Optimal Estimation on the Graph Cycle Space

Wm. Joshua Russell* Daniel J. Klein João P. Hespanha

{wjrussell,djklein,hespanha}@ece.ucsb.edu

Abstract

This paper addresses the problem of estimating the states of a group of agents from noisy measurements of pairwise differences between agents' states. The agents can be viewed as nodes in a graph and the relative measurements between agents as the graph's edges. We propose a new distributed algorithm that exploits the existence of cycles in the graph to compute the best linear state estimates.

For large graphs, the new algorithm significantly reduces the total number of message exchanges that are needed to obtain an optimal estimate. We show that the new algorithm is guaranteed to converge for planar graphs and provide explicit formulas for its converge rate for regular lattices.

I. Introduction

As large scale sensor networks have become more prevalent in military and consumer applications, increasing interest has been paid to distributed estimation and its applications. Common applications for distributed estimation include localization, target tracking, and sensor fusion. The goal of all estimation problems is to combine noisy measurements of some unknown network state into an estimate that is a measurably better (typically lower error variance) representation of the network's state than the measurements. As sensor networks grow in size, "distributed estimation" algorithms are used to reduce communication and computational loads within the sensor network [10]. Distributed estimation algorithms are so named because computation is distributed to various computational elements in the network. Distributed estimation algorithms typically approximate or converge to some existing (non-distributed) optimal estimation method.

There have been many approaches to the distributed estimation problem for different applications. Among the simplest distributed estimation problems is the linear consensus problem addressed in [11], [14], [26], [29], [30], in which the state to be estimated is observable to all sensors in the network. More

This material is based upon work supported by the Institute for Collaborative Biotechnologies through grant DAAD19-03-D-0004 from the U.S. Army Research Office and by the National Science Foundation under Grant No. CNS-0720842.

challenging problems such as distributed Kalman filtering [1], [5], [6], [20], [24], [27] and distributed Kalman smoothing [3], [4], [7] have been used for estimation of a dynamic states. More challenging still, are the problem of track-to-track fusion [8], [19], in which highly correlated tracks are fused into an estimate, and the problem of localization from time-difference-of-arrival measurements [12], where one must estimate the intersection of hyperbolic functions.

This paper proposes a solution to the distributed estimation from relative difference measurements problem explored in [2], [17], [22]. The relative difference measurement problem assumes that each node in a network has an associated state that cannot be measured absolutely (e.g. a node's global position), but a node can measure the relative difference between its state and its neighbor's states (e.g. a node can measure the range and bearing to its neighbors). Barooah et al. explored several distributed algorithms for this problem that converge to the optimal minimum error variance solution [2]. Barooah's algorithms assume that nodes have limited topological knowledge and communication abilities. In these algorithms, each node updates an estimate of its own state synchronously using an affine transformation of neighboring state estimates. Methods that estimate the node state directly and referred to as *node estimation* methods. The node estimation methods introduced in [2] were shown to converge to the optimal estimate and were shown to be robust to interruptions in communication.

The work reported in the present paper is motivated by the observation that, in the absence of noise, the sum of the relative difference measurements around any cycle of the graph should be zero. However, due to measurement noise, these sums are generally not equal to zero — we refer to these sums as discrepancies. The Cycle Space Estimation algorithms introduced in this paper consist of two steps: first, one computes a "corrected" version of the measurement set that has zero discrepancy and then one uses this corrected measurement set to estimate the state variables. The first step can be done either in a centralized or distributed fashion. Its decentralized version utilizes a Jacobi-like algorithm [21] on a "cycles graph" whose nodes correspond to the cycles of the original graph. The second step can also be done in a centralized or distributed fashion. In the latter case, convergence is guaranteed within a finite number of steps (equal to the graph diameter) using the so-called flagged initialization algorithm [2].

For large classes of graphs, the new distributed algorithms converge significantly faster than previous algorithms. Our results show that, for large graphs, the Cycle Space Estimation algorithm provides significant gains in terms of the number of messages that are required to compute optimal estimates. This happens because the cycles graph often has much better convergence properties than the original graph. We show through Monte Carlo simulations that this is the case for several classes of regular graphs, which include square, triangular, and hexagonal lattices of varying size. For square lattices, we are also able to determine explicit formulas to compute the convergence rate of the distributed algorithm.

This paper expands upon results previously presented in conference [25]. Specifically, in this paper we present new convergence results for the distributed algorithm presented herein as well as expanded simulation results comparing this method to a preexisting distributed estimation algorithm.

This paper proceeds in the following manner. Section II provides a review of the problem statement and optimal estimation. In Section III, the Centralized Optimal Cycle Space Estimate is introduced. The Distributed Optimal Cycle Space Estimation algorithm is covered in Section IV. The convergence properties of Distributed Optimal Cycle Space Estimation algorithm are discussed in Section V. Simulation results are given in Section VI. Conclusions and future research goals are discussed in Section VII.

II. Centralized Node Estimation

Consider a network of sensors, each associated with a vector-valued state in d-dimensions. The sensors take relative measurements, which consist of the difference between the states of two sensors. The measurements are corrupted by uncorrelated zero-mean Gaussian noise with (possibly) distinct covariances. The problem we are interested in is to estimate each sensor's state in an optimal fashion.

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be the measurement directed graph associated with the estimation problem. The node set, $\mathcal{V}$, represents the set of sensors. The edge set, $\mathcal{E}$, contains all pairs of sensors for which there is a relative measurement. By convention, the direction of the edge goes from the node that appears with the plus sign to the node that appears with the minus sign in the relative difference. Let n and m be the cardinality of $\mathcal{V}$ and the cardinality of $\mathcal{E}$ respectively. The measurement graph can be represented compactly by its incidence matrix $\bar{B} = \left[\bar{b}_{ij}\right]$, $i \in \{1, 2, ..., n\}$, $j \in \{1, 2, ..., m\}$,

$$\bar{b}_{ij} \equiv \begin{cases} 1 & \text{if edge } j \text{ leaves node } i \\ -1 & \text{if edge } j \text{ enters node } i \\ 0 & \text{otherwise.} \end{cases}$$

Denote by B_g the Kronecker product of $\bar{B}$ and an identity matrix of size d, by B_g^T the transpose of B_g, by $\tilde{z}$ the vector of stacked measurements, by ε the vector of measurement noise, and by x_g the vector of states (note that $\tilde{z}$ and ε are og length dm and x_g is of length dn), we have that:

$$\tilde{z} = B_g^T x_g + \varepsilon. \tag{1}$$

Because measurements are taken as relative differences in state, any estimate, $\hat{x}_g$, of x_g in (1) is unique only up to an additive constant. To remove this degree of freedom, a subset of the nodes are taken as references and it is assumed that the states of these nodes are known to the corresponding sensors. Often a single reference node is used and its state is assumed to be zero, but here we consider the more general case of multiple references. The whole state, x_g, can be split into two vectors x_r and x containing, respectively, the states of the references nodes and the states of the nodes that need to be estimated. Likewise, B_g can be split into two matrices, B_r and B containing, respectively, the rows of B_g corresponding to reference nodes and the rows of B_g corresponding to nodes whose state needs to be estimated. Moving the reference states (and the corresponding columns of B^T) to the left-hand side of (1), we obtain:

$$\tilde{z} - B_r^T x_r = B^T x + \varepsilon.$$

It was shown in [2] that given a weakly connected (i.e. disregarding edge direction, the graph is connected) measurement graph, the Best Linear Unbiased Estimator (BLUE) for the non-reference states x given the measurements $\tilde{z}$ is:

$$\hat{x}^* = (BP^{-1}B^T)^{-1}BP^{-1}(\tilde{z} - B_r^T x_r), \tag{2}$$

where $P \equiv E[\varepsilon\varepsilon']$. The expression in (2) gives the centralized state estimate with minimum variance.

