Distributed Estimation from Relative Measurements in Sensor Networks

#Prabir Barooah and João P. Hespanha

Abstract— **We consider the problem of estimating vector-valued variables from noisy "relative" measurements. The measurement model can be expressed in terms of a graph, whose nodes correspond to the variables being estimated and the edges to noisy measurements of the difference between the two variables. We take the value of one particular variable as a reference and consider the optimal estimator for the differences between the remaining variables and the reference. This type of measurement model appears in several sensor network problems, such as sensor localization and time synchronization.**

Two algorithms are proposed to compute the optimal estimate in a distributed, iterative manner. The first algorithm implements the Jacobi method to iteratively compute the optimal estimate, assuming all communication is perfect. The second algorithm is robust to temporary communication failures, and converges to the optimal estimate when certain mild conditions on the failure rate are satisfied. It also employs an initialization scheme to improve accuracy in spite of the slow convergence of the Jacobi method.

I. Introduction

We consider an estimation problem that is relevant to a large number of sensor networks applications, such as localization and time synchronization. Consider the estimation of n vector-valued variables $x_1, x_2, \ldots, x_n \in \mathbb{R}^k$ based on several noisy "relative measurements" ζ_{ij}. The measurement indices (i, j) take values in some set $\mathbf{E}$ of pairs of values from $\mathbf{V} := \{1, 2, \ldots, n\}$. The term "relative" comes from the measurement model considered:

$$\zeta_{uv} = x_u - x_v + \epsilon_{uv}, \qquad \forall (u, v) \in \mathbf{E}, \tag{1}$$

where the ϵ_{uv} are uncorrelated zero-mean noise vectors with known covariance matrices $P_{uv} = \mathrm{E}[\epsilon_{uv}\epsilon_{uv}^T]$. Just with relative measurements, determining the x_u's is only possible up to an additive constant. To avoid this ambiguity, we assume that a particular variable (say x_1) is used as the *reference* and therefore $x_1 = 0$.

The measurement equations (1) can be expressed in terms a directed graph $G = (\mathbf{V}, \mathbf{E})$ with $|\mathbf{V}| = n$ vertices (nodes) and $|\mathbf{E}| = m$ edges, with an edge (u, v) if the measurement ζ_{uv} is available. The vector x_u is called the *u-th node variable*. Our objective is to construct the optimal estimate $\hat{x}_u^*$ of x_u for every node $u \in \mathbf{V} \setminus \{1\}$. The *Optimal estimate* refers to the estimate produced by the classical Unbiased Minimum Variance Estimator (UMVE), which achieves the minimum variance among all linear unbiased estimators.

When applied to the location estimation problem, a node variable x_u could be the position of node u in 2-d or 3-d space w.r.t. the reference node, and when applied to time synchronization, it could be time shift of u's local clock w.r.t. the clock of node 1. For a through discussion on how these problems can be modeled with (1), see [1], [2] and references therein.

To compute the optimal estimate directly, one seems to need all the measurements and the topology of the graph (see beginning of section III). Thus, if one node in the network has to compute it, all this information has to be transmitted to that node. For networks with a large number of measurements, doing so will be prohibitively expensive in terms of energy consumption, bandwidth and communication time. Moreover, such a centralized computation will be less robust to dynamic changes in topology resulting from link and node failures over time.

In this paper we propose an iterative algorithm to compute the estimate of the node variables in a distributed manner. By distributed we mean that at every step, each node computes its own estimate and the data required for the computation performed at a node is obtained through communication with its one-hop neighbors. We show that the estimate produced by the algorithm asymptotically approaches the optimal estimate even in the presence of faulty communication links, as long as some mild conditions on the duration of faults are satisfied. We first propose an algorithm that implements the Jacobi iterative method to compute the optimal estimate, assuming that all communication is perfect (no failure), for which we can establish performance bounds. This algorithm was then improved to handle dynamic changes in the communication topology brought about by temporary link failures. It also employs an initialization scheme to achieve greater accuracy. Accuracy of an estimate is measured by the norm of the difference between it and the optimal one.

A similar algorithm was alluded to in [1] where the problem of time synchronization from measurements of time differences was considered; but the algorithm was not investigated. The algorithm proposed in this paper is more general, since it works for vector valued variables such as positions and not just scalar valued ones such as clock times. Moreover, our algorithm is proven to work even in the presence of faulty communication. Our work was inspired by [3] where the Jacobi and other iterative algorithms were applied to computing the optimal estimate in a different problem, one where absolute measurements of random node variables (such as temperature) were available, but the node variables were correlated.

