Distributed MAXVAR: Identifying Common Signal Components across the Nodes of a Sensor Network

Charles Hovine, Alexander Bertrand
KU Leuven, Department of Electrical Engineering (ESAT)
STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics
KU Leuven Institute for Artificial Intelligence (Leuven.AI)
Leuven, Belgium
{charles.hovine, alexander.bertrand}@esat.kuleuven.be

***Abstract*—A wireless sensor network (WSN) consists of a collection of sensor nodes, which are equipped with processing and wireless communication facilities to share data between each other. In some WSN applications, it would be relevant for each node to identify which signal components it shares with other nodes in the network. However, this is hard to realize in a distributed context, in particular between node pairs that do not share a direct wireless link. In this paper, we introduce a distributed algorithm for estimating the signal subspace that (on average) is closest to the pairwise intersections between any two of the per-node sensor signal subspaces. In order to facilitate an efficient data fusion, we assume the WSN has (or can be pruned to) a tree-topology. As opposed to a centralized algorithm where all the sensor signals are transmitted to a fusion center (FC), the per-node bandwidth and processing requirements are independent of the network-size and only depend on the number of neighbors per node and a chosen compression parameter. By construction, our algorithm converges to the solution of the so-called "maximum variance" (MAXVAR) formulation of the generalized canonical correlation anlalysis (GCCA) problem in which observations of every node act as a separate "view" in the GCCA formulation. Therefore, even though our work is formalized within a WSN context, it can be used as a generic distributed MAXVAR algorithm in other application contexts as well.**

***Index Terms*—Wireless sensor networks, distributed estimation, generalized canonical correlation analysis, MAXVAR.**

I. Introduction

In the context of Wireless Sensor Networks (WSNs), where sensor channels are spread accross different nodes communicating via wireless links, two paradigms are considered when applying array processing methods. Centralized fusion relies on collecting the network-wide observations in an FC where they are jointly processed, at the cost of large bandwidth and processing requirements at the FC. Distributed processing on the other hand, relies on the nodes collaboratively solving a task without any single node accessing the full network-wide observations. As the value of many array processing methods often depends on the presence of correlation between the channels of interest, the nodes can save bandwidth by identifying nodes whose channels correlate with their own (i.e. whose sensors observe common latent phenomena) and solve the given task by only communicating with those nodes.

To achieve this goal of identifying nodes observing common phenomena in a bottom-up fashion, we wish to estimate the *pairwise intersections* between the *per-node* signal subspaces. As this problem scales quadratically with the network's size, we instead look for the single subspace which is closest to all the pairwise intersections, thus corresponding to the *average* pairwise intersection. Using this subspace, the nodes could for example adaptively group themselves based on the degrees at which they observe each of the average intersection subspace components (which ideally correspond to different latent phenomena). It is known that canonical correlation analysis (CCA) can in fact be used for finding the intersection between two subspaces [1], and that its GCCA extensions achieve the same objective for more than two subspaces [2].

Most previous works on distributed subspace estimation aim to identify a subspace from the *union* of the per-node signal subspaces in order to optimize a certain criterion, such as maximizing variance [3], [4], maximizing SNR [5], [6], or maximizing correlation between two signal types [7], [8]. Our work differs in the sense that we are specifically targetting the *intersections* of the per-node sensor signal subspaces, which would be achieved by a distributed realization of a GCCA-type algorithm. The two main formulations of GCCA are the so-called SUMCORR and MAXVAR formulations [9]. A scalable distributed algorithm to solve the former has been proposed in [10] but, to our knowledge, there exists no distributed algorithm for solving the latter.

In this paper, we present a distributed algorithm to compute the average pairwise intersection of the per-node sensor signal subspaces in a distributed fashion, which can be shown to be equivalent to a distributed MAXVAR GCCA. The proposed algorithm can operate in tree-topology networks with a per-node communication and processing cost which is independent of the network's size.

The paper is organized as follows. The problem statement is formalized in section II. The distributed algorithm is derived in section III. Simulations supporting the algorithm's performance are presented in section IV. Finally, general conclusions are given in section V.

