Proceedings of the
47th IEEE Conference on Decision and Control
Cancun, Mexico, Dec. 9-11, 2008

Distributed Calculation of Linear Functions in Noisy Networks via Linear Iterations

Shreyas Sundaram and Christoforos N. Hadjicostis

Abstract— **Given a fixed network where each node has some given initial value, and under the constraint that each node receives noisy transmissions from its immediate neighbors, we provide a distributed scheme for any node to calculate an unbiased estimate of an arbitrary linear function of the initial values. Our scheme consists of a linear iteration where, at each time-step, each node updates its value to be a weighted average of its own previous value and those of its neighbors. We show that after repeating this process with almost any set of weights for a finite number of time-steps (upper bounded by the size of the network), any node in the network will be able to calculate an unbiased estimate of any linear function by taking a linear combination of the values that it sees over the course of the linear iteration. For a given set of weights, this linear combination can also be optimized to minimize the variance of the unbiased estimate calculated by each node.**

I. Introduction

In distributed systems and networks, it is often necessary for some or all of the nodes to calculate some function of certain parameters distributed throughout the network. This problem has been studied by the computer science, communication, and control communities over the past few decades, leading to the development of various protocols [1], [2], [3]. Special cases of the distributed function calculation problem include the transmission of data from one or multiple sources to one or multiple sinks, and the *distributed consensus* problem, where all nodes in the network calculate the same function [1].

The notion of consensus has recently experienced a resurgence in the control literature, due to its applicability to diverse topics ranging from cooperative control and multi-agent systems to modeling flocking behavior in biological and physical systems [4]. In these cases, the approach to consensus is to use a linear iteration, where each node in the network repeatedly updates its value to be a weighted linear combination of its own value and those of its neighbors (e.g., see [4] and the references therein). These works have revealed that if the network topology satisfies certain conditions, the weights for the linear iteration can be chosen so that all of the nodes asymptotically converge to the same value. However, the fact that consensus is only reached asymptotically implies that a large number of time-steps (and communication) will be required before all nodes are sufficiently close to the consensus value. Furthermore, it was shown in [5] that if the linear iteration is affected by additive noise at each time-step, the reliance on asymptotic convergence can cause the nodes to be driven arbitrarily far away from the desired consensus value. The problem of consensus via linear iterations with noisy transmissions has only recently started to gain attention (e.g., see [6], [7], [8]), and the existing schemes only allow each node to calculate an unbiased[1] estimate of the consensus value asymptotically (i.e., they do not obtain an unbiased estimate in a finite number of time-steps).

Recently, we showed in [10] (for noise-free networks) that the linear iterative strategy described above can actually be applied to the more general function calculation problem, allowing any node in time-invariant networks to calculate any arbitrary function of the node values in a finite number of time-steps (upper bounded by the size of the network). In this paper, we extend these results to the case where each node only obtains a noisy (or uncertain) measurement of its neighbors' values. While noisy transmissions between nodes in networks can often be handled by utilizing source or channel coding, the model that we consider in this paper applies to situations where coding is not available, or where nodes directly sense the values of their neighbors, and their sensing or measurement capabilities are subject to noise [7]. Our model can also be used as an abstraction for the case where each node can only transmit quantized versions of its values to its neighbors [8]. Using only the first order statistics of the noise, we show that each node can obtain an unbiased estimate of any desired linear function (in strongly connected graphs) as a linear combination of the noisy values it receives from its neighbors, along with its own values; furthermore, this can be done for almost any choice of weights in the linear iteration, and after running the iteration for a finite number of time-steps. If the second order statistics of the noise are also known (perhaps only after running the linear iteration), we show how each node can refine its estimate of the linear function by choosing this linear combination in order to minimize the variance of the estimation error.

In our development, we will use the notation $\mathbf{e}_i$ to indicate the column vector with a 1 in its i–th position and zeros elsewhere. The symbol $\mathbf{1}$ represents the column vector (of

This material is based upon work supported in part by the National Science Foundation under NSF Career Award 0092696 and NSF ITR Award 0426831. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of NSF.

S. Sundaram is with the Coordinated Science Laboratory and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, USA. E-mail: ssundarm@illinois.edu. C. N. Hadjicostis is with the Department of Electrical and Computer Engineering, University of Cyprus, and also with the Coordinated Science Laboratory and the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign. E-mail: chadjic@ucy.ac.cy.

[1]An estimate $\widehat{\Theta}$ of a parameter Θ is said to be *unbiased* if the expected value of $\widehat{\Theta}$ is equal to Θ [9].

appropriate size) that has all entries equal to one. The $n \times n$ identity matrix is denoted by I_n. The notation $\mathbf{A}'$ indicates the transpose of matrix $\mathbf{A}$. We will denote the rank of matrix $\mathbf{A}$ by $\rho(\mathbf{A})$. The expected value of a random parameter A is denoted by $E[A]$.

II. Background

The interaction constraints in distributed systems and networks can be conveniently modeled via a directed graph $\mathcal{G} = \{\mathcal{X}, \mathcal{E}\}$, where $\mathcal{X} = \{x_1, \ldots, x_N\}$ is the set of nodes in the system and $\mathcal{E} \subseteq \mathcal{X} \times \mathcal{X}$ is the set of directed edges (i.e., directed edge $(x_j, x_i) \in \mathcal{E}$ if node x_i can receive information from node x_j). All nodes whose values can be received by node x_i are said to be neighbors of node i, and are represented by the set $\mathcal{N}_i$. The number of neighbors of node i is called the in-degree of node i, and denoted by $\deg_i$.

