DIFFUSION STRATEGIES FOR DISTRIBUTED KALMAN FILTERING: FORMULATION AND PERFORMANCE ANALYSIS

Federico S. Cattivelli *Cassio G. Lopes* *Ali H. Sayed*

Department of Electrical Engineering
University of California, Los Angeles, CA 90095
Emails: {fcattiv, cassio, sayed}@ee.ucla.edu

ABSTRACT

We consider the problem of distributed Kalman filtering, where a set of nodes are required to collectively estimate the state of a linear dynamic system from their individual measurements. Our focus is on diffusion strategies, where nodes communicate with their direct neighbors only, and the information is diffused across the network. We derive and analyze the mean and mean-square performance of the proposed algorithms and show by simulation that they outperform previous solutions.

1. INTRODUCTION

In this work we consider the problem of distributed Kalman filtering over a network of nodes (for example, a sensor network). It is assumed that some system of interest is evolving according to a linear state-space model, and that every node in the network takes measurements that are linearly related to the unobserved state. The objective is for every node to estimate the state of the system.

The performance of the state estimation will depend heavily on the collaboration strategy employed. In the centralized solution, all nodes relay their measurements to a fusion center, which uses a conventional Kalman filter to obtain the optimal state estimate, and then sends the global estimate back to every node. This strategy requires a large amount of energy for communications [1] and has a potential failure point (the central node). Distributed strategies are an attractive alternative, since they are in general more robust, require fewer communications, and allow for parallel processing.

Decentralized Kalman filtering has been proposed previously in [2] for a decentralized control problem, where it is assumed that the network is fully connected. The same assumption is used in [3] and [4], though the latter considers the case of severely quantized communications. In [5], the Kalman filtering iterations are parallelized over a set of sensors. However, the algorithm is aimed at multiprocessor systems, and still requires a fusion center to combine the estimates. Recent work in [6] employs reduced state-space models and suggests an aggregating procedure based on average consensus. The work of [7] is also based on average consensus.

Our focus is on diffusion Kalman filtering, where nodes communicate only with their neighbors, and no fusion center is present. Inspired by the connection between Kalman and RLS filtering [8, 9, 10], we extend our previous work on diffusion RLS on adaptive networks [11, 12] to the KF domain. Our algorithm computes, for every measurement and for every node, a local state estimate using the data from the neighborhood. Subsequently, every node computes a local average of the estimates of the neighborhood. This second step makes the resulting algorithm considerably different from consensus-based algorithms, where, in the general case, several averaging iterations are required to obtain the estimate [13]. We motivate and derive the algorithm, and analyze its mean and mean-square performance. We also compare the theoretical expressions with simulation results, and show performance improvement over prior solutions.

This material was based on work supported in part by the National Science Foundation under awards ECS-0725441 and ECS-0601266.

2. DISTRIBUTED KALMAN FILTER

2.1. The Kalman filter

Consider a state-space model of the form:

$$\begin{array}{rcl} x_{i+1} & = & F_i x_i + G_i n_i \\ y_i & = & H_i x_i + v_i \end{array} \qquad (1)$$

where $x_i \in \mathbb{C}^M$ and $y_i \in \mathbb{C}^{PN}$ denote the state and measurement vectors of the system, respectively, at time i. The signals n_i and v_i denote state and measurement noises, respectively, and are assumed to be zero-mean and white, with covariance matrices denoted by

$$\mathrm{E} \left[\begin{array}{c} n_i \\ v_i \end{array} \right] \left[\begin{array}{c} n_j \\ v_j \end{array} \right]^* = \left[\begin{array}{cc} Q_i & 0 \\ 0 & R_i \end{array} \right] \delta_{ij}$$

where * denotes conjugate transposition. The initial state x_0 is assumed to have zero mean, covariance matrix Π_0, and to be uncorrelated with n_i and v_i, for all i.

Let $\hat{x}_{i|j}$ denote the linear minimum mean-square error estimate of x_i given observations up to and including time j. The Kalman filter in its time- and measurement-update forms can be computed by starting from $\hat{x}_{0|-1} = 0$ and $P_{0|-1} = \Pi_0$ and iterating the following equations [9, 10]:

$$\begin{array}{l} \underline{\text{Measurement-Update:}} \\ \quad \begin{array}{rcl} R_{e,i} & = & R_i + H_i P_{i|i-1} H_i^* \\ \hat{x}_{i|i} & = & \hat{x}_{i|i-1} + P_{i|i-1} H_i^* R_{e,i}^{-1} [y_i - H_i \hat{x}_{i|i-1}] \\ P_{i|i} & = & P_{i|i-1} - P_{i|i-1} H_i^* R_{e,i}^{-1} H_i P_{i|i-1} \end{array} \\ \underline{\text{Time-Update:}} \\ \quad \begin{array}{rcl} \hat{x}_{i+1|i} & = & F_i \hat{x}_{i|i} \\ P_{i+1|i} & = & F_i P_{i|i} F_i^* + G_i Q_i G_i^* \end{array} \end{array} \qquad (2)$$

where $P_{i|j}$ denotes the covariance matrix of the estimation error $\tilde{x}_{i|j} \triangleq x_i - \hat{x}_{i|j}$.

