

# Distributed Local Linear Parameter Estimation using Gaussian SPAWN

Mei Leng, Wee Peng Tay, Tony Q.S. Quek, and Hyundong Shin

## Abstract

We consider the problem of estimating local sensor parameters, where the local parameters and sensor observations are related through linear stochastic models. Sensors exchange messages and cooperate with each other to estimate their own local parameters iteratively. We study the Gaussian Sum-Product Algorithm over a Wireless Network (gSPAWN) procedure, which is based on belief propagation, but uses fixed size broadcast messages at each sensor instead. Compared with the popular diffusion strategies for performing network parameter estimation, whose communication cost at each sensor increases with increasing network density, the gSPAWN algorithm allows sensors to broadcast a message whose size does not depend on the network size or density, making it more suitable for applications in wireless sensor networks. We show that the gSPAWN algorithm converges in mean and has mean-square stability under some technical sufficient conditions, and we describe an application of the gSPAWN algorithm to a network localization problem in non-line-of-sight environments. Numerical results suggest that gSPAWN converges much faster in general than the diffusion method, and has lower communication costs, with comparable root mean square errors.

## Index Terms

Local estimation, distributed estimation, sum-product algorithm, belief propagation, diffusion, wireless sensor network.

## I. INTRODUCTION

A wireless sensor network (WSN) consists of many sensors or nodes capable of on-board sensing, computing, and communications. WSNs are used in numerous applications like environmental monitoring,

M. Leng and W.P. Tay are with the Nanyang Technological University, Singapore. (e-mail:{lengmei,wptay}@ntu.edu.sg).

T.Q.S. Quek is with the Information Systems Technology and Design Pillar, SUTD, Singapore. (e-mail:qsquek@ieee.org).

H. Shin is with the Department of Electronics and Radio Engineering, Kyung Hee University, Yongin-si, Gyeonggi-do, 446-701, Korea (e-mail: hshin@khu.ac.kr).

pollution detection, control of industrial machines and home appliances, event detection, and object tracking [1]–[4]. In distributed estimation [5]–[7], sensors cooperate with each other by passing information between neighboring nodes, which removes the necessity of transmitting local data to a central fusion center. Distributed estimation schemes hence have the advantages of being scalable, robust to node failures, and are more energy efficient due to the shorter sensor-to-sensor communication ranges, compared with centralized schemes which require transmission to a fusion center. It also improves local estimation accuracy [8], [9]. For example, by cooperating with each other to perform localization, nodes that have information from an inadequate number of anchors can still successfully self-localize [10], [11].

In this paper, we consider distributed local linear parameter estimation in a WSN in which each sensor cooperates with its neighbors to estimate its own local parameter, which is related to its own observations as well as its neighbors' observations via a linear stochastic model. This is a special case of the more general problem in which sensors cooperate to estimate a *global* vector parameter of interest, as the local parameters can be collected into a single vector parameter. Many distributed estimation algorithms for this problem have been investigated in the literature, including consensus strategies [12]–[15], the incremental least mean square (LMS) algorithm [16], [17], the distributed LMS consensus-based algorithm [18] and diffusion LMS strategies (see the excellent tutorial [19] and the references therein). The consensus strategies are relatively less computationally intensive for each sensor. However, their performance in terms of the convergence rate and mean-square deviation (MSD) are often not as good as the diffusion LMS strategies [20]. The incremental LMS algorithm requires that sensors set up a Hamiltonian path through the network, which is impractical for a large WSN.

To ensure mean and mean-square convergence, the diffusion LMS strategy requires that a pre-defined step size in each diffusion update is smaller than twice the reciprocal of the maximum eigenvalue of the system matrix covariance [19], [21]. Since this eigenvalue is not known a priori, a very small step size is thus typically chosen in the diffusion LMS strategy. However, this leads to slow convergence rates, resulting in higher communication costs. Furthermore, the diffusion strategy is not specifically designed to perform local sensor parameter estimation in a WSN. For example, when sensors need to estimate their own clock skews and offsets in network synchronization [22], the diffusion LMS strategy either requires that every node estimates the same global parameter, which is a collection of all the sensor local clock skews and offsets, or at least transmits estimates of the clock skews and offsets of all its neighbors (cf.

Section III-B for a detailed discussion). In both cases, communication cost for each sensor per iteration increases with the density of the network, and does not make full use of the broadcast nature of wireless communications in a WSN. The distributed LMS consensus-based algorithm of [18] requires the selection of a set of bridge sensors, and the passing of several messages between neighboring sensors, which again does not utilize the broadcast nature of a WSN. It is also more computationally complex than the diffusion algorithm but with better MSD performance.

We therefore ask if there exists a distributed estimation algorithm with similar MSD performance as the diffusion algorithms, and that allows sensors to *broadcast a fixed size message*, regardless of the network size or density, to all its neighbors? Our work suggests that the answer is affirmative for a somewhat more restrictive data model than that used in the LMS literature [19], and can be found in the Sum-Product Algorithm over a Wireless Network (SPAWN) method, first proposed by [11]. The SPAWN algorithm is based on belief propagation [23], [24]; the main differing characteristic is that the *same* message is broadcast to all neighboring nodes by each sensor, in contrast to traditional belief propagation where a different message is transmitted to each neighbor. The reference [11] however does not address the issue of error convergence in the SPAWN algorithm, and it is well known that loopy belief propagation may not converge [25]–[27].

In this paper, we consider an adaptive version of the Gaussian SPAWN (gSPAWN) algorithm for distributed local linear parameter estimation. We assume that sensors have only local observations with respect to its neighbors, such as the pairwise distance measurement, the relative temperature with respect to one another, and the relative clock offset between two nodes. Similar to [21], [28], [29], we assume that the observations follow a linear model. Our main contribution in this paper is the derivation of sufficient conditions for the mean and MSD convergence of the gSPAWN algorithm. Note that although the gSPAWN algorithm is based on Gaussian belief propagation, the methods of [26], [27] for analyzing the convergence of Gaussian belief propagation in a loopy graph do not apply due to the difference in messages transmitted by each node in gSPAWN versus that in Gaussian belief propagation. In fact, due to the broadcast nature of the messages in gSPAWN, our analysis is simpler than that in [26], [27].

As an example, we apply the gSPAWN algorithm to cooperative self-localization in non-line-of-sight (NLOS) multipath environments. To the best of our knowledge, most distributed localization methods [14], [30], [31] consider only line-of-sight (LOS) signals, because NLOS environments introduce non-linearities

in the system models, and measurement noises can no longer be modeled as Gaussian random variables. In this application, we assume that individual propagation paths can be resolved, and we adopt a ray tracing model to characterize the relationship between sensor locations, range and angle measurements [32]–[34]. When all scatterers in the environment are either parallel or orthogonal to each other, we show that the sensor location estimates given by the gSPAWN algorithm converges in the mean to the true sensor positions. We compare the performance of our algorithm with that of a peer-to-peer localization method and the diffusion LMS strategy, with numerical results suggesting that the gSPAWN algorithm has better average accuracy and convergence rate.

The rest of this paper is organized as follows. In Section II, we define the system model. In Section III, we describe the gSPAWN algorithm and compare it to the diffusion algorithm. We provide sufficient conditions for mean convergence and mean-square stability of the gSPAWN algorithm, and numerical comparison results in Section IV. We then show an application of the gSPAWN algorithm to network localization in NLOS environments in Section V. Finally, we summarize and conclude in Section VI.

*Notations:* We use upper-case letters to represent matrices and lower-case letters for vectors and scalars. Bold faced symbols are used to denote random variables. The conjugate of a matrix  $A$  is  $\bar{A}$ . The transpose and conjugate transpose of  $A$  are denoted as  $A^T$  and  $A^*$ , respectively. The minimum and maximum non-zero singular values of  $A$  are denoted as  $\underline{\delta}(A)$  and  $\bar{\delta}(A)$ , respectively. The maximum absolute eigenvalue or spectral radius of  $A$  is denoted as  $r(A)$ . When  $A$  is Hermitian, we have  $r(A) = \bar{\delta}(A)$ . We write  $A \succeq B$  and  $A \succ B$  if  $A - B$  is positive semi-definite and positive definite, respectively. If all entries of  $B - A$  are non-negative, we write  $A \leq B$ . The operation  $A \otimes B$  is the Kronecker product between  $A$  and  $B$ . The vector  $\text{vec}\{A\}$  is formed by stacking the columns of  $A$  together into a column vector. The matrix  $\text{diag}(A_1, \dots, A_n)$  is a block diagonal matrix consisting of the sub-matrices  $A_1, \dots, A_n$  on the main diagonal. The symbol  $I_m$  represents a  $m \times m$  identity matrix. We use  $0_m$  to denote a  $m \times 1$  vector of all zeroes. The operator  $\mathbb{E}$  denotes mathematical expectation. The density function of a multivariate Gaussian distribution with mean  $\mu$  and covariance  $P$  is given by  $\mathcal{N}(\cdot; \mu, P)$ .

## II. SYSTEM MODEL

In this section, we describe our system model, and present some assumptions that we make throughout this paper. We consider a network of  $n + 1$  sensors  $\{0, \dots, n\}$ , where each sensor  $i$  wants to estimate a  $d \times 1$  parameter  $s_i$ . Sensor 0 is set to be the reference node in the network, and  $s_0$  is a global reference

level for the other sensor parameters. For example, in a localization problem, we are often interested to find the relative locations of nodes in the network with respect to (w.r.t.) a reference node. In the sensor network synchronization problem, we want to estimate the relative clock offset and skew of each sensor w.r.t. to a reference sensor. In this paper, we use the terms “sensors” and “nodes” interchangeably. We say that sensors  $i$  and  $j$  are neighbors if they are able to communicate with each other. A network consisting of all sensors therefore corresponds to a graph  $\mathcal{G} = (\mathcal{E}, \mathcal{S})$ , where the set of vertices  $\mathcal{S} = \{0, \dots, n\}$ , and the set of edges  $\mathcal{E}$  consists of communication links in the network. We let the set of neighbors of sensor  $i$  be  $\mathcal{B}_i$  (which may include sensor  $i$  itself or not, depending on the application data model).

Sensors interact with neighbors and estimate their local parameters through in-network processing. In most applications, each sensor can obtain only local measurements w.r.t. its neighbors via communication between each other. Examples include the pseudo distance measurement between two sensors when performing wireless ranging, and the pairwise clock offset when performing clock synchronization. For each sensor  $i$ , and neighbor  $j \in \mathcal{B}_i$ , we consider the data model given by

$$\mathbf{d}_{ij}^{(l)} = \mathbf{G}_{ij} s_i - \mathbf{H}_{ij}^{(l)} s_j + \boldsymbol{\omega}_{ij}^{(l)}, \quad (1)$$

where  $\mathbf{d}_{ij}^{(l)}$  is the  $m \times 1$  vector measurement obtained at sensor  $i$  w.r.t. sensor  $j$  at time  $l$ ,  $\mathbf{G}_{ij}$  is a known  $m \times d$  system coefficient matrix,  $\mathbf{H}_{ij}^{(l)}$  is an observed  $m \times d$  regression matrix at time  $l$ , and  $\boldsymbol{\omega}_{ij}^{(l)}$  is a measurement noise of dimensions  $m \times 1$ . We note that this data model is a special case of the widely studied LMS data model (see e.g., [19]), since we can describe (1) by using a global parameter  $s = [s_0^T, \dots, s_n^T]^T$  with appropriate stacking of the measurements  $\mathbf{d}_{ij}^{(l)}$ . On the other hand, if every sensor  $i$  is interested in the same global parameter  $s_i = s$ , we have from (1) that  $\mathbf{d}_{ij}^{(l)} = \mathbf{U}_{ij}^{(l)} s + \boldsymbol{\omega}_{ij}^{(l)}$ , where  $\mathbf{U}_{ij}^{(l)} = \mathbf{G}_{ij} - \mathbf{H}_{ij}^{(l)}$  is the regression matrix in [19].<sup>1</sup> Although we have assumed for convenience that all quantities have the same dimensions across sensors, our work can be easily generalized to the case where measurements and parameters have different dimensions at each sensor. We have the following assumptions similar to [19] on our data model.

### Assumption 1

- (i) *The regression matrices  $\mathbf{H}_{ij}^{(l)}$  are independent over sensor indices  $i$  and  $j$ , and over time  $l$ . They are also independent of the measurement noises  $\boldsymbol{\omega}_{ij}^{(l)}$  for all  $i, j$ , and  $l$ . For all  $i, j$ ,  $\mathbf{H}_{ij}^{(l)}$  is stationary over  $l$ .*

<sup>1</sup>The data model in [19] assumes that  $m = 1$ , but can be easily extended to  $m > 1$ .

