Resource Tradeoffs in Distributed Subspace Tracking over the Wireless Medium

Matthew Nokleby* and Waheed U. Bajwa†
*Dept. of Electrical and Computer Engineering, Duke University, Durham, NC (matthew.nokleby@duke.edu)
†Dept. of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ (waheed.bajwa@rutgers.edu)

***Abstract*—This paper studies distributed subspace tracking in wireless networks based on consensus averaging. Most prior approaches to this require the exchange of many inter-node messages between the arrival of new measurements, forcing communications to happen at a faster timescale than measurements. By contrast, this paper proposes a technique, termed *hierarchical subspace tracking*, which leverages recent advances in consensus over wireless networks in order to track subspaces when communications and measurements occur on the same timescale. It is shown that the resource consumption of hierarchical subspace tracking scales slowly in the size of the network. Further, it is shown that the convergence speed of hierarchical subspace tracking is the same as if measurements were known globally.**

I. Introduction

The subspace signal model, which assumes that the signals of interest lie in a low-dimensional subspace, has long been the mainstay of signal processing. In particular, many advances in denoising, interference cancellation, data compression, array processing, etc., can be attributed to this model. An important challenge in this regard is the estimation of *signal subspace* from noisy measurements. Given a collection of signals $\{s_t \in \mathbb{C}^N\}_{t\in\mathbb{Z}_+}$ drawn independently from a zero-mean distribution with covariance $C \in \mathbb{C}^{N\times N}$, this translates into estimating the subspace spanned by the $K \ll N$ eigenvectors associated with the K largest eigenvalues of C.

In this paper, we focus on the problem of *tracking* a signal subspace when measurements *stream* into a *distributed* network of sensors, agents, nodes, etc. For the sake of this exposition, we focus on the relatively simple problem of tracking the one-dimensional subspace given by the principal eigenvector of the covariance. Each node observes one of the entries of s_t at time t and maintains a running estimate of the principal eigenvector (or its "flipped" version) within the network using local communications. While distributed estimation of eigenvectors has been studied in the literature in different contexts [1]–[4], the problem formulation in this paper is most closely related to that of [3], [4]. In particular, [4] proposes and analyzes a solution for online, distributed estimation of the subspace spanned by the principal eigenvector that leverages the well-known Oja's learning rule [5] for subspace tracking and distributed average consensus for data fusion. A major limitation of this work is its reliance on gossip algorithms for consensus averaging, which suffer from slow convergence in common network topologies. In order for the proposed method in [4] to converge to the true subspace in a random geometric topology with N nodes, for instance, the network must be capable of making roughly $O(N^2)$ transmissions between any two measurements. Stated differently, distributed subspace tracking in [4] implicitly relies on *two timescales* in the network: measurements arrive on one (coarser) timescale and communications between nodes take place on another (finer) timescale. Returning to the case of a random geometric network, the estimate in [4] converges only if the communication timescale is much finer than the measurement timescale; otherwise, it diverges.

In contrast to [4] and other related works, we propose and analyze an approach to distributed subspace tracking in which measurements arrive and communications take place on a *single* timescale. The proposed approach leverages *hierarchical averaging*—a novel approach to consensus averaging that exploits the broadcast and superposition natures of the wireless medium to accelerate consensus [6]—for data fusion. In order to facilitate subspace tracking in this case, we tailor Oja's rule to hierarchical averaging, resulting in a distributed scheme in which transmit power and bandwidth need only scale as $O(\log N)$ in a grid network of N nodes for reliable tracking. Furthermore, we show that our proposed scheme is optimal in the sense that the estimate of the subspace converges to the true subspace at a rate that matches the one achieved by the canonical, centralized Oja's rule.

II. Preliminaries

A. System Model and Problem Formulation

Consider a grid network of N nodes, $\mathcal{N} = \{1, \ldots, N\}$, within a unit square in which nodes are geographically distributed on a rectangular grid of size $\sqrt{N} \times \sqrt{N}$. We assume that time and bandwidth in the network are slotted, $t = 1, 2, \ldots$, $b = 1, 2, \ldots$, and each node transmits/receives messages within one of these time–bandwidth slots. We use $P_n(t, b)$ to denote the transmit power of node n at time t and in band b, and assume that node n successfully communicates a single, infinite-precision scalar[1] to node m provided: (i) node m lies within the transmit radius of node n, as defined by the free-space path-loss model: $r_n(t, b) = \gamma \cdot P_n^2(t, b)$, where $\gamma > 0$ is a constant determined by the transmit frequency and the receiver characteristics; and (ii) node m *does not* lie within the transmit radius of any other node transmitting at the same time and over the same band.