At each time-step k, nodes can update their values based on some strategy. The scheme that we study in this paper makes use of linear iterations; specifically, at each time-step, each node updates its value as

$$x_i[k+1] = w_{ii}x_i[k] + \sum_{j \in \mathcal{N}_i} w_{ij}x_j[k],$$

where the w_{ij}'s are a set of weights. For ease of analysis, the values of all nodes at time-step k can be aggregated into the value vector $\mathbf{x}[k] = \begin{bmatrix} x_1[k] & x_2[k] & \cdots & x_N[k] \end{bmatrix}'$, so that the update strategy for the entire system can be represented as

$$\mathbf{x}[k+1] = \underbrace{\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NN} \end{bmatrix}}_{\mathbf{W}} \mathbf{x}[k] \qquad (1)$$

for $k = 0, 1, \ldots$, with the constraint that $w_{ij} = 0$ if $j \notin \mathcal{N}_i$. We assume that each node i has some initial value $x_i[0]$ that is potentially required for functions calculated by other nodes in the network.

In [10], it was shown that, for almost any[2] choice of weights, the nodes in the system can calculate an arbitrary function of the other node values after running the linear iteration (1) for a finite number of time-steps (as long as there are paths from the nodes that hold the needed values to the nodes that have to calculate the functions). The analysis in that paper starts by modeling the linear iteration as

$$\begin{aligned} \mathbf{x}[k+1] &= \mathbf{W}\mathbf{x}[k] \\ \mathbf{y}_i[k] &= \mathbf{C}_i\mathbf{x}[k], \quad 1 \le i \le N \end{aligned} , \qquad (2)$$

where $\mathbf{y}_i[k]$ denotes the outputs (node values) that are available to node i during the k–th time-step. Specifically, $\mathbf{C}_i$ is the $(\deg_i + 1) \times N$ matrix with a single 1 in each row denoting the positions of the state-vector $\mathbf{x}[k]$ that are available to node i (i.e., these positions correspond to the nodes that are neighbors of node i, along with node i itself). Since $\mathbf{x}[k] = \mathbf{W}^k\mathbf{x}[0]$, the set of all outputs seen by node i over $L+1$ time-steps is given by

$$\underbrace{\begin{bmatrix} \mathbf{y}_i[0] \\ \mathbf{y}_i[1] \\ \mathbf{y}_i[2] \\ \vdots \\ \mathbf{y}_i[L] \end{bmatrix}}_{\mathbf{y}_i[0:L]} = \underbrace{\begin{bmatrix} \mathbf{C}_i \\ \mathbf{C}_i\mathbf{W} \\ \mathbf{C}_i\mathbf{W}^2 \\ \vdots \\ \mathbf{C}_i\mathbf{W}^L \end{bmatrix}}_{\mathcal{O}_{i,L}} \mathbf{x}[0] \ . \qquad (3)$$

When $L = N-1$, the matrix $\mathcal{O}_{i,L}$ in the above equation is the *observability matrix* for the pair $(\mathbf{W}, \mathbf{C}_i)$ [11]. The row space of $\mathcal{O}_{i,L}$ characterizes the set of all linear functions[3] of $\mathbf{x}[0]$ that can be calculated by node i up to time-step L. Specifically, if the row space of the observability matrix $\mathcal{O}_{i,L}$ contains a vector $\mathbf{c}'$, then one can find a matrix Γ_i such that $\Gamma_i\mathcal{O}_{i,L} = \mathbf{c}'$. Thus, after running the linear iteration (1) for $L+1$ time-steps, node i can immediately calculate the linear function $\mathbf{c}'\mathbf{x}[0]$ as a linear combination of the outputs of the system over those time steps, i.e.,

$$\Gamma_i\mathbf{y}_i[0:L] = \Gamma_i\mathcal{O}_{i,L}\mathbf{x}[0] = \mathbf{c}'\mathbf{x}[0] \ . \qquad (4)$$

If $\rho(\mathcal{O}_{i,L}) = N$, the pair $(\mathbf{W}, \mathbf{C}_i)$ is said to be *observable*. In this case, node i can determine the entire initial value vector $\mathbf{x}[0]$ from the outputs of the system, and can therefore calculate any function of those values.

An important feature of the observability matrix is that there exists an integer ν_i such that the rank of the matrix $\mathcal{O}_{i,L}$ monotonically increases with L until $L = \nu_i - 1$, at which point it stops increasing. This means that the outputs of the system $\mathbf{y}_i[0], \mathbf{y}_i[1], \ldots, \mathbf{y}_i[\nu_i - 1]$ contain the maximum amount of information that is possible to obtain about the initial state, and future outputs of the system do not provide any extra information to node i. The integer ν_i is upper bounded as $\nu_i \le N - \deg_i$ [10], which implies that if it is possible for node i to calculate the desired function $f(x_1[0], x_2[0], \ldots, x_N[0])$, it can do so in at most $N - \deg_i$ time-steps.

The following theorem from [10] indicates that, for almost any choice of weight matrix, the observability matrix for each node i will allow node i to obtain the initial value of all nodes that have a path in the network to node i.