III. Centralized Optimal Cycle Space Estimation

To gain insight into the approach followed here, it is convenient to regard the states of the nodes as voltages in an electric circuit that takes the shape of the measurement graph. The measurements would then correspond to tensions measured across the edges of the graph. When the graph has cycles, one should obtain zero by summing the tensions along a cycle (a la Kirchhoff's Voltage Law), but because of measurement errors this is generally not the case. In this light, one can view the estimation procedure as computing a set of voltages that are internally consistent, in the sense that they add up to zero around any cycle (i.e. the set satisfies KVL).

Let us define the vector of *optimal tension estimates* as:

$$\hat{z}^* \equiv \begin{bmatrix} B_r^T & B^T \end{bmatrix} \begin{bmatrix} x_r \\ \hat{x}^* \end{bmatrix}, \tag{3}$$

where $\hat{x}^*$ is the minimum variance estimate for x from (2). The vector $\hat{z}^*$ defined above is obtained by taking differences between all elements of the optimal estimate $\hat{x}^*$ for which we have measurements available. In the absence of noise, $\hat{x}^*$ would be equal to x and $\hat{z}^*$ would be equal to the measurement vector, $\tilde{z}$.

While it appears from (3) that we first need to compute the optimal estimate $\hat{x}^*$ to compute the optimal tension estimate $\hat{z}^*$, we will see shortly that it is possible to compute $\hat{z}^*$ directly from the measurements $\tilde{z}$, and that it is then straightforward to obtain $\hat{x}^*$ from $\hat{z}^*$. We shall also see that this indirect procedure can be advantageous from the perspective of reducing communication and computation.

A. Cycle Space

A *simple cycle* of $\mathcal{G}$ is a directed path in $\mathcal{G}$ that starts and ends at the same node and has only this one repeated node. By indexing the edges with the natural from numbers 1 to m, we can associate to each simple cycle a *simple cycle vector*, which is defined to be the vector $c = c_j,\ j \in (1, 2, ..., m)$,

$$c_j \equiv \begin{cases} 1 & \text{if cycle traverses edge j forward} \\ -1 & \text{if cycle traverses edge j backward} \\ 0 & \text{otherwise.} \end{cases}$$

The *cycle space*, $\mathcal{C}$, of $\mathcal{G}$ is then defined to be the subspace of $\mathbb{R}^m$ generated by the simple cycle vectors associated with the simple cycles in $\mathcal{G}$. Subsequently, C will be used to denote the Kronecker product of a matrix whose columns are simple cycle vectors that form a basis for the cycle space and the identity matrix of size d, the dimension of the measurements. Considering that every column of B_g corresponds to an edge in $\mathcal{G}$, C is a basis for the nullspace of B_g [23]. The cycle space matrix, C, has height dm and width dr, where r is the dimension of $\mathcal{C}$.

The Centralized Optimal Cycle Space Estimation (COCSE) algorithm uses $\tilde{z}$ and C to compute $\hat{z}^*$. The premise of the COCSE algorithm is that one can add a correction term to the measurements, $\tilde{z}$, to form the optimal set of tensions, $\hat{z}^*$.

B. The Centralized Optimal Cycle Space Estimate

Theorem 1: [The Optimal Cycle Space Estimate]

The optimal tension estimate,

$$\hat{z}^* \equiv \begin{bmatrix} B_r^T & B^T \end{bmatrix} \begin{bmatrix} x_r \\ \hat{x}^* \end{bmatrix},$$

is the solution to the quadratic programming problem:

$$\hat{z}^* = \underset{\hat{z}}{\operatorname{argmin}}[(\tilde{z}-\hat{z})^T P^{-1}(\tilde{z}-\hat{z})] \qquad s.t.\ C^T\hat{z} = 0. \tag{4}$$

$$= \begin{cases} \tilde{z} - PC(C^T PC)^{-1}C^T\tilde{z} & \text{if } r \neq 0. \\ \tilde{z} & \text{if } r = 0 \end{cases} \tag{5}$$

Proof: When $r = 0$ ($\mathcal{G}$ has no cycles), C^T is trivial and the constraint in (4) is satisfied for any $\hat{z}$. Consequentially, $\tilde{z}^* = \hat{z}$. Furthermore, with no cycles in the graph, the BLUE of a particular node is simply the summation of measurements along the unique path from the reference node to the node in question. As such, the difference in the BLUEs of two neighboring nodes is simply the measurement connecting these nodes. Hence, the optimal tension set for which (3) holds is the measurement set, z.

When $r \neq 0$, the Lagrangian for (4) is:

$$L(\hat{z}, \lambda) = (\tilde{z}-\hat{z})^T P^{-1}(\tilde{z}-\hat{z}) + \lambda^T C^T \hat{z}.$$

Taking the partial derivatives of the Lagrangian equal to zero:

$$\frac{\partial L}{\partial \hat{z}} = 2P^{-1}(\hat{z}-\tilde{z}) + C\lambda = 0$$
$$\frac{\partial L}{\partial \lambda} = C^T\hat{z} = 0,$$

which can be written in the form:

$$\begin{bmatrix} 2P^{-1} & C \\ C^T & 0 \end{bmatrix} \begin{bmatrix} \hat{z} \\ \lambda \end{bmatrix} = \begin{bmatrix} 2P^{-1}\tilde{z} \\ 0 \end{bmatrix}. \tag{6}$$

Using the Schur Complement, (6) can be solved to find $\hat{z}^*$:

$$\hat{z}^* = \left[\tfrac{1}{2}P - \tfrac{1}{2}PC^T(CPC^T)^{-1}CP\right] 2P^{-1}\tilde{z}$$
$$= [I - PC^T(CPC^T)^{-1}C]\tilde{z}. \tag{7}$$

To show that (7) is related to the BLUE by (3), we make use of Lemma 1 (in the Appendix). Lemma 1 states that for two full column rank matrices, α and β^T, the image of α is the nullspace of β if and only if

$$\alpha(\alpha^T\alpha)^{-1}\alpha^T = I - \beta^T(\beta\beta^T)^{-1}\beta.$$

Consider the matrices $\beta = \text{image}(BP^{-\frac{1}{2}})$ and $\alpha = \text{image}(P^{\frac{1}{2}}C)$. Note that $\text{null}(\beta) = \text{image}(\alpha)$. Considering Lemma 1 and the fact that $C^T B_r^T = 0$, the following equalities hold:

$$\begin{aligned}
0 &= [\alpha(\alpha^T\alpha)^{-1}C^T P^{\frac{1}{2}}]P^{-\frac{1}{2}}B_r^T x_r \\
&= [\alpha(\alpha^T\alpha)^{-1}\alpha^T]P^{-\frac{1}{2}}B_r^T x_r \\
&= [I - \beta^T(\beta\beta^T)^{-1}\beta]P^{-\frac{1}{2}}B_r^T x_r \\
&= P^{\frac{1}{2}}[I - \beta^T(\beta\beta^T)^{-1}\beta]P^{-\frac{1}{2}}B_r^T x_r \\
&= [I - P^{\frac{1}{2}}\beta^T(\beta\beta^T)^{-1}\beta P^{-\frac{1}{2}}]B_r^T x_r \\
&= [I - B^T(BP^{-1}B^T)^{-1}BP^{-1}]B_r^T x_r.
\end{aligned} \tag{8}$$