We assume signals $\{s_t \in \mathbb{C}^N\}_{t\in\mathbb{Z}_+}$ drawn independently from a zero-mean distribution with covariance $C := E[s_t s_t^H]$ stream into the network, with each node n measuring the nth

[1]While quantization errors can be incorporated into our analysis, we assume for the sake of this exposition that the quantization errors are negligible.

component of the signal, $s_t(n)$, at time t. Using v to denote the principal eigenvector of C, the main challenge then is for each node n to maintain an estimate $\hat{v}_t(n)$ of the nth component of v, such that $\hat{v}_t \to \pm v$ as quickly as possible. Mathematically, the *asymptotic, scaled error covariance*

$$K = \lim_{t\to\infty} tE[(v \pm \hat{v}_t)(v \pm \hat{v}_t)^H] \tag{1}$$

can be used as a measure of the rate at which $\hat{v}_t$ converges to $\pm v$. In the context of distributed subspace tracking in a wireless network, however, the rate of convergence alone is not sufficient to characterize the performance of a tracking algorithm. Rather, one also needs the measures of *average bandwidth* and *average transmit power*. Specifically, using $B(t) = |\{b : \exists n, P_n(t,b) > 0\}|$ to denote the total bandwidth utilized by the network at time t, we define the average bandwidth as

$$B_{\text{avg}} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} B(t). \tag{2}$$

Similarly, we define the average transmit power as

$$P_{\text{avg}} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \sum_{b=1}^{B(t)} \sum_{n=1}^{N} P_n(t,b), \tag{3}$$

which represents the average total power used by the network in each time slot. We are then interested in tracking methods that not only have a fast rate of convergence, but that also minimize the metrics of B_{avg} and P_{avg}.

B. Review of Oja's Rule in Distributed Settings

The baseline (centralized) subspace tracking method that we consider for comparison purposes in this paper is the well-known Oja's rule [5], which is known for its simplicity and fast convergence. Given a sequence of signals $\{s_t\}$, Oja's rule maintains an estimate of the subspace spanned by the principal eigenvector via the iterations

$$\hat{v}_{t+1} = \hat{v}_t + \epsilon_t[(s_t s_t^H)\hat{v}_t - \hat{v}_t \hat{v}_t^H s_t s_t^H \hat{v}_t], \tag{4}$$

where $\epsilon_t > 0$ is an appropriately-chosen step size and $\hat{v}_0$ is an arbitrary, unit-norm initial estimate. It can be shown that a proper choice of ϵ_t, along with some mild assumptions on C (such as largest eigenvalue of multiplicity one and $\hat{v}_0$ not being orthogonal to v) guarantee $\hat{v}_t \to \pm v$ with probability 1. Further, the asymptotic, scaled error covariance of Oja's rule (cf. (1)) in this case can be expressed as

$$K_{\text{oja}} = A(I - C)E[s_t s_t^H v v^H s_t s_t^H](I - C)A^H, \tag{5}$$

where A is a constant matrix that we leave unspecified for brevity. We refer interested readers to the literature on stochastic approximation theory in general and [7] in particular for technical details.

In order to implement Oja's rule given by (4) in a distributed setting, [4] observed that each node $n \in \mathcal{N}$ needs only the inner product $s_t^H \hat{v}_t$ to compute the nth component of $\hat{v}_{t+1}$. In addition, since the inner product is a linear operation, [4] observed that it can be computed in a distributed fashion via *consensus averaging*. Among the many approaches to consensus averaging in the literature, [4] proposed the use of *gossip* algorithms [8] and analyzed the performance of the resulting tracking algorithm.