- (ii) The measurement noises  $\omega_{ij}^{(l)}$  are stationary with zero mean, and  $\mathbb{E}[\omega_{ij}^{(l)}(\omega_{ij}^{(l)})^*] = C_{ij}$ , where  $C_{ij}$  is a positive semi-definite matrix.
- (iii) For every  $i = 1, \dots, n$ , the matrix  $\sum_{j \in \mathcal{B}_i} G_{ij}^* G_{ij}$  is positive definite.

In the LMS framework, each sensor  $i$  seeks to estimate  $s_i$  so that the overall network mean square error,

$$\sum_{i=1}^n \sum_{j \in \mathcal{B}_i} \mathbb{E} \left\{ \left\| \mathbf{d}_{ij}^{(1)} - G_{ij} s_i + \mathbf{H}_{ij}^{(1)} s_j \right\|^2 \right\}$$

is minimized. Our goal is to design a distributed algorithm to perform local parameter estimation, which makes full use of the broadcast nature of the wireless medium over which sensors communicate. In particular, the messages broadcast by each sensor do not depend on the number of neighbors of the sensor or the network size. This is achieved by the gSPAWN algorithm. However, in order to ensure convergence of the gSPAWN algorithm, we need the following additional technical assumptions as well as Assumption 3, which will be discussed later in Section IV-B. This makes our system model more restrictive than that considered in [19].

### Assumption 2

- (i) The network graph  $\mathcal{G}$  is connected, and  $\mathcal{G} \setminus \{0\}$  consists of strongly connected components.<sup>2</sup>
- (ii) The reference sensor 0 has an a priori estimate  $\hat{s}_0 = s_0 + w_0$  of its parameter  $s_0$ , where  $w_0$  is a zero mean Gaussian random variable with covariance matrix  $\alpha_0 \mathbf{I}_d$ .
- (iii) Let  $\mathbf{H}_{ij} = \mathbb{E}[\mathbf{H}_{ij}^{(l)}]$  for all  $l \geq 1$ ,  $\eta_i^2 = \underline{\delta}(\sum_{j \in \mathcal{B}_i} G_{ij}^* G_{ij})$ , and  $\rho_i^2 = \max_{j \in \mathcal{B}_i} \bar{\delta}(\mathbf{H}_{ij} \mathbf{H}_{ij}^*)$ . For all  $i \geq 1$ , we have  $\rho_i^2 \leq \eta_i^2 / |\mathcal{B}_i|$  if  $|\mathcal{B}_i| \leq 2$ , and  $\rho_i^2 \leq \eta_i^2 / 3$  if  $|\mathcal{B}_i| \geq 3$ .

Assumption 2(i) does not result in much loss of generality. If the graph  $\mathcal{G}$  is not connected, the same analysis can be repeated on each connected component of  $\mathcal{G}$ . Furthermore, in most practical applications, sensor measurements are symmetric, i.e., if sensor  $i$  can obtain a measurement w.r.t. sensor  $j$ , then sensor  $j$  can also obtain a measurement w.r.t. sensor  $i$ . Assumption 2(ii) also holds in our applications of interest, where typically,  $\hat{s}_0 = s_0$  is taken to be a known reference value with  $\alpha_0 = 0$ . For example, when localizing sensors w.r.t. to the reference node 0, there is no loss in generality in assuming that the reference node is at  $\hat{s}_0 = s_0 = 0_d$ , the origin of the frame of reference. Assumption 2(iii) on the other hand restricts our

<sup>2</sup>The notation  $\mathcal{G} \setminus \{0\}$  denotes a graph with node 0 and its incident edges removed.

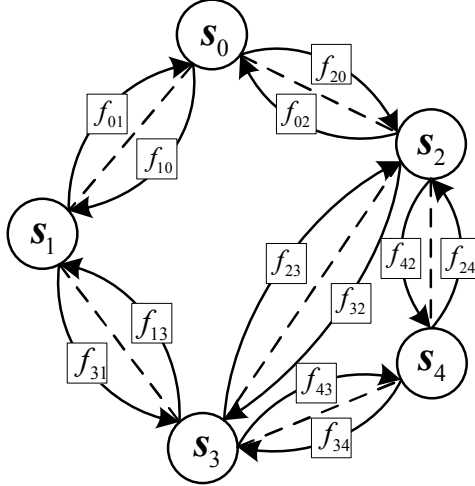


Fig. 1. A factor graph of a network with 5 sensors, where a dashed line indicates that a communication path exists between the corresponding two sensors, and the arrows indicate the direction of message flows.

data model to those in which the system coefficient matrices  $G_{ij}$  and expected regression matrices  $H_{ij}$  do not differ by too much.

### III. THE GAUSSAIN SPAWN ALGORITHM

In this section, we briefly review the SPAWN algorithm, and derive the adaptive gSPAWN algorithm. Since belief propagation is a Bayesian estimation approach, we will *temporarily* view the parameters  $s_1, \dots, s_n$  as random variables with uniform priors in this section. Suppose that we make the observations  $\mathbf{O}^{(l)} = (\mathbf{d}_{ij}^{(l)}, \mathbf{H}_{ij}^{(l)})_{i \geq 1, j \in \mathcal{B}_i}$  at a time  $l$ . In the Bayesian framework, we are interested to find values  $s_1, \dots, s_n$  that maximize the *a posteriori* probability

$$p(\{s_i\}_{i=1}^n \mid \mathbf{O}^{(l)}),$$

where the notation  $p(\mathbf{x} \mid \mathbf{y})$  denotes the conditional probability of  $\mathbf{x}$  given  $\mathbf{y}$ .<sup>3</sup> Since the distributions of  $\mathbf{d}_{ij}^{(l)}$  and  $\mathbf{d}_{ji}^{(l)}$  depend only on  $s_i$  and  $s_j$ , we have from (1) that

$$\begin{aligned} p(\{s_i\}_{i=1}^n \mid \mathbf{O}^{(l)}) &\propto \prod_{(i,j) \in \mathcal{E}} p(\mathbf{d}_{ij}^{(l)}, \mathbf{d}_{ji}^{(l)} \mid s_i, s_j, \mathbf{H}_{ij}^{(l)}, \mathbf{H}_{ji}^{(l)}) \\ &= \prod_{(i,j) \in \mathcal{E}} p(\mathbf{d}_{ij}^{(l)} \mid s_i, s_j, \mathbf{H}_{ij}^{(l)}) p(\mathbf{d}_{ji}^{(l)} \mid s_j, s_i, \mathbf{H}_{ji}^{(l)}). \end{aligned} \quad (2)$$

<sup>3</sup>Note the abuse of notation here:  $\mathbf{x}$  is used to denote both the random variable and its realization.

Let  $f_{ij}(s_i, s_j) = p(\mathbf{d}_{ij}^{(l)} | s_i, s_j, \mathbf{H}_{ij}^{(l)})$  and  $f_{ji}(s_i, s_j) = p(\mathbf{d}_{ji}^{(l)} | s_j, s_i, \mathbf{H}_{ji}^{(l)})$ . We construct a factor graph using  $\{s_i\}_{i=0}^n$  as the variable nodes, and for each pair of neighboring sensors  $i$  and  $j$  with  $(i, j) \in \mathcal{E}$ , we connect their corresponding variable nodes using the two factor nodes  $f_{ij}$  and  $f_{ji}$ . See Figure 1 for an example, where each random variable is represented by a circle and each factor node is represented by a square. We note that our factor graph construction is somewhat untypical compared to those used in traditional loopy belief propagation, where usually a single factor  $f_{ij}(s_i, s_j)f_{ji}(s_i, s_j)$  is used between the two variable nodes  $s_i$  and  $s_j$ . The reason we separate it into two factors is to allow us to design an algorithm that allows sensors to broadcast the same message to all neighboring nodes, as described below.

To apply the sum-product algorithm on a loopy factor graph, we need to define a message passing schedule [24]. We adopt a fully parallel schedule here. In each iteration, all factor nodes send messages to their neighboring variable nodes in parallel, followed by messages from variable nodes to neighboring factor nodes in parallel. We constrain the updates by allowing messages to flow in only one direction. Specifically, for every  $i$  and for all  $j \in \mathcal{B}_i$ ,  $s_i$  sends messages only to  $f_{ji}$  and not to  $f_{ij}$ , and  $f_{ij}$  sends messages only to  $s_i$  and not to  $s_j$ . The two types of messages sent between variable and factor nodes at the  $k$ th iteration are the following:

- Each factor node  $f_{ij}$  passes a message  $h_{f_{ij} \rightarrow s_i}^{(k)}(s_i)$  to the variable node  $s_i$  representing  $f_{ij}$ 's belief of  $s_i$ 's state. This message is given by

$$h_{f_{ij} \rightarrow s_i}^{(k)}(s_i) = \int p(\mathbf{d}_{ij}^{(l)} | s_i, s_j, \mathbf{H}_{ij}^{(l)}) b_j^{(k-1)}(s_j) ds_j. \quad (3)$$

- Each variable node  $s_i$  broadcasts a belief to all neighboring factor nodes  $\{f_{ji}\}_{j \in \mathcal{B}_i}$ , and its belief is given by

$$b_i^{(k)}(s_i) = \prod_{j \in \mathcal{B}_i} h_{f_{ij} \rightarrow s_i}^{(k)}(s_i). \quad (4)$$

The above scheme is called the SPAWN algorithm by [11]. Note that the same belief message from the variable node  $s_i$  is broadcast to all factor nodes  $\{f_{ji}\}_{j \in \mathcal{B}_i}$ , since node  $s_i$  receives only from factor node  $f_{ij}$  and the belief  $b_i^{(k)}(s_i)$  does not include any messages from  $f_{ji}$  in the previous iterations. The communication cost therefore *does not depend on the size or density of the network*, making this scheme more suitable for a WSN. In the following, we derive the exact messages passed when  $p(\mathbf{d}_{ij}^{(l)} | s_i, s_j, \mathbf{H}_{ij}^{(l)})$ , for all  $i, j$ , are assumed to be Gaussian distributions. In addition, we allow the observations to be updated at each iteration so that the algorithm becomes adaptive in nature. We call this the gSPAWN algorithm.



Although Gaussian distributions are assumed in the derivation of the gSPAWN algorithm, we show in Section IV that this assumption is not required for mean and MSD convergence of the algorithm.<sup>4</sup>

#### A. Derivation of Beliefs and Messages for gSPAWN

We first consider the message from the reference node 0 to a neighboring node  $i$ . The reference node's belief is held constant over all iterations. From Assumption 2(ii), we have  $b_0^{(l)}(s_0) = \mathcal{N}(s_0; \boldsymbol{\mu}_0^{(l)}, \mathbf{P}_0^{(l)})$ , where  $\boldsymbol{\mu}_0^{(l)} = \hat{s}_0$  and  $\mathbf{P}_0^{(l)} = \alpha_0 \mathbf{I}_d$ , for all  $l \geq 0$ . The message from the factor  $f_{i0}$  to sensor  $i$  is given by

$$\begin{aligned} h_{f_{i0} \rightarrow s_i}^{(l)}(s_i) &= \int p(\mathbf{d}_{i0}^{(l)} | s_i, s_0, \mathbf{H}_{i0}^{(l)}) b_0^{(l)}(s_0) ds_0 \\ &\propto \mathcal{N}(G_{i0} s_i; \boldsymbol{\nu}_{i0}^{(l)}, \Pi_{i0}^{(l)}). \end{aligned} \quad (5)$$

where  $\boldsymbol{\nu}_{i0}^{(l)} = \mathbf{d}_{i0}^{(l)} + \mathbf{H}_{i0}^{(l)} \boldsymbol{\mu}_0^{(l)}$ ,  $\Pi_{i0}^{(l)} = \mathbf{C}_{i0} + \mathbf{H}_{i0} \mathbf{P}_0^{(l)} \mathbf{H}_{i0}^*$ , and the symbol  $\propto$  means approximately proportional to. The approximation in (5) arises because we have replaced  $\mathbf{H}_{i0}^{(l)}$  with  $\mathbf{H}_{i0}$  in  $\Pi_{i0}^{(l)}$ .