Consider again Lemma 1:

$$\begin{aligned}
I - \alpha(\alpha^T\alpha)^{-1}\alpha^T &= \beta^T(\beta\beta^T)^{-1}\beta \\
I - P^{\frac{1}{2}}\alpha(\alpha^T\alpha)^{-1}\alpha^T P^{-\frac{1}{2}} &= P^{\frac{1}{2}}\beta^T(\beta\beta^T)^{-1}\beta P^{-\frac{1}{2}} \\
I - PC(C^TPC)^{-1}C^T &= B^T(BP^{-1}B^T)^{-1}BP^{-1} \\
[I - PC(C^TPC)^{-1}C^T]\tilde{z} &= B^T(BP^{-1}B^T)^{-1}BP^{-1}\tilde{z}.
\end{aligned} \tag{9}$$

Now, add (8) to (9):

$$\begin{aligned}
[I - PC(C^TPC)^{-1}C^T]\tilde{z} &= B^T(BP^{-1}B^T)^{-1}BP^{-1}\tilde{z} + [I - B^T(BP^{-1}B^T)^{-1}BP^{-1}]B_r^T x_r \\
&= B_r^T x_r + B^T(BP^{-1}B^T)^{-1}BP^{-1}(\tilde{z} - B_r^T x_r) \\
&= B_r^T x_r + B^T\hat{x}^* \equiv \hat{z}^*.
\end{aligned}$$

■

In light of Theorem 1 the following interpretation of $\hat{z}^*$ can be made: The set of tensions corresponding to the BLUE, $\hat{z}^*$, is the set of tensions with the least square difference from the set of measurements, $\tilde{z}$. That is to say, it is the set of tensions with minimum squared edge modification.

C. Cycle Laplacian

Motivated by (5) we derive the *cycle Laplacian matrix*,

$$\mathcal{L}_c \equiv C^T PC. \tag{10}$$

While $\mathcal{L}_c$ is generally not doubly stochastic, we call it a Laplacian because it characterizes the weighted cycle degree and the weighted cycle adjacency, which are defined as follows: let $\gamma_i \subset \mathcal{E}$ be the set of edges in c_i, the cycle corresponding to the i^{th} column of the cycle space basis matrix C. The *weighted cycle degree*, D_i, of this cycle is the sum of the error covariance matrices that correspond to the measurements in c_i, i.e., $D_i = \sum_{\ell \in \gamma_i} P_\ell$. The *weighted cycle adjacency*, $A_{i,j}$ between two cycles, c_i and c_j, is the net sum of error covariance matrices corresponding to edges that exist in both cycles. The cycle adjacency between two cycles increases if both cycles traverse the edge in the same direction and decreases if the cycles traverse the edge in opposite directions. The cycle adjacency between c_i and c_j is $A_{ij} = \sum_{\ell \in \gamma_{\{i \cap j\}}} P_\ell c_{\ell i} c_{\ell j}$, where $c_{\ell i}$ is the ℓ^{th} element of c_i (the element in the ℓ^{th} row and i^{th} column of the cycle space matrix, C). If two cycles have no common edges, then the weighted adjacency between the cycles is zero. The weighted cycle degree and weighted cycle adjacency, so derived, are related to the cycle Laplacian by

$$\begin{aligned} \mathcal{L}_c &= D - A, \\ D &= \text{diag}(D_1, \ldots, D_o), \\ A &= [A_{ij}]. \end{aligned}$$

IV. Distributed Estimation in the Cycle Space

In a large network it is often impractical to collect all of the measurements at one node, compute the optimal estimate at that node, and then redistribute the estimate to every other node. Instead, it is desirable to find an estimate of the state in a distributed manner. To construct such a distributed algorithm, we make the following assumptions:

1) A node knows the measurements corresponding its edges,
2) Each node knows how to send messages to each one of its neighbors in $\mathcal{G}$,
3) A basis C for the cycle space of the graph is known and a "leader" has been elected for each simple cycle corresponding to the columns of C,
4) All nodes that belong to one (or more) of the simple cycles associated with a column of C know how to send measurements to the leaders of these cycles (perhaps through multi-hop communication).

The Jacobi-Based Cycle Space Estimation (JBCSE) algorithm is a distributed algorithm that computes optimal tension estimates, $\hat{z}^*$. JBCSE starts with each cycle leader collecting the sum of measurements in its cycle. Each cycle leader then computes the total discrepancy in its cycle by adding the measurements of the cycle: $\Delta_i = \sum_{\ell \in \gamma_i} \tilde{z}_\ell$. Note that all of the discrepancies can be represented in one equation as $\Delta = C^T \tilde{z}$.

Each cycle leader possesses an iteration variable $\hat{y}_i$ that is initialized to zero. Let κ_i be the set of cycles that share at least one edge with cycle i. Then, as the JBCSE algorithm proceeds, at each step each cycle leader:

1) Sends the value of its iteration variable, $\hat{y}_i$, to the leaders of the cycles that share edges with its own cycle, as well as to all the nodes in its cycle,
2) Receives iteration variables, $\hat{y}_{\kappa_i}$ from the leaders of neighboring cycles,
3) Updates its iteration variable,

$$\hat{y}_i^{t+1} = D_i^{-1} \left(\sum_{\ell \in \kappa_i} A_{i\ell}\, \hat{y}_\ell^t - \Delta_i \right). \tag{11}$$

In addition to the iteration above carried out by the cycle leaders, each node also maintains internal estimates of the optimal tensions for all its incident edges. In particular, the estimate of the optimal tension $\hat{z}_j^t$ for edge j is updated as follows:

$$\hat{z}_j^t = \tilde{z}_j + P_j \sum_{\ell \in \Gamma_j} \hat{y}_\ell^t, \tag{12}$$

where $\tilde{z}_j$ is the measurement corresponding to the edge j, P_j the error covariance for that edge, and Γ_j denotes the set of cycles that contain the edge j. The set Γ_j can be empty if the edge does not belong to any cycle, in which case $\hat{z}_j^t = \tilde{z}_j$.

