

# A Simple Peer-to-Peer Algorithm for Distributed Optimization in Sensor Networks

Björn Johansson, Maben Rabi and Mikael Johansson

**Abstract**—We propose a distributed algorithm that solves a special class of optimization problems using only peer-to-peer communication. One application is parameter estimation problems in sensor networks. Current decentralized algorithms for solving this class of optimization problems typically rely on passing around a parameter estimate in a ring consisting of all network nodes. In our algorithm, which extends the randomized incremental subgradient method with fixed stepsize due to Nedić and Bertsekas, nodes maintain individual estimates and need to exchange information only with their neighbors. We establish approach of the solution to an interval around the optimum value. We illustrate the algorithm's performance, in terms of convergence rate and communication cost relative to alternative schemes, through several numerical examples.

## I. INTRODUCTION

Many tasks of wireless sensor networks, including estimation, detection, localization, and resource sharing, can be cast as optimization problems (see, for example, [7] and [3]). To solve these problem optimally without relying on a single node (hence increasing system robustness) and minimizing the energy consumption needed for signalling and coordination, there is a desire to devise decentralized algorithms that can provide this functionality in sensor networks. This paper proposes an optimization approach that solves a rather general optimization problem, applicable to many of the tasks outlined above, relying only on peer-to-peer communication.

### A. Problem Formulation

We consider the following specific class of optimization problems

$$\begin{aligned} & \underset{\theta, \mathbf{x}_1, \dots, \mathbf{x}_N}{\text{minimize}} && \sum_{i=1}^N f_i(\mathbf{x}_i, \theta) \\ & \text{subject to} && \mathbf{x}_i \in \mathcal{X}_i, \quad i = 1, \dots, N \\ & && \theta \in \Theta. \end{aligned} \quad (1)$$

Here  $f_i(\mathbf{x}_i, \theta)$  is a cost function associated with node  $i$ ,  $\mathbf{x}_i$  is a vector of variables local to node  $i$ , and  $\mathcal{X}_i$  is the feasible set for the local variables. The set  $\Theta$  is the feasible set of a global (network-wide) decision variable  $\theta$ . Although our algorithms are readily extended to the vector case, we will assume that  $\theta$  is scalar to simplify notation. We assume that  $f_i$  are convex functions and that  $\mathcal{X}_i$  and  $\Theta$  are convex sets<sup>1</sup> with non-empty interior. Associated to the problem is a communication topology represented by a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

with vertex set  $\mathcal{V} = \{1, \dots, N\}$  and edge set  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ . The presence of an edge  $(i, j)$  in  $\mathcal{E}$  means that node  $i$  can communicate directly with node  $j$ , and vice versa. The graph  $\mathcal{G}$  has  $N$  nodes (vertices).

Although the communication structure is dictated by  $\mathcal{G}$ , one can structure the inter-node communications for computing the optimizer  $(\mathbf{x}, \theta)$  in many ways. The most obvious way is a *centralized* approach where nodes pass information about  $(f_i, \mathcal{X}_i)$  to a central node that solves the convex optimization problem (1) using, for example the techniques in [3], and distributes the solution to the nodes. This solution is sensitive to the failure of a single node (the central node), requires individual nodes to reveal their cost functions and constraint sets, and demands potentially complex information transfer from nodes to the computational unit. Another alternative uses *estimate passing*: the nodes are organized in a ring, and the current estimate of the optimizer is passed from one node to the other [7]. There also exists a randomized version where the estimate is sent to a randomly chosen node in the network [1]. When a node receives the current estimate, it performs a local update accounting for its own cost function and constraints, before passing the modified estimate to the next node. In this approach, nodes do not need to reveal their private objectives and constraints, but the method requires reliable communications (typically over multiple hops) as the messages conveying the current estimate must not be lost.

Our approach is an extension of the randomized version of the estimate passing scheme. Instead of nodes sending the estimate to a randomly chosen node in the *whole network*, we show that it is sufficient to send to a *random neighbor*. Thus, our scheme can be implemented without relying on complex and energy consuming multi-hop communications. The transition probabilities, which determine how the parameter estimate is passed around in the network, can be computed using local network topology characteristics.

### B. Related Work

A similar problem formulation for sensor network applications is considered by [7] and [8]. In [7], the incremental subgradient method (see for example [1]) is applied to the primal problem formulation. The result is an algorithm where a parameter estimate is circulated in the network, and each

<sup>1</sup>The optimization problem is especially easily solved if the sets are described by upper and lower bounds or by quadratic constraints. If the sets are more complicated, the algorithm we develop still works, but the computational burden on each node will significantly increase.