Theorem 1 ([10]): Let $\mathcal{G}$ denote the graph of the network. Define the set $R_i = \{x_j \mid \text{There exists a path from } x_j \text{ to } x_i \text{ in } \mathcal{G}\} \cup \{x_i\}$. Then, for almost any choice of weight matrix $\mathbf{W}$, node i can obtain the value $x_j[0]$, $x_j \in R_i$, after running the linear iteration (1) for $L_i + 1$ time-steps, for some $0 \le L_i < |R_i| - \deg_i$; node i can therefore calculate any arbitrary function of the values $\{x_j[0] \mid x_j \in R_i\}$.

In the above theorem, the phrase "almost any" indicates that the set of parameters for which the theorem does not hold has Lebesgue measure zero [10]. As discussed in

[2]As we will explain in more detail later, the phrase "almost any" in this context means that the set of weights for which the property is violated has Lebesgue measure zero.

[3]A function $f(x_1[0], x_2[0], \ldots, x_N[0])$ of the initial values is *linear* if it is of the form $Qx[0]$ for some matrix Q.

[10], the weights can be chosen (almost arbitrarily) by a centralized entity and provided to the nodes[4] *a priori*, or they can be chosen independently by each node and discovered by the network after following a simple distributed protocol.

Remark 1: Note that unlike asymptotic consensus schemes, where $\mathbf{x}[k]$ converges to a constant vector after an infinite number of time-steps, the protocol described above *does not* require $\mathbf{x}[k]$ to converge to any particular vector (or even to converge at all).

III. The Noise Model

In the rest of the paper, we will extend the above results to the case where the system is operating in the presence of noise. We will first introduce the noise model, and then show that each node in the system can use a modified version of the above techniques to obtain an unbiased estimate of any desired linear function[5] in a finite number of time-steps.

Consider the linear iteration in (2). To simplify the development, we will assume without loss of generality that the rows of $\mathbf{C}_i$ are ordered such that the first row of each $\mathbf{C}_i$ corresponds to node i's own value in the state vector $\mathbf{x}[k]$ (i.e., the i–th element of the first row of $\mathbf{C}_i$ is 1, and all other entries in that row are zero). Suppose that the values that each node i receives (or senses) from its neighbors are corrupted by noise (i.e., there is a noise component associated with each exchange of values between two neighboring nodes). Let $\mathbf{n}_i[k]$ denote the $\deg_i \times 1$ vector containing the noise that affects the values received by node i at time-step k. For each $1 \le i \le N$, let $\widehat{\mathbf{D}}_i$ denote the $(\deg_i + 1) \times (\deg_i)$ matrix given by $\widehat{\mathbf{D}}_i = \begin{bmatrix} 0 \\ I_{\deg_i} \end{bmatrix}$; i.e., the first row of $\widehat{\mathbf{D}}_i$ has all entries equal to zero, and the remaining rows form the $\deg_i \times \deg_i$ identity matrix. The noisy values that node i receives at time-step k are then given by $\mathbf{y}_i[k] = \mathbf{C}_i\mathbf{x}[k] + \widehat{\mathbf{D}}_i\mathbf{n}_i[k]$. Note that the reason for setting the top row of $\widehat{\mathbf{D}}_i$ equal to zero is to model the fact that node i has noise-free access to its own value. If we define

$$\begin{aligned}\mathbf{n}[k] &\equiv \begin{bmatrix}\mathbf{n}_1'[k] & \mathbf{n}_2'[k] & \cdots & \mathbf{n}_N'[k]\end{bmatrix}' ,\\ \mathbf{D}_i &\equiv \begin{bmatrix}0 & \cdots & 0 & \widehat{\mathbf{D}}_i & 0 & \cdots & 0\end{bmatrix} ,\end{aligned}$$

where each $\mathbf{D}_i$ matrix has $\sum_{j=1}^{i-1}\deg_j$ columns of zeros, followed by the matrix $\widehat{\mathbf{D}}_i$, followed by $\sum_{j=i+1}^{N}\deg_j$ columns of zeros, the output seen by node i then becomes $\mathbf{y}_i[k] = \mathbf{C}_i\mathbf{x}[k] + \mathbf{D}_i\mathbf{n}[k]$.

Now consider the update equation. Recall that each node uses the values that it receives from its neighbors to update its own value. In particular, node i multiplies the value that it receives from node j by the weight w_{ij}. Let $\bar{\mathbf{w}}_i$ denote the $1 \times \deg_i$ vector that contains the weights corresponding to the neighbors of node i. The update for node i is then given by

$$\begin{aligned}x_i[k+1] &= \begin{bmatrix}w_{ii} & \bar{\mathbf{w}}_i\end{bmatrix}\mathbf{y}_i[k]\\ &= \begin{bmatrix}w_{ii} & \bar{\mathbf{w}}_i\end{bmatrix}(\mathbf{C}_i\mathbf{x}[k] + \mathbf{D}_i\mathbf{n}[k])\\ &= w_{ii}x_i[k] + \sum_{j\in\mathcal{N}_i} w_{ij}x_j[k]\\ &\quad + \begin{bmatrix}0 & \cdots & 0 & \bar{\mathbf{w}}_i & 0 & \cdots & 0\end{bmatrix}\mathbf{n}[k] .\end{aligned}$$

Defining the matrix

$$\mathbf{B} \equiv \begin{bmatrix}\bar{\mathbf{w}}_1 & 0 & \cdots & 0\\ 0 & \bar{\mathbf{w}}_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \bar{\mathbf{w}}_N\end{bmatrix} , \tag{5}$$

one obtains the noise model

$$\begin{aligned}\mathbf{x}[k+1] &= \mathbf{W}\mathbf{x}[k] + \mathbf{B}\mathbf{n}[k]\\ \mathbf{y}_i[k] &= \mathbf{C}_i\mathbf{x}[k] + \mathbf{D}_i\mathbf{n}[k],\ \ 1 \le i \le N .\end{aligned} \tag{6}$$

Note that the noise vector $\mathbf{n}[k]$ in this case has dimension $\left(\sum_{j=1}^{N}\deg_j\right) \times 1$ (since $\sum_{j=1}^{N}\deg_j$ is equal to the number of edges in the graph [12], and since each edge corresponds to a noisy transmission, one requires a noise term at each time-step for every edge in the graph).

