52nd IEEE Conference on Decision and Control
December 10-13, 2013. Florence, Italy

Distributed Weighted Least Squares Estimation with Fast Convergence in Large-scale Systems

Damin Marelli[1] and Minyue Fu[1,2]

[1] *School of Electrical Engineering and Computer Science, University of Newcastle, Australia.*

[2] *Department of Control Science and Engineering, Zhejiang University, China.*

Abstract— **We propose a distributed method for weighted least squares estimation. Our method is suitable for large-scale systems, in which each node only estimates a subset of the unknown parameters. As opposed to other works, our goal is to maximize the convergence speed of the distributed algorithm. To this end, we propose a distributed method for estimating the optimal value of certain scaling parameter on which this speed depends. To further speed the convergence, we use a simple preconditioning method, and we bound the difference between the resulting speed, and the fastest theoretically achievable using preconditioning. We include numerical experiments to illustrate the performance of the proposed method.**

I. Introduction

A sensor network is a web of a large number of intelligent sensing and computing devices connected via a communication network [1]. The emergence of sensor networks calls for the development of distributed algorithms for a number of tasks for which only centralized methods are available. These algorithms carry out the desired task by executing a cooperative strategy over all the nodes of the network. In particular, the development of distributed algorithms for parameter estimation have recently attracted a great deal of attention [2]–[9]. They find applications in environmental and weather monitoring, industrial process monitoring and control, surveillance, smart grid state estimation, etc.

A division of distributed estimation algorithms is between static and dynamic ones. In static estimation, a set of parameters is estimated using the measurements of all nodes, which collectively form a snapshot at a given time [2], [4]. On the other hand, in dynamic estimation the nodes track the evolution of a set of parameters for which a dynamic model is available [6]–[9]. Some "hybrid" methods exist, which permit tracking a time-varying sequence of parameters, without a dynamic model, by somehow updating a static estimation strategy at each time [3], [5]. Another division of distributed estimation methods is between small-scale and large-scale methods. In small-scale methods, all nodes estimate a common set of parameters. In large-scale methods, each node can only reconstruct some part of the whole parameter vector, i.e., knowledge of these parameters is itself distributed [7], [10]. Large-scale estimation is in general more challenging than its small-scale counterpart.

In this paper we study static estimation of large-scale systems. We derive a method which asymptotically achieves the global weighted least squares (WLS) estimate, as the number of time-steps tends to infinity. As opposed to previous works, we address the problem of maximizing the convergence rate. Since our method is based on Richardson's method for solving systems of linear equations [11], its convergence rate depends on certain scaling parameter and a preconditioning matrix. Choosing the optimum scaling parameter requires knowledge of the largest and the smallest eigenvalues of certain positive definite matrix (the estimation error covariance). A distributed algorithm for estimating these values can be obtained using the power method [11]. However, to prevent numerical instability, this approach requires periodically executing a normalization step, which needs to be carried out in a distributed manner. In [12] this is done using average consensus. A drawback of this approach is that consensus itself converges asymptotically. This significantly slows down the convergence of the eigenvalue estimation. To avoid this, we propose a different method in which normalization is done locally, at each node, without inter-node communication. In this way, the optimum scaling parameter can be distributively obtained. Our last problem is to design the preconditioning matrix. Our distributed scenario constrains us to use a block diagonal matrix. Choosing the optimal matrix under this constraint, and using only distributed processing, is a very challenging problem for which we are not able to provide a solution. Nevertheless, we are able to bound the difference between convergence rate achieved using this optimal matrix, and the one resulting using a simple matrix design. This bound turns out to have a simple expression which depends on the network connectivity.

The rest of paper is organized as follows. In Section II we describe distributed WLS estimation the problem. In Section III we derive a distributed method which converges asymptotically. In Section IV we describe a distributed method for finding the value of the scaling parameter which yields the fastest convergence rate. In Section V we describe a sub-optimal choice for preconditioning matrix, and we bound its sub-optimality. Numerical experiments are presented in Section VI.

II. Problem description

Consider a network formed by I nodes. For each $i = 1, \cdots, I$, Node i has an associated parameter vector $x_i \in \mathbb{R}^{d_i}$, and measures the vector $y_i \in \mathbb{R}^{p_i}$, which is given by

$$y_i = \sum_{j=1}^{I} A_{i,j} x_j + v_i, \tag{1}$$

978-1-4673-5717-3/13/$31.00 ©2013 IEEE

where $v_i \sim \mathcal{N}(0, R_i)$ denotes the measurement noise. The noises v_i and v_j are independent, whenever $i \neq j$. Let $\mathcal{I}_i = \{j : A_{j,i} \neq 0\}$ denote the set of nodes whose measurements involve the parameters of Node i, and $\mathcal{O}_i = \{j : A_{i,j} \neq 0\}$ denote the set of nodes whose parameters are involved in the measurements of Node i. We call $\mathcal{N}_i = \mathcal{I}_i \cup \mathcal{O}_i$ the set of *neighbors* of Node i. We do the following assumption:

Assumption 1: For each $i = 1, \cdots, I$, the nodes in $\mathcal{N}_i$ can send/receive information to/from Node i. Also, $A_{i,j}$, for all $j \in \mathcal{O}_i$, and R_i are available at Node i.

