Dynamic Average Consensus over Random Networks with Additive Noise

Jing Wang and Nicola Elia

Abstract—In this paper, we consider distributed dynamic average consensus problem in the presence of uncertainties on information exchange. Two categories of noise are used to characterize these uncertainties: the first is multiplicative noise that captures the randomness of network connections, while the second is additive noise that describes several uncertainty sources. We propose an iterative algorithm that allows each agent to compute/track the average of their private dynamic signals in the presence of both kinds of noise. This algorithm relaxes restrictive assumptions on consensus over random directed network topologies, such as doubly stochastic weights, symmetric link switching styles, etc., and introduces new mechanisms for mitigating effects of communication uncertainties on information aggregation.

Keywords: Consensus, Additive noise, Link failures, Sensor fusion, Random networks, Dynamic average consensus.

I. INTRODUCTION

Consider a network of a set of nodes (agents), where each node has access to a private dynamic signal. The objective of each node is to compute or track the average of all the signals via information exchange between neighboring nodes. This problem is referred as distributed dynamic average consensus problem (DDACP), which was first investigated in [6] and later on studied by many researchers, e.g., [9], [10].

A special case of DDACP is when the signals to be averaged are static, which is referred as distributed average consensus problem and has attracted a lot of interest in control community recently, see, e.g., the seminal works [1], [2], [3]. The broad interest on DDACP is motivated from its vast applications including distributed Kalman filtering, load balancing, estimation and detection over sensor networks, distributed optimization, distributed in-channel coding, and multi-agent coordination, etc.

Restricted by limited computation, memory and power resources at each node, it is desirable to solve DDACP by iterative algorithms, algorithms that generate a sequence of points where each point depends on the preceding ones. We shall restrict our discussion to iterative algorithms (or iterative computation models) in this paper. One common feature of distributed iterative algorithms is the need of communication. However, the uncertainties during information exchange, such as additive noise or link failures could bring detrimental effects on their performance or even loss of convergence.

It was recognized in [17] that consensus algorithms are sensitive to additive noise in the sense that the agents' state trajectories behave as random walks as simple stochastic disturbances are added to the system. This is due to the marginally stable nature of the classical consensus algorithms. See also [12] for the trade off between convergence speed and robustness to additive noise of consensus algorithms. To eliminate this random walk behavior, [13] has proposed an algorithm that uses the diminishing step size, see also [14]. This model guarantees that all nodes converge to a common value, but unfortunately, that value is not predictable, i.e., it assumes different values for each sequence of iterations depending on particular realizations of noise. Thus, this algorithm may be useful for synchronization problems but not for distributed computation. Based on optimization theory, [4] introduced an algorithm which is robust to additive noise in the sense that the deviation between each node value and their average is bounded, which works for undirected graphs. Different from mentioned previous works that consider static private signals, [5] has proposed another computation model, which eliminates the random walks, and allows dynamic average tracking at the expenses of more communications. The scheme has similar noise resilient property as that in [4] and is applicable to more general network topologies.

Distributed consensus in the presence of random networks are considered in [7], [8], [20], [11], [4], [15] to cite a few. To our knowledge, distributed average consensus in random or time varying networks whose topologies are modeled as directed graphs is a difficult problem. This difficulty always translates into some strong or even impractical assumptions, e.g., the well known double stochastic weights assumption, symmetric link switchings over undirected graphs [20], or the probability of link failures being known as a priori [4].

The main contribution of this paper is that by appropriate extension of the algorithm in [5], we show that it can be used to solve DDACP over random networks without the above mentioned assumptions, and also exhibits robustness to additive noise.

Our stochastic setup begins with a network of fixed topology, called the connectivity graph. At each discrete time index, we assume the links of the connectivity graph are randomly independently switching (on or off). Therefore, network topologies at all time are subgraphs of the connectivity graph and randomly changing. To account for random link failures, we introduce a particular decomposition of Laplacian matrix that captures its component associated with each link. Moreover, we allow each agent to use the Channel State Information (CSI) of its incoming links. The simple strategy consists of using the latest received (good) message (packet) when the channel is bad and the message...
has been lost (switch open). This protocol is implemented through the introduction of an extra state to hold the previous good message. This holding technique is able to keep the information flow of each link smooth and also allows dynamic average consensus under independent link switching.

To establish the convergence property of our algorithm, we first consider the system without extra input (called the unforced system), and decompose it into two parts, namely, the conserved system and the deviated system. Adopting the framework of [21], we provide necessary and sufficient conditions for mean square (MS) stability of the deviated system, which is shown to be sufficient for MS convergence (or bounded MS deviation in the presence of additive noise) of our computation model.

