POWER ITERATION-BASED DISTRIBUTED TOTAL LEAST SQUARES ESTIMATION IN AD HOC SENSOR NETWORKS

Alexander Bertrand[*], *Marc Moonen*

ESAT-SCD / IBBT - Future Health Department
KU Leuven, University of Leuven
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
E-mail: alexander.bertrand@esat.kuleuven.be; marc.moonen@esat.kuleuven.be

ABSTRACT

In this paper, we revisit the distributed total least squares (D-TLS) algorithm, which operates in an ad hoc sensor network where each node has access to a subset of the equations of an overdetermined set of linear equations. The D-TLS algorithm computes the total least squares (TLS) solution of the full set of equations in a fully distributed fashion (without fusion center). We modify the D-TLS algorithm to eliminate the large computational complexity due to an eigenvalue decomposition (EVD) at every node and in each iteration. In the modified algorithm, a single power iteration (PI) is performed instead of a full EVD computation, which significantly reduces the computational complexity. Since the nodes then do not exchange their true eigenvectors, the theoretical convergence results of the original D-TLS algorithm do not hold anymore. Nevertheless, we find that this PI-based D-TLS algorithm still converges to the network-wide TLS solution, under certain assumptions, which are often satisfied in practice. We provide simulation results to demonstrate the convergence of the algorithm, even when some of these assumptions are not satisfied.

Index Terms— Distributed estimation, wireless sensor networks (WSNs), total least squares

1. INTRODUCTION

Consider the linear regression problem $\mathbf{U}\mathbf{w} = \mathbf{d}$ in the unknown P-dimensional regressor $\mathbf{w}$, with $\mathbf{U}$ an $M \times P$ regression matrix and $\mathbf{d}$ an M-dimensional data vector with $M \geq P$. The problem of solving such an overdetermined set of equations is often encountered in sensor network applications, where nodes can either have access to subsets of the columns of $\mathbf{U}$, e.g. for distributed signal enhancement and beamforming [1], or to subsets of the equations, i.e., subsets of the rows of $\mathbf{U}$ and $\mathbf{d}$, e.g. for distributed system identification [2–4].

In this paper, we consider total least squares (TLS) estimation, which is a popular solution method when both the regression matrix $\mathbf{U}$ and the data vector $\mathbf{d}$ are corrupted by additive noise [5]. The TLS solution can be found by computing the eigenvector corresponding to the smallest eigenvalue of the squared extended data matrix $\mathbf{R} = [\mathbf{U} \mid \mathbf{d}]^H [\mathbf{U} \mid \mathbf{d}]$. In [6], the TLS problem is defined in a wireless sensor network, where the rows of $\mathbf{U}$ and their corresponding entries in $\mathbf{d}$ are distributed over the nodes. The distributed TLS (D-TLS) algorithm computes the network-wide TLS solution in a distributed fashion, i.e., without gathering all the data in a fusion center [6]. In this iterative algorithm, each node solves a local TLS problem in each iteration, and shares its result with its neighbors to update the local TLS matrix at each node.

The D-TLS algorithm has a large computational complexity due to the fact that a local TLS problem needs to be solved in each node and in each iteration of the algorithm, which requires a (partial) eigenvalue decomposition (EVD). This may be too computationally expensive if the unknown regressor $\mathbf{w}$ has a large dimension. In this paper, we relax the task for each node to solve a full local TLS problem in each iteration. Instead, each node performs a single (inverse) power iteration (PI), and then shares the resulting vector with its neighbors. This signficantly decreases the computational cost in each node. We refer to the new algorithm as the PI-based D-TLS algorithm. Since the local TLS problems are only approximately solved in each iteration of the PI-based D-TLS algorithm, the theoretical convergence results from [6] are not valid anymore. Nevertheless, we find that, under certain conditions, which are often satisfied in practice, the PI-based D-TLS algorithm does converge to the network-wide D-TLS solution at each node[1]. We demonstrate by means of simulations that convergence is still obtained, even if some of the conditions are not met (e.g. when using a fixed step size instead of a decreasing step size).

2. DISTRIBUTED TOTAL LEAST SQUARES (D-TLS)

In this section, we briefly review the distributed TLS problem statement and the D-TLS algorithm. For further details, we refer to [6].