A major drawback of relying on gossip algorithms for consensus averaging is their slow convergence. In grid and random geometric topologies, for instance, they require roughly $O(N^2)$ transmissions in the network for convergence. This slow convergence of gossip algorithms forces [4] to carry out communications between nodes on a timescale that is much finer than the timescale on which measurements arrive within the network. In particular, this imposes the condition in [4] that the signal s_{t+1} arrives in network only after the network has achieved (approximate) consensus on $s_t^H \hat{v}_t$. In the sequel, we show that this restrictive condition can be removed through a modification of Oja's rule and utilization of a newer consensus averaging method, termed *hierarchical averaging*, introduced and analyzed in [6]. Before proceeding further, we briefly review this method.

C. Review of Hierarchical Averaging

Hierarchical averaging [6] is a novel approach to consensus averaging in wireless networks that takes explicit advantage of the broadcast and superposition natures of the wireless medium for faster convergence. Instead of communications between neighbors described through edges in a graph as in traditional approaches to consensus, nodes in hierarchical averaging broadcast messages to other nodes in geographically-defined cells. These cells, as illustrated in Fig. 1, are organized into a hierarchy of $R = \lceil \log_4 N \rceil$ layers. The entire network is a single cell at layer R, the network is divided into four square cells at layer $R-1$, and this partitioning of the cells continues until the network comprises 4^{R-1} cells at layer 1.

In order to illustrate the workings of hierarchical averaging, suppose each node in our grid network has a a single scalar measurement z_n and that we are interested in computing the average $(\sum_{n\in\mathcal{N}} z_n)/N$ at each node. In order for this, hierarchical averaging starts at layer 1 and nodes broadcast their measurements z_n to other nodes in their respective cells. At the end of these broadcasts (i.e., round 1 of communications), each node averages the measurements received from other nodes within its cell. Therefore, all nodes within a cell end up with identical estimates of the network average at the end of round 1. Hierarchical averaging then moves to layer 2, in which a representative node from each layer 1 cell is selected to broadcast its estimate of the network average to all of the nodes within its cell at layer 2. Once again, at the end of these round 2 of communications, each node averages the measurements received from other nodes within its layer 2 cell, resulting in all nodes within every layer 2 cell to have identical estimates. This selection–broadcast–average process continues till layer R, at which point four representative nodes—one from each of the cells at layer $R-1$—broadcast their estimates to the entire network. Every node in the network then computes an average of these final four measurements, which ends up being precisely the average $(\sum_{n\in\mathcal{N}} z_n)/N$. In words, hierarchical averaging carries out consensus averaging by fusing estimates up the hierarchy in R rounds of communications.

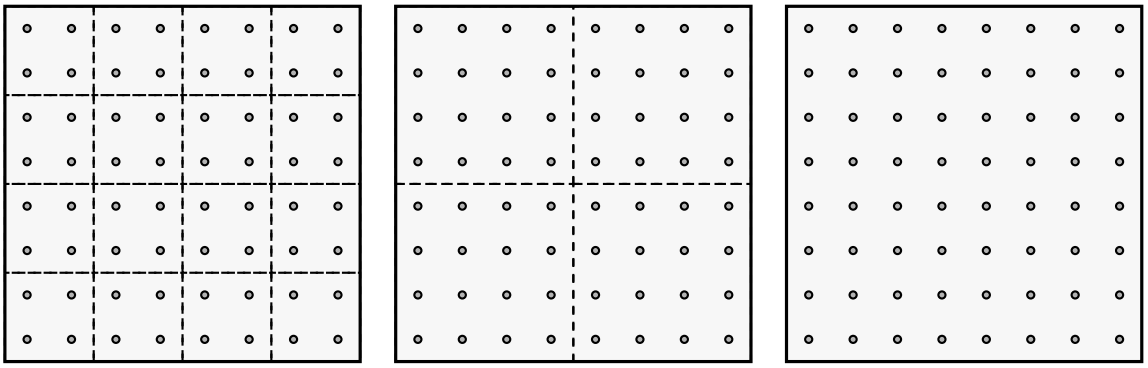

Fig. 1. An illustration of hierarchical averaging at three different layers. At layer 1, nodes first broadcast messages that reach other nodes within their cells and then nodes average the received messages. At subsequent layers (up to layer $R = \lceil \log_4 N \rceil$), a representative from each cell in the lower layer broadcasts messages that reach other nodes within their cells in the current layer, followed once again by averaging of received messages by the nodes.