This research was partially funded by the Swedish Research Council, the Swedish Foundation for Innovation Systems, and the European Commission. The authors are with the School of Electrical Engineering, Royal Institute of Technology (KTH), 100 44 Stockholm, Sweden. Email: `firstname.lastname@ee.kth.se`.

node refines the estimate. This approach is extended in [8], into an algorithm that exploits the clustered structure of many sensor networks: within each cluster, the primal incremental subgradient method is used to reach consensus of the optimal parameter estimate, given the data in the cluster. The clusters then fuse their estimates, and the optimization within each cluster is repeated. This scheme is iterated until convergence is reached. A related problem formulation for the case of distributed least-squares estimation in sensor networks is investigated in [9]. In that paper, a peer-to-peer approach is combined with consensus algorithms.

Subgradient algorithms with randomization and diminishing stepsizes (also called stochastic quasi-gradient methods) are well-known in stochastic optimization theory [4]. However, our approach is an extension of the less known randomized incremental subgradient algorithm with *fixed* stepsize [1], [6].

### C. Outline

We start with presenting our algorithm and proving its convergence properties in Section II. We then describe some extensions to tackle practical issues in Section III. In Section IV, we focus on the specific problem of parameter estimation via Huber norm minimization. Our approach is compared with other algorithms. Finally, the paper is concluded with Section V.

## II. PEER-TO-PEER OPTIMIZATION ALGORITHM

We rewrite (1) as

$$\begin{aligned} & \underset{\theta}{\text{minimize}} && \sum_{i=1}^N q_i(\theta) \\ & \text{subject to} && \theta \in \Theta, \end{aligned} \quad (2)$$

where  $q_i(\theta) = \min_{\mathbf{x}_i \in \mathcal{X}_i} f_i(\mathbf{x}_i, \theta)$  and we introduce  $q^*$  as the optimal value of (2) and  $q(\theta) = \sum_{i=1}^N q_i(\theta)$ . Let each (convex) component  $q_i(\cdot)$  have a subgradient  $g_i(\theta)$  at  $\theta$  and assume that each of these subgradients are bounded as follows

$$|g_i(\theta)| \leq C \text{ for all } \theta \in \Theta \text{ and all } i = 1, \dots, N.$$

We now develop an extension of the randomized incremental subgradient method (RISM) [1]. The update equation of the estimate,  $\theta_k$ , of the optimizer is

$$\theta_{k+1} = \mathcal{P}_{\Theta} \{ \theta_k - \alpha g_{w_k}(\theta_k) \}, \quad (3)$$

where  $\mathcal{P}_{\Theta}\{\cdot\}$  denotes projection on the set  $\Theta$  and  $\alpha > 0$  is a fixed stepsize. In the standard RISM, the random variable  $w_k$  is IID and takes on values from the set  $\{1, \dots, N\}$  with equal probability. This means that the estimate is passed around between randomly chosen nodes in the network, and successive estimates could be updated by nodes possibly several hops away. Intuitively, we should save energy spent on communication if the estimate is sent to a neighboring node instead, and this is precisely what we can do when we let  $w_k$  be the state of a Markov chain corresponding to the communication structure. Thus, in our extension,  $w_k$  is the state of a special Markov chain that takes on values in the finite set  $\{1, 2, \dots, N\}$  and has the transition matrix  $P$ .

### Algorithm 1 Peer-to-peer optimization algorithm, MRISM.

- 1: Initialize  $\theta_0$  and  $\alpha$ . Set  $k := 0$  and  $w_k = 1$ .
- 2: **loop**
- 3: At node  $w_k$ , compute  $g_{w_k}(\theta_k)$  for  $q_{w_k}(\theta_k)$ .
- 4:  $\theta_{k+1} := \mathcal{P}_{\Theta}\{\theta_k - \alpha g_{w_k}(\theta_k)\}$ .
- 5: Send  $\theta_{k+1}$  to a random neighbor,  $w_{k+1}$ , with transition probability according to  $P$ .
- 6:  $k := k + 1$ .
- 7: **end loop**

The speciality of the Markov chain lies in the fact that its transition matrix respects the communication topology of the graph  $\mathcal{G}$ .

