the leaf nodes, as these have only one neighbor. Eventually, the data gathered at the updating node q is:

$$\widehat{\mathbf{y}}_{n\to q}^i = X_n^{iT}\mathbf{y}_n + \sum_{k\in\mathcal{N}_n\backslash\{q\}} \widehat{\mathbf{y}}_{k\to n}^i = \sum_{k\in\mathcal{B}_{nq}} \widehat{\mathbf{y}}_k^i,\ \forall n \in \mathcal{N}_q, \tag{41}$$

where $\mathcal{B}_{nq}$ is the subgraph containing node n when the link between nodes n and q is cut, as shown in Figure 3.

Stacking all signals available at node q results in:

$$\widetilde{\mathbf{y}}_q^i = \left[\mathbf{y}_q^T, \widehat{\mathbf{y}}_{n_1\to q}^{iT}, \ldots, \widehat{\mathbf{y}}_{n_{|\mathcal{N}_q|}\to q}^{iT}\right]^T, \tag{42}$$

with $\{n_1, \ldots, n_{|\mathcal{N}_q|}\} = \mathcal{N}_q$, and we define $\widetilde{\mathbf{v}}_q^i$ similarly. Even though node q does not have access to all $\widehat{\mathbf{y}}_k^i$'s individually, it is still able to compute the global objective. Indeed, as shown in (17) and (18), evaluating the global objective only requires the sum of the compressed signals $\widehat{\mathbf{y}}_k^i$ and $\widehat{\mathbf{v}}_k^i$, which node q has access to, since:

$$X_q^{iT}\mathbf{y}_q + \sum_{n\in\mathcal{N}_q} \widehat{\mathbf{y}}_{n\to q}^i = \sum_{k\in\mathcal{K}} \widehat{\mathbf{y}}_k^i, \tag{43}$$

and similarly for the data vector $\mathbf{v}$, which leads to:

$$\rho^i = \frac{\mathbb{E}\left[||X_q^{iT}\mathbf{y}_q + \sum_{n\in\mathcal{N}_q} \widehat{\mathbf{y}}_{n\to q}^i||^2\right]}{\mathbb{E}\left[||X_q^{iT}\mathbf{v}_q + \sum_{n\in\mathcal{N}_q} \widehat{\mathbf{v}}_{n\to q}^i||^2\right]}. \tag{44}$$

These results are similar to the star network case except for the presence of a branch of hidden nodes "behind" the neighbors of the updating node q, but from the perspective of any single node, the network is perceived as a star, to which it is the center node. Note that, in case *all* nodes need access to the projected data $X^{iT}\mathbf{y}$ and $X^{iT}\mathbf{v}$ for *each* sample time, then, the observations of the fused signals $\widehat{\mathbf{y}}^i$:

$$\widehat{\mathbf{y}}^i = X_q^{iT}\mathbf{y}_q + \sum_{n\in\mathcal{N}_q}\sum_{k\in\mathcal{B}_{nq}} X_k^{iT}\mathbf{y}_k = \widehat{\mathbf{y}}_q^i + \sum_{n\in\mathcal{N}_q} \widehat{\mathbf{y}}_{n\to q}^i, \tag{45}$$

and $\widehat{\mathbf{v}}^i$ (defined similarly), as computed in node q should be disseminated throughout the network, which would double the required communication cost at all non-leaf nodes.

3) General Connected Graphs: We can easily extend previous discussions to the case of any random connected graph, such as the "Connected" graph in Figure 2. We denote by $\mathcal{G}$ the graph representing the network. The idea is to prune the network so that the resulting graph is a tree, where in each iteration, a different tree can be selected. We denote $\mathcal{T}^i(\mathcal{G}, q)$ as the tree selected in iteration i, in which q is the updating node. The pruning function $\mathcal{T}^i$'s dependence on q is to achieve better efficiency. An important property of $\mathcal{T}^i$ should be to avoid cutting edges between the updating node q and its neighbors to keep a higher number of degrees of freedom at the updating node, i.e., to keep $\mathcal{N}_q$ as large as possible. Indeed, note that the dimension of the stacked vector (42) then becomes larger, which means that the local update at node q has more degrees of freedom to achieve a higher objective $\tilde{\varrho}_q^i(\widetilde{X}_q^{i+1}) = \varrho(X^{i+1})$ based on the maximization in (30). An example of a pruning function achieving these desired properties is the shortest path pruning [45], i.e., the connections in the resulting tree guarantee a minimal length path between all nodes $k \neq q$ and q, which has the additional advantage that it reduces the communication cost for disseminating the projected data $\widehat{\mathbf{y}}^i$ and $\widehat{\mathbf{v}}^i$ throughout the network as well as the G_k and L_k parameters, as discussed in the next subsection.

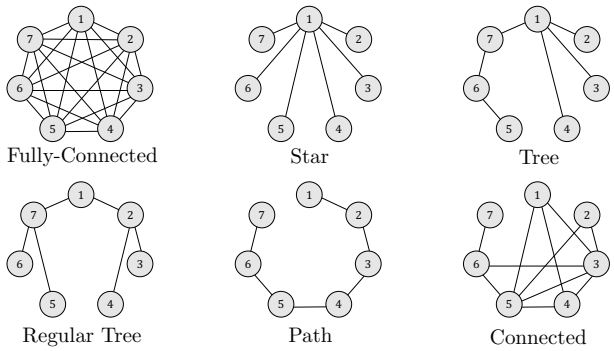

Fig. 2: Examples of various network topologies with seven nodes.

Algorithm 3: Topology-Independent Distributed Trace Ratio Optimization (TI-DTRO)

output: X^*, ρ^*

X^0 initialized randomly, $i \leftarrow 0$

repeat

$q \leftarrow (i \mod K) + 1$

1) The network $\mathcal{G}$ is pruned to obtain a tree $\mathcal{T}^i(\mathcal{G}, q)$

2) Node q receives $L^i_{n\to q}$ as in (48) and $\widehat{\mathbf{y}}^i_{n\to q}(t)$, $\widehat{\mathbf{v}}^i_{n\to q}(t)$ computed as in (41) for $t = iN+1, \ldots, iN+N$ from all $n \in \mathcal{N}_q$

3) Node q estimates $R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q}, R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}$ based on the stacking defined in (42)

4) ρ^i is updated using (44)

5) $\widetilde{X}^{i+1}_q \leftarrow \text{GEVC}_Q(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}, K^i_q)$, where $K^i_q = C^{iT}_q C^i_q$ from (46)

6) $\widetilde{X}^{i+1}_q \leftarrow \widetilde{X}^{i+1}_q U^{i+1}$, where $U^{i+1} = \text{argmin}_{U\in\mathcal{D}} ||X^{i+1}_q U - X^i_q||_F$ and $\mathcal{D}$ is the set of $Q \times Q$ signature matrices

7) Partition $\widetilde{X}^{i+1}_q$ as in (49), disseminate G^{i+1}_n in $\mathcal{B}_{nq}, \forall n \in \mathcal{N}_q$

8) Every node updates X^{i+1}_k according to (50)

$i \leftarrow i+1$

until *Convergence*

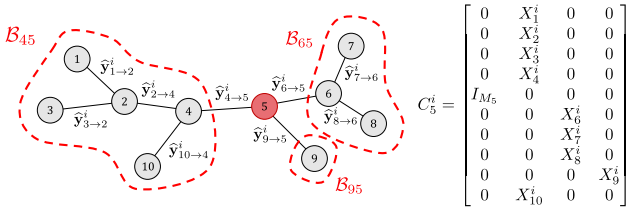

Fig. 3: Example of a tree network where the updating node is node 5, showing also $\mathcal{B}_{45}$, $\mathcal{B}_{65}$, $\mathcal{B}_{95}$ and C^i_5.

B. Topology-Independent Distributed Trace Ratio Algorithm

The main ideas of Algorithm 2 described in Section III-C remain applicable in this context, the main difference being the data flow, explained previously, and the change in definitions of variables it leads to.