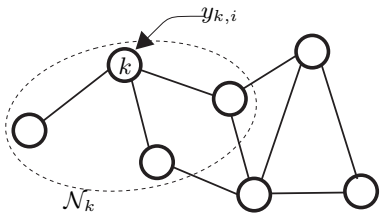

Fig. 1. At every time i, node k collects a measurement $y_{k,i}$.

2.2. Distributed Kalman filter

Consider the case where a set of N nodes are spatially distributed over some region. Let $\mathcal{N}_k$ denote the closed neighborhood of node k (i.e., the set of nodes connected to node k including itself). It is assumed that at time i, every node k collects a measurement $y_{k,i} \in \mathbb{C}^P$ according to model (1) as follows:

$$y_{k,i} = H_{k,i}x_i + v_{k,i} \qquad k = 1, ..., N \tag{3}$$

The process is shown in Figure 1. It is assumed that model (1) corresponds to collecting all N measurements from (3) as follows:

$$y_i = \begin{bmatrix} y_{1,i} \\ \vdots \\ y_{N,i} \end{bmatrix}, \quad H_i = \begin{bmatrix} H_{1,i} \\ \vdots \\ H_{N,i} \end{bmatrix}, \quad v_i = \begin{bmatrix} v_{1,i} \\ \vdots \\ v_{N,i} \end{bmatrix} \tag{4}$$

We further assume that the measurement noises $v_{k,i}$ are spatially uncorrelated, i.e.,

$$\mathrm{E} \begin{bmatrix} n_i \\ v_{k,i} \end{bmatrix} \begin{bmatrix} n_j \\ v_{l,j} \end{bmatrix}^* = \begin{bmatrix} Q_i & 0 \\ 0 & R_{k,i} \end{bmatrix} \delta_{ij}\delta_{kl}$$

The objective in a distributed Kalman filter implementation is for every node k in the network to compute an estimate of the unknown state x_i, while sharing data only with its neighbors $\{l \in \mathcal{N}_k\}$. We will denote the predicted and filtered estimates of x_i obtained by node k as $\hat{x}_{k,i|i-1}$ and $\hat{x}_{k,i|i}$, respectively. It is also desirable that the quality of the estimates be comparable to the global estimate of x_i had node k had access to all measurements across the entire network and not just its neighborhood.

3. DIFFUSION KALMAN FILTER

3.1. Local Kalman filtering

In order to motivate the diffusion Kalman filter we start by assuming that every node is able to share data with its neighbors, and uses the data to obtain the optimal state estimate given the data from the neighborhood *only*. We call this estimate the "local" Kalman filter estimate at the neighborhood of node k. It can be computed from (2) by running several measurement-updates, one for every neighbor [9, p. 329]. The iterations are shown in (5):

$$\begin{aligned}
&\psi_{k,i} \leftarrow \hat{x}_{k,i|i-1} \\
&P_{k,i} \leftarrow P_{k,i|i-1} \\
&\text{for } l \in \mathcal{N}_k \text{ repeat:} \\
&\quad R_e \leftarrow R_{l,i} + H_{l,i}P_{k,i}H_{l,i}^* \\
&\quad \psi_{k,i} \leftarrow \psi_{k,i} + P_{k,i}H_{l,i}^*R_e^{-1}[y_{l,i} - H_{l,i}\psi_{k,i}] \\
&\quad P_{k,i} \leftarrow P_{k,i} - P_{k,i}H_{l,i}^*R_e^{-1}H_{l,i}P_{k,i} \\
&\text{end} \\
&\hat{x}_{k,i|i} \leftarrow \psi_{k,i} \\
&P_{k,i|i} \leftarrow P_{k,i} \\
&\hat{x}_{k,i+1|i} = F_i\hat{x}_{k,i|i} \\
&P_{k,i+1|i} = F_iP_{k,i|i}F_i^* + G_iQ_iG_i^*
\end{aligned} \tag{5}$$

where the arrow "$\leftarrow$" denotes a sequential, or non-concurrent assignment. We also refer to (5) as the *incremental* step [14], since an optimal local estimate is generated by incrementally incorporating estimates and data sequentially from the neighborhood. The iterations (5) compute the optimal estimate for every neighborhood only, but do not take into account the fact that the neighborhoods are interconnected. In the following section, we add a diffusion step to the process that enhances the performance considerably, and allows information to be propagated throughout the network.

3.2. The diffusion Kalman filter algorithm (diffKF)

In a diffusion implementation, nodes communicate with their neighbors in an isotropic manner and cooperate to obtain better estimates than they would without cooperation. Diffusion algorithms for adaptive filters such as LMS and RLS have been proposed in [15, 16, 11]. The connection between Kalman filtering and RLS was established in [8, 10]. Here we follow similar guidelines to derive a diffusion Kalman filtering algorithm (diffKF) which is closely related to the diffusion RLS algorithm of [11, 12].

