Simple and fast cooperative localization for sensor networks

Cláudia Soares*, *Student Member, IEEE,* João Xavier, *Member, IEEE,* and João Gomes, *Member, IEEE*

***Abstract*—We address the sensor network localization problem given noisy range measurements between pairs of nodes. We approach the non-convex maximum-likelihood formulation via a known simple convex relaxation. We exploit its favorable optimization properties to the full to obtain an approach that: is completely distributed, has a simple implementation at each node, and capitalizes on an optimal gradient method to attain fast convergence. We offer a parallel but also an asynchronous flavor, both with theoretical convergence guarantees and iteration complexity analysis. Experimental results establish leading performance. Our algorithms top the accuracy of a comparable state of the art method by one order of magnitude, using one order of magnitude fewer communications.**

***Index Terms*—Distributed algorithms, convex relaxations, non-convex optimization, maximum likelihood estimation, distributed iterative sensor localization, wireless sensor networks.**

EDICS Category: OPT-CVXR NET-CONT NET-DISP OPT-DOPT

I. Introduction

Sensor networks are becoming ubiquitous. From environmental and infrastructure monitoring to surveillance, and healthcare networked extensions of the human senses in contemporary technological societies are improving our quality of life, our productivity, and our safety. Applications of sensor networks recurrently need to be aware of node positions to fulfill their tasks and deliver meaningful information. Nevertheless, locating the nodes is not trivial: these small, low cost, low power motes are deployed in large numbers, often with imprecise prior knowledge of their locations, and are equipped with minimal processing capabilities. Such limitations call for localization algorithms which are scalable, fast, and parsimonious in their communication and computational requirements.

A. Problem statement

The sensor network is represented as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In the node set $\mathcal{V} = \{1, 2, \dots, n\}$ we represent the sensors with unknown positions. There is an edge $i \sim j \in \mathcal{E}$ between sensors i and j if a noisy range measurement between nodes i and j is available (at both of them) and nodes i and j can communicate with each other. Anchors are elements with known positions and are collected in the set $\mathcal{A} = \{1, \dots, m\}$. For each sensor $i \in \mathcal{V}$, we let $\mathcal{A}_i \subset \mathcal{A}$ be the subset of anchors (if any) whose distance to node i is quantified by a noisy range measurement. The set N_i collects the neighbors of node i.

Let $\mathbb{R}^p$ be the space of interest ($p = 2$ for planar networks, and $p = 3$ otherwise). We denote by $x_i \in \mathbb{R}^p$ the position of sensor i, and by d_{ij} the noisy range measurement between sensors i and j, available at both i and j. Following [1], we assume $d_{ij} = d_{ji}$[1]. Anchor positions are denoted by $a_k \in \mathbb{R}^p$. We let r_{ik} denote the noisy range measurement between sensor i and anchor k, available at sensor i.

The distributed network localization problem addressed in this work consists in estimating the sensors' positions $x = \{x_i : i \in \mathcal{V}\}$, from the available measurements $\{d_{ij} : i \sim j\} \cup \{r_{ik} : i \in \mathcal{V}, k \in \mathcal{A}_i\}$, through collaborative message passing between neighboring sensors in the communication graph $\mathcal{G}$.

Under the assumption of zero-mean, independent and identically-distributed, additive Gaussian measurement noise, the maximum likelihood estimator for the sensor positions is the solution of the optimization problem

$$\underset{x}{\text{minimize}}\, f(x), \tag{1}$$

where

$$f(x) = \sum_{i \sim j} \frac{1}{2}(\|x_i - x_j\| - d_{ij})^2 + \sum_i \sum_{k \in \mathcal{A}_i} \frac{1}{2}(\|x_i - a_k\| - r_{ik})^2.$$

Problem (1) is nonconvex and difficult to solve [2].

B. Contributions

We set forth a convex underestimator of the maximum likelihood cost for the sensor network localization problem (1) based on the convex envelopes of its parcels.

We present an optimal parallel algorithm to minimize this convex underestimator — with proven convergence guarantees. We also propose an asynchronous variant of this algorithm and proof it converges almost surely. Furthermore, we analyze its iteration complexity.

Moreover, we assert the superior performance of our algorithms by computer simulations; we compared several aspects of our method with [3], [2] and [4], and our approach always yields better performance metrics. When compared with the method in [2], which operates under the same conditions, our method outperforms it by one order of magnitude in accuracy and in communication volume.

This work was partially supported by FCT, under projects PEst-OE/EEI/LA0009/2011, PTDC/EMS-CRO/2042/2012, MORPH project (FP 7, 2012-2016), and Grant SFRH/BD/72521/2010. The authors are with the Institute for Systems and Robotics (ISR), Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisboa, Portugal (e-mail: csoares@isr.ist.utl.pt; jxavier@isr.ist.utl.pt; jpg@isr.ist.utl.pt).

[1]This entails no loss of generality: it is readily seen that, if $d_{ij} \neq d_{ji}$, then it suffices to replace $d_{ij} \leftarrow (d_{ij} + d_{ji})/2$ and $d_{ji} \leftarrow (d_{ij} + d_{ji})/2$ in the forthcoming optimization problem (1).

C. Related work

With the advent of large-scale networks, the computational paradigm of information processing algorithms — centralized *versus* distributed — becomes increasingly critical. A centralized method can be less suited for a network with meager communication and computation resources, while a distributed algorithm might not be adequate if the network is supposed to deliver in one place the global result of its computations. Further, none of the available techniques to address Problem (1) claims convergence to the global optimum— due to the nonconvexity, but also due to ambiguities in the network topology which create more than one distant global optimum [5].

Centralized paradigm: The centralized approach to the problem of sensor network localization summoned up a wide body of research. It involves a central processing unit to which all sensor nodes communicate their collected measurements. Centralized architectures are prone to data traffic bottlenecks close to the central node. Resilience to failure, security and privacy issues are, also, not naturally accounted for by the centralized architecture. Moreover, as the number of nodes in the network grows, the problem to be solved at the central node becomes increasingly complex, thus raising scalability concerns.

Focusing on recent work, several different approaches are available, such as the work in [6], where sensor network localization is formulated as a regression problem over adaptive bases. The method has an initialization step using eigendecomposition of an affinity matrix; its entries are functions of squared distance measurements between sensors. The refinement is done by conjugate gradient descent over a discrepancy function of squared distances — which is mathematically more tractable but amplifies measurement errors and outliers and does not benefit from the limiting properties of maximum likelihood estimators. This approach is closely related to multidimensional scaling, where the sensor network localization problem is posed as a least-squares problem, as in [7]. Multidimensional scaling is unreliable in large-scale networks due to their sparse connectivity. Also relying on the well-tested weighted least squares approach, the work in [5] performs successive minimizations of a weighted least squares cost function convolved with a Gaussian kernel of decreasing variance.

Another successfully pursued approach is to perform semidefinite or weaker second-order cone relaxations of the original nonconvex problem (1) [3], [8]. These approaches do not scale well, since the centralized SDP or SOCP problem gets very large even for a small number of nodes. In [3] and [9] the majorization-minimization framework was used with quadratic cost functions to derive centralized approaches to the sensor network localization problem.

Distributed paradigm: In the present work, the expression *distributed method* denotes an algorithm requiring no central or fusion node where all nodes perform the same types of computations. Distributed approaches for cooperative localization have been less frequent than centralized ones, despite the more suited nature of this computational paradigm to sensor networks, when the target application does not require that the global result of the computations be available in one place.