For all nodes  $i$  other than the reference node 0, we set the initial belief  $b_i^{(0)}(s_i)$  of node  $i$  to be a Gaussian distribution with mean  $\boldsymbol{\mu}_i^{(0)} = \mathbf{0}_d$  and covariance

$$\mathbf{P}_i^{(0)} = \begin{cases} 4\alpha \mathbf{I}_d, & \text{if } |\mathcal{B}_i| = 1, \\ 3\alpha \mathbf{I}_d, & \text{if } |\mathcal{B}_i| = 2, \\ 5\alpha/3 \mathbf{I}_d, & \text{otherwise,} \end{cases} \quad (6)$$

where  $\alpha$  is a positive number chosen to be sufficiently large so that  $\alpha \geq \alpha_0$ , and  $\alpha \geq \frac{\max_{i,j} r(\mathbf{C}_{ij})}{\min_i \rho_i^2}$ , where  $\rho_i^2 = \max_{j \in \mathcal{B}_i} \bar{\delta}(\mathbf{H}_{ij} \mathbf{H}_{ij}^*)$  (cf. Assumption 2(iii)). A larger initial variance is used for those nodes with less neighbors because such nodes have relatively less information than other nodes in the network. We will see later in the proof of Theorem 1 that the chosen multipliers for the cases where there are not more than two neighbors are the right ones.

Consider the  $l$ th iteration of the gSPAWN algorithm. After  $l - 1$  iterations, the belief of node  $j$  is  $b_j^{(l-1)}(s_j) = \mathcal{N}(s_j; \boldsymbol{\mu}_j^{(l-1)}, \mathbf{P}_j^{(l-1)})$ . The message  $b_j^{(l-1)}(s_j)$  is a Gaussian distribution, so only the mean  $\boldsymbol{\mu}_j^{(l-1)}$  and covariance matrix  $\mathbf{P}_j^{(l-1)}$  need to be passed to node  $i$ , which then computes

$$\begin{aligned} h_{f_{ij} \rightarrow s_i}^{(l)}(s_i) &= \int p(\mathbf{d}_{ij}^{(l)} | s_i, s_j, \mathbf{H}_{ij}^{(l)}) b_j^{(l-1)}(s_j) ds_j \\ &\propto \mathcal{N}(G_{ij} s_i; \boldsymbol{\nu}_{ij}^{(l-1)}, \Pi_{ij}^{(l-1)}), \end{aligned}$$

<sup>4</sup>This is analogous to the philosophy of using the B.L.U.E. for parameter estimation with non-Gaussian system models.

where

$$\boldsymbol{\nu}_{ij}^{(l-1)} = \mathbf{d}_{ij}^{(l)} + \mathbf{H}_{ij}^{(l)} \boldsymbol{\mu}_j^{(l-1)}, \quad (7)$$

$$\Pi_{ij}^{(l-1)} = \mathbf{C}_{ij} + \mathbf{H}_{ij} \mathbf{P}_j^{(l-1)} \mathbf{H}_{ij}^*. \quad (8)$$

We obtain the belief of node  $i$  at the  $l$ th iteration from (4). Since all the messages in the product in (4) are Gaussian distributions, we have  $b_i^{(l)}(s_i) = \mathcal{N}(s_i; \boldsymbol{\mu}_i^{(l)}, \mathbf{P}_i^{(l)})$  with

$$\mathbf{P}_i^{(l)} = \left( \sum_{j \in \mathcal{B}_i} \mathbf{G}_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \mathbf{G}_{ij} \right)^{-1} \quad (9)$$

and

$$\boldsymbol{\mu}_i^{(l)} = \mathbf{P}_i^{(l)} \sum_{j \in \mathcal{B}_i} \mathbf{G}_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \boldsymbol{\nu}_{ij}^{(l-1)}. \quad (10)$$

We will show in Lemma 3 that  $\mathbf{P}_j^{(l)}$  is positive definite for all  $j = 1, \dots, n$  and  $l \geq 0$ , so (9) is well defined.

The gSPAWN algorithm is formally presented in Algorithm 1. Note that factor nodes are virtual, and are introduced only to facilitate the derivation of the algorithm. In practice, the update is performed at each individual node directly. At the end of  $l$  iterations, node  $i$  estimates its parameter  $s_i$  by maximizing the belief  $b_i^{(l)}(s_i)$  with respect to  $s_i$ . Since  $b_i^{(l)}(s_i)$  is a Gaussian distribution, the local estimator for  $s_i$  is given by  $\hat{s}_i^{(l)} = \boldsymbol{\mu}_i^{(l)}$ , and has a covariance of  $\mathbf{P}_i^{(l)}$ .

### B. Comparison with Diffusion Adapt-Then-Combine

The gSPAWN algorithm is reminiscent of the Adapt-Then-Combine (ATC) diffusion algorithm, first proposed by [28]. We present the ATC algorithm in our context of local parameter estimation,<sup>5</sup> and discuss the differences with the gSPAWN algorithm. As the ATC algorithm assumes that the data model at every sensor is based on a common global parameter, we stack the local sensor parameters into a single global parameter  $s = [s_0^T, \dots, s_n^T]^T$ , with the data model at sensor  $i$  given by

$$\mathbf{d}_i^{(l)} = \mathbf{W}_i^{(l)} s + (\boldsymbol{\omega}_i^{(l)})^T, \quad (11)$$

<sup>5</sup>We consider only the version of ATC in which there is no exchange of sensor observations with neighbors. We also do not discuss the Combine-Then-Adapt diffusion algorithm as this is known to perform worse than ATC in general [28].

---

**Algorithm 1** gSPAWN: Gaussian Sum-Product Algorithm over a Wireless Network
 

---

1: **Initialization:**

2: For the reference node 0, set  $\boldsymbol{\mu}_0^{(l)} = \hat{s}_0$  and  $\mathbf{P}_0^{(l)} = \alpha_0 \mathbf{I}_d$ , for all  $l \geq 0$ .

3: For  $i = 1, \dots, n$ , let  $\boldsymbol{\mu}_i^{(0)} = \mathbf{0}_d$  and choose  $\mathbf{P}_i^{(0)}$  according to (6).

4: **for** the  $l^{\text{th}}$  iteration **do**

5:   **nodes**  $i = 1 : n$  **in parallel**

6:   broadcast the current belief  $b_i^{(l-1)}(s_i)$  to neighboring nodes;

7:   receive  $b_j^{(l-1)}(s_j)$  from neighboring nodes  $j$ , where  $j \in \mathcal{B}_i$ ;

8:   update its belief as  $b_i^{(l)}(s_i) \sim \mathcal{N}(s_i; \boldsymbol{\mu}_i^{(l)}, \mathbf{P}_i^{(l)})$  with

$$\mathbf{P}_i^{(l)} = \left( \sum_{j \in \mathcal{B}_i} \mathbf{G}_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \mathbf{G}_{ij} \right)^{-1},$$

and

$$\boldsymbol{\mu}_i^{(l)} = \mathbf{P}_i^{(l)} \sum_{j \in \mathcal{B}_i} \mathbf{G}_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \boldsymbol{\nu}_{ij}^{(l-1)},$$

where  $\boldsymbol{\nu}_{ij}^{(l-1)}$  and  $\Pi_{ij}^{(l-1)}$  are given in (7) and (8) respectively.

9:   **end parallel**

10: **end for**

---

where  $\boldsymbol{\omega}_i^{(l)} = \left[ (\boldsymbol{\omega}_{i1}^{(l)})^T, \dots, (\boldsymbol{\omega}_{in}^{(l)})^T \right]$ , and  $\mathbf{W}_i^{(l)}$  is a  $(n+1) \times (n+1)$  block matrix consisting of  $d \times d$  blocks, with the  $(j, k)$ -th block for  $0 \leq j, k \leq n$  being

$$\mathbf{W}_i^{(l)}(j, k) = \begin{cases} \mathbf{G}_{ij}, & \text{if } k = i \text{ and } j \in \mathcal{B}_i \setminus \{i\}, \\ \mathbf{G}_{ii} - \mathbf{H}_{ii}^{(l)}, & \text{if } k = i \text{ and } j = i \in \mathcal{B}_i, \\ -\mathbf{H}_{ij}^{(l)}, & \text{if } k = j \text{ and } j \in \mathcal{B}_i \setminus \{i\}, \\ 0 \cdot \mathbf{I}_d, & \text{otherwise.} \end{cases}$$

The ATC update at time  $l$  at sensor  $i$  consists of the following steps:

$$\boldsymbol{\psi}_i^{(l)} = \mathbf{s}_i^{(l-1)} + \xi_i \cdot (\mathbf{W}_i^{(l)})^* (\mathbf{d}_i^{(l)} - \mathbf{W}_i^{(l)} \mathbf{s}_i^{(l-1)}), \quad (12)$$

$$\mathbf{s}_i^{(l)} = \sum_{j \in \mathcal{B}_i} a_{ij} \boldsymbol{\psi}_j^{(l)}, \quad (13)$$

where  $\boldsymbol{\psi}_i^{(l)}$  and  $\mathbf{s}_i^{(l)}$  are the intermediate and local estimates of  $s$  at sensor  $i$  respectively,  $\xi_i$  is a step size chosen to be sufficiently small, and  $(a_{ij})$  are non-negative weights with  $\sum_j a_{ij} = 1$  for all  $i$ . The

following remarks summarize the major differences of gSPAWN and ATC.

- (i) In the  $l$ -th iteration of the ATC update (12)-(13), each sensor  $i$  first computes  $\psi_i^{(l)}$  and broadcasts it so that its neighbors can perform the combination step (13). The message  $\psi_i^{(l)}$  is an intermediate estimate of the global parameter  $s$ , and is of size  $dn \times 1$ . Each sensor  $i$  needs to know the total number of nodes  $n$  in the network, which determines the size of its broadcast message. Its communication cost therefore increases linearly with  $n$ . In contrast, in the gSPAWN algorithm, sensor  $i$  broadcasts a fixed size message  $(\mu_i^{(l)}, P_i^{(l)})$ , which does not depend on the size of the network.
- (ii) Alternatively, instead of broadcasting the message  $\psi_i^{(l)}$  in full, sensor  $i$  can choose to broadcast only those components of  $\psi_i^{(l)}$  that it updates, which correspond to the parameters  $\{s_j : j \in \mathcal{B}_i\}$ . In this case, the sensor has to add identity headers in its broadcast message so that the receiving sensors know which parameters each component corresponds to. In a static network, this can be done once but in a mobile network, the header information has to be re-broadcast each time the network topology changes. The message itself also has size linear in the number of neighbors of the sensor. Therefore, the communication cost per sensor per iteration increases with the density of the network. We note that the communication cost for a network using gSPAWN does not depend on the network node density.
- (iii) In the gSPAWN algorithm, the matrix weights in (10) need not be non-negative, unlike those in (13). Moreover, the weights in (10) evolve over the iterations, and reflect each sensor's confidence in its belief. The weights for the ATC algorithm we have presented are fixed, although adaptive combination weights for ATC have also been proposed in [19] but no convergence analysis is available.
- (iv) In (12), to ensure mean and mean-square convergence, the step size  $\xi_i$  is chosen so that

$$0 < \xi_i < \frac{2}{r(\mathbf{R}_i)}, \quad (14)$$

where  $\mathbf{R}_i = \mathbb{E}[\mathbf{W}_i^* \mathbf{W}_i]$  [21]. Since  $r(\mathbf{R}_i)$  is in general not known a priori, the step size  $\xi_i$  is chosen to be very small in most applications. The convergence rate of the ATC algorithm is controlled by the step size, with a smaller step size leading to a lower convergence rate. In the gSPAWN algorithm, the parameter  $\alpha$  in (6) plays an analogous role, but it, in effect, does not control the convergence rate since it appears in both  $\Pi_{ij}^{(l-1)}$  and  $P_i^{(l)}$  in (10), and can be chosen as large as desired (see Section III-A).

- (v) In Section IV, we show the mean convergence and mean-square stability of gSPAWN under some

technical assumptions, which make our model less general than the LMS model used in the ATC algorithm. The advantages of gSPAWN over ATC as described above are only valid under these assumptions. Nevertheless, these assumptions hold for several important sensor network applications including distributed sensor localization, which we discuss in Section V, and distributed sensor clock synchronization [22]. We also remark that gSPAWN is more suitable for *local* parameter estimation, while ATC has lower communication costs in the case of *global* network parameter estimation if the number of parameters is fixed, since gSPAWN needs to broadcast covariance information as well as parameter estimates.

#### IV. CONVERGENCE ANALYSIS OF gSPAWN

In this section, we show that the covariance matrices  $P_i^{(l)}$  in the gSPAWN algorithm converge, and establish sufficient conditions for the belief means to converge to the true parameter values. We also show that the MSD converges under additional assumptions. Although the gSPAWN algorithm follows the general framework of belief propagation, its message passing is different from traditional belief propagation. Moreover, we consider an adaptive version of the algorithm in this paper. Therefore, the general convergence analysis in [25]–[27] cannot be applied here.

##### A. Convergence of Covariance Matrices

We first state several elementary results. The first lemma is from [35], and shows that a non-increasing sequence of positive definite matrices (in the order defined by  $\succeq$ ) converges. The proof of Lemma 2 can be found in [36].