We assume that the Markov chain is irreducible, acyclic, and has the uniform distribution as its stationary distribution. This assumption means that the underlying communication structure is connected; after a sufficient amount of time the chain can be in any state; and that all states in the chain will be visited the same number of times in the long run. Now the question is how to construct the transition matrix using only local information. It turns out there is a simple way to do this using the so called Metropolis-Hastings scheme [2]. If the underlying communication topology is connected, then all assumptions on the transition matrix are valid if the elements are set to

$$[P]_{ij} = \begin{cases} \min\{\frac{1}{d_i}, \frac{1}{d_j}\} & \text{if } (i, j) \in \mathcal{E} \text{ and } i \neq j \\ \sum_{(i,k) \in \mathcal{E}} \max\{0, \frac{1}{d_i} - \frac{1}{d_k}\} & \text{if } i = j \\ 0 & \text{otherwise,} \end{cases} \quad (4)$$

where  $d_i$  is node  $i$ 's number of edges.

To summarize, we make the following assumptions on the optimization problem and the Markov chain.

*Assumption 1:* a) The functions  $f_i(\cdot)$  are convex, and the sets  $\mathcal{X}_i$  and  $\Theta$  are convex with non-empty interior. b) The subgradients are bounded as follows,  $|g_i(\theta)| \leq C$  for all  $\theta \in \Theta$  and all  $i = 1, \dots, N$ . c) The Markov chain corresponding to the graph  $\mathcal{G}$  is irreducible, acyclic, and has the uniform distribution as its stationary distribution.

The peer-to-peer algorithm is described in pseudocode in Algorithm 1 (MRISM, Markov randomized incremental subgradient method). We now proceed to show its convergence. The main idea of the proof is that we can look at the estimate in one node (state). Since the Markov chain is recurrent and has the uniform distribution as its stationary distribution, the algorithm will return to this node and we can look at what "happens" during such an excursion on average.

### A. Preliminaries

To show convergence we need three lemmas. The first lemma concerns the average number of visits to other nodes over a return time.

Without loss of generality, we let  $w_k = 1$  and let  $R_k(1)$  be the random time (the return time) it takes for the Markov

chain to return to state 1 for the first time, i.e.,

$$R_k(1) = \inf_t \{t - k | w_k = 1, w_t = 1, t \geq k + 1\}.$$

Successive first return times to a state form an IID sequence, because of the Markov property, and we note that the statistics of  $R_k(1)$  will not depend on  $k$ . Furthermore, we let  $v_k(j)$  be the random number of visits to state  $j$  during the time interval  $[k + 1, k + R_k(1)]$ .

*Lemma 2:* Under Assumption 1, we have that

$$(\mathbb{E}[v_k(1)] \quad \dots \quad \mathbb{E}[v_k(N)]) = \mathbf{1}_N^T.$$

*Proof:* Let the Markov chain have the following transition matrix

$$P = \begin{pmatrix} p_{11} & P_{12} \\ P_{21} & Q \end{pmatrix},$$

where  $p_{11}$  is the probability of staying in state 1. Due to the definition of  $R_k(1)$  and  $v_k(\cdot)$ , it follows that  $v_k(1) = 1$ . Now consider the Markov chain with transition matrix

$$P' = \begin{pmatrix} 1 & 0 \\ P_{21} & Q \end{pmatrix}.$$

This new chain has the same state space as the original and has the first row of the transition modified from  $(p_{11} \quad P_{12})$  to  $(1 \quad 0 \quad \dots \quad 0)$ . This makes state 1 absorbent and every other state transient in the new Markov chain. If the absorbing chain is started in a transient state  $m$ , then the expected number of visits to transient state  $n$  (including the starting position),  $[Z]_{mn}$ , is given by (the  $(N - 1) \times (N - 1)$  matrix) [5, Theorem 3.2.4]

$$Z = (I - Q)^{-1}.$$

If we now consider the original Markov chain with transition matrix  $P$ , then if we start in state 1, the expected number of visits to the other states before returning to state 1 are given by

$$(\mathbb{E}[v_k(2)] \quad \dots \quad \mathbb{E}[v_k(N)]) = P_{12}Z.$$

Thus, we have  $(\mathbb{E}[v_k(1)] \quad \dots \quad \mathbb{E}[v_k(N)]) = (1 \quad P_{12}Z)$ . It turns out that the vector of expected visits is an eigenvector to  $P$  since

$$\begin{aligned} (1 \quad P_{12}Z)P &= (1 \quad P_{12}Z) \begin{pmatrix} p_{11} & P_{12} \\ P_{21} & Q \end{pmatrix} \\ &= (p_{11} + P_{12}ZP_{21} \quad P_{12}Z(I - Q + Q)) \\ &= (1 \quad P_{12}Z), \end{aligned}$$

where the last step follows from  $ZP_{21} = \mathbf{1}_{N-1}$  [5, Theorem 3.3.7], where  $\mathbf{1}_{N-1}$  denotes the  $N - 1$  column vector with all entries equal to one. The transition matrix  $P$  has only one eigenvector with eigenvalue 1, namely the invariant distribution [5, Theorem 4.1.6]. Since  $P$  is assumed to have a uniform stationary distribution, we have desired result. ■

The second lemma concerns the second moments of the return times,  $R_k(i) = \inf_t \{t - k | w_k = i, w_t = i, t \geq k + 1\}$ .