Let $x = \left[x_1^T, \cdots, x_I^T\right]$, $y = \left[y_1^T, \cdots, y_I^T\right]$, $v = \left[v_1^T, \cdots, v_I^T\right]$, $A = [A_{i,j}]_{i,j=1,\cdots,I}$ and $R = \operatorname{diag}\{R_1, \cdots, R_I\}$. Then, we can write the measurement model of the whole network as

$$y = Ax + v, \tag{2}$$

with $v \sim \mathcal{N}(0, R)$. The WLS estimation $\hat{x}$ of x is given by

$$\hat{x} \;=\; \Psi^{-1}\alpha \tag{3}$$

with

$$\alpha = A^T R^{-1} y \qquad \text{and} \qquad \Psi = A^T R^{-1} A.$$

For the WLS problem to be well defined, we do the following further assumption:

Assumption 2: The matrix A has full column rank and R is non-singular.

Computing (3) requires global network information. Our goal is to derive distributed methods in which Node i computes the block components $\hat{x}_i$ of the estimate $\hat{x}$, corresponding to x_i, using only information received from its neighbors. In Section III, we derive an iterative method in which this is asymptotically achieved on the limit when the number of iterations tends to infinity.

III. Distributed algorithm

Let Π be a positive definite matrix. Let also $\Upsilon = \Pi^{1/2}\Psi\Pi^{1/2}$ and

$$0 < \gamma < \frac{2}{\|\Upsilon\|}. \tag{4}$$

Let $\tilde{\alpha} = (\gamma\Pi)^{1/2}\alpha$ and $\tilde{\hat{x}} = (\gamma\Pi)^{-1/2}\hat{x}$. From (3) we have

$$\tilde{\hat{x}} \;=\; (\gamma\Upsilon)^{-1}\tilde{\alpha}.$$

From (4), $0 < \gamma\Upsilon < 2I$. Hence, $-I < \gamma\Upsilon - I < I$ and therefore $\|I - \gamma\Upsilon\| < 1$. Then, using Richardson's method [11], we have that $\tilde{\hat{x}}$ can be recursively computed by

$$\tilde{\hat{x}}(t+1) = (I - \gamma\Upsilon)\tilde{\hat{x}}(t) + \tilde{\alpha}. \tag{5}$$

Then, it is straightforward to obtain

$$\hat{x}(t+1) \;=\; (I - \gamma\Pi\Psi)\hat{x}(t) + \gamma\Pi\alpha. \tag{6}$$

As it will be explained in Section V, the preconditioning matrix Π is used to increase the convergence speed of the recursions 6. Let $[\Psi\hat{x}(t)]_i$ denote the i-th block entry of the vector $\Psi\hat{x}(t)$. We have,

$$[\Pi\Psi\hat{x}(t)]_i = \sum_{j=1}^{I} \Pi_{i,j}\,[\Psi\hat{x}(t)]_j\,, \tag{7}$$

where $\Pi_{i,j}$ are the block entries of Π, i.e., $\Pi = [\Pi_{i,j}]_{i,j=1,\cdots,I}$. Suppose that $[\Psi\hat{x}(t)]_i$ is available at Node i. Equation (7) means that, unless Π is block diagonal, the nodes have to exchange information for the purpose of preconditioning. This goes against the goal for preconditioning. Thus, we constrain our analysis to

$$\Pi = \operatorname{diag}\{\Pi_1, \cdots, \Pi_I\}. \tag{8}$$

We use d_i to denote dimension of each square matrix Π_i, $i = 1, \cdots, I$.

Let

$$\alpha_i \;=\; \sum_{k \in \mathcal{I}_i} \alpha_i^{(k)}, \tag{9}$$

with $\alpha_i^{(k)} = A_{k,i}^T R_k^{-1} y_k$, for $k = 1, \cdots, I$. Also, for $i, j = 1, \cdots, I$, let

$$\Psi_{i,j} \;=\; \sum_{k : i,j \in \mathcal{O}_k} \Psi_{i,j}^{(k)}, \tag{10}$$

where $\Psi_{i,j}^{(k)} = A_{k,i}^T R_k^{-1} A_{k,j}$, for all $k = 1, \cdots, I$. Hence, $\alpha = \left[\alpha_1^T, \cdots, \alpha_I^T\right]^T$ and $\Psi = [\Psi_{i,j}]_{i,j=1,\cdots,I}$. We have

$$\begin{aligned} [\Psi\hat{x}(t)]_i &= \sum_{j=1}^{I} \Psi_{i,j}\hat{x}_j(t) \\ &= \sum_{k \in \mathcal{I}_i} \sum_{j \in \mathcal{O}_k} \Psi_{i,j}^{(k)}\hat{x}_j(t). \end{aligned} \tag{11}$$