**Lemma 1** *If the sequence  $\{X^{(l)}\}_{l=1}^{+\infty}$  of positive definite matrices is non-increasing, i.e.,  $X^{(l)} \succeq X^{(l+1)}$  for  $l = 1, 2, \dots$ , then this sequence converges to a positive semi-definite matrix.*

**Lemma 2** *If the matrices  $X$  and  $Y$  are positive definite, then  $X \succeq Y$  if and only if  $Y^{-1} \succeq X^{-1}$ .*

**Lemma 3** *Suppose that Assumption 1(iii) holds. For all  $i = 1, \dots, N$ , and for all  $l \geq 0$ ,  $P_i^{(l)}$  is positive definite.*

*Proof:* We prove the claim by induction on  $l$ . From (6), the claim holds for  $l = 0$ . Suppose the claim holds for  $l - 1$ , we have that  $\Pi_{ij}^{(l-1)}$  is positive definite for all  $j \in \mathcal{B}_i$ . Therefore, the inverse  $(\Pi_{ij}^{(l-1)})^{-1}$

is also positive definite, and from (9) and Assumption 1(iii), since  $\sum_{j \in \mathcal{B}_i} G_{ij}^* G_{ij}$  has full rank, we obtain that  $P_i^{(l)}$  is positive definite. The induction is now complete, and the lemma is proved.  $\blacksquare$

The following result shows that the covariance matrices of the beliefs at each variable node in the factor graph converges.

**Theorem 1** *Suppose that Assumptions 1 and 2 hold. Then, the covariance matrices  $P_i^{(l)}$  at every node  $i$  in the gSPAWN algorithm converges, i.e., there exists unique positive semi-definite matrices  $\{P_i^{(\infty)}\}_{i=1}^n$  such that for all  $i = 1, \dots, n$ ,*

$$\lim_{l \rightarrow \infty} P_i^{(l)} = P_i^{(\infty)}. \quad (15)$$

*Proof:* All nodes are updated using (9) and (10). Since the covariance matrix  $C_{ij}$  is positive definite, we can perform eigen decomposition on  $C_{ij}$ . Let  $C_{ij} = V_{ij}^* D_{ij} V_{ij}$ , where  $V_{ij}$  is a unitary matrix, and  $D_{ij}$  is a diagonal matrix with positive entries. Let  $K_i^{(l)} = \left(P_i^{(l)}\right)^{-1}$ . From (8) and (9), it can be shown that

$$K_i^{(l)} = \sum_{j \in \mathcal{B}_i} G_{ij}^* V_{ij}^* \left( D_{ij} + V_{ij} H_{ij} P_j^{(l-1)} H_{ij}^* V_{ij}^* \right)^{-1} V_{ij} G_{ij}. \quad (16)$$

Further performing singular value decomposition on  $G_{ij}$  and  $H_{ij}$ , we have  $G_{ij} = U_{ij} A_{ij} \tilde{U}_{ij}^*$  and  $H_{ij} = W_{ij} B_{ij} \tilde{W}_{ij}^*$ , where  $U_{ij}$ ,  $\tilde{U}_{ij}$ ,  $W_{ij}$ , and  $\tilde{W}_{ij}$  are unitary matrices, and  $A_{ij}$  and  $B_{ij}$  are diagonal matrices with corresponding singular values on the diagonal. We have,

$$K_i^{(1)} = \sum_{j \in \mathcal{B}_i} \tilde{U}_{ij} A_{ij}^* U_{ij}^* V_{ij}^* \left( D_{ij} + V_{ij} W_{ij} B_{ij} P_j^{(l-1)} B_{ij}^* W_{ij}^* V_{ij}^* \right)^{-1} V_{ij} U_{ij} A_{ij} \tilde{U}_{ij}^*. \quad (17)$$

We show by induction on  $l$  that  $K_i^{(l)}$  is non-decreasing for all  $i = 0, \dots, N$ . Since the reference node 0 does not update its belief, this clearly holds for  $i = 0$ . In the following, we prove the claim for  $i = 1, \dots, n$ . Since  $\alpha \geq \max_{i,j} r(D_{ij})/\rho_i^2$ , it follows that  $D_{ij} \preceq \alpha \rho_i^2 I_m$  for all  $i, j$ . From Assumption 2(iii), we also have  $\bar{\delta}(H_{ij} H_{ij}^*) \leq \rho_i^2$ , and hence  $B_{ij} B_{ij}^* \preceq \rho_i^2 I_m$  for all  $i, j$ . Using these two properties, we now consider different cases for the size of  $\mathcal{B}_i$ .

Case  $|\mathcal{B}_i| = 1$ : From Assumption 2(iii), we have  $\rho_i^2 \leq \eta_i^2$ . We consider the cases where the reference node 0 is a neighbor of node  $i$  or not separately. Suppose that the node 0 is the neighbor of node  $i$ . Then,

since  $P_0^{(0)} = \alpha_0 I_2$ , we have from (17) that

$$\begin{aligned} K_i^{(1)} &\succeq \frac{\eta_i^2}{(\alpha + \alpha_0)\rho_i^2} I_d \\ &\succeq \frac{\eta_i^2}{2\alpha\rho_i^2} I_d \\ &\succeq \frac{1}{4\alpha} I_d = K_i^{(0)}, \end{aligned}$$

where the second inequality follows from  $\alpha_0 \leq \alpha$ . If node 0 is not a neighbor of node  $i$ , suppose that  $\mathcal{B}_i = \{j\}$ , where  $j \neq 0$ . From Assumption 2(i), we have  $j \neq i$ , and  $|\mathcal{B}_j| \geq 2$ , yielding

$$\begin{aligned} K_i^{(1)} &\succeq \frac{\eta_i^2}{\alpha(\rho_i^2 + 3\rho_j^2)} I_d \\ &\succeq \frac{1}{4\alpha} I_d = K_i^{(0)}. \end{aligned}$$

Case  $|\mathcal{B}_i| = 2$ : From Assumption 2(iii), we have  $\rho_i^2 \leq \eta_i^2/2$ . Denote  $\mathcal{B}_i = \{j_1, j_2\}$ , we have  $|\mathcal{B}_{j_1}| = 1$  and  $|\mathcal{B}_{j_2}| = 2$  in the worst case, and it follows from (17) that

$$\begin{aligned} K_i^{(1)} &\succeq \frac{1}{\alpha} \frac{G_{ij_1}^* G_{ij_1}}{\rho_i^2 + 4\rho_i^2} + \frac{1}{\alpha} \frac{G_{ij_2}^* G_{ij_2}}{\rho_i^2 + 3\rho_i^2} \\ &= \frac{1}{\alpha} \frac{4G_{ij_1}^* G_{ij_1} + 5G_{ij_2}^* G_{ij_2}}{20\rho_i^2} \\ &\succeq \frac{1}{\alpha} \frac{\eta_i^2}{5\rho_i^2} I_d \\ &\succeq \frac{1}{3\alpha} I_d = K_i^{(0)}. \end{aligned}$$

Case  $|\mathcal{B}_i| \geq 3$ : From (17), we have

$$K_i^{(1)} \succeq \sum_{j \in \mathcal{B}_i} \tilde{U}_{ij} A_{ij}^* U_{ij}^* V_{ij}^* (D_{ij} + 4\alpha V_{ij} W_{ij} B_{ij} B_{ij}^* W_{ij}^* V_{ij}^*)^{-1} V_{ij} U_{ij} A_{ij} \tilde{U}_{ij}^*.$$

From Assumption 2(iii), we have  $\rho_i^2 \leq \eta_i^2/3$ , and we obtain

$$\begin{aligned} K_i^{(1)} &\succeq \frac{\sum_{j \in \mathcal{B}_i} \tilde{U}_{ij} A_{ij}^* A_{ij} \tilde{U}_{ij}^*}{5\alpha\rho_i^2} \\ &\succeq \frac{1}{\alpha} \frac{\eta_i^2}{5\rho_i^2} I_d \\ &\succeq \frac{3}{5\alpha} I_d = K_i^{(0)}. \end{aligned}$$

Therefore, we have shown that  $K_i^{(1)} \succeq K_i^{(0)}$  for all nodes  $i \geq 1$ .

Suppose that  $K_j^{(l)} \succeq K_j^{(l-1)}$  for all  $j$ . From Lemmas 2 and 3, we have  $P_j^{(l-1)} \succeq P_j^{(l)}$ , and it follows that  $V_{ij} H_{ij} P_j^{(l-1)} H_{ij}^* V_{ij}^* \succeq V_{ij} H_{ij} P_j^{(l)} H_{ij}^* V_{ij}^*$ . Applying Lemma 2, we have

$$\left( D_{ij} + V_{ij} H_{ij} P_j^{(l)} H_{ij}^* V_{ij}^* \right)^{-1} \succeq \left( D_{ij} + V_{ij} H_{ij} P_j^{(l-1)} H_{ij}^* V_{ij}^* \right)^{-1},$$

which together with (16) yields

$$K_i^{(l+1)} \succeq K_i^{(l)}.$$

This completes the induction, and the claim that  $K_i^{(l)}$  is non-decreasing in  $l$  for each  $i$  is now proven. This implies that  $P_i^{(l)}$  is non-increasing in  $l$ . The theorem now follows from Lemma 1, and the proof is complete.  $\blacksquare$

Theorem 1 shows that the covariance matrices at every node converges to the positive semi-definite matrices  $\{P_i^{(\infty)}\}_{i=1}^n$ , which is the fixed point solution to the set of equations

$$\left(P_i^{(\infty)}\right)^{-1} = \sum_{j \in \mathcal{B}_i} G_{ij}^* \left(C_{ij} + H_{ij} P_j^{(\infty)} H_{ij}^*\right)^{-1} G_{ij}, \quad (18)$$

for all  $i \geq 1$ , with  $P_0^{(\infty)} = \alpha_0 I_d$ .

### B. Convergence of Belief Means

In this subsection, we provide a technical assumption that together with Assumptions 1 and 2 ensure that the gSPAWN algorithm converges in mean. We then provide sufficient conditions that satisfy the given technical assumption. Using the covariance matrices  $\{P_i^{(\infty)}\}_{i=0}^n$ , which can be derived using (18), we let  $Q^{(\infty)}$  be a  $dn \times dn$  matrix consisting of  $n \times n$  blocks of  $d \times d$  submatrices, where for  $1 \leq i, j \leq n$ , the  $(i, j)$ -th block is

$$Q^{(\infty)}(i, j) = \begin{cases} P_i^{(\infty)} G_{ij}^* \left(C_{ij} + H_{ij} P_j^{(\infty)} H_{ij}^*\right)^{-1} H_{ij}, & \text{if } j \in \mathcal{B}_i, \\ 0 \cdot I_d, & \text{otherwise.} \end{cases} \quad (19)$$

**Assumption 3** *The spectral radius  $r(Q^{(\infty)})$  is less than 1.*

Assumption 3, together with Assumptions 1 and 2, are sufficient for the mean convergence (Theorem 2) of the gSPAWN algorithm. If the observations  $(\mathbf{d}_{ij}^{(l)}, \mathbf{H}_{ij}^{(l)})_{i,j,l}$  are scalars, then it can be shown that Assumption 3 holds under quite general conditions, see [22]. However, for vector observations,  $Q^{(\infty)}$  in general does not have spectral radius less than 1 even when  $H_{ij} = G_{ij}$  for all  $i, j$  (a numerical example is provided in Appendix A). To verify Assumption 3 requires a priori knowledge of the sensor network architecture, which may not be practical. To overcome this difficulty, we present local sufficient conditions in the following proposition that ensure Assumption 3 is satisfied.



**Proposition 1** *Suppose that Assumptions 1 and 2 hold. In addition, there exists a unitary matrix  $U$  so that for all  $i, j = 0, \dots, n$ , there exists a unitary matrix  $V_{ij}$  such that*

- (i)  $G_{ij} = V_{ij}A_{ij}U^*$ , where  $A_{ij}$  is a diagonal matrix with non-negative real numbers on the diagonal;
- (ii)  $H_{ij} = V_{ij}B_{ij}U^*$ , where  $B_{ij}$  is a diagonal matrix with non-negative real numbers on the diagonal, with  $B_{ij} \leq A_{ij}$ ; and
- (iii)  $C_{ij} = V_{ij}D_{ij}V_{ij}^*$ , where  $D_{ij}$  is a diagonal matrix with positive real numbers on the diagonal.

Then, Assumption 3 holds.