Remark 2: Note that the noise model in (6) can also handle the case where noise only affects the update equation for each node (e.g., due to quantization), but does not affect the exchange of values between nodes, simply by setting $\mathbf{B}$ to be the $N \times N$ identity matrix, and choosing each $\mathbf{D}_i$ to be the zero matrix. Note that in this case, the noise vector $\mathbf{n}[k]$ will only have N components.

In [5], the authors considered the problem of asymptotic consensus in the presence of update noise via a linear iteration of the form $\mathbf{x}[k+1] = \mathbf{W}\mathbf{x}[k] + \mathbf{n}[k]$, where $\mathbf{n}[k]$ is zero mean white noise with covariance matrix $E[\mathbf{n}[k]\mathbf{n}'[k]] = I$, and $\mathbf{W}$ is a symmetric matrix providing asymptotic consensus, i.e., $\lim_{k\to\infty} x_i[k] = \frac{1}{N}\mathbf{1}'\mathbf{x}[0]$ for all i. They showed that as $k \to \infty$, the variance of the node values $x_i[k]$ from the value $\frac{1}{N}\mathbf{1}'\mathbf{x}[0]$ increases without bound. This phenomenon is essentially due to the fact that any weight matrix that provides asymptotic consensus must necessarily have a (marginally stable) eigenvalue at 1 [13], and thus the components of the noise that excite this mode of the system will accumulate and cause the values of the nodes to evolve according to a random walk.

Recent work has focused on addressing this issue in various ways [8], [7], [6], but unlike our approach here, all of these works focus on obtaining convergence in an asymptotic number of time-steps. In the next section, we show that for almost any choice of weight matrix $\mathbf{W}$, each node can obtain an unbiased estimate of any linear function of $\mathbf{x}[0]$ after running the linear iteration for a finite number of time-steps. Furthermore, if the second order statistics of the noise are known, we show how each node can minimize the variance of its estimate of the function (for a given choice of weight matrix $\mathbf{W}$).

[4]Actually, each node i only requires the weights corresponding to the i-th row of $\mathbf{W}$, along with the coefficient matrix Γ_i that is used in (4) to solve for the desired vector $\mathbf{c}'$.

[5]We will focus on linear functions because, as we will see, unbiased estimates of such functions can be obtained as a linear combination of the values seen by each node over the linear iteration. However, one can also use our results to obtain unbiased estimates of more general nonlinear functions.

Remark 3: As we will see in the next section, given an appropriate weight matrix $\mathbf{W}$, each node can obtain an unbiased estimate of its desired linear function simply by knowing the first order statistics of the noise (and not necessarily the second order statistics). If the second order statistics are also known (perhaps only after running the linear iteration), we will show that each node can refine its estimate of its linear function by taking an appropriate linear combination (given by the matrix Γ_i) of the values it sees over the course of the linear iteration. Note that the implicit assumption of a fixed and known topology is also made by much of the existing literature on distributed consensus in the presence of noise (e.g., see [5], [8]). In noise-free networks, it is possible for the nodes to calculate their observability matrices (and the gains Γ_i) in a distributed manner (see [10]), but the extension of such techniques to noisy networks is an open question and an avenue for future research.

IV. Unbiased Minimum-Variance Estimation

A. Unbiased Estimation

Consider the noisy system model given by (6). The output seen by node i over $L_i + 1$ time-steps is given by

$$\underbrace{\begin{bmatrix} \mathbf{y}_i[0] \\ \mathbf{y}_i[1] \\ \mathbf{y}_i[2] \\ \vdots \\ \mathbf{y}_i[L_i] \end{bmatrix}}_{\mathbf{y}_i[0:L_i]} = \underbrace{\begin{bmatrix} \mathbf{C}_i \\ \mathbf{C}_i\mathbf{W} \\ \mathbf{C}_i\mathbf{W}^2 \\ \vdots \\ \mathbf{C}_i\mathbf{W}^{L_i} \end{bmatrix}}_{\mathcal{O}_{i,L_i}} \mathbf{x}[0] +$$

$$\underbrace{\begin{bmatrix} \mathbf{D}_i & 0 & \cdots & 0 \\ \mathbf{C}_i\mathbf{B} & \mathbf{D}_i & \cdots & 0 \\ \mathbf{C}_i\mathbf{W}\mathbf{B} & \mathbf{C}_i\mathbf{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{C}_i\mathbf{W}^{L_i-1}\mathbf{B} & \mathbf{C}_i\mathbf{W}^{L_i-2}\mathbf{B} & \cdots & \mathbf{D}_i \end{bmatrix}}_{\mathcal{M}_{i,L_i}} \underbrace{\begin{bmatrix} \mathbf{n}[0] \\ \mathbf{n}[1] \\ \vdots \\ \mathbf{n}[L_i] \end{bmatrix}}_{\mathbf{n}[0:L_i]} . \quad (7)$$

