DECENTRALIZED LEARNING IN THE PRESENCE OF LOW-RANK NOISE

Roula Nassif [(1)], *Virginia Bordignon* [(2)], *Stefan Vlaski* [(3)], *Ali H. Sayed* [(2)]
[(1)]Université Côte d'Azur, France
[(2)]Ecole Polytechnique Fédérale de Lausanne, Switzerland
[(3)]Imperial College London, UK

ABSTRACT

Observations collected by agents in a network may be unreliable due to observation noise or interference. This paper proposes a distributed algorithm that allows each node to improve the reliability of its own observation by relying solely on local computations and interactions with immediate neighbors, assuming that the field (graph signal) monitored by the network lies in a low-dimensional subspace and that a low-rank noise is present in addition to the usual full-rank noise. While oblique projections can be used to project measurements onto a low-rank subspace along a direction that is oblique to the subspace, the resulting solution is not distributed. Starting from the centralized solution, we propose an algorithm that performs the oblique projection of the overall set of observations onto the signal subspace in an iterative and distributed manner. We then show how the oblique projection framework can be extended to handle distributed learning and adaptation problems over networks.

***Index Terms*—Low-rank noise, subspace constraints, distributed oblique projection, learning and adaptation.**

I. INTRODUCTION

We consider N agents in a network, collecting data with the objective of collaboratively solving some inference task. The locally observed data may be unreliable due to the presence of measurement noise or interference. Most prior literature treats noise as a full-rank process in the measurement space. The work [1], for instance, models the desired graph signal as a vector that lies in a low-rank subspace and the noise as a vector that may fall anywhere in the observation space. Orthogonal projection techniques have been used to recover the original signal and to mitigate the effect of noise. The projection in [1] is carried out through a distributed network, with no fusion center, where each node exchanges information only with its neighbors. While centralized solutions can be powerful, decentralized solutions are more attractive since they are more robust, and allow agents to keep their local data private [2]. Distributed algorithms and their ability to perform globally optimal processing tasks (such as minimizing aggregate sums of individual costs, solving constrained optimization problems, etc.) have been widely studied in the literature [2]–[14].

In this paper, we consider distributed estimation in the presence of low-rank, or structured, noise in addition to the usual full-rank noise. In the first part, and for motivational purposes, we consider a de-noising problem where the graph signal to be estimated lies in a low-dimensional subspace. That is, we consider a connected network (or graph) of N nodes and we let y_k denote the scalar measurement collected by node k. Let $y = \text{col}\{y_1, \ldots, y_N\}$ denote the collection of observations from across the network. We assume the following model:

$$y = \underbrace{Wx_W}_{\text{useful signal}} + \underbrace{Zx_Z + v}_{\text{noise component}}, \tag{1}$$

where W is an $N \times P$ full-column rank matrix with $P \ll N$ and x_W is a $P \times 1$ column vector. The additive noise in the network is modeled in two parts: the unstructured $N \times 1$ vector noise v and the structured, or low-rank, noise Zx_Z that lies in the subspace spanned by the columns of the $N \times L$ full-column rank matrix Z (with $L \ll N$ and x_Z an $L \times 1$ vector). The structured noise Zx_Z can be any signal that interferes with the signal of primary interest Wx_W. We assume that the columns of the matrices W and Z are linearly independent so that the composite matrix $[W\ Z]$ is full-column rank ($P + L \leq N$). Note that the linear independence assumption implies that the intersection between the spaces spanned by the columns of W and Z contains only the zero vector.

The linear data model (1), which assumes that part of the noise is a process occurring in a space of lower dimensionality, arises in many signal processing applications. For example, in narrowband array processing, the measurement model is of the form (1) where $Wx_W = a(\phi_0)x_W$ corresponds to the signal arriving from angle ϕ_0, and $Zx_Z = [a(\phi_1)\ldots a(\phi_M)]x_Z$ is the interference corresponding to other propagating signals from angles $\phi_1, \ldots, \phi_M$ [15], [16]. The objective in this application is to enhance Wx_W and to null Zx_Z.

In Sec. II, we assume that the model matrices W and Z are known, and that the objective is to estimate in a distributed manner the useful signal $w \triangleq Wx_W$ based on the measurement vector y. We explain how oblique projections can solve the problem by projecting onto $\mathcal{R}(W)$ along the parallel direction to $\mathcal{R}(Z)$, and then we show how the projection can be performed in a distributed and iterative manner. The concepts developed in Sec. II will then serve as the foundation for the design of learning algorithms in Sec. III where the static data model (1) is generalized by allowing for *streaming data* scenarios, *vector valued observations* at the agents, and more *general observation models*. In particular, we generalize the oblique projection framework by considering learning problems of the form:

$$\begin{aligned}&\underset{\mathcal{Y},\mathcal{X}_{\mathcal{W}},\mathcal{X}_{\mathcal{Z}}}{\text{argmin}} \left\{ J^{\text{glob}}(\mathcal{Y}) \triangleq \sum_{k=1}^{N} J_k(y_k) \right\} \\ &\text{subject to } \mathcal{Y} = \mathcal{W}\mathcal{X}_{\mathcal{W}} + \mathcal{Z}\mathcal{X}_{\mathcal{Z}}\end{aligned} \tag{2}$$