II. Problem statement

We consider a WSN consisting of K nodes in which each node $k \in \mathcal{K} = \{1, \ldots, K\}$ collects discrete observations of a complex-valued M_k-channel sensor signal $\boldsymbol{x}_k =$

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) and from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

$[x_{k,1},\ldots,x_{k,M_k}]^T$. We model $\boldsymbol{x}_k$ as a stochastic process and denote $\boldsymbol{x}_k[t]$ as its value at time t. Let $\boldsymbol{X}_k$ denote the $M_k \times T$ observation matrix containing T observations of $\boldsymbol{x}_k$ in its columns and $\boldsymbol{X}_k[t]$ the matrix containing $T' \ll T$ observations of $\boldsymbol{x}_k$ in a window centered around t. We assume that $\boldsymbol{x}_k$ is zero-mean, ergodic and short-time stationary, allowing us to estimate the slowly varying covariance matrices from sample averages over short segments of data:

$$\mathbb{E}\left\{\boldsymbol{x}_k[t]\boldsymbol{x}_q^H[t]\right\} = \boldsymbol{R}_{\boldsymbol{x}_k\boldsymbol{x}_q}[t] \approx \frac{1}{T'}\boldsymbol{X}_k[t]\boldsymbol{X}_q^H[t] \quad (1)$$

where $(\cdot)^H$ denotes the conjugate transpose operator. Finally, we define the network-wide observation vector as the M-channel vector $\boldsymbol{x}$ obtained by stacking the $\boldsymbol{x}_k$'s and where $M = \sum_k M_k$.

We wish to estimate the Q-dimensional signal subspace $\mathrm{Span}\left(s_1,\ldots,s_Q\right)$ that is (on average) closest to the pairwise intersections[1] of the per-node sensor signal subspaces. More formally, we are looking for an ordered set of Q basis signals $\boldsymbol{s} = [s_1,\ldots,s_Q]$ and $M_k \times Q$ projection matrices $\boldsymbol{W}_k$ such that

$$\{\boldsymbol{W}_1,\ldots,\boldsymbol{W}_K\} = \underset{\{\boldsymbol{W}_1,\ldots,\boldsymbol{W}_K\}}{\operatorname{argmin}} \min_{\boldsymbol{s}} \sum_{k=1}^{K} \mathbb{E}\left\{\left\|\boldsymbol{s} - \boldsymbol{W}_k^H\boldsymbol{x}_k\right\|^2\right\} \quad (2)$$

$$\text{s.t.} \quad \mathbb{E}\left\{\boldsymbol{s}\boldsymbol{s}^H\right\} = \boldsymbol{I}_Q. \quad (3)$$

This is known as the MAXVAR generalization of CCA [11].

To understand the relationship with the aforementioned goal of approximating pairwise intersections of the sensor signal subspaces across all node pairs, we substitute $\boldsymbol{s}$ in (2) with its optimal[2] value

$$\boldsymbol{s}_{opt} = \frac{1}{K}\sum_k \boldsymbol{W}_k^H\boldsymbol{x}_k \quad (4)$$

which results in the equivalent problem [12]

$$\{\boldsymbol{W}_1,\ldots,\boldsymbol{W}_K\} = \underset{\{\boldsymbol{W}_1,\ldots,\boldsymbol{W}_K\}}{\operatorname{argmin}} \sum_{k,l=1}^{K} \mathbb{E}\left\{\left\|\boldsymbol{W}_k^H\boldsymbol{x}_k - \boldsymbol{W}_l^H\boldsymbol{x}_l\right\|^2\right\} \quad (5)$$

$$\text{s.t.} \quad \sum_{k,l=1}^{K} \boldsymbol{W}_k^H\boldsymbol{R}_{\boldsymbol{x}_k\boldsymbol{x}_l}\boldsymbol{W}_l = K^2\boldsymbol{I}_Q. \quad (6)$$

Intuitively, this shows that the Q-dimensional signal subspace defined by $\boldsymbol{s}$ aims to capture signal components that are shared between a large part of the individual node pairs. It can therefore be viewed as a proxy for the collection of pairwise intersections between the signal subspaces of pairs of sensor nodes. Note that if two nodes are not direcly connected, evaluating the corresponding pairwise distance in (5) can be challenging.

Similarly[3] to [12], we can show that the solution to (5)-(6), $\boldsymbol{W}$, can be expressed as a generalized eigenvalue decomposition (GEVD):

$$\boldsymbol{R}_{D_{xx}}\boldsymbol{W} = \frac{1}{K^2}\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}\boldsymbol{W}\boldsymbol{\Lambda} \quad (7)$$

with $\boldsymbol{W} = [\boldsymbol{W}_1^T \cdots \boldsymbol{W}_K^T]^T$ the block matrix obtained by stacking the $\boldsymbol{W}_k$'s and $\boldsymbol{R}_{D_{xx}} = \mathrm{Blkdiag}(\boldsymbol{R}_{\boldsymbol{x}_1\boldsymbol{x}_1},\ldots,\boldsymbol{R}_{\boldsymbol{x}_K\boldsymbol{x}_K})$ the block diagonal matrix containing the node-specific correlation matrices and where $\boldsymbol{\Lambda}$ is the diagonal matrix of generalized eigenvalues (GEVL).