We consider two main approaches to the distributed sensor network localization problem: 1) one where the nonconvex Problem (1) (or some other nonconvex discrepancy minimization) is attacked directly, and hence the quality of the solution is highly dependent on the quality of the algorithm's initialization; 2) and another, where the original nonconvex sensor network localization problem is relaxed to a convex problem, whose tightness will determine how close the solution of the convex problem will approximate the global solution of the original problem, not needing any particular initialization.

a) Initialization dependent: In reference [10] the authors develop a distributed implementation of multidimensional scaling for solution refinement. They support their method on the majorization-minimization framework, but they do not provide a formal proof of convergence for the Jacobi-like iteration. The work in [11] puts forward two distributed methods optimizing the discrepancy of squared distances: a gradient algorithm with Barzilai-Borwein step sizes calculated in a first consensus phase, followed by a gradient computation phase, and a Gauss-Newton algorithm also with a consensus phase and a gradient computation phase. Both are refinement methods that need good initializations to converge to the global optimum.

b) Initialization independent: The work in [12] proposes a parallel distributed algorithm. However, the sensor network localization problem adopts the previously discussed squared distances discrepancy function. Also, each sensor must solve a second order cone program at each algorithm iteration, which can be a demanding task for the simple hardware used in sensor networks' motes. Furthermore, the formal convergence properties of the algorithm are not established. The work in [13] also considers network localization outside a maximum likelihood framework. The approach proposed in [13] is not parallel, operating sequentially through layers of nodes: neighbors of anchors estimate their positions and become anchors themselves, making it possible in turn for their neighbors to estimate their positions, and so on. Position estimation is based on planar geometry-based heuristics. In [14], the authors propose an algorithm with assured asymptotic convergence, but the solution is computationally complex since a triangulation set must be calculated, and matrix operations are pervasive. Furthermore, in order to attain good accuracy, a large number of range measurement rounds must be acquired, one per iteration of the algorithm, thus increasing energy expenditure. On the other hand, the algorithm presented in [1] and based on the non-linear Gauss Seidel framework, has a pleasingly simple implementation, combined with the convergence guarantees inherited from the framework. Notwithstanding, this algorithm is sequential, *i.e.*, nodes perform their calculations in turn, not in a parallel fashion. This entails the existence of a network-wide coordination procedure to precompute the processing schedule upon startup, or whenever a node joins or leaves the network. The sequential nature of the work in [1] was superseded by the work in [2] which puts forward a parallel method based on two consecutive relaxations of the maximum likelihood estimator in (1). The first relaxation is a semidefinite program with a rank relaxation, while the second is an edge based relaxation, best suited for the Alternating Direction

Method of Multipliers (ADMM). The main drawback is the amount of communications required to manage the ADMM variable local copies, and by the prohibitive complexity of the problem at each node. In fact, each one of the simple sensing units must solve a semidefinite program at each ADMM iteration and after the update copies of the edge variables must be exchanged with each neighbor. A simpler approach was devised in [4] by extending the source localization Projection Onto Convex Sets algorithm in [15] to the problem of sensor network localization. The proposed method is sequential, activating nodes one at a time according to a predefined cyclic schedule; thus, it does not take advantage of the parallel nature of the network and imposes a stringent timetable for individual node activity.

II. Convex relaxation

Problem (1) can be written as

$$\underset{x}{\text{minimize}} \sum_{i\sim j} \frac{1}{2}\mathrm{d}^2_{\mathrm{S}_{ij}}(x_i - x_j) + \sum_i \sum_{k\in\mathcal{A}_i} \frac{1}{2}\mathrm{d}^2_{\mathrm{S}_{\mathrm{a}ik}}(x_i), \quad (2)$$

where $\mathrm{d}^2_C(x)$ represents the squared euclidean distance of point x to the set C, *i.e.*, $\mathrm{d}^2_C(x) = \inf_{y\in C}\|x-y\|^2$, and the sets S_{ij} and $\mathrm{S}_{\mathrm{a}ik}$ are defined as the spheres generated by the noisy measurements d_{ij} and r_{ik}

$$\mathrm{S}_{ij} = \{z : \|z\| = d_{ij}\}, \quad \mathrm{S}_{\mathrm{a}ik} = \{z : \|z - a_k\| = r_{ik}\}.$$

Non-convexity of (2) follows from the non-convexity of the building block

$$\frac{1}{2}\mathrm{d}^2_{\mathrm{S}_{ij}}(z) = \frac{1}{2}\inf_{\|y\|=d_{ij}}\|z-y\|^2. \quad (3)$$

A simple convexification consists in replacing it by

$$\frac{1}{2}\mathrm{d}^2_{\mathrm{B}_{ij}}(z) = \frac{1}{2}\inf_{\|y\|\le d_{ij}}\|z-y\|^2 \quad (4)$$

where $\mathrm{B}_{ij} = \{z\in\mathbb{R}^p : \|z\| \le d_{ij}\}$, is the convex hull of S_{ij}. Actually, (4) is the convex envelope[2] of (3). This fact is illustrated in Figure 1 with a one-dimensional example. The

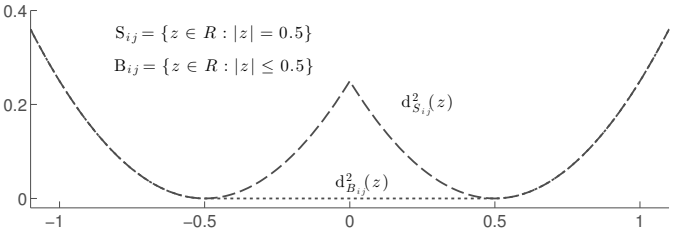

Fig. 1. Illustration of the convex envelope for intersensor terms of the nonconvex cost function (2). The squared distance to the ball B_{ij} (dotted line) is the convex hull of the squared distance to the sphere S_{ij} (dashed line). In this one dimensional example the value of the range measurement is $d_{ij} = 0.5$

terms of (2) associated with anchor measurements are similarly relaxed as

$$\mathrm{d}^2_{\mathrm{B}_{\mathrm{a}ik}}(z) = \inf_{\|y-a_k\|\le r_{ik}}\|z-y\|^2, \quad (5)$$

[2]The convex envelope (or convex hull) of a function γ is its best possible convex underestimator, *i.e.*, $\mathrm{conv}\,\gamma(x) = \sup\{\eta(x) : \eta \le \gamma,\ \eta \text{ is convex}\}$, and is hard to determine in general.

where the set $\mathrm{B}_{\mathrm{a}ik}$ is the convex hull of $\mathrm{S}_{\mathrm{a}ik}$: $\mathrm{B}_{\mathrm{a}ik} = \{z\in\mathbb{R}^p : \|z-a_k\| \le r_{ik}\}$. Replacing the nonconvex parcels in (2) by the sums of terms (4) and (5) we obtain the convex problem

$$\underset{x}{\text{minimize}}\ \hat{f}(x) = \sum_{i\sim j}\frac{1}{2}\mathrm{d}^2_{\mathrm{B}_{ij}}(x_i - x_j) + \sum_i\sum_{k\in\mathcal{A}_i}\frac{1}{2}\mathrm{d}^2_{\mathrm{B}_{\mathrm{a}ik}}(x_i). \quad (6)$$

The function in Problem (6) is an underestimator of (2) but it is not the convex envelope of the original function. We argue that in our application of sensor network localization it is generally a very good approximation whose sub-optimality can be quantified, as discussed in Section IV-A. The cost function (6) also appears in [4] albeit via a distinct reasoning; our convexification mechanism seems more intuitive. But the striking difference with respect to [4] is how (6) is exploited to generate distributed solution methods. Whereas [4] lays out a sequential block-coordinate approach, we show that (6) is amenable to distributed solutions either via the fast Nesterov's gradient method (for synchronous implementations) or exact/inexact randomized block-coordinate methods (for asynchronous implementations).

III. Distributed sensor network localization

Problem (6) can be rewritten as

$$\underset{x}{\text{minimize}}\ \frac{1}{2}\mathrm{d}^2_{\mathrm{B}}(Ax) + \sum_i\sum_{k\in\mathcal{A}_i}\frac{1}{2}\mathrm{d}^2_{\mathrm{B}_{\mathrm{a}ik}}(x_i), \quad (7)$$

where $A = C\otimes I_p$, C is the arc-node incidence matrix of $\mathcal{G}$, I_p is the identity matrix of size p, and B is the cartesian product of the balls B_{ij} corresponding to all the edges in $\mathcal{E}$. We denote the two parcels in (7) as

$$g(x) = \frac{1}{2}\mathrm{d}_{\mathrm{B}}{}^2(Ax), \qquad h(x) = \sum_i h_i(x_i),$$

where $h_i(x_i) = \sum_{k\in\mathcal{A}_i}\frac{1}{2}\mathrm{d}^2_{\mathrm{B}_{\mathrm{a}ik}}(x_i)$.