II. SETUP, PRELIMINARIES AND PROBLEM FORMULATION

We consider a set of $n$ nodes that are connected over a network. The interconnections among the agents are properly described by a directed graph $G = (V,E)$, where $V = \{1, \ldots, n\}$ is the set of nodes and $E \subseteq V \times V$ is the set of edges. An edge of $G$ is denoted by $(i,j)$ implying there exists a communication channel from node $j$ to node $i$. For any node $i$, we define its information neighbor as

$$N_i = \{j \in V : (i,j) \in E\}$$

i.e., the nodes from which node $i$ can obtain the information.

For each edge $(i,j)$, we associate it with a scalar weight $a_{ij} > 0$ and define the graph Laplacian of $G$ as $L = [l_{ij}]$ with

$$l_{ij} := \begin{cases} \sum_{j \in N_i} a_{ij} & \text{if } j = i \\ -a_{ij} & \text{if } j \neq i \end{cases}$$

The graph Laplacian has an important property that $\text{Null}(L) = \text{span}\{1\}$, where $1$ is a vector composed by all ones. The in-degree of each node is defined as $a_{in} = \sum_{j \in N_i} a_{ij}$ and the out-degree of each node is defined as $a_{out} = \sum_{j: (j,i) \in E} a_{ij}$. The Laplacian is said to be balanced if $a_{in} = a_{out}$ for all $i \in V$. The balanced Laplacian has the property that $1^TL = 0$.

A graph with the property that for any $(i,j) \in E$, the edge $(j,i) \in E$ as well is said to be undirected. The graph is said to be strongly connected if every node can reach every other node through a sequence of elements in $E$. In this paper, we consider the following network topologies

**Assumption 2.1:** The graph $G$ associated with the network topology is strongly connected.

This assumption\(^1\) ensures that the information of every node can be received by every other node through a directed path. It is well known that if the graph is strongly connected, the corresponding Laplacian matrix has only one eigenvalue of zero and all other eigenvalues are in open right half plane.

\(^1\)Although weaker assumptions on connectivity leads to consensus, e.g., the topology only contains a spanning tree, such assumption does not ensure that the final consensus value contains information of all nodes, see, e.g., [16].

Besides, the eigenvector associated with the zero eigenvalue is $1$.

In this paper, we consider the following distributed dynamic average consensus problem (DDACP).

**Problem 2.1:** (DDACP) Consider a network with its topology denoted by the graph $G = (V,E)$, let $u_i(k) \in \mathbb{R}$ (dynamic or static signal) be the private signal available to node $i$. Design the iterative algorithm for each node with information exchange among neighboring nodes such that each node tracks the average of all the dynamic signals, i.e., $\frac{1}{n}\sum_{i=1}^{n} u_i(k)$, in some useful sense\(^2\).

III. DYNAMIC AVERAGE CONSENSUS IN THE PRESENCE OF ADDITIVE NOISE

In this section, we review the model in [5] that can be used to solve DDACP in the presence of additive noise. This is the basis for our further development.

Let $x_i(k)$ denote the state of agent $i$ at each time $k$, we propose the following iterative algorithm for each node:

$$x_i(k + 1) = x_i(k) + \beta \sum_{j \in N_i} a_{ij}(y_j(k) - y_i(k)) + w_i(k),$$

$$y_i(k) = u_i(k) - \sum_{j \in N_i} a_{ij}(x_j(k) - x_i(k)) + v_i(k)$$

where $y_i$ is the output of agent $i$, $\beta > 0$ is some positive constant, $a_{ij} > 0$ is the weight for the link $(i,j)$, $w_i(k)$ and $v_i(k)$ are Gaussian random variables with mean 0 and variance $\sigma_w^2$ and $\sigma_v^2$, respectively. We assume $\{w_i(k), v_i(k)\}$ is an independent set for all spatial index $i$ and time index $k$. Note that the noise $v_i$ and $w_i$ may represent the channel noise\(^3\), or a model for the disruptions or malicious attacks to the system [18], or measurement noise in estimation problems.

We can write (1) for all agent $i$ compactly as

$$x(k + 1) = x(k) + \beta Ly(k) + w(k),$$

$$y(k) = u(k) - Lx(k) + v(k).$$

where $w(k) \sim \mathcal{N}(0, \Sigma_w)$ and $v(k) \sim \mathcal{N}(0, \Sigma_v)$ for all $k$. Note that (2) can be seen as state space realization of an LTI system whose block diagram is shown in Figure 1, where two Laplacian matrices are in the feedback loop. This implies that two stages of information exchange and computation are

\(^2\)Here, we do not provide a mathematical characterization of the tracking, which will be specified in our later discussion.