Consider an ad hoc WSN with the set of nodes $\mathcal{J} = \{1, \ldots, J\}$ and with a random (connected) topology, where nodes can exchange data with their respective neighbors through a wireless link. We denote the set of neighbor nodes of node k as $\mathcal{N}_k$, i.e., all the nodes that can share data with node k, node k excluded. $|\mathcal{N}_k|$ denotes cardinality of the set $\mathcal{N}_k$, i.e., the number of neighbors of node k. Node k collects a noisy $M_k \times P$ data matrix $\mathbf{U}_k = \overline{\mathbf{U}}_k + \mathbf{N}_k$ and a noisy M_k-dimensional data vector $\mathbf{d}_k = \overline{\mathbf{d}}_k + \mathbf{n}_k$, for which the clean versions $\overline{\mathbf{U}}_k$ and $\overline{\mathbf{d}}_k$ are assumed to be related through a linear regressor $\mathbf{w}$, i.e., $\overline{\mathbf{U}}_k \mathbf{w} = \overline{\mathbf{d}}_k$. The goal is then to solve a TLS problem for the network-wide system of equations in which all $\mathbf{U}_k$'s and $\mathbf{d}_k$'s are

[*] The work of A. Bertrand was supported by a Postdoctoral Fellowship of the Research Foundation - Flanders (FWO). This work was carried out at the ESAT Laboratory of KU Leuven, in the frame of KU Leuven Research Council CoE EF/05/006 'Optimization in Engineering' (OPTEC) and PFV/10/002 (OPTEC), Concerted Research Action GOA-MaNet, the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office IUAP P6/04 (DYSCO, 'Dynamical systems, control and optimization', 2007-2011), Research Project IBBT, and Research Project FWO nr. G.0600.08 ('Signal processing and network design for wireless acoustic sensor networks'). The scientific responsibility is assumed by its authors.

[1] The convergence proof is omitted in this paper due to space constraints (the proof can be found in [7]).

978-1-4673-0046-9/12/$26.00 ©2012 IEEE

Table 1. The distributed total least squares (D-TLS) algorithm.

```
1. $\forall\, k \in \mathcal{J}$: Initialize $\mathbf{R}_k^{(0)} = \mathbf{R}_k$.
2. $i \leftarrow 0$
3. Each node $k \in \mathcal{J}$ computes the eigenvector $\mathbf{x}_k^{(i)}$ corresponding to the smallest eigenvalue of $\mathbf{R}_k^{(i)}$, where $\mathbf{x}_k^{(i)}$ is scaled such that $\|\mathbf{x}_k^{(i)}\| = 1$.
4. Each node $k \in \mathcal{J}$ transmits $\mathbf{x}_k^{(i)}$ to the nodes in $\mathcal{N}_k$.
5. Each node $k \in \mathcal{J}$ updates (with stepsize $\mu_i > 0$)
       $\mathbf{R}_k^{(i+1)} \leftarrow \mathbf{R}_k^{(i)} + \mu_i \left( |\mathcal{N}_k| \mathbf{x}_k^{(i)} \mathbf{x}_k^{(i)\,H} - \sum_{q \in \mathcal{N}_k} \mathbf{x}_q^{(i)} \mathbf{x}_q^{(i)\,H} \right)$.
6. $i \leftarrow i + 1$.
7. return to step 3.
```

stacked. This means we need to solve the optimization problem

$$\min_{\mathbf{w},\triangle\mathbf{U}_1,\ldots,\triangle\mathbf{U}_J,\triangle\mathbf{d}_1,\ldots,\triangle\mathbf{d}_J} \sum_{k\in\mathcal{J}} \left(\|\triangle\mathbf{U}_k\|_F^2 + \|\triangle\mathbf{d}_k\|^2 \right) \qquad (1)$$

$$\text{s.t. } \left(\mathbf{U}_k + \triangle\mathbf{U}_k\right)\mathbf{w} = \left(\mathbf{d}_k + \triangle\mathbf{d}_k\right),\ k = 1, \ldots, J \qquad (2)$$

where $\|.\|_F$ and $\|.\|$ denote the Frobenius norm and the 2-norm, respectively. In [8], it is shown that the TLS estimate is unbiased when the noises $\mathbf{N}_k$ and $\mathbf{n}_k$ contaminating $\overline{\mathbf{U}}_k$ and $\overline{\mathbf{d}}_k$ are zero mean and white (this is not the case for the least squares (LS) estimate, which is always biased when the regressor matrix $\mathbf{U}_k$ is noisy [3]).