where $J_k(y_k)$ is a differentiable convex cost associated with agent k, y_k is an $M_k \times 1$ vector, $\mathcal{Y} = \text{col}\{y_k\}_{k=1}^N$, $\mathcal{X}_{\mathcal{W}}$ is a $P \times 1$ vector and $\mathcal{X}_{\mathcal{Z}}$ is an $L \times 1$ vector. Agent k is interested in estimating w_k, the k-th subvector of $\mathcal{w} = \mathcal{W}\mathcal{X}_{\mathcal{W}} \in \mathcal{R}(\mathcal{W})$. Let $M = \sum_{k=1}^{N} M_k$. The $M \times P$ and $M \times L$ matrices $\mathcal{W}$ and $\mathcal{Z}$ are full-column rank ($P \ll M$ and $L \ll M$) and their columns $\mathcal{W}$ and $\mathcal{Z}$ are linearly independent. The cost $J_k(y_k)$ is assumed to be expressed as the expectation of some loss function $Q_k(\cdot)$ and written as $J_k(y_k) = \mathbb{E}Q_k(y_k; \boldsymbol{\zeta}_k)$, where $\boldsymbol{\zeta}_k$ denotes the random data. We are interested in solving the problem in the stochastic setting when the distribution of the data $\boldsymbol{\zeta}_k$ is unknown. In this case, and instead of employing true gradient vectors at iteration i, it is common to employ approximate vectors of the form [2]:

$$\widehat{\nabla_{y_k} J_k}(y_k) = \nabla_{y_k} Q_k(y_k; \boldsymbol{\zeta}_{k,i}) \tag{3}$$

where $\boldsymbol{\zeta}_{k,i}$ represents the data observed at iteration i.

Notation: We use boldface letters for random quantities and normal letters for deterministic quantities. Lowercase letters denote column vectors and uppercase letters denote matrices. Unless otherwise specified, we use calligraphic fonts to denote block matrices and block vectors. In fact, block quantities appear in Sec. III where inference problems over networks are considered.

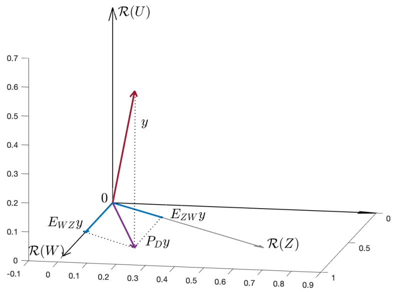

Fig. 1. Illustration of oblique projection.

II. DISTRIBUTED OBLIQUE PROJECTION

In this section, we explain how the useful signal Wx_W in (1) can be estimated in a distributed and iterative manner. To that end, we re-write model (1) as:

$$y = Dx + v, \tag{4}$$

where $D = [W \; Z]$ and $x = \text{col}\{x_W, x_Z\}$. By minimizing the norm of the error, namely, $\|y - Dx\|^2$, we obtain:

$$x^o = (D^\top D)^{-1} D^\top y = \begin{bmatrix} W^\top W & W^\top Z \\ Z^\top W & Z^\top Z \end{bmatrix}^{-1} \begin{bmatrix} W^\top y \\ Z^\top y \end{bmatrix} \tag{5}$$

Assuming, without loss of generality, that the columns of W and Z are orthonormal (i.e., $W^\top W = I_P$ and $Z^\top Z = I_L$), and by applying the 2×2 block matrix inversion identity [17], we find:

$$w^o = W x_W^o = E_{WZ}\, y, \tag{6}$$

where E_{WZ} is the $N \times N$ square matrix:

$$E_{WZ} \triangleq P_W (I - Z X^{-1} Z^\top P_W^\perp), \tag{7}$$

$P_W = WW^\top$ is the orthogonal projection onto $\mathcal{R}(W)$, $X = I - Z^\top P_W Z = Z^\top P_W^\perp Z$, and $P_W^\perp = I - P_W$. The matrix E_{WZ} is referred to as the *oblique projection* whose range is $\mathcal{R}(W)$ and whose null space contains $\mathcal{R}(Z)$ [15]. It has the following properties–see [15, Sec. III]:

1) It is equal to $W(W^\top P_Z^\perp W)^{-1} W^\top P_Z^\perp$, where $P_Z = ZZ^\top$ and $P_Z^\perp = I - P_Z$;
2) It is idempotent ($E_{WZ} = E_{WZ}^2$), but not symmetric;
3) Its range is $\mathcal{R}(W)$ (since $E_{WZ}W = W$);
4) Its null space is $\mathcal{R}([Z \; U])$ where U spans the perpendicular space to $\mathcal{R}([W \; Z])$ ($E_{WZ}Z = 0$ and $E_{WZ}U = 0$);
5) It has P eigenvalues at 1 and $N - P$ eigenvalues at 0. Its singular values are $0, 1$, or any value greater than 1;
6) The orthogonal projection onto $\mathcal{R}(D)$ can be written as:

$$P_D = D(D^\top D)^{-1} D^\top = E_{WZ} + E_{ZW}, \tag{8}$$

where $E_{ZW} = Z X^{-1} Z^\top P_W^\perp$. As illustrated in Fig. 1, the oblique projector operator E_{WZ} projects vectors onto $\mathcal{R}(W)$ along the direction parallel to $\mathcal{R}(Z)$, and likewise for the oblique projector E_{ZW}. Any vector $y \in \mathbb{R}^N$ can be decomposed as:

$$y = E_{WZ}\, y + E_{ZW}\, y + P_U\, y, \tag{9}$$

where P_U denotes the orthogonal projector onto $\mathcal{R}(U)$.

