Quantized Incremental Algorithms for Distributed Optimization

Michael G. Rabbat and Robert D. Nowak

Abstract

Wireless sensor networks are capable of collecting an enormous amount of data. Often, the ultimate objective is to estimate a parameter or function from these data, and such estimators are typically the solution of an optimization problem (e.g., maximum likelihood, minimum mean squared error, or maximum a posteriori). This paper investigates a general class of distributed optimization algorithms for "in-network" data processing, aimed at reducing the amount of energy and bandwidth used for communication. Our intuition tells us that processing the data in-network should, in general, require less energy than transmitting all of the data to a fusion center. In this paper we address the questions: *When, in fact, does in-network processing use less energy, and how much energy is saved?* The proposed distributed algorithms are based on incremental optimization methods. A parameter estimate is circulated through the network, and along the way each node makes a small gradient descent-like adjustment to the estimate based only on its local data. Applying results from the theory of incremental subgradient optimization we find that the distributed algorithms converge to an approximate solution for a broad class of problems. We extend these results to the case where the optimization variable is quantized before being transmitted to the next node and find that quantization does not affect the rate of convergence. Bounds on the number of incremental steps required for a certain level of accuracy provide insight into the trade-off between estimation performance and communication overhead. Our main conclusion is that as the number of sensors in the network grows, in-network processing will always use less energy than a centralized algorithm while maintaining a desired level of accuracy.

Index Terms

Distributed algorithms, gradient methods, wireless sensor networks, energy-accuracy tradeoff.

I. Introduction

In many envisioned applications of wireless sensor networks, the ultimate objective is not the collection of "raw" data, but rather an estimate of certain environmental parameters or functions of interest (e.g., source locations, spatial distributions). One means of achieving this objective is to transmit all data to a central point for processing.

This work was sponsored by Applied Signal Technology, the National Science Foundation, grants CCR-0310889, CCR-0325571, and ANI-0099148, and the Office of Naval Research, grant N00014-00-1-0966.

Manuscript received December 2003, revised October 2004. Portions of this work were presented at IPSN'04. See [1].

The authors are with the Department of Electrical and Computer Engineering at the University of Wisconsin-Madison, 1415 Engineering Drive, Madison, WI, 53706. Email: rabbat@cae.wisc.edu, nowak@engr.wisc.edu.

However, transmitting data from each sensor node to a central processing location may place a significant drain on communication and energy resources. Such concerns could place undesirable limits on the practical application of sensor networks. This paper considers an alternate approach based on distributed *in-network* processing, in which the environmental parameters are computed in a decentralized fashion. While the focus of this paper is on applications in wireless sensor networks, as that was the motivation for this work, the quantized incremental subgradient algorithm developed and analyzed in this paper could find use in other arenas (e.g., a distributed Internet anomaly detection system). When data are distributed amongst a collection of networked nodes with limited communication or energy resources and the nodes must collaborate to solve a problem, distributed optimization methods can significantly decrease the communication and energy resources consumed.

As an illustration of the basic idea, consider a sensor network comprised of n nodes randomly distributed uniformly over the region $[0,1]^2$, each of which collects m measurements. Suppose, for example, that our objective is simply to compute the average value of all the measurements. There are three approaches one might consider:

1) Sensors transmit all the data to a central processor which then computes the average. In this approach, assuming a constant number of bits per sample, $O(mn)$ bits need to be transmitted over an average distance of length $O(1)$ per bit to reach the fusion center.
2) Sensors first compute a local average and then transmit the local averages to a central processor which computes the global average. This obvious improvement requires only $O(n)$ bits to be transmitted over an average distance of length $O(1)$ per bit to reach the fusion center.
3) Construct a path through the network which visits each node once. Assume the sequence of nodes can be constructed so that the path hops from neighbor to neighbor. Such sequences occur with high probability in large networks, and a method for finding one is discussed in Section VII. The global average can be computed by a single accumulation process from start node to finish, with each node adding its own local average to the total along the way. This requires $O(n)$ bits to be transmitted over an average distance of only $O(\sqrt{\log^2 n/n})$ per bit[1].