In order to compute the generalized eigenvectors (GEVC) in (7), all nodes would need to share their observations to an FC where the full covariance matrix could be estimated. This would require a large communication bandwidth between the nodes and the FC, in particular if all nodes are not directly connected to the FC, which increases the stress on the communication links of the nodes that are close to the fusion center. In addition, such a centralized processing does not leverage the fact that we are only interested in the Q first components of the solution.

In this paper, we present a distributed algorithm for solving (2)-(3) in networks pruned to a tree-topology and relying on the transmission of Q-dimensional compressed versions of the data from neighboring nodes, thus lifting the need to transfer the raw M_k-channel sensor observations of all nodes to an FC through a possibly multi-hop network. Instead, the signal observations are directly fused with other observations within the network, where each node will eventually have access to the estimate of $\boldsymbol{s}$, thereby avoiding the need for a fusion center altogether.

III. Distributed MAXVAR Algorithm

The derivation of the D-MAXVAR algorithm uses some ingredients from [5], where a similar GEVD problem as (7) is addressed. However, the algorithm of [5] is not directly applicable here due to the particular structure of $\boldsymbol{R}_{D_{xx}}$, which is a collection of submatrices of $\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}$ (details omitted). Furthermore, the algorithm of [5] was only defined for fully-connected WSNs, in which all nodes have a direct link with all other nodes. Here, we start from a star topology (for the sake of an easy exposition), and then generalize this to more general tree topologies.

The D-MAXVAR algorithm iteratively updates the $M \times Q$ block matrix of projection vectors $\boldsymbol{W}^i$, where i is the iteration index, with the goal of obtaining $\lim_{i\to\infty}\boldsymbol{W}^i = \boldsymbol{W}$ as defined in (7). Based on the partitioning defined above, node k will be responsible for updating submatrix $\boldsymbol{W}_k^i$. In order to derive the algorithm for updating $\boldsymbol{W}^i$, we note that (7) can be formulated as a constrained optimization problem in the variable $\boldsymbol{W}$ [5]:

$$\min_{\boldsymbol{W}} \quad \mathrm{Tr}\left(\boldsymbol{W}^H\boldsymbol{R}_{D_{xx}}\boldsymbol{W}\right) \quad (8)$$

$$\text{s.t.} \quad \boldsymbol{W}^H\boldsymbol{R}_{\boldsymbol{x}\boldsymbol{x}}\boldsymbol{W} = K^2\boldsymbol{I}_Q \quad (9)$$

where $\mathrm{Tr}\left(\cdot\right)$ denotes the trace operator.

A. Star-Topology Networks

A star-topology network is a special case of a tree-topology network where all nodes are leaf nodes (i.e. nodes with a single neighbor) except for a center node k_c. We denote the

[1] The term "intersection" is not to be taken literally here, as the actual intersection is typically zero due to measurement noise. A formal definition of the targeted subspace is given in (2)-(3).

[2] (4) can be obtained by formulating (2)-(3) in terms of samples rather than random signals and differentiating with respect to the sample matrix of $\boldsymbol{s}$.

[3] In the formulation of [12], $\boldsymbol{R}_{D_{xx}}$ and $\boldsymbol{R}_{xx}$ are switched. This results in the same solution, yet with a different normalization than (3).

set of leaf nodes as $\mathcal{K}_l = \mathcal{K} \setminus \{k_c\}$. In what follows, we show how (8)-(9) (and therefore (2)-(3)) can be solved in a star-topology network in which each node only transmits Q-dimensional signals to its neighbors. As a basis for our developments, we define the following algorithm for solving (8)-(9) via an alternating optimization (AO) procedure:

1) Set $i \leftarrow 0$, $q \leftarrow 1$ and randomly initialize $\boldsymbol{W}^0$.
2) Choose $\boldsymbol{W}^{i+1}$ as a solution of

$$\min_{\boldsymbol{W}} \quad \text{Tr}\left(\boldsymbol{W}^H \boldsymbol{R}_{D_{\boldsymbol{xx}}} \boldsymbol{W}\right) \tag{10}$$

$$\text{s.t.} \quad \boldsymbol{W}^H \boldsymbol{R}_{\boldsymbol{xx}} \boldsymbol{W} = K^2 \boldsymbol{I}_Q \tag{11}$$

$$\mathcal{C}(\boldsymbol{W}_{-q}) \subseteq \mathcal{C}(\boldsymbol{W}^i_{-q}) \text{ if } q \in \mathcal{K}_l \tag{12}$$

$$\mathcal{C}(\boldsymbol{W}_k) \subseteq \mathcal{C}(\boldsymbol{W}^i_k) \; \forall k \in \mathcal{K}_l \text{ if } q \notin \mathcal{K}_l \tag{13}$$

where $\mathcal{C}(\boldsymbol{W})$ denotes the column space of $\boldsymbol{W}$ and $\boldsymbol{W}_{-q}$ is the block matrix obtained by removing the rows of $\boldsymbol{W}$ corresponding to $\boldsymbol{W}_q$.
3) Set $i \leftarrow i+1$ and $q \leftarrow (q \mod K) + 1$.
4) Return to step 2.