Then, from (6), (11) and (9), we obtain

$$\begin{aligned} \hat{x}_i(t+1) &= \hat{x}_i(t) - \gamma\Pi_i \sum_{j=1}^{I} \Psi_{i,j}\hat{x}_j(t) + \gamma\Pi_i\alpha_i \\ &= \hat{x}_i(t) - \gamma\Pi_i \left(\sum_{k \in \mathcal{I}_i} \sum_{j \in \mathcal{O}_k} \Psi_{i,j}^{(k)}\hat{x}_j(t) - \sum_{k \in \mathcal{I}_i} \alpha_i^{(k)} \right). \end{aligned} \tag{12}$$

Notice that the matrices $\Psi_{i,j}^{(k)}$ are only available at Node k. Hence, the natural role of Node k is that of an intermediary between Node j, sending $\hat{x}_j(t)$, and Node i, receiving $\sum_{j \in \mathcal{O}_k} \Psi_{i,j}^{(k)}\hat{x}_j(t)$. Considering this, we obtain the following algorithm:

Algorithm 1 - distributed WLS estimation:

Initialization:

1) For each $k = 1, \cdots, I$ and $i \in \mathcal{O}_k$, Node k computes $\alpha_i^{(k)}$ and sends it to Node i.
2) On reception, Node i computes $\alpha_i = \sum_{k \in \mathcal{I}_i} \alpha_i^{(k)}$.
3) For each $i = 1, \cdots, I$, Node i sets $\hat{x}_i(1) = 0$.

Main loop: At time $t \in \mathbb{N}$:

1) For each $j = 1, \cdots, I$ and $k \in \mathcal{N}_j$, Node j sends its current estimate $\hat{x}_j(t)$ to Node k.[1]

[1] Notice that, according to (12), Node j should transmit $\hat{x}_j(t)$ to all nodes k such that $j \in \mathcal{O}_k$, or equivalently, to all nodes in $\mathcal{I}_j$. However, Node j does not know which nodes are in $\mathcal{I}_j$. Hence it transmits to all nodes in $\mathcal{N}_j$, and it is up to the receiving Node k to detect Node j is in $\mathcal{O}_k$.

2) On reception, for each $k = 1, \cdots, I$ and $i \in \mathcal{O}_k$, Node k sends to Node i

$$\check{x}_{i,k}(t) = \sum_{j \in \mathcal{O}_k} \Psi_{i,j}^{(k)} \hat{x}_j(t).$$

3) On reception, for each $i = 1, \cdots, I$, Node i computes

$$\hat{x}_i(t+1) = \hat{x}_i(t) - \gamma \Pi_i \left(\sum_{k \in \mathcal{I}_i} \check{x}_{i,k}(t) - \alpha_i \right).$$

To implement Algorithm 1, we need to design the rescaling factor γ and the preconditioning matrices Π_i, for all $i = 1, \cdots, I$. This is done in Sections IV and V, respectively.

IV. Design of the rescaling factor γ

In view of (5), the value of γ that maximizes the convergence rate is

$$\gamma = \frac{2}{\|\Upsilon\| + \|\Upsilon^{-1}\|^{-1}}, \tag{13}$$

since this is the value that minimizes $\|I - \gamma\Upsilon\|$. In this section we derive a distributed method to find the value of γ given in (13). To this end, we need distributed methods for finding $\|\Upsilon\|$ and $\left\|\Upsilon^{-1}\right\|^{-1}$. We give these methods below. These methods yield, at Node i and time step t, estimates $\overline{\Upsilon}_i(t)$ and $\underline{\Upsilon}_i(t)$, of $\|\Upsilon\|$ and $\left\|\Upsilon^{-1}\right\|^{-1}$, respectively. Then, at the same node and time step, the estimate $\gamma_i(t)$ of γ is obtained by

$$\gamma_i(t) = \frac{2}{\overline{\Upsilon}_i(t) + \underline{\Upsilon}_i(t)}.$$

A. Distributed method for finding $\|\Upsilon\|$

Choose any vector $b(0) \neq 0$ and let $b(t) = \Upsilon^t b(0)$. Using (11), at Node i, we have

$$\begin{aligned} b_i(t+1) &= \Pi_i^{1/2} \sum_{j=1}^{I} \Psi_{i,j} \Pi_j^{1/2} b_j(t) \\ &= \Pi_i^{1/2} \sum_{k \in \mathcal{I}_i} \sum_{j \in \mathcal{O}_k} \Psi_{i,j}^{(k)} \Pi_j^{1/2} b_j(t), \end{aligned} \tag{14}$$

where $b_i(t)$ denotes the i-th block component of $b(t)$. Then, using the power method [11], Node i can asymptotically compute $\|\Upsilon\|$ as follows

$$\|\Upsilon\| = \lim_{t\to\infty} \frac{\|b_i(t)\|}{\|b_i(t-1)\|}. \tag{15}$$

An inconvenience with the approach above is that $b(t)$ either increases or decreases indefinitely. To avoid this, the vector $b(t)$ can be periodically normalized. In [12], this was done using average consensus (in the continuous-time case). As we mentioned in Section I, we avoid the drawbacks of that method by providing an alternative approach in which $b(t)$ is normalized at each node, without inter-node communication.