Consider the tree graph $\mathcal{T}^i(\mathcal{G}, q)$ used in iteration i. As in the fully-connected case, the projection of $\mathbf{y}$ and $\mathbf{v}$ onto X^i can be written as a sum of the per-node projections as given in (45) and ρ^i can be computed as in (44). Therefore, stacking the observations in $\widetilde{\mathbf{y}}^i_q$ as in (42), we can define an $M \times \widetilde{M}_q$ matrix C^i_q, with $\widetilde{M}_q = M_q + |\mathcal{N}_q| \cdot Q$, so that $\widetilde{\mathbf{y}}^i_q = C^{iT}_q \mathbf{y}$, where C^i_q is given by:

$$C^i_q = \left[A^T_q, B^i_{-q}\right], \tag{46}$$

with A_q as in (23). Figure 3 contains an example of such a matrix C^i_q for node 5 in the tree graph depicted in the same figure. B^i_{-q} is a block matrix containing X^i_k's $k \neq q$. Each block is therefore of the size of the corresponding X^i_k and we denote by $B^i_{-q}(k, c)$ the block of B^i_{-q} in the $k-$th "block-row" and the $c-$th "block-column". There is one block-column per neighbor $n \in \mathcal{N}_q$, and we denote by c_n the block-column belonging to neighbor n. Then, we have:

$$B^i_{-q}(k, c_n) = \begin{cases} X^i_k & \text{if } k \in \mathcal{B}_{nq} \\ 0 & \text{otherwise} \end{cases}. \tag{47}$$

Node q solves the GEVD problem similar to (32) with $K^i_q = C^{iT}_q C^i_q$ from (46), and $\widetilde{\mathbf{y}}^i_q$, $\widetilde{\mathbf{v}}^i_q$ have been obtained as in (42). We note that this implies matrices $L^i_k = X^{iT}_k X^i_k$ to be shared across the network in a similar fashion as the compressed observations $\widehat{\mathbf{y}}^i_k$ and $\widehat{\mathbf{v}}^i_k$, which results in node q receiving:

$$L^i_{n\to q} = X^{iT}_n X^i_n + \sum_{k\in\mathcal{N}_n\setminus\{q\}} L^i_{k\to n} = \sum_{k\in\mathcal{B}_{nq}} X^{iT}_k X^i_k, \tag{48}$$

from all $n \in \mathcal{N}_q$. The resulting solution $\widetilde{X}^{i+1}_q$ is given in the following form:

$$\widetilde{X}^{i+1}_q = \left[X^{(i+1)T}_q, G^{(i+1)T}_{n_1}, \ldots, G^{(i+1)T}_{n_{|\mathcal{N}_k|}}\right]^T, \tag{49}$$

with $n_k \in \mathcal{N}_q$. Finally, G^{i+1}_n is disseminated in the full branch $\mathcal{B}_{nq}$ of the network and the updating rule in a tree topology is given as:

$$X^{i+1}_k = \begin{cases} X^{i+1}_q & \text{if } k = q \\ X^i_k G^{i+1}_n & \text{if } k \neq q, \text{ with } k \in \mathcal{B}_{nq} \end{cases}. \tag{50}$$

This reflects the fact that node q is not explicitly aware of the nodes beyond its neighbors and the contributions of these nodes are present only in the form of the summed observations $\widehat{\mathbf{y}}^i_{n\to q}$ received from $n \in \mathcal{N}_q$. Therefore, we expect slower convergence compared to the fully-connected case because of less degrees of freedom. This can be seen from the expression of the projection of $\mathbf{y}$ onto the updated vector X^{i+1}:

$$X^{(i+1)T}\mathbf{y} = X^{(i+1)T}_q \mathbf{y}_q + \sum_{n\in\mathcal{N}_q} G^{(i+1)T}_n \sum_{k\in\mathcal{B}_{nq}} X^{iT}_k \mathbf{y}_k, \tag{51}$$

whereas in the fully-connected case, we had:

$$X^{(i+1)T}\mathbf{y} = X^{(i+1)T}_q \mathbf{y}_q + \sum_{k\in\mathcal{K}\setminus\{q\}} G^{(i+1)T}_k X^{iT}_k \mathbf{y}_k, \tag{52}$$

where the parameters that can be chosen by node q are X_q and the G_k's or G_n's. In (52), there are $K-1$ of such G_k's whereas in (51), there are only $|\mathcal{N}_q|$.

Algorithm 3 summarizes the steps of the Topology-Independent Distributed Trace Ratio Algorithm (TI-DTRO). These modifications brought to the FC-DTRO algorithm to obtain Algorithm 3 still allow for convergence, as stated in the following theorem.

Theorem 4. *Consider a connected network represented by the graph $\mathcal{G}$. For any initialization $X^0 \in \mathbb{R}^{M\times Q}$, the updates of Algorithm 3 satisfy $\lim_{i\to+\infty} X^i \overset{*}{=} X^*$ where X^* is a solution of Algorithm 1, i.e., Algorithm 3 converges to a solution of the TRO problem (10).*

We do not provide a formal proof because of its similarity with the proof of Theorem 2 in Appendix A, where the main difference is the definition and size of variables depending on the network topology, which can in this case also change between iterations. While this substantially complicates the notation, the main strategy to prove convergence remains exactly the same.

On the other hand, an analysis analogous to the one in Section III-E is harder in this case. Even though we observe in general a convergence of the TI-DTRO algorithm when using the updating rule (35) instead of (50) in simulations, the sequence $\{\rho^i\}_i$ is not monotonic. As shown in Appendix A, the monotonic increase of the objective is an essential part of the proof of convergence hence we omit more details on the updating scheme reducing the communication overhead for the TI-DTRO Algorithm.

V. Communication and Computational Aspects

A. Communication Costs

Based on the description of the DTRO algorithms in Sections III and IV and as summarized in Figure 1, the data and parameters that are being communicated between two neighboring nodes at iteration i are the N samples of $Q-$dimensional compressed signals $\widehat{\mathbf{y}}_k^i$ and $\widehat{\mathbf{v}}_k^i$ and the $Q\times Q$ matrices L_k^i and G_k^i, which implies the communication cost is in $\mathcal{O}(NQ+2Q^2)$ per transmission link and per iteration, where $\mathcal{O}$ denotes the Landau notation. This result is the same for any given link in the network, whether the latter is fully-connected or not. In general $N \gg Q$, which makes the first term NQ the dominant one. In comparison, in a centralized setting the N samples of uncompressed signals of dimension M_k, are being sent from each node k towards the fusion center leading to a communication cost of $\mathcal{O}(NM_k)$ for node k where typically $M_k > Q$. The compression ratio at node k is therefore roughly M_k/Q.

The updating scheme which reduces the communication overhead discussed in Section III-E also allows to decrease the number of parameters being sent from the updating node to the rest of the network. Instead of sending Q^2 real numbers, using the updating rule (35) only requires communicating Q binary numbers, whereas the updating rule (36) allows to avoid the transmission entirely. However, these schemes might not reduce significantly the communication cost in practice since $N \gg Q$. Nevertheless, it reduces the overhead of disseminating extra parameters through the network.

B. Computational Complexity

At each iteration i of Algorithms 2 and 3, the updating node q needs to compute the two sample covariance matrices $R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q}$ and $R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}$ which requires around $\widetilde{M}_q^2 N$ operations each, while the GEVC computation has a complexity of $\mathcal{O}(\widetilde{M}_q^3)$. Resolving the sign ambiguity can be done by comparing the norm of two M_q-dimensional vectors, which is done Q times, once for each column of X_q^i, and therefore requires around M_qQ operations, which, considering a large enough sample size N is negligible compared to the two previous computations. Node q's computations therefore have a complexity of $\mathcal{O}(\widetilde{M}_q^2(N+\widetilde{M}_q))$. On the other hand, at non-updating nodes $k\neq q$, $2M_kQN$ and M_kQ^2 operations are required to compress the local signals and to compute the L_k^i's respectively. These nodes then update the estimation of their local filters, which is done in $\mathcal{O}(M_kQ^2)$ operations. The computations at nodes $k\neq q$ have therefore a complexity of $\mathcal{O}(M_kQN)$, again considering a large N. In comparison, the adaptive (sliding window) version of the centralized TRO algorithm (Algorithm 1) would have a complexity of $\mathcal{O}(M^2(N+M))$ per window. For example, in a fully-connected network with $K=30$ nodes, $Q=1$, $N=1000$ and $M_k=15\ \forall k$, the centralized adaptive algorithm requires $\sim 294\times10^6$ operations per window of N samples (assuming a single iteration per window), whereas the distributed algorithm requires $\sim 2.02\times10^6$ operations per window.