*Lemma 3* ([5, Theorem 4.5.2]): Under Assumption 1, the second moments of the return times  $R_k(i)$  are finite and are given as

$$\mathbb{E}[R_k^2(i)] = -\frac{1}{N} + \frac{2[\Gamma]_{ii}}{N^2},$$

with  $\Gamma = (I - P + \lim_{n \rightarrow \infty} P^n)^{-1}$ .

The last lemma concerns a bounding inequality that we will use in the main proof.

*Lemma 4:* Under Assumption 1, the sequence  $\{\theta_l\}_{l=k}^{\infty}$  generated by Algorithm 1 fulfills

$$\mathbb{E} \left[ |\theta_{k+R_k(i)} - y|^2 | \theta_k, w_k \right] \leq |\theta_k - y|^2 - 2\alpha(q(\theta_k) - q(y)) + \alpha^2 C^2 K, \quad (5)$$

with  $K < \infty$ .

*Proof:* Without loss of generality, we focus on the case where  $i = 1$ . Using the definition of a subgradient and that the subgradients are assumed to be bounded, we have for any  $y \in \Theta$

$$\begin{aligned} |\theta_{k+1} - y|^2 &\leq |\theta_k - y|^2 - 2\alpha g_{w_k}(\theta_k)(\theta_k - y) + \alpha^2 C^2 \\ &\leq |\theta_k - y|^2 - 2\alpha(q_{w_k}(\theta_k) - q_{w_k}(y)) + \alpha^2 C^2. \end{aligned}$$

Along the same lines of reasoning, we get the family of inequalities

$$\begin{aligned} |\theta_{k+1} - y|^2 &\leq |\theta_k - y|^2 - 2\alpha(q_{w_k}(\theta_k) - q_{w_k}(y)) + \alpha^2 C^2, \\ |\theta_{k+2} - y|^2 &\leq |\theta_{k+1} - y|^2 \\ &\quad - 2\alpha(q_{w_{k+1}}(\theta_{k+1}) - q_{w_{k+1}}(y)) + \alpha^2 C^2, \end{aligned}$$

and so forth up to

$$\begin{aligned} |\theta_{k+R_k(1)} - y|^2 &\leq |\theta_{k+R_k(1)-1} - y|^2 \\ - 2\alpha(q_{w_{k+R_k(1)-1}}(\theta_{k+R_k(1)-1}) - q_{w_{k+R_k(1)-1}}(y)) &+ \alpha^2 C^2. \end{aligned}$$

Combining all of them together we get

$$\begin{aligned} |\theta_{k+R_k(1)} - y|^2 &\leq |\theta_k - y|^2 \\ - 2\alpha \sum_{j=0}^{R_k(1)-1} (q_{w_{k+j}}(\theta_{k+j}) - q_{w_{k+j}}(y)) &+ R_k(1)\alpha^2 C^2, \end{aligned}$$

which can be rewritten as

$$\begin{aligned} |\theta_{k+R_k(1)} - y|^2 &\leq |\theta_k - y|^2 \\ - 2\alpha \sum_{j=0}^{R_k(1)-1} (q_{w_{k+j}}(\theta_{k+j}) - q_{w_{k+j}}(\theta_k)) & \\ - 2\alpha \sum_{j=0}^{R_k(1)-1} (q_{w_{k+j}}(\theta_k) - q_{w_{k+j}}(y)) &+ R_k(1)\alpha^2 C^2. \quad (6) \end{aligned}$$

Notice that

$$\begin{aligned} q_{w_{k+j}}(\theta_k) - q_{w_{k+j}}(\theta_{k+j}) &\leq |g_{w_{k+j}}(\theta_k)| \times |\theta_{k+j} - \theta_k| \\ &\leq C |\theta_{k+j} - \theta_k| \\ &\leq \alpha j C^2. \end{aligned}$$