\(^3\)When we consider channel noise, we should move $v_i$ and $w_i$ inside the brackets. This can be done with little definition adjustments and without loss of generality.
during each time slot, which can be partitioned as two sub time slots. In the first sub time slot, all agents exchange their states and compute their output $y_i$; while in the second sub time slot, all agents exchange their output to update the new states.

A. Feasible Laplacian Matrices

From (2), we obtain the state equation as

$$x(k+1) = (I - \beta L^2)x(k) + \beta Lu(k) + \beta Lv(k) + w(k).$$

If $L^2$ has one of its eigenvalues in the open left half plane, then the above system is unstable and $x$ will exploding. To avoid this, certain condition has to be imposed on $L$ to restrict the eigenvalues of $L^2$ in the right half plane.

Definition 3.1: (feasible Laplacian matrix) The Laplacian matrix is said to be feasible if $L^2$ has only one eigenvalue at 0 and all the other eigenvalues in the strictly right half plane.

Lemma 3.2: [5] Let $L$ be a Laplacian matrix of a directed graph $G$, then $L$ is feasible if and only if there exists $X > 0$ such that

$$\begin{bmatrix} \tilde{L}X + XL' & \tilde{L}X - XL' \\ XL' - \tilde{L}X & \tilde{L}X + XL' \end{bmatrix} > 0$$

(3)

where $\tilde{L} = L + c \cdot 1 \gamma', \gamma' L = 0, \gamma' 1 = 1$ and $c > 0$ is a constant.

This condition ensures that, except for the zero eigenvalue, all other eigenvalues of $L$ have their real part strictly greater than the absolute value of their imaginary part. Note that this condition is satisfied if the graph $L$ is symmetric (corresponding to undirected graph), as in this case, all the eigenvalues of $L$ are real nonnegative numbers.

B. Mean Square Boundedness Tracking

After deriving the LMI condition for feasible Laplacian matrices, we are going to state the result concerning the tracking property of the model (1). The main feature of the model is that it eliminates the random walk behavior in the presence of additive noise of most existing consensus algorithms.

Theorem 3.3: [5] Consider the network $G$ with its Laplacian $L$ feasible, for system (2), if the following conditions are satisfied,

1) $L$ is balanced;
2) $I - \beta L^2$ has only one eigenvalue at 1 and all the other eigenvalues inside the unit disk;
3) the $Z$ transform of $u(k)$, $U(z)$, has at most one pole at 1 and all the other poles inside the unit circle.

then

$$\lim_{k \to \infty} E \left( y(k) - \frac{11'}{n} u(k) \right) \left( y(k) - \frac{11'}{n} u(k) \right)' < \infty.$$

If the input is static, i.e., $u(k) = u$, then the above quantity converges to a constant matrix depending on $L$ and the variances of $v$ and $w$.

The above result means that, in the presence of noise, the deviation between the output and the average of the input signals are bounded. In case when the input signals are static, as the noise has zero mean, we are able to approximately recover the average of the input signals by averaging the outputs signals over time. It is worth mentioning that here we use a very simple design parameter $\beta$ to guarantee the stability of the system. It is possible to use systematic design approach developed in [19] to ensure the stability.

IV. DYNAMIC AVERAGE CONSENSUS OVER RANDOM NETWORKS

A. Stochastic Setup

We now consider DDACP (cf. Problem 2.1) when the topologies of the network are stochastically changing over time. Our stochastic setup begins with a fixed network topology, say $G = (V, E)$, which is referred as the connectivity graph. We also adopt Assumption 2.1, namely $G$ is strongly connected. At each time slot, we assume each link $(i, j) \in E$ is stochastically breaking. Since there are two transmission procedures in each time slot, we use two Bernoulli random variables $\xi_1^k$ and $\xi_2^k$ to characterize this randomness property. $\xi_1^k = 1$ implies that the link $(i, j)$ is connected and $y_i(k)$ is transmitted to node $i$, while $\xi_1^k = 0$ implies that the link $(i, j)$ is disconnected during the transmission of $y_j(k)$. Similarly, $\xi_2^k$ describes the stochastic breaking property of $(i, j)$ when $x_j(k)$ is transmitted.