The computation in (6) is centralized since matrix E_{WZ} is dense in general, requiring nodes to send their measurements $\{y_k\}$ to a fusion center that performs the oblique projection. The objective is to compute the projection in (6) with a network, where each node performs local computations and exchanges information only with its neighbors. We propose to replace the $N \times N$ oblique projector E_{WZ} in (6) by an $N \times N$ matrix A that satisfies the following conditions [5], [7]:

$$\lim_{i \to \infty} A^i = E_{WZ}, \tag{10}$$

$$[A]_{k\ell} = a_{k\ell} = 0, \quad \text{if } \ell \notin \mathcal{N}_k \text{ and } k \neq \ell, \tag{11}$$

where $a_{k\ell}$ denotes the (k, ℓ)-th component of A. The sparsity condition (11) characterizes the network topology and ensures local exchange of information at each iteration i. By replacing the projector E_{WZ} in (6) with A, we obtain the following recursion at agent k:

$$w_k(i) = \sum_{\ell \in \mathcal{N}_k} a_{k\ell}\, w_\ell(i-1), \qquad i \geq 0 \tag{12}$$

where $w_k(i)$ is the estimate of $[w^o]_k$ at iteration i and where node k initializes its state variable with its local measurement, i.e., $w_k(-1) = y_k$. Let $w_i = \text{col}\{w_1(i), \ldots, w_N(i)\}$. Condition (12) ensures that the network vector w_i converges to the oblique projection of the initial vector $w_{-1} = y$ onto $\mathcal{R}(W)$ along the direction parallel to $\mathcal{R}(Z)$. Necessary and sufficient conditions for (10) to hold are given in the following lemma.

Lemma 1. *The matrix equation* (10) *holds, if and only if,* $E_{WZ} = E_{WZ}^2$ *(this condition is satisfied by the oblique projector* E_{WZ}*) and the following conditions on* A *are satisfied:*

$$A\, E_{WZ} = E_{WZ}, \tag{13}$$

$$E_{WZ}\, A = E_{WZ}, \tag{14}$$

$$\rho(A - E_{WZ}) < 1, \tag{15}$$

where $\rho(\cdot)$ *denotes the spectral radius of its matrix argument. It follows that any* A *satisfying condition* (10) *has one as an eigenvalue with multiplicity* P*, and all other eigenvalues are strictly less than one in magnitude.*

Proof. The arguments are along the lines developed in [5, Appendix A] for learning under subspace constraints. □

If we replace E_{WZ} by (7) and multiply both sides of (13) by W, we find that condition (13) implies $AW = W$. Thus, the P columns of W are the right eigenvectors of A associated with the eigenvalue 1. The left-eigenvectors corresponding to the eigenvalue 1 are given by $W^\top E_{WZ}$. To see this, replace E_{WZ} by (7) and multiply both sides of (14) by $W^\top$.

Let $\widetilde{w}_i \triangleq w^o - w_i$ denote the network error vector. Using (6), (12), and (13), we have $w^o = Aw^o$ so that:

$$\widetilde{w}_i = w^o - Aw_{i-1} = (A - E_{WZ})w^o - Aw_{i-1} + E_{WZ}w^o. \tag{16}$$

Using the fact that $w_{i-1} = A^i w_{-1} = A^i y$, we can write:

$$E_{WZ}w^o \overset{(6)}{=} E_{WZ}\, y \overset{(14)}{=} E_{WZ}A^i y = E_{WZ}w_{i-1}. \tag{17}$$

Replacing (17) into (16), we arrive at:

$$\widetilde{w}_i = (A - E_{WZ})\widetilde{w}_{i-1}. \tag{18}$$

Condition (15) guarantees convergence of the network error vector toward 0, namely, $\lim_{i \to \infty} \widetilde{w}_i = 0$.

Remark 1: The signal subspace model considered in [1] can be recast in the form (1) with $W = 0$ (since the noise process in [1] is assumed to be full-rank). In this case, the oblique projector E_{WZ} in (6) reduces to the orthogonal projector P_W and the convergence conditions of the algorithm proposed in [1] can be obtained from conditions (13)–(15) by replacing E_{WZ} by P_W.

Whenever the sparsity constraint (11) and the signal subspace lead to a feasible problem, the matrix A maximizing the convergence speed of (12) to w^o and satisfying (13)–(15) and (11) can be found by solving an appropriate semi-definite program (SDP) using convex optimization packages such as [18]. This can be achieved by following arguments similar to those used in [14], [19].

III. ADAPTATION AND LEARNING OVER NETWORKS IN THE PRESENCE OF OBLIQUE PROJECTIONS

We now consider inference problems over networks of the form (2) where each agent k is interested in estimating w_k^o, the k-th subvector of $\mathcal{w}^o = \mathcal{W}\mathcal{x}^o_{\mathcal{W}}$. We first derive the centralized solution, and then we propose a distributed solution. Recall that the composite matrix $\mathcal{D} \triangleq [\mathcal{W}\ \mathcal{Z}]$ is full-column rank ($P + L \leq M$). We also assume that the columns of $\mathcal{W}$ and $\mathcal{Z}$ are orthonormal.