The third procedure makes much more efficient use of key resources — it requires far fewer communications than the former schemes and hence consumes less bandwidth and energy. Similar procedures could be employed to compute any average quantity (i.e., a least squares fit to a model with any number of parameters). Averages can be viewed as the values minimizing quadratic cost functions. Quadratic optimization problems are very special since their solutions are linear functions of the data, in which case an accumulation process leads to a solution.

More general optimization problems do not share this nice feature, but nevertheless can often be solved using simple, distributed algorithms reminiscent of the way the average was calculated above in the third approach. In particular, many estimation cost functions possess the following important factorization:

$$f(\theta; x) \quad = \quad \frac{1}{n} \sum_{i=1}^{n} f_i(\theta; x_i), \tag{1}$$

[1]This rather non-intuitive value is related to the transmission radius required to ensure that such a path through the network exists [2].

where θ is the parameter or function to be estimated, and $f(\theta; x)$ is a global cost function which can be expressed as a sum of n "local" cost functions, $\{f_i(\theta; x_i)\}_{i=1}^n$, in which $f_i(\theta; x_i)$ only depends on the data x_i measured at sensor i. For example, in the case of the sample average considered above, $f(\theta; x) = \frac{1}{n}\sum_{i=1}^n \sum_{j=1}^m (\theta - x_{i,j})^2$ is the cost function to be minimized, and $f_i(\theta; x_i) = \sum_{j=1}^m (\theta - x_{i,j})^2$, where $x_{i,j}$ is the jth measurement at the ith sensor.

The distributed algorithms proposed in this paper operate in a very simple manner. An estimate of the parameter θ is passed from node to node. Each node updates the parameter by adjusting the previous value to improve (i.e., reduce) its *local* cost and then passes the update to the next node. In the case of a quadratic cost function, one could construct a simple algorithm such as the one described in the third approach above which would solve the optimization after one pass through the network. For general cost functions, the algorithms are slightly more complex and several "cycles" through the network are required to obtain a solution. These distributed algorithms can be viewed as incremental subgradient optimization procedures, and the number of cycles required to obtain a good solution can be characterized theoretically. Roughly speaking, a typical result states that after K cycles, the distributed minimization procedure is guaranteed to produce an estimate $\widehat{\theta}$ satisfying $f(\widehat{\theta}) \leq f(\theta^*) + O(1/\sqrt{K})$, where θ^* is the minimizer of f. Also, the procedure only requires that a total of $O(nK)$ bits be communicated over an average distance of $O(\sqrt{\log^2 n/n})$ meters. This should be contrasted with transmitting all data to a central processor, which requires that $O(mn)$ bits be transmitted over an average of $O(1)$ meter. If m and n are large then a high quality estimate can be obtained using a distributed optimization algorithm for far less energy and far fewer communications than the centralized approach. Additionally, we analyze an incremental subgradient algorithm where the parameter estimates θ are quantized before they are transmitted between nodes. We find that while quantization marginally affects the estimate quality, the number of cycles K required for a certain performance does not change because of quantization.

The remainder of this paper is organized as follows. In the next section we formally state the problem and our assumptions. In Section III we analyze the algorithm described above using existing results from the theory of incremental subgradient optimization. These results are extended for incremental subgradient methods with quantized steps in Section IV. In Section V we derive and discuss the energy-accuracy tradeoff which arises between distributed incremental and centralized algorithms. Robust estimation is presented as an example application in Section VI. Some practical issues pertaining to incremental distributed algorithms are discussed in Section VII. Finally, we conclude in Section VIII.