We will assume here that the noise is zero mean (i.e., $E[\mathbf{n}[k]] = 0$ for all k); this assumption can be easily relaxed, but we adopt it here for simplicity. Suppose each node i wants to calculate an unbiased estimate of the function $\mathbf{c}_i'\mathbf{x}[0]$, for some vector $\mathbf{c}_i'$. We will find a gain Γ_i and the smallest integer L_i for each node i so that the quantity $\Gamma_i\mathbf{y}_i[0 : L_i]$ is an unbiased estimate of $\mathbf{c}_i'\mathbf{x}[0]$. To this end, we use (7) to examine the estimation error

$$\begin{aligned}\epsilon_i &\equiv \Gamma_i\mathbf{y}_i[0 : L_i] - \mathbf{c}_i'\mathbf{x}[0] \\ &= (\Gamma_i\mathcal{O}_{i,L_i} - \mathbf{c}_i')\,\mathbf{x}[0] + \Gamma_i\mathcal{M}_{i,L_i}\mathbf{n}[0 : L_i] \ . \end{aligned} \quad (8)$$

The estimate $\Gamma_i\mathbf{y}_i[0 : L_i]$ will be unbiased (i.e., $E[\epsilon_i] = 0$) for any given $\mathbf{x}[0]$ if and only if matrix Γ_i satisfies

$$\Gamma_i\mathcal{O}_{i,L_i} = \mathbf{c}_i' \ . \quad (9)$$

In other words, the vector $\mathbf{c}_i'$ must be in the row-space of the matrix $\mathcal{O}_{i,L_i}$. Following the notation in Theorem 1, let R_i denote the set of all nodes that have a path to node i in the network. As long as all nonzero entries of $\mathbf{c}_i'$ are in columns corresponding to nodes in R_i, Theorem 1 indicates that, for almost any choice of weight matrix $\mathbf{W}$, $\mathbf{c}_i'$ will be in the row space of $\mathcal{O}_{i,L_i}$, for some $0 \le L_i < |R_i| - \deg_i$. One can then find the smallest L_i for which the vector $\mathbf{c}_i'$ is in the row-space of $\mathcal{O}_{i,L_i}$, and this will also be the smallest number of time-steps required for unbiased estimation by node i (for that choice of $\mathbf{W}$). The above discussion immediately leads to the following theorem.

Theorem 2: Let $\mathcal{G}$ denote the graph of the network. Define the set $R_i = \{x_j |$ There exists a path from x_j to x_i in $\mathcal{G}\} \cup \{x_i\}$. Then, for almost any choice of weight matrix $\mathbf{W}$, node i can obtain an unbiased estimate of any linear function of values in $\{x_j[0] | x_j \in R_i\}$ after running the linear iteration (1) for $L_i + 1$ time-steps, for some $0 \le L_i < |R_i| - \deg_i$.

Note that for a given $\mathbf{W}$, there may be multiple choices of Γ_i satisfying (9). This leads us to ask the question: if the second order statistics of the noise are also known (perhaps *a posteriori*), can we obtain a better estimate of the linear function by choosing the gain Γ_i appropriately? We will address this question in the following section.

B. Minimizing the Variance of the Estimate

In order to minimize the mean square error of each node's estimate of its linear function, suppose that the covariance of the noise is known (or obtained over the course of the linear iteration), and given by $E[\mathbf{n}[k]\mathbf{n}'[j]] = \mathbf{Q}_{kj}$. Note that we are not assuming any constraints on the second order statistics (e.g., the noise does not have to be stationary, and can be colored). Examining the estimation error given by (8), we note that after satisfying the unbiased condition (9), the expression for the variance of the error is given by

$$\begin{aligned}\sigma_i &\equiv E\left[\epsilon_i\epsilon_i'\right] \\ &= \Gamma_i\mathcal{M}_{i,L_i}E\left[\mathbf{n}[0 : L_i]\mathbf{n}[0 : L_i]'\right]\mathcal{M}_{i,L_i}'\Gamma_i' \\ &= \Gamma_i\mathcal{M}_{i,L_i}\underbrace{\begin{bmatrix} \mathbf{Q}_{00} & \mathbf{Q}_{01} & \cdots & \mathbf{Q}_{0L_i} \\ \mathbf{Q}_{10} & \mathbf{Q}_{11} & \cdots & \mathbf{Q}_{1L_i} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Q}_{L_i0} & \mathbf{Q}_{L_i1} & \cdots & \mathbf{Q}_{L_iL_i} \end{bmatrix}}_{\Pi_{L_i}}\mathcal{M}_{i,L_i}'\Gamma_i' \ . \end{aligned} \quad (10)$$

Suppose L_i is chosen as the smallest integer for which (9) has a solution (this will be the smallest delay required for unbiased estimation by node i with the given weight matrix $\mathbf{W}$). Let the singular value decomposition of the matrix $\mathcal{O}_{i,L_i}$ be given by $\mathcal{O}_{i,L_i} = \mathbf{U}_i\left[\begin{smallmatrix}\Lambda_i & 0 \\ 0 & 0\end{smallmatrix}\right]\mathbf{V}_i'$, where $\mathbf{U}_i$ and $\mathbf{V}_i$ are unitary matrices, and Λ_i is a diagonal matrix with positive entries. Furthermore, $\rho(\Lambda_i) = \rho(\mathcal{O}_{i,L_i})$ [14]. Substituting this into (9), we get

$$\Gamma_i\mathbf{U}_i\begin{bmatrix}\Lambda_i & 0 \\ 0 & 0\end{bmatrix} = \mathbf{c}_i'\mathbf{V}_i \ . \quad (11)$$