One of the biggest advantages of hierarchical averaging, as established in [6], is its frugal usage of network resources. Specifically, it can compute a single average (equivalently, a single linear operation of initial measurements) by using only $R = \lceil \log_4 N \rceil$ time slots, $O(\log N)$ units of bandwidth, and $O(\log N)$ units of transmit energy. Note that if one were interested in computing x linear operations of network measurements then that can also be accomplished by hierarchical averaging at the expense of an increase in (cumulative) bandwidth and (cumulative) energy, resulting in $R = \lceil \log_4 N \rceil$ time slots, $O(x \log N)$ units of bandwidth, and $O(x \log N)$ units of transmit energy. In the following, we exploit these facts about hierarchical averaging for distributed subspace tracking on a single timescale.

III. Hierarchical Subspace Tracking

While hierarchical averaging enables relatively fast and low-resource consensus averaging, the need for $R = \lceil \log_4(N) \rceil$ time slots to compute a linear functional precludes the use of canonical Oja's rule for single timescale subspace tracking. Instead, we modify the Oja iterations (4) in a way that they accommodate the $R = \lceil \log_4(N) \rceil$ time slots for consensus averaging but still manage to incorporate every signal s_t into the final estimate. Our approach is to run an Oja-like iteration once every R slots, but which is based on an R-point estimate of the covariance formed from the previous R received signals.

Specifically, consider slot t with t/R an integer and define

$$C_t = \frac{1}{R} \sum_{\tau=t-R+1}^{t} s_\tau s_\tau^H, \tag{6}$$

an estimate of the covariance formed from the previous R measurements. In hierarchical subspace tracking, nodes update their estimates according to the following "Oja-like" rule:

$$\hat{v}_{t+R} = \hat{v}_t + \epsilon_t [C_t \hat{v}_t - \hat{v}_t \hat{v}_t^H C_t \hat{v}_t]. \tag{7}$$

In words, the network updates its estimate of v once every R slots, and this update employs a rank-R covariance estimate, rather than the rank-one estimate employed under Oja's rule.

In terms of computing (7) in a distributed setting, notice that each node n can compute $\hat{v}_{t+R}(n)$ from the nth element of $C_t \hat{v}_t$ and the scalar $\hat{v}_t^H C_t \hat{v}_t$. These quantities in turn can be computed from the R inner products $\{s_\tau^H \hat{v}_t\}_{\tau=t-R+1}^{t}$. In hierarchical subspace tracking, the nodes compute these R inner products over R time slots by means of hierarchical averaging. The complete algorithm proceeds as follows.

In time slot $t \in R\mathbb{Z}_+$, each node n computes the R scalars $s_\tau(n)\hat{v}_t(n)$ for each $t - R + 1 \leq \tau \leq t$. Then, the nodes compute the sum of these scalars via R parallel instances of hierarchical averaging as described in Section II-C. That is, over R time slots, the nodes perform distributed computations of the R necessary inner products. Then, in time slot $t + R$, having the necessary quantities, each node computes its component of the new iterate described by (7). Meanwhile, R newer signals arrived in the network during the hierarchical averaging step and the nodes continue this process with these newer signals, which now involves computing the scalars $s_\tau(n)\hat{v}_t(n)$ for $t + 1 \leq \tau \leq t + R$, computing the inner products $\{s_\tau^H \hat{v}_t\}_{\tau=t+1}^{t+R}$ via hierarchical averaging, and updating the network estimate using (7).

To summarize, the network processes R measurements via R parallel instances of hierarchical averaging in R time slots. In this fashion, the network can incorporate all of the measurements in its estimates while operating on a single timescale. The following theorem shows that the resource requirements of this subspace tracking method are rather favorable.