Now we call on a key result from convex analysis (see [16, Prop. X.3.2.2, Th. X.3.2.3]): the function in (4) is convex, differentiable, and its gradient is

$$\nabla\phi_{\mathrm{B}_{ij}}(z) = z - \mathrm{P}_{\mathrm{B}_{ij}}(z), \quad (8)$$

where $\mathrm{P}_{\mathrm{B}_{ij}}(z)$ is the orthogonal projection of point z onto the closed convex set B_{ij}

$$\mathrm{P}_{\mathrm{B}_{ij}}(z) = \underset{y\in\mathrm{B}_{ij}}{\mathrm{argmin}}\,\|z-y\|.$$

Further, function $\phi_{\mathrm{B}_{ij}}$ has a Lipschitz continuous gradient with constant $L_\phi = 1$, *i.e.*,

$$\|\nabla\phi_{\mathrm{B}_{ij}}(x) - \nabla\phi_{\mathrm{B}_{ij}}(y)\| \le \|x-y\|. \quad (9)$$

The gradient of g is

$$\begin{aligned}\nabla g(x) &= A^\top\nabla\phi_{\mathrm{B}}(Ax)\\ &= A^\top(Ax - \mathrm{P}_{\mathrm{B}}(Ax))\\ &= \mathcal{L}x - A^\top\mathrm{P}_{\mathrm{B}}(Ax), \end{aligned} \quad (10)$$

where the second equality follows from (8) and $\mathcal{L} = L\otimes I_p$, and L is the Laplacian matrix of $\mathcal{G}$. This gradient is Lipschitz

continuous and we can obtain an easily computable Lipschitz constant L_g as follows

$$\begin{aligned}\|\nabla g(x) - \nabla g(y)\| &= \|A^\top (\nabla\phi_{\mathrm{B}}(x) - \nabla\phi_{\mathrm{B}}(y))\| \\ &\leq |||A|||\,\|Ax - Ay\| \\ &\leq |||A|||^2\,\|x-y\| \\ &= \lambda_{\max}(A^\top A)\|x-y\| \\ &\overset{(a)}{=} \lambda_{\max}(L)\|x-y\| \\ &\leq \underbrace{2\delta_{\max}}_{L_g}\|x-y\|, \end{aligned} \tag{11}$$

where $|||A|||$ is the maximum singular value norm; equality (a) is a consequence of Kronecker product properties. In (11) we denote the maximum node degree of $\mathcal{G}$ by $\delta_{\max}$. A proof of the bound $\lambda_{\max}(L) \leq 2\delta_{\max}$ can be found in [17]. The gradient of h is $\nabla h(x) = (\nabla h_1(x_1), \ldots, \nabla h_n(x_n))$, where the gradient of each h_i is

$$\nabla h_i(x_i) = \sum_{k\in\mathcal{A}_i} \nabla\phi_{\mathrm{B}_{aik}}(x_i). \tag{12}$$

We are now able to write $\nabla\hat{f}$, the gradient of our cost function, as

$$\nabla\hat{f}(x) = \mathcal{L}x - A^\top \mathrm{P}_{\mathrm{B}}(Ax) + \begin{bmatrix} \sum_{k\in\mathcal{A}_1} x_1 - \mathrm{P}_{\mathrm{Ba}1k}(x_1) \\ \vdots \\ \sum_{k\in\mathcal{A}_n} x_n - \mathrm{P}_{\mathrm{Ba}nk}(x_n) \end{bmatrix}. \tag{13}$$

The gradient of h is also Lipschitz continuous. The constants L_{h_i} for ∇h_i are

$$\begin{aligned}\|\nabla h_i(x_i) - \nabla h_i(y_i)\| &\leq \sum_{k\in\mathcal{A}_i} \|\nabla\phi_{\mathrm{B}_{aik}}(x_i) - \nabla\phi_{\mathrm{B}_{aik}}(y_i)\| \\ &\leq |\mathcal{A}_i|\|x_i - y_i\|, \end{aligned} \tag{14}$$

where $|\mathcal{C}|$ is the cardinality of set $\mathcal{C}$. We now have an overall constant L_h for ∇h,

$$\begin{aligned}\|\nabla h(x) - \nabla h(y)\| &= \sqrt{\sum_i \|\nabla h_i(x_i) - \nabla h_i(y_i)\|} \\ &\leq \sqrt{\sum_i |\mathcal{A}_i^2|\|x_i - y_i\|} \\ &\leq \underbrace{\max(|\mathcal{A}_i| : i\in\mathcal{V})}_{L_h}\|x-y\|. \end{aligned} \tag{15}$$

A Lipschitz constant $L_{\hat{f}}$ is, thus,

$$L_{\hat{f}} = 2\delta_{max} + \max(|\mathcal{A}_i| : i\in\mathcal{V}). \tag{16}$$

This constant is easy to precompute by, *e.g.*, a diffusion algorithm — c.f. [18, Ch. 9] for more information.

In summary, we can compute the gradient of $\hat{f}$ using Equation (13) and a Lipschitz constant by (16), which leads us to the algorithms described in Sections III-A and III-B for minimizing $\hat{f}$.

A. Parallel method

Since $\hat{f}$ has a Lipschitz continuous gradient we can follow Nesterov's optimal method [19]. Our approach is detailed in Algorithm 1. In step 7, $c_{(i\sim j,i)}$ is the entry $(i\sim j, i)$ in the arc-node incidence matrix C, and δ_i is the degree of node i.

Algorithm 1 Parallel method

Input: $L_{\hat{f}}$; $\{d_{ij} : i\sim j\in\mathcal{E}\}$; $\{r_{ik} : i\in\mathcal{V}, k\in\mathcal{A}\}$;
Output: $\hat{x}$

1: $k = 0$;
2: each node i chooses random $x_i(0)$ and $x_i(-1)$;
3: **while** some stopping criterion is not met, each node i **do**
4: $\quad k = k+1$
5: $\quad w_i = x_i(k-1) + \dfrac{k-2}{k+1}\left(x_i(k-1) - x_i(k-2)\right)$;
6: $\quad$ node i broadcasts w_i to its neighbors
7: $\quad \nabla g_i(w_i) = \delta_i w_i - \sum_{j\in N_i} w_j +$
$\quad + \sum_{j\in N_i} c_{(i\sim j,i)}\mathrm{P}_{\mathrm{B}ij}(w_i - w_j)$;
8: $\quad \nabla h_i(w_i) = \sum_{k\in\mathcal{A}_i} w_i - \mathrm{P}_{\mathrm{B}_{aik}}(w_i)$;
9: $\quad x_i(k) = w_i - \dfrac{1}{L_{\hat{f}}}(\nabla g_i(w_i) + \nabla h_i(w_i))$;
10: **end while**
11: **return** $\hat{x} = x(k)$

Parallel nature of Algorithm 1: It is clear from (12) that $\nabla h_i(x_i)$ can be computed in parallel. It is also known that $(\mathcal{L}x)_i$ can be computed at each node i with knowledge of the estimates of its own position and those of its neighbors: Step 7 in Algorithm 1 makes this computation explicit. The less obvious parallel term is $A^\top\mathrm{P}_{\mathrm{B}}(Ax)$. We start the analysis by the concatenated projections $\mathrm{P}_{\mathrm{B}}(Ax) = \{\mathrm{P}_{\mathrm{B}ij}(x_i - x_j)\}_{i\sim j\in\mathcal{E}}$. Each one of those projections only depends on the edge terminals and the noisy measurement d_{ij}. The product with $A^\top$ will collect, at the entries corresponding to each node, the sum of the projections relative to edges where it intervenes, with a positive or negative sign depending on the arbitrary edge direction agreed upon at the onset of the algorithm. More specifically, $(A^\top\mathrm{P}_{\mathrm{B}}(Ax))_i = \sum_{j\in N_i} c_{(i\sim j,i)}\mathrm{P}_{\mathrm{B}ij}(x_i - x_j)$, as presented in step 7 of Algorithm 1.

B. Asynchronous method

The method described in Algorithm 1 is fully parallel but still depends on some synchronization between all the nodes — so that their updates of the gradient are consistent. This requirement can be inconvenient in some applications of sensor networks; to circumvent it, we present a fully asynchronous method, achieved by means of a broadcast gossip scheme (c.f. [20] for an extended survey of gossip algorithms).

Nodes are equipped with independent clocks ticking at random times (say, as Poisson point processes). When node i's clock ticks, it performs the update of its variable x_i and broadcasts the update to its neighbors. Let the order of node activation be collected in $\{\xi_k\}_{k\in\mathbb{N}}$, a sequence of independent random variables taking values on the set $\mathcal{V}$, such that

$$\mathbb{P}(\xi_k = i) = P_i > 0. \tag{17}$$

Then, the asynchronous update of variable x_i on node i can be described as in Algorithm 2.