The above procedure must result in a monotonic decrease of the objective function (10), since the solution in the previous iteration is by definition also in the constraint set of the current iteration. The introduction of constraint (12) and (13), although limiting the available degrees of freedom, is the essential element allowing the algorithm to be extented to solve (8)-(9) in a distributed fashion, without each node needing access to the full $\boldsymbol{x}$, as is shown next. We first note that the constraints can be equivalently formulated as

$$\exists \boldsymbol{H} \in \mathbb{R}^{Q \times Q} : \boldsymbol{W}_k = \boldsymbol{W}^i_k \boldsymbol{H} \; \forall k \in \mathcal{K} \setminus \{q\} \;\text{ if } q \in \mathcal{K}_l \tag{14}$$

$$\exists \boldsymbol{G}_k \in \mathbb{R}^{Q \times Q} : \boldsymbol{W}_k = \boldsymbol{W}^i_k \boldsymbol{G}_k \; \forall k \in \mathcal{K}_l \;\text{ if } q \notin \mathcal{K}_l \tag{15}$$

In iteration i of the D-MAXVAR algorithm, node k will send observations of a fused Q-channel signal defined as

$$\overline{\boldsymbol{x}}^i_k = \begin{cases} \boldsymbol{W}^{iH}_k \boldsymbol{x}_k & \text{if } q \in \mathcal{K}_l \\ \boldsymbol{W}^{iH}_{k_c} \boldsymbol{x}_{k_c} + \sum_{l \in \mathcal{K}_l} \overline{\boldsymbol{x}}^i_l & \text{if } k = k_c \end{cases} \tag{16}$$

The definition is recursive, resulting in the center node aggregating the compressed observations of the leaf nodes in a sum[4]. We can write for $q = k_c$:

$$\boldsymbol{W}^H \boldsymbol{x} = \boldsymbol{W}^{iH}_q \boldsymbol{x}_q + \sum_{k \neq q} \boldsymbol{G}^H_k \overline{\boldsymbol{x}}^i_k = \tilde{\boldsymbol{W}}^H_q \tilde{\boldsymbol{x}}^i_q \tag{17}$$

with

$$\tilde{\boldsymbol{W}}_q = \left[\boldsymbol{W}^T_q \mid \boldsymbol{G}^T_1 \mid \cdots \mid \boldsymbol{G}^T_{q-1} \mid \boldsymbol{G}^T_{q+1} \mid \cdots \mid \boldsymbol{G}^T_K\right]^T \tag{18}$$

$$\tilde{\boldsymbol{x}}^i_q = \left[\boldsymbol{x}^T_q \mid \overline{\boldsymbol{x}}^{iT}_1 \mid \cdots \mid \overline{\boldsymbol{x}}^{iT}_{q-1} \mid \overline{\boldsymbol{x}}^{iT}_{q+1} \mid \cdots \mid \overline{\boldsymbol{x}}^{iT}_K\right]^T \tag{19}$$

Similarly for $q \in \mathcal{K}_l$, we have

$$\boldsymbol{W}^H \boldsymbol{x} = \boldsymbol{W}^{iH}_q \boldsymbol{x}_q + \boldsymbol{H}^H (\overline{\boldsymbol{x}}^i_{k_c} - \overline{\boldsymbol{x}}^i_q) = \tilde{\boldsymbol{W}}^H_q \tilde{\boldsymbol{x}}^i_q \tag{20}$$

where the substraction of node q's own compressed observations is required as they are present in the center node's $\overline{\boldsymbol{x}}^i_{k_c}$. This results in the following definitions for $\tilde{\boldsymbol{x}}^i_q$ and $\tilde{\boldsymbol{W}}^i_q$ when node q is a leaf node:

$$\tilde{\boldsymbol{W}}_q = \left[\boldsymbol{W}^T_q \mid \boldsymbol{H}^T\right]^T \tag{21}$$

$$\tilde{\boldsymbol{x}}^i_q = \left[\boldsymbol{x}^T_q \mid \overline{\boldsymbol{x}}^{iT}_{k_c} - \overline{\boldsymbol{x}}^{iT}_q\right]^T \tag{22}$$

Note that we have different definitions for $\tilde{\boldsymbol{x}}^i_q$ and $\tilde{\boldsymbol{W}}^i_q$ for center and leaf nodes.