In a previous paper [2], the authors considered how the variance of the optimal estimator (for the problem considered in this paper) grows with the distance of a node from the reference, and how that growth depends on the structure of the graph. A classification of graphs were obtained that determined the bounds on the variance achieved by the optimal estimate.

In this paper, we propose an algorithm that asymptotically obtains the optimal estimate (when the number of iterations approaches infinity), while simultaneously being simple, scalable, distributed and robust to communication failures.

The paper is organized as follows. In section III, we describe the first distributed algorithm that implements the Jacobi iterative method and establish its performance bounds. In section IV we modify this algorithm to handle faulty communication links. Section V describes a modification to this algorithm to improve its performance. Simulations done with the resulting algorithm are presented in section VI. The paper concludes with a note on future directions in section VII.

II. The Optimal Estimate

Consider a measurement graph G with n nodes and m edges. Let X be a vector in $\mathbb{R}^{(n-1)k}$ obtained by stacking together all the unknown node variables, i.e., $X := [x_2^T, \dots, x_n^T]^T$. Define $Z := [\zeta_1^T, \zeta_2^T, ..., \zeta_m^T]^T \in \mathbb{R}^{km}$ and $\boldsymbol{\epsilon} := [\epsilon_1^T, \epsilon_2^T, ..., \epsilon_m^T]^T \in \mathbb{R}^{km}$. Eq. (1) can now be rewritten as follows:

$$Z \;=\; \mathcal{A}_b^T X + \boldsymbol{\epsilon}, \quad \mathcal{A}_b := A_b \otimes I_k, \tag{2}$$

where $A_b \in \mathbb{R}^{n-1 \times m}$ is called the *basis incidence matrix* of G, defined as $A_b = [a_{uj}]$, where $a_{uj} = 1, \ -1$ or 0, if edge j is incident on node u and directed away from it, is incident on node u and directed toward it, or is not incident on node u, respectively. A_b has $n-1$ rows, each row corresponding to one node in G, except the reference node. For a measurement graph with node variables $x_u \in \mathbb{R}^k$, the we call the matrix $\mathcal{A}_b \in \mathbb{R}^{k(n-1)\times km}$ defined in (2) the *Generalized Basis Incidence Matrix*.

This optimal estimate $\hat{X}^*$ with the measurement model 2 is the solution to the following system of linear equations [4]:

$$\mathcal{L}\hat{X}^* = \mathbf{b}, \tag{3}$$
$$\text{where} \quad \mathcal{L} := (\mathcal{A}_b \mathscr{P}^{-1} \mathcal{A}_b^T), \;\; \mathbf{b} := \mathcal{A}_b \mathscr{P}^{-1} Z,$$

and $\mathscr{P} := \mathrm{E}[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T]$ is the covariance matrix of the measurement error vector. The error covariance of the optimal estimate $\mathrm{E}[(X-\hat{X}^*)(X-\hat{X}^*)^T]$ is equal to [4] $\mathcal{L}^{-1}$. Since the measurement errors on two different edges are assumed to be uncorrelated, $\mathscr{P}$ is a symmetric positive definite block diagonal matrix with the measurement error covariances along the diagonal: $\mathscr{P} = \mathrm{diag}(P_1, P_2, \dots, P_m) \in \mathbb{R}^{km \times km}$, where P_e is the covariance of the measurement represented by the edge $e \in \mathbf{E}$.

A path from a node to another node that does not respect the orientation of the edges is called an *undirected path*. A directed graph is said to be *weakly connected* if there is a undirected path from any node to any other node. In this paper we consider only weakly connected graphs. Under this assumption, the estimation error covariance always exists and is of finite norm [2], and therefore the optimal estimate $\hat{X}^*$ is unique for a given set of measurements Z. We call the matrix $\mathcal{L}$ the *Weighted Generalized Grounded Laplacian.*

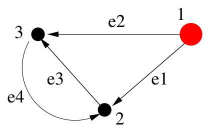

Fig. 1. A measurement graph G with 3 nodes and 4 edges. Node 1 is the reference.