*Proof:* We first show, by induction on  $l$ , that for all  $i \geq 0$  and  $l \geq 0$ ,  $P_i^{(l)}$  has the form  $U\Lambda_i^{(l)}U^*$ , where  $\Lambda_i^{(l)}$  is a non-negative diagonal matrix. This is trivially true for all  $i \geq 0$  when  $l = 0$ . It is also true for  $i = 0$  and for all  $l \geq 1$  since the reference node 0 does not update its belief. Suppose that the claim holds for  $l - 1$ . Then we have for  $i \geq 1$ ,

$$\begin{aligned} \left(P_i^{(l)}\right)^{-1} &= \sum_{j \in \mathcal{B}_i} G_{ij}^* \left(C_{ij} + H_{ij}P_j^{(l-1)}H_{ij}^*\right)^{-1} G_{ij} \\ &= U \underbrace{\sum_{j \in \mathcal{B}_i} A_{ij} \left(D_{ij} + B_{ij}\Lambda_j^{(l-1)}B_{ij}\right)^{-1} A_{ij}}_{\triangleq \Lambda_i^{(l)}} U^*, \end{aligned} \quad (20)$$

so the claim holds since  $A_{ij} \left(D_{ij} + B_{ij}\Lambda_j^{(l-1)}B_{ij}\right)^{-1} A_{ij}$  is a diagonal matrix for all  $i, j$ . This implies that for all  $i \geq 0$ ,  $P_i^{(\infty)} = U\Lambda_i^{(\infty)}U^*$ , for some non-negative diagonal matrix  $\Lambda_i^{(\infty)}$  as  $l \rightarrow \infty$ . We hence have from (19) for  $j \in \mathcal{B}_i$ ,

$$\begin{aligned} Q^{(\infty)}(i, j) &= P_i^{(\infty)}G_{ij}^* \left(C_{ij} + H_{ij}P_j^{(\infty)}H_{ij}^*\right)^{-1} H_{ij} \\ &= U \left(\Lambda_i^{(\infty)}\right)^{-1} A_{ij} \left(D_{ij} + B_{ij}\Lambda_j^{(\infty)}B_{ij}\right)^{-1} B_{ij}U^*. \end{aligned} \quad (21)$$

Let  $\tilde{Q}$  be a  $dn \times dn$  matrix consisting of  $n \times n$  blocks of  $d \times d$  submatrices, where the  $(i, j)$ -th block is  $(\Lambda_i^{(\infty)})^{-1}A_{ij}(D_{ij} + B_{ij}\Lambda_j^{(\infty)}B_{ij})^{-1}B_{ij}$  for  $1 \leq i, j \leq n$  and  $j \in \mathcal{B}_i$ . Then from (21),  $r(Q^{(\infty)}) = r(\tilde{Q})$ .

We have

$$\sum_{\substack{j \in \mathcal{B}_i \\ j \neq 0}} \tilde{Q}(i, j) \leq \left(\Lambda_i^{(\infty)}\right)^{-1} \sum_{\substack{j \in \mathcal{B}_i \\ j \neq 0}} A_{ij} \left(D_{ij} + B_{ij}\Lambda_j^{(\infty)}B_{ij}\right)^{-1} A_{ij}. \quad (22)$$

From (20), the right hand side (R.H.S.) of (22) is equal to  $I_d$  if  $0 \notin \mathcal{B}_i$ , and  $\prec I_d$  if  $0 \in \mathcal{B}_i$ . This implies that  $\tilde{Q}$  is a non-negative sub-stochastic matrix. From Assumption 2,  $\tilde{Q}$  can be expressed, by relabeling the nodes if necessary, as a block diagonal matrix where each diagonal block is irreducible (when the reference

node 0 is removed from the network, the network breaks into irreducible components represented by each diagonal block). Applying the Perron-Frobenius theorem [37] on each block, we obtain the proposition. ■

Let  $\tilde{\boldsymbol{\mu}}_i^{(l)} = \boldsymbol{\mu}_i^{(l)} - s_i$ . Substituting (7) and (8) into (10), we obtain

$$\tilde{\boldsymbol{\mu}}_i^{(l)} = P_i^{(l)} \sum_{j \in \mathcal{B}_i} G_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \mathbf{H}_{ij}^{(l)} \tilde{\boldsymbol{\mu}}_j^{(l-1)} + P_i^{(l)} \sum_{j \in \mathcal{B}_i} G_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \boldsymbol{\omega}_{ij}^{(l)}.$$

Denote  $\tilde{\boldsymbol{\mu}}^{(l)} = \left[ \tilde{\boldsymbol{\mu}}_1^{(l)}, \dots, \tilde{\boldsymbol{\mu}}_n^{(l)} \right]^T$ , and  $\boldsymbol{\Omega}^{(l)} = \left[ \boldsymbol{\omega}_1^{(l)}, \dots, \boldsymbol{\omega}_n^{(l)} \right]^T$ , where  $\boldsymbol{\omega}_i^{(l)} = \left[ (\boldsymbol{\omega}_{i1}^{(l)})^T, \dots, (\boldsymbol{\omega}_{in}^{(l)})^T \right]$ , we have

$$\tilde{\boldsymbol{\mu}}^{(l)} = \mathbf{Q}^{(l)} \tilde{\boldsymbol{\mu}}^{(l-1)} + \mathbf{R}^{(l)} \boldsymbol{\Omega}^{(l)}, \quad (23)$$

where  $\mathbf{Q}^{(l)}$  is a  $dn \times dn$  matrix consisting of  $n \times n$  blocks of  $d \times d$  submatrices, with the  $(i, j)$ -th block for  $1 \leq i, j \leq n$  being

$$\mathbf{Q}^{(l)}(i, j) = \begin{cases} P_i^{(l)} G_{ij}^* \left( \Pi_{ij}^{(l-1)} \right)^{-1} \mathbf{H}_{ij}^{(l)}, & \text{if } j \in \mathcal{B}_i, \\ 0 \cdot \mathbf{I}_d, & \text{otherwise.} \end{cases} \quad (24)$$

The matrix  $\mathbf{R}^{(l)} = \mathbf{E}(\mathbf{I}_n \otimes \tilde{\mathbf{Q}}^{(l)})$ , where  $\tilde{\mathbf{Q}}^{(l)}$  is the same as  $\mathbf{Q}^{(l)}$  except that it is missing the  $\mathbf{H}_{ij}^{(l)}$  terms on the R.H.S. of (24), and  $\mathbf{E}$  is a selection matrix defined as

$$\mathbf{E} = \begin{bmatrix} e_1^T \otimes \mathbf{I}_d \\ \vdots \\ e_{(i-1)n+i}^T \otimes \mathbf{I}_d \\ \vdots \\ e_{n^2}^T \otimes \mathbf{I}_d \end{bmatrix},$$

with  $e_i$  being a  $n^2 \times 1$  vector where there is a 1 at the  $i$ -th component, and all other entries are zero.

From Theorem 1, the following lemma follows easily.

**Lemma 4** *Let  $\mathbf{Q}^{(l)} = \mathbb{E} \{ \mathbf{Q}^{(l)} \}$ . Suppose that Assumptions 1-3 hold. Then,  $\lim_{l \rightarrow \infty} \mathbf{Q}^{(l)} = \mathbf{Q}^{(\infty)}$  in (19), and for all  $l$  sufficiently large, we have  $r(\mathbf{Q}^{(l)}) < 1$ . Furthermore, there exists a finite constant  $c$  such that for all  $l$ ,  $\|\mathbf{R}^{(l)}\| \leq c$ .*

**Theorem 2** *Suppose that Assumptions 1-3 hold. Then, for all  $i \geq 0$ , we have*

$$\lim_{l \rightarrow \infty} \mathbb{E} \left\{ \tilde{\boldsymbol{\mu}}_i^{(l)} \right\} = 0_d.$$

*Proof:* Let  $\tilde{u}^{(l)} = \mathbb{E} \{ \tilde{\boldsymbol{\mu}}^{(l)} \}$ , then taking expectation on both sides of (23), we have  $\tilde{u}^{(l)} = \mathbf{Q}^{(l)} \tilde{u}^{(l-1)}$ , where  $\mathbf{Q}^{(l)} = \mathbb{E} \{ \mathbf{Q}^{(l)} \}$ , and we have used the independence of  $\mathbf{Q}^{(l)}$  and  $\tilde{\boldsymbol{\mu}}^{(l-1)}$ . The theorem now follows from Lemma 4.  $\blacksquare$

### C. MSD Convergence

Since (23) is a linear recursive equation, its mean-square stability analysis follows from similar methods used in Section 6.5 of [19]. We note however that our analysis is complicated by the fact that  $\mathbf{Q}^{(l)}$  in (23) does not have the same form as that in [19], which allows decomposition of the recursion into components that then makes it possible to replace the random matrix  $\mathbf{Q}^{(l)}$  with  $\mathbf{Q}^{(l)} = \mathbb{E} \{ \mathbf{Q}^{(l)} \}$ . In addition, we do not have the luxury of a step size parameter that can be chosen sufficiently small so that higher order terms can be ignored. Therefore, we require stronger assumptions on the distributions of  $\mathbf{H}_{ij}^{(l)}$  in order to ensure MSD convergence for gSPAWN.

The MSD at iteration  $l$  is defined to be  $\mathbb{E} \{ \|\tilde{\boldsymbol{\mu}}^{(l)}\|_w^2 \}$ . Consider a positive definite matrix  $W$ . Let  $w = \text{vec}(W)$  be the vector formed by stacking the columns of  $W$  together. For any vector  $a$ , let  $\|a\|_w^2 = \|a\|_W^2 = a^* W a$ . Following the same derivation as in [19], it can be shown that

$$\mathbb{E} \left\{ \|\tilde{\boldsymbol{\mu}}^{(l)}\|_w^2 \right\} = \mathbb{E} \left\{ \|\tilde{\boldsymbol{\mu}}^{(l-1)}\|_{\mathcal{F}^{(l)} w}^2 \right\} + \mathcal{R}^{(l)} w, \quad (25)$$

where

$$\mathcal{F}^{(l)} = \mathbb{E} \left\{ (\mathbf{Q}^{(l)})^T \otimes (\mathbf{Q}^{(l)})^* \right\}, \quad (26)$$

and

$$\mathcal{R}^{(l)} = \left( \text{vec} \left( \mathbf{R}^{(l)} \mathbf{C} (\mathbf{R}^{(l)})^T \right) \right)^T,$$

with  $\mathbf{C} = \text{diag}(C_{11}, \dots, C_{n1}, \dots, C_{1n}, \dots, C_{nn})$ . The MSD at any iteration  $l$  can be found by setting  $W = \mathbf{I}_{dn}$ . We say that gSPAWN is mean-square stable if the MSD converges as  $l \rightarrow \infty$ .<sup>6</sup>

By iterating the recursive relationship (25), we have

$$\mathbb{E} \left\{ \|\tilde{\boldsymbol{\mu}}^{(l)}\|_w^2 \right\} = \mathbb{E} \left\{ \|\tilde{\boldsymbol{\mu}}^{(0)}\|_{\prod_{i=1}^l \mathcal{F}^{(i)} \text{vec}\{\mathbf{I}_{dn}\}}^2 \right\} + \sum_{i=1}^l \mathcal{R}^{(i)} \prod_{j=i+1}^l \mathcal{F}^{(j)} \text{vec}\{\mathbf{I}_{dn}\}.$$

Therefore, to show mean square stability, it suffices to show that the matrices  $\mathcal{F}^{(l)}$  are stable for all  $l$  sufficiently large because of Lemma 4. Without any restrictions on the distributions of  $\mathbf{H}_{ij}^{(l)}$ ,  $\mathcal{F}^{(l)}$  may not

<sup>6</sup>Similar to [19], the mean-square error and excess mean-square error can be defined, but we do not discuss those here.

be stable. In the following, we present some sufficient conditions for mean-square stability, which apply in various practical applications.