The problem (1)-(2) is referred to as the distributed total least squares (D-TLS) problem, since each node only has access to a part of the data. Its solution is denoted as $\mathbf{w}^*$. The goal is to compute $\mathbf{w}^*$ in a distributed fashion, without gathering all data in a fusion center.

In the sequel, $\mathbf{e}_N$ is as an N-dimensional vector with all zeros except for the last entry which is equal to 1, $\mathbf{o}_N$ is the N-dimensional all-zero vector, $\mathbf{I}_N$ denotes the $N \times N$ identity matrix, and $\mathbf{O}_{N\times Q}$ is the $N \times Q$ all-zero matrix. Let $N = P + 1$ and define the $N \times N$ matrix $\mathbf{R}_k = \mathbf{U}_{k+}^H \mathbf{U}_{k+}$, with $\mathbf{U}_{k+} = [\mathbf{U}_k \mid \mathbf{d}_k]$ and superscript H denoting the conjugate transpose operator. Then the solution of the D-TLS problem is given by [5]:

$$\mathbf{w}^* = -\frac{1}{\mathbf{e}_N^T \mathbf{x}^*} \left[\ \mathbf{I}_P \mid \mathbf{o}_P\ \right] \mathbf{x}^* \qquad (3)$$

where $\mathbf{x}^*$ is the eigenvector corresponding to the smallest eigenvalue of $\mathbf{R} = \sum_{k\in\mathcal{J}} \mathbf{R}_k$. This eigenvector can be computed in an iterative distributed fashion by means of the D-TLS algorithm[2] [6], which is given in Table 1. In [6], it has been shown that, $\forall\, k \in \mathcal{J}$, the $\mathbf{x}_k^{(i)}$ in the D-TLS algorithm converges to the eigenvector corresponding to the smallest eigenvalue of $\mathbf{R} = \sum_{k\in\mathcal{J}} \mathbf{R}_k$ under the step size conditions

$$\sum_{i=0}^{\infty} \mu_i = \infty, \quad \sum_{i=0}^{\infty} (\mu_i)^2 < \infty\ . \qquad (4)$$

In each node, the TLS solution of (1)-(2) can then be extracted from this eigenvector (based on (3)).

3. D-TLS WITH POWER ITERATIONS

The computation of the eigenvector corresponding to the smallest eigenvalue of $\mathbf{R}_k^{(i)}$ is an $O(N^3)$ procedure, and therefore the most expensive step of the D-TLS algorithm. In this section, we will modify the D-TLS algorithm such that this step is replaced with a single $N \times N$ matrix-vector multiplication.

Let $\mathbf{R}_k^{(i)} = \mathbf{B}_k^{(i)} \mathbf{\Lambda}_k^{(i)} \mathbf{B}_k^{(i)\,H}$ denote the eigenvalue decomposition of $\mathbf{R}_k^{(i)}$ where $\mathbf{\Lambda}_k^{(i)} = \text{diag}\{\lambda_{k,1}^{(i)}, \ldots, \lambda_{k,N}^{(i)}\}$ such that $\lambda_{k,1}^{(i)} \geq \lambda_{k,2}^{(i)} \geq \ldots \geq \lambda_{k,N}^{(i)}$, $\mathbf{B}_k^{(i)}$ is a unitary matrix, and $\mathbf{b}_{k,j}^{(i)}$ is its j-th column. Let $\mathbf{P}_k^{(i)} = \left(\mathbf{R}_k^{(i)}\right)^{-1}$ (assuming $\lambda_{k,N}^{(i)} > 0$), and assume that $\lambda_{k,N}^{(i)} \neq \lambda_{k,N-1}^{(i)}$. The eigenvector $\mathbf{b}_{k,N}^{(i)}$, corresponding to the smallest eigenvalue $\lambda_{k,N}^{(i)}$ can then be computed by iterating

$$\mathbf{x} \leftarrow \frac{\mathbf{P}_k^{(i)} \mathbf{x}}{\|\mathbf{P}_k^{(i)} \mathbf{x}\|} \qquad (5)$$