As a simple example, consider the weakly connected measurement graph G shown in figure 1. The basis incidence matrix for this graph is $A_b = \left[\begin{smallmatrix} -1 & 0 & 1 & -1 \\ 0 & -1 & -1 & 1 \end{smallmatrix}\right]$. Writing the measurement equations (1) for the four edges explicitly shows how we get (2) from (1):

$$\begin{bmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \\ \zeta_4 \end{bmatrix} = \begin{bmatrix} -I_k & 0 \\ 0 & -I_k \\ I_k & -I_k \\ -I_k & I_k \end{bmatrix} \begin{bmatrix} x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \end{bmatrix}$$

For the graph in figure 1, with every measurement covariance being I_k, (3) becomes

$$\begin{bmatrix} 3I_k & -2I_k \\ -2I_k & 3I_k \end{bmatrix} \begin{bmatrix} \hat{x}_2^* \\ \hat{x}_3^* \end{bmatrix} = \begin{bmatrix} \zeta_3 - \zeta_1 - \zeta_4 \\ \zeta_4 - \zeta_2 - \zeta_3 \end{bmatrix}. \tag{4}$$

III. Distributed Estimation with Perfect Communication Channels

In order to compute the optimal estimate $\hat{X}^*$ by solving the equations (3) directly, one needs all the measurements and their covariances (Z, $\mathscr{P}$), and topology of the graph ($\mathcal{A}_b$). Our goal is to compute the estimate in a distributed manner, employing only local communication. We use the Jacobi iterative method to achieve this. A discussion of the Jacobi method can be found in standard textbooks [5], hence we refrain from describing it here. Instead we make the presentation of the algorithm self-contained. At first we assume that there are no communication failures, and design an algorithm for that scenario. In section IV, we modify the algorithm to make it robust to temporary communication failures.

A node u' is said to be a neighbor of another node v' in a graph $(\mathbf{V}', \mathbf{E}')$ if there is an edge $(u', v') \in \mathbf{E}'$ or $(v', u') \in \mathbf{E}'$. The set of neighbors of u in the measurement graph G are denoted by $\mathcal{N}_u$. We assume that after deployment of the network, the nodes detect their neighbors and exchange their relative measurements well as the associated covariances. So every node has access to the measurements on the edges that are incident on it, whether the edge is directed to or away from it. Each node uses the measurements obtained initially for all future computation. Arguably, in future inter-node communication, additional measurements between the same node pair may become available. However, we want to compare the estimate produced by the proposed algorithm with the optimal estimate, and the optimal estimate is defined for a given set of measurements. Therefore we assume that the measurements are given a priori and the measurement graph does not change with time. For simplicity, it is assumed that computation time is negligible compared to communication time.

SCG OPTIMAL ESTIMATOR[1] : After the deployment of the network, the reference node initializes its "estimate" to 0 and never changes it. Every other nodes initializes its estimate to an arbitrary value. At the start of iteration $i+1$, a node sends its most recent estimate $\hat{x}_p^{(i)}$ to its neighbors along with the corresponding iteration number i. It also gathers the i-th estimates of its neighbors, $\hat{x}_v^{(i)}$, $v \in \mathcal{N}_p$, and then updates its estimate by solving the following k equations for $\hat{x}_p^{(i+1)}$:

$$M_p \hat{x}_p^{(i+1)} = b_p + \sum_{(p,v)\in\mathbf{E}} P_{pv}^{-1}\hat{x}_v^{(i)} + \sum_{(v,p)\in\mathbf{E}} P_{vp}^{-1}\hat{x}_v^{(i)}, \quad (5)$$

$$\text{where} \quad M_p := \sum_{(p,v)\in\mathbf{E}} P_{pv}^{-1} + \sum_{(v,p)\in\mathbf{E}} P_{vp}^{-1}, \quad (6)$$

$$b_p := \sum_{(p,v)\in\mathbf{E}} P_{pv}^{-1}\zeta_{p,v} - \sum_{(v,p)\in\mathbf{E}} P_{vp}^{-1}\zeta_{vp}. \quad (7)$$

To understand where this update equation came from, we note that the optimal estimate of a particular node, say $p \in \mathbf{V}$, satisfies the following equations, which are obtained simply by expanding (3) and using the definitions of M_p and b_p:

$$M_p\hat{x}_p^* - \Big(\sum_{(p,v)\in\mathbf{E}} P_{pv}^{-1}\hat{x}_v^* + \sum_{(v,p)\in\mathbf{E}} P_{vp}^{-1}\hat{x}_v^*\Big) = b_p. \quad (8)$$

If at some iteration i, $\hat{x}_p^{(i+1)} = \hat{x}_p^{(i)}$ for all $p \in \mathbf{V}$, then equations 5 and 8 become equivalent. Thus, if the estimates $\hat{x}_p^{(i)}, \forall p \in \mathbf{V}$ converge to anything (and in a moment we show that they do),they must converge to the optimal estimates.