To obtain its current state estimate from the estimates of the optimal tensions, a node needs only to obtain the sum of the tensions along a path between itself and a reference node, which can be accomplished via a distributed flagging method similar to that introduced in [2]. In this way, all non-reference nodes start with a flagged state value, $\hat{x}^0 = \infty$. The first time that a node, say $\mathcal{V}_A$, communicates with a neighbor, say $\mathcal{V}_B$, that has a non-flagged state, $\mathcal{V}_A$ is aware that $\mathcal{V}_B$ is the last node in a path connected to a reference node. $\mathcal{V}_A$ will then update its state, $\hat{x}_A^{t+1}$, to $\mathcal{V}_B$'s state, $\hat{x}_B^t$, plus or minus (depending on the orientation of the edge that connects the nodes, $\mathcal{E}_{AB}$) the tension estimate connecting the two nodes, $\hat{z}_{AB}^t$. As the state estimates, $\hat{x}^t$, and tension estimates, $\hat{z}^t$, are refined, $\mathcal{V}_A$ will maintain its estimate as the current state estimate for $\mathcal{V}_B$, $\hat{x}_B^t$, plus or minus the current tension estimate for the edge connecting the two nodes, $\hat{z}_{AB}^t$. Mathematically, this process can be expressed as:

$$\hat{x}_A^0 \equiv \begin{cases} \infty & \text{if } x_A \in x \\ \text{reference} & \text{if } x_A \in x_r \end{cases}$$

$$\hat{x}_A^{t+1} \equiv \begin{cases} \infty & \text{if } \hat{x}_B^t = \infty \\ \hat{x}_B^t + \hat{z}_{AB}^t & \text{if } \mathcal{E}_{AB} \text{ leaves } \mathcal{V}_A \ \& \ \hat{x}_B^t \neq \infty \\ \hat{x}_B^t - \hat{z}_{AB}^t & \text{if } \mathcal{E}_{AB} \text{ enters } \mathcal{V}_A \ \& \ \hat{x}_B^t \neq \infty \end{cases} \tag{13}$$

Using this flagging method, a breadth first tree (or trees) will grow out from a reference node (or reference nodes) as shown in Figure 1. Under the assumption that the measurement graph is weakly connected and neighbors in the graph can communicate, the number of iterations for the flagged initialization process to complete and all nodes to have a state estimate is less than or equal to the diameter of the graph.

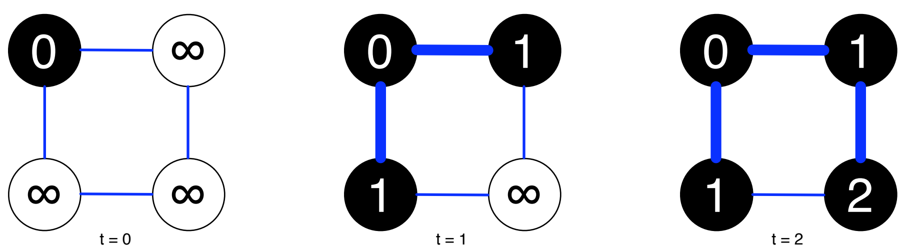


Fig. 1. An example of breadth the breadth first flagging method. In the first time instant only the reference node has an estimate of its state. At $t = 1$ both of the reference node's neighbors have estimates of their states. By $t = 2$, all nodes have state estimates and a breadth first tree has been created for the graph (indicated by bold edges).

For a given graph, there can be many trees rooted at the reference nodes. Choosing different trees will give different initial state estimates, $\hat{x}$. However, as the tension estimates converge to the optimal estimate, $\hat{z}^*$, the state estimates obtained using any tree will converge to the minimum variance estimate, $\hat{x}^*$, because $\hat{z}^*$ is consistent.

V. Convergence of the JBCSE Algorithm

When the iteration variables, $\hat{y}_i$, are stacked into a single vector, $\hat{y}$, the update law (11) can be rewritten:

$$\hat{y}^{t+1} = D^{-1}\left(A\hat{y}^t - \Delta\right) \tag{14}$$

and when the tension estimates are stacked into a single vector, $\hat{z}$, the update law corresponding to (12) can be expressed as:

$$\hat{z}^t = \tilde{z} + PC\hat{y}^t. \tag{15}$$

Once the flagging process (13) has finished the state estimates, $\hat{x}$, can be represented using the equation:

$$\chi^{t+1} = A_T\chi^t + B_T\hat{z}^t, \tag{16}$$

where

$$\chi = \begin{bmatrix} x_r \\ \hat{x}^{t+1} \end{bmatrix},$$

and A_T and B_T are the adjacency and the incidence matrices, respectively, for a directed tree (trees) rooted at a reference node (nodes). Equations (14), (15), and (16) can be combined into the following linear equation representing the JBCSE algorithm:

$$\begin{bmatrix} \hat{y}^+ \\ \chi^+ \end{bmatrix} = \begin{bmatrix} D^{-1}A & 0 \\ B_T PC & A_T \end{bmatrix} \begin{bmatrix} \hat{y} \\ \chi \end{bmatrix} - \begin{bmatrix} D^{-1}C^T \\ -B_T \end{bmatrix} \tilde{z}. \tag{17}$$

A. Condition for Convergence of the JBCSE Algorithm

Theorem 2: [Sufficient Condition for Convergence]

Given the cycle Laplacian of a graph, $\mathcal{L}_c = D - A$, where D and A are the weighted cycle degree matrix and the weighted cycle adjacency matrix, respectively, if the spectral radius of $D^{-1}A$ is strictly less than 1, then the JBCSE algorithm converges, the tension estimate, $\hat{z}^t$, converges to the optimal tension estimate, $\hat{z}^*$, and the state estimate, $\hat{x}$, converges to the BLUE, $\hat{x}^*$.

Proof: In view of (17), the JBCSE algorithm is a cascade linear system; for stability, both sections of the cascade must be stable. All eigenvalues for the adjacency matrix for the directed tree, A_T, are zero. This can be seen by enumerating nodes in the tree according to their proximity to reference nodes. The reference nodes in the graph have the lowest number, followed by neighbors of reference nodes, etc; the leafs on the tree are labeled with the the largest numbers. When this is done, A_T is strictly lower triangular and therefore has only zero eigenvalues. From this, the rate of convergence for (17) is bounded by the spectral radius, $\rho(J)$, of the iteration matrix, $J = D^{-1}A$.

From (14), when $\hat{y}^t$ converges, $\hat{y}^* = D^{-1}(A\hat{y}^* - \Delta)$. The error between $\hat{y}^t$ and $\hat{y}^*$ has the following dynamics:

$$\begin{aligned}\hat{y}^{t+1} - \hat{y}^* &= D^{-1}(A\hat{y}^t - \Delta) - D^{-1}(A\hat{y}^* - \Delta) \\ &= D^{-1}A(\hat{y}^t - \hat{y}^*).\end{aligned}$$

When the spectral radius of $D^{-1}A$ is strictly less than 1, the error between $\hat{y}^t$ and $\hat{y}^*$ goes to zero as t goes to infinity. Similarly, from (15), when $\hat{z}^t$ converges, $\hat{z}^* = \tilde{z} + PC\hat{y}^*$. The error between $\hat{z}^t$ and $\hat{z}^*$ has the following dynamics:

$$\begin{aligned}\hat{z}^{t+1} - \hat{z}^* &= \tilde{z} + PC\hat{y}^t - \tilde{z} + PC\hat{y}^* \\ &= PC(\hat{y}^t - \hat{y}^*).\end{aligned}$$

If the spectral radius of $D^{-1}A$ is less than 1 and $\hat{y}^t - \hat{y}^*$ goes to zero as t goes to infinity, then $\hat{z}^t - \hat{z}^*$ will also go to zero as t goes to infinity. Furthermore, when $\hat{y}^t$ converges:

$$\begin{aligned}\hat{y}^* &= D^{-1}(A\hat{y}^* - \Delta) \\ D\hat{y}^* &= A\hat{y}^* - \Delta \\ (D - A)\hat{y}^* &= -\Delta \\ (C^T PC)\hat{y}^* &= -C^T\tilde{z} \\ \hat{y}^* &= -(C^T PC)^{-1}C^T\tilde{z},\end{aligned}$$

from which, $\hat{z}^* = \tilde{z} - PC(C^T PC)^{-1}C^T\tilde{z}$. Hence the tension estimate, $\hat{z}$ converges to the optimal tension estimate, (5).