This enables us to rewrite inequality (6) as follows

$$\begin{aligned} |\theta_{k+R_k(1)} - y|^2 &\leq |\theta_k - y|^2 \\ - 2\alpha \sum_{j=0}^{R_k(1)-1} (q_{w_{k+j}}(\theta_k) - q_{w_{k+j}}(y)) &+ \alpha^2 C^2 R_k^2(1). \end{aligned}$$

Using  $v_k(j)$  as defined in Lemma 2, we express (6) as

$$\begin{aligned} |\theta_{k+R_k(1)} - y|^2 &\leq |\theta_k - y|^2 \\ &\quad - 2\alpha \sum_{j=1}^N v_k(j) (q_j(\theta_k) - q_j(y)) + \alpha^2 C^2 R_k^2(1). \end{aligned} \quad (7)$$

Now, given only the history  $\theta_k$  and  $w_k$  we have

$$\begin{aligned} \mathbb{E} \left[ \sum_{j=1}^N v_k(j) (q_j(\theta_k) - q_j(y)) \middle| \theta_k, w_k \right] &= \\ \sum_{j=1}^N \mathbb{E}[v_k(j)] (q_j(\theta_k) - q_j(y)) &= q_j(\theta_k) - q_j(y), \end{aligned}$$

due to the Markov property and Lemma 2. Now let

$$K = \max_i \mathbb{E} [R_k^2(i)],$$

using Lemma 3. If we combine the above parts, we have the desired result. ■

### B. Convergence

We cannot guarantee convergence for Algorithm 1, but we can guarantee approach to the optimum value of the function to within a computable error. Fix  $i$  to be a state of the Markov chain  $\{w_k\}$ . We want to show that it is possible to describe how much closer the  $\theta$ -sequence gets to the optimum set on average, every time the chain returns to the state  $i$ . This is a meaningful approach since the time horizon can be counted in terms of successive random return times to the special state  $i$ .

*Theorem 5:* Let  $\{\theta_k\}$  be generated by Algorithm 1. Under Assumption 1 and with probability 1, the following holds for all  $i = 1, \dots, N$ . Let the subsequence  $\{\theta_k^i\}$  of  $\{\theta_k\}$  be formed by sampling  $\{\theta_k\}$  whenever  $\{w_k\}$  visits the state  $i$ . This subsequence fulfills

$$\begin{cases} \inf_{k \geq 0} q(\theta_k^i) = q^*, & \text{if } q^* = -\infty \\ \inf_{k \geq 0} q(\theta_k^i) \leq q^* + \frac{\alpha C^2 K}{2}, & \text{if } q^* > -\infty. \end{cases}$$

*Proof:* Without loss of generality, we can assume that at time zero, the chain is in state  $i$ . With probability 1, all states are visited infinitely often since the stationary distribution is uniform, and thus we can form the subsequence  $\{\theta_k^i\}$  of  $\{\theta_k\}$  by sampling it whenever  $\{w_k\}$  visits the state  $i$ .

Now the proof proceeds similarly as the proof of Proposition 3.1 in [6]. Given a positive integer  $M$ , let  $y_M \in \Theta$  be such that

$$q(y_M) = \begin{cases} -M, & \text{if } q^* = -\infty \\ q^* + \frac{1}{M}, & \text{if } q^* > -\infty. \end{cases}$$

Consider the level set  $L_M$  defined by

$$L_M = \left\{ \theta \in \Theta \middle| q(\theta) \leq q(y_M) + \frac{1}{M} + \frac{\alpha C^2 K}{2} \right\}.$$

This set includes  $y_M$ . We now derive a new sequence from  $\{\theta_k^i\}$ . Define the sequence  $\{s_k\}$  as follows

$$s_k = \begin{cases} \theta_k^i & \text{if } \theta_k^i \notin L_M \quad \forall j \leq k \\ y_M & \text{otherwise.} \end{cases}$$

When  $s_k \notin L_M$ , by setting  $y = y_M$  in (5), we get

$$\begin{aligned} \mathbb{E} \left[ |s_{k+1} - y_M|^2 \middle| \{s_j\}_0^k, \{w_j\}_0^k \right] &\leq |s_k - y_M|^2 \\ &\quad + \alpha^2 C^2 K - 2\alpha (q(s_k) - q(y_M)). \end{aligned}$$

On the other hand, whenever  $s_k \in L_M$ , the sequence is forced to stay at  $y_M$ , and we have the trivial inequality

$$\mathbb{E} \left[ |s_{k+1} - y_M|^2 \middle| \{s_j\}_0^k, \{w_j\}_0^k \right] \leq |s_k - y_M|^2 + 0.$$