Note that to compute the update $\hat{x}_p^{(i+1)}$, node p needs M_p, b_p and the neighbors' estimates $\hat{x}_v^{(i)}, \forall v \in \mathcal{N}_p$. The quantities M_p and b_p are computed in the beginning from the measurements and the associated covariances on the edges that are incident on p, so all the computation needs only local information. The algorithm is summarized in table I.

```
Name:            SCG OPTIMAL ESTIMATOR
Goal:            Compute estimates of node variables
                 that approach the optimal estimate.
Initialization:  $x_1^{(0)} = 0$, $x_u^{(0)}$ is arbitrary for $u \in \mathbf{V} \setminus \{1\}$.
----------------------------------------------------------------
After deployment, every node $p \in \mathbf{V} \setminus \{1\}$ performs:
  1  Detect all neighbors $\mathcal{N}_p$.
  2  Obtain measurements $\zeta_{pv}$, $\zeta_{vp}$ and the associated
     covariances $P_{uv}, P_{vu}$ for every $v \in \mathcal{N}_p$. Compute $M_p$
     and $b_p$ from (6) and (7).
----------------------------------------------------------------
At every iteration $i$, a node $p$ performs:
  3  Send $\hat{x}_p^{(i)}, i$ to every $v \in \mathcal{N}_p$, get $\hat{x}_v^{(i)}$ from all $v \in \mathcal{N}_p$.
  4  Compute $\hat{x}_p^{(i+1)}$ from neighbors' estimates $\hat{x}_v^{(i)}$ for every
     $v \in \mathcal{N}_p$ and previously computed $M_p$ and $b_p$, using (5).
```

TABLE I
PSEUDO-CODE FOR THE SCG OPTIMAL ESTIMATOR ALGORITHM.

A. Correctness and Performance

Let $\hat{X}^{(i)} = [\hat{x}_2^{(i)T}, \ldots, \hat{x}_n^{(i)T}]^T$ be the vector of node estimates after the ith iteration of the algorithm has been completed. One iteration is said to be complete when all nodes update their estimate once. The error at the ith iteration is $\hat{X}^{(i)} - \hat{X}^*$, where $\hat{X}^*$ is the optimal estimate. Let $\varepsilon^{(i)} := \|\hat{X}^{(i)} - \hat{X}^*\|/\|\hat{X}^{(0)} - \hat{X}^*\|$. The algorithm is said to be *correct* if the $\varepsilon^{(i)} \to 0$ as $i \to \infty$ for any initial condition $X^{(0)}$.

Apart from correctness, a quantifiable measure of performance is also desirable. In networks with energy and bandwidth constrained sensor nodes, achieving a good accuracy with as few iterations as possible is important. In light of this, we use as performance metric the number of iterations i required so that $\varepsilon^{(i)}$ achieves a given value ϵ.

To make the analysis of correctness and performance of the algorithm tractable, we make the following additional assumption:

Assumption 1: Either the node variables are scalars ($k = 1$) and the measurement error variances are bounded, or, the node variables are vectors ($k > 1$) and all the measurement error covariance matrices are equal.

For establishing the performance bounds, we restrict our attention to graphs with low Fiedler value. Recall that the Fiedler value, or the Algebraic connectivity, $\alpha(G)$ of a graph G is the second smallest eigenvalue of the graph Laplacian.

Theorem 1. *Consider a weakly connected measurement graph G which satisfies Assumption* 1. *Then the proposed algorithm is correct. If, in addition,* $\alpha(G) << 1$, *for every* $0 < \epsilon < 1$, *the number of iterations* $N(\epsilon)$ *required so that* $\varepsilon^{(i)} < \epsilon, \forall i > N(\epsilon)$, *satisfies*

$$N(\epsilon) > \Omega\big(\frac{|\log \epsilon|}{\alpha(G)}\big). \qquad \square$$

Proof: When assumption 1 holds, the algorithm is simply a Jacobi iteration on a non-singular M matrix [6], and the Jacobi method is correct if the spectral radius of the Jacobi iterative operator is < 1. It follows from standard results (see Theorem 7.5.2 in [6]) that is true in our case. For the lower bound on performance, see [7]. ■

Remark 1: The reason for considering graphs with low Fiedler value is that for most large ad-hoc networks, which

[1]SCG: Static communication Graph

are the networks of interest here, the Fiedler value is generally quite small. For example, the Fiedler value of a 10×10 grid graph is 0.094.