Theorem 1, tells us that the set of optimal tension estimates are the pairwise differences of the BLUE, $\hat{x}^*$, corresponding to edges in the graph. As such, summing $\hat{z}^*$ along the directed tree provided by A_T and B_T renders the BLUE. ■

Because we only require C to be a basis for the cycle space, C is not unique. Consequentially, the choice of C affects the behavior of the JBCSE because A and D depend on C. While Theorem 2 provides conditions for convergence based on the spectral radius of $D^{-1}A$, it does little to explain which cycle space matrices lead to convergent cycle space estimates. The optimal choice of C remains an open problem; however, the following convergence results can be used to evaluate potential C matrices.

B. Convergence of JBCSE on a Planar Graph

When a planar graph is embedded in the Euclidean Plane such that no edges cross, the plane is divided into contiguous regions called *faces* (see Figure 2 for an example of graph faces); each graph face characterizes a cycle that forms its boundary [15]. The *faces of the graph* are taken to be the set of interior faces in the graph, which is a basis for the cycle space of the graph [16]. The following theorem provides a convergence result for planar graphs. In this theorem, the faces of the graph are chosen to derive the basis for the cycle space.

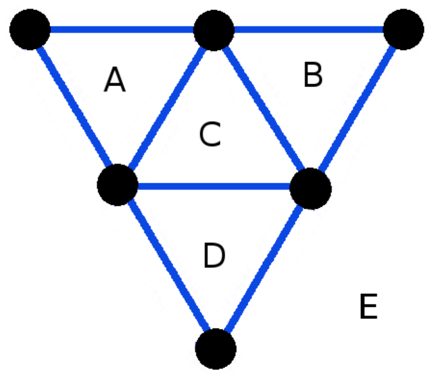


Fig. 2. A simple graph with interior faces A,B,C, and D and exterior face E.

Theorem 3: [Convergence for Planar Graphs]
When $\mathcal{G}$ is a planar graph and C is the basis of the cycle space associated with the internal faces of the graph, then the JBCSE, (17), converges and $\lim_{t\to\infty} \hat{x}^t = \hat{x}^*$, the BLUE of x.

Proof: The cycle degree matrix, D, is a block diagonal matrix whose blocks are sums of covariance matrices. As such, D and D^{-1} are symmetric positive definite matrices. The cycle Laplacian matrix is defined as $\mathcal{L}_c \equiv C^T P C$, where P is symmetric positive definite and C is a basis, hence $\mathcal{L}_c$ is a symmetric positive definite matrix. From [13] (Threorem 7.6.3) all eigenvalues of $D^{-1}\mathcal{L}_c$ are positive, from which:

$$\begin{aligned}
\rho\left(D^{-1}\mathcal{L}_c\right) &> 0 \\
\rho\left(I - D^{-1}\mathcal{L}_c\right) &< 1 \\
\rho\left(I - D^{-1}(D-A)\right) &< 1 \\
\rho\left(D^{-1}A\right) &< 1
\end{aligned}$$

Hence, the JBCSE algorithm converges. ■

C. Convergence Rate of JBCSE on the Faces of a Square Lattice

Currently, no closed form solution exists to find $\rho(J)$ for a general measurement graph. However, a closed form expression for $\rho(J)$ can be found for the JBCSE algorithm applied to a square lattice of measurements with equal error covariances.

Theorem 4: [Convergence Rate of JBCSE on the Faces of a Square Lattice]
Consider a K x L grid of nodes connected by measurements with equal covariances. The convergence rate of the JBCSE algorithm is:

$$\rho(D^{-1}A) = \frac{1}{2}\left[\cos\left(\frac{\pi}{K}\right) + \cos\left(\frac{\pi}{L}\right)\right].$$

Proof: Start by labeling the nodes with grid coordinates: $\mathcal{V} = v_{\bar{k},\bar{l}}$, where $\bar{k} \in \{1, 2, \dots, K\}$ and $\bar{l} \in \{1, 2, \dots, L\}$. Let the columns of C be the set of simple cycle vectors corresponding to the cycles graph, which is a $(K-1)$ x $(L-1)$ grid: $C = c_{k,l}$ where $k \in \{1, 2, \dots, K-1\}$ and $l \in \{1, 2, \dots, L-1\}$.

Because the cycle interconnections form of a grid, the iteration variables, $\hat{y}_{k,l}$, for $k \in \{1, 2, \dots, K-1\}$ and $l \in \{1, 2, \dots, L-1\}$, are updated as per (11):

$$\hat{y}^{+}_{k,l} = \frac{1}{4}(\hat{y}_{k+1,l} + \hat{y}_{k-1,l} + \hat{y}_{k,l+1} + \hat{y}_{k,l-1}) - \frac{1}{4}\Delta_{k,l},$$

where $y_{0,l} = y_{K,l} = y_{k,0} = y_{k,L} = 0$. In steady state:

$$\hat{y}^{*}_{k,l} = \frac{1}{4}(\hat{y}^{*}_{k+1,l} + \hat{y}^{*}_{k-1,l} + \hat{y}^{*}_{k,l+1} + \hat{y}^{*}_{k,l-1}) - \frac{1}{4}\Delta_{k,l}.$$

The error between $\hat{y}_{k,l}$ and $\hat{y}^{*}_{k,l}$, for $k \in \{1, 2, \dots, K-1\}$ and $l \in \{1, 2, \dots, L-1\}$, evolves as:

$$\hat{e}^{+}_{k,l} = \frac{1}{4}(\hat{e}_{k+1,l} + \hat{e}_{k-1,l} + \hat{e}_{k,l+1} + \hat{e}_{k,l-1}), \tag{18}$$

where $e_{0,l} = e_{K,l} = e_{k,0} = e_{k,L} = 0$. Equation (18) is a Discrete Poisson Equation with homogeneous Dirichlet boundary conditions. From [9] (Chapter 4, Section 3.3), a Discrete Poisson Equation with homogeneous Dirichlet boundary conditions has eigenfunctions of the form:

$$\hat{e}_{k,l} = \sin\left(\frac{\pi\mu k}{K}\right)\sin\left(\frac{\pi\nu l}{L}\right), \tag{19}$$

where $\mu \in \{1, 2, \ldots, K-1\}$ and $\nu \in \{1, 2, \ldots, L-1\}$. Eigenvalues for (18) can be found by substituting (19):

$$\lambda \hat{e}_{k,l} = \frac{1}{4}(\hat{e}_{k+1,l} + \hat{e}_{k+1,l} + \hat{e}_{k,l+1} + \hat{e}_{k,l-1}),$$

as such:

$$\lambda \left[\sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) \right] = \frac{1}{4}\Bigg[\sin\left(\frac{\pi\mu(k+1)}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) + \sin\left(\frac{\pi\mu(k-1)}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) + \sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu(l+1)}{L}\right) + \sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu(l-1)}{L}\right) \Bigg]. \tag{20}$$

Making use of the trigonometric identity, $\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b)$, (20) reduces to:

$$\lambda \left[\sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) \right] = \frac{1}{2}\Bigg[\sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) \cos\left(\frac{\pi\mu}{K}\right) + \sin\left(\frac{\pi\mu k}{K}\right) \sin\left(\frac{\pi\nu l}{L}\right) \cos\left(\frac{\pi\nu}{L}\right) \Bigg],$$

which simplifies further to:

$$\lambda = \frac{1}{2}\left[\cos\left(\frac{\pi\mu}{K}\right) + \cos\left(\frac{\pi\nu}{L}\right) \right],$$

which is largest in magnitude when $\mu = 1$ and $\nu = 1$. Hence, the spectral radius of linear system (18) is

$$\lambda_{max} = \frac{1}{2}\left[\cos\left(\frac{\pi}{K}\right) + \cos\left(\frac{\pi}{L}\right) \right].$$

■

D. Single Step Convergence

Theorem 5: [Convergence in One Iteration]
If C is chosen to be a normal basis, then the JBCSE algorithm converges in a single iteration.