By acknowledging that (14)-(15) define a parametrization of $\boldsymbol{W}$ which by construction satisfies the constraints (12)-(13) and that $\boldsymbol{H}$ or the $\boldsymbol{G}_k$'s (if $q = k_c$) can be used by node q to manipulate the $\boldsymbol{W}_k$'s of nodes $k \neq q$, the above definitions allow us to reformulate (10)-(13) as a local problem at node q of the same form as our original centralized problem (i.e. a GEVD):

$$\min_{\tilde{\boldsymbol{W}}_q} \quad \text{Tr}\left(\tilde{\boldsymbol{W}}^H_q \boldsymbol{R}^i_{\tilde{D}_q} \tilde{\boldsymbol{W}}_q\right) \tag{23}$$

$$\text{s.t.} \quad \tilde{\boldsymbol{W}}^H_q \boldsymbol{R}^i_{\tilde{\boldsymbol{x}}_q \tilde{\boldsymbol{x}}_q} \tilde{\boldsymbol{W}}_q = K^2 \boldsymbol{I}_Q \tag{24}$$

where $\boldsymbol{R}^i_{\tilde{D}_q}$ is the block diagonal matrix with each diagonal block corresponding to the partionionings defined in (19) and (22), that is if $q = k_c$,

$$\begin{aligned} \boldsymbol{R}^i_{\tilde{D}_q} = \text{Blkdiag}(&\boldsymbol{R}_{\boldsymbol{x}_q \boldsymbol{x}_q}, \boldsymbol{R}^i_{\overline{\boldsymbol{x}}_1 \overline{\boldsymbol{x}}_1}, \ldots, \\ &\ldots, \boldsymbol{R}^i_{\overline{\boldsymbol{x}}_{q-1} \overline{\boldsymbol{x}}_{q-1}}, \boldsymbol{R}^i_{\overline{\boldsymbol{x}}_{q+1} \overline{\boldsymbol{x}}_{q+1}}, \ldots, \\ &\ldots, \boldsymbol{R}^i_{\overline{\boldsymbol{x}}_K \overline{\boldsymbol{x}}_K}) \end{aligned} \tag{25}$$

else for $q \in \mathcal{K}_l$,

$$\boldsymbol{R}^i_{\tilde{D}_q} = \text{Blkdiag}(\boldsymbol{R}_{\boldsymbol{x}_q \boldsymbol{x}_q}, \boldsymbol{R}^i_{\Sigma_q}) \tag{26}$$

where $\boldsymbol{R}^i_{\Sigma_q}$ contains the sum of pairwise compressed variables covariance matrices:

$$\boldsymbol{R}^i_{\Sigma_q} = \sum_{l \neq q} \boldsymbol{R}^i_{\overline{\boldsymbol{x}}_l \overline{\boldsymbol{x}}_l} \tag{27}$$

Note that the constraints (12)-(13) are automatically satisfied due to the implicit parameterization of $\boldsymbol{W}$ through (14)-(15).

The AO procedure described above can now be efficiently distributed by defining a three-step procedure applied at each iteration i with updating node q:

1) *Aggregation:* The center node collects the compressed observations defined in (16) from all the leaf nodes except the updating node q. If node q is a leaf node, the center node transmits the aggregated compressed observations defined by (16) and $\boldsymbol{R}^i_{\Sigma_q}$ defined in (27) to node q.
2) *Local solution:* The updating node q forms the matrix $\boldsymbol{R}^i_{\tilde{D}_q}$ and vector $\tilde{\boldsymbol{x}}^i_q$, estimates $\boldsymbol{R}^i_{\tilde{\boldsymbol{x}}_q \tilde{\boldsymbol{x}}_q}$ and solves the local problem defined by (23)-(24).
3) *Update:* The appropriate update matrices $\boldsymbol{G}_{(\cdot)}$ or $\boldsymbol{H}$ are obtained from the local solution $\tilde{\boldsymbol{W}}_q$ according to the partitionings (18) or (21), respectively. The update matrices are then propagated into the network such that each node can update its local solution as

$$\boldsymbol{W}^{i+1}_k = \begin{cases} \boldsymbol{W}_q & \text{if } k = q \\ \boldsymbol{W}^i_k \boldsymbol{G}_k \text{ or } \boldsymbol{W}^i_k \boldsymbol{H} & \text{if } k \neq q \end{cases} \tag{28}$$

where the $\boldsymbol{G}_k$'s are used instead of $\boldsymbol{H}$ if the updating node q is the center node.

[4] Note that this implies a two-step process to generate the compressed observations at the center node: the leaf nodes first send their compressed observations to the center node, after which they are combined with the center node's observations.