III-A. Centralized adaptive solution

To solve the constrained problem (2), we employ a penalty method and solve instead the following unconstrained problem:

$$\underset{\mathcal{y},\mathcal{x}_{\mathcal{W}},\mathcal{x}_{\mathcal{Z}}}{\text{argmin}}\ J^{\text{glob}}(\mathcal{y}) + \frac{\eta}{2}\|\mathcal{y} - \mathcal{W}\mathcal{x}_{\mathcal{W}} - \mathcal{Z}\mathcal{x}_{\mathcal{Z}}\|^2 \tag{19}$$

where $\eta > 0$ is a finite large regularization parameter. Since problem (19) is convex in $\{\mathcal{y}, \mathcal{x}_{\mathcal{W}}, \mathcal{x}_{\mathcal{Z}}\}$, minimizing over $\mathcal{x} = \text{col}\{\mathcal{x}_{\mathcal{W}}, \mathcal{x}_{\mathcal{Z}}\}$ and $\mathcal{y}$ in (19) is equivalent to solving:

$$\min_{\mathcal{y}} \min_{\mathcal{x}_{\mathcal{W}},\mathcal{x}_{\mathcal{Z}}} F(\mathcal{y}, \mathcal{x}_{\mathcal{W}}, \mathcal{x}_{\mathcal{Z}}) \triangleq J^{\text{glob}}(\mathcal{y}) + \frac{\eta}{2}\|\mathcal{y} - \mathcal{W}\mathcal{x}_{\mathcal{W}} - \mathcal{Z}\mathcal{x}_{\mathcal{Z}}\|^2. \tag{20}$$

By minimizing $F(\mathcal{y}, \mathcal{x}_{\mathcal{W}}, \mathcal{x}_{\mathcal{Z}})$ over $\mathcal{x}_{\mathcal{Z}}$, we obtain:

$$\mathcal{x}^o_{\mathcal{Z}} = (\mathcal{Z}^\top \mathcal{Z})^{-1}\mathcal{Z}^\top(\mathcal{y} - \mathcal{W}\mathcal{x}_{\mathcal{W}}). \tag{21}$$

By substituting (21) into (20), we arrive at:

$$\min_{\mathcal{y}} \min_{\mathcal{x}_{\mathcal{W}}} F'(\mathcal{y}, \mathcal{x}_{\mathcal{W}}) \triangleq J^{\text{glob}}(\mathcal{y}) + \frac{\eta}{2}\|\mathcal{P}^\perp_{\mathcal{Z}}(\mathcal{y} - \mathcal{W}\mathcal{x}_{\mathcal{W}})\|^2, \tag{22}$$

where $\mathcal{P}^\perp_{\mathcal{Z}} = I - \mathcal{Z}(\mathcal{Z}^\top\mathcal{Z})^{-1}\mathcal{Z}^\top$. By minimizing $F'(\mathcal{y}, \mathcal{x}_{\mathcal{W}})$ over $\mathcal{x}_{\mathcal{W}}$, we obtain:

$$\mathcal{x}^o_{\mathcal{W}} = (\mathcal{W}^\top \mathcal{P}^\perp_{\mathcal{Z}} \mathcal{W})^{-1}\mathcal{W}^\top \mathcal{P}^\perp_{\mathcal{Z}}\mathcal{y}. \tag{23}$$

By substituting (23) into (22), we arrive at:

$$\min_{\mathcal{y}} J^{\text{glob}}(\mathcal{y}) + \frac{\eta}{2}\|\mathcal{P}^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}})\mathcal{y}\|^2, \tag{24}$$

where

$$\mathcal{E}_{\mathcal{W}\mathcal{Z}} = \mathcal{W}(\mathcal{W}^\top \mathcal{P}^\perp_{\mathcal{Z}}\mathcal{W})^{-1}\mathcal{W}^\top\mathcal{P}^\perp_{\mathcal{Z}}. \tag{25}$$

Problem (24) can be solved using the stochastic gradient descent algorithm–notice that approximate gradient vectors $\widehat{\nabla_{y_k} J_k}(\cdot)$ are used instead of true gradient vectors $\nabla_{y_k} J_k(\cdot)$:

$$\boldsymbol{\mathcal{y}}_i = \boldsymbol{\mathcal{y}}_{i-1} - \mu\,\text{col}\left\{\widehat{\nabla_{y_k} J_k}(\boldsymbol{y}_{k,i-1})\right\}_{k=1}^N - \mu\eta\mathcal{P}^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}})\boldsymbol{\mathcal{y}}_{i-1} \tag{26}$$

where μ a small step-size and where we used the fact that

$$(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}})P^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}}) = \mathcal{P}^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}}). \tag{27}$$

Instead of implementing the update (26) in one step, we implement it in two successive steps according to:

$$\begin{aligned} \boldsymbol{\psi}_{k,i} &= \boldsymbol{y}_{k,i-1} - \mu\widehat{\nabla J_k}(\boldsymbol{y}_{k,i-1}) \\ \boldsymbol{\mathcal{y}}_i &= \boldsymbol{\psi}_i - \mu\eta\mathcal{P}^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}})\boldsymbol{\mathcal{y}}_{i-1} \end{aligned} \tag{28}$$

where $\boldsymbol{\psi}_{k,i}$ is an intermediate estimate of $\mathcal{y}_k$ at agent k and iteration i and $\boldsymbol{\psi}_i = \text{col}\{\boldsymbol{\psi}_{k,i}\}_{k=1}^N$. The intermediate value $\boldsymbol{\psi}_{k,i}$ at node k is generally a better estimate than $\boldsymbol{y}_{k,i-1}$. Therefore, we replace $\boldsymbol{\mathcal{y}}_{i-1}$ by $\boldsymbol{\psi}_i$ in the second step of (28). This step is reminiscent of incremental-type approaches to optimization, which have been widely studied in the literature [20]–[22]. By doing so, and by setting $\eta = \mu^{-1}$, we obtain:

$$\begin{aligned} \boldsymbol{\psi}_{k,i} &= \boldsymbol{y}_{k,i-1} - \mu\widehat{\nabla J_k}(\boldsymbol{y}_{k,i-1}), \\ \boldsymbol{\mathcal{y}}_i &= \mathcal{E}'_{\mathcal{W}\mathcal{Z}}\boldsymbol{\psi}_i, \end{aligned} \tag{29}$$