Fix $t \in \mathbb{N}$. Let $k_i(t) \in \mathbb{R}$ and

$$\bar{b}_i(t) = k_i(t) b_i(t). \tag{16}$$

From (14), we have

$$\bar{b}_i(t+1) = \frac{k_i(t+1)}{k_i(t)} \Pi_i^{1/2} \sum_{k \in \mathcal{I}_i} \sum_{j \in \mathcal{O}_k} \frac{k_i(t)}{k_j(t)} \Psi_{i,j}^{(k)} \Pi_j^{1/2} \bar{b}_j(t).$$

Let $k_i(0) = 1$ and

$$\varsigma_i(t) = \frac{k_i(t)}{k_i(t-1)},$$

so that $k_i(t) = \prod_{\tau=1}^{t} \varsigma_i(\tau)$. Then,

$$\bar{b}_i(t+1) = \varsigma_i(t+1) \tilde{b}_i(t+1), \tag{17}$$

with

$$\begin{aligned} \tilde{b}_i(t+1) &= \Pi_i^{1/2} \sum_{k \in \mathcal{I}_i} \sum_{j \in \mathcal{O}_k} \upsilon_{i,j}(t) \Psi_{i,j}^{(k)} \Pi_j^{1/2} \bar{b}_j(t), \\ \upsilon_{i,j}(t) &= \prod_{\tau=1}^{t} \frac{\varsigma_i(\tau)}{\varsigma_j(\tau)}. \end{aligned}$$

We need to design $k_i(t+1)$, or equivalently $\varsigma_i(t+1)$, to avoid the indefinite increase or decrease of $b(t)$. In principle, this could be achieved by choosing

$$k_i(t) = \|b_i(t)\|^{-1},$$

so that $\left\|\bar{b}_i(t)\right\| = 1$, for all $t \in \mathbb{N}$. From (17), this would lead to

$$\varsigma_i(t+1) = \left\|\tilde{b}_i(t+1)\right\|^{-1}.$$

However, the question then arises as to whether some of the scalars $\upsilon_{i,j}(t)$ would grow to infinity. Notice that

$$\upsilon_{i,j}(t) = \frac{k_i(t)}{k_j(t)}.$$

Hence, this could only happen if some vector in the eigenspace associated to the largest eigenvalue of Υ has zero components in the entries corresponding to $b_j(t)$. We call a matrix satisfying this property, *ill-posed*. Although the set of ill-posed matrices is nowhere dense, (i.e., it is unlikely to have an ill-posed matrix Υ), we can avoid the indefinite growth of $\upsilon_{i,j}(t)$ by choosing $\varsigma_i(t+1)$ so that $\left\|\bar{b}_i(t+1)\right\| \leq 1$ and, for all $j \in \mathcal{B}_i = \{j : \Psi_{i,j} \neq 0\}$,

$$\varsigma_i(t+1)\upsilon_{i,j}(t) \leq 1.$$

This leads to

$$\varsigma_i(t+1) = \max\left\{\left\|\tilde{b}_i(t+1)\right\|, \upsilon_{i,j}(t), j \in \mathcal{B}_i\right\}^{-1}.$$

From (15) and (16), the estimate $\bar{\Upsilon}_i(t)$ of $\|\Upsilon\|$ at t is

$$\begin{aligned} \bar{\Upsilon}_i(t) &= \frac{\|b_i(t)\|}{\|b_i(t-1)\|} \\ &= \varsigma_i^{-1}(t) \frac{\left\|\bar{b}_i(t)\right\|}{\left\|\bar{b}_i(t-1)\right\|}. \end{aligned}$$

However, if Υ is ill-posed, $\left\|\bar{b}_i(t)\right\|$ will tend to zero. In such case, $\bar{\Upsilon}_i(t)$ can be computed by

$$\bar{\Upsilon}_i(t) = \varsigma_j^{-1}(t) \frac{\left\|\bar{b}_j(t)\right\|}{\left\|\bar{b}_j(t-1)\right\|},$$

for some neighbor node j for which $\left\|\bar{b}_j(t)\right\|$ does not tend to zero. Notice that such a neighbor always exists, for otherwise Node i would be isolated from all other nodes.

We summarize below the resulting algorithm.

Algorithm 2 - distributed norm estimation: For each $k = 1, \cdots, I$, Node k, chooses $\bar{b}_k(1)$, with $\left\|\bar{b}_k(1)\right\| = 1$ and sets $\varsigma_k(1) = 1$ and $\upsilon_{i,j}^{(k)}(1) = 1$, for all $i, j \in \mathcal{N}_k$. Then, at time $t \in \mathbb{N}$:

1) For each $j = 1, \cdots, I$ and $k \in \mathcal{N}_j$, Node j sends $\left(\Pi_j^{1/2}\bar{b}_j(t), \varsigma_j(t)\right)$ to Node k.
2) On reception, for each $k = 1, \cdots, I$ and $i \in \mathcal{O}_k$, Node k sends $\left(\check{b}_i^{(k)}(t), \bar{\varsigma}_i^{(k)}(t)\right)$ to Node i, where
$$\begin{aligned}\check{b}_i^{(k)}(t) &= \sum_{j\in\mathcal{O}_k} \upsilon_{i,j}^{(k)}(t)\Psi_{i,j}^{(k)}\Pi_j^{1/2}\bar{b}_j(t),\\ \bar{\varsigma}_i^{(k)}(t) &= \max_{j\in\mathcal{N}_k} \upsilon_{i,j}^{(k)}(t),\end{aligned}$$
and
$$\upsilon_{i,j}^{(k)}(t) = \frac{\varsigma_i(t)}{\varsigma_j(t)}\upsilon_{i,j}^{(k)}(t-1).$$
3) On reception, for each $i = 1, \cdots, I$, Node i computes
$$\begin{aligned}\bar{b}_i(t+1) &= \varsigma_i(t+1)\tilde{b}_i(t+1),\\ \varsigma_i(t+1) &= \max\left\{\left\|\tilde{b}_i(t+1)\right\|, \bar{\varsigma}_i^{(k)}(t), k \in \mathcal{I}_i\right\}^{-1},\end{aligned}$$
with
$$\tilde{b}_i(t+1) = \Pi_i^{1/2}\sum_{k\in\mathcal{I}_i}\check{b}_i^{(k)}(t). \tag{18}$$
Also, the estimate $\overline{\Upsilon}_i(t)$ of $\|\Upsilon\|$ is
$$\overline{\Upsilon}_i(t) = \varsigma_i(t+1)^{-1}. \tag{19}$$

B. Distributed method for finding $\left\|\Upsilon^{-1}\right\|^{-1}$

Let $c \geq \|\Upsilon\|$ and $\Phi = cI - \Upsilon$. It follows that
$$\begin{aligned}\left\|\Upsilon^{-1}\right\|^{-1} &= \underline{\mathrm{eig}}\left(\Upsilon\right)\\ &= c - \overline{\mathrm{eig}}\left(\Phi\right)\\ &= c - \|\Phi\|.\end{aligned}$$
Hence, we can find $\left\|\Upsilon^{-1}\right\|^{-1}$ by applying Algorithm 2 on Φ, to find $\|\Phi\|$. To this end, at Node i and time t, we choose $c = \overline{\Upsilon}_i(t)$. The resulting algorithm is similar to Algorithm 2, with (18) replaced by
$$\tilde{b}_i(t+1) = \overline{\Upsilon}_i(t)\bar{b}_i(t) - \Pi_i^{1/2}\sum_{k\in\mathcal{I}_i}\check{b}_{i,k}(t),$$
and (19) replaced by
$$\begin{aligned}\underline{\Upsilon}_i(t) &= \overline{\Upsilon}_i(t) - \overline{\Phi}_i(t),\\ \overline{\Phi}_i(t) &= \varsigma_i(t+1)^{-1}.\end{aligned}$$

V. Design of the preconditioning matrix Π

As mentioned above, for a given choice of Υ, the fastest convergence rate of Algorithm 1 is achieved when γ is chosen as in (13). It is straightforward to verify that, under this choice of γ,
$$\|I - \gamma\Upsilon\| = \frac{\kappa\left(\Upsilon\right) - 1}{\kappa\left(\Upsilon\right) + 1},$$
where $\kappa(\Upsilon) = \|\Upsilon\|\left\|\Upsilon^{-1}\right\|$ denotes the condition number of Υ. Then, from (5), there exists $K \geq 0$, such that
$$\begin{aligned}\|\hat{x} - \hat{x}(t)\| &\leq K\,\|I - \gamma\Upsilon\|^t\\ &= Ke^{t\log\frac{\kappa(\Upsilon)-1}{\kappa(\Upsilon)+1}}.\end{aligned}$$
Then, we define the time constant $\tau(\Upsilon)$ of the distributed WLS algorithm by
$$\tau(\Upsilon) = \frac{1}{\log\frac{\kappa(\Upsilon)+1}{\kappa(\Upsilon)-1}}. \tag{20}$$
Hence, a natural question is whether the preconditioning matrices Π_i, $i = 1, \cdots, I$, can be chosen so that $\tau(\Upsilon)$ is minimized. While we are not able to answer this question, we have the following result, which follows using an argument similar to the one in [13, Th. 2]. Its proof is omitted, and will appear in a journal version.

Theorem 1: If $\Pi_i = \Psi_{i,i}^{-1}$, for all $i = 1, \cdots, I$, then
$$\kappa\left(\Upsilon\right) \leq \beta\kappa_\star,$$
where
$$\begin{aligned}\beta &= \max_i |\mathcal{B}_i|,\\ \kappa_\star &= \min_{\tilde{\Pi}\in\mathcal{P}} \kappa\left(\tilde{\Pi}^{1/2}\Psi\tilde{\Pi}^{1/2}\right),\end{aligned}$$
with $\mathcal{B}_i = \{j : \Psi_{i,j} \neq \emptyset\}$ and $\mathcal{P}$ denoting the set of positive definite block diagonal matrices of the form (8).