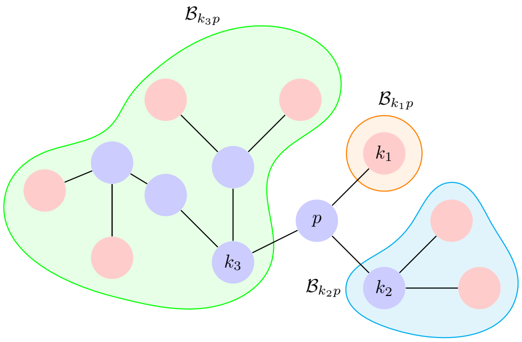

Fig. 1. In this example tree, the subtree $\mathcal{B}_{k_1p}$ is highlighted in orange, $\mathcal{B}_{k_2p}$ in blue and $\mathcal{B}_{k_3p}$ in green. Leaf nodes belonging to $\mathcal{K}_l$ are colored red.

Note that this procedure can be trivially extented to work in fully-connected networks by always considering node q as the center node and all other nodes as leaf nodes. Due to space constraints and considering that a star-topology is a special case of the tree-topology described hereafter, we do not give a detailed description of the algorithm for star-topology networks and refer the reader to the next section for the detailed description of a generalization of this algorithm.

B. Tree-Topology Networks

We now consider a set of nodes organised in a tree-topology network. As a tree-topology network consists of a set of nested star-topology networks, the above procedure can be relatively straightforwardly extented to tree-topology networks.

We denote $\mathcal{N}_k$ the set of neighboring nodes of node k (i.e. sharing a link with node k) and denote $\overline{\boldsymbol{x}}_{kp}^i$ the compressed Q-channel signal that node k sends to node p such that:

$$\overline{\boldsymbol{x}}_{kp}^i = \boldsymbol{W}_k^{iH}\boldsymbol{x}_k + \sum_{l\in\mathcal{N}_k\setminus\{p\}} \overline{\boldsymbol{x}}_{lk}^i \tag{29}$$

According to this new definition, $\overline{\boldsymbol{x}}_{kp}^i$ contains the sum of the compressed observations of the nodes in the subtree with node k at its root and obtained by ignoring the link between node k and node p. We denote the set of nodes in that subtree $\mathcal{B}_{kp}$. An example tree depicting this concept is visible on fig. 1. Similarly, we generalize (27) as

$$\boldsymbol{R}_{\Sigma_{kp}}^i = \boldsymbol{R}_{\overline{\boldsymbol{x}}_k^i\overline{\boldsymbol{x}}_k^i} + \sum_{l\in\mathcal{N}_k\setminus\{p\}} \boldsymbol{R}_{\Sigma_{lk}}^i \tag{30}$$

The recursive procedure to obtain $\overline{\boldsymbol{x}}_{kq}^i$ and $\boldsymbol{R}_{\Sigma_{kq}}^i$ for a window of T' samples is described by Algorithm 1.

We redefine (18), (19) and (26) as

$$\tilde{\boldsymbol{W}}_q = \left[\boldsymbol{W}_q^T \mid \boldsymbol{G}_{l_1}^T \mid \cdots \mid \boldsymbol{G}_{l_{n_q}}^T\right]^T \tag{31}$$

$$\tilde{\boldsymbol{x}}_q^i = \left[\boldsymbol{x}_q^T \mid \overline{\boldsymbol{x}}_{l_1q}^{iT} \mid \cdots \mid \overline{\boldsymbol{x}}_{l_{n_q}q}^{iT}\right]^T \tag{32}$$

$$\boldsymbol{R}_{\tilde{D}_q} = \text{Blkdiag}(\boldsymbol{R}_{\boldsymbol{x}_q\boldsymbol{x}_q}, \boldsymbol{R}_{\Sigma_{l_1}q}^i, \ldots, \boldsymbol{R}_{\Sigma_{l_{n_q}}q}^i) \tag{33}$$

with $\{l_1, \ldots, l_{n_q}\} = \mathcal{N}_q$. Those definitions allow us to reformulate the global problem (10)-(13) as the local problem (23)-(24) in terms of the locally accessible variables $\tilde{\boldsymbol{x}}_q^i$, $\boldsymbol{x}_q$ and $\boldsymbol{R}_{\tilde{D}_q}$. The first aggregation step of the procedure defined at the end of section III-A can be carried out in a tree-topology network using Algorithm 1. The full D-MAXVAR algorithm involving all three steps is described by Algorithm 2.