Proof: When C is chosen to be a normal basis of the cycle space, the cycle Laplacian matrix, $\mathcal{L}_c$, is a block diagonal matrix, with blocks of size d, the dimension of the measurements. Consequentially, all elements of cycle adjacency matrix, A, and iteration matrix, J, are equal to zero. Hence, the cycle iteration variables, y_i, are set to the cycle discrepancy, Δ_i, on the first iteration. ■

In general, when C is chosen to be a normal basis for the cycle space, the cycle Laplacian matrix, $\mathcal{L}_c$, will be fairly dense, indicating that cycle leaders performing JBCSE will need information from across the graph. While this is not a very distributed algorithm, iteration on this basis may render solutions with less communication than the classic centralized solution (one in which all measurements are passed to a central processor). This is because the cycle leaders only need to know the cycle discrepancy (a weighted sum of measurements as opposed to the measurements themselves).

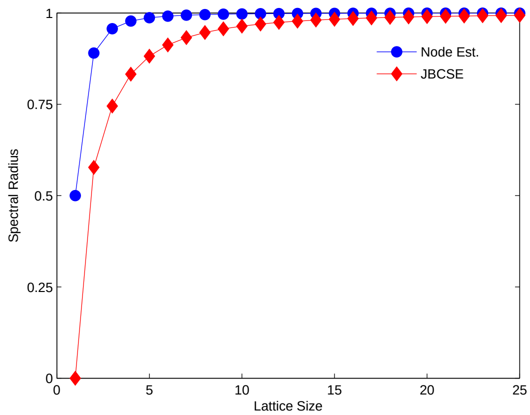
For example, consider a ring graph with n nodes. Computing the sum around the ring takes n messages. However, to pass all of the messages to one node for processing would take $\frac{n^2-1}{4}$ messages (for an odd number of nodes). This can be seen by considering a graph with an odd number of nodes. The furthest distance a measurement must travel is $\frac{n-1}{2}$ (there are two such measurements). The next furthest distance is $\frac{n-1}{2} - 1$, etc. Two times the sum from 1 to $\frac{n-1}{2}$ is $\frac{n^2-1}{4}$.

VI. Simulations

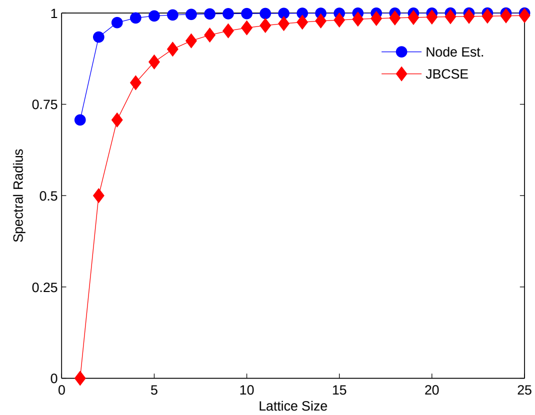
The chief aim of this work is to develop a distributed estimation algorithm that swiftly converges to the BLUE. As such, it is appropriate to offer a comparison with previously developed algorithms. The Jacobi Algorithm introduced in [2] is a node estimation method that uses relative state difference measurements to compute the optimal estimate of a graph's state. Figure 3 compares the spectral radii of this node estimation method and the JBCSE algorithm (where the cycle space is chosen to be the faces of the graph). For triangular, square, and hexagonal lattices, the JBCSE always has a smaller spectral radius and therefore has a faster convergence rate.

Comparing the spectral radii for these two algorithms only compares the slowest modes of the algorithms. To provide a valid comparison of the two algorithms, one must consider the additional overhead of initializing the algorithms and the fact that the iterations will not necessarily occur on the slowest possible modes. While it is not practical to compare performance on every type of graph, this section considers a wide range of regular graphs that include triangular, square, and hexagonal lattices. The simulation results demonstrate the performance gains possible when using JBCSE with respect to the Jacobi Algorithm with flagged initialization (JAWFI) introduced in [2].

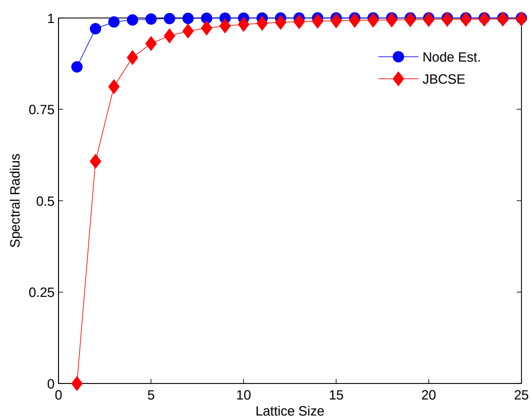
The simulations were run on lattices of various size. The nodes in each lattice are one unit away from their edge neighbors. The goal of each node is to estimate its (two dimensional) position from measurements that are corrupted by zero mean Gaussian noise with standard deviation 0.25. For each set of measurements, both algorithms were run until the 2-norm of the estimation error normalized by the number of nodes, $\frac{\|\hat{x}^* - \hat{x}\|}{n}$, was below 0.025 (one tenth of the standard deviation of the measurement noise). This process was repeated on two thousand sets of measurements for each size of each lattice



(a) Triangular Lattice



(b) Square Lattice



(c) Hexagonal Lattice

Fig. 3. The spectral radii of the Jacobi iteration's state transition matrices for various sized (a) triangular, (b) square, and (c) hexagonal lattices.

graph. The size of a lattice was taken to be the number of tessellations on one side of a given lattice. Figure 4 shows the first three sizes for each type of lattice.