B. Communication Cost

An important performance measure for a distributed algorithm is the total number of messages that have to be exchanged before a given accuracy is achieved. In sensor networks with energy constrained nodes and unreliable communication channels, this issue is particularly acute. In the SCG OPTIMAL ESTIMATOR algorithm, every pair of neighboring nodes has to exchange two messages per iteration – to tell each other their current estimates. Therefore it will take $2mi$ messages to complete i iterations, where m is the number of edges.

For comparison, it takes at most nD_G messages to transmit all the measurements and node IDs to a central node that can then compute the optimal estimate directly, where n is the number of nodes and D_G is the diameter of G. The diameter of a graph is the length of the longest shortest path between any two nodes. Interestingly, communication cost of the centralized estimator can be *lower* than that of the distributed one, especially if a higher accuracy is desired.

IV. DISTRIBUTED ESTIMATION WITH FAULTY COMMUNICATION

One desirable attribute of any distributed algorithm is robustness to communication failures, such failures being unavoidable in practice. We modify our algorithm slightly to make it robust to such failures. The resulting algorithm is call the DCG OPTIMAL ESTIMATOR[2] algorithm. We assume for the sake of simplicity that during the estimation process, no nodes fail permanently and no new nodes become active. This assumption does not place any restriction on employing the algorithm when there are permanent changes to the network. It is made solely to facilitates a fair comparison between the estimates produced by the algorithm to the optimal one, the latter being defined only for a given set of node variables and measurements.

Since a neighbor may become unavailable at any time, every node stores in its local memory the estimates of its neighbors' variables recorded from the last successful communication. We denote by $(\hat{x}_v)_p^{(i)}$ the estimate of x_v kept in p's local memory at the end of the ith iteration. If the last successful communication between p and v took place during the jth iteration, $j < i$, then $(\hat{x}_v)_p^{(i)} = \hat{x}_v^{(j)}$. We assume that computation time is negligible compared to communication time.

DCG OPTIMAL ESTIMATOR: Let $t_p^{(i)}$ be the local time at node p in the beginning of the $i+1$th iteration. At this time, every node p tries to communicate with all its neighbors $\mathcal{N}_p$ for $\tau_c(i,p)$ seconds, after which it stops attempts at communication until the next iteration. Let $\overline{\mathcal{N}}_p^{(i)} \subseteq \mathcal{N}_p$ be the set of nodes p is able to and get data from successfully during the time period $(t_p^{(i)}, t_p^{(i)} + \tau_c)$. Thus, node p gets from every $v \in \overline{\mathcal{N}}_p^{(i)}$ its most recent estimates $\hat{x}_v^{(i)}$. Moreover, it sends its own most recent estimate, $\hat{x}_p^{(i)}$, to as many nodes in $\mathcal{N}_p$ as possible during this period. After the communication period of τ_c seconds, node p then updates its copy of its neighbors' estimates with the recent estimates gathered: $(\hat{x}_v)_p^{(i)} \leftarrow \hat{x}_v^{(i)},\ v \in \overline{\mathcal{N}}_p^{(i)}$. For the nodes it was not able to get data from, it keeps the local copies of their estimates unchanged: $(\hat{x}_v)_p^{(i)} \leftarrow (\hat{x}_v)_p^{(i-1)},\ \forall v \in \mathcal{N}_p \setminus \overline{\mathcal{N}}_p^{(i)}$. After this, node p computes its own estimate update, $\hat{x}_p^{(i+1)}$, by solving the following system of k linear equations:

$$M_p \hat{x}_p^{(i+1)} = b_p + \sum_{(p,v)\in\mathbf{E}} P_{pv}^{-1} (\hat{x}_v)_p^{(i)} + \sum_{(v,p)\in\mathbf{E}} P_{vp}^{-1} (\hat{x}_v)_p^{(i)}.$$

The pseudo-code in Table II implements the algorithm just described. An additional difference in the algorithm described in Table II from the first algorithm is the initialization scheme described in V. The DCG OPTIMAL ESTIMATOR algorithm in its final form uses therefore not the update equation shown above, but (9).

A. Correctness under Faulty Communication

In the presence of temporary link failures, the DCG OPTIMAL ESTIMATOR algorithm executes what are known in parallel computing literature as *asynchronous Iterations*. This is a well-studied problem and the conditions under which it converges are known [8], [9]. Suppose a node p is at the i-th iteration, and the last successful communication between p and one of its neighbors v took place at the jth iteration, with $j < i$, meaning that the previous $i - j$ communications between the nodes had failed.