We assume the pairs $(\xi_1^k, \xi_2^k)$ are independent across both the time index $k$ and spatial index $i, j$ and independently distributed for each $k$. For each pair, we assume $\xi_1^k = \xi_2^k = \xi_{ij}^k$ are independently identically distributed according to

$$\xi_{ij}^k = \begin{cases} 1 & \text{with Prob. } \mu_{ij} \\ 0 & \text{with Prob. } 1 - \mu_{ij} \end{cases}$$

for $m = 1, 2$.

This setup captures two important limitations on information exchange. First of all, over each time slot $k$, associated with $\xi_1^k$ and $\xi_2^k$, there are two graphs, namely $G_1^k = (V, E_1^k)$ and $G_2^k = (V, E_2^k)$, whose set of edges satisfy $E_1^k \subset E, E_2^k \subset E$. Therefore, for all $k$, $G_1^k$ and $G_2^k$ are all subgraphs of $G$ and randomly changing over time. Besides the limitation of random connectivity, the stochastic setup also characterizes packet drops in the transport layer in communication networks.

From now on, we adopt the following assumptions for the rest of the paper.

Assumption 4.1: The Laplacian matrix associated with $G$ is balanced and feasible.

Assumption 4.2: We assume that $0 < \mu_{ij} < 1$ for all $i$ and $j \in N_i$.

B. The Computation Model

Before describing the model, we introduce a decomposition of Laplacian matrices that captures its elements associated with each link. For a graph with $p$ directed edges, its Laplacian $L$ can be written as $L = BC$, where $B \in \mathbb{R}^{p \times p}$ serves for permutation purpose and $C \in \mathbb{R}^{p \times n}$ is a concatenation of Laplacian elements corresponding to

4791
each link. To illustrate this decomposition, we consider the Laplacian matrix
\[
L = \begin{bmatrix}
2 & -1 & -1 \\
-1 & 1 & 0 \\
0 & -1 & 1
\end{bmatrix},
\]
then, we can decompose \( L \) as \( L = BC \), where
\[
B = \begin{bmatrix}
1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix},
C = \begin{bmatrix}
1 & -1 & 0 \\
1 & 0 & -1 \\
-1 & 1 & 0 \\
0 & -1 & 1
\end{bmatrix}.
\]
Note that each row of \( C \) represents the elements of Laplacian associated with each link, e.g., the first row of \( C \) corresponds to link \((1, 3)\). Therefore, we use \( C_{ij} \) to represent the row of \( C \) that corresponds to link \((i, j)\).

At each time index \( k \), we assume every agent has the state information about all its incoming links and associates each incoming link \((i, j)\) with a state \( h_{ij} \), whose dynamics is governed by
\[
h_{ij}(k) = \xi_{ij}^0(k)C_{ij}x(k) + (1 - \xi_{ij}^0(k))h_{ij}(k-1),
\]
(4)
Moreover, each agent use the following updating strategy
\[
\begin{align*}
x_i(k+1) &= x_i(k) + \beta \sum_{j \in N_i} \xi_{ij}^2(k)(y_j(k) - y_i(k)), \\
y_i(k) &= u_i(k) - B(i)h(k),
\end{align*}
\]
where we use \( B(i) \) to represent the \( i \)-th row of the matrix \( B \) and \( h(k) \) is the concatenation of \( h_{ij}(k) \).

Note that our model is fully distributed, i.e., every agent only use its local information to update its state and output. (To see this, note that the nonzero elements of \( B(i) \) correspond to \( h_{ij} \) for \( j \in N_i \).) As the random variables \( \xi_{ij}^0 \) and \( \xi_{ij}^2 \) are randomly switched between 0 and 1, they are also named as multiplicative noise, see, e.g., [22], chap. 9.

C. Separation of Uncertainties

We first consider the system (4) and (5) without input \( u \), namely, the uniform system, because this lays the foundation to establish the convergence property of the algorithm with input \( u \).

We write (4) and (5) compactly without input \( u \) as
\[
\begin{align*}
x(k+1) &= x(k) + \beta B \Lambda_1(k) C y(k), \\
h(k) &= \Lambda_2(k) C x(k) + (I_p - \Lambda_2(k))h(k-1), \\
y(k) &= -B h(k),
\end{align*}
\]
(6)
where \( \Lambda_1(k) \) is a diagonal matrix with diagonal entries as \( \xi_{ij}^0 \) and \( \Lambda_2(k) \) is a diagonal matrix with entries \( \xi_{ij}^2 \). Here, \( h(k) \) is a concatenation of \( h_{ij}(k) \) and \( h(k) \in \mathbb{R}^p \).