Clearly, since $\mathbf{c}_i'$ is in the row-space of $\mathcal{O}_{i,L_i}$, we have

$$\mathbf{c}_i'\mathbf{V}_i = \begin{bmatrix}\mathbf{a}_i' & 0\end{bmatrix} \quad (12)$$

for some vector $\mathbf{a}_i'$ with $\rho(\mathcal{O}_{i,L_i})$ entries. Define the matrix

$$\widehat{\Gamma}_i \equiv \Gamma_i\mathbf{U}_i \ , \quad (13)$$

and partition it as $\widehat{\Gamma}_i = \begin{bmatrix}\widehat{\Gamma}_{i1} & \widehat{\Gamma}_{i2}\end{bmatrix}$, where $\widehat{\Gamma}_{i1}$ has $\rho(\mathcal{O}_{i,L_i})$ columns. Equation (11) then becomes $\begin{bmatrix}\widehat{\Gamma}_{i1} & \widehat{\Gamma}_{i2}\end{bmatrix}\left[\begin{smallmatrix}\Lambda_i & 0\\ 0 & 0\end{smallmatrix}\right] = \begin{bmatrix}\mathbf{a}_i' & 0\end{bmatrix}$. From this equation, it is apparent that

$$\widehat{\Gamma}_{i1} = \mathbf{a}_i'\Lambda_i^{-1} \ , \tag{14}$$

and $\widehat{\Gamma}_{i2}$ is completely unconstrained. In other words, $\widehat{\Gamma}_{i2}$ represents the freedom in the gain Γ_i after satisfying the unbiased constraint given by (9).

To minimize the variance of the estimation error, we substitute the above parameterization of the gain Γ_i into (10) to obtain

$$\sigma_i = \begin{bmatrix}\widehat{\Gamma}_{i1} & \widehat{\Gamma}_{i2}\end{bmatrix}\mathbf{U}_i'\mathcal{M}_{i,L_i}\Pi_{L_i}\mathcal{M}_{i,L_i}'\mathbf{U}_i\begin{bmatrix}\widehat{\Gamma}_{i1} & \widehat{\Gamma}_{i2}\end{bmatrix}' \ .$$

Define

$$\begin{bmatrix}\mathbf{\Phi}_i\\ \mathbf{\Psi}_i\end{bmatrix} \equiv \mathbf{U}_i'\mathcal{M}_{i,L_i} \ , \tag{15}$$

where $\mathbf{\Phi}_i$ has $\rho(\mathcal{O}_{i,L_i})$ rows. Using (14), the variance of the error becomes

$$\begin{aligned}\sigma_i &= \left(\mathbf{a}_i'\Lambda_i^{-1}\mathbf{\Phi}_i + \widehat{\Gamma}_{i2}\mathbf{\Psi}_i\right)\Pi_{L_i}\left(\mathbf{a}_i'\Lambda_i^{-1}\mathbf{\Phi}_i + \widehat{\Gamma}_{i2}\mathbf{\Psi}_i\right)'\\ &= \mathbf{a}_i'\Lambda_i^{-1}\mathbf{\Phi}_i\Pi_{L_i}\mathbf{\Phi}_i'\Lambda_i^{-1}\mathbf{a}_i + \mathbf{a}_i'\Lambda_i^{-1}\mathbf{\Phi}_i\Pi_{L_i}\mathbf{\Psi}_i'\widehat{\Gamma}_{i2}'\\ &\quad + \widehat{\Gamma}_{i2}\mathbf{\Psi}_i\Pi_{L_i}\mathbf{\Phi}_i'\Lambda_i^{-1}\mathbf{a}_i + \widehat{\Gamma}_{i2}\mathbf{\Psi}_i\Pi_{L_i}\mathbf{\Psi}_i'\widehat{\Gamma}_{i2}' \ .\end{aligned}$$

To minimize the above expression, we take the gradient with respect to $\widehat{\Gamma}_{i2}$ and set it equal to zero, which produces

$$\widehat{\Gamma}_{i2} = -\mathbf{a}_i'\Lambda_i^{-1}\mathbf{\Phi}_i\Pi_{L_i}\mathbf{\Psi}_i'\left(\mathbf{\Psi}_i\Pi_{L_i}\mathbf{\Psi}_i'\right)^{\dagger} \ ,$$

where the notation $(\cdot)^{\dagger}$ indicates the pseudo-inverse of a matrix [15]. From (13) and (14), we now obtain the optimal gain for node i as

$$\Gamma_i = \mathbf{a}_i'\Lambda_i^{-1}\begin{bmatrix}I & -\mathbf{\Phi}_i\Pi_{L_i}\mathbf{\Psi}_i'\left(\mathbf{\Psi}_i\Pi_{L_i}\mathbf{\Psi}_i'\right)^{\dagger}\end{bmatrix}\mathbf{U}_i' \ . \tag{16}$$

The variance of the optimal estimate can now be obtained from (10).

Remark 4: In the above derivation, we took L_i to be the smallest delay for which unbiased estimation is possible by node i. If one increases L_i past this minimum value, node i can potentially reduce the variance of its estimate. The variance of the estimate will be a nonincreasing function of the delay L_i, and thus the tradeoff between delay and variance can be taken as a design parameter for a given graph. The gain Γ_i and the variance σ_i for any value of L_i (above the minimum required for unbiased estimation) can be obtained by following the above procedure. A quantitative characterization of the relationship between delay and variance will be the subject of future research.