Figure 5(a) depicts the ratio of the average number of iterations needed by the JBCSE over the average number of iterations needed by for the JAWFI method, as a function of the lattice size. Significant gains in terms of the number of iterations can be seen for large graphs as predicted from the analysis of the convergence rates (see Figure 3). Figure 5(b) charts the ratio of the average number of messages passed by the JBCSE algorithm over the average number of messages passed by the JAWFI method, as a function of lattice size. For the JBCSE algorithm, we counted all messages passed during computation of cycle discrepancies, during the flagged initialization process, between cycle leaders during the iteration

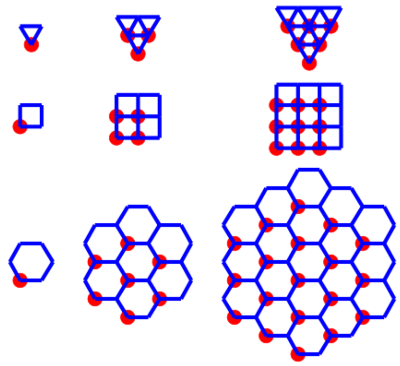
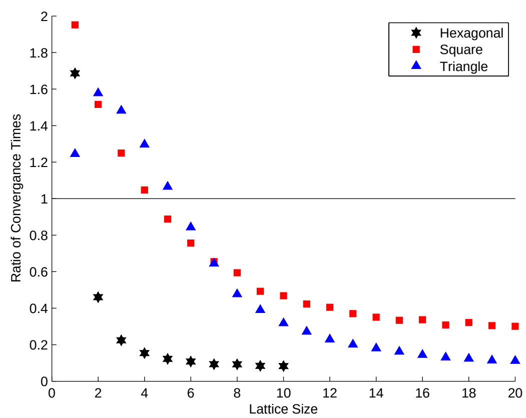
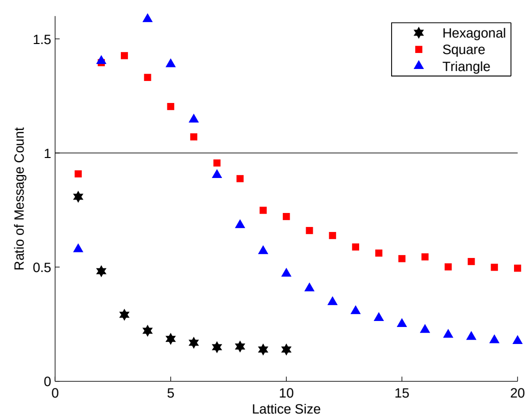


Fig. 4. Size one, two, and three lattices with cycle leaders highlighted.



(a) Iteration Count

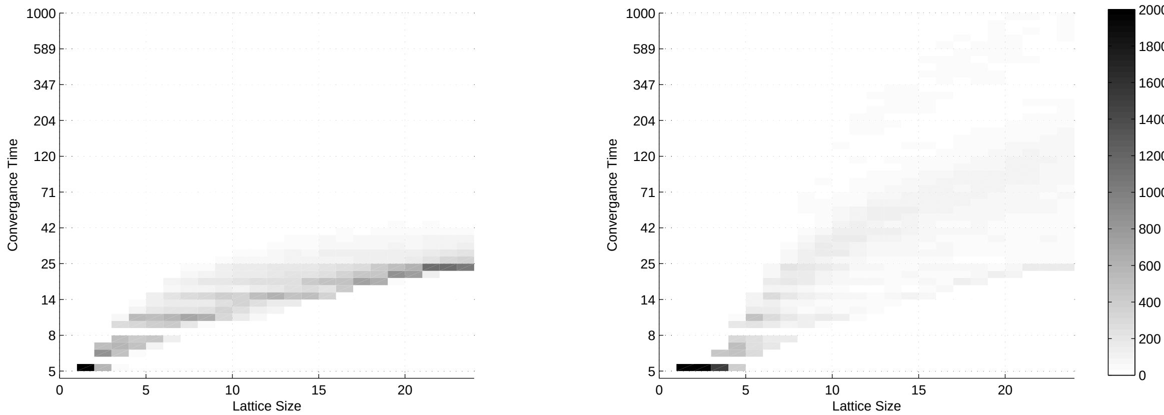


(b) Message Count

Fig. 5. (a) The average number of iterations for JBCSE to finish over the average number of iterations for JAWFI to finish. (b) The average number of messages passed during JBCSE over the average number of messages passed during JAWFI.

process, and from cycle leaders to other nodes for state updates. In both of these figures, the horizontal axis corresponds to the size of the lattice. For smaller lattices, the overhead of obtaining the cycle discrepancies makes JBCSE less advantageous, but for larger graphs, JBCSE shows faster convergence with less message passing. For both plots in Figure 5, the results for hexagonal lattices stop at size ten due to the lengthy JAWFI convergence times.

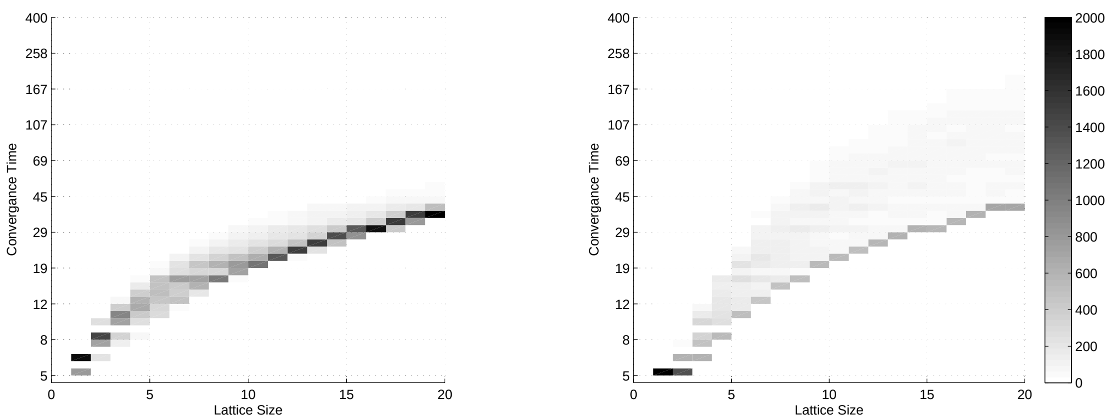
Figures 6, 7, and 8 are density plots of the number of iterations for each algorithm to reach the completion criteria on triangular, square, and hexagonal lattices, respectively. Due to skewness in the histograms, these figures are plotted using a log-linear scaling. These histograms reveal another benefit of the JBCSE algorithm; the settling times for the JBCSE are fairly consistent.



(a) JBCSE Algorithm

(b) JAWFI Algorithm

Fig. 6. Number of iterations to meet the completion criteria on a triangular lattice for the (a) JBCSE algorithm and the (b) JAWFI algorithm from [2].



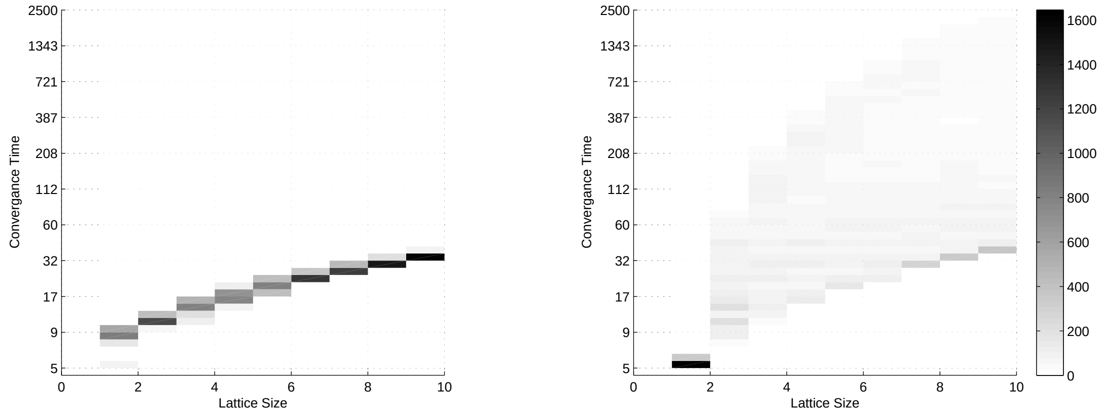
(a) JBCSE Algorithm

(b) JAWFI Algorithm

Fig. 7. Number of iterations to meet the completion criteria on a square lattice for the (a) JBCSE algorithm and the (b) JAWFI algorithm.