It is important to note that in the case of non-fully-connected networks, the $\widetilde{M}_k$'s depend on the number neighbors a node has and not the total number of nodes in the network. Therefore, the non-fully-connected topologies scale better compared to fully-connected ones. To follow-up on the previous example, if the average number of neighbors per node is equal to 3, the distributed algorithm would require only $\sim 0.33\times10^6$ operations per window of N samples. However, as will be demonstrated in the next section, the DTRO algorithm converges the fastest in fully-connected networks in general, among every topology considered in this paper. Hence in practice, there is a tradeoff between computational complexity and convergence speed.

VI. Simulations and Discussions

In this section, we provide simulation results of the DTRO algorithms we have presented previously in various experimental settings[3].

A. Experiment Settings

We consider that each node has an equal number of sensors, therefore $M_k = M/K$ for any k. The data model we consider for $\mathbf{y}$ and $\mathbf{v}$ is the following:

$$\mathbf{y}(t) = \Gamma_d\cdot\mathbf{d}(t) + \mathbf{v}(t), \tag{53}$$

where $\mathbf{d}$ is a set of desired source signals and $\mathbf{v}$ is a combination of spatially correlated noise and white noise:

$$\mathbf{v}(t) = \Gamma_s\cdot\mathbf{s}(t) + \mathbf{n}(t), \tag{54}$$

and with $\Gamma_d\in\mathbb{R}^{M\times L_d}$ and $\Gamma_s\in\mathbb{R}^{M\times L_s}$ denoting the steering or mixture matrices. Therefore, the model is a mixture of L_d+L_s point sources of which L_s are interfering, represented by $\mathbf{s}\in\mathbb{R}^{L_s}$, and continuously active, whereas the L_d desired sources, represented by $\mathbf{d}\in\mathbb{R}^{L_d}$ have an on-off behavior. When the latter are "off", the noise signal $\mathbf{v}$ can be observed. The observations of $\mathbf{s},\mathbf{d}$ independently follow a zero-mean normal distribution with variance 0.5, i.e., $\mathcal{N}(0,0.5)$. On the other hand, the elements of $\mathbf{n}$ independently follow $\mathcal{N}(0,0.1)$. In all the simulations, we consider that $10-L_s=L_d=Q$ and that the number of samples used is $N=10000$ per iteration i.

All results shown are averaged over 200 independent Monte-Carlo runs (MCRs), i.e., 200 different realizations of (53)-(54). In each MCR, all elements of Γ_s,Γ_d are drawn independently at random from $\mathcal{U}([-0.5,0.5])$, where $\mathcal{U}$ is the uniform distribution. The theoretical optima X^* have been estimated using the centralized TRO algorithm (Algorithm 1) using $\rho^{i+1}-\rho^i<10^{-12}$ as a stopping criterion. In the following discussions, a randomly generated tree where each node has between 0 and 3 children (with 1.7 children on average) is referred to as a "random tree". Moreover, the order of the updating nodes is a random permutation over the nodes, and the pruning function $\mathcal{T}^i(\cdot,q)$ we used is the shortest path computation. In these simulations, the matrices G_k are shared between the nodes, i.e., the full updating scheme (33) and (50) is used for the fully-connected and topology-independent cases respectively. Numerical values of main parameters are summarized in Table I for the various simulation settings presented below.

[3]An open-source implementation of the algorithms can be found on https://github.com/CemMusluoglu/DTROcodes.

TABLE I: Summary of parameters used in the simulations.

Experiment	Varying Q	Varying K	Varying Topology	Convergence Rate Analysis
Q	$\{1, 3, 5, 7\}$	3	5	5
K	30	$\{10, 30, 50\}$	31	30
M, M_k	$M = 450$, $M_k = M/K$, $\forall k \in \mathcal{K}$			
N	10000			
MCRs	200			

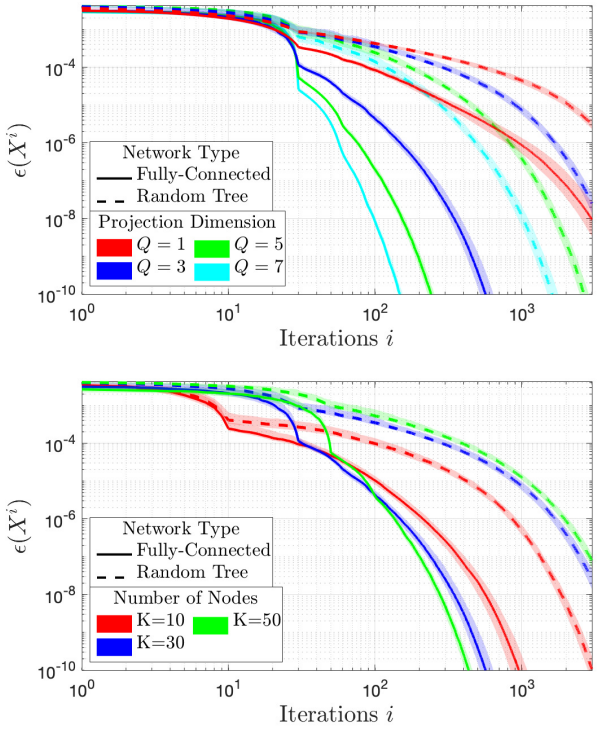

Fig. 4: Convergence in MSE of DTRO methods. The plots corresponding to fully-connected networks are represented in solid line whereas dashed lines represent results for random trees. *Top*: Varying value of Q for a fixed number of nodes $K = 30$. *Bottom*: Varying value of K for a fixed latent dimension $Q = 3$.

B. Results

We assess the convergence based on the mean squared error (MSE) between X^* and the result from the DTRO algorithms defined as:

$$\epsilon(X^i) = \min_{U \in \mathcal{D}} \frac{1}{MQ} ||X^i U - X^*||_F^2. \tag{55}$$

Note that in (55) we also implicitly resolve the sign ambiguity by choosing the sign of each column X^i such that the MSE is minimal.

In the top figure of Figure 4 we can see a comparison between fully-connected networks and randomly generated trees for a varying Q, while the number of nodes K is fixed to 30 and $M_k = 15$. On the other hand, the bottom figure shows the effects of the number of nodes K on convergence, while Q is fixed to 3 and we fix the total number of sensors to be $M = 450$. For both figures and the ones that will follow, lines in bold represent median values across Monte Carlo runs and shaded areas delimit quartiles. From these figures, we observe that, in general, fully-connected networks converge faster than their tree network counterpart in the same setting. A higher projection subspace dimension Q also leads to faster convergence. Both these behaviors are expected because they translate to more information being shared between nodes and more degrees of freedom during the updates.

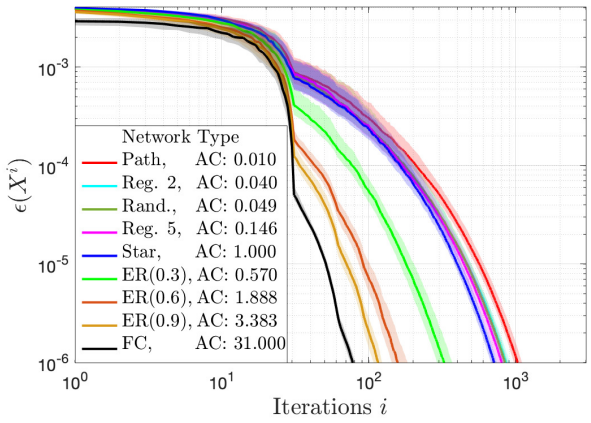

Fig. 5: Convergence comparison for various network topologies along with their algebraic connectivity (AC). In the legend, the following codewords are used for the graph type. *Path*: Path shaped tree. *Reg. 2 and 5*: Regular tree, i.e., each parent has equal number of children, the number being equal to 2 and 5 respectively. *Rand*: Random tree. *Star*: Star shaped tree. *ER(p)*: Random graph generated using the Erdős-Rényi model, with connection probability p. *FC*: Fully-connected graph.