The diffusion KF algorithm and its variants require the definition of a diffusion matrix $C \in \mathbb{R}^{N\times N}$ with the following properties:

$$\mathbb{1}^*C = \mathbb{1}^* \qquad c_{l,k} = 0 \text{ if } l \notin \mathcal{N}_k \qquad c_{l,k} \geq 0, \forall l, k \tag{6}$$

where $\mathbb{1}$ is a $N \times 1$ column vector with unity entries, and $c_{l,k}$ is the (l, k) element of matrix C. We call C the *diffusion* matrix, since it governs the diffusion process, and plays an important role in the steady-state performance of the network.. The entries in C represent the weights that are used by the diffusion algorithm to combine nearby estimates. We also define a *link* matrix L as follows:

$$[L]_{l,k} = \begin{cases} 1 & \text{if } l \in \mathcal{N}_k \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Drawing a parallelism with the diffusion RLS algorithm of [11, 12], and keeping in mind the local Kalman filter (5), the diffusion Kalman filter can be derived, by adding a diffusion step after the Kalman filter update. This diffusion step could be a convex combination of the estimates of the neighborhood, i.e.,

$$\hat{x}_{k,i|i} = \sum_{l\in\mathcal{N}_k} c_{l,k}\psi_{l,i} \quad \text{with} \quad \sum_{l=1}^{N} c_{l,k} = 1 \tag{8}$$

The diffusion step is an attempt to achieve the global KF performance via local node interactions. It can be shown that combinations of the form (8) are least-squares optimal [11]. The diffusion KF algorithm is presented below.

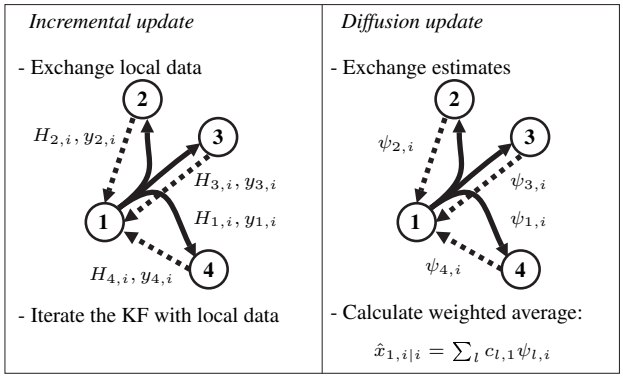

Fig. 2. Diffusion Kalman filter update (transmission of $R_{k,i}$ has been omitted to simplify the figure).

Algorithm 1: Diffusion Kalman filter (time- and measurement-update form)

Consider a state-space model as in (1) and a diffusion matrix as in (6). Start with $\hat{x}_{0|-1} = 0$ and $P_{0|-1} = \Pi_0$ and at every time instant i, compute:

Step 1: Incremental Update:

$\psi_{k,i} \leftarrow \hat{x}_{k,i|i-1}$
$P_{k,i} \leftarrow P_{k,i|i-1}$
for every neighboring node $l \in \mathcal{N}_k$, repeat:
$R_e \leftarrow R_{l,i} + H_{l,i}P_{k,i}H_{l,i}^*$
$\psi_{k,i} \leftarrow \psi_{k,i} + P_{k,i}H_{l,i}^*R_e^{-1}\left[y_{l,i} - H_{l,i}\psi_{k,i}\right]$
$P_{k,i} \leftarrow P_{k,i} - P_{k,i}H_{l,i}^*R_e^{-1}H_{l,i}P_{k,i}$
end

Step 2: Diffusion Update:

$\hat{x}_{k,i|i} \leftarrow \sum_{l\in\mathcal{N}_k} c_{l,k}\psi_{l,i}$
$P_{k,i|i} \leftarrow P_{k,i}$
$\hat{x}_{k,i+1|i} = F_i\hat{x}_{k,i|i}$
$P_{k,i+1|i} = F_iP_{k,i|i}F_i^* + G_iQ_iG_i^*$

Algorithm 1 requires that at every instant i, nodes communicate with their neighbors their measurement matrices $H_{k,i}$, the covariance matrices $R_{k,i}$, and the measurements $y_{k,i}$ for the incremental update, and their pre-estimates $\psi_{k,i}$ for the diffusion update. The total communication requirement for every node and for every measurement is $PM + M + P^2/2 + 3P/2$ complex scalars, and it requires one matrix inversion per incremental update.

This process is shown schematically in Figure 2, where the transmission of $R_{k,i}$ has been omitted to simplify the figure, and due to the following argument. Note that communication of $R_{k,i}$ may not be necessary if its Cholesky factor is computed, $R_{k,i} = L_{k,i}L_{k,i}^*$, and $\bar{H}_{k,i} = L_{k,i}^{-1}H_{k,i}$ and $\bar{y}_{k,i} = L_{k,i}^{-1}y_{k,i}$ are transmitted instead of $H_{k,i}$ and $y_{k,i}$. In this case, the error covariance is updated using $R_e \leftarrow I + \bar{H}_{l,i}P_{k,i}\bar{H}_{l,i}^*$, and the remaining recursions replace $H_{k,i}$ and $y_{k,i}$ by $\bar{H}_{k,i}$ and $\bar{y}_{k,i}$. In this scenario, Algorithm 1 requires transmission of $PM + M + P$ complex scalars per node per measurement.