Algorithm 1: Centralized adaptive solution for solving (2)

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{y}_{k,i-1} - \mu\widehat{\nabla J_k}(\boldsymbol{y}_{k,i-1}), \tag{33a}$$
$$\boldsymbol{\mathcal{y}}_i = \mathcal{P}_{\mathcal{D}}\boldsymbol{\psi}_i, \tag{33b}$$
$$\boldsymbol{\mathcal{w}}_i = \mathcal{E}_{\mathcal{W}\mathcal{Z}}\boldsymbol{\mathcal{y}}_i. \tag{33c}$$

where

$$\mathcal{E}'_{\mathcal{W}\mathcal{Z}} \triangleq I - \mathcal{P}^\perp_{\mathcal{Z}}(I - \mathcal{E}_{\mathcal{W}\mathcal{Z}}). \tag{30}$$

Using identity (27), we can show that $\mathcal{E}'_{\mathcal{W}\mathcal{Z}} = (\mathcal{E}'_{\mathcal{W}\mathcal{Z}})^2$, $(\mathcal{E}'_{\mathcal{W}\mathcal{Z}})^\top = \mathcal{E}'_{\mathcal{W}\mathcal{Z}}$, $\mathcal{E}'_{\mathcal{W}\mathcal{Z}}\mathcal{W} = \mathcal{W}$, and $\mathcal{E}'_{\mathcal{W}\mathcal{Z}}\mathcal{Z} = \mathcal{Z}$. Now, using the fact that

$$\mathcal{E}_{\mathcal{Z}\mathcal{W}} = \mathcal{P}_{\mathcal{Z}}\left(I - \mathcal{W}(\mathcal{W}^\top\mathcal{P}^\perp_{\mathcal{Z}}\mathcal{W})^{-1}\mathcal{W}^\top P^\perp_{\mathcal{Z}}\right) \tag{31}$$

we can further show that $\mathcal{E}'_{\mathcal{W}\mathcal{Z}} = \mathcal{E}_{\mathcal{W}\mathcal{Z}} + \mathcal{E}_{\mathcal{Z}\mathcal{W}} = \mathcal{P}_{\mathcal{D}}$ where $\mathcal{D} = [\mathcal{W}\ \mathcal{Z}]$. Finally, using (23), we obtain the estimate $\boldsymbol{\mathcal{w}}_i$ of the vector $\mathcal{w}^o = \mathcal{W}\mathcal{x}^o_{\mathcal{W}}$ at iteration i:

$$\boldsymbol{\mathcal{w}}_i = \mathcal{W}(\mathcal{W}^\top\mathcal{P}^\perp_{\mathcal{Z}}\mathcal{W})^{-1}\mathcal{W}^\top\mathcal{P}^\perp_{\mathcal{Z}}\boldsymbol{\mathcal{y}}_i = \mathcal{E}_{\mathcal{W}\mathcal{Z}}\boldsymbol{\mathcal{y}}_i. \tag{32}$$

By combining (29) and (32), we arrive at Algorithm 1.

III-B. Distributed adaptive solution

Although step (33a) is decentralized, the projection steps (33b) and (33c) in (33) require a fusion center. To handle the orthogonal projection, we follow similar arguments as in [5], [6] and replace the $M \times M$ matrix $\mathcal{P}_{\mathcal{D}}$ in (33b) by an $M \times M$ matrix $\mathcal{C}$ that satisfies the following conditions:

$$\lim_{i\to\infty} \mathcal{C}^i = \mathcal{P}_{\mathcal{D}}, \tag{34}$$
$$C_{k\ell} = [\mathcal{C}]_{k\ell} = 0, \quad \text{if } \ell \notin \mathcal{N}_k \text{ and } k \neq \ell. \tag{35}$$

with $C_{k\ell}$ denoting the (k, ℓ)-th block of $\mathcal{C}$ of size $M_k \times M_\ell$. Doing so, we obtain the following distributed adaptive solution at each agent k [5], [6]:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{y}_{k,i-1} - \mu\widehat{\nabla J_k}(\boldsymbol{y}_{k,i-1}), \\ \boldsymbol{y}_{k,i} = \sum_{\ell\in\mathcal{N}_k} C_{k\ell}\boldsymbol{\psi}_{\ell,i}, \end{cases} \tag{36}$$

where $\boldsymbol{y}_{k,i}$ is the estimate of $y^o_k = w^o_k + z^o_k$ at agent k and iteration i. It was shown in [5, Lemma 1] that the matrix equation (34) holds, if and only if, the following three conditions are satisfied:

$$\mathcal{C}\mathcal{P}_{\mathcal{D}} = \mathcal{P}_{\mathcal{D}}, \quad \mathcal{P}_{\mathcal{D}}\mathcal{C} = \mathcal{P}_{\mathcal{D}}, \quad \rho(\mathcal{C} - \mathcal{P}_{\mathcal{D}}) < 1 \tag{37}$$