Algorithm 2 Asynchronous method

Input: $L_{\hat{f}}; \{d_{ij} : i \sim j \in \mathcal{E}\}; \{r_{ik} : i \in \mathcal{V}, k \in \mathcal{A}\}$;
Output: $\hat{x}$
1: each node i chooses random $x_i(0)$;
2: $k = 0$;
3: **while** some stopping criterion is not met, each node i **do**
4: $\quad k = k + 1$;
5: $\quad x_i(k) = \begin{cases} \operatorname{argmin}_{w_i} \hat{f}(x_1, \ldots, w_i, \ldots, x_n) & \text{if } \xi_k = i \\ x_i(k-1) & \text{otherwise} \end{cases}$
6: **end while**
7: **return** $\hat{x} = x(k)$

Algorithm 3 Asynchronous update at each node i

Input: $\xi_k; L_{\hat{f}}; \{d_{ij} : j \in N_i\}; \{r_{ik} : k \in \mathcal{A}_i\}$;
Output: $x_i(k)$
1: **if** ξ_k **not** i **then**
2: $\quad x_i(k) = x_i(k-1)$;
3: $\quad$ **return** $x_i(k)$;
4: **end if**
5: choose random $z(0)$ and $z(-1)$;
6: $l = 0$;
7: **while** some stopping criterion is not met **do**
8: $\quad l = l + 1$;
9: $\quad w = z(l-1) + \dfrac{l-2}{l+1}(z(l-1) - z(l-2))$;
10: $\quad \nabla \hat{f}_{sl_i}(w) = \dfrac{1}{2} \sum_{j \in N_i} w - \mathrm{P}_{\mathrm{B_S}ij}(w) + \sum_{k \in \mathcal{A}_i} w - \mathrm{P}_{\mathrm{B_a}ik}(w)$
11: $\quad z(l) = w - \dfrac{1}{L_{\hat{f}}} \nabla \hat{f}_{sl_i}(w)$
12: **end while**
13: **return** $x_i(k) = z(l)$

To compute the minimizer in step 5 of Algorithm 2 it is useful to recast Problem (7) as

$$\underset{x}{\text{minimize}} \sum_i \left(\sum_{j \in N_i} \frac{1}{4} \mathrm{d}^2_{\mathrm{B}_{ij}}(x_i - x_j) + \sum_{k \in \mathcal{A}_i} \frac{1}{2} \mathrm{d}^2_{\mathrm{B_a}_{ik}}(x_i) \right), \tag{18}$$

where the factor $\frac{1}{4}$ accounts for the duplicate terms when considering summations over nodes instead of over edges. By fixing the neighbor positions, each node solves a single source localization problem; this setup leads to the Problem

$$\underset{x_i}{\text{minimize}}\, \hat{f}_{sl_i}(x_i) := \sum_{j \in N_i} \frac{1}{4} \mathrm{d}^2_{\mathrm{B_s}_{ij}}(x_i) + \sum_{k \in \mathcal{A}_i} \frac{1}{2} \mathrm{d}^2_{\mathrm{B_a}_{ik}}(x_i), \tag{19}$$

where $\mathrm{B_s}_{ij} = \{z \in \mathbb{R}^p : \|z - x_j\| \leq d_{ij}\}$. We solve Problem (19) at each node by employing Nesterov's optimal accelerated gradient method as described in Algorithm 3. The asynchronous method proposed in Algorithm 2 converges to the set of minimizers of function $\hat{f}$, as established in Theorem 2, in Section IV.

We also propose an inexact version in which nodes do not solve Problem (19) but instead take just one gradient step. That is, simply replace Step 5 in Algorithm 2 by

$$x_i(k) = \begin{cases} x_i(k-1) - \frac{1}{L_{\hat{f}}} \nabla_i \hat{f}(x(k-1)) & \text{if } \xi_k = i \\ x_i(k-1) & \text{otherwise} \end{cases}, \tag{20}$$

where $\nabla_i \hat{f}(x_1, \ldots, x_n)$ is the gradient with respect to x_i, and assume

$$\mathbb{P}(\xi_k = i) = \frac{1}{n}. \tag{21}$$

The convergence terms of the resulting algorithm are established in Theorem 3, Section IV.

IV. Theoretical analysis

A relevant question regarding Algorithms 1 and 2 is whether they will return a good solution to the problem they are designed to solve, after a reasonable amount of computations. Sections IV-B and IV-C address convergence issues of the proposed methods, and discusses some of the assumptions on the problem data. Section IV-A provides a formal bound for the gap between the original and the convexified problems.

A. Quality of the convexified problem

While evaluating any approximation method it is important to know how far the approximate optimum is from the original one. In this Section we will focus on this analysis.

It was already noted in Section II that $\phi_{\mathrm{B}_{ij}}(z) = \phi_{\mathrm{S}_{ij}}(z)$ for $\|z\| \geq d_{ij}$; when the functions differ, for $\|z\| < d_{ij}$, we have that $\phi_{\mathrm{B}_{ij}}(z) = 0$. The same applies to the terms related to anchor measurements. The optimal value of function f, denoted by $f^\star$, is bounded by $\hat{f}^\star = \hat{f}(x^\star) \leq f^\star \leq f(x^\star)$, where $x^\star$ is the minimizer of the convexified problem (6), and $\hat{f}^\star = \inf_x \hat{f}(x)$ is the minimum of function $\hat{f}$. With these inequalities we can compute a bound for the optimality gap, after (6) is solved, as

$$\begin{aligned} f^\star - \hat{f}^\star &\leq f(x^\star) - \hat{f}^\star \\ &= \sum_{i \sim j \in \mathcal{E}} \frac{1}{2} \left(\mathrm{d}^2_{\mathrm{S}_{ij}}(x_i^\star - x_j^\star) - \mathrm{d}^2_{\mathrm{B}_{ij}}(x_i^\star - x_j^\star) \right) \\ &\quad + \sum_{i \in \mathcal{V}} \sum_{k \in \mathcal{A}_i} \frac{1}{2} \left(\mathrm{d}^2_{\mathrm{S_a}_{ik}}(x_i^\star) - \mathrm{d}^2_{\mathrm{B_a}_{ik}}(x_i^\star) \right) \\ &= \sum_{i \sim j \in \mathcal{E}_2} \frac{1}{2} \mathrm{d}^2_{\mathrm{S}_{ij}}(x_i^\star - x_j^\star) + \sum_{i \in \mathcal{V}} \sum_{k \in \mathcal{A}_{2_i}} \frac{1}{2} \mathrm{d}^2_{\mathrm{S_a}_{ik}}(x_i^\star). \end{aligned} \tag{22}$$

In Equation (22), we denote the set of edges where the distance of the estimated positions is less than the distance measurement by $\mathcal{E}_2 = \{i \sim j \in \mathcal{E} : \mathrm{d}^2_{\mathrm{B}_{ij}}(x_i^\star - x_j^\star) = 0\}$, and similarly $\mathcal{A}_{2_i} = \{k \in \mathcal{A}_i : \mathrm{d}^2_{\mathrm{B_a}_{ik}}(x_i^\star) = 0\}$. Inequality (22) suggests a simple method to compute a bound for the optimality gap of the solution returned by the algorithms:

1) Compute the optimal solution $x^\star$ using Algorithm 1 or 2;
2) Select the terms of the convexified problem (6) which are zero;
3) Add the nonconvex costs of each of these edges, as in (22).

TABLE I
BOUNDS ON THE OPTIMALITY GAP FOR THE EXAMPLE IN FIGURE 2

$f^\star - \hat{f}^\star$	**Equation** (22)	**Equation** (23)
0.0367	0.0487	3.0871

Our bound is tighter than the one (available *a priori*) from applying [21, Th. 1], which is

$$f^\star - \hat{f}^\star \leq \sum_{i\sim j\in\mathcal{E}} \frac{1}{2} d_{ij}^2 + \sum_{i\in\mathcal{V}} \sum_{k\in\mathcal{A}_i} \frac{1}{2} r_{ik}^2. \tag{23}$$

For the one dimensional example of the star network costs depicted in Figure 2 the bounds in (22), and (23) averaged

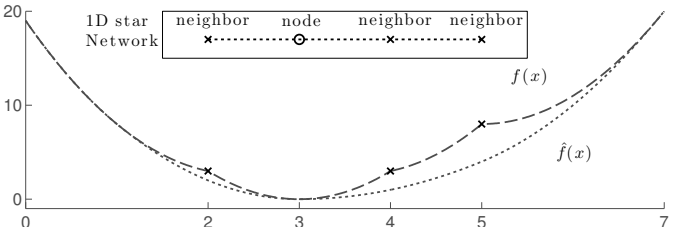

Fig. 2. One-dimensional example of the quality of the approximation of the true nonconvex cost $f(x)$ by the convexified function $\hat{f}(x)$ in a star network. Here the node positioned at $x = 3$ has 3 neighbors.

over 500 Monte Carlo trials are presented in Table I. The true average gap $f^\star - \hat{f}^\star$ is also shown. In the Monte Carlo trials we sampled a zero mean gaussian random variable with $\sigma = 0.25$ and obtained a noisy range measurement as described later by (28). These results show the tightness of the convexified function and how loose the bound (23) is when applied to our problem.