It is important to note that even though the notation $P_{k,i|i}$ and $P_{k,i|i-1}$ has been retained for simplicity, these two matrices do *not* represent the covariance of the estimation error any longer, since the diffusion update is not taken into account in the recursions for these matrices. Exact expressions for the covariances of the estimates will be derived in Section 4.

An alternate formulation of Algorithm 1 may be obtained by using the information form of the Kalman filter. In this case, the incremental update (5) is replaced by a sum of terms of the form $H_{l,i}^*R_{l,i}^{-1}H_{l,i}$ and $H_{l,i}^*R_{l,i}^{-1}y_{l,i}$. We call this form Algorithm 2 , and present it below.

Algorithm 2: Diffusion Kalman filter (information form)

Consider a state-space model as in (1) and a diffusion matrix as in (6). Start with $\hat{x}_{0|-1} = 0$ and $P_{0|-1} = \Pi_0$ and at every time instant i, compute:

Step 1: Incremental Update:

$S_{k,i} = \sum_{l\in\mathcal{N}_k} H_{l,i}^*R_{l,i}^{-1}H_{l,i}$
$q_{k,i} = \sum_{l\in\mathcal{N}_k} H_{l,i}^*R_{l,i}^{-1}y_{l,i}$
$P_{k,i|i}^{-1} = P_{k,i|i-1}^{-1} + S_{k,i}$
$\psi_{k,i} = \hat{x}_{k,i|i-1} + P_{k,i|i}\left[q_{k,i} - S_{k,i}\hat{x}_{k,i|i-1}\right]$

Step 2: Diffusion Update:

$\hat{x}_{k,i|i} = \sum_{l\in\mathcal{N}_k} c_{l,k}\psi_{l,i}$
$\hat{x}_{k,i+1|i} = F_i\hat{x}_{k,i|i}$
$P_{k,i+1|i} = F_iP_{k,i|i}F_i^* + G_iQ_iG_i^*$

The total communication required per measurement per node, is $M^2/2 + 3M/2 + P$ scalars, and it requires two matrix inversions per incremental update. Algorithm 1 and Algorithm 2 are mathematically equivalent and will produce the same estimation results.

The incremental update of Algorithm 2 is similar to the update proposed in [7]. An important difference in the algorithms is in the diffusion step. In [7], the author uses a consensus-based approach for averaging as follows (Algorithm 2 in [7]):

$$\hat{x}_{k,i|i} = \psi_{k,i} + \epsilon\sum_{l\in\mathcal{N}_k}(\psi_{l,i} - \psi_{k,i})$$

On the contrary, we use a convex combination of the estimates of the neighbors as in (8). This difference produces a significant improvement in the performance of our algorithm for the reasons we state next.

In average consensus methods, the network must perform repeated averaging steps before arriving at a consensus. When these methods are used for distributed estimation, the nodes update their local estimates (using, for instance, a Kalman filter [6] or a least-squares update [13]) and then they run a consensus step with several iterations to fuse the estimates. After the data have been fused, a new measurement is taken, the estimates are updated again, and so on. Thus, in consensus estimation algorithms there are two time-scales: one for collecting the measurements, and one for running the averaging consensus.

In the proposed diffusion KF, on the other hand, the operations are done in real-time using a single time-scale. This feature allows for better adaptation and tracking abilities and leads to improved performance; see also [15, 16].

4. PERFORMANCE ANALYSIS

In this section we analyze the mean and mean-square performance of the diffusion Kalman filter (Algorithm 1). We provide closed form

expressions for the mean-square deviation (MSD) for every node, which is defined for node k as:

$$\text{MSD}_{k,i} = \text{E}||x_i - \hat{x}_{k,i|i}||^2$$

The MSD is indexed by time i and node k, since for diffusion algorithms, different nodes produce different estimates in general. Also, since the model dynamics may be changing with time, the MSD is a function of time.

For our analysis, we use the information form of the diffusion KF (Algorithm 2) to derive the expressions. The analysis holds for both Algorithms 1 and 2, since they are mathematically equivalent. Let $\tilde{\psi}_{k,i} = x_i - \psi_{k,i}$ denote the estimation error at the *end* of the incremental update. Then, it holds that

$$\begin{aligned}
\tilde{\psi}_{k,i} &= \tilde{x}_{k,i|i-1} - P_{k,i|i} \sum_{l\in\mathcal{N}_k} H_{l,i}^* R_{l,i}^{-1} \left[H_{l,i}\tilde{x}_{k,i|i-1} + v_{l,i}\right] \\
&= P_{k,i|i}\left[P_{k,i|i}^{-1} - S_{k,i}\right]\tilde{x}_{k,i|i-1} - P_{k,i|i}\sum_{l\in\mathcal{N}_k} H_{l,i}^* R_{l,i}^{-1} v_{l,i} \\
&= P_{k,i|i}P_{k,i|i-1}^{-1}\tilde{x}_{k,i|i-1} - P_{k,i|i}\sum_{l\in\mathcal{N}_k} H_{l,i}^* R_{l,i}^{-1} v_{l,i} \qquad (9)
\end{aligned}$$

where $\tilde{x}_{k,i|i-1} = x_i - \hat{x}_{k,i|i-1}$ denotes the estimation error at node k. We also have

$$\tilde{x}_{k,i|i-1} = F_{i-1}\tilde{x}_{k,i-1|i-1} + G_{i-1}n_{i-1} \qquad (10)$$