To handle the oblique projection, we replace the $M \times M$ matrix $\mathcal{E}_{\mathcal{W}\mathcal{Z}}$ in (33c) by an $M \times M$ matrix $\mathcal{A}^S$, where $\mathcal{A}$ satisfies the following conditions:

$$\lim_{i\to\infty} \mathcal{A}^i = \mathcal{E}_{\mathcal{W}\mathcal{Z}}, \tag{38}$$
$$A_{k\ell} = [\mathcal{A}]_{k\ell} = 0, \quad \text{if } \ell \notin \mathcal{N}_k \text{ and } k \neq \ell, \tag{39}$$

with S a positive integer denoting the number of hops. Doing so and using the fact that $\mathcal{A}^S\boldsymbol{\psi}_i$ can be implemented in S communication steps, step (33c) can be replaced by the following multi-hop step at agent k:

$$\begin{cases} \boldsymbol{w}^{(s)}_{k,i} = \sum_{\ell\in\mathcal{N}_k} A_{k\ell}\boldsymbol{w}^{(s-1)}_{\ell,i}, \quad s = 1, \ldots, S, \\ \boldsymbol{w}_{k,i} = \boldsymbol{w}^{(S)}_{k,i}, \end{cases} \tag{40}$$

with $\boldsymbol{w}^{(0)}_{k,i} = \boldsymbol{y}_{k,i}$. To avoid the multi-hop step (40), we propose to replace step (33c) by the following smoothing step:

$$\boldsymbol{\mathcal{w}}_i = (1 - \nu)\mathcal{A}\boldsymbol{\mathcal{w}}_{i-1} + \nu\mathcal{A}\boldsymbol{\mathcal{y}}_i, \tag{41}$$

Algorithm 2: Oblique diffusion algorithm

$$\boldsymbol{\psi}_{k,i} = \boldsymbol{y}_{k,i-1} - \mu \widehat{\nabla J_k}(\boldsymbol{y}_{k,i-1}), \quad (42a)$$

$$\boldsymbol{y}_{k,i} = \sum_{\ell \in \mathcal{N}_k} C_{k\ell} \boldsymbol{\psi}_{\ell,i}, \quad (42b)$$

$$\boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} A_{k\ell}((1-\nu)\boldsymbol{w}_{\ell,i-1} + \nu\, \boldsymbol{y}_{\ell,i}). \quad (42c)$$

where $0 < \nu \ll 1$ is a forgetting factor. By combining (36) and (41), we arrive at the distributed Algorithm 2, which allows for significant communication savings when compared with the multi-hop implementation (40). By building upon the findings of [5], [6], we analyze in the following section the performance of Alg. 2. The first step (42a) corresponds to the stochastic gradient step and results in $\boldsymbol{\psi}_{k,i}$, an intermediate estimate of y_k^o at iteration i. This step is followed by the combination step (42b), where node k combines the intermediate estimates $\{\boldsymbol{\psi}_{\ell,i}\}$ from its neighbors using the combination blocks $C_{k\ell}$. The result of this step is $\boldsymbol{y}_{k,i}$, an estimate of y_k^o at iteration i. In the third step, node k combines the intermediate estimates $\{\boldsymbol{y}_{\ell,i}\}$ and the previous estimates $\{\boldsymbol{w}_{\ell,i-1}\}$ from its neighbors according to (42c). The result of this step is $\boldsymbol{w}_{k,i}$, an estimate of w_k^o at iteration i.

III-C. Performance results

Observe that the evolution of $\boldsymbol{\psi}_i = \text{col}\{\boldsymbol{\psi}_{k,i}\}_{k=1}^N$ and $\boldsymbol{\mathcal{y}}_i = \text{col}\{\boldsymbol{y}_{k,i}\}_{k=1}^N$ in Alg. 2 is independent of the evolution of $\boldsymbol{\mathcal{w}}_i = \text{col}\{\boldsymbol{w}_{k,i}\}_{k=1}^N$. We already know from [5, Theorem 1] and [6, Appendix H] that, under some assumptions on the cost functions and gradient noises, and after sufficient time, the iterates $\boldsymbol{y}_{k,i}$ generated by (42a), (42b) converge to the true models y_k^o in the mean and in the mean-square-error sense according to:

$$\limsup_{i\to\infty} \|\mathbb{E}(y_k^o - \boldsymbol{y}_{k,i})\| = O(\mu), \qquad k = 1, \ldots, N. \quad (43)$$

$$\limsup_{i\to\infty} \mathbb{E}\|y_k^o - \boldsymbol{y}_{k,i}\|^2 = O(\mu), \qquad k = 1, \ldots, N, \quad (44)$$

for small enough μ. To study the convergence w.r.t. w_k^o, we study the smoothing step (41), which can be re-written as:

$$\boldsymbol{\mathcal{w}}_i = (1-\nu)^i \mathcal{A}^i \boldsymbol{\mathcal{w}}_0 + \nu \sum_{j=0}^{i-1} (1-\nu)^j \mathcal{A}^{j+1} \boldsymbol{\mathcal{y}}_{i-j} \quad (45)$$