starting with a random unity-norm vector that satisfies $\mathbf{x}^H \mathbf{b}_{k,N}^{(i)} \neq 0$. This is referred to as the (inverse) power iteration (PI) method. We could then replace step 3 in the D-TLS algorithm with the above PI procedure. Assuming that the stepsize μ_i is not too large, the eigenvectors $\mathbf{b}_{k,N}^{(i-1)}$ and $\mathbf{b}_{k,N}^{(i)}$ of $\mathbf{R}_k^{(i-1)}$ and $\mathbf{R}_k^{(i)}$ will be close to each other, and hence only a small number of PI's are required if the procedure is initialized with the computed eigenvector $\mathbf{x}_k^{(i-1)}$ from the previous iteration. In the modified algorithm, we will therefore only perform a single PI in each D-TLS iteration.

It is noted that the above procedure requires the inversion of $\mathbf{R}_k^{(i)}$ to obtain $\mathbf{P}_k^{(i)}$, which is an $O(N^3)$ procedure. A recursive update of $\mathbf{P}_k^{(i-1)}$ to obtain $\mathbf{P}_k^{(i)}$ significantly reduces the computational complexity, assuming that $N \gg |\mathcal{N}_k|$. Indeed, step 5 of the algorithm shows that $\mathbf{R}_k^{(i+1)}$ consists of $|\mathcal{N}_k| + 1$ rank-1 updates applied to $\mathbf{R}_k^{(i)}$. The corresponding inverse matrix update can then be computed efficiently with the Woodbury matrix identity [9, Section 2.1.3]:

$$\left(\mathbf{A} \pm \mathbf{x}\mathbf{x}^H\right)^{-1} = \mathbf{A}^{-1} \mp \frac{\left(\mathbf{A}^{-1}\mathbf{x}\right)\left(\mathbf{x}^H \mathbf{A}^{-1}\right)}{1 \pm \mathbf{x}^H \mathbf{A}^{-1} \mathbf{x}}\ . \qquad (6)$$

Assuming $\mathbf{P}_k^{(i-1)}$ is known, we can compute $\mathbf{P}_k^{(i)}$ by applying this rank-1 update $|\mathcal{N}_k| + 1$ times, which is an $O(|\mathcal{N}_k| N^2)$ procedure. We define $R1^{\pm}$ as the operator that computes $\left(\mathbf{A} \pm \mathbf{x}\mathbf{x}^H\right)^{-1}$ based on (6), such that $R1^{\pm}\left(\mathbf{A}^{-1}, \mathbf{x}\right) = \left(\mathbf{A} \pm \mathbf{x}\mathbf{x}^H\right)^{-1}$.

This in effect yields the PI-based D-TLS algorithm, as described in Table 2. The overall complexity at node k is $O(|\mathcal{N}_k| N^2)$.

4. CONVERGENCE PROPERTIES OF THE PI-BASED D-TLS ALGORITHM

The following theorem guarantees the convergence and optimality of the PI-based D-TLS algorithm under the assumptions

Assumption 1: $\sum_{i=0}^{\infty} \mu_i = \infty$
Assumption 2: $\sum_{i=0}^{\infty} (\mu_i)^2 < \infty$
Assumption 3: $\exists\, \kappa_1 > 0,\ \exists\, L_1 \in \mathbb{N},\ \forall\, k \in \mathcal{J}$:

[2] In this paper, we use the more efficient implementation of the D-TLS algorithm (see [6], Section IV-B, Remark I), which has a reduced overhead and memory usage, at the cost of less robustness against node failures. However, all results in this paper also hold for the original implementation of the D-TLS algorithm as derived in [6].

Table 2. The power-iteration based D-TLS algorithm.

```
1. $\forall\, k \in \mathcal{J}$: Initialize $\mathbf{P}_k^{(1)} = (\mathbf{R}_k)^{-1}$ and choose a random $N$-dimensional vector $\mathbf{x}_k^{(0)}$ with unity-norm.
2. $i \leftarrow 1$
3. Each node $k \in \mathcal{J}$ computes
       $\mathbf{x}_k^{(i)} = \frac{\mathbf{P}_k^{(i)}\mathbf{x}_k^{(i-1)}}{\|\mathbf{P}_k^{(i)}\mathbf{x}_k^{(i-1)}\|}$ .
4. Each node $k \in \mathcal{J}$ transmits $\mathbf{x}_k^{(i)}$ to the nodes in $\mathcal{N}_k$.
5. Each node $k \in \mathcal{J}$ computes $\mathbf{P}_k^{(i+1)}$ as follows (for stepsize $\mu_i > 0$)
       • $\mathbf{A} \leftarrow \mathbf{P}_k^{(i)}$
       • $\forall\, q \in \mathcal{N}_k :\ \mathbf{A} \leftarrow R1^{-}\left(\mathbf{A}, \sqrt{\mu_i}\mathbf{x}_q^{(i)}\right)$
       • $\mathbf{P}_k^{(i+1)} = R1^{+}\left(\mathbf{A}, \sqrt{\mu_i|\mathcal{N}_k|}\mathbf{x}_k^{(i)}\right)$.
6. $i \leftarrow i + 1$.
7. return to step 3.
```