In order to analyze the stability property of (6), we adopt the general framework of [21] and assume \( \mu_{ij} = \mu \) for all links. This assumption only simplifies the following analysis of MS stability and does not affect the main result of the paper, i.e., our algorithm converges with non-uniform link failures as well.

Let \( z(k) = (x(k), h(k-1))' \) and \( \tilde{\xi}_{ij} = \mu + \Delta_{ij}^l \) for all \( i \in V, j \in N_i \) and \( l = 1, 2 \), then \( \Delta_{ij}^l \) has zero mean and variance \( \mu(1 - \mu) \), and \( \Lambda_i = \mu I + \Delta_i \) for \( i = 1, 2 \), where \( \Delta_i \) is a diagonal matrix with entries equal to \( \Delta_{ij} \). In this way, (6) can be written as
\[
\begin{align*}
z(k+1) &= Az(k) + B_1 w(k), \\
q(k) &= C_1 z(k) + Dw(k), \\
w(k) &= \Delta q(k),
\end{align*}
\]
(7)
where
\[
A = \begin{bmatrix}
I_n - \beta \mu^2 L^2 & \beta \mu (1 - \mu) L B \\
\mu C & (1 - \mu) I_p
\end{bmatrix},
B_1 = \begin{bmatrix}
\beta B & -\beta \mu L B \\
0 & I_p
\end{bmatrix},
C_1 = \begin{bmatrix}
-\mu C L & (1 - \mu) C B \\
C & -I_p
\end{bmatrix},
D = \begin{bmatrix}
0 & -C B \\
0 & 0
\end{bmatrix},
\]
and \( \Delta \) is a diagonal matrix with \( \Delta_1 \) and \( \Delta_2 \) on the diagonal. The block diagram of system (6) and the corresponding transformation are shown in Figure 2 (a) and (b). Here, (7) can be seen as the interconnection between the structured uncertainty \( \Delta \) and the mean system \( G \), which has the state space as \( G := (A, B_1, C_1, D) \). This separation of uncertainty is shown in Figure 2(c).

D. Orthogonal Decomposition and MS stability

We start this section by investigating the structural property of \( A \) and \( C \) that will be used later. We first note that if \( L \) is balanced, then \( A \) has a right and left eigenvector of the form \( e = [1_m, 0_p]' \) corresponding to eigenvalue \( 1 \), i.e., \( A e = e \) and \( e A = e \). We define the projection matrix \( P := I - \frac{ee'}{e' e} \). The following lemma characterizes some properties of this projection matrix.

**Lemma 4.3:** Let \( L \) be a balanced graph Laplacian, then we have that
\[
1) \quad P^2 = P,
2) \quad P' = P.
\]
\footnote{We neglect the output equation here as we are interested in internal stability of the system.}

Fig. 2. Block diagram transformation of System (6).
2) $PA = AP$;
3) $C_1P = C_1$;
4) the null space of $P$ is span{$e$}.

The key idea in our analysis is to decompose the system (7) into two orthogonal parts and analyze them separately. We first decompose the system state $z(k)$ into two parts.

**Definition 4.4:** The deviated state is defined as $z_d(k) := Pz(k)$ and the conserved state is defined as $z_c(k) := Fz(k)$, where $F = I - P = \frac{P}{\|P\|}$ and the subscripts stand for obvious meaning.

Note that $z = z_d + z_c$. Once we define the two states, the deviated system and the conserved system, with inherited meaning from their states, can respectively be obtained from (7) as

$$
\begin{align*}
    z_d(k+1) &= \tilde{A}z_d(k) + \tilde{B}_1w(k), \\
    q(k) &= \tilde{C}_1z_d(k) + Dw(k) \\
    w(k) &= \Delta q(k), \\
    z_c(k+1) &= z_c(k) + FB_1w(k).
\end{align*}
$$

where $\tilde{A} = PAP$, $\tilde{B} = PB_1$ and $\tilde{C} = CP$, and

In the above derivations, we use the results of Lemma 4.3. It can be seen from (8) and (9) that the evolution of $z_d$ is completely decoupled from that of $z_c$ while the dynamics of $z_c$ depends on that of $z_d$ through $w(k)$. System (8) can also be seen as the interconnection of a deterministic plant $\tilde{G} := (\tilde{A}, \tilde{B}_1, \tilde{C}_1, D)$ and $\Delta$. Another important point is that by projection, $\tilde{A}$ has one eigenvalue at 0 which is originally the eigenvalue 1 of $A$. This implies that if $A$ contains one eigenvalue at 1 and all other eigenvalues strictly inside the open unit disk, then $\tilde{A}$ is Hurwitz.