Theorem 2. *Consider a weakly connected measurement graph G which satisfies Assumption 1. Then the* DCG OPTIMAL ESTIMATOR *algorithm is correct if there is a positive integer $\ell < \infty$ such that the number of consecutive communication failures between every pair of neighboring nodes in G is less than ℓ.* □

Proof: This follows from theorem 4.1 in [9] which states that asynchronous iterations for a system of linear equations converge to the correct solution when the spectral radius of the iteration operator is < 1, and the proof of theorem 1 where it was shown that when Assumption 1 is satisfied, the spectral radius of the Jacobi iterative operator is < 1 for the system of linear equations considered in this paper. ■

V. IMPROVING PERFORMANCE

It may be possible to improve the convergence rate by using other iterative techniques such as Gauss – Siedel, SOR or the conjugate gradient [5] methods, or even by preconditioning, but any such improvement will come at the cost of increased communication.

For the problem at hand, however, one feasible strategy of improving performance without improving the convergence

[2]DCG: Dynamic Communication Graph

rate is to reduce the initial error by choosing a "better" initial condition. The question is how to provide a node with a "better" initial condition without requiring much more communication or computation. We add the following modification to the DCG OPTIMAL ESTIMATOR algorithm to help a node detect which of its neighbors have "good" estimates so that it may update its estimate based only on those neighbors. After a while, all neighbors will be recognized to have good estimates. Therefore this modification makes only the initial phase of the algorithm's execution different from the previously described algorithm.

After deployment of the network, every node other than the reference initializes a flag `have_estimate` to 0, meaning it has no estimate of its variable. The reference node initializes its flag to 1. During every inter-update communication, a node gets from its neighbors their current estimates and the values of their flags. If the recent communication fails, it uses the flag values gathered during the last successful communication. The node then computes its update based *only on* the estimates of those neighbors whose flags were 1 in the recently concluded communication. If any of its neighbors had its flag as 1, after computing its own estimate it sets its own flag to 1 and *never resets it*. Otherwise it keeps the flag at 0 and does not compute any estimate. At the beginning, only the reference has an estimate. In the first iteration, neighbors of the reference can compute estimates by adding measurements they share with the reference, and then update their `have_estimate` flags to 1. In the next iteration, their neighbors do the same, and so on. This way, a node can detect if global coordinate information from the reference node has reached it. With successive iteration, the accuracy increases as the algorithm converges to the optimal estimate.

Let $h_v^{(i)}$ be the value of the flag `have_estimate` of node v at iteration i and $(h_v)_p^{(i)}$ be the local copy of this variable at node p. At iteration i, let $\mathcal{N}_p^h(i) \subseteq \mathcal{N}_p$ be the set of neighbors of $p \in \mathbf{V}$ that p knows have their flag values at 1, i.e, $\mathcal{N}_p^h(i) := \{u \in \mathcal{N}_p | (h_u)_p^{(i)} = 1\}$. When $\mathcal{N}_p^h(i)$ is not empty, node p will be employ the following update equations:

$$M_p^h \hat{x}_p^{(i+1)} = b_p^h + \sum_{\substack{(p,v)\in\mathbf{E} \\ h(v)=1}} P_{pv}^{-1} (\hat{x}_v)_p^{(i)} + \sum_{\substack{(v,p)\in\mathbf{E} \\ h(v)=1}} P_{vp}^{-1} (\hat{x}_v)_p^{(i)}, \qquad (9)$$

where

$$M_p^h := \sum_{\substack{(p,v)\in\mathbf{E} \\ h(v)=1}} P_{pv}^{-1} + \sum_{\substack{(v,p)\in\mathbf{E} \\ h(v)=1}} P_{vp}^{-1},$$

$$b_p^h := \sum_{\substack{(p,v)\in\mathbf{E} \\ h(v)=1}} P_{pv}^{-1} \zeta_{pv} - \sum_{\substack{(v,p)\in\mathbf{E} \\ h(v)=1}} P_{vp}^{-1} \zeta_{vp}.$$