$$i > L_1 \Rightarrow \lambda_{k,N-1}^{(i)} - \lambda_{k,N}^{(i)} > \kappa_1$$

Assumption 4: $\exists\, \xi_1 > 0, \forall\, i \in \mathbb{N}, \forall\, k \in \mathcal{J} :\ \lambda_{k,N}^{(i)} > \xi_1$

Theorem 4.1. *Let Assumptions 1 to 4 be satisfied. Then the following holds for the PI-based D-TLS algorithm:*

$$\forall\, k \in \mathcal{J} :\ \lim_{i\to\infty} \mathbf{x}_k^{(i)} = \mathbf{x}^* \qquad (7)$$

where $\mathbf{x}^*$ *is the eigenvector corresponding to the smallest eigenvalue of* $\mathbf{R} = \sum_{k\in\mathcal{J}} \mathbf{R}_k$.

The proof of this theorem is elaborate and is omitted due to space constraints (the proof can be found in [7]). In the remaining of this section, we will discuss the assumptions that were made, and how they impact the implementation and behavior of the PI-based D-TLS algorithm in practice.

4.1. Assumptions 1 and 2

Assumptions 1 and 2 are often imposed to prove convergence in (sub)gradient or relaxation methods (see e.g. [6, 10]), and were already assumed in expression (4) to guarantee convergence of the original D-TLS algorithm. As with the original D-TLS algorithm, Assumption 2 may yield slow convergence in the PI-based D-TLS algorithm, and it is not a practical assumption in tracking applications. However, the fact that convergence can be proven under these conditions is good news since it means that in priniciple an infinite accuracy can be obtained. Furthermore, this usually indicates that the algorithm will at least converge to a neighborhood of the exact solution when using a fixed step size that is sufficiently small. This neighborhood then shrinks with the chosen step size. Simulations will demonstrate that this is indeed true for the PI-based D-TLS algorithm.

4.2. Assumption 3

Assumption 3 guarantees that, if sufficient iterations have passed, the smallest and one but smallest eigenvalue of the matrix $\mathbf{R}_k^{(i)}$ are well-separated in each node, i.e., the smallest eigenvalue does not degenerate. This is a reasonable assumption if the step sizes μ_i are small compared to the separation of the two smallest eigenvalues in the initial matrices $\mathbf{R}_k^{(0)}$, i.e., if the initial local TLS problems are not too ill-conditioned. If the step size is sufficiently small, the eigenvector corresponding to the smallest eigenvalue will then follow a smooth trajectory in the direction of the true solution, rather than making abrupt jumps due to eigenvalue swaps.

Note that, even if the smallest eigenvalue degenerates at a certain node at some iteration, this is not a problem as such, as long as the eigenvalue later iterates away from this degeneration, which is usually the case. Only if the smallest eigenvalue degenerates in (or near) the equilibrium point $\mathbf{R}_k^{(\infty)}$, a problem may occur to reach consensus amongst the nodes. It is noted that the original D-TLS algorithm also breaks down in this case, since step 3 of the algorithm is then ill-defined which results in random behavior. In fact, the PI-based D-TLS algorithm generally behaves better than the D-TLS algorithm in such situations. Indeed, since the eigenvector computation is replaced with a single PI, the estimate will not change very abruptly, even when the eigenvalues $\lambda_{k,N}^{(i)}$ and $\lambda_{k,N-1}^{(i)}$ get 'swapped'.

Since this issue may hinder the consensus, convergence cannot be proven in such situations (for both algorithms). This can only be resolved by choosing a smaller μ_i or by adding extra functionality to the algorithm to ensure that each node selects the same eigenvector. We will not further elaborate on this, since it rarely occurs and it is beyond the scope of this paper.