Remark 5: It may be the case that some nodes can obtain an unbiased estimate of their desired functions faster than others. In such cases, one can have all nodes run the linear iteration for $\max_{1\le j\le N} L_j + 1$ time-steps, so that every node receives enough information to obtain an unbiased estimate. Each node i can either calculate the function $\mathbf{c}_i'\mathbf{x}[0]$ after the first $L_i + 1$ time-steps of the linear iteration (i.e., with minimum delay), or it can use the outputs over all $\max_{1\le j\le N} L_j + 1$ time-steps (which could reduce the variance of its estimate).

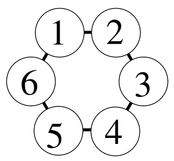

Fig. 1. Ring with 6 nodes from Example 1.

Example 1: Consider the ring with $N = 6$ nodes in Fig. 1. The transmissions between nodes are assumed to be corrupted by zero-mean white noise with unit variance. The objective in this system is for each node to calculate an unbiased estimate of the average of the initial values. To accomplish this, we set each edge weight to 1, and each self-weight to zero. We can now determine the gain matrix Γ_i and the minimum delay L_i required by each node i to calculate the desired function. For example, node 1 in Fig. 1 receives values from nodes 2 and 6, and has access to its own value, which means that $\mathbf{C}_1 = \left[\begin{smallmatrix}1&0&0&0&0&0\\0&1&0&0&0&0\\0&0&0&0&0&1\end{smallmatrix}\right]$. To find the number of time-steps required for node 1 to calculate an unbiased estimate of the average, we first have to find the smallest integer L_1 for which $\mathbf{c}' = \frac{1}{6}\mathbf{1}'$ is in the row-space of the matrix $\mathcal{O}_{1,L_1}$. With $L_1 = 0$, we have $\mathcal{O}_{1,0} = \mathbf{C}_1$, and this condition does not hold. For $L_1 = 1$, we have $\mathcal{O}_{1,1} = \begin{bmatrix}\mathbf{C}_1' & (\mathbf{C}_1\mathbf{W})'\end{bmatrix}'$, and again the condition does not hold. With $L_1 = 2$, we have $\mathcal{O}_{1,2} = \begin{bmatrix}\mathbf{C}_1' & (\mathbf{C}_1\mathbf{W})' & (\mathbf{C}_1\mathbf{W}^2)'\end{bmatrix}'$, and we find that the row-space of this matrix contains the vector $\frac{1}{6}\mathbf{1}'$. Therefore, node 1 can calculate an unbiased estimate of the average after $L_1 + 1 = 3$ time-steps (i.e., after it sees the outputs $\mathbf{y}_1[0]$, $\mathbf{y}_1[1]$ and $\mathbf{y}_1[2]$). Not surprisingly, we find that $L_i = 2$ for each node i, and thus all nodes in the system can calculate an unbiased estimate of the average after running the linear iteration for $L_i + 1 = 3$ time-steps. Note that both the radius and the diameter of this graph are equal to 3, and so no scheme can allow all nodes to calculate the average in fewer than three time-steps (i.e., the linear iterative scheme is time-optimal for this graph).

Having determined the minimum delay for unbiased estimation by each node, our task becomes to choose the gain Γ_i for each node i satisfying the unbiased condition (9), while minimizing the variance of the estimation error. To do this, we have to first construct the noisy system model in (6). For example, consider node 1 in Fig. 1. Since the values received by node 1 from its neighbors are corrupted by noise, the output seen by node 1 at time-step k is given by

$$\mathbf{y}_1[k] = \mathbf{C}_1\mathbf{x}[k] + \underbrace{\begin{bmatrix}0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix}}_{\widehat{\mathbf{D}}_1}\mathbf{n}_1[k] \ ,$$

where $\mathbf{n}_1[k]$ contains the additive noise on the links from node 2 and node 6 to node 1 at time-step k. The outputs seen by all the other nodes can be obtained in a similar manner. We can group the noise vectors seen by each node into the single noise vector

$$\mathbf{n}[k] = \begin{bmatrix}\mathbf{n}_1'[k] & \mathbf{n}_2'[k] & \mathbf{n}_3'[k] & \mathbf{n}_4'[k] & \mathbf{n}_5'[k] & \mathbf{n}_6'[k]\end{bmatrix}' ,$$

which has twelve entries, since there are six bidirectional links in the graph. Note that $\mathbf{n}[k]$ is white noise with $E[\mathbf{n}[k]] = 0$ and $E[\mathbf{n}[k]\mathbf{n}'[k]] = I_{12}$ (by assumption in this example). With this notation, the output for node 1 is given by

$$\mathbf{y}_1[k] = \mathbf{C}_1\mathbf{x}[k] + \underbrace{\begin{bmatrix}\widehat{\mathbf{D}}_1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}}_{\mathbf{D}_1}\mathbf{n}[k] \ .$$

Since each node weighs the values that it receives from its neighbors by 1, the state update equation is given by the first equation in (6), where the matrix $\mathbf{B}$ is obtained from (5) as $\mathbf{B} = I_6 \otimes \begin{bmatrix}1 & 1\end{bmatrix}$ (the symbol $\otimes$ denotes the Kronecker product).