After a while, all of p's neighbors will have their flags at 1, at which point $M_p^h = M_p$ and $b_p^h = b_p$. This modification is included in the summary of the DCG OPTIMAL ESTIMATOR algorithm presented in table II.

```
Name:            DCG OPTIMAL ESTIMATOR
Goal:            Compute estimates of node variables that approach the optimal estimate in the presence of
                 faulty communication links.
Data:            A rule for every node to determine its communication time-out interval
                 $\tau_c$ (seconds) at the $i$ iteration.
Initialization:  $x_1^{(0)} = 0,\ h_1^{(0)} = 1$.
                 $x_v(0) = \emptyset,\ \forall v \in \mathbf{V} \setminus \{1\}$ and
                 $h_u^{(0)} = 0,\ \forall v \in \mathbf{V} \setminus \{1\}$.
----------------------------------------------------------------
After deployment, every node $p \in \mathbf{V}$ performs:
  1   Detect all neighbors $\mathcal{N}_p$.
  2   Obtain measurements $\zeta_{pv}, \zeta_{vp}$ and the associated
      covariances $P_{uv}, P_{vu}$ for every $v \in \mathcal{N}_p$.
----------------------------------------------------------------
At every iteration $i+1$, during the time interval $(t_p^{(i)}, t_p^{(i)} + \tau_c)$
seconds, every node $p \in \mathbf{V} \setminus \{1\}$ performs:
  3a  Get $\hat{x}_v^{(i)}$, $h_v^{(i)}$ from every $v \in \overline{\mathcal{N}}_p^{(i)}$, and
      send $\hat{x}_p^{(i)}, h_p^{(i)}, i$ to as many nodes in $\mathcal{N}_p$.
  3b  Update local copies of neighbors' estimates as
      IF $v \in \overline{\mathcal{N}}_p^{(i)}$
          $(\hat{x}_v)_p^{(i)} \leftarrow \hat{x}_v^{(i)}$, $(h_v)_p^{(i)} \leftarrow h_v^{(i)}$.
      ELSE
          $(\hat{x}_v)_p^{(i)} \leftarrow (\hat{x}_v)_p^{(i-1)}$, $(h_v)_p^{(i)} \leftarrow (h_v)_p^{(i-1)}$.
At the end of $t_p^{(i)} + \tau_c$ seconds, node $p$ performs
  4   IF $\exists v \in \mathcal{N}_p$ s.t. $(h_v)_p^{(i)} = 1$,
          a) Compute $\overline{b}_p, \overline{M}_p$, then update $\hat{x}_p^{(i+1)}$ using (9).
          b) Set $h_p^{(i+1)} \leftarrow 1$.
      ELSE
          $\hat{x}_p^{(i+1)} \leftarrow \hat{x}_p^{(i)}$, $h_p^{(i+1)} \leftarrow h_p^{(i)}$.
```

TABLE II
PSEUDO-CODE FOR THE DCG OPTIMAL ESTIMATOR ALGORITHM WITH THE INITIALIZATION SCHEME DESCRIBED IN SEC. V. THE VARIABLE $h_v^{(i)}$ IS THE VALUE OF THE FLAG `HAVE_ESTIMATE`OF NODE v AT ITERATION i.

VI. SIMULATION

For simulations, we pick location estimation as an application of the general problem described in this paper. The node variable x_u is node u's position in 2-d Euclidean space. 200 nodes were randomly placed in a 1×1 area (figure 2) and pairs of nodes that are within a range of 0.11 took measurements of each others' relative positions. Every measurement was corrupted by Gaussian noise with a covariance matrix of $Po = 10^{-3} \begin{bmatrix} 0.28 & 0.27 \\ 0.27 & 0.28 \end{bmatrix}$. A single set of measurements were used for all the simulations; the locations estimated by the optimal estimator are also shown in Figure 2. The Fiedler value of this graph was computed to be 0.073.

Figure 3 compares the effect of different initializations on the accuracy achieved by the DCG OPTIMAL ESTIMATOR. The initialization described in V performs better than fixed initialization (all positions initialized to 0) or random initialization (initial positions chosen from a uniform distribution). The error at the i-th iteration as a fraction of the

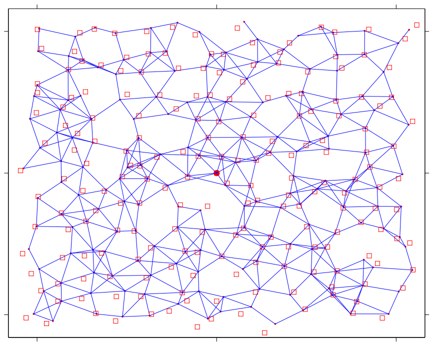

Fig. 2. A sensor network with 200 nodes randomly distributed in a unit square area. The edges of the measurement graph are shown as line segments connecting the true nodes positions. The little squares are the positions estimated by the (centralized) optimal estimator. The reference node is at $(0, 0)$.