Theorem 1 states that if the preconditioning matrices Π_i, $i = 1, \cdots, I$, are chosen as
$$\Pi_i = \left(\sum_{k\in\mathcal{I}_i}\Psi_{i,i}^{(k)}\right)^{-1}, \tag{21}$$
then $\kappa(\Upsilon)$ is at most β times bigger than the smallest possible value $\kappa_\star$ achievable using block diagonal preconditioning matrices. Notice that $\mathcal{B}_i = \{j : \mathcal{I}_i \cap \mathcal{I}_j \neq \emptyset\} \subseteq \{j : \mathcal{N}_i \cap \mathcal{N}_j \neq \emptyset\}$. Hence, β is bounded by the maximum number of two-hop neighbors over the whole network. Hence, it does not necessarily grow with the network size.

Now, we have
$$\lim_{\kappa\to\infty} \kappa\log\left(\frac{\kappa+1}{\kappa-1}\right) = 2.$$
Hence, from Theorem 1, for large $\kappa\left(\Upsilon\right)$ we have
$$\begin{aligned}\tau\left(\Upsilon\right) &\simeq \frac{\kappa\left(\Upsilon\right)}{2}\\ &\leq \frac{\beta}{2}\min_{\tilde{\Pi}} \kappa\left(\tilde{\Pi}^{1/2}\Psi\tilde{\Pi}^{1/2}\right)\\ &\simeq \beta\tau_\star,\end{aligned} \tag{22}$$
where
$$\tau_\star = \min_{\tilde{\Pi}\in\mathcal{P}} \tau\left(\tilde{\Pi}^{1/2}\Psi\tilde{\Pi}^{1/2}\right).$$

Hence, if Π_i, $i = 1, \cdots, I$, are chosen as in (21), and $\kappa\left(\Upsilon\right)$ is large, then the time constant $\tau\left(\Upsilon\right)$ is at most β away from the optimal one $\tau_\star$.

Remark 1: In view of (21), computing $\Pi_i = \Psi_{i,i}^{-1}$ requires the matrices $\Psi_{i,i}^{(k)}$, $k \in \mathcal{I}_i$, to be transmitted from Node k to Node i during an initialization stage.

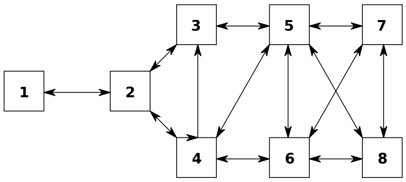

Fig. 2. Network topology induced by the nodes in Figure 1.

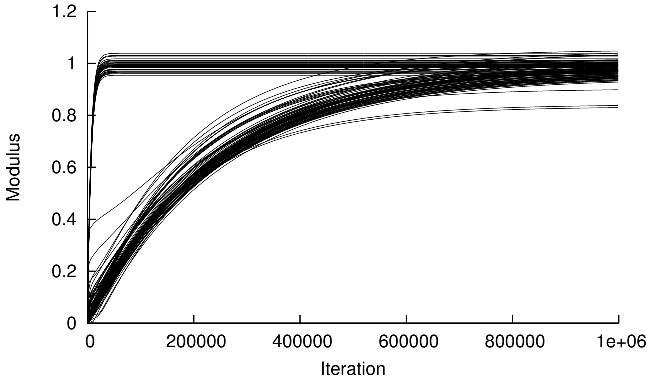

Fig. 3. Convergence of the distributed WLS method, without preconditioning.

VI. Simulations

A. State estimation in power systems

In the first simulation we use the proposed distributed method for state estimation in smart electricity networks involving multi-area interconnected power systems [14]. We use the IEEE 118-bus test system, shown in Figure 1, where buses are represented by circles and lines by edges. Some buses have a phasor measurement unit (PMU) installed. These buses are shown in gray. Each PMU measures the voltage of the bus where it is installed, as well as the currents of the lines attached to that bus. The goal is to estimate the voltage (a complex phasor) at each bus. We place the PMUs using the method in [15]. This guarantees that the matrix A in (2) has full column rank. We also assume that $R = \sigma^2 I$, with $\sigma = 0.05$. This leads to a squared estimation error of -17.45 dB.

We cluster the buses into eight nodes, as shown in Figure 1. From the definition of $\mathcal{N}_i$, it follows that $j \in \mathcal{N}_i$ if there is a bus with a PMU installed, having a neighbor bus (including possibly itself) in each node. Figure 2 shows the topology of the network induced by the clustering shown in Figure 1.

Figure 3 shows the convergence of the asymptotic method without preconditioning. To this end, we show the modulus of the estimated voltage of each bus at each step. We see that the convergence is very slow. The reason for this is that the condition number of Ψ is 478972. The preconditioning matrix in (21) gives a condition number of 700, which leads to a much faster convergence. This is shown in Figure 4. Figure 5 shows the fast convergence of the estimation of $\|\Upsilon\|$ and $\left\|\Upsilon^{-1}\right\|^{-1}$ at each node.