Combining (9) and (10) into the diffusion step of Algorithm 2 , we obtain

$$\begin{aligned}
\tilde{x}_{k,i|i} &= \sum_{l\in\mathcal{N}_k} c_{l,k}\tilde{\psi}_{l,i} \\
&= \sum_{l\in\mathcal{N}_k} c_{l,k}\Big[P_{l,i|i}P_{l,i|i-1}^{-1}\big(F_{i-1}\tilde{x}_{l,i-1|i-1} + \\
&\qquad G_{i-1}n_{i-1}\big) - P_{l,i|i}\sum_{m\in\mathcal{N}_l} H_{m,i}^* R_{m,i}^{-1} v_{m,i}\Big] \qquad (11)
\end{aligned}$$

4.1. Mean performance

Taking expectations of both sides of (11), we obtain the following recursion for the expectation of the estimate of the diffusion KF algorithm:

$$\text{E}\tilde{x}_{k,i|i} = \sum_{l\in\mathcal{N}_k} c_{l,k}P_{l,i|i}P_{l,i|i-1}^{-1}F_{i-1}\text{E}\tilde{x}_{l,i-1|i-1} \qquad (12)$$

Since $\text{E}\tilde{x}_{k,0|-1} = 0$ and $\text{E}\tilde{x}_{k,-1|-1} = 0$, we conclude from (12) that the diffusion KF estimate is unbiased.

4.2. Mean-square performance

Consider the augmented state-error vector $\tilde{\chi}_{i|i}$ and the block-diagonal matrices $\mathcal{H}_i$, $\mathcal{P}_{i|i}$ and $\mathcal{P}_{i|i-1}$ defined as follows:

$$\begin{aligned}
\tilde{\chi}_{i|i} &\triangleq \begin{bmatrix} \tilde{x}_{1,i|i} \\ \vdots \\ \tilde{x}_{N,i|i} \end{bmatrix} \\
\mathcal{H}_i &\triangleq \text{diag}\{H_{1,i},\ldots,H_{N,i}\} \\
\mathcal{P}_{i|i} &\triangleq \text{diag}\{P_{1,i|i},\ldots,P_{N,i|i}\} \\
\mathcal{P}_{i|i-1} &\triangleq \text{diag}\{P_{1,i|i-1},\ldots,P_{N,i|i-1}\}
\end{aligned}$$

Consider also the extended matrices (from (6) and (7)):

$$\mathcal{C} \triangleq C \otimes I_M \qquad \mathcal{L} \triangleq L \otimes I_M$$

where $\otimes$ denotes Kronecker product. We may now express (11) in a global form that captures the evolution of the entire network:

$$\begin{aligned}
\tilde{\chi}_{i|i} = \mathcal{C}^T &\begin{bmatrix} P_{1,i|i}P_{1,i|i-1}^{-1}[F_{i-1}\tilde{x}_{1,i-1|i-1} + G_{i-1}n_{i-1}] \\ \vdots \\ P_{N,i|i}P_{N,i|i-1}^{-1}[F_{i-1}\tilde{x}_{N,i-1|i-1} + G_{i-1}n_{i-1}] \end{bmatrix} - \\
&\mathcal{C}^T\begin{bmatrix} P_{1,i|i} & & \\ & \ddots & \\ & & P_{N,i|i} \end{bmatrix}\mathcal{L}^T\begin{bmatrix} H_{1,i}R_{1,i}^{-1}v_{1,i} \\ \vdots \\ H_{N,i}R_{N,i}^{-1}v_{N,i} \end{bmatrix}
\end{aligned}$$

or equivalently:

$$\begin{aligned}
\tilde{\chi}_{i|i} = \; &\mathcal{C}^T\mathcal{P}_{i|i}\big[\mathcal{P}_{i|i-1}^{-1}(I\otimes F_{i-1})\tilde{\chi}_{i-1|i-1} + \\
&\mathcal{P}_{i|i-1}^{-1}(I\otimes G_{i-1})(\mathbb{1}\otimes n_{i-1}) - \mathcal{L}^T\mathcal{H}_i^* R_i^{-1}v_i\big] \qquad (13)
\end{aligned}$$

where v_i was defined in (4) and $R_i = \text{E}v_i v_i^*$ is a block-diagonal matrix. Equation (13) can be rewritten more compactly as:

$$\tilde{\chi}_{i|i} = A_i\tilde{\chi}_{i-1|i-1} + B_i(\mathbb{1}\otimes n_{i-1}) - D_i v_i$$

where

$$\begin{aligned}
A_i &\triangleq \mathcal{C}^T\mathcal{P}_{i|i}\mathcal{P}_{i|i-1}^{-1}(I\otimes F_{i-1}) \\
B_i &\triangleq \mathcal{C}^T\mathcal{P}_{i|i}\mathcal{P}_{i|i-1}^{-1}(I\otimes G_{i-1}) \\
D_i &\triangleq \mathcal{C}^T\mathcal{P}_{i|i}\mathcal{L}^T\mathcal{H}_i^* R_i^{-1}
\end{aligned}$$