VII. Conclusion

This paper introduced cycle space estimation from relative measurements and showed its equivalence to BLUE. We explored a Jacobi-based distributed algorithm that was shown to converge to the BLUE (under appropriate conditions). Conditions for convergence of the distributed algorithm were given, as well as closed form convergence rates for the algorithm on a grid network. We showed that cycle space estimation reduces communication and convergence time for a large class of graphs.



(a) JBCSE Algorithm

(b) JAWFI Algorithm

Fig. 8. Number of iterations to meet the completion criteria on a hexagonal lattice for the (a) JBCSE algorithm and the (b) JAWFI algorithm.

To make the methods in this paper more attractive, a few items need to be explored further. The inverse relationship between distributability and convergence rate leads to an interesting tuning problem; the JBCSE algorithm can be tuned for faster convergence by choosing a set of cycles that is less local. It is also interesting to look at how the selection of the cycle leader nodes in the graph affects performance of distributed cycle space algorithms; cycle leaders that are closer to each other (in the communication graph) can share information faster.

VIII. Appendix

Lemma 1: Given matrices α and β where α and the transpose of β, β^T, are full column rank, the nullspace of β is the image of α if and only if

$$\alpha(\alpha^T\alpha)^{-1}\alpha^T = I - \beta^T(\beta\beta^T)^{-1}\beta.$$

Proof:

Denote the orthogonal complement of a subspace by $\perp$,

$$\text{image}(\alpha) = \text{null}(\beta) = \text{image}(\beta^T)^{\perp}.$$

Replacing α and β^T with their respective projection matrices,

$$\text{image}(\alpha(\alpha^T\alpha)^{-1}\alpha^T) = \text{image}(\beta^T(\beta\beta^T)^{-1}\beta)^{\perp}.$$

As demonstrated in [28], a matrix projection, $\mathcal{P}$, is an orthogonal complement to $I - \mathcal{P}$, implying

$$\text{image}(\alpha(\alpha^T\alpha)^{-1}\alpha^T) = \text{image}(I - \beta^T(\beta\beta^T)^{-1}\beta).$$

Because the projection matrices associated with two complementary subspaces are unique [18],

$$\alpha(\alpha^T\alpha)^{-1}\alpha^T = I - \beta^T(\beta\beta^T)^{-1}\beta.$$

■

ACKNOWLEDGEMENTS

Thank you to Saul Teukolsky for personal correspondence outlining the proof of Theorem 4.

REFERENCES

[1] P. Alriksson and A. Rantzer. Distributed Kalman filtering using weighted averaging. In *Proceedings of the 17th International Symposium on Mathematical Theory of Networks and Systems*, 2006.

[2] P. Barooah, N.M. da Silva, and J.P. Hespanha. Distributed optimal estimation from relative measurements for localization and time synchronization. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems*, 2006.

[3] P. Barooah, W.J. Russell, and J.P. Hespanha. Approximate distributed Kalman filtering for cooperative multi-agent localization. In *Proceedings of the International Conference on Distributed Computing in Sensor Systems*, 2010.

[4] B. Bell and G. Pillonetto. A distributed Kalman filter. In *Proceedings of the 1st IFAC Workshop on Estimation and Control of Networked Systems*, 2009.

[5] R. Carli, A Chiuso, L. Schenato, and S. Zampieri. Distributed Kalman filtering using consensus strategies. In *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007.

[6] F.S. Cattivelli, C.G. Lopes, and A.H. Sayed. Diffusion strategies for distributed Kalman filtering: Formulation and performance analysis. In *Proceedings of the IARP Workshop on Cognitive Information Processing*, 2008.

[7] F.S. Cattivelli and A.H. Sayed. Diffusion mechanisms for fixed-point distributed Kalman smoothing. In *Proceedings of the 16th European Signal Processing Conference*, 2008.

[8] H. Chen, T. Kirubarajan, and Y. Bar-Shalom. Performance limits of track-to-track fusion versus centralized estimation: Theory and application. *IEEE Transactions on Aerospace and Electronic Systems*, 39(2):386–400, 2003.

[9] D. Dubin. *Numerical and Analytical Methods for Scientists and Engineers Using Mathematica*. Wiley-Interscience, Hoboken, NJ, 2003.

[10] D. Estrin, L. Girod, G. Pottie, and M. Srivastava. Instrumenting the world with wireless networks. In *International Conference on Acoustics, Speech, and Signal Processing*, 2001.

[11] J.A. Fax and R.M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[12] F. Gustafsson and F. Gunnarsson. Positioning using time-difference of arrival measurements. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, 2003.

[13] R.A. Horn and Johnson C.R. *Matrix Analysis*. Cambridge University Press, New York, NY, 1985.

[14] A. Jadbabaie, J. Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, 2003.

[15] R. Johnsonbaugh. *Discrete Mathematics*. Pearson Prentice Hall, Upper Saddle River, NJ, 2009.

[16] S. Mac Lane. A structural characterization of planar combinatorial graphs. *Duke Mathematical Journal*, 3:460–472, 1937.

[17] G. Mao, B. Fidan, and B. Anderson. Wireless sensor network localization techniques. *Computer Networks*, 51(10):2529–2553, 2007.

[18] C.D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, Philadelphia, PA, 2001.

[19] N. Okello and S. Challa. Joint sensor registration and track-to-track fusion for distributed trackers. *IEEE Transactions on Aerospace and Electronic Systems*, 40(3):808–823, 2004.

[20] R. Olfati-Saber. Distributed Kalman filtering in sensor networks. In *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007.

[21] J.M. Ortega. *Numerical Analysis A Second Course*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 1990.

[22] N. Patwari, A.O. Hero III, J. Ash, R.L. Moses, S. Kyperountas, and N.S. Correal. Locating the nodes: Cooperative localization in wireless sensor networks. *IEEE Signal Processing Magazine*, 22(4):54–69, 2005.

[23] R.T. Rockafellar. *Network Flows and Monotropic Optimization*. Athena Scientific, Nashua, NH, 1998.

[24] S.I. Roumeliotis and G.A. Bekey. Collective localization: A distributed Kalman filter approach to localization of mobile robots. In *Proceedings of the International Conference on Robotics and Automation*, 2000.

[25] W.J. Russell, D.J. Klein, and J.P. Hespanha. Optimal estimation on the graph cycle space. In *Proceedings of the American Control Conference*, 2010.

[26] D.P. Spanos and R.M. Murray. Distributed sensor fusion using dynamic consensus. In *IFAC World Congress*, 2005.

[27] D.P. Spanos, R. Olfati-Saber, and R.M. Murray. Approximate distributed Kalman filtering in sensor networks with quantifiable performance. In *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, 2005.

[28] E.S. Suhubi. *Functional Analysis*. Kluwer Academic Publishers, Norwell, MA, 1990.

[29] L. Xiao, S. Boyd, and S. Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33–46, 2007.

[30] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *Proceedings of the 4th International Conference on Information Processing in Sensor Networks*, 2005.