B. Parallel method: convergence guarantees and iteration complexity

As Problem (7) is convex and the cost function has a Lipschitz continuous gradient, Algorithm 1 is known to converge at the optimal rate $O\left(k^{-2}\right)$ [19], [22]: $\hat{f}(x(k)) - \hat{f}^\star \leq \frac{2L_{\hat{f}}}{(k+1)^2} \left\|x(0) - x^\star\right\|^2$.

C. Asynchronous method: convergence guarantees and iteration complexity

To state the convergence properties of Algorithm 2 we only need Assumption 1.

Assumption 1. *There is at least one anchor for each connected component in $\mathcal{G}$.*

This assumption holds generally in practice as one needs $p+1$ anchors to eliminate translation, rotation, and flip ambiguities while performing localization in $\mathbb{R}^p$. We present two convergence results, — Theorem 2, and Theorem 3 — and the iteration complexity analysis for Algorithm 2 in Proposition 4. Proofs of the Theorems are detailed in Appendix A.

The following Theorem establishes the almost sure (*a.s.*) convergence of Algorithm 2.

Theorem 2 (Almost sure convergence of Algorithm 2)**.** *Let $\{x(k)\}_{k\in\mathbb{N}}$ be the sequence of points produced by Algorithm 2, or of Algorithm 2 with the update* (20)*, and let $\mathcal{X}^\star = \{x^\star : \hat{f}(x^\star) = \hat{f}^\star\}$ be the set of minimizers of function $\hat{f}$ defined in Equation* (6)*. Then it holds:*

$$\lim_{k\to\infty} \mathrm{d}_{\mathcal{X}^\star}(x(k)) = 0, \qquad a.s. \tag{24}$$

In words, with probability one, the iterates $x(k)$ will approach the set $\mathcal{X}^\star$ of minimizers of $\hat{f}$; this does not imply $\{x(k)\}_{k\in\mathbb{N}}$ will converge to one single $x^\star \in \mathcal{X}^\star$, but it does imply that $\lim_{k\to\infty} \hat{f}(x(k)) = \hat{f}^\star$, since $\mathcal{X}^\star$ is a compact set, as proven in Appendix A, Lemma 5.

Theorem 3 (Almost sure convergence to a point)**.** *Let $\{x(k)\}_{k\in\mathbb{N}}$ be a sequence of points generated by Algorithm 2, with the update* (20) *in Step 5, and let all nodes start computations with uniform probability. Then, with probability one, there exists a minimizer of $\hat{f}$, denoted by $x^\star \in \mathcal{X}^\star$, such that*

$$x(k) \to x^\star. \tag{25}$$

This result tells us that the iterates of Algorithm 2 with the modified Step 5 stated in Equation (20) not only converge to the solution set, but also guarantees that they will not be jumping around the solution set $\mathcal{X}^\star$ (unlikely to occur in Algorithm 2, but not ruled out by the analysis). One of the practical benefits of Theorem 3 is that the stopping criterion can safely probe the stability of the estimates along iterations. To the best of our knowledge, this kind of strong type of convergence (the whole sequence converges to a point in $\mathcal{X}^\star$) was not established previously in the context of randomized approaches for convex functions with Lipschitz continuous gradients.

Proposition 4 (Iteration complexity for Algorithm 2)**.** *Let $\{x(k)\}_{k\in\mathbb{N}}$ be a sequence of points generated by Algorithm 2, with the update* (20) *in Step 5, and let the nodes be activated with equal probability. Choose $0 < \epsilon < \hat{f}(x(0)) - \hat{f}^\star$ and $\rho \in (0,1)$. There exists a constant $b(\rho, x(0))$ such that*

$$\mathbb{P}\left(\hat{f}(x(k)) - \hat{f}^\star \leq \epsilon\right) \geq 1 - \rho \tag{26}$$

for all

$$k \geq K = \frac{2nb(\rho, x(0))}{\epsilon} + 2 - n. \tag{27}$$

The constant $b(x(0), \rho)$ can be computed from inequality (19) in [23]; it depends only on the initialization and the chosen ρ. Proposition 4 is saying that, with high probability, the function value $\hat{f}(x(k))$ for all $k \geq K$ will be at a distance ϵ of the optimal, and the number of iterations K depends inversely on the chosen ϵ.

Proof of Proposition 4: As $\hat{f}$ is differentiable and has Lipschitz gradient, the result is trivially deduced from [23, Th. 2]. ■

V. NUMERICAL EXPERIMENTS

In this Section we present experimental results that demonstrate the superior performance of our methods when compared with four state of the art algorithms: Euclidean Distance

Matrix (EDM) completion presented in [3], Semidefinite Program (SDP) relaxation and Edge-based Semidefinite Program (ESDP) relaxation, both implemented in [2], and a sequential projection method (PM) in [4] optimizing the same convex underestimator as the present work, with a different algorithm. The fist two methods — EDM completion and SDP relaxation — are centralized, whereas the ESDP relaxation and PM are distributed.

Methods: We conducted simulations with two uniquely localizable geometric networks with sensors randomly distributed in a two-dimensional square of size 1×1 with 4 anchors in the corners of the square. Network 1 has 10 sensor nodes with an average node degree[3] of 4.3, while network 2 has 50 sensor nodes and average node degree of 6.1. The ESDP method was only evaluated in network 1 due to simulation time constraints, since it involves solving an SDP at each node, and each iteration. The noisy range measurements are generated according to

$$d_{ij} = \quad |\|x_i^\star - x_j^\star\| + \nu_{ij}|, \; r_{ik} = \quad |\|x_i^\star - a_k\| + \nu_{ik}|, \quad (28)$$

where $x_i^\star$ is the true position of node i, and $\{\nu_{ij} : i \sim j \in \mathcal{E}\} \cup \{\nu_{ik} : i \in \mathcal{V}, k \in \mathcal{A}_i\}$ are independent gaussian random variables with zero mean and standard deviation σ. The accuracy of the algorithms is measured by the original nonconvex cost value in (1) and by the Root Mean Squared Error (RMSE) per sensor, defined as

$$RMSE = \sqrt{\frac{1}{n}\left(\frac{1}{MC}\sum_{mc=1}^{MC} \|x^\star - \hat{x}(mc)\|^2\right)}, \quad (29)$$

where MC is the number of Monte Carlo trials performed.

A. Relaxation quality

The first experiment aimed at exploring the relaxation quality when compared with two state of the art methods. For the proposed disk relaxation Algorithm 1 was stopped when the gradient norm $\|\nabla \hat{f}(x)\|$ reached 10^{-6} while both EDM completion and SDP relaxation were solved with the default *SeDuMi* solver [24] `eps` value of 10^{-9}, so that algorithm properties did not mask the real quality of the relaxations. Figures 3 and 4 report the results of the experiment with 50 Monte Carlo trials over network 2 and measurement noise with $\sigma = [0.01,\ 0.05,\ 0.1,\ 0.3]$; so, we had a total of 200 runs, equally divided by the 4 noise levels. In Figure 3 we can see that the disk relaxation in (6) has better performance for all noise levels. Figure 4 locates the results of optimizing the three convex functions for the same problems in RMSE *versus* execution time, indicating the complexity of the optimization of the considered costs. The convex surrogate used in the present work combined with our methods is faster by at least one order of magnitude.

[3]To characterize the used networks we resort to the concepts of *node degree* k_i, which is the number of edges connected to node i, and *average node degree* $< k >= 1/n \sum_{i=1}^{n} k_i$.

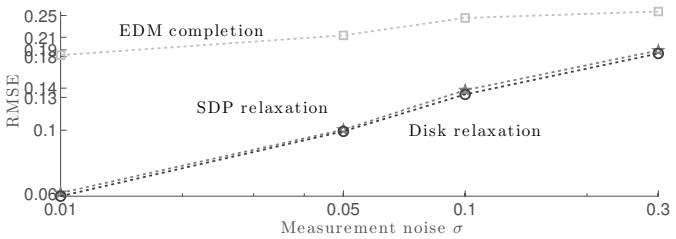

Fig. 3. Relaxation quality: Root mean square error comparison of EDM completion in [3], SDP relaxation in [2] and the disk relaxation used in the present work; measurements were perturbed with noise with different values for the standard deviation σ. The disk relaxation approach in (6) improved on the RMSE values of both EDM completion and SDP relaxation for all noise levels, even though it does not rely on the SDP machinery. The performance gap to EDM completion is substantial.