**Proposition 2** *Suppose that Assumptions 1-3 hold, and  $\mathbf{H}_{ij}^{(l)} = \mathbf{H}_{ij}$  is non-random for all  $i, j$  and  $l$ . Then, the MSD in the gSPAWN algorithm converges.*

*Proof:* From (26), the eigenvalues of  $\mathcal{F}^{(l)}$  are of the form  $\phi\xi^*$ , where  $\phi$  and  $\xi$  are eigenvalues of  $\mathbf{Q}^{(l)}$ , which is assumed here to be non-random. The proposition now follows from Assumption 3. ■

**Proposition 3** *Suppose that Assumptions 1 and 2 hold. In addition, there exists a unitary matrix  $\mathbf{U}$  so that for all  $i, j = 0, \dots, n$ , there exists a unitary matrix  $\mathbf{V}_{ij}$  such that*

- (i)  $\mathbf{G}_{ij} = \mathbf{V}_{ij}\mathbf{A}_{ij}\mathbf{U}^*$ , where  $\mathbf{A}_{ij}$  is a diagonal matrix with non-negative real numbers on the diagonal;
- (ii)  $\mathbf{H}_{ij}^{(l)} = \mathbf{V}_{ij}\mathbf{B}_{ij}^{(l)}\mathbf{U}^*$  almost surely for all  $l$ , where  $\mathbf{B}_{ij}^{(l)}$  is a diagonal matrix. Furthermore,  $\mathbf{B}_{ij} = \mathbb{E}\{\mathbf{B}_{ij}^{(l)}\}$  and  $\tilde{\mathbf{B}}_{ij} = \mathbb{E}\{(\mathbf{B}_{ij}^{(l)})^2\}$  are diagonal matrices with non-negative real numbers on the diagonal, with  $\tilde{\mathbf{B}}_{ij} \leq \mathbf{A}_{ij}^2$ ; and
- (iii)  $\mathbf{C}_{ij} = \mathbf{V}_{ij}\mathbf{D}_{ij}\mathbf{V}_{ij}^*$ , where  $\mathbf{D}_{ij}$  is a diagonal matrix with positive real numbers on the diagonal.

*Then, the MSD in the gSPAWN algorithm converges.*

*Proof:* The same proof in Proposition 1 shows that for all  $i \geq 0$  and  $l \geq 0$ ,  $\mathbf{P}_i^{(l)}$  has the form  $\mathbf{U}\Lambda_i^{(l)}\mathbf{U}^*$ , where  $\Lambda_i^{(l)}$  is a non-negative diagonal matrix. For fixed  $l \geq 0$  and  $a \geq 0$ , we have

$$\prod_{k=a}^l \mathcal{F}^{(k)} \text{vec}\{\mathbf{I}_{dn}\} = \text{vec}\{\mathbf{F}_l^{(a)}\},$$

where  $\mathbf{F}_l^{(k)}$  is defined recursively as  $\mathbf{F}_l^{(k)} = \mathbb{E}\{(\mathbf{Q}^{(k)})^* \mathbf{F}_l^{(k+1)} \mathbf{Q}^{(k)}\}$ , with  $\mathbf{F}_l^{(l+1)} = \mathbf{I}_{dn}$ . Let  $\bar{\mathbf{U}} = \text{diag}(\mathbf{U}, \dots, \mathbf{U})$  and  $\mathbf{J}^{(k)}$  be matrices consisting of  $n \times n$  blocks of size  $d \times d$ , where the  $(i, j)$ -th block of  $\mathbf{J}^{(k)}$  is given by

$$\mathbf{J}^{(k)}(i, j) = \begin{cases} \Lambda_i^{(k)} \mathbf{A}_{ij} \left( \mathbf{D}_{ij} + \mathbf{B}_{ij} \Lambda_j^{(k-1)} \mathbf{B}_{ij} \right)^{-1} \mathbf{A}_{ij}, & \text{if } j \in \mathcal{B}_i, \\ 0 \cdot \mathbf{I}_d, & \text{otherwise.} \end{cases}$$

Viewing  $\mathbf{F}_l^{(k)}$  as a  $n \times n$  block matrix, we show by backward induction on  $k$  that each block  $\mathbf{U}^* \mathbf{F}_l^{(k)}(i, j) \mathbf{U}$  is a non-negative diagonal matrix and  $\bar{\mathbf{U}}^* \mathbf{F}_l^{(k)} \bar{\mathbf{U}} \leq (\mathbf{J}^{(k)})^* \bar{\mathbf{U}}^* \mathbf{F}_l^{(k+1)} \bar{\mathbf{U}} \mathbf{J}^{(k)}$ . It then follows from the Perron-Frobenius theorem [37] that since  $\mathbf{J}^{(k)}$  has spectral radius less than one,  $\mathbf{F}_l^{(a)} \rightarrow 0 \cdot \mathbf{I}_{dn \times dn}$  exponentially fast as  $l \rightarrow \infty$  so that the proposition holds.

Suppose the claim holds for  $k + 1$ . Let  $\tilde{\Pi}_{ij}^{(k)} = \left( D_{ij} + B_{ij} \Lambda_j^{(k)} B_{ij} \right)^{-1}$ , which is a non-negative diagonal matrix for all  $k$ . It can be shown algebraically that

$$\begin{aligned} \mathbf{U}^* \mathbf{F}_l^{(k)}(i, j) \mathbf{U} &= \sum_{q=1}^n \sum_{q'=1}^n \mathbb{E} \left\{ \mathbf{B}_{iq}^{(k)} \tilde{\Pi}_{iq}^{(k-1)} A_{iq} \mathbf{U}^* \mathbf{F}_l^{(k+1)}(q, q') \mathbf{U} A_{jq'} \tilde{\Pi}_{jq'}^{(k-1)} \mathbf{B}_{jq'}^{(k)} \right\} \\ &\leq \sum_{q=1}^n \sum_{q'=1}^n A_{iq}^{(k)} \tilde{\Pi}_{iq}^{(k-1)} A_{iq} \mathbf{U}^* \mathbf{F}_l^{(k+1)}(q, q') \mathbf{U} A_{jq'} \tilde{\Pi}_{jq'}^{(k-1)} A_{jq'}^{(k)} \\ &= \left( (\mathbf{J}^{(k)})^* \bar{\mathbf{U}}^* \mathbf{F}_l^{(k+1)} \bar{\mathbf{U}} \mathbf{J}^{(k)} \right) (i, j), \end{aligned}$$

where the inequality holds because all the matrices are non-negative and diagonal,  $\mathbb{E} \left\{ \mathbf{B}_{iq}^{(k)} \mathbf{B}_{jq'}^{(k)} \right\} = \mathbb{E} \left\{ \mathbf{B}_{iq}^{(k)} \right\} \mathbb{E} \left\{ \mathbf{B}_{jq'}^{(k)} \right\} \leq A_{iq} A_{jq'}$  if  $i \neq j$  or  $q \neq q'$ , and  $\mathbb{E} \left\{ (\mathbf{B}_{iq}^{(k)})^2 \right\} \leq A_{iq}^2$  by assumption. The claim trivially holds for  $k = l$ , and the induction is complete. The proposition is now proved.  $\blacksquare$

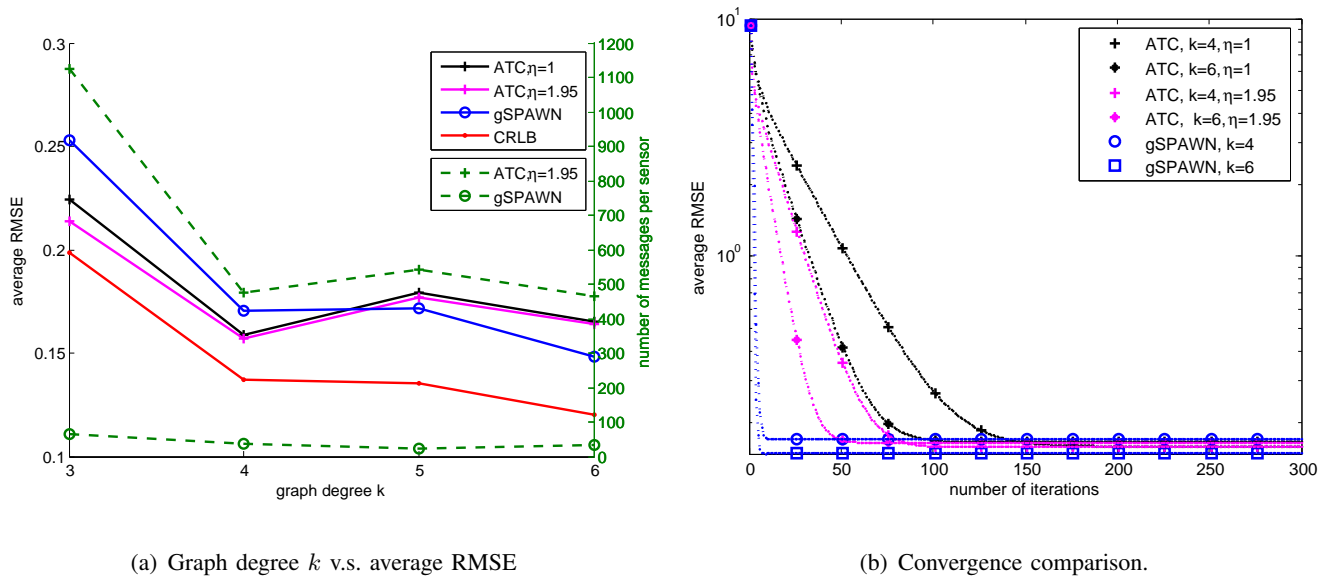
#### D. Numerical Examples and Discussions

In this section, we provide numerical results to compare the performance of gSPAWN and the ATC algorithm. For the network graph  $\mathcal{G}$ , we use a  $k$ -regular graph with 8 nodes generated using results from [38]. Each sensor  $i$  has a  $2 \times 1$  local parameter  $s_i$  to be estimated. The matrices  $G_{ij}$  and  $\mathbf{H}_{ij}^{(l)}$  are  $1 \times 2$  vectors. For each  $i, j = 0, \dots, 7$ , we let  $G_{ij}$  to be either  $[15, 0]$  or  $[0, 15]$  randomly so that Assumption 1(iii) is satisfied. The expectation  $H_{ij}$  is chosen to be either  $[10, 0]$  or  $[0, 10]$  randomly, and each  $\mathbf{H}_{ij}^{(l)}$  is generated from a Gaussian distribution with covariance  $2\mathbf{I}_2$ . The parameter  $\alpha$  in gSPAWN is chosen to be 10. The relative degree-variance in [21] is used as the combination weight in the ATC algorithm. The step-size in the ATC algorithm for sensor  $i$  (cf. equation (14)) is chosen to be  $\eta/r(R_i)$ , where  $0 < \eta < 2$ .

The root mean squared error (RMSE) for each sensor  $i$  is given by  $\sqrt{\mathbb{E} \left\{ \left\| s_i - \hat{s}_i^{(l)} \right\|_2^2 \right\}}$ , where  $\hat{s}_i^{(l)}$  is the local estimate of  $s_i$  in both the gSPAWN and ATC algorithm. The iteration  $l$  is chosen to be sufficiently large so that the RMSE is within  $10^{-5}$  of the steady state value, at which point we say that the algorithm has converged. We use the average RMSE over all sensors as the performance measure. The Cramer-Rao lower bound (CRLB) is used as a benchmark. The likelihood function  $\mathcal{L}^{(l)}(\{s_i\}_{i=1}^n)$  at iteration  $l$  follows from (1) as

$$\mathcal{L}^{(l)}(\{s_i\}_{i=1}^n) \propto \exp \left\{ -\frac{1}{2} (\mathbf{E}^{(l)} s - \mathbf{D}^{(l)})^* \mathbf{C}^{-1} (\mathbf{E}^{(l)} s - \mathbf{D}^{(l)}) \right\}, \quad (27)$$

where  $s = [s_1^T, \dots, s_n^T]^T$ ,  $\mathbf{E}^{(l)}$  is a selection matrix consisting of all system matrices  $\{G_{ij}, \mathbf{H}_{ij}^{(l)}\}$  at iteration  $l$ , and  $\mathbf{D}^{(l)}$  and  $\mathbf{C}$  are the matrices obtained by stacking all measurements  $\mathbf{d}_{ij}^{(l)}$  and noise covariances  $C_{ij}$

(a) Graph degree  $k$  v.s. average RMSE

(b) Convergence comparison.

Fig. 2. Performance comparison between gSPAWN and ATC for regular graphs.

together in the order dictated by  $\mathbf{E}^{(l)}$ . It is well known that Fisher information matrix for (27) is given by  $\mathbf{E}^{(l)*} \mathbf{C}^{-1} \mathbf{E}^{(l)}$  [39]. The average CRLB is obtained by averaging over  $l$ .

In Figure 2(a), we see that gSPAWN and ATC have comparable average RMSE when the graph degree is greater than 3. A larger graph degree results in a better average RMSE for gSPAWN, whereas the ATC algorithm performance remains relatively flat once the graph degree is greater than 3. We also observe that the total number of messages transmitted per sensor for gSPAWN is much smaller than that for ATC. This is because gSPAWN not only converges faster, but also passes fewer messages in each iteration. To be specific, the number of messages per sensor per iteration for gSPAWN is 2, and that for ATC is the graph degree  $k$ .

In Figure 2(b), we observe that the gSPAWN algorithm converges within 30 iterations, while the ATC algorithm convergence rate depends not only on the graph degree but also on the value of  $\eta$ , with a lower graph degree and a smaller value of  $\eta$  leading to a slower convergence rate. This can also be seen by comparing the average spectral radius<sup>7</sup>. In our simulations, the spectral radius for ATC with  $\eta = 1.95$  is between 0.8905 and 0.9879, while that of the gSPAWN algorithm is between 0.5315 and 0.6711.