Additionally, we expect smaller networks to converge faster because the number of iterations between two consecutive instances where a certain node is the updating one is smaller. This is indeed observed in the tree topology networks, and in the initial updating round in the case of fully-connected networks. However, in the fully-connected case, larger networks start to show faster convergence than the smaller networks after all nodes have updated at least once (i.e., once all nodes were able to move away from their initialization points). The explanation here is that the number of signals a node receives in the fully-connected case scales linearly with the number of nodes K. Therefore, the degrees of freedom in each update step also scale linearly with K (this is reflected in the size of the compressed covariance matrices and the number of G_k matrices at the updating node). For a fixed value of M and Q, this leads to larger downwards steps in the MSE function in each iteration. In the tree networks, the number of degrees of freedom at each node is determined by the number of neighbors and not by the total size of the network, which is why they do not show this behavior.

In Figure 5, we compare the convergence of the DTRO algorithms for various network topologies under the settings $K = 31$, $Q = 5$ and $M_k = 15$. We see that, as in the previous cases, the fully-connected network topology is the one converging to the optimal value X^* the fastest. It is then followed by connected random networks generated following the Erdős-Rényi (ER) model, generated using [46]. The remaining plots correspond to various tree topologies, including star, path, regular tree (i.e., each node has a fixed and equal number of children nodes) and random tree topologies. These latter are observed to have similar convergence properties between each other. However, an interesting observation is that, in most cases, the larger the algebraic connectivity (AC) of the graph corresponding to the network, the faster the convergence. Since the AC of a graph measures how connected a graph is, where a large AC means a well connected graph, we can deduce that the DTRO method converges faster in more connected networks. Note that this is a rule of thumb, which is not always perfectly satisfied (e.g., when comparing the $ER(0.3)$ and the star topology convergence in Figure 5).

Finally, we look at asymptotic convergence properties of

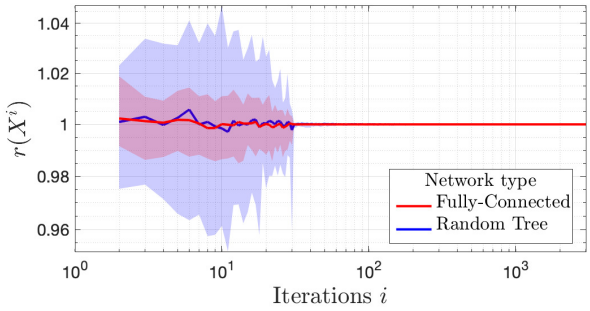

Fig. 6: DTRO algorithms' asymptotic convergence rate in fully-connected and random tree topologies.

the DTRO algorithms in fully-connected and random tree topologies ($K = 30$, $Q = 5$, $M_k = 15$). We measure the asymptotic convergence rate using:

$$r(X^i) = \frac{||X^{i+1} - X^*||_F^2}{||X^i - X^*||_F^2}. \tag{56}$$

Figure 6 shows that $\lim_{i\to+\infty} r(X^i) = 1$ which implies a sublinear asymptotic convergence rate.

In all previous figures, we observe breaking points in the convergence speed of the algorithms. It can be verified that this abrupt change happens at iteration K, i.e., the number of nodes. This is because at K iterations, the algorithm has made its first full round update. Since the algorithms are initialized randomly and by the lack of full freedom for a node q to update the local variables of other nodes, we observe a high convergence speed towards the end of the first full round update.

VII. Conclusion and Future Work

In this paper, we have proposed distributed algorithms for solving the trace ratio optimization problem in a distributed WSN setting, by solving local GEVD problems at the nodes. After analyzing thoroughly the fully-connected case, we have adapted the FC-DTRO algorithm to any connected random graph to obtain the TI-DTRO algorithm. Convergence guarantees of both algorithms have been stated and proven and extensive simulation results demonstrate the convergence properties of the algorithms in various settings. We have analyzed the communication and processing cost of both the centralized and the distributed TRO algorithms, and concluded that the distributed algorithm has a significantly smaller communication and processing burden, at the cost of slower convergence (adaptation). Furthermore, we have pointed out that the processing burden at each node scales linearly with the number of neighbors per node.

As a next step, it can be thought of exploiting some structure in the graph topology for faster convergence. For example, as in [42], the FC-DTRO can be implemented in fully-connected cliques of a graph, while the rest of the network uses TI-DTRO. On the other hand, we have considered sequential updating schemes in this paper, where only one node solves its local optimization problem at any given time. An interesting future work would be to consider asynchronous updates, where more than one node can apply the steps of FC/TI-DTRO at any iteration. In that case, an immediate problem is that the resulting local variables are not optimal, since other nodes would have already modified their local compression matrix in the meantime. However, if convergence can be achieved, it is expected to be significantly faster than the sequential setting.

Appendix A
Proof of Theorem 2

As mentioned in Section III-D, there are two main results to establish before concluding convergence. The first important objective for showing convergence of Algorithm 2 is to show that the sequence $\{\rho^i\}_{i>0}$ is non-decreasing.

Lemma 1. *Under the updates of Algorithm 2, the sequence $\{\rho^i\}_{i>0}$ is monotonic non-decreasing and satisfies $\lim_{i\to+\infty} \rho^i = \rho^*$.*

Proof. Let us denote by q the updating node at iteration i and we define $\widetilde{X}_q^i = [X_q^{iT}, I_Q, \ldots, I_Q]^T$. Then, at the end of iteration $i > 0$, we have:

$$h(X^i, \rho^i) \triangleq \text{tr}\left(X^{iT}(R_{\mathbf{yy}} - \rho^i R_{\mathbf{vv}})X^i\right) \tag{57}$$

$$= \text{tr}\left(\widetilde{X}_q^{iT}(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q})\widetilde{X}_q^i\right) \tag{58}$$

$$= 0, \tag{59}$$

where the second equation follows from (24) and $X^i = C_q^i \widetilde{X}_q^i$ (see (25)). The last equation is obtained by observing that $\rho^i = \varrho(X^i) = \varrho(C_q^i \widetilde{X}_q^i) = \tilde{\varrho}_q^i(\widetilde{X}_q^i)$ where the latter is defined in (20), which after plugging it in (58) yields (59). On the other hand, using (24)-(25), we can also write:

$$h(X^{i+1}, \rho^i) \triangleq \text{tr}\left(X^{(i+1)T}(R_{\mathbf{yy}} - \rho^i R_{\mathbf{vv}})X^{i+1}\right) \tag{60}$$

$$= \text{tr}\left(\widetilde{X}_q^{(i+1)T}(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q})\widetilde{X}_q^{i+1}\right). \tag{61}$$

Since $\widetilde{X}_q^{(i+1)T}$ maximizes $\text{tr}\left(\widetilde{X}_q^T(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q})\widetilde{X}_q\right)$ under the constraint $\widetilde{X}_q^T K_q^i \widetilde{X}_q = I_Q$ (or equivalently $\widetilde{X}_q \in \widetilde{\mathcal{S}}_q^i$) as given in (30)-(32), we have:

$$h(X^{i+1}, \rho^i) \geq \text{tr}\left(\widetilde{X}_q^T(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q})\widetilde{X}_q\right), \quad \forall \widetilde{X}_q \in \widetilde{\mathcal{S}}_q^i. \tag{62}$$