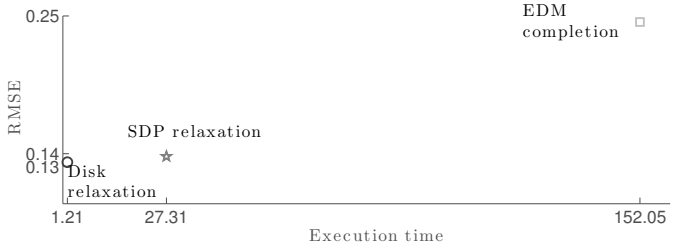

Fig. 4. Relaxation quality: Comparison of the best achievable root mean square error *versus* overall execution time of the algorithms. Measurements were contaminated with noise with $\sigma = 0.1$. Although disk relaxation has a distributed implementation, running it sequentially can be faster by one order of magnitude than the centralized methods.

B. Parallel and asynchronous performance

A second experiment consisted on testing the performance of the parallel and the asynchronous flavors of our method, presented respectively in Algorithms 1 and 2. The metric was the value of the convex cost function $\hat{f}$ in (6) evaluated at each algorithm's estimate of the minimum. To have a fair comparison, both algorithms were allowed to run until they reached a preset number of communications. In Figure 5 we present the effectiveness of both algorithms in optimizing the disk relaxation cost in (6), with the same amount of communications. We chose the uniform probability law for the random variables ξ_k representing the sequence of updating nodes in the asynchronous version of our method. Again, we ran 50 Monte Carlo trials, each with 3 noise levels, thus leading to 150 samplings of the noise variables in (28).

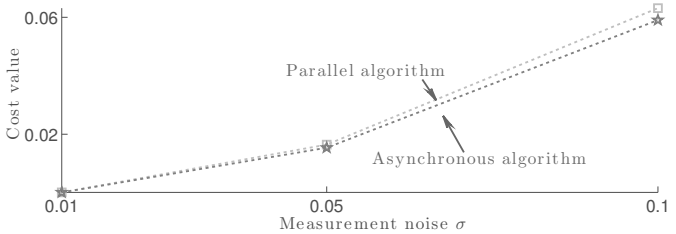

Fig. 5. Final cost of the parallel Algorithm 1 and its asynchronous counterpart in Algorithm 2 for the same number of communications. Results for the asynchronous version degrade less than those of the parallel one as the noise level increases. The stochastic Gauss-Seidel iterations prove to be more robust to intense noise.

C. Performance of distributed optimization algorithms

To measure the performance of the presented algorithm in a distributed setting we compared it with the state of the art methods in [4] and the distributed algorithm in [2]. The results are shown, respectively, in Figures 6 and 7. The experimental setups were different, since the authors proposed different stopping criteria for their algorithms and, in order to do a fair comparison, we ran our algorithm with the specific criterion set by each benchmark method. So, in the experiment

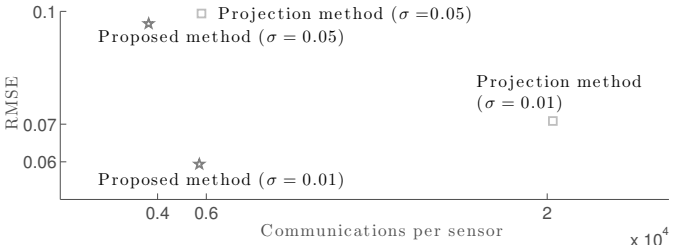

Fig. 6. Performance of the proposed method and of the Projection method presented in [4]. The stopping criterion for both algorithms was a relative improvement of 10^{-6} in the estimate. The proposed method uses fewer communications to achieve better RMSE for the tested noise levels. Our method outperforms the projection method with one forth of the number of communications for a noise level of 0.01.

illustrated in Figure 6, the stopping criterion for both the projection method and the presented method was the relative improvement of the solution; we stress that this is not a distributed stopping criterion, we adopted it just for algorithm comparison. We can see that the proposed method fares better not only in RMSE but, foremost, in communication cost. The experiment comprised 120 Monte Carlo trials and two noise levels.

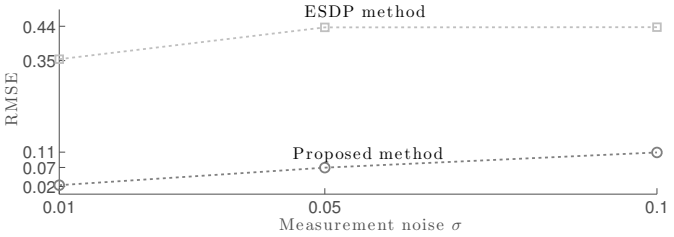

Fig. 7. Performance of the proposed method and of the ESDP method in [2]. The stopping criterion for both algorithms was the number of algorithm iterations. The performance advantage of the proposed method is even more remarkable when considering the number of communications presented in Table II.

To compare with the distributed ESDP method in [2] we had to use a smaller network of 10 sensors because of simulation time constraints — as the ESDP method entails solving an SDP problem at each node, the simulation time becomes prohibitively large, at least using a general purpose solver. The number of Monte Carlo trials was 32, with 3 noise levels, leading to 96 realizations for each noisy measurement. From the analysis of both Figure 7 and Table II we can see that the ESDP method is one order of magnitude worse in RMSE performance, using one order of magnitude more communications, than the proposed algorithm.

TABLE II
NUMBER OF COMMUNICATIONS PER SENSOR FOR THE RESULTS IN FIG. 7

ESDP method	**Proposed method**
21600	2000

VI. CONCLUDING REMARKS

Experiments in Section V show that our method is superior to the state of the art in all measured indicators. While the comparison with the projection method published in [4] is favorable to our proposal, it should be further considered that the projection method has a different nature when compared to ours: It is sequential, and such algorithms will always have a larger computation time than parallel ones, since nodes run in sequence; moreover, this computation time grows with the number of sensors while parallel methods retain similar speed, no matter how many sensors the network has.

When comparing with the ESDP method in [2], which is distributed and parallel,similarly to our proposed method in Algorithm 1, we can see one order of magnitude improvement in RMSE for one order of magnitude fewer communications of our method — and this score is achieved with a simpler, easy to implement algorithm, performing simple computations at each node that are well suited to the kind of hardware commonly found in sensor networks.

There are some important questions not addressed here. For example, it is not clear what influence the number of anchors and their spatial distribution can have in the performance of the proposed and state of the art algorithms. Also, an exhaustive study on the impact of varying topologies and number of sensors could lead to interesting results.

But with the data presented here one can already grasp the advantages of our fast and easily implementable distributed method, where the optimality gap of the solution can also be easily quantified, and which offers two implementation flavours for different localization needs.

ACKNOWLEDGEMENTS

The authors would like to thank Pinar Oguz-Ekim and Andrea Simonetto for providing the Matlab implementation of the methods in their papers.

APPENDIX A
PROOFS

A. Auxiliary Lemmas

In this Section we establish basic properties of Problem (7) in Lemma 5 and also two technical Lemmas, instrumental to prove our convergence results in Theorem 2.

Lemma 5 (Basic properties)**.** *Let $\hat{f}$ as defined in* (7)*. Then the following properties hold.*

1) *$\hat{f}$ is coercive;*
2) *$\hat{f}^\star \geq 0$ and $\mathcal{X}^\star \neq \varnothing$;*
3) *$\mathcal{X}^\star$ is compact;*

Proof:

1) By Assumption 1 there is a path from each node i to some node j which is connected to an anchor k. If $\|x_i\| \to \infty$ then there are two cases: (1) there is at least one edge $t \sim u$ along the path from i to j where $\|x_t\| \to \infty$ and $\|x_u\| \not\to \infty$, and so $\mathrm{d}^2_{\mathrm{B}_{tu}}(x_t - x_u) \to \infty$; (2) if $\|x_u\| \to \infty$ for all u in the path between i and j, in particular we have $\|x_j\| \to \infty$ and so $\mathrm{d}^2_{\mathrm{B}_{ajk}}(x_j) \to \infty$, and in both cases $\hat{f} \to \infty$, thus, $\hat{f}$ is coercive.
2) Function $\hat{f}$ defined in (6) is a sum of squares, continuous, convex and real valued function lower bounded by zero; so, the infimum $\hat{f}^\star$ exists and is non-negative. To prove this infimum is attained and $\mathcal{X}^\star \neq \varnothing$, we consider the set $T = \{x : \hat{f}(x) \leq \alpha\}$; T is a sublevel set of a continuous, coercive function and, thus, it is compact. As $\hat{f}$ is continuous, by the Weierstrass Theorem, the value $p = \inf_{x \in T} \hat{f}(x)$ is attained; the equality $\hat{f}^\star = p$ is evident.
3) $\mathcal{X}^\star$ is a sublevel set of a continuous coercive function and, thus, compact.