<sup>7</sup>The spectral radius for gSPAWN is calculated for  $\mathbf{Q}^{(l)}$  in (24), and that for ATC is calculated for the recursion matrix in equation (246) of [21].

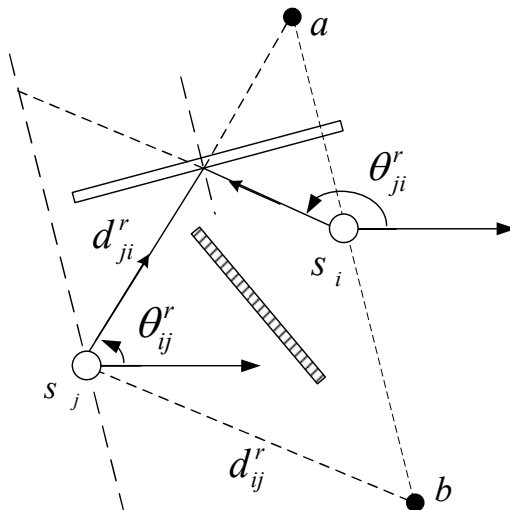


Fig. 3. An example for single-bounce scattering path between sensor  $i$  and sensor  $j$ .

## V. APPLICATION TO COOPERATIVE LOCALIZATION IN NLOS ENVIRONMENTS

In this section, we apply the gSPAWN algorithm to the problem of distributed sensor localization in a NLOS environment. Suppose that there are  $n + 1$  sensors  $0, \dots, n$  in a network, where the position of sensor  $i$  is  $s_i = [x_i, y_i]^T$  with  $x_i$  and  $y_i$  being the  $x$  and  $y$  coordinates respectively. Without loss of generality, we assume that the position of the reference sensor 0 is known up to some estimation accuracy. The objective of each sensor  $i$  is to perform self-localization relative to sensor 0 by measuring time-of-arrival (TOA) or received signal strength, and angle-of-arrival (AOA) of signals broadcast by each sensor. In most urban and indoor environments, sensors may not have direct LOS to each other. Signals transmitted by one sensor may be propagated to another sensor through multiple paths. The multipath propagation between sensors can be characterized by a ray tracing model [33]. In this application, we restrict ourselves to paths that have at most one reflection. An example of a single-bounce scattering path is shown in Figure 3, where a signal transmitted from sensor  $j$  to sensor  $i$  is reflected at a nearby scatterer. Suppose that there are  $k_{ij}$  LOS or NLOS paths between sensor  $i$  and sensor  $j$ , and that individual paths can be resolved from the received signal at each node. This is the case if, for example, the signal bandwidth is sufficiently large.

Consider the  $k$ th path between sensor  $i$  and  $j$  as shown in Figure 3. Let  $d_{ij}^k$  be the distance measured by sensor  $i$  using the TOA or received signal strength along this path from sensor  $j$ , and  $\theta_{ij}^k$  be the corresponding AOA of the signal. We assume that a particular direction has been fixed to be the horizontal

direction, and all angles are measured w.r.t. this direction. In addition, sensors do not have prior knowledge of the positions of the scatterers. It can be shown using geometrical arguments [33], [40] that

$$\mathbf{d}_{ij}^k = g(\theta_{ij}^k, \theta_{ij}^k)^T (s_i - s_j), \quad (28)$$

where

$$g(\theta, \phi) = \begin{bmatrix} \frac{\sin \theta + \sin \phi}{\sin(\theta - \phi)} \\ -\frac{\cos \theta + \cos \phi}{\sin(\theta - \phi)} \end{bmatrix}. \quad (29)$$

When the signal path between sensor  $i$  and  $j$  is a LOS path, we have  $|\theta_{ij}^k - \theta_{ij}^k| = \pi$  and  $d_{ij}^k = \|s_i - s_j\|$ , therefore we define  $g(\theta_{ij}^k, \theta_{ij}^k) = [\cos(\theta_{ij}^k), \sin(\theta_{ij}^k)]^T$  in (28).

We stack the measurements from all  $k_{ij}$  paths into vectors  $\tilde{\mathbf{d}}_{ij} = [\mathbf{d}_{ij}^1, \dots, \mathbf{d}_{ij}^{k_{ij}}]^T$  and  $\mathbf{T}_{ij} = [g(\theta_{ij}^1, \theta_{ij}^1), \dots, g(\theta_{ij}^{k_{ij}}, \theta_{ij}^{k_{ij}})]^T$ . We assume that  $k_{ij} > 1$  and  $\mathbf{T}_{ij}$  has linearly independent columns, otherwise some of the paths are redundant as they do not provide additional information to perform localization. Let  $\mathbf{d}_{ij} = \mathbf{T}_{ij}^\dagger \tilde{\mathbf{d}}_{ij}$ , where  $\mathbf{T}_{ij}^\dagger$  is the Moore-Penrose pseudo-inverse of  $\mathbf{T}_{ij}$ . We adopt the data model

$$\mathbf{d}_{ij} = s_i - s_j + \boldsymbol{\omega}_{ij}, \quad (30)$$

where we assume that the AOA of each arriving signal can be measured sufficiently accurately [41] so that the distance and AOA measurement noise can be included together in  $\boldsymbol{\omega}_{ij}$ , which is a  $2 \times 1$  measurement noise vector with covariance  $\mathbf{C}_{ij} = \sigma_{ij}^2 (\mathbf{T}_{ij}^\dagger) (\mathbf{T}_{ij}^\dagger)^T = \sigma_{ij}^2 (\mathbf{T}_{ij}^T \mathbf{T}_{ij})^{-1}$ . Note that  $\mathbf{C}_{ij}$  is a function of the AOA of the signal from sensor  $j$  to sensor  $i$ .

This corresponds to the general model in (1), where sensor observations are the same over all iterations  $l$ , and  $\mathbf{G}_{ij} = \mathbf{H}_{ij}^{(l)} = \mathbf{I}_2$  for all  $l$ . The gSPAWN algorithm in Section III can now be used to estimate  $s_i$  in a distributed manner. In this non-adaptive application, we assume that only one measurement is observed at each sensor, but the same algorithm can also be applied if new observations are available at each iteration of the gSPAWN algorithm as shown in Section III. The local updates (9) and (10) at sensor  $i$  in the  $l$ th iteration are given by

$$\mathbf{P}_i^{(l)} = \left( \sum_{j \in \mathcal{B}_i} (\mathbf{C}_{ij} + \mathbf{P}_j^{(l-1)})^{-1} \right)^{-1}, \quad (31)$$

$$\boldsymbol{\mu}_i^{(l)} = \mathbf{P}_i^{(l)} \sum_{j \in \mathcal{B}_i} (\mathbf{C}_{ij} + \mathbf{P}_j^{(l-1)})^{-1} (\boldsymbol{\mu}_j^{(l-1)} + \mathbf{d}_{ij}), \quad (32)$$

where  $\boldsymbol{\mu}_i^{(l)}$  is the sensor's local estimate of its position after  $l$  iterations.



### A. Convergence of Location Estimates

Suppose that all scatterers in the environment are either parallel or orthogonal to each other (e.g., in indoor environments, the strongest scatterers are the walls, floor and ceiling). The convergence of the gSPAWN algorithm in distributed localization of the sensors then follows from results in Section IV.

**Corollary 1** *In the gSPAWN algorithm for the NLOS localization problem in a strongly connected sensor network using data model (30), we have*

- (i) *the covariance matrix in (31) converges;*
- (ii) *the belief means in (32) converges in mean to the true sensor locations when all scatterers are either parallel or orthogonal to each other, and*
- (iii) *the location MSDs converge when all scatterers are either parallel or orthogonal to each other.*

*Proof:* From (30), it can be checked that Assumptions 1 and 2 are satisfied with  $G_{ij} = H_{ij} = \mathbf{H}_{ij}^{(l)} = I_2$ . Claim (i) now follows from Theorem 1. We next show that Assumption 3 holds when all scatterers are either parallel or orthogonal to each other. In this case, there exists a unitary matrix  $U$  such that after applying  $U$  to the system model, all the scatterers are either horizontal or vertical. Then for all  $i = 1, \dots, n$ ,  $j \in \mathcal{B}_i$ , and each path  $k$ , we have  $(\theta_{ij}^k + \theta_{ij}^k - 2\rho) \pmod{2\pi} = \pi$ , where  $\rho = 0$  or  $\pi/2$  depending on whether the path is reflected by a horizontal or vertical scatterer, respectively. From (29), we obtain

$$U^* C_{ij} U = \sigma_{ij}^2 \begin{bmatrix} \sum_{k \in \mathcal{K}_h} [\sec(\theta_{ij}^k)]^2 & 0 \\ 0 & \sum_{k \in \mathcal{K}_v} [\csc(\theta_{ij}^k)]^2 \end{bmatrix}^{-1},$$

where  $\mathcal{K}_h$  and  $\mathcal{K}_v$  are the index sets of paths from horizontal and vertical scatterers, respectively. Assumption 3 now follows from Proposition 1, and claims (ii) and (iii) then follow from Theorem 2 and Proposition 2 respectively. The corollary is now proved. ■

### B. Simulation Results and Discussions

In this section, we present simulation results for the distributed localization problem. We consider a network with 25 nodes scattered in an enclosed area of  $100\text{m} \times 100\text{m}$ . The communication links between nodes are shown in Figure 4. Scatterers are placed throughout the area. Simulations are conducted with different values of the measurement noise parameter  $\sigma_{ij} = \sigma$  for all  $i, j$ . Each simulation run consists of

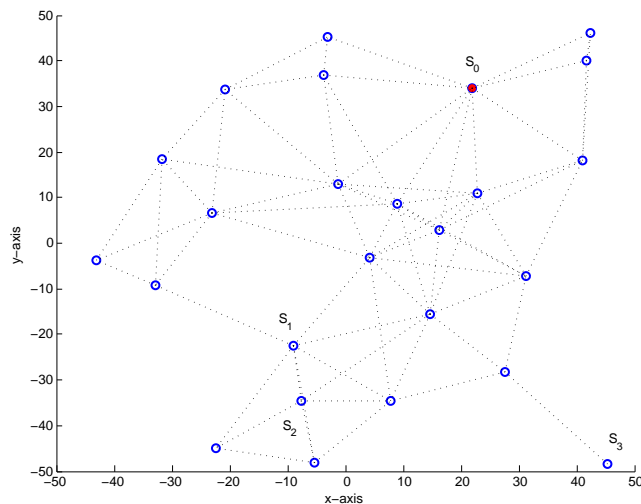
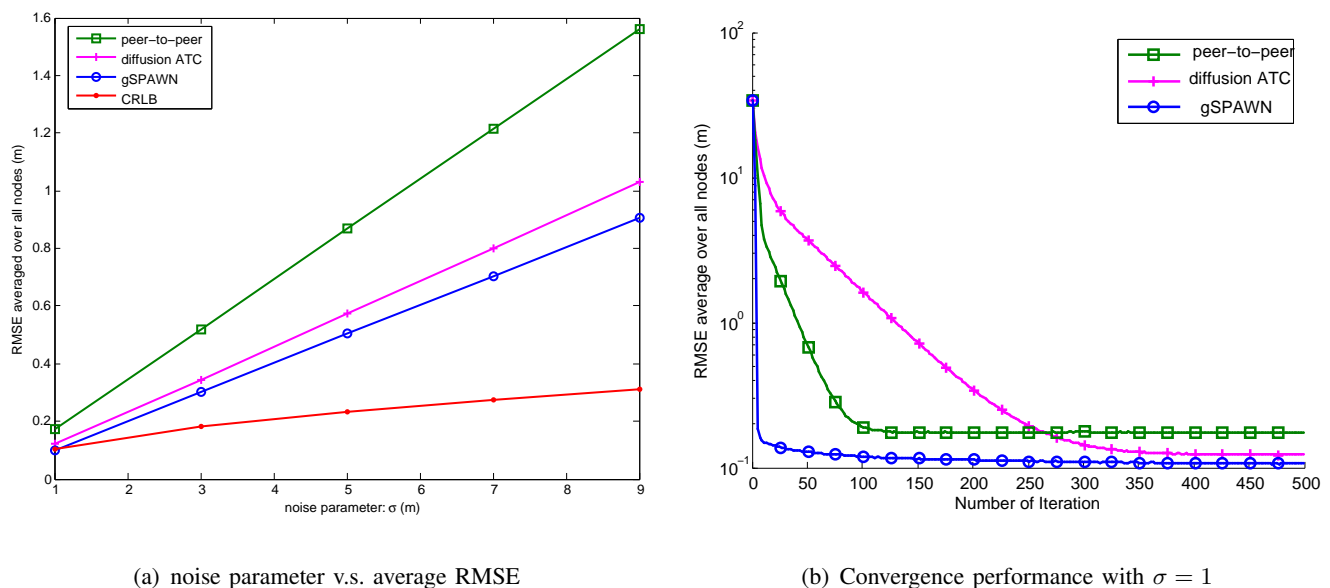


Fig. 4. A signal graph corresponding to a network of 25 nodes. The reference node is  $s_0$ .