4.3. Assumption 4

Assumption 4 assures that the smallest eigenvalue remains strictly positive. This avoids rank deficiency of $\mathbf{R}_k^{(i)}$, and it avoids that the (inverse) power iteration method converges to an eigenvector other than $\mathbf{b}_{k,N}^{(i)}$ in case the corresponding eigenvalue $\lambda_{k,N} < 0$ and $\exists\, j < N :\ |\lambda_{k,j}| < |\lambda_{k,N}|$. Assumption 4 is a reasonable assumption if small step sizes μ_i are used and if the initial smallest eigenvalues $\lambda_{k,N}^{(0)}$'s are not too close to zero. If the smallest eigenvalue would still become negative, one can add an identity matrix to $\mathbf{R}_k^{(i)}$ (and make the corresponding changes in $\mathbf{P}_k^{(i)} = \left(\mathbf{R}_k^{(i)}\right)^{-1}$). This does not affect the solution of the D-TLS problem, since the sum $\sum_{k\in\mathcal{J}} \mathbf{R}_k^{(i)}$ is then equal to $\nu\mathbf{I}_N + \sum_{k\in\mathcal{J}} \mathbf{R}_k = \nu\mathbf{I}_N + \mathbf{R}$ (for some ν), which has the same eigenvectors and the same ranking of eigenvalues as the original matrix $\mathbf{R}$.

5. SIMULATION RESULTS

In this section, we provide numerical simulation results that demonstrate the convergence properties of the PI-based D-TLS algorithm. To illustrate the general behavior, we show results that are averaged over multiple Monte-Carlo (MC) runs. The scenario is different in each MC run, and is generated according to the same procedure as described in the simulation section in [6] (which is omitted here due to space constraints).

In each experiment, we choose $J = 20$, $N = P + 1 = 10$. To assess the convergence and optimality of the algorithm, we use the error between the true solution and the local estimate, averaged over

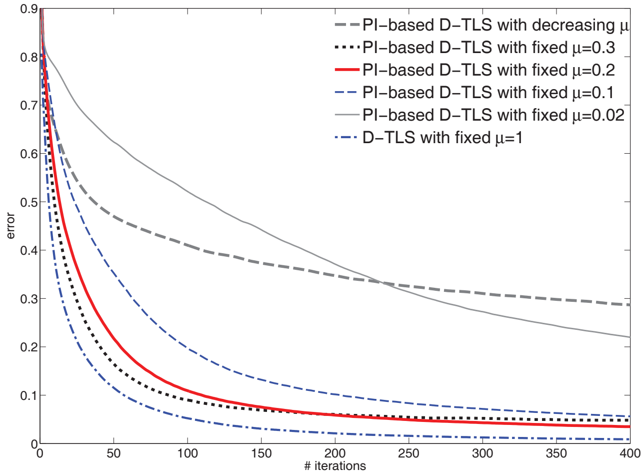

Fig. 1. Convergence properties of the D-TLS and PI-based D-TLS algorithm for different step sizes, averaged over 1000 Monte-Carlo runs.

the J nodes in the network:

$$\frac{1}{|\mathcal{J}|}\sum_{k\in\mathcal{J}} \|\mathbf{x}_k^{(i)} \pm \mathbf{x}^*\|_2 \tag{8}$$

where $\mathbf{x}^*$ is the normalized eigenvector corresponding to the smallest eigenvalue of $\mathbf{R} = \sum_{k\in\mathcal{J}} \mathbf{R}_k$. The '$\pm$' is used to resolve the sign ambiguity (we choose the one that yields the smallest error).

Fig 1 shows the convergence properties over 400 iterations of the PI-based D-TLS algorithm for different choices of the step size μ_i, averaged over 1000 MC runs. We used a fixed step size ranging from 0.02 to 0.3, which results in a converging algorithm even though Assumptions 1 and 2 are not satisfied. Larger step sizes often caused the algorithm not to converge. The figure also shows the results of a variable step size strategy where $\mu_i = 0.3\frac{1}{i^{0.7}}$, which satisfies Assumptions 1 and 2. It is observed that this results in an impractically slow convergence.