Theorem 1: Hierarchical subspace tracking results in

$$B_{\text{avg}} = O(\log N) = P_{\text{avg}}. \tag{8}$$

Proof: The proof follows straightforwardly from known results on hierarchical averaging [6]. To compute R inner products in parallel, hierarchical averaging requires $O(R \log N)$ units of energy and bandwidth over R slots. Substituting these into the definitions of B_{avg} and P_{avg} yields the claim. ∎

Next, we prove convergence of hierarchical subspace tracking. In particular, we show that $\hat{v}_t$ converges on $\pm v$ with probability 1 and that the convergence speed is identical to that of the centralized, canonical Oja's rule. In other words, our algorithm converges as fast as if we were able to run an Oja's rule iteration at every time slot.

Theorem 2: For hierarchical subspace tracking, $\hat{v}_t \to \pm v$ with probability 1. Furthermore, let K_{h} be the asymptotic, scaled error covariance for hierarchical subspace tracking. Then we have $K_{\text{oja}} = K_{\text{h}}$.

Proof sketch: That $\hat{v}_t \xrightarrow{\text{w.p.1}} v$ is a direct corollary to the

fact that the original Oja's rule converges. Indeed, hierarchical subspace tracking implies only a decrease in the noise on each iteration, therefore convergence is assured with probability 1.

Proving the convergence rate, in the form of K_{h}, requires the framework of stochastic approximation. Since we cannot fully present the framework here, we give only a sketch of the proof; see [7], particularly Ch. 10, for more details. Our sketch focuses on the vector-valued sequence[2]

$$\delta M_t = (C_t\hat{v}_t - \hat{v}_t\hat{v}_t^H C_t\hat{v}_t) - E_{i<t}[(C_t\hat{v}_t - \hat{v}_t\hat{v}_t^H C_t\hat{v}_t)]. \quad (9)$$

In particular, we are interested in the matrix

$$\Sigma = \lim_{t\to\infty} E_{i<t}[\delta M_t \delta M_t^H]. \quad (10)$$

Under proper assumptions (including ones on the step size ϵ_t) [7], it can be shown that $K_{\mathrm{h}} = RA\Sigma A^H$, where A is the same constant matrix as in (5). Here, the factor of R is due to the fact that nodes update their estimates only every R slots.

To specify Σ, we first evaluate the second term of δM_t:

$$E_{i<t}[(C_t\hat{v}_t - \hat{v}_t\hat{v}_t^H C_t\hat{v}_t)] = C\hat{v}_t - \hat{v}_t\hat{v}_t^H C\hat{v}_t. \quad (11)$$

Then, algebraic manipulations yield

$$\delta M_t = (C_t - C)\hat{v}_t - \hat{v}_t\hat{v}_t^H(C_t - C)\hat{v}_t. \quad (12)$$

Now, in the limit $\hat{v}_t = \pm v_t$. Supposing $\hat{v}_t$ converges on v_t (the other case follows similarly), we get

$$\begin{aligned}\Sigma &= E_{i<t}[((C_t - C)v - vv^H(C_t - C)v)\times \\ &\qquad ((C_t - C)v - vv^H(C_t - C)v)^H] \\ &= (I - vv^H)E_{i<t}[C_t vv^H C_t^H](I - vv^H). \end{aligned} \quad (13)$$

Next, we expand $E_{i<t}[C_t vv^H C_t^H]$ to obtain

$$\begin{aligned}E_{i<t}[C_t vv^H C_t^H] &= \frac{1}{R^2}\sum_{m,n=1}^{R} E[s_m s_m^H vv^h s_n s_n^H] \\ &= \frac{R(R-1)}{R^2}vv^H + \frac{1}{R}E[s_n s_n^H vv^H s_n s_n]. \end{aligned} \quad (14)$$

Observing that $(I - vv^H)vv^H = 0$, we finally obtain

$$\Sigma = \frac{1}{R}(I - C)E[s_t s_t^H vv^H s_t s_t^H](I - C), \quad (15)$$

which yields the claim through pre- and post-multiplication of Σ by RA and A^H, respectively. ∎

IV. Numerical Results

To numerically demonstrate the convergence speed of our method, we consider a network of $N = 64$ nodes. We let the signals s_t be zero-mean Gaussian with a random covariance C having unit ℓ_2-norm columns, and we randomly initialize $\hat{v}_0$. We choose the stepsize $\epsilon_t = 0.02$ for every iteration t. In Fig. 2, we show the tracking error $t||\hat{v}_t \pm v||^2$ averaged over 100 initializations as a one-dimensional surrogate for (1). In addition to showing the results for hierarchical subspace tracking, we show the error performance for standard Oja's rule, which requires instant access to the measurements, as well as a naive implementation in which a single, standard Oja's iteration is carried out every R rounds (i.e., $(R-1)$ measurements are skipped between successive iterations).