After sufficient iterations, the influence of the initial condition in (45) can be ignored and we approximate $\boldsymbol{\mathcal{w}}_i$ by the geometric series:

$$\lim_{i\to\infty} \mathbb{E}\boldsymbol{\mathcal{w}}_i \approx \nu \sum_{j=0}^{\infty} (1-\nu)^j \mathcal{A}^{j+1} \mathcal{y}_\infty = \nu\mathcal{A}\left(\sum_{j=0}^{\infty} ((1-\nu)\mathcal{A})^j\right) \mathcal{y}_\infty, \quad (46)$$

where $\mathcal{y}_\infty = \mathcal{y}^o + O(\mu)$ and $\mathcal{y}^o = \text{col}\{y_k^o\}_{k=1}^N$. From Lemma 1, and using similar arguments as in [5, Appendix C], we can re-write the $M \times M$ combination matrix $\mathcal{A}$ in the following Jordan canonical decomposition form:

$$\mathcal{A} = \mathcal{V}_\epsilon \Lambda_\epsilon \mathcal{V}_\epsilon^{-1} \quad (47)$$

where $\mathcal{V}_\epsilon = [\mathcal{W}\ \mathcal{V}_{R,\epsilon}]$, $\Lambda_\epsilon = \text{diag}\{I_P, \mathcal{J}_\epsilon\}$, and $\mathcal{V}_\epsilon^{-1} = \text{col}\{\mathcal{W}^\top \mathcal{E}_{\mathcal{W}\mathcal{Z}}, \mathcal{V}_{L,\epsilon}^\top\}$. The matrix $\mathcal{J}_\epsilon$ consists of Jordan blocks with $\epsilon > 0$ any small number and where the eigenvalue λ may be complex but has magnitude less than one. Since $(1-\nu)\mathcal{A}$ is stable (i.e., $\rho((1-\nu)\mathcal{A}) < 1$), we obtain:

$$\sum_{j=0}^{\infty} ((1-\nu)\mathcal{A})^j = (I - (1-\nu)\mathcal{A})^{-1} \quad (48)$$

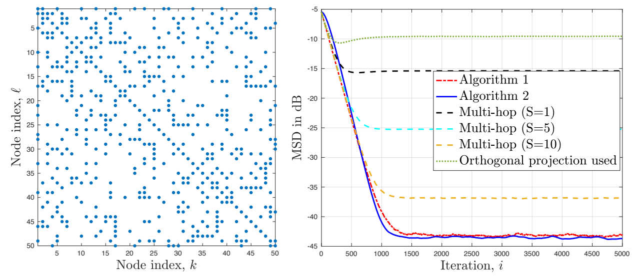

Fig. 2. Inference in the presence of low-rank interference. *(Left)* Link matrix. *(Right)* Performance of Algorithms 1, 2, and the multi-hop strategy (40).

By replacing (47) into (48), we can write:

$$\nu\mathcal{A}(I - (1-\nu)\mathcal{A})^{-1} = \mathcal{V}_\epsilon \begin{bmatrix} I_P & 0 \\ 0 & \nu\mathcal{J}_\epsilon(I - \mathcal{J}_\epsilon + \nu\mathcal{J}_\epsilon)^{-1} \end{bmatrix} \mathcal{V}_\epsilon^{-1} \quad (49)$$

For $\nu \ll 1$, the above matrix (49) becomes approximately equal to $\mathcal{W}\mathcal{W}^\top \mathcal{E}_{\mathcal{W}\mathcal{Z}} = \mathcal{E}_{\mathcal{W}\mathcal{Z}}$ and, thus, $\lim_{i\to\infty} \mathbb{E}\boldsymbol{\mathcal{w}}_i \approx \mathcal{E}_{\mathcal{W}\mathcal{Z}}\mathcal{y}^o + O(\mu)$.

IV. SIMULATION RESULTS

In this section, we consider a mean-square-error (MSE) network with $N = 50$ nodes and $M_k = 5$ $\forall k$, generated randomly with the link matrix shown in Fig. 2 (left). Each agent is subjected to streaming data $\{\boldsymbol{d}_k(i), \boldsymbol{u}_{k,i}\}$ assumed to satisfy the linear model:

$$\boldsymbol{d}_k(i) = \boldsymbol{u}_{k,i}^\top (w_k^o + z_k^o) + \boldsymbol{v}_k(i), \quad (50)$$