We now have to find the gain Γ_1 for node 1 satisfying (9), while minimizing the error variance in (10). To do this, we first find the singular value decomposition of $\mathcal{O}_{1,2}$ as $\mathcal{O}_{1,2} = \mathbf{U}_1 \left[\begin{smallmatrix}\Lambda_1 \\ 0\end{smallmatrix}\right] \mathbf{V}_1'$, where $\Lambda_i = \text{diag}(3.682, 3.113, 1.4142, 1, 0.6653, 0.5564)$. We omit the values of $\mathbf{U}_1$ and $\mathbf{V}_1$ in the interest of space. From the above decomposition, we obtain the vector $\mathbf{a}_1'$ in (12) as

$$\begin{aligned}\mathbf{a}_1' &= \frac{1}{6}\mathbf{1}'\mathbf{V}_1 \\ &= \begin{bmatrix}-0.2787 & -0.267 & 0 & 0 & -0.0752 & -0.1098\end{bmatrix} .\end{aligned}$$

We also obtain the matrices $\mathbf{\Phi}_i$ and $\mathbf{\Psi}_i$ from (15) (again, these values are omitted in the interest of space). Substituting these values into the expression for the optimal gain in (16), with $\Pi_2 \equiv E[\mathbf{n}[0:2]\mathbf{n}'[0:2]] = I_{36}$ (since the noise is white with unit variance), we obtain $\Gamma_1 = \frac{1}{36}\begin{bmatrix}-6 & -1 & -1 & -2 & 2 & 2 & 4 & 3 & 3\end{bmatrix}$. The mean-square error for the estimate of the average obtained by node 1 is calculated by substituting the above gain into (10), and is found to be $\sigma_1 = 0.25$. The above procedure can be repeated to obtain the optimal gains for all nodes, and in this example the minimum mean-square error for all nodes is the same (i.e., $\sigma_i = 0.25$ for all i).

Once the optimal gains are calculated and provided to each node, suppose that the initial values of the nodes are given by $\mathbf{x}[0] = \begin{bmatrix}0.8372 & -5.3279 & 3.6267 & 4.4384 & -2.7324 & 8.9546\end{bmatrix}'$, which has a mean of 1.6328. The nodes run the linear iteration given by (6) for three time-steps with $\mathbf{x}[0]$ as given above, and driven by zero-mean white noise with unit variance on each link. An example of the outputs seen by node 1 during the three time-steps, with a particular sampling of noise vectors $\mathbf{n}[0], \mathbf{n}[1]$ and $\mathbf{n}[2]$, is given by

$$\begin{aligned}\mathbf{y}_1[0] &= \begin{bmatrix} 0.8372 & -5.7054 & 8.6587 \end{bmatrix}', \\ \mathbf{y}_1[1] &= \begin{bmatrix} 2.9533 & 2.9668 & -1.7085 \end{bmatrix}', \\ \mathbf{y}_1[2] &= \begin{bmatrix} 1.2583 & 1.5285 & 17.8313 \end{bmatrix}' .\end{aligned}$$

Node 1 then obtains a minimum-variance unbiased estimate of the average as $\Gamma_1 \begin{bmatrix}\mathbf{y}_1'[0] & \mathbf{y}_1'[1] & \mathbf{y}_1'[2]\end{bmatrix}' = 1.4374$. The other nodes follow the same procedure, and the values calculated by nodes 2–6 (for this sample run) are 1.2716, 1.3985, 1.9846, 1.7501 and 1.4736, respectively. One can verify empirically (i.e., by running several simulations with different initial conditions and calculating the average squared estimation error) that the variance of the estimation error at each node i is indeed close to the theoretical value of 0.25.

V. Summary

We have studied the problem of performing distributed calculation of linear functions in systems operating in the presence of noise. In particular, we analyzed a linear iteration-based scheme where each node updates its value as a linear combination of its own value, and those of its neighbors. In the study of traditional linear iterative schemes that rely on asymptotic convergence, it has been shown that the presence of noise can drive the node values arbitrarily far away from the desired function. To solve this problem, we utilized results on finite-time function calculation via linear iterations. Specifically, for a given set of update weights, we showed that it is possible to calculate a set of gains for each node that allows it to obtain a minimum-variance unbiased estimate of the function after running the linear iteration for a finite number of time-steps.

References

[1] N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc., 1996.

[2] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 755–764, Apr. 2005.

[3] M. Rabbat and R. D. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN)*, 2004, pp. 20–27.

[4] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[5] L. Xiao, S. Boyd, and S.-J. Kim, "Distributed average consensus with least-mean-square deviation," *Journal of Parallel and Distributed Computing*, vol. 67, no. 1, pp. 33–46, Jan. 2007.

[6] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc WSNs with noisy links – Part I: distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, Jan. 2008.

[7] M. Huang and J. H. Manton, "Stochastic double array analysis and convergence of consensus algorithms with noisy measurements," in *Proceedings of the American Control Conference*, 2007, pp. 705–710.

[8] M. E. Yildiz and A. Scaglione, "Differential nested lattice encoding for consensus problems," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN)*, 2007, pp. 89–98.

[9] H. V. Poor, *An introduction to signal detection and estimation*, 2nd ed. Springer-Verlag, 1994.

[10] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation and consensus using linear iterative strategies," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, May 2008, to appear.

[11] C.-T. Chen, *Linear System Theory and Design*. Holt, Rinehart and Winston, 1984.

[12] D. B. West, *Introduction to Graph Theory*. Prentice-Hall Inc., Upper Saddle River, New Jersey, 2001.

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

[14] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1985.

[15] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1979.