Algorithm 1: Recursive procedure for aggregating observations in a tree-topology network

```
procedure aggregate(k, p, t)
    for $l \in \mathcal{N}_k \setminus \{p\}$ do
        aggregate(l, k, t)
    At node $k$
        Compute
          $\overline{\boldsymbol{X}}_{kp}^i[t] = \boldsymbol{W}_k^{iH}\boldsymbol{X}_k[t] + \sum_{l\in\mathcal{N}_k\setminus\{p\}} \overline{\boldsymbol{X}}_{lk}^i[t]$
        if $k \notin \mathcal{K}_l$ then
            Compute
              $\boldsymbol{R}_{\Sigma_{kp}}^i[t] = \boldsymbol{R}_{\overline{\boldsymbol{x}}_k^i\overline{\boldsymbol{x}}_k^i}[t] + \sum_{l\in\mathcal{N}_k\setminus\{p\}} \boldsymbol{R}_{\Sigma_{lk}}^i[t]$
            Send $(\overline{\boldsymbol{X}}_{kp}^i[t], \boldsymbol{R}_{\Sigma_{kp}}^i[t])$ to node $p$
        else
            Send $\overline{\boldsymbol{X}}_{kp}^i[t]$ to node $p$
```

Algorithm 2: D-MAXVAR algorithm in a tree-topology network.

```
begin
    $i \leftarrow 0$
    $t \leftarrow t_0$
    Initialize updating node as $q \leftarrow 1$
    Randomly initialize the the $\boldsymbol{W}_k^0$'s
    loop
        for $k \in \mathcal{N}_q$ do
            aggregate(k, q, t) (see Algorithm 1)
        At node $q$
            Estimate $\boldsymbol{R}_{\tilde{\boldsymbol{x}}_q\tilde{\boldsymbol{x}}_q}^i[t]$ and $\boldsymbol{R}_{\tilde{D}_q}[t]$
            $\tilde{\boldsymbol{W}}_q \leftarrow Q$ GEVC corresponding to the
              smallest GEVL of the matrix pencil
              $(\boldsymbol{R}_{\tilde{D}_q}[t], \frac{1}{K^2}\boldsymbol{R}_{\tilde{\boldsymbol{x}}_q\tilde{\boldsymbol{x}}_q}^i[t])$
            $\boldsymbol{W}_q^{i+1} \leftarrow \left[\boldsymbol{I}_{M_q}\boldsymbol{0}\right]\tilde{\boldsymbol{W}}_q$
            for $k \in \mathcal{N}_q$ do
                Select $\boldsymbol{G}_k$ as the block of $\tilde{\boldsymbol{W}}_q$ (see
                  (31)) corresponding to node $k$ and
                  disseminate within the branch $\mathcal{B}_{kq}$
                for $l \in \mathcal{B}_{kq}$ do
                    At node $l$
                        $\boldsymbol{W}_l^{i+1} \leftarrow \boldsymbol{W}_l^i\boldsymbol{G}_k$
        $i \leftarrow i + 1$
        $t \leftarrow t + T'$
        $q \leftarrow (q \mod K) + 1$
```

Finally we note that each node k can obtain the current network-wide estimate of $\boldsymbol{s}_{opt}$ as

$$\hat{\boldsymbol{s}}^i = \frac{1}{K}\left(\boldsymbol{W}_k^{iH}\boldsymbol{x}_k + \sum_{l\in\mathcal{N}_k} \overline{\boldsymbol{x}}_{lk}^i\right). \tag{34}$$

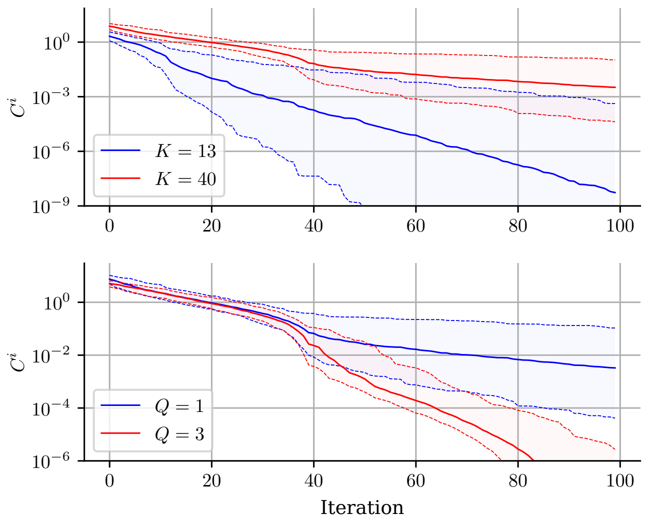

Fig. 2. Plots of the performance metric (36) for 100 MC runs. Top: $Q = 1$, varying tree size K. Bottom: $K = 40$, varying Q. The curves correspond to median values, the shaded areas between dashed lines depict the 5%-95% percentiles regions.

C. Convergence

Convergence of Algorithm 2 can only be obtained in expectation as each iteration of Algorithm 2 uses a different block of samples to estimate the second-order statistics (exploiting the stationarity assumption). While a formal convergence proof is omitted due to page limitations, we provide a brief proof outline. The convergence in the star topology follows from the fact that it exactly mimics the iterations of the AO procedure defined in section III-A. Convergence of the latter can be proven based on the monotonic increase of its objective function across iterations (a formal proof of a very similar AO algorithm is given in [6]). Finally, the convergence of the tree-topology algorithm follows from very similar arguments, where the main difference is that the constraint set (12)-(13) (which is here linked to a star topology) is re-defined based on the tree-topology (details omitted).