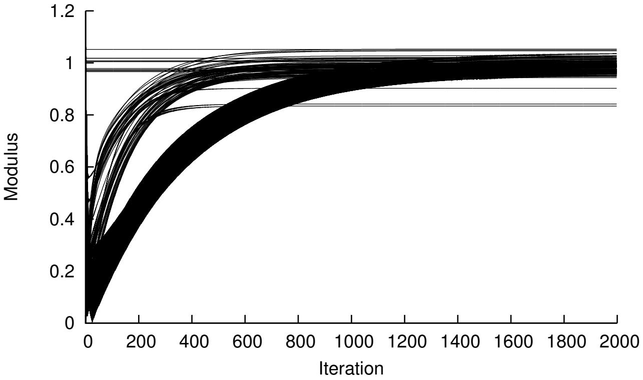

Fig. 4. Convergence of the distributed WLS method, with preconditioning.

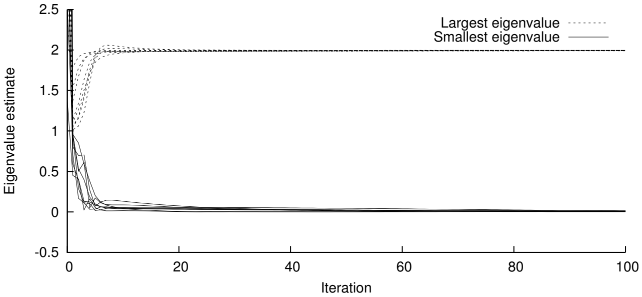

Fig. 5. Convergence of the distributed eigenvalue estimation algorithm.

B. Sensor localization

Sensor localization refers to the problem of obtaining the locations of each node in a network, based on the knowledge of the locations of a few (anchor) nodes a well as the mutual distances between neighbor nodes. A distributed method for carrying out this task is proposed in [16]. In this method, the vector x of unknown node coordinates is obtained by solving the equation $y = Ax$, where y is the vector of coordinates of the anchor nodes and the matrix A is build using inter-node distances. Due to inaccuracy in distance measurements, the formula above can be approximately expressed as in (2). In that case, we can use our proposed distributed method to obtain, at each node, a WLS estimation of its coordinates. The experiment setup is shown in Figure 6. It includes three anchor nodes, as well and $I = 20$ nodes, randomly placed in their convex hull. We use a noise covariance matrix $R = \sigma^2 I_d$, where I_d denotes the identity matrix, and $\sigma^2 = 10^{-4}$. This results in a localization error of

$$e \quad = \quad 10\log_{10}\sum_{i=1}^{I}(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 = -33\text{dB}.$$

The convergence of the coordinate estimates at each node, using the proposed method with the preconditioning matrix in (21), is shown in Figure 7.

References

[1] Waltenegus Dargie and Christian Poellabauer. *Fundamentals of Wireless Sensor Networks: Theory and Practice (Wireless Communications and Mobile Computing)*. Wiley, 1 edition, 8 2010.

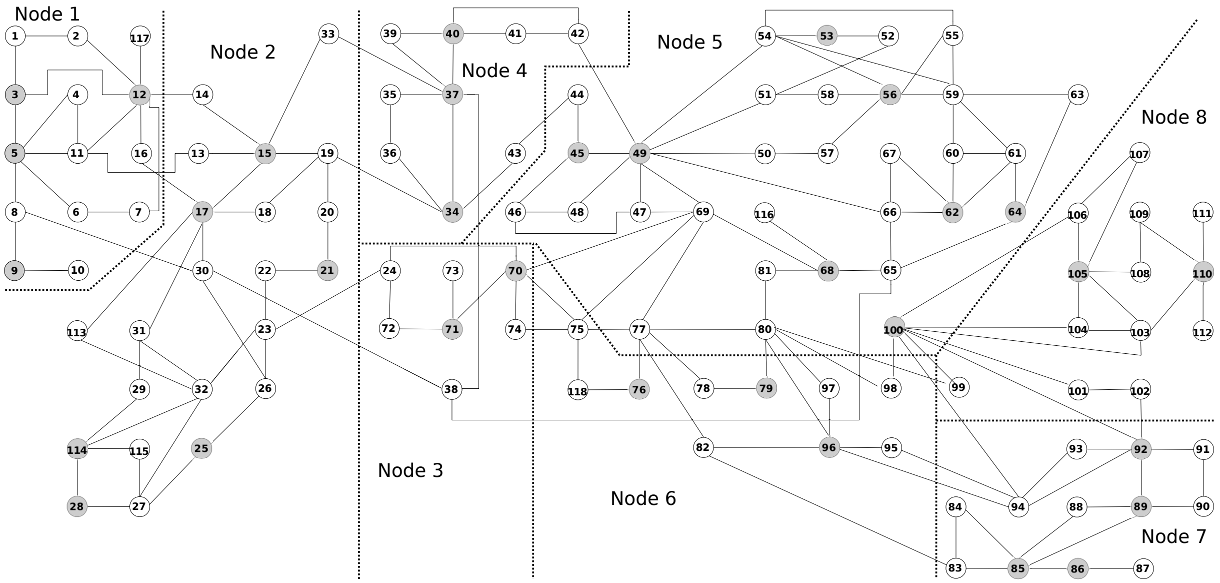

Fig. 1. Diagram of the IEEE 118-bus test system.