Let $\mathcal{P}_{\tilde{\chi},i} = \text{E}\{\tilde{\chi}_{i|i}\tilde{\chi}_{i|i}^*\}$ denote the covariance matrix of $\tilde{\chi}_{i|i}$. From (14) and the whiteness assumptions on the state and measurement noises, we obtain

$$\mathcal{P}_{\tilde{\chi},i} = A_i\mathcal{P}_{\tilde{\chi},i-1}A_i^* + B_i(\mathbb{1}\mathbb{1}^*\otimes Q_{i-1})B_i^* + D_i R_i D_i^* \qquad (14)$$

where we have used the property of Kronecker products that $(A\otimes B)(C\otimes D) = (AC)\otimes(BD)$.

In order to analyze the mean-square steady-state performance, we introduce the following assumptions.

Assumption 1 *The matrices in model (1) are time-invariant, i.e., the matrices F, G, H, R and Q do not depend on time i. Moreover, we assume that the matrix F is stable, i.e., all of its eigenvalues lie inside the unit circle.*

Assumption 2 *A Kalman filter that uses data from a neighborhood converges for* every *neighborhood, i.e.,* $\lim_{i\to\infty} P_{k,i|i-1} \triangleq P_k^-$ *and* $\lim_{i\to\infty} P_{k,i|i} \triangleq P_k$, $k \in \{1,\ldots,N\}$ *(see [9] for conditions on Kalman filter convergence).*

Under these assumptions, the matrices A_i, B_i and D_i also converge in steady-state, and their steady-state values are given by

$$\begin{aligned}
\mathcal{P} \triangleq \lim_{i\to\infty}\mathcal{P}_{i|i} &= \text{diag}\{P_1,\ldots,P_N\} \\
\mathcal{P}^- \triangleq \lim_{i\to\infty}\mathcal{P}_{i|i-1} &= \text{diag}\{P_1^-,\ldots,P_N^-\} \\
A \triangleq \lim_{i\to\infty} A_i &= \mathcal{C}^T\mathcal{P}(\mathcal{P}^-)^{-1}(I\otimes F) \qquad (15) \\
B \triangleq \lim_{i\to\infty} B_i &= \mathcal{C}^T\mathcal{P}(\mathcal{P}^-)^{-1}(I\otimes G) \\
D \triangleq \lim_{i\to\infty} D_i &= \mathcal{C}^T\mathcal{P}\mathcal{L}^T\mathcal{H}^* R^{-1}
\end{aligned}$$

Assumptions 1 and 2 are sufficient to guarantee the convergence of the diffusion KF algorithm, i.e., we can show that the matrix A in (15) is stable, and that (14) converges to the unique solution of the Lyapunov equation:

$$\mathcal{P}_{\tilde{x}} = A\mathcal{P}_{\tilde{x}}A^* + B(\mathbb{1}\mathbb{1}^* \otimes Q)B^* + DRD^* \qquad (16)$$

Complete proofs of these statements will be provided elsewhere due to space considerations.

Now we can solve for the steady-state covariance of the estimation error of the diffusion KF algorithm, $\mathcal{P}_{\tilde{x}}$. The solution may be expressed using the *vec* operator, which vectorizes a matrix by stacking its columns, and the property that $\text{vec}(P\Sigma Q) = (Q \otimes P^T)\text{vec}(\Sigma)$. In this case, we have

$$\text{vec}\left(\mathcal{P}_{\tilde{x}}\right) = \left(I - A^* \otimes A^T\right)^{-1} \text{vec}\left(B(\mathbb{1}\mathbb{1}^* \otimes Q)B^* + DRD^*\right)$$

and we can recover $\mathcal{P}_{\tilde{x}}$ from $\text{vec}\left(\mathcal{P}_{\tilde{x}}\right)$. Note that since A is stable, the matrix $I - A^* \otimes A^T$ is non-singular.

The MSD at node k may now be expressed as:

$$\text{MSD}_k = \lim_{i\to\infty} \text{E}||x_i - \hat{x}_{k,i|i}||^2 = \text{Tr}\left(\mathcal{P}_{\tilde{x}}\mathcal{I}_k\right) \qquad (17)$$

where $\mathcal{I}_k$ is an $NM \times NM$ block matrix with blocks of size $M \times M$, with an identity matrix at block (k,k) and zeros elsewhere. Finally, the average MSD across the network is:

$$\text{MSD}^{\text{ave}} = \frac{1}{N}\text{Tr}\left(\mathcal{P}_{\tilde{x}}\right) \qquad (18)$$

We summarize our results with the following Lemma, which follows directly from the convergence of $\mathcal{P}_{\tilde{x},i}$ in (14) to the solution of (16) and the derivation of (17).