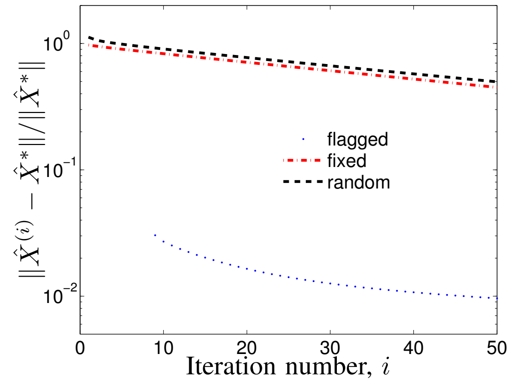

Fig. 3. Comparison different initialization schemes. "Flagged" refers to the initialization scheme described in section V. It took 9 iterations for all the nodes to have their `have_estimate` flags to 1, hence errors are shown only after $i = 9$ for that case.

optimal estimate is shown to make it independent of the unit of length used.

To simulate the algorithm with faulty communication, the following model of link failure was adopted. Every link fails independently of other links, and during every iteration it fails with a probability p_f that is constant for all links. Thus, the time instants that a particular link fails forms a sequence of Poisson points. Figure 4 shows three different error histories for three different failure-probabilities: $p_f = 0.05$, 0.1 and 0.2. In all the cases, the initialization scheme described in section V was used. The error trends show the algorithm converging to the optimal estimate even with link failures. As expected, though, higher failure rates degrade performance.

VII. Summary and Future Work

We have implemented a modified version of the Jacobi iterative scheme to compute the optimal estimator; the resulting algorithm is robust to link failures, distributed, scalable and simple. The main drawback of the proposed algorithms is the potentially large number of iterations that maybe required to achieve a desired accuracy when the measurement graph has a low algebraic connectivity (theorem 1). We improved the convergence of the algorithm by employing a particular initialization scheme. Other ways to reduce the number of iterations or messages needed to achieve a given accuracy will be explored in future research. Another important issue is that of security, when one or more node estimates may be manipulated by a hostile party. Making the algorithm robust to security threats and extending it to be able to compute the variance of the estimate are some of the avenues for future research.

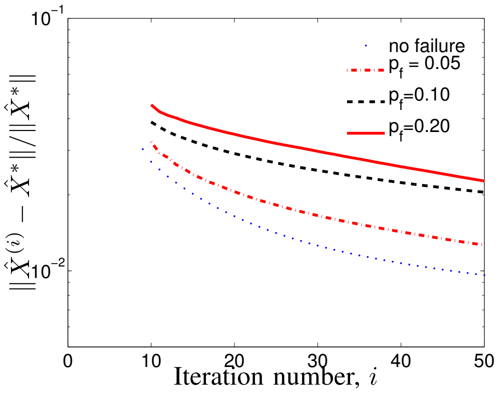

Fig. 4. Estimation error with iteration number with link failures. Three different failure probabilities are compared with the case of no link failure.

References

[1] R. Karp, J. Elson, D. Estrin, and S. Shenker, "Optimal and global time synchronization in sensornets," Center for Embedded Networked Sensing,Univ. of California, Los Angeles, Tech. Rep., 2003.

[2] P. Barooah and J. P. Hespanha, "Estimation from relative measurements: Error bounds from electrical analogy," in *Proceedings of the 2nd International Conference on Intelligent Sensing and Information Processing, Chennai, India.*, January 2005.

[3] V. Delouille, R. Neelamani, and R. Baraniuk, "Robust distributed estimation in sensor networks using the embedded polygon algorithms," in *Third International Workshop on Information Processing in Sensor Networks (IPSN)*, April 26-27 2004.

[4] J. M. Mendel, *Lessons in Estimation Theory for Signal Processing, Communications and Control*, A. V. Oppenheim, Ed. Prentice Hall P T R, 1995.

[5] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. The John Hopkins University Press, 1996.

[6] A. Berman and R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, ser. Computer Science and Applied Mathematics. Academic Press, 1979.

[7] P. Barooah and J. P. Hespanha, "Distributed estimation from relative measurements in sensor networks," University of California, Santa Barbara, Tech. Rep., September 2005.

[8] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Englewood Cliffs, NJ, USA: Prentice-Hall, Inc., 1989.

[9] A. Frommer and D. Szyld, "On asynchronous iterations," *Journal of Comp. Appl. Math., 123*, pp. 201–216, 2000.