■

Lemma 6. *Let $\{x(k)\}_{k\in\mathbb{N}}$ be the sequence of iterates of Algorithm 2, or of Algorithm 2 with the update* (20)*, and $\nabla \hat{f}\left(x(k)\right)$ be the gradient of function $\hat{f}$ evaluated at each iterate. Then,*

1) $\sum_{k\geq 1} \|\nabla \hat{f}\left(x(k)\right)\|^2 < \infty,\ a.s.$;
2) $\nabla \hat{f}\left(x(k)\right) \to 0,\ a.s.$

Proof: Let $\mathcal{F}_k = \sigma\left(x(0), \cdots, x(k)\right)$ be the sigma-algebra generated by all algorithm iterations until time k. We are interested in $\mathbb{E}\left[\hat{f}\left(x(k)\right)|\mathcal{F}_{k-1}\right]$, the expected value of the cost value of the kth iteration, given the knowledge of the past $k-1$ iterations. Firstly, let us examine function $\phi : \mathbb{R}^p \to \mathbb{R}$, the slice of $\hat{f}$ along a coordinate direction, $\phi(y) = \hat{f}(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_n)$. As $\hat{f}$ has Lipschitz continuous gradient with constant L, so will ϕ: $\|\nabla\phi(y) - \nabla\phi(z)\| \leq L\|y - z\|$, for all y and x, and, thus, it will inherit the property

$$\phi(y) \leq \phi(z) + \langle \nabla\phi(z), y - z\rangle + \frac{L}{2}\|y - z\|^2. \tag{30}$$

The minimizer of the quadratic upper-bound in (30) is $z - \frac{1}{L}\nabla\phi(z)$, which can be plugged back in (30), obtaining

$$\phi^\star \leq \phi\left(z - \frac{1}{L}\nabla\phi(z)\right) \leq \phi(z) - \frac{1}{2L}\|\nabla\phi(z)\|^2. \tag{31}$$

In the sequel, for a given $x = (x_1, \ldots, x_n)$, we let

$$\hat{f}_i^\star(x_{-i}) = \inf\{\hat{f}(x_1, \ldots, x_{i-1}, z, x_{i+1}, \ldots, x_n) : z\}.$$

Going back to the expectation $\mathbb{E}\left[\hat{f}\left(x(k)\right)|\mathcal{F}_{k-1}\right] = \sum_{i=1}^n P_i \hat{f}_i^\star\left(x_{-i}(k-1)\right)$, we can bound it from above, recurring to (31), by

$$\begin{aligned}
&\sum_{i=1}^n P_i \left(\hat{f}(x(k-1)) - \frac{1}{2L}\|\nabla_i \hat{f}(x(k-1))\|^2\right) \\
&= \hat{f}(x(k-1)) - \frac{1}{2L}\sum_{i=1}^n P_i \|\nabla_i \hat{f}(x(k-1))\|^2 \\
&\overset{(a)}{\leq} \hat{f}(x(k-1)) - \frac{P_{\min}}{2L}\|\nabla \hat{f}(x(k-1))\|^2,
\end{aligned} \tag{32}$$

where we used $0 < P_{\min} \leq P_i$, for all $i \in \{1, \cdots, n\}$ in (a). To alleviate notation, let $g(k) = \nabla \hat{f}(x(k))$; we then have

$$\|g(k)\|^2 = \sum_{i\leq k} \|g(i)\|^2 - \sum_{i\leq k-1} \|g(i)\|^2,$$

and adding $\frac{P_{\min}}{2L}\sum_{i\leq k-1}\|g(i)\|^2$ to both sides of the inequality in (32), we find that

$$\mathbb{E}\left[Y_k|\mathcal{F}_{k-1}\right] \leq Y_{k-1}, \tag{33}$$

where $Y_k = \hat{f}(x(k)) + \frac{P_{\min}}{2L}\sum_{i\leq k-1}\|g(i)\|^2$. Inequality (33) defines the sequence $\{Y_k\}_{k\in\mathbb{N}}$ as a supermartingale. As $\hat{f}(x)$ is always non-negative, then Y_k is also non-negative and so [25, Corollary 27.1],

$$Y_k \to Y,\ a.s.$$

In words, the sequence Y_k converges almost surely to an integrable random variable Y. This entails that $\sum_{k\geq 1}\|g(k)\|^2 < \infty,\ a.s.$, and so, $g(k) \to 0,\ a.s.$ ■

Lemma 7. *Let $\{x(k)\}_{k\in\mathbb{N}}$ be one of the sequences generated with probability one according to Lemma 6. Then,*

1) The function value decreases to the optimum: $\hat{f}(x(k)) \downarrow \hat{f}^\star$;
2) There exists a subsequence of $\{x(k)\}_{k\in\mathbb{N}}$ converging to a point in $\mathcal{X}^\star$: $x(k_l) \to y,\ y \in \mathcal{X}^\star$.

Proof: As $\hat{f}$ is coercive, then the sublevel set $\mathcal{X}_{\hat{f}} = \left\{x : \hat{f}(x) \leq \hat{f}(x(0))\right\}$ is compact and, because $\hat{f}(x(k))$ is non increasing, all elements of $\{x(k)\}_{k\in\mathbb{N}}$ belong to this set. From the compactness of $\mathcal{X}_{\hat{f}}$ we have that there is a convergent subsequence $x(k_l) \to y$. We evaluate the gradient at this accumulation point, $\nabla\hat{f}(y) = \lim_{l\to\infty}\nabla\hat{f}(x(k_l))$, which, by assumption, vanishes, and we therefore conclude that y belongs to the solution set $\mathcal{X}^\star$. Moreover, the function value at this point is, by definition, the optimal value. ■

B. Theorems

Equipped with the previous Lemmas, we are now ready to prove the Theorems stated in Section IV.

Proof of Theorem 2: Suppose the distance does not converge to zero. Then, there exists an $\epsilon > 0$ and some subsequence $\{x(k_l)\}_{l\in\mathbb{N}}$ such that $\mathrm{d}_{\mathcal{X}^\star}(x(k_l)) > \epsilon$. But, as $\hat{f}$ is coercive (by Lemma 5), continuous, and convex, and whose gradient, by Lemma 6, vanishes, then by Lemma 7, there is a subsequence of $\{x(k_l)\}_{l\in\mathbb{N}}$ converging to a point in $\mathcal{X}^\star$, which is a contradiction. ■

Proof of Theorem 3: Fix an arbitrary point $x^\star \in \mathcal{X}^\star$. We start by proving that the sequence of squared distances

to $x^\star$ of the estimate produced by Algorithm 2, with the update defined in Equation (20), converges almost surely; that is, the sequence $\{\|x(k)-x^\star\|^2\}_{k\in\mathbb{N}}$ is convergent with probability one. We have

$$\mathbb{E}\left[\|x(k)-x^\star\|^2|\mathcal{F}_{k-1}\right] = \sum_{i=1}^{n}\frac{1}{n}\left\|x(k-1)-\frac{1}{L_{\hat{f}}}g_i(k-1)-x^\star\right\|^2 \quad (34)$$

where $g_i(k-1)=(0,\ldots,0,\nabla_i\hat{f}(x(k-1)),0,\ldots,0)$ and $\mathcal{F}_k=\sigma\left(x(1),\ldots,x(k)\right)$ is the sigma-algebra generated by all iterates until time k. Expanding the left-hand side of (34) yields

$$\|x(k-1)-x^\star\|^2+\frac{1}{nL_{\hat{f}}^2}\left\|\nabla\hat{f}(x(k-1))\right\|^2 -\frac{2}{nL_{\hat{f}}}(x(k-1)-x^\star)^\top\nabla\hat{f}(x(k-1)).$$

Since $(x(k-1)-x^\star)^\top\nabla\hat{f}(x(k-1)) = (x(k-1)-x^\star)^\top\left(\nabla\hat{f}(x(k-1))-\nabla f(x^\star)\right)\geq 0$, we conclude that

$$\mathbb{E}\left[\|x(k)-x^\star\|^2|\mathcal{F}_{k-1}\right] \leq \|x(k-1)-x^\star\|^2+\frac{1}{nL_{\hat{f}}^2}\left\|\nabla\hat{f}(x(k-1))\right\|^2.$$

Now, as proved in Lemma 6, the sum $\sum_k\|\nabla\hat{f}(x(k))\|^2$ converges almost surely. Thus, invoking the result in [26], we get that $\|x(k)-x^\star\|^2$ converges almost surely.