It is observed that the convergence speed strongly depends on the choice of the step size. The proper choice of the step size is crucial to obtain a sufficiently fast convergence and sufficient accuracy (as it is also the case in the original D-TLS algorithm). Small step sizes yield slow convergence, but too large step sizes may yield unstable behavior. It is still an open question how the optimal step size strategy can be determined on-line.

The convergence of the original D-TLS algorithm is also shown in Fig. 1 as a reference (with fixed step size $\mu = 1$, which was empirically found to give the best convergence properties for this scenario [6]). It is observed that the original D-TLS algorithm converges faster than the PI-based D-TLS algorithm, especially when the algorithm gets close to the optimal solution. This demonstrates that the original D-TLS algorithm has better tracking capabilities than the PI-based algorithm. However, the true strength of the PI-based D-TLS algorithm is its low computational complexity, especially so for the estimation of large regressors.

Fig. 2 shows the results of 500 MC runs for different $N = P + 1$, i.e., for increasing dimension of the D-TLS problem. It can be observed that the PI-based D-TLS algorithm still converges to the correct solution if the dimension of the TLS problem is large. We did not show the results for the original D-TLS algorithm since the processing time for $N \geq 100$ is too long.

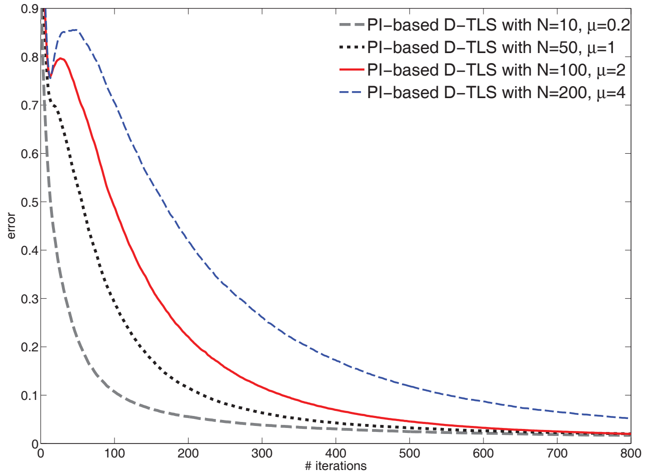

Fig. 2. Convergence properties of the PI-based D-TLS algorithm for different values of N, averaged over 500 Monte-Carlo runs.

6. CONCLUSIONS

In this paper, we have modified the D-TLS algorithm to reduce its computational complexity, by replacing its EVD's with single (inverse) PI's. The PI-based D-TLS algorithm still converges to the network-wide TLS solution under certain assumptions which are often satisfied in practice. We have provided simulation results to demonstrate the convergence of the algorithm, even when some of these assumptions are not satisfied (e.g. when a fixed step size is used instead of a decreasing step size).

7. REFERENCES

[1] A. Bertrand and M. Moonen, "Distributed adaptive node-specific signal estimation in fully connected sensor networks – part I: sequential node updating," *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5277–5291, 2010.

[2] F. Cattivelli, C. G. Lopes, and A. H. Sayed, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Trans. Signal Processing*, vol. 56, no. 5, pp. 1865–1877, May 2008.

[3] A. Bertrand, M. Moonen, and A. H. Sayed, "Diffusion bias-compensated RLS estimation over adaptive networks," *IEEE Transactions on Signal Processing*, 2011.

[4] G. Mateos, I. D. Schizas, and G. B. Giannakis, "Performance analysis of the consensus-based distributed LMS algorithm," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 981030, 19 pages, 2009. doi:10.1155/2009/981030.

[5] I. Markovsky and S. Van Huffel, "Overview of total least-squares methods," *Signal Processing*, vol. 87, no. 10, pp. 2283 – 2302, 2007, special Section: Total Least Squares and Errors-in-Variables Modeling.

[6] A. Bertrand and M. Moonen, "Consensus-based distributed total least squares estimation in ad hoc wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2320–2330, 2011.

[7] ——, "Low-complexity distributed total least squares estimation in ad hoc sensor networks," *Internal report, KU Leuven ESAT/SCD (submitted)*, 2012.

[8] C. E. Davila, "An efficient recursive total least squares algorithm for FIR adaptive filtering," *IEEE Transactions on Signal Processing*, vol. 42, no. 2, pp. 267–280, 1994.

[9] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. Baltimore: The Johns Hopkins University Press, 1996.

[10] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003.