D. Complexity and Communication Cost

The exact order of complexity and communication cost obviously depends on the specific topology under consideration but they can still be expressed per node in terms of the number of neighbors $|\mathcal{N}_k|$. The complexity of the local GEVDs at node k is $\mathcal{O}\left((M_k + Q|\mathcal{N}_k|)^3\right)$ while the average communication cost over each link is $\mathcal{O}((Q + T')Q)$ per iteration (where T', typically $\gg Q$, is the window length used in Algorithm 2).

IV. Simulations

In this section, we validate our algorithm with Monte-Carlo (MC) simulations. We applied Algorithm 2 to tree-topology networks with a branching factor of 3 and $M_k = 6\ \forall k \in \mathcal{K}$. For each run, a synthetic observation vector $\boldsymbol{x}_k$ was generated for each node k as

$$\boldsymbol{x}_k = \boldsymbol{A}_k \boldsymbol{F}_k \boldsymbol{y} + \boldsymbol{n}_k \quad \forall k \in \mathcal{K} \tag{35}$$

where $\boldsymbol{y}$ is a 3-dimensional zero-mean unit-variance gaussian latent signal common to all nodes, $\boldsymbol{A}_k$ an $M_k \times 3$ random mixing matrix whose entries are drawn from a gaussian distribution with zero mean and unit variance and $\boldsymbol{F}_k$ is a 3×3 diagonal matrix whose diagonal entries are set to 1 with probability 0.2 or else are set to 0. This results in each latent signal y_i being sensed by 20% of the nodes on average. $\boldsymbol{n}_k$ is an M_k-dimensional vector of additive gaussian noise with zero mean and unit variance. As a performance metric, we used

$$C^i = 1 - \frac{J(\boldsymbol{W^i})}{J(\boldsymbol{W^*})} \tag{36}$$

where $\boldsymbol{W}^*$ is the projection matrix obtained by centrally solving (8)-(9) and $J(\boldsymbol{W})$ is the objective minimized in (5). The resulting convergence curves are visible on fig. 2.

V. Conclusions

We have proposed a novel distributed MAXVAR algorithm (D-MAXVAR) allowing the estimation of the average pairwise intersection of per-node sensor signal subspaces. By exchanging jointly compressed sensor observations, the nodes have bandwidth and processing requirements depending solely on their number of neighbors and the fixed compression parameter Q and indepentent of the total number of nodes in the network. The algorithm converges to the centralized MAXVAR solution, which was also demonstrated by simulations on synthetic data (a formal convergence proof was omitted due to page constraints). Future work will focus on how the knowledge of $\boldsymbol{s}_{opt}$ can be exploited to cluster nodes according to similarities in their observed signals.

References

[1] G. H. Golub and C. F. Van Loan, “Matrix computations, 4th,” *Johns Hopkins*, 2013.

[2] M. Sørensen, C. I. Kanatsoulis, and N. D. Sidiropoulos, “Generalized canonical correlation analysis: A subspace intersection approach,” *arXiv preprint arXiv:2003.11205*, 2020.

[3] A. Bertrand and M. Moonen, “Distributed adaptive estimation of covariance matrix eigenvectors in wireless sensor networks with application to distributed PCA,” *Signal Processing*, vol. 104, pp. 120–135, 2014.

[4] A. Scaglione, R. Pagliari, and H. Krim, “The decentralized estimation of the sample covariance,” in *2008 42nd Asilomar Conference on Signals, Systems and Computers*. IEEE, 2008, pp. 1722–1726.

[5] 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.

[6] 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.

[7] 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.

[8] J. Chen and I. D. Schizas, “Distributed sparse canonical correlation analysis in clustering sensor data,” in *2013 Asilomar Conference on Signals, Systems and Computers*. IEEE, 2013, pp. 639–643.

[9] J. R. Kettenring, “Canonical analysis of several sets of variables,” *Biometrika*, vol. 58, no. 3, pp. 433–451, 1971.

[10] X. Fu, K. Huang, E. E. Papalexakis, H. A. Song, P. Talukdar, N. D. Sidiropoulos, C. Faloutsos, and T. Mitchell, “Efficient and distributed generalized canonical correlation analysis for big multiview data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2304–2318, 2018.

[11] P. Horst, *Generalized canonical correlations and their application to experimental data*. Journal of clinical psychology, 1961, no. 14.

[12] J. Vía, I. Santamaría, and J. Pérez, “A learning algorithm for adaptive canonical correlation analysis of several data sets,” *Neural Networks*, vol. 20, no. 1, pp. 139–152, 2007.