Lemma 1 *Under Assumptions 1 and 2, the diffusion KF algorithm (Algorithm 1) is unbiased and converges, and the steady-state mean-square deviation for every node is given by (17).*

5. SIMULATIONS

We now show simulation results for the diffusion KF algorithm (Algorithm 1) and compare them to the distributed solution of [7]. We use a similar example to the one presented in [7], where the network is attempting to track the position of a rotating object. The state of the system is the unknown position of the object, a 2-dimensional vector where the first and second entries are the x and y coordinates, respectively. The state-space model matrices in (1) are:

$$F = \begin{bmatrix} 0.992 & -0.1247 \\ 0.1247 & 0.992 \end{bmatrix}, \; G = \begin{bmatrix} 0.625 & 0 \\ 0 & 0.625 \end{bmatrix}, \; Q = I_2$$

The nodes take measurements of the unknown position of the object either in the x or y direction. Thus, the measurement matrix $H_{k,i}$ is chosen to be either $[0 \quad 1]$ or $[1 \quad 0]$, at random, but with the requirement that every neighborhood should have nodes with both types of matrices (to guarantee convergence of the local Kalman filter as in Assumption 2). Finally, the measurement noise matrices are $R_{k,i} = 10\sqrt{k}$, $k = 1, \ldots, N$. Note that node 1 will be the one with the least amount of measurement noise, and node N will be the noisiest. The network has $N = 20$ nodes, and the connections are shown schematically in Figure 3. In all cases, the MSD was averaged over 200 experiments. The diffusion matrix C was chosen such that every neighbor is weighted according to the number of neighbors it has, as follows:

$$c_{lk} = \begin{cases} \alpha_k |\mathcal{N}_l| & \text{if } l \in \mathcal{N}_k \\ 0 & \text{otherwise} \end{cases}$$

where $|\mathcal{N}_k|$ is the cardinality of the closed neighborhood of node k (i.e., the number of neighbors including itself), and α_k is a parameter chosen such that $\mathbb{1}^* C = \mathbb{1}^*$.

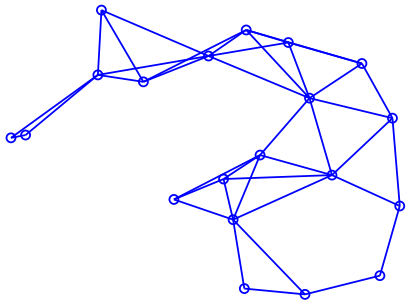

Fig. 3. Network graph.

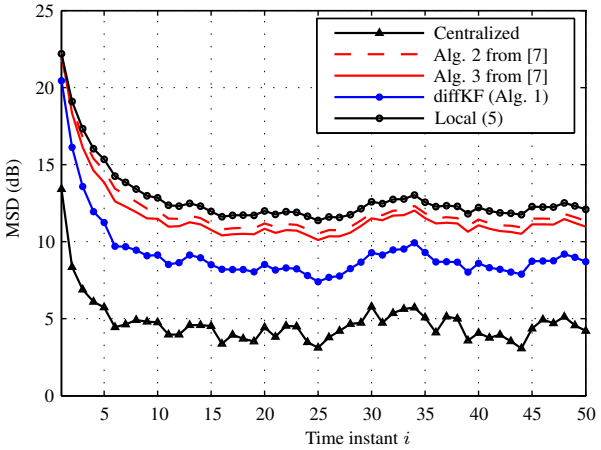

Fig. 4. Average MSD over the entire network as a function of time, averaged over 200 experiments.

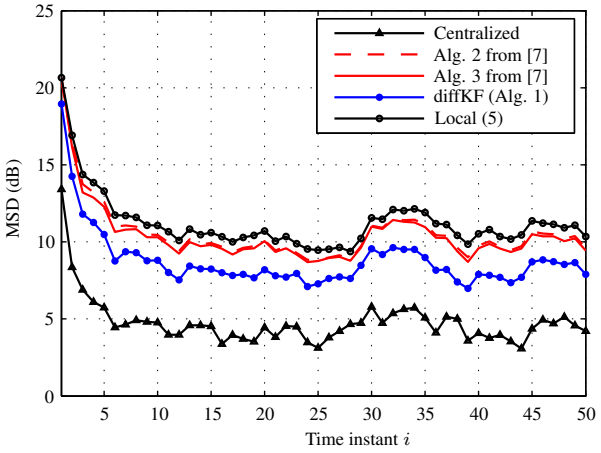

Fig. 5. MSD for node 1 as a function of time, averaged over 200 experiments.