Let $M_d(k) := \mathbb{E}\{z_d(k)z_d(k)\}$, we provide the following definition.

**Definition 4.5:** System (8) is MS stable if $M_d(k)$ is well defined for all $k$ and $\lim_{k \to \infty} M_d(k) = 0$.

The following condition is necessary and sufficient for the MS stability of the deviated system, which is based on [21] with extra algebra.

**Theorem 4.6:** Assume $\tilde{A}$ is Hurwitz, then system (8) is MS stable if and only if $\tilde{G}$ is stable and $\rho(\sigma^2 \tilde{G}) < 1$. where

$$
\tilde{G} = \left[
\begin{array}{ccc}
\|G_{11}\|_2^2 & \cdots & \|G_{1p}\|_2^2 \\
\vdots & \ddots & \vdots \\
\|G_{p1}\|_2^2 & \cdots & \|G_{pp}\|_2^2
\end{array}
\right].
$$

**Remark 4.7:** The system $G$ and $\tilde{G}$ has the same input-output map with different descriptions. The above result implies that the MS stability of the deviated system can be assessed from the original system (6) without decomposition.

**E. MS convergence of the Forced System**

In this section, we will establish the convergence property of (4) and (5). The following theorem states that the output of the forced system tracks the average of the inputs with zero variance.

**Theorem 4.8:** Assume that the $Z$ transform of $u(k), U(z)$, has at most one pole at 1 and all the other poles in the open unit circle. If the deviated system (8) is MS stable, then we have

$$
\lim_{k \to \infty} \mathbb{E}\left( y(k) - \frac{11'}{n} u(k) \right) \left( y(k) - \frac{11'}{n} u(k) \right)' = 0.
$$

Theorem 4.8 shows that MS stability of the deviated system actually guarantees the convergence of individual output to the average of the input signals in the mean square sense. When we consider both random connectivity and additive noise, the computation model is given by

$$
\begin{align*}
    h_{ij}(k) &= \xi_{ij}(k)C_{ij}x(k) + (1 - \xi_{ij}(k))h_{ij}(k-1), \\
    x_i(k+1) &= x_i(k) + \beta \sum_{j \in N_i} \xi_{ij}(k)(y_j(k) - y_i(k)) + w_i(k), \\
    y_i(k) &= u_i(k) - B(i)h(k) + v_i(k),
\end{align*}
$$

(10)

We have the following result for (10).

**Theorem 4.9:** Assume that the $Z$ transform of $u(k), U(z)$, has at most one pole at 1 and all the other poles in the open unit circle. If the deviated system (8) is MS stable, then for (10) we have

$$
\lim_{k \to \infty} \mathbb{E}\left( y(k) - \frac{11'}{n} u(k) \right) \left( y(k) - \frac{11'}{n} u(k) \right)' < \infty.
$$

If the input is static, i.e., $u(k) = u$, then the above quantity converges to a constant matrix.

**F. Illustrative Examples**

In this section, we present an example to illustrate the effectiveness of our model. We consider a network of four
nodes, whose interconnection is shown in Figure 3. We assume the weights associated with the links are all equal to 1, therefore the graph Laplacian is balanced and satisfy the LMI condition (3) (this can be easily verified). The other parameters are given as: $\beta = 0.04$, $\mu_{ij}$ is randomly chosen between $[0, 1]$ and the noise is assumed to be $\sigma_{w_i} = \sigma_{v_i} = 0.01^2$ for each $i \in V$.

We consider the situation when the input signals are static due to space limits. We set $u = [1, 2, 3, 4]^T$ and the output of each agent converges to the average of $u_i$ with no uncertainty, i.e., every time we run the model, each output converges to the average. This is shown in Figure 4. We next consider the case when both additive noise and link failures are present, the output trajectories in this case for $u = [1, 2, 3, 4]^T$ is shown in Figure 5. To approximately recover the average consensus value, each agent can store her output at different times and compute their average.

V. CONCLUSIONS

In this paper, we have developed a novel computation model for distributed averaging which shows robust feature with respect to both random connectivity and additive noise. We provide numerical conditions to check MS stability of part of the autonomous system, which in turn, guarantees the convergence of the algorithm with input signals. It would be of interest to investigate the effects of communication delays on the algorithm.

REFERENCES