(a) noise parameter v.s. average RMSE

(b) Convergence performance with  $\sigma = 1$

Fig. 5. Performance comparison between gSPAWN, diffusion ATC, and peer-to-peer algorithm.

1000 independent trials. We use the average RMSE over all sensors as the performance measure, and we also use CRLB as a benchmark, which are calculated in a way similar to that in Section IV-D.

We compare the performance of gSPAWN with that of a peer-to-peer localization method and the ATC method. In the peer-to-peer localization approach, sensors that have direct measurements w.r.t. the reference node are localized first, and then other sensors are localized by treating previously localized neighbors as virtual anchor nodes. In the ATC strategy, the step-size parameter for  $s_i$  is set to be  $1/r(R_i)$ , where  $r(R_i)$  is as defined in (14). The relative degree-variance is used as combination weights.

In Figure 5(a), we show the average RMSE with respect to different values of  $\sigma$ . It can be seen that

the gSPAWN algorithm outperforms both ATC and the peer-to-peer localization method, with average RMSE close to that of the ATC strategy. The convergence performance is shown in Figure 5(b), where  $\sigma = 1$ . The gSPAWN algorithm converges within 100 iterations, while the ATC converges only after 350 iterations. In this numerical example, the peer-to-peer method uses one broadcast per sensor per iteration, the gSPAWN algorithm uses two broadcasts, i.e., one for the estimated mean and the other for the covariance information, per sensor per iteration, while the ATC strategy uses an average of 5.68 broadcasts per sensor per iteration.

## VI. CONCLUSIONS

We have studied a distributed local linear parameter estimation algorithm called gSPAWN, based on the sum-product algorithm. The gSPAWN algorithm lets each node in the network broadcast a fixed size message to all neighbors in each iteration, with the message size remaining constant regardless of the network size and density. It is therefore well suited for implementation in WSNs. We show that the gSPAWN algorithm converges in mean, and has mean-square stability under some technical sufficient conditions. We also show how to apply the gSPAWN algorithm to a network localization problem in NLOS environments. Our simulation results suggest that gSPAWN has better average RMSE performance than the diffusion ATC and peer-to-peer localization methods. As part of future research work, it would be of interest to investigate the convergence properties of the more general SPAWN algorithm, which can be used for non-linear local parameter estimation, and which also has the advantage of broadcasting a fixed size message per iteration per sensor.

## APPENDIX A

### NUMERICAL EXAMPLE OF $r(Q^{(\infty)}) > 1$ .

In this appendix, we provide a numerical example that shows that unlike the case where measurements are scalar with  $m = d = 1$ , the matrix  $Q^{(\infty)}$  given by (19) may have spectral radius greater than one if additional conditions are not imposed. We consider a linear graph with  $n = 2$  nodes and a reference node 0. There is an edge between node 0 and node 1, and another edge between node 1 and node 2. Let

$G_{ij} = H_{ij} = I_2$  for all  $i, j$  in (19),  $\alpha_0 = 0$ ,  $C_{01} = 100 \cdot I_2$ , and for  $i, j = 1, 2$ ,  $C_{ij} = A_{ij}^{-1} - P_j^{(\infty)}$ , where

$$A_{11} = \begin{bmatrix} 0.3586 & -0.1580 \\ -0.1580 & 0.1135 \end{bmatrix}, A_{12} = \begin{bmatrix} 0.0709 & -0.0890 \\ -0.0890 & 0.1118 \end{bmatrix},$$

$$A_{21} = \begin{bmatrix} 0.3670 & -0.2359 \\ -0.2359 & 0.1522 \end{bmatrix}, A_{22} = \begin{bmatrix} 0.3067 & -0.0163 \\ -0.0163 & 0.0357 \end{bmatrix},$$

and

$$\left(P_1^{(\infty)}\right)^{-1} = \begin{bmatrix} 0.4395 & -0.2470 \\ -0.2470 & 0.2353 \end{bmatrix}, \left(P_2^{(\infty)}\right)^{-1} = \begin{bmatrix} 0.6737 & -0.2522 \\ -0.2522 & 0.1879 \end{bmatrix}.$$

It can be checked that  $P_1^{(\infty)}$  and  $P_2^{(\infty)}$  satisfy (18), and

$$Q^{(\infty)} = \begin{bmatrix} 1.0695 & -0.2156 & -0.1250 & 0.1574 \\ 0.4512 & 0.2560 & -0.5094 & 0.6403 \\ 0.1503 & -0.0943 & 0.8497 & 0.0943 \\ -1.0537 & 0.6834 & 1.0537 & 0.3166 \end{bmatrix},$$

which has a spectral radius of 1.017.

## REFERENCES

- [1] I. F. Akyildiz, T. Melodia, and K. R. Chowdury, "Wireless multimedia sensor networks: A survey," *IEEE Wireless Commun. Mag.*, vol. 14, no. 6, pp. 32–39, 2007.
- [2] N. Bulusu and S. Jha, *Wireless Sensor Networks: A Systems Perspective*. Artech House, 2005.
- [3] W. P. Tay, J. N. Tsitsiklis, and M. Z. Win, "Bayesian detection in bounded height tree networks," *IEEE Trans. Signal Process.*, vol. 57, no. 10, pp. 4042–4051, Oct. 2009.
- [4] —, "On the impact of node failures and unreliable communications in dense sensor networks," *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2535–2546, Jun. 2008.
- [5] Y.-W. Hong, W.-J. Huang, and F.-H. Chiu, "Cooperative communications in resource-constrained wireless networks," *IEEE Signal Process. Mag.*, vol. 24, no. 3, pp. 47–57, 2007.
- [6] H. Zhu, A. Cano, and G. B. Giannakis, "Distributed consensus-based demodulation: algorithms and error analysis," *IEEE Trans. Wireless Commun.*, vol. 9, no. 6, pp. 2044–2054, 2010.
- [7] A. Speranzon, C. Fischione, and K. H. Johansson, "Distributed and collaborative estimation over wireless sensor networks," in *Decision and Control, 2006 45th IEEE Conference on*, 2006, pp. 1025–1030.
- [8] M. R. Gholami, S. Gezici, and E. G. Strom, "Improved position estimation using hybrid tw-toa and tdoa in cooperative networks," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3770–3785, 2012.
- [9] D. Dardari, A. Conti, J. Lien, and M. Z. Win, "The effect of cooperation on uwb-based positioning systems using experimental data," *EURASIP J. Adv. Signal Process.*, vol. 2008, pp. 124:1–124:11, Jan. 2008.

- [10] H. Wymeersch, U. Ferner, and M. Win, "Cooperative bayesian self-tracking for wireless networks," *Communications Letters, IEEE*, vol. 12, no. 7, pp. 505–507, 2008.
- [11] H. Wymeersch, J. Lien, and M. Z. Win, "Cooperative localization in wireless networks," *Proc. IEEE*, vol. 97, no. 2, pp. 427–450, 2009.
- [12] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2508–2530, 2006.
- [13] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2748–2761, 2009.
- [14] S. Zhu and Z. Ding, "Distributed cooperative localization of wireless sensor networks with convex hull constraint," *IEEE Trans. Wireless Commun.*, vol. 10, no. 7, pp. 2150–2161, 2011.
- [15] M. Kriegleder, R. Oung, and R. D'Andrea, "Asynchronous implementation of a distributed average consensus algorithm," in *IEEE/RSJ International Conference of Intelligent Robots and Systems*, Tokyo, Japan, Nov. 3-7 2013.
- [16] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Trans. Signal Processing*, vol. 55, no. 8, pp. 4064–4077, August 2007.
- [17] F. Cattivelli and A. H. Sayed, "Analysis of spatial and incremental LMS processing for distributed estimation," *IEEE Trans. on Signal Process.*, vol. 59, no. 4, pp. 1465–1480, April 2011.
- [18] I. Schizas, G. Mateos, and G. Giannakis, "Distributed lms for consensus-based in-network adaptive processing," *Signal Processing, IEEE Transactions on*, vol. 57, no. 6, pp. 2365–2382, 2009.
- [19] A. H. Sayed, "Diffusion adaptation over networks," in *E-Reference Signal Processing*, R. Chellapa and S. Theodoridis, Eds. Elsevier, 2013. [Online]. Available: <http://arxiv.org/abs/1205.4220>
- [20] S.-Y. Tu and A. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *Signal Processing, IEEE Transactions on*, vol. 60, no. 12, pp. 6217–6234, 2012.
- [21] A. H. Sayed, S.-Y. Tu, J. Chen, X. Zhao, and Z. J. Towfic, "Diffusion strategies for adaptation and learning over networks: an examination of distributed strategies and network behavior," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 155–171, 2013.
- [22] M. Leng and Y.-C. Wu, "Distributed clock synchronization for wireless sensor networks using belief propagation," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5404–5414, 2011.
- [23] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, 2001.
- [24] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.
- [25] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, 2nd ed. San Francisco, CA: Morgan Kaufmann., 1988.
- [26] J. K. Johnson, D. M. Malioutov, and A. S. Willsky, "Walk-sum interpretation and analysis of Gaussian belief propagation," *Advances in Neural Information Processing System*, vol. 18, 2006.
- [27] Y. Weiss and W. T. Freeman, "Correctness of belief propagation in gaussian graphical models of arbitrary topology," *Neural Comput.*, vol. 13, no. 10, pp. 2173–2200, Oct. 2001. [Online]. Available: <http://dx.doi.org/10.1162/089976601750541769>
- [28] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, 2010.
- [29] N. Takahashi, I. Yamada, and A. H. Sayed, "Diffusion least-mean squares with adaptive combiners: Formulation and performance analysis," *IEEE Trans. Signal Process.*, vol. 58, no. 9, pp. 4795–4810, 2010.

- [30] S. Srirangarajan, A. Tewfik, and Z.-Q. Luo, "Distributed sensor network localization using SOCP relaxation," *IEEE Trans. Wireless Commun.*, vol. 7, no. 12, pp. 4886–4895, 2008.
- [31] A. T. Ihler, I. Fisher, J. W., R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, 2005.
- [32] H. Miao, K. Yu, and M. J. Juntti, "Positioning for NLOS propagation: Algorithm derivations and Cramer-Rao bounds," *IEEE Trans. Veh. Technol.*, vol. 56, no. 5, pp. 2568–2580, 2007.
- [33] C. K. Seow and S. Y. Tan, "Non-line-of-sight localization in multipath environments," *IEEE Trans. Mobile Comput.*, vol. 7, no. 5, pp. 647–660, 2008.
- [34] Y. Xie, Y. Wang, P. Zhu, and X. You, "Grid-search-based hybrid TOA/AOA location techniques for NLOS environments," *IEEE Commun. Lett.*, vol. 13, no. 4, pp. 254–256, 2009.
- [35] A. M. Sarhan, N. M. El-Shazly, and E. M. Shehata, "On the existence of extremal positive definite solutions of the nonlinear matrix equation  $\mathbf{X}^r + \sum_{i=1}^m \mathbf{A}_i^* \mathbf{X}^{\delta_i} \mathbf{A}_i = \mathbf{I}$ ," *Mathematical and Computer Modelling*, vol. 51, no. 9-10, 2010.
- [36] G. A. Milliken, "A theorem on the difference of the generalized inverse of two nonnegative matrices," *Commun. Statist.-Theor. Meth.*, vol. A6(1), 1977.
- [37] E. Seneta, *Nonnegative matrices and markov chains*, 2nd ed., ser. Springer Series in Statistics. Springer, 1981.
- [38] M. Meringer, "Fast generation of regular graphs and construction of cages," *Journal of Graph Theory*, vol. 30, pp. 137–146, 1999.
- [39] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Englewood Cliffs, NJ, 1993.
- [40] M. Leng, W.-P. Tay, and T. Quek, "Cooperative and distributed localization for wireless sensor networks in multipath environments," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, 2012, pp. 3125–3128.
- [41] J. B. Andersen and K. I. Pedersen, "Angle-of-arrival statistics for low resolution antennas," *IEEE Trans. Antennas Propag.*, vol. 50, no. 3, pp. 391–395, 2002.