In particular, taking $\widetilde{X}_q = [X_q^{iT}, I_Q, \ldots, I_Q]^T$, we have $C_q^i \widetilde{X}_q = X^i$. Therefore, the algorithm updates satisfy $X^{iT}X^i = I_Q$ for any i and since $K_q^i \triangleq C_q^{iT} C_q^i$, it follows that this choice for $\widetilde{X}_q$ satisfies the constraint in (62). Plugging this choice in (62), we obtain:

$$h(X^{i+1}, \rho^i) \geq \text{tr}\left(X^{iT}(R_{\mathbf{yy}} - \rho^i R_{\mathbf{vv}})X^i\right) \tag{63}$$

$$= h(X^i, \rho^i) = 0, \tag{64}$$

where the last equation follows from (59). Combining (61) with (63)-(64) results in:

$$h(X^{i+1}, \rho^i) = \text{tr}\left(\widetilde{X}_q^{(i+1)T}(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q})\widetilde{X}_q^{i+1}\right) \geq 0, \tag{65}$$

which finally results in:

$$\frac{\text{tr}\left(\widetilde{X}_q^{(i+1)T} R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q}\widetilde{X}_q^{i+1}\right)}{\text{tr}\left(\widetilde{X}_q^{(i+1)T} R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}\widetilde{X}_q^{i+1}\right)} \geq \rho^i, \tag{66}$$

where it is noted that the denominator is strictly larger than zero due to the fact that $R_{\mathbf{vv}}$ (and consequently any $R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}$) is positive definite. Note that the left-hand side of (66) is equal to ρ^{i+1} by definition, which implies that $\rho^{i+1} \geq \rho^i$, which proves that the sequence $\{\rho^i\}_{i>0}$ is monotonic non-decreasing. By monotonicity, and since the sequence $\{\rho^i\}_{i>0}$ has an upper bound ρ^*, the sequence is Cauchy, which means:

$$\lim_{i\to+\infty} (\rho^{i+1} - \rho^i) = 0. \tag{67}$$

Therefore, $\forall \varepsilon > 0$, $\exists J_\varepsilon \in \mathbb{N}$: $i > J_\varepsilon \Rightarrow \rho^{i+1} - \rho^i < \varepsilon$. Using the definition of $\rho^{i+1} = \varrho(X^{i+1})$, we write:

$$\frac{\text{tr}(X^{(i+1)T} R_{\mathbf{yy}} X^{i+1})}{\text{tr}(X^{(i+1)T} R_{\mathbf{vv}} X^{i+1})} - \rho^i < \varepsilon. \tag{68}$$

Reordering the terms, this results in:

$$\underbrace{\text{tr}\left(X^{(i+1)T}(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^{i+1}\right)}_{h(X^{i+1},\rho^i)} < \varepsilon \cdot \text{tr}\left(X^{(i+1)T}R_{\mathbf{vv}}X^{i+1}\right), \tag{69}$$

and by definition, $h(X^{i+1},\rho^i) = \max_{X\in\mathcal{S}} h(X,\rho^i)$, we finally have:

$$\max_{X\in\mathcal{S}} h(X,\rho^i) < \varepsilon \cdot \text{tr}\left(X^{(i+1)T}R_{\mathbf{vv}}X^{i+1}\right). \tag{70}$$

Since ε can be made arbitrarily small for sufficiently large i, since $\text{tr}\left(X^{(i+1)T}R_{\mathbf{vv}}X^{i+1}\right)$ is bounded and positive, and by continuity of f [39], [47]:

$$\lim_{i\to+\infty}\max_{X\in\mathcal{S}} h(X,\rho^i) = 0, \tag{71}$$

and from (81)-(82), we conclude that:

$$\lim_{i\to+\infty}\rho^i = \rho^*. \qquad \square$$

The second result describes the optimality of the equilibrium points of Algorithm 2.

Lemma 2. *Let $\mathcal{X}^*$ denote the set of all equilibrium points of Algorithm 2. Then, any $X\in\mathcal{X}^*$ is a solution of the TRO problem (10).*

Proof. We assume that at iteration i, we have achieved equilibrium, i.e., $X^{i+1}=X^i$, and the updating node is q. By the updating rule given in (33), we have $X_k^{i+1} = X_k^i G_k^{i+1}\ \forall k\in\mathcal{K}\setminus\{q\}$, but the equilibrium condition leads to $G_k^{i+1}=I_Q$ $\forall k\neq q$ such that $\widetilde{X}_q^{i+1} = \left[X_q^{iT}, I_Q,\dots,I_Q\right]^T$ in (26). From (32), we have:

$$\left(R^i_{\widetilde{\mathbf{y}}_q\widetilde{\mathbf{y}}_q} - \rho^i R^i_{\widetilde{\mathbf{v}}_q\widetilde{\mathbf{v}}_q}\right)\widetilde{X}_q^{i+1} = C_q^{iT}C_q^i\widetilde{X}_q^{i+1}\widetilde{\Lambda}_q^{i+1}. \tag{72}$$

After equilibrium, the GEVLs do not change over iterations, therefore $\widetilde{\Lambda}_q^{i+1}=\widetilde{\Lambda}_q^i$. Using (24)-(25) and replacing $\widetilde{X}_q^{i+1}$ with $\widetilde{X}_q^i$ (equilibrium assumption), we obtain:

$$C_q^{iT}\left(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}}\right)X^i = C_q^{iT}X^i\widetilde{\Lambda}_q^i. \tag{73}$$

The first M_q rows of C_q^{iT} correspond to A_q as given in (23). When selecting the first M_q rows of (73), we therefore obtain:

$$A_q\left(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}}\right)X^i = A_q X^i\widetilde{\Lambda}_q^i \tag{74}$$

$$= X_q^i\widetilde{\Lambda}_q^i. \tag{75}$$

At equilibrium, we can apply the same reasoning to every node such that (75) will hold for any node q. After stacking all the corresponding equations, we obtain:

$$\left(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}}\right)X^i = \begin{bmatrix} X_1^i\widetilde{\Lambda}_1^i \\ \vdots \\ X_K^i\widetilde{\Lambda}_K^i\end{bmatrix}. \tag{76}$$

Left multiplying (73) by $\left[X_q^{iT}, I_Q,\dots,I_Q\right]^T$, we obtain:

$$X^{iT}\left(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}}\right)X^i = X^{iT}X^i\widetilde{\Lambda}_q^i = \widetilde{\Lambda}_q^i, \tag{77}$$

where the last equation follows from the orthogonality constraint $X^{iT}X^i = I_Q,\ \forall i>0$ (as obtained in the proof of Lemma 1). We observe that the left-hand side of (77) does not depend on the node q, therefore $\widetilde{\Lambda}_q^i$ is equal for all nodes and so we can set $\widetilde{\Lambda}_k^i=\Lambda^i,\ \forall k\in\mathcal{K}$. We can then rewrite equation (76) as:

$$\left(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}}\right)X^i = X^i\Lambda^i. \tag{78}$$

Therefore, at equilibrium, the columns of X^i can only contain EVCs of $(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})$. Additionally, at equilibrium, $\rho^{i+1}=\rho^i$, hence:

$$\rho^{i+1} = \frac{\text{tr}(X^{(i+1)T}R_{\mathbf{yy}}X^{i+1})}{\text{tr}(X^{(i+1)T}R_{\mathbf{vv}}X^{i+1})} = \rho^i, \tag{79}$$

and reordering the equation, we have:

$$\text{tr}(X^{(i+1)T}R_{\mathbf{yy}}X^{i+1} - \rho^i X^{(i+1)T}R_{\mathbf{vv}}X^{i+1}) = 0, \tag{80}$$

which implies that $h(X^{i+1},\rho^i)=0$ from definition (5). Since X^{i+1} contains EVCs of $(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})$, it maximizes $h(\cdot,\rho^i)$, and we have:

$$\max_{X\in\mathcal{S}} h(X,\rho^i) = h(X^{i+1},\rho^i) = 0 \iff f(\rho^i)=0 \tag{81}$$

$$\iff \rho^i=\rho^*, \tag{82}$$

where the first equivalence comes from (6) and the second one from Theorem 1. Since $\rho^*=\rho^i=\varrho(X^i)$, the equilibrium point X^i must be a solution of the TRO problem (10). $\square$

Convergence of Algorithm 2 is obtained by combining Lemmas 1 and 2 and analyzing the iterations before convergence.