If we define  $z_k$  through

$$z_k = \begin{cases} 2\alpha (q(s_k) - q(y_M)) - \alpha^2 C^2 K & \text{if } s_k \notin L_M \\ 0 & \text{if } s_k \in L_M. \end{cases}$$

we can write

$$\mathbb{E} \left[ |s_{k+1} - y_M|^2 \middle| \{s_j\}_0^k, \{w_j\}_0^k \right] \leq |s_k - y_M|^2 - z_k, \quad \forall k.$$

When  $s_k \notin L_M$ , we have

$$(q(s_k) - q(y_M)) \geq \frac{1}{M} + \frac{\alpha C^2 K}{2},$$

which is equivalent to

$$2\alpha (q(s_k) - q(y_M)) - \alpha^2 C^2 K \geq \frac{2\alpha}{M}.$$

Hence,  $z_k \geq 0 \quad \forall k$ , and by the Supermartingale Convergence theorem [1, Proposition 8.2.10], we have with probability 1 that  $\sum_{k=0}^{\infty} z_k < \infty$ , which means that with probability 1,  $z_k = 0$  for all  $k$  greater than some finite number. This is true because the series  $\frac{2\alpha}{M} \sum_{k=0}^{\infty} 1$  is divergent.

This means that with probability 1, the sequence  $\{s_k\}$  equals  $y_M$  starting from some finite value of  $k$ . Thus,

$$\inf_{k \geq 0} q(\theta_k^i) \leq \begin{cases} -M + \frac{1}{M} + \frac{\alpha C^2 K}{2}, & \text{if } q^* = -\infty \\ q^* + \frac{1}{M} + \frac{\alpha C^2 K}{2}, & \text{if } q^* > -\infty. \end{cases}$$

By letting  $M$  go to  $\infty$ , we are able to prove the theorem. ■

*Corollary 6:* With probability 1, the following holds for the sequence  $\{\theta_k\}$  generated by Algorithm 1

$$\begin{cases} \inf_{k \geq 0} q(\theta_k) = q^*, & \text{if } q^* = -\infty \\ \inf_{k \geq 0} q(\theta_k) \leq q^* + \frac{\alpha C^2 K}{2}, & \text{if } q^* > -\infty. \end{cases}$$

This corollary is valid because the infimum over a subsequence cannot be lower than that over the original sequence.

### III. EXTENSIONS

According to Theorem 5, the best estimate at each node will be in a interval around the optimal value. However, it is impossible to find out which the best estimate is in a decentralized way. A pragmatic solution is to use the estimate last computed by the node,

$$\begin{cases} \phi_{k+1}(j) = \theta_{k+1} & \text{if } w_{k+1} = j \\ \phi_{k+1}(j) = \phi_k(j) & \text{otherwise,} \end{cases}$$

where  $\phi_k(j)$  is the local estimate of the optimizer for node  $j$  at time  $k$ . However, there are two problems with this approach: oscillations and bias.

As we will see in the numerical examples, Section IV, the local estimates in each node usually oscillate. One way to mitigate this problem is to form the local estimate,  $\phi_k(j)$ , in each node by taking the mean value of the estimates that have passed the node. However, if we desire to track a changing parameter, then the mean should be taken over a fixed window.

The second problem, bias, occurs if the network is large and sparsely connected. Then the estimate will take a long time to traverse the network, and the estimate could get stuck in one area. This means that some nodes will not receive any updates and the estimates will be biased. A simple way to tackle this predicament is to pass around several estimates. More precisely, we execute several independent versions of MRISM in parallel. The local estimate can be formed by taking the arithmetic mean of a fixed number of the most recent estimates that pass through the node. This average is computed at a node without regard to which instance of the MRISM algorithm that produced the estimates. This will hasten convergence and promote exploration of the network. However, the energy consumption will also increase.

A potential threat to the algorithm is that estimates can be lost while in transit. Due to the decentralized nature of the algorithm, it is impossible to know if the estimate is lost or just being passed around in a distant part of the network. This problem can be solved using the following scheme: If a node has not received an estimate within a fixed time frame, then it initiates a new estimate, for example its own local estimate, and starts passing it around.

#### IV. NUMERICAL EXAMPLES

We investigate the performance of Algorithm 1 (MRISM) and its extension to multiple estimates (MMRISM, multiple-MRISM) by comparing them with existing algorithms in two sensor networks with different topologies. In MMRISM, we use three estimates and take the average of the 10 last estimates that has passed each node as the local estimate. We compare with the RISM and the deterministic incremental subgradient method (ISM).