II. Problem Statement and Assumptions

Before analyzing the proposed in-network algorithm, we introduce some notation and assumptions. Consider a sensor network comprised of n nodes. Each sensor, $i = 1, \ldots, n$, collects a set of measurements, x_i, which one can either think of as deterministic values or realizations of random variables for our purposes. The measurements are related to an unknown set of global parameters $\theta \in \Theta$ through the functions f_i. We would like to find a $\theta \in \Theta$

which minimizes

$$f(\theta; x) \quad = \quad \frac{1}{n} \sum_{i=1}^{n} f_i(\theta; x_i).$$

Note that each node's local objective function need not depend on all components of θ. Likewise, all functions f_i need not be of the same form.

Throughout this paper $\|\cdot\|$ denotes the Euclidean norm. In our analysis we assume that

(A1) the functions f_i are convex (but not necessarily differentiable) and there exists a constant C_0 such that for all $i = 1, \ldots, n$ and all $\theta \in \Theta$, the subgradient[2] $g \in \partial f_i(\theta)$ has magnitude bounded according to $\|g\| \leq C_0$,

(A2) the set $\Theta \subset \mathbb{R}^d$ is nonempty, convex, and compact with diameter $B = \sup_{\theta_1, \theta_2 \in \Theta} \|\theta_1 - \theta_2\|$, and

(A3) the optimal value $f^* \equiv \inf_{\theta \in \Theta} f(\theta; x) > -\infty$.

Additionally, note that since f is a convex function (the sum of convex functions is convex) and since Θ is a convex, compact subset of Euclidean space, the set of optimal solutions $\Theta^* \equiv \{\theta \in \Theta : f(\theta; x) = f^*\}$ is a non-empty compact, convex subset of Θ. With this setup, we are ready to define and analyze the algorithm.

III. Distributed Incremental Algorithms for Sensor Networks

Incremental methods have an established tradition in optimization theory, and we feel they are well suited for data processing applications in the context of networked systems. In this setup, a parameter estimate is cycled through the network. When each sensor receives the current estimate, it makes a small adjustment based on its local data and then passes the updated estimate on to one of its neighbors. Without loss of generality, assume that sensors have been numbered $i = 1, 2, \ldots, n$, with these numbers corresponding to their order in the cycle. Let $\mathcal{P}_\Theta : \mathbb{R}^d \to \Theta$ be an operator which projects its argument to the nearest point in Θ. Here, and in the remainder of this paper, when we refer to the "nearest point" in a set, we mean nearest in the Euclidean distance sense. Note that such an operator is well defined since Θ is compact and thus closed. On the kth cycle, sensor i receives an estimate $\psi_{i-1,k}$ from its predecessor and computes an update according to

$$\psi_{i,k} \quad = \quad \mathcal{P}_\Theta[\psi_{i-1,k} - \alpha g_i/n], \tag{2}$$

where α is a small positive scalar step size and $g_i \in \partial f_i(\psi_{i-1,k})$. Thus, the sensor makes an update based on the previous value of the estimate, $\psi_{i-1,k}$ received from its neighbor, and based on its local data reflected in g_i. After each complete cycle through the network we get the next iterate, $\theta_k \equiv \psi_{n,k} \equiv \psi_{0,k+1}$.

Such an algorithm fits the framework of incremental subgradient algorithms first studied by Kibardin [3], and more recently by Nedić and Bertsekas in [4], [5]. Because each step of the algorithm uses only local information, one cannot guarantee convergence in general. One condition under which convergence to the globally optimal value

[2]Subgradients generalize the notion of a gradient to non-differentiable functions. For a convex function $f(x)$, a subgradient of f at x_0 is any direction g such that $f(x) \geq f(x_0) + (x - x_0)^T g$ for all x. The set of subgradients of f at a point x is denoted $\partial f(x)$. At points x_0 where f is differentiable, the gradient of f is the only subgradient of f at x_0.

is guaranteed is when the step size gradually decreases to zero, however the rate of convergence is usually very slow in this case. Instead, we advocate the use of a fixed step size (positive constant). Although convergence is not guaranteed in this case, it has been observed that incremental algorithms perform well in practice. In general, the iterates quickly reach a neighborhood of the optimal value and then continue to move around within this neighborhood. However, in terms of rigorously analyzing their performance, the best we can hope to do is to characterize the limiting behavior as exemplified in the following theorem.