Proof of Theorem 2. From (32), we note that the only possible ambiguity at Step 4 of Algorithm 2 is the sign of the columns of $\widetilde{X}_q^{i+1}$. By continuity of the function ϱ and the fact that the sequence of objectives $\{\rho^i\}_{i>0}$ converges to ρ^*, for all $\varepsilon>0$, there exists an integer J_ε and a signature matrix $U^{i+1}\in\mathcal{D}$ such that:

$$||\widetilde{X}_q^{i+1}U^{i+1} - [X_q^{iT}, I_Q,\dots,I_Q]^T||_F < \varepsilon, \tag{83}$$

and in particular:

$$||X_q^{i+1}U^{i+1} - X_q^i||_F < \varepsilon, \tag{84}$$

when $i>J_\varepsilon$ for the updating node q. We determine these matrices U^{i+1} at Step 5 of Algorithm 2 by minimizing the left-hand side of the previous equation and apply it to the full local variable $\widetilde{X}_q^{i+1}$, Then, knowing that $X^{i+1}=C_q^i\widetilde{X}_q^{i+1}$ and $X^i = C_q^i\cdot[X_q^{iT}, I_Q,\dots,I_Q]^T$, we have:

$$\lim_{i\to+\infty}||X^{i+1}-X^i||_F = 0. \tag{85}$$

Let us return now to (73), which followed from (72) under the equilibrium assumption $X^{i+1}=X^i$. In the case where equilibrium is not reached, a node-specific (unknown) correction term Δ_q^i should be added in (73) in order for the equality to hold:

$$C_q^{iT}(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i + \Delta_q^i = C_q^{iT}X^i\widetilde{\Lambda}_q^i. \tag{86}$$

Then, following a similar development as in the proof of Lemma 2, we select the first M_q rows of each side of the previous equation, and apply the same reasoning to every node, to obtain the stacked equations:

$$(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i + \Delta^i = \begin{bmatrix} X_1^i\widetilde{\Lambda}_1^i \\ \vdots \\ X_K^i\widetilde{\Lambda}_K^i\end{bmatrix}, \tag{87}$$

where Δ^i is an error term containing all the Δ_q^i's. Similarly, there exists an error term E_q^i in the equivalent of (77) in the non-equilibrium case:

$$X^{iT}(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i + E_q^i = \widetilde{\Lambda}_q^i. \tag{88}$$

However, from (85), the difference between X^i's tend to zero therefore the error terms vanish, such that (88) leads to:

$$\lim_{i\to+\infty}||X^{iT}(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i - \widetilde{\Lambda}_k^i||_F = 0,\ \forall k\in\mathcal{K}. \tag{89}$$

This means all $\widetilde{\Lambda}_k^i,\ \forall k\in\mathcal{K}$, all coincide in the limit to the same diagonal matrix, which is equal to $\Lambda^i = X^{iT}(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i$. Therefore, and since also the error term in (87) will vanish when $i\to+\infty$, we find:

$$\lim_{i\to+\infty}||(R_{\mathbf{yy}}-\rho^i R_{\mathbf{vv}})X^i - X^i\Lambda^i||_F = 0. \tag{90}$$

Since Λ^i is diagonal, and using the result of Lemma 1, it follows that $\{X^i\}_{i>0}$ converges to EVCs of $R_{\mathbf{yy}}-\rho^* R_{\mathbf{vv}}$. Moreover, (85) shows that the columns of X^i cannot switch as i gets large. Hence, the sequence $\{X^i\}_{i>0}$ satisfies $\lim_{i\to+\infty}X^i \overset{*}{=}$

X^*, where X^* contains Q principal EVCs of $R_{\mathbf{yy}} - \rho^* R_{\mathbf{vv}}$ and maximizes ϱ, which concludes the proof. □

Appendix B
Proofs of Theorem 3 and Corollary 1

Proof of Theorem 3. In this proof, variables with an underline correspond to the ones obtained by using the updating rule (35) in Algorithm 2 instead of (33). We assume that both algorithms receive the same stream of data for $\mathbf{y}$ and $\mathbf{v}$ at each iteration. Consider that ρ is fixed for now. Then, at iteration i, the updating node q finds $\widetilde{X}_q^{i+1}$, which solves (32):

$$C_q^{iT}\left(R_{\mathbf{yy}} - \rho R_{\mathbf{vv}}\right) C_q^i \widetilde{X}_q^{i+1} = K_q^i \widetilde{X}_q^{i+1} \widetilde{\Lambda}_q^{i+1}, \tag{91}$$

and in the case of the update rule (35), we can write:

$$\underline{C}_q^{iT}\left(R_{\mathbf{yy}} - \rho R_{\mathbf{vv}}\right) \underline{C}_q^i \underline{\widetilde{X}}_q^{i+1} = \underline{K}_q^i \underline{\widetilde{X}}_q^{i+1} \underline{\widetilde{\Lambda}}_q^{i+1}, \tag{92}$$

the analogous equation.

We will now prove the theorem by induction. Suppose that we are in iteration i in which node q is the updating node and assume that there exist iterations $i(k)$ $\forall k \neq q$ such that the property P1) of the theorem holds, i.e.:

$$\begin{aligned} &X_1^{i(1)+1} \overset{*}{=} \underline{X}_1^{i(1)+1}, \ldots, X_{q-1}^{i(q-1)+1} \overset{*}{=} \underline{X}_{q-1}^{i(q-1)+1}, \\ &X_{q+1}^{i(q+1)+1} \overset{*}{=} \underline{X}_{q+1}^{i(q+1)+1}, \ldots, X_K^{i(K)+1} \overset{*}{=} \underline{X}_K^{i(K)+1}. \end{aligned} \tag{93}$$

Therefore, there exist matrices $D_1^{i(1)}, \ldots, D_K^{i(K)} \in \mathbb{R}^{Q\times Q}$ such that:

$$\begin{aligned} &X_1^{i(1)+1} = \underline{X}_1^{i(1)+1} D_1^{i(1)}, \ldots, X_{q-1}^{i(q-1)+1} = \underline{X}_{q-1}^{i(q-1)+1} D_{q-1}^{i(q-1)}, \\ &X_{q+1}^{i(q+1)+1} = \underline{X}_{q+1}^{i(q+1)+1} D_{q+1}^{i(q+1)}, \ldots, X_K^{i(K)+1} = \underline{X}_K^{i(K)+1} D_K^{i(K)}. \end{aligned} \tag{94}$$

Then, for $i(k)+1 > 0$, let us take $i(k)$ as the last iteration at which the updating node was the node k before iteration i, for every node $k \neq q$. As seen in update equation (33), at each subsequent iteration, node k's local variable gets multiplied on the right with a new matrix G_k provided by the main updating node q. The cumulative factor up to the start of iteration i is:

$$\mathbf{G}_k^i = \prod_{j=i(k)+1}^{i-1} G_k^{j+1}, \tag{95}$$

where G_k^{j+1} is obtained from the partitioning (26) of $\widetilde{X}_{q(j)}^{j+1}$ at iteration j, where $q(j)$ is the node updating at iteration j. From (33) and (95), this results in:

$$X_k^i = X_k^{i(k)+1} \mathbf{G}_k^i. \tag{96}$$

Similarly, for the updating scheme (35), we have:

$$\mathbf{D}_k^i = \prod_{j=i(k)+1}^{i-1} D_k^{j+1} \tag{97}$$

$$\underline{X}_k^i = \underline{X}_k^{i(k)+1} \mathbf{D}_k^i. \tag{98}$$

Using (96), (98) and (94), for any $k \in \mathcal{K}\backslash\{q\}$, we may write:

$$\begin{aligned} X_k^i &= X_k^{i(k)+1} \mathbf{G}_k^i && (99)\\ &= \underline{X}_k^{i(k)+1} D_k^{i(k)} \mathbf{G}_k^i && (100)\\ &= \underline{X}_k^i \mathbf{D}_k^i D_k^{i(k)} \mathbf{G}_k^i, && (101) \end{aligned}$$

where the last equation is obtained by observing that any signature matrix D satisfies $D \cdot D = I$ and then right-multiplying both sides of (98) by $\mathbf{D}_k^i$. Then, denoting by $T_q \in \mathbb{R}^{\widetilde{M}_q \times \widetilde{M}_q}$ the block diagonal matrix such that:

$$\begin{aligned} T_q = \text{BlkDiag}(&I_{M_q}, \mathbf{D}_1^i D_1^{i(1)} \mathbf{G}_1^i, \ldots, \mathbf{D}_{q-1}^i D_{q-1}^{i(q-1)} \mathbf{G}_{q-1}^i, \\ &\mathbf{D}_{q+1}^i D_{q+1}^{i(q+1)} \mathbf{G}_{q+1}^i, \ldots, \mathbf{D}_K^i D_K^{i(K)} \mathbf{G}_K^i), \end{aligned} \tag{102}$$

we can see that the relationship between C_q^i and $\underline{C}_q^i$ is:

$$C_q^i = \underline{C}_q^i T_q, \tag{103}$$

and equation (91) becomes:

$$T_q^T \underline{C}_q^{iT} \left(R_{\mathbf{yy}} - \rho R_{\mathbf{vv}}\right) \underline{C}_q^i T_q \widetilde{X}_q^{i+1} = T_q^T \underline{K}_q^i T_q \widetilde{X}_q^{i+1} \widetilde{\Lambda}_q^{i+1}. \tag{104}$$

The aim is now to show that T_q is invertible. From Remark 1, since K_q^i is full rank, then so is C_q^i, $\forall q, i$. Looking at the definition (22) of C_q^i, we deduce that $\forall k$, X_k^i is full rank. Since this holds for any iteration i, in particular $X_k^{i+1} = X_k^i G_k^{i+1}$ is also full rank. Then, by rank properties of matrix products, matrices G_k are all full rank. The multiplication in each block-diagonal of T_q as given in (102) by non-singular matrices do not change the rank of the resulting matrix.

Therefore, T_q is full rank, and by being square, it is also invertible. Finally, we can left-multiply (104) on both sides with T_q^{-T}, and observe that the solutions to (91) and (92) satisfy the following relationship:

$$\underline{\widetilde{X}}_q^{i+1} \overset{*}{=} T_q \widetilde{X}_q^{i+1}. \tag{105}$$

For the initial conditions, i.e., $i(k)+1 = 0$, the algorithms are both initialized with $X^0 = \underline{X}^0$. Therefore, equation (93) is satisfied with strict equality since $X_k^0 = \underline{X}_k^0$ $\forall k$, where we would have $D_k^{i(k)} = I_Q$, $\forall k$ in (94). Then, we can deduce from (103) and (105) that:

$$X^{i+1} = C_q^i \widetilde{X}_q^{i+1} \overset{*}{=} \underline{C}_q^i \underline{\widetilde{X}}_q^{i+1}. \tag{106}$$

According to Algorithm 2, in the updating scheme (33), the sequence of objectives computed is given by $\{\varrho(C_q^i \widetilde{X}_q^{i+1})\}_i$ and in the updating scheme (35) by $\{\varrho(\underline{C}_q^i \underline{\widetilde{X}}_q^{i+1})\}_i$. Since the starting point ρ^0 is the same for both methods, the full sequences of objectives are also the same. By induction, we have that equation (105) is satisfied for all iterations i. In particular, for the updating node q at iteration i, we have:

$$\begin{aligned} X_q^{i+1} &= [I_{M_q}|0] \widetilde{X}_q^{i+1} \\ &\overset{*}{=} [I_{M_q}|0] \left(T_q \underline{\widetilde{X}}_q^{i+1}\right) = [I_{M_q}|0] \underline{\widetilde{X}}_q^{i+1} = \underline{X}_q^{i+1}, \end{aligned} \tag{107}$$

since $[I_{M_q}|0] T_q = [I_{M_q}|0]$. This proves property P1).

For property P2), we first note that, by Theorem 2, $\lim_{i\to+\infty} X_k^{i+1} \overset{*}{=} X_k^*$, $\forall k$. Suppose that we take an X^i:

$$X^i = \left[(X_1^* B_1)^T, \ldots, (X_K^* B_K)^T\right]^T, \tag{108}$$

where $B_k \in \mathbb{R}^{Q\times Q}$ are signature matrices, i.e., they allow to change the signs of the columns of X_k^*'s, such that $X_k^i \overset{*}{=} X_k^*$, $\forall k$ but possibly $X^i \overset{*}{\neq} X^*$. This implies that convergence has already been achieved locally but the stacked matrix X^i is not a solution of (10) due to the sign mismatches in the columns across the different X_k's. It can be deduced from Theorem 2 that, if we were to update using (33), we would obtain:

$$X^{i+1} \overset{*}{=} X^*. \tag{109}$$

Then, this implies that $G_k^{i+1} \overset{*}{=} B_k$, for all non-updating nodes k. Therefore, in the case of updating with (35), we can conclude from (34) that:

$$D_k^{i+1} \overset{*}{=} B_k \Rightarrow \underline{X}^{i+1} \overset{*}{=} X^*. \tag{110}$$

Since we have a guarantee of local convergence, we see that the updating scheme (35) also satisfies:

$$\lim_{i\to+\infty} \underline{X}^i \overset{*}{=} X^*. \qquad \square$$

Proof of Corollary 1. If we do not communicate the signs by following the updating scheme (36), we can replace D_k^i by the $Q \times Q$ identity matrix $\forall k, i$ in the previous proof. We see that we would still achieve local convergence but in the general case, we would not obtain the optimal network-wide variable because of block-wise sign differences in its elements, since (110) would not be possible to achieve in that case. □

APPENDIX C
MODIFICATIONS FOR RANK-DEFICIENCY

Although from a statistical point of view this would be a rare event, we discuss in this section some possible modifications that can be applied to the methods presented previously for special cases where the assumptions of Remark 1 do not hold. Similar discussions and solutions have been presented in [48].

Suppose that there exist iterations at which the matrix $(R^i_{\tilde{\mathbf{y}}_q\tilde{\mathbf{y}}_q} - \rho^i R^i_{\tilde{\mathbf{v}}_q\tilde{\mathbf{v}}_q})$ or K^i_q is not full rank. This implies that C^i_q is not full rank which in turn means that there is at least a node k for which X^i_k has linearly dependent columns. A way to fix this problem would be to make node k replace all but one of the linearly dependent columns by random vectors.

On the other hand, suppose that the largest $Q+1$ GEVLs of the matrix pencil $(R^i_{\tilde{\mathbf{y}}_q\tilde{\mathbf{y}}_q} - \rho^i R^i_{\tilde{\mathbf{v}}_q\tilde{\mathbf{v}}_q}, K^i_q)$ are not all distinct, making the solution of (32) ill-defined. Then, one solution would be to skip node q at iteration i, while assuming the problem will not occur again. Otherwise, suppose there are $d < Q+1$ different GEVLs in the first $Q+1$ GEVLs (when ordered in decreasing order). We are looking for an $\widetilde{X}_q$ such that its range space is in $\mathcal{V}^i_q$, where $\mathcal{V}^i_q$ denotes the space spanned by all GEVCs corresponding to the d largest GEVLs, while making a minimal change from the current X^i to ensure local convergence at node q. If the basis vectors of $\mathcal{V}^i_q$ are put in the columns of the matrix $V^i_q \in \mathbb{R}^{M\times Q}$, then $\widetilde{X}^{i+1}_q = V^i_q P^{i+1}_q$, where $P^{i+1}_q \in \mathbb{R}^{Q\times Q}$ is obtained through solving:

$$P^{i+1}_q = \underset{P_q}{\operatorname{argmin}} \; ||V^i_q P_q - [X^{iT}_q, I_Q, \ldots, I_Q]^T||_F \quad \text{subject to} \quad P^T_q P_q = I_Q, \tag{111}$$

which is known as the Orthogonal Procrustes Problem [49].