The objective is to solve the optimization problem (2) with  $q_i(\theta) = \sum_{j=1}^{10} h(\theta - y_{ij})$ , where  $y_{ij}$  is the  $j$ :th measurement for node  $i$  and  $h(\cdot)$  is the Huber norm, which is an alternative to the Euclidean norm. Seeking an estimate that minimizes the aggregate Huber norm of deviations from data results in less sensitivity to outliers in the data compared to that for the Euclidean norm. The Huber norm is a differentiable convex function and is defined as follows

$$h(\theta) = \begin{cases} \frac{1}{2}\theta^2, & |\theta| < \beta \\ \beta(|\theta| - \frac{\beta}{2}), & |\theta| \geq \beta, \end{cases}$$

where  $\beta$  is a positive tuning parameter. This implies that the subgradient (gradient in this case) is bounded as follows,  $|\nabla q_i(\theta)| \leq 10\beta$ . We assume that each node have 10 measurements according to  $y_{ij} = 10 + 10e_{ij}$ , where  $\{e_{ij}\}$  is an

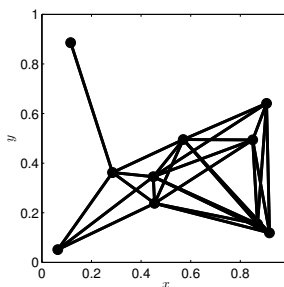


Fig. 1. Topology of the random 10 node network.

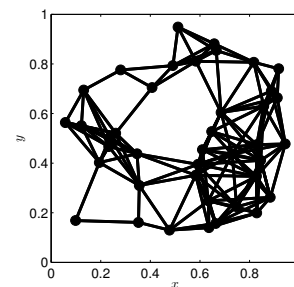


Fig. 2. Topology of the random 40 node network.

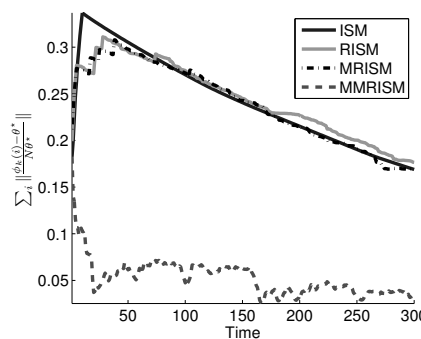


Fig. 3. Performance in the random 10 node network.

IID collection of Gaussian random variables with zero mean and variance 1. Each node also has one outlier, which is a Gaussian random variable with zero mean and variance 50, in its data. The stepsize is set to  $\alpha = .05$ .

We use the average normalized deviation from an optimizer  $\sum_{i=1}^N \|\frac{\phi_k(i) - \theta^*}{N\theta^*}\|$  as performance metric. The energy consumption is assumed to be proportional to the distance<sup>2</sup> the estimates have travelled when passed around in the network.

We investigate the performance for 2 random networks with 10 and 40 nodes, respectively. The nodes are randomly

<sup>2</sup>Note that this is an optimistic measure of communication cost, as in most wireless networks the required energy expenditure increases as distance <sup>$\alpha$</sup>  for  $\alpha \in [2, 4]$ .

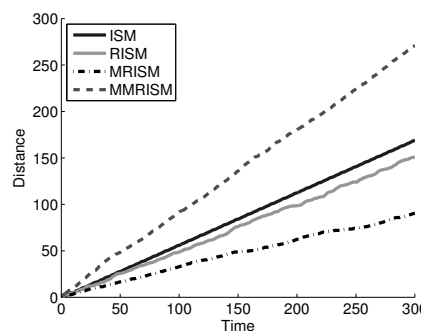


Fig. 4. Energy consumption in the random 10 node network.

TABLE I

THE MEAN AND VARIANCE OF THE AVERAGE PERFORMANCE (AP) AND THE ENERGY CONSUMPTION (EC) IN 1000 REALIZATIONS FOR THE 10 NODE RANDOM NETWORK.