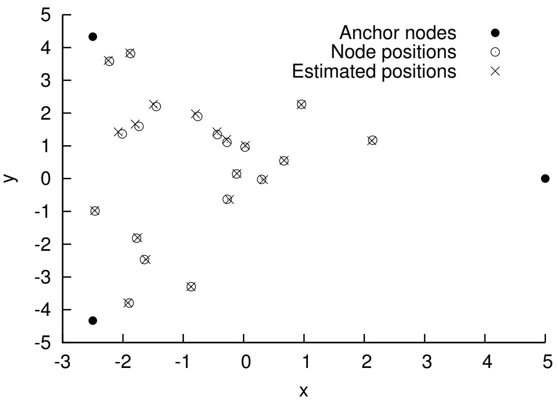

Fig. 6. Node positions and estimates.

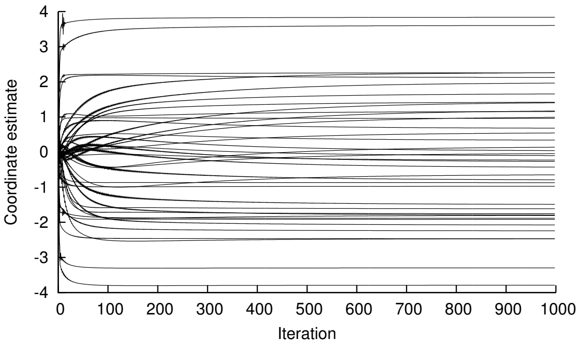

Fig. 7. Convergence of the node coordinate estimates.

[2] J.J. Xiao, A. Ribeiro, Z.Q. Luo, and G.B. Giannakis. Distributed compression-estimation using wireless sensor networks. *IEEE Signal Processing Magazine*, 23(4):27–41, 2006.

[3] C.G. Lopes and A.H. Sayed. Diffusion least-mean squares over adaptive networks: Formulation and performance analysis. *IEEE Transactions on Signal Processing*, 56(7):3122–3136, 2008.

[4] Ioannis D Schizas, Alejandro Ribeiro, and Georgios B Giannakis. Consensus in ad hoc wsns with noisy links - part i: Distributed estimation of deterministic signals. *Signal Processing, IEEE Transactions on*, 56(1):350–364, 2008.

[5] S. Kar, J.M.F. Moura, and K. Ramanan. Distributed parameter estimation in sensor networks: Nonlinear observation models and imperfect communication. *IEEE Transactions on Information Theory*, 58(6):3575 –3605, june 2012.

[6] R. Carli, A. Chiuso, L. Schenato, and S. Zampieri. Distributed kalman filtering based on consensus strategies. *IEEE Journal on Selected Areas in Communications*, 26(4):622–633, 2008.

[7] U.A. Khan and J.M.F. Moura. Distributing the kalman filter for large-scale systems. *IEEE Transactions on Signal Processing*, 56(10):4919–4935, 2008.

[8] A. Ribeiro, I. Schizas, S. Roumeliotis, and G. Giannakis. Kalman filtering in wireless sensor networks. *IEEE Control Systems*, 30(2):66–86, 2010.

[9] O. Hlinka, O. Sluciak, F. Hlawatsch, P.M. Djuric, and M. Rupp. Likelihood consensus and its application to distributed particle filtering. *IEEE Transactions on Signal Processing*, 60(8):4334–4349, 2012.

[10] X. Tai, D. Marelli, Z. Lin, M. Fu, and Y. Sun. A distributed approach to globally optimal state estimation in smart electricity networks. submitted to IEEE Transactions on Power Systems.

[11] Dimitri P Bertsekas and John N Tsitsiklis. Parallel and distributed computation: Numerical methods, 1997.

[12] Peng Yang, Randy A Freeman, Geoffrey J Gordon, Kevin M Lynch, Siddhartha S Srinivasa, and Rahul Sukthankar. Decentralized estimation and control of graph connectivity for mobile sensor networks. *Automatica*, 46(2):390–396, 2010.

[13] James Demmel. The condition number of equivalence transformations that block diagonalize matrix pencils. *SIAM Journal on Numerical Analysis*, 20(3):599–610, 1983.

[14] Yih-Fang Huang, S. Werner, Jing Huang, N. Kashyap, and V. Gupta. State estimation in electric power grids: Meeting new challenges presented by the requirements of the future grid. *IEEE Signal Processing Magazine*, 29(5):33 –43, sept. 2012.

[15] X. Tai, D. Marelli, E. Rohr, and M. Fu. Optimal pmu placement for power system state estimation with random component outages. to appear in International Journal of Electrical Power & Energy Systems.

[16] Usman A Khan, Soummya Kar, and José MF Moura. Distributed sensor localization in random environments using minimal number of anchor nodes. *Signal Processing, IEEE Transactions on*, 57(5):2000–2016, 2009.