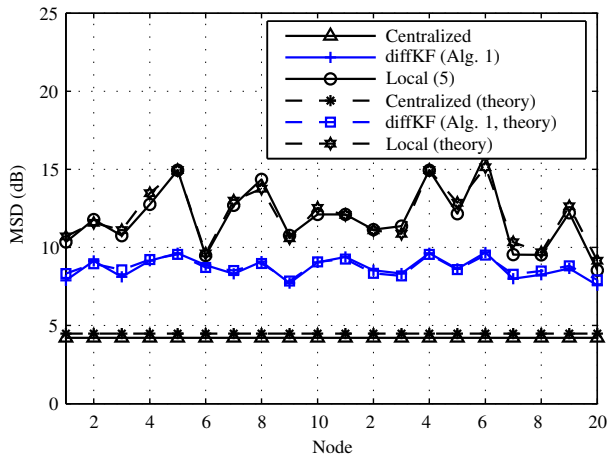

Fig. 6. Steady-state MSD for all nodes after 50 iterations, averaged over 200 experiments.

Figure 4 shows the average MSD (Equation (18)) over the entire network for different algorithms. The algorithm denoted "Local" is computed assuming there is no diffusion process, but every node has access to the data of its neighbors as in (5). Therefore, every node would run a conventional Kalman filter using the data from its neighborhood. This algorithm is included for comparison, to evaluate the performance improvement introduced by the diffusion exchange. Also shown are Algorithms 2 and 3 from [7], which are consensus-based. The algorithm denoted "diffKF" corresponds to our proposed Algorithm 1, and finally the algorithm denoted "Centralized" corresponds to the best we can do, i.e., a conventional Kalman filter that has access to all the data. It can be observed from the plots that the diffusion KF algorithm improves considerably over the "Local" and consensus-based algorithms by about 2-4 dB in this example.

Figure 5 shows the MSD for node 1 in the network. Again, the diffusion KF algorithm outperforms the remaining distributed solutions (and this is true for every other node).

The steady-state expressions from Section 4 are compared to the simulation results in Figure 6, where we show the individual steady-state MSD for every node. The theoretical expression for the diffusion KF algorithm was obtained using (17), and the theoretical expressions for the local and centralized Kalman filters can be readily obtained as the trace of the error covariance matrix of every node. The expressions derived show good agreement with the simulation results.

6. CONCLUSIONS

We presented a diffusion Kalman filtering strategy for distributed state estimation in linear systems. The algorithm requires every node to communicate with its neighbors: first to share the data, and second to share the estimates. The diffusion procedure ensures that information is propagated throughout the network. We also provided steady-state mean and mean-square analysis of the algorithm, and showed by simulation that it outperforms previous solutions.

7. REFERENCES

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," in *Proc. ICASSP*, Salt Lake City, UT, May 2001, pp. 2033–2036.

[2] J. L. Speyer, "Computation and transmission requirements for a decentralized linear-quadratic-Gaussian control problem," *IEEE Transactions on Automatic Control*, vol. AC-24, no. 2, pp. 266–269, April 1979.

[3] B.S. Rao and H.F. Durrant-Whyte, "Fully decentralised algorithm for multisensor Kalman filtering," *IEE Proceedings-D*, vol. 138, no. 5, pp. 413–420, September 1991.

[4] A. Ribeiro, G. B. Giannakis, and R. I. Roumeliotis, "SOI-KF: Distributed Kalman filtering with low-cost communications using the sign of innovations," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4782–4795, 2006.

[5] H. R. Hashemipour, S. Roy, and A. J. Laub, "Decentralized structures for parallel Kalman filtering," *IEEE Transactions on Automatic Control*, vol. 33, no. 1, pp. 88–94, January 1988.

[6] U. A. Khan and J. M. Moura, "Distributed Kalman filters in sensor networks: Bipartite fusion graphs," in *Proc. of the 14th IEEE Workshop on Statistical Signal Processing*, Madison, WI, August 2007, pp. 700–704.

[7] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. 46th IEEE Conf. Decision and Control*, New Orleans, LA, December 2007.

[8] A. H. Sayed and T. Kailath, "A state-space approach to adaptive RLS filtering," *IEEE Signal Processing Magazine*, vol. 11, no. 3, pp. 18–60, July 1994.

[9] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*, Prentice Hall, NJ, 2000.

[10] A. H. Sayed, *Fundamentals of Adaptive Filtering*, Wiley, NJ, 2003.

[11] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "A diffusion RLS scheme for distributed estimation over adaptive networks," in *Proc. IEEE SPAWC*, Helsinki, Finland, June 2007, pp. 1–5.

[12] F. S. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," to appear, *IEEE Transactions on Signal Processing*, 2008.

[13] L. Xiao, S. Boyd, and S. J. Kim, "Distributed average consensus with least-mean-square deviation," in *Proc. 17th International Symposium on Mathematical Theory of Networks and Systems (MTNS)*, Kyoto, Japan, July 2006, pp. 2768–2776.

[14] C. G. Lopes and A. H. Sayed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, August 2007.

[15] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks," in *Proc. IEEE ICASSP*, Honolulu, Hawaii, April 2007, pp. 917–920.

[16] C. G. Lopes and A. H. Sayed, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," to appear, *IEEE Transactions on Signal Processing*, 2008.