	ISM	RISM	MRISM	MMRISM
Mean(AP)	0.243	0.244	0.244	0.0808
Variance(AP)	-	0.0000147	0.0000225	0.000876
Mean(EC)	169.0	154.0	88.9	266.0
Variance(EC)	-	48.7	31.7	91.1

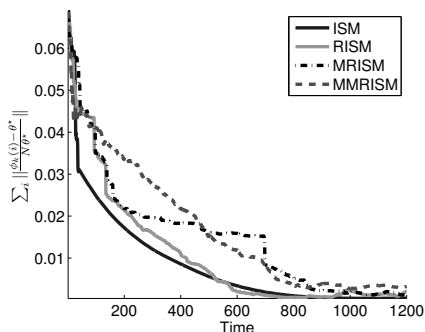


Fig. 5. Performance in the random 40 node network.

placed in a  $1 \times 1$  area, and only nodes that are at most radius  $r$  apart can communicate with each other. The radius  $r$  is increased from zero until the network is connected.

The topology, performance, and energy consumption of the 10 node network is shown in Fig. 1, Fig. 3, and Fig. 4, respectively. The performance plot suggests that the MMRISM has the best performance and the others are tied. However, it turns out that the performance is highly dependent on the realization. In order to capture the average efficiency, 1000 realizations were simulated and the average performance and energy consumption for each of these realizations were saved. Now we know how the average performance and energy consumption depend on the realization. To get a compact measure, the average and variance of these time series were evaluated, see Table I. The MMRISM has the best performance also in this average respect. The performance of the MRISM is still tied with the other two, but the MRISM has much lower energy consumption.

The topology, performance, and energy consumption of the 40 node network is shown in Fig. 2, Fig. 5, and Fig. 6, respectively. The performance plot suggests that the MRISM and MMRISM perform worse than the deterministic and stochastic. This is also verified in the 1000 realization average, presented in Table II. However, the MRISM and MMRISM both have significantly lower energy consumption.

## V. CONCLUSIONS

We propose a distributed algorithm, MRISM, that solves a special class of optimization problems using only peer-to-peer communication. Our algorithm extends the randomized

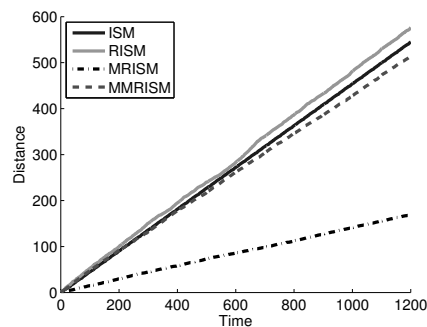


Fig. 6. Energy consumption in the random 40 node network.

TABLE II

THE MEAN AND VARIANCE OF THE AVERAGE PERFORMANCE (AP) AND THE ENERGY CONSUMPTION (EC) IN 1000 REALIZATIONS FOR THE 40 NODE RANDOM NETWORK.

	ISM	RISM	MRISM	MMRISM
Mean(AP)	0.00840	0.0103	0.0141	0.0155
Variance(AP)	-	0.00000126	0.00000532	0.0000386
Mean(EC)	545.0	574.0	168.0	505.0
Variance(EC)	-	82.9	13.8	40.4

incremental subgradient method with fixed stepsize due to Nedić and Bertsekas. The main merits of the proposed algorithm is its simplicity and energy efficiency through the strategy of one-hop estimate passing.

The numerical examples clearly show that the MRISM saves a significant amount of energy. The optimization performance of the MRISM is well on par with the other existing algorithms. We can also see that the use of multiple estimates, MMRISM, boosts initial convergence, and in some cases also the overall rate of convergence.

## ACKNOWLEDGMENTS

The authors wish to thank Yung Yi of Princeton University for helpful discussions.

## REFERENCES

- [1] D.P. Bertsekas, A. Nedić, and A.E. Ozdaglar. *Convex Analysis and Optimization*. Athena Scientific, 2003.
- [2] S. Boyd, P. Diaconis, and L. Xiao. Fastest mixing markov chain on a graph. *SIAM Review*, 46:667–689, 2004.
- [3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [4] P. Kall and S. W. Wallace. *Stochastic Programming*. John Wiley & Sons, 1994.
- [5] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Van Nostrand, 1960.
- [6] Angelia Nedić. *Subgradient Methods for Convex Minimization*. PhD thesis, MIT, 2002.
- [7] M. Rabbat and R. Nowak. Distributed optimization in sensor networks. In *IPSN*, 2004.
- [8] S.-H. Son, M. Chiang, S.R. Kulkarni, and S.C. Schwartz. The value of clustering in distributed estimation for sensor networks. In *IEEE Wirelesscom*, 2005.
- [9] L. Xiao, S. Boyd, and S. Lall. A space-time diffusion scheme for peer-to-peer least-squares estimation. In *IPSN*, 2006.