[2]The operator $E_{i<t}$ denotes expectation conditioned on the initial estimate $\hat{v}_0$ and the previous update terms $C_i\hat{v}_i - \hat{v}_i\hat{v}_i^H C_i\hat{v}_i$, $i < t$, where we slightly abuse the notation and use t for the iteration count, rather than the "slot index."

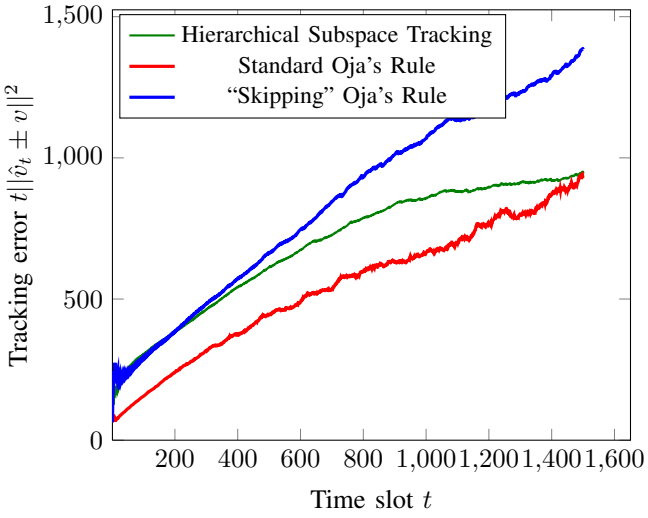

Fig. 2. Distributed tracking error as a function of the time-slot index t.

The results bear out the predictions in Theorem 2. Although the convergence of hierarchical subspace tracking is initially slower than standard Oja, it "catches up" as asymptotic effects begin to dominate. On the other hand, hierarchical subspace tracking offers a large performance increase over discarding intervening measurements.

V. Conclusion

We have presented and analyzed *hierarchical subspace tracking*, a distributed approach to subspace tracking in wireless networks. Unlike previous methods, which require internode communications at a rate much faster than arrival of new measurements, hierarchical subspace tracking succeeds when measurements and communications occur on the same timescale. Hierarchical subspace tracking has excellent convergence speed, with the estimation error decaying as though nodes had complete access to the distributed data, and its resource requirements scale only logarithmically in network size, making it an effective solution to subspace tracking.

References

[1] D. Kempe and F. McSherry, "A decentralized algorithm for spectral analysis," *J. Computer and System Sciences*, pp. 70–83, Feb. 2008.

[2] B. Oreshkin, M. Coates, and M. Rabbat, "Optimization and analysis of distributed averaging with short node memory," *IEEE Trans. Signal Processing*, pp. 2850–2865, May 2010.

[3] L. Li, X. Li, A. Scaglione, and J. Manton, "Decentralized subspace tracking via gossiping," in *Dist. Comp. Sensor Syst.*, 2010, pp. 130–143.

[4] L. Li, A. Scaglione, and J. Manton, "Distributed principal subspace estimation in wireless sensor networks," *IEEE J. Selected Topics Signal Proc.*, pp. 725–738, Aug. 2011.

[5] E. Oja, "Simplified neuron model as a principal component analyzer," *J. Math. Biology*, pp. 267–273, Nov. 1982.

[6] M. Nokleby, W. U. Bajwa, A. R. Calderbank, and B. Aazhang, "Toward resource-optimal consensus over the wireless medium," *IEEE J. Selected Topics Signal Proc.*, pp. 284–295, Apr. 2013.

[7] H. J. Kushner and G. G. Yin, *Stochastic approximation and recursive algorithms and applications*. Springer, 2003.

[8] A. Dimakis, S. Kar, J. Moura, M. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, pp. 1847–1864, Nov. 2010.