Equipped with this key tool, we proceed with the proof of Theorem 3. If $\mathcal{X}^\star$ is a singleton, we are done. Otherwise, let $d\geq 1$ be the affine dimension of the convex set $\mathcal{X}^\star$, and fix $d+1$ affinely independent points $x_0^\star,x_1^\star,\ldots,x_d^\star\in\mathcal{X}^\star$. It follows that, with probability one, $\{\|x(k)-x_i^\star\|^2\}_{k\in\mathbb{N}}$ converges for $i=0,1,\ldots,d$. In the sequel, we assume that $\{x(k)\}_{k\in\mathbb{N}}$ is such a sequence.

Let $V=x_0^\star+L$ where $L=\{\sum_{i=1}^d a_i(x_i^\star-x_0^\star) : a_i\in\mathbb{R}\}$ is a d-dimensional linear subspace. Note that V is the affine span of $\mathcal{X}^\star$ (the smallest affine subspace containing $\mathcal{X}^\star$). An important point now is to note that each $x\in V$ is uniquely defined by $v(x)=(\|x-x_0^\star\|^2,\ldots,\|x-x_d^\star\|^2)$, the vector of its squared distances to the $d+1$ points $x_i^\star$, $i=0,\ldots,d$. Indeed, if $v(x)=(v_0(x),v_1(x),\ldots,v_d(x))$ then $\|x-x_i^\star\|^2=v_i(x)$ for $i=0,1,\ldots,d$, and subtracting each such ith equation (with $i\geq 1$) from the first one (with $i=0$) gives $(x_i^\star-x_0^\star)^\top x=\frac{1}{2}(v_0(x)-v_i(x))-\frac{1}{2}(\|x_0^\star\|^2-\|x_i^\star\|^2)$, for $i=1,\ldots,d$. For conciseness, write the last linear system as $B^\top x=b$ where the ith column of $B\in\mathbb{R}^{d\times d}$ is $x_i^\star-x_0^\star$. Since $x\in V$ by assumption, there holds $x=x_0^\star+Ba$ for $a\in\mathbb{R}^d$. Plugging this representation in the former linear system gives $B^\top Ba=b-B^\top x_0^\star$, which can be solved (uniquely) for a as B is nonsingular (thanks to the affine independence of $x_i^\star$, $i=0,\ldots,d$). For further reference, note that a is an affine map of $v(x)$. Finally, note that each $x\in\mathbb{R}^{np}$ can be uniquely written as $x=x_0^\star+y+z$ where $y\in L$ (thus, $x_0^\star+y\in V$) and $z\in L^\perp$ (the orthogonal complement of L); there holds: $\mathrm{d}_{\mathcal{X}^\star}(x)\geq\mathrm{d}_V(x)=\|z\|$ and $\|x-w\|^2=\|x_0^\star+y-w\|^2+\|z\|^2$ for any $w\in V$.

Write $x(k)=x_0^\star+y(k)+z(k)$ as per the last definition. Since $\mathrm{d}_{\mathcal{X}^\star}(x(k))\to 0$ we conclude that $z(k)\to 0$. Using this result in the equalities $\|x-x_i^\star\|^2=\|x_0^\star+y(k)-x_i^\star\|^2+\|z(k)\|^2$, $i=0,1,\ldots,d$, shows that $\|x_0^\star+y(k)-x_i^\star(k)\|^2$ converges for $i=0,\ldots,d$. Since, as shown above, $x_0^\star+y(k)$ is an affine map of the vector $v(x_0^\star+y(k))=(\|x_0^\star+y(k)-x_0^\star(k)\|^2,\ldots,\|x_0^\star+y(k)-x_d^\star(k)\|^2)$, we conclude that $x_0^\star+y(k)$ is convergent and the theorem is proved. ■

REFERENCES

[1] Q. Shi, C. He, H. Chen, and L. Jiang, "Distributed wireless sensor network localization via sequential greedy optimization algorithm," *Signal Processing, IEEE Transactions on*, vol. 58, no. 6, pp. 3328 – 3340, jun. 2010.

[2] A. Simonetto and G. Leus, "Distributed maximum likelihood sensor network localization," *Signal Processing, IEEE Transactions on*, vol. 62, no. 6, pp. 1424–1437, March 2014.

[3] P. Oguz-Ekim, J. Gomes, J. Xavier, and P. Oliveira, "Robust localization of nodes and time-recursive tracking in sensor networks using noisy range measurements," *Signal Processing, IEEE Transactions on*, vol. 59, no. 8, pp. 3930 –3942, aug. 2011.

[4] M. Gholami, L. Tetruashvili, E. Strom, and Y. Censor, "Cooperative wireless sensor network positioning via implicit convex feasibility," *Signal Processing, IEEE Transactions on*, vol. 61, no. 23, pp. 5830–5840, Dec 2013.

[5] G. Destino and G. Abreu, "On the maximum likelihood approach for source and network localization," *Signal Processing, IEEE Transactions on*, vol. 59, no. 10, pp. 4954 –4970, oct. 2011.

[6] Y. Keller and Y. Gur, "A diffusion approach to network localization," *Signal Processing, IEEE Transactions on*, vol. 59, no. 6, pp. 2642 –2654, jun. 2011.

[7] Y. Shang, W. Rumi, Y. Zhang, and M. Fromherz, "Localization from connectivity in sensor networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 15, no. 11, pp. 961 – 974, nov. 2004.

[8] P. Biswas, T.-C. Liang, K.-C. Toh, Y. Ye, and T.-C. Wang, "Semidefinite programming approaches for sensor network localization with noisy distance measurements," *Automation Science and Engineering, IEEE Transactions on*, vol. 3, no. 4, pp. 360 –371, oct. 2006.

[9] S. Korkmaz and A.-J. van der Veen, "Robust localization in sensor networks with iterative majorization techniques," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*, apr. 2009, pp. 2049 –2052.

[10] J. Costa, N. Patwari, and A. Hero III, "Distributed weighted-multidimensional scaling for node localization in sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 2, no. 1, pp. 39–64, 2006.

[11] G. Calafiore, L. Carlone, and M. Wei, "Distributed optimization techniques for range localization in networked systems," in *Decision and Control (CDC), 2010 49th IEEE Conference on*, Dec 2010, pp. 2221–2226.

[12] S. Srirangarajan, A. Tewfik, and Z.-Q. Luo, "Distributed sensor network localization using SOCP relaxation," *Wireless Communications, IEEE Transactions on*, vol. 7, no. 12, pp. 4886 –4895, dec. 2008.

[13] F. Chan and H. So, "Accurate distributed range-based positioning algorithm for wireless sensor networks," *Signal Processing, IEEE Transactions on*, vol. 57, no. 10, pp. 4100 –4105, oct. 2009.

[14] U. Khan, S. Kar, and J. Moura, "DILAND: An algorithm for distributed sensor localization with noisy distance measurements," *Signal Processing, IEEE Transactions on*, vol. 58, no. 3, pp. 1940 –1947, mar. 2010.

[15] D. Blatt and A. Hero, "Energy-based sensor network source localization via projection onto convex sets," *Signal Processing, IEEE Transactions on*, vol. 54, no. 9, pp. 3614–3619, Sept 2006.

[16] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms*. Springer-Verlag Limited, 1993.

[17] F. R. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[18] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[19] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o (1/k2)," in *Soviet Mathematics Doklady*, vol. 27, no. 2, 1983, pp. 372–376.

[20] D. Shah, *Gossip algorithms*. Now Publishers Inc, 2009.

[21] M. Udell and S. Boyd, "Bounding duality gap for problems with separable objective," ONLINE, 2014.
[22] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Kluwer Academic Publishers, 2004.
[23] Z. Lu and L. Xiao, "On the complexity analysis of randomized block-coordinate descent methods," *arXiv preprint arXiv:1305.4723*, 2013.
[24] J. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11–12, pp. 625–653, 1999, version 1.05 available from `http://fewcal.kub.nl/sturm`.
[25] J. Jacod and P. Protter, *Probability Essentials*. Springer, 2003, vol. 1.
[26] H. Robbins and D. Siegmund, "A convergence theorem for non negative almost supermartingales and some applications," in *Herbert Robbins Selected Papers*. Springer, 1985, pp. 111–135.