REFERENCES

[1] C. A. Musluoglu and A. Bertrand, "Distributed trace ratio optimization in fully-connected sensor networks," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 1991–1995.

[2] Y.-F. Guo, S.-J. Li, J.-Y. Yang, T.-T. Shu, and L.-D. Wu, "A generalized Foley–Sammon transform based on generalized Fisher discriminant criterion and its application to face recognition," *Pattern Recognition Letters*, vol. 24, no. 1-3, pp. 147–158, 2003.

[3] H. Wang, S. Yan, D. Xu, X. Tang, and T. Huang, "Trace ratio vs. ratio trace for dimensionality reduction," in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[4] Y. Jia, F. Nie, and C. Zhang, "Trace ratio problem revisited," *IEEE Transactions on Neural Networks*, vol. 20, no. 4, pp. 729–735, 2009.

[5] S. Yan and X. Tang, "Trace quotient problems revisited," in *European Conference on Computer Vision*. Springer, 2006, pp. 232–244.

[6] F. Nie, S. Xiang, Y. Jia, C. Zhang, and S. Yan, "Trace ratio criterion for feature selection." in *AAAI*, vol. 2, 2008, pp. 671–676.

[7] C. Shen, H. Li, and M. J. Brooks, "Supervised dimensionality reduction via sequential semidefinite programming," *Pattern Recognition*, vol. 41, no. 12, pp. 3644–3652, 2008.

[8] Y. Wang, S. Gao, and X. Gao, "Common spatial pattern method for channel selelction in motor imagery based brain-computer interface," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*. IEEE, 2006, pp. 5392–5395.

[9] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[10] J. W. Sammon, "An optimal discriminant plane," *IEEE Transactions on Computers*, vol. 100, no. 9, pp. 826–829, 1970.

[11] D. H. Foley and J. W. Sammon, "An optimal set of discriminant vectors," *IEEE Transactions on Computers*, vol. 100, no. 3, pp. 281–289, 1975.

[12] K. Liu, Y.-Q. Cheng, and J.-Y. Yang, "A generalized optimal set of discriminant vectors," *Pattern Recognition*, vol. 25, no. 7, pp. 731–739, 1992.

[13] A. Bertrand, J. Callebaut, and M. Moonen, "Adaptive distributed noise reduction for speech enhancement in wireless acoustic sensor networks," in *Proc. of the International Workshop on Acoustic Echo and Noise Control (IWAENC)*, 2010.

[14] A. Bertrand and M. Moonen, "Distributed LCMV beamforming in a wireless sensor network with single-channel per-node signal transmission," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3447–3459, 2013.

[15] S. Markovich-Golan, A. Bertrand, M. Moonen, and S. Gannot, "Optimal distributed minimum-variance beamforming approaches for speech enhancement in wireless acoustic sensor networks," *Signal Processing*, vol. 107, pp. 4–20, 2015.

[16] M. Cobos, F. Antonacci, A. Mouchtaris, and B. Lee, "Wireless acoustic sensor networks and applications," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 1085290, 3 pages, 2017.

[17] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "xDAWN algorithm to enhance evoked potentials: application to brain–computer interface," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 8, pp. 2035–2043, 2009.

[18] A. Bertrand, "Distributed signal processing for wireless EEG sensor networks," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 6, pp. 923–935, 2015.

[19] A. M. Narayanan and A. Bertrand, "Analysis of miniaturization effects and channel selection strategies for EEG sensor networks with application to auditory attention detection," *IEEE Transactions on Biomedical Engineering*, vol. 67, no. 1, pp. 234–244, 2020.

[20] Z. Quan, S. Cui, and A. H. Sayed, "Optimal linear cooperation for spectrum sensing in cognitive radio networks," *IEEE Journal of selected topics in signal processing*, vol. 2, no. 1, pp. 28–40, 2008.

[21] S. Maleki, A. Pandharipande, and G. Leus, "Energy-efficient distributed spectrum sensing for cognitive sensor networks," *IEEE sensors journal*, vol. 11, no. 3, pp. 565–573, 2010.

[22] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*, 2004, pp. 20–27.

[23] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2155–2169, 2020.

[24] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed kalman filtering and smoothing," *IEEE Transactions on automatic control*, vol. 55, no. 9, pp. 2069–2084, 2010.

[25] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[26] I. F. Akyildiz, T. Melodia, and K. R. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer networks*, vol. 51, no. 4, pp. 921–960, 2007.

[27] D. Ciuonzo, P. S. Rossi, and P. Willett, "Generalized Rao test for decentralized detection of an uncooperative target," *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 678–682, 2017.

[28] J.-F. Chamberland and V. V. Veeravalli, "Wireless sensors in distributed detection applications," *IEEE signal processing magazine*, vol. 24, no. 3, pp. 16–25, 2007.

[29] X. Cheng, D. Ciuonzo, and P. S. Rossi, "Multibit decentralized detection through fusing smart and dumb sensors based on Rao test," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 2, pp. 1391–1405, 2019.

[30] T. T. Ngo, M. Bellalij, and Y. Saad, "The trace ratio optimization problem," *SIAM review*, vol. 54, no. 3, pp. 545–569, 2012.

[31] J. Ye, R. Janardan, and Q. Li, "Two-dimensional linear discriminant analysis," in *Advances in neural information processing systems*, 2005, pp. 1569–1576.

[32] A. H. Sameh and J. A. Wisniewski, "A trace minimization algorithm for the generalized eigenvalue problem," *SIAM Journal on Numerical Analysis*, vol. 19, no. 6, pp. 1243–1259, 1982.

[33] D. Cai, X. He, J. Han, and H.-J. Zhang, "Orthogonal Laplacianfaces for face recognition," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3608–3614, 2006.

[34] R. E. Prieto, "A general solution to the maximization of the multidimensional generalized Rayleigh quotient used in linear discriminant analysis for signal classification," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, vol. 6. IEEE, 2003, pp. VI–157.

[35] S. Crandall, "Iterative procedures related to relaxation methods for eigenvalue problems," *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, vol. 207, no. 1090, pp. 416–423, 1951.

[36] P. Lancaster, "A generalised Rayleigh quotient iteration for lambda-matrices," *Archive for Rational Mechanics and Analysis*, vol. 8, no. 1, p. 309, 1961.

[37] B. N. Parlett, *The symmetric eigenvalue problem*. siam, 1998, vol. 20.

[38] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2012, vol. 3.

[39] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[40] A. Bertrand and M. Moonen, "Distributed adaptive generalized eigenvector estimation of a sensor signal covariance matrix pair in a fully connected sensor network," *Signal Processing*, vol. 106, pp. 209–214, 2015.

[41] ——, "Distributed adaptive estimation of covariance matrix eigenvectors in wireless sensor networks with application to distributed PCA," *Signal Processing*, vol. 104, pp. 120–135, 2014.

[42] J. Szurley, A. Bertrand, and M. Moonen, "Distributed adaptive node-specific signal estimation in heterogeneous and mixed-topology wireless sensor networks," *Signal Processing*, vol. 117, pp. 44–60, 2015.

[43] ——, "Topology-independent distributed adaptive node-specific signal estimation in wireless sensor networks," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 1, pp. 130–144, 2016.

[44] A. Bertrand and M. Moonen, "Distributed canonical correlation analysis in wireless sensor networks with application to distributed blind source separation," *IEEE Transactions on Signal Processing*, vol. 63, no. 18, pp. 4800–4813, 2015.

[45] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[46] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints*, Aug. 2014.
[47] E. E. Tyrtyshnikov, *A brief introduction to numerical analysis*. Springer Science & Business Media, 2012.
[48] A. Hassani, A. Bertrand, and M. Moonen, "GEVD-based low-rank approximation for distributed adaptive node-specific signal estimation in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 64, no. 10, pp. 2557–2572, 2015.
[49] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966.