for some unknown $M_k \times 1$ vector w_k^o to be estimated by agent k with $\boldsymbol{v}_k(i)$ denoting a zero-mean measurement noise. The vector $\mathcal{w}^o = \text{col}\{w_1^o, \ldots, w_N^o\}$ is assumed to lie in a low-dimensional subspace $\mathcal{R}(\mathcal{W})$. On the other hand, $\mathcal{z}^o = \text{col}\{z_1^o, \ldots, z_N^o\}$ is assumed to lie in a second low-dimensional subspace $\mathcal{R}(\mathcal{Z})$. The matrices $\mathcal{W}$ and $\mathcal{Z}$ are generated according to $\mathcal{W} = W \otimes I_5$ and $\mathcal{Z} = z \otimes I_5$, respectively, with W an $N \times 2$ randomly generated semi-orthogonal matrix ($W^\top W = I_2$) and z an $N \times 1$ randomly generated unit vector ($z^\top z = 1$). The vectors $\mathcal{w}^o = \text{col}\{w_k^o\}_{k=1}^N$ and $\mathcal{z}^o = \text{col}\{z_k^o\}_{k=1}^N$ are generated according to $\mathcal{w}^o = \mathcal{W}x_{\mathcal{W}}^o$ and $\mathcal{z}^o = \mathcal{Z}x_{\mathcal{Z}}^o$, where the vectors $x_{\mathcal{W}}^o$ and $x_{\mathcal{Z}}^o$ are randomly generated from the Gaussian distributions $\mathcal{N}(0.1 \times \mathbb{1}_{2M_k}, I_{2M_k})$ and $\mathcal{N}(0.1 \times \mathbb{1}_{M_k}, I_{M_k})$, respectively. The processes $\{\boldsymbol{u}_{k,i}, \boldsymbol{v}_k(i)\}$ are zero-mean jointly wide-sense stationary with: i) $\mathbb{E}\boldsymbol{u}_{k,i}\boldsymbol{u}_{\ell,i}^\top = R_{u,k} = \sigma_{u,k}^2 I_5 > 0$ if $k = \ell$ and zero otherwise; ii) $\mathbb{E}\boldsymbol{v}_k(i)\boldsymbol{v}_\ell(i) = \sigma_{v,k}^2$ if $k = \ell$ and zero otherwise; and iii) $\boldsymbol{u}_{k,i}$ and $\boldsymbol{v}_\ell(j)$ are independent for all k, ℓ, i, j. The variances $\sigma_{u,k}^2$ and $\sigma_{v,k}^2$ are generated from the uniform distributions $\text{unif}(1, 4)$ and $\text{unif}(0.1, 0.4)$, respectively. For MSE networks [3], the risk function is of the form $J_k(y_k) \triangleq \frac{1}{2}\mathbb{E}|\boldsymbol{d}_k(i) - \boldsymbol{u}_{k,i}^\top y_k|^2$, where $y_k = w_k + z_k$. Since the inference problem described in this section can be written in the form (2), we apply strategy (42) to solve it. We set $\mu = \nu = 0.005$. The matrices $\mathcal{A}$ and $\mathcal{C}$ are set as the solutions of properly formulated SDPs (solved using the CVX package [18]) guaranteeing the sparsity and the convergence conditions (34), (35), (38), and (39). We report the network MSD learning curves $\frac{1}{N}\mathbb{E}\|\mathcal{w}^o - \boldsymbol{\mathcal{w}}_i\|^2$ in Fig. 2 (right). The results are averaged over 200 Monte-Carlo runs. The learning curve of the centralized solution (33) is also reported ($\mu = 0.0018$ in this case to obtain similar convergence rate as the distributed solution). We report also the learning curves of the multi-hop strategy obtained from (42) by replacing the smoothing step (42c) by the multi-hop step (40) when $S = \{1, 5, 10\}$. The results show that strategy (42) performs well compared with the centralized one (33) without the need to perform S communication steps at each iteration. Finally, to illustrate the importance of the oblique projection, we simulate the case where the low-rank interference problem is treated through the orthogonal projection onto $\mathcal{R}(\mathcal{W})$ (i.e., $\boldsymbol{w}_{k,i} = \boldsymbol{y}_{k,i}$).

V. REFERENCES

[1] S. Barbarossa, G. Scutari, and T. Battisti, "Distributed signal subspace projection algorithms with maximum convergence rate for sensor networks with topological constraints," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Taipei, Taiwan, Apr. 2009, pp. 2893–2896.

[2] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[3] A. H. Sayed, S. Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 155–171, 2013.

[4] R. Nassif, S. Vlaski, C. Richard, J. Chen, and A. H. Sayed, "Multitask learning over graphs: An approach for distributed, streaming machine learning," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 14–25, 2020.

[5] R. Nassif, S. Vlaski, and A. H. Sayed, "Adaptation and learning over networks under subspace constraints–Part I: Stability analysis," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1346–1360, 2020.

[6] R. Nassif, S. Vlaski, and A. H. Sayed, "Adaptation and learning over networks under subspace constraints–Part II: Performance analysis," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2948–2962, 2020.

[7] P. Di Lorenzo, S. Barbarossa, and S. Sardellitti, "Distributed signal processing and optimization based on in-network subspace projections," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2061–2076, 2020.

[8] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proceedings of the IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

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

[10] P. Braca, S. Marano, and V. Matta, "Enforcing consensus while monitoring the environment in wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3375–3380, Jul. 2008.

[11] S. Chouvardas, K. Slavakis, and S. Theodoridis, "Adaptive robust distributed learning in diffusion sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4692–4707, Oct. 2011.

[12] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, "Distributed optimization with local domains: Applications in MPC and network flows," *IEEE Transactions on Automatic Control*, vol. 60, no. 7, pp. 2004–2009, Jul. 2015.

[13] A. Koppel, B. M. Sadler, and A. Ribeiro, "Proximity without consensus in online multi-agent optimization," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Shanghai, China, May 2016, pp. 3726–3730.

[14] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[15] R. T. Behrens and L. L. Scharf, "Signal processing applications of oblique projection operators," *IEEE Transactions on Signal Processing*, vol. 42, no. 6, pp. 1413–1424, 1994.

[16] A. H. Sayed and T. Kailath, "Oblique state-space estimation algorithms," in *Proc. of 1995 American Control Conference*, Seattle, WA, USA, Jun. 1995, vol. 3, pp. 1969–1973.

[17] T. Kailath, *Linear systems*, vol. 156, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[18] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[19] R. Nassif, S. Vlaski, and A. H. Sayed, "Distributed inference over networks under subspace constraints," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, Brighton, UK, May 2019, pp. 5232–5236.

[20] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM J. Optim.*, vol. 7, no. 4, pp. 913–926, 1997.

[21] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Process.*, vol. 55, no. 8, pp. 4064–4077, Aug. 2007.

[22] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 798–808, 2005.