Theorem 1 (Nedić and Bertsekas, '99): Under assumptions (A1)-(A3), for the sequence $\{\theta_k\}$ generated by the incremental subgradient algorithm described above we have

$$\liminf_{k\to\infty} f(\theta_k) \quad \leq \quad f^* + \frac{\alpha C_0^2}{2}.$$

See [4] for the proof.

Additionally, we are interested in the rate at which the incremental algorithm reaches this limiting behavior since the amount of communication, and thus the amount of energy required for the algorithm to operate, is directly proportional to this value. Let $\mathrm{dist}(\theta_0, \Theta^*)$ denote the Euclidean distance between an arbitrary initial value $\theta_0 \in \Theta$ and the nearest point in Θ^*. The following result characterizes the rate at which the limiting behavior of the algorithm is reached.

Theorem 2 (Nedić and Bertsekas, '00): Under assumptions (A1)-(A3), for the sequence $\{\theta_k\}$ generated by the incremental subgradient algorithm described above and for any $\epsilon > 0$ we have

$$\min_{0\leq k\leq K} f(\theta_k) \quad \leq \quad f^* + \frac{\alpha C_0^2 + \epsilon}{2},$$

where K is given by

$$K \quad = \quad \left\lfloor \frac{\left(\mathrm{dist}(\theta_0, \Theta^*)\right)^2}{\alpha\epsilon} \right\rfloor.$$

See [5] for the proof.

The theorem above confirms that the iterates attain a value of arbitrary accuracy in a finite number of cycles. The theorem characterizes the distance between $f(\theta_k)$ and f^*. Setting $\epsilon = \alpha C_0^2$, it is clear that for α arbitrarily small we obtain $\min_{0\leq k\leq K} f(\theta_k)$ arbitrarily close to f^*. Here, however, there is a trade-off in that the number of cycles K (and thus the amount of communication) required is inversely related to the step size. More generally, this theorem tells us that the iterates $f(\theta_k)$ are guaranteed to reach a value within $O(\alpha)$ of f^* after $O(1/\alpha^2)$ cycles.

Alternatively, one could consider using a decreasing sequence of step sizes $\alpha_k \to 0$ as $k \to \infty$ (e.g. $\alpha_k \propto 1/k$) in which case convergence to f^* is guaranteed under very mild assumptions. However, while the decreasing step size approach may convergence to a neighborhood around the solution in a reasonable number of steps, the rate of convergence slows down dramatically as α_k gets small, and the overall convergence behavior is generally slower than that of a constant step size algorithm. In many applications of wireless sensor networks, acquiring a coarse estimate of the desired parameter or function may be an acceptable trade-off if the amount of energy and bandwidth used by the network is less than that required to achieve a more accurate estimate. Furthermore, many of the proposed

applications of wireless sensor networks involve deployment in a dynamic environment for the purpose of not only identifying but also tracking phenomena. For these reasons we advocate the use of a fixed step size.

IV. A Quantized Distributed Incremental Algorithm

We modify the incremental subgradient algorithm discussed above by quantizing each iterate in a very simple fashion as described next. For a given scalar $\Delta > 0$, consider the quantization lattice defined by

$$\Lambda \quad = \quad \{(\lambda_1\Delta, \lambda_2\Delta, \ldots, \lambda_d\Delta)^T : \lambda_i \in \mathbb{Z}, i = 1, \ldots, d\} \subseteq \mathbb{R}^d.$$

This lattice consists of points regularly spaced by Δ along each coordinate axis. Set $\Theta_q = \Theta \cap \Lambda$, and let $Q : \mathbb{R}^d \to \Theta_q$ be an operator which projects its argument first onto the set Θ and then onto the nearest lattice point in Θ_q. Thus $Q[\cdot] \equiv \mathcal{P}_{\Theta_q}[\mathcal{P}_\Theta[\cdot]]$. Note that applying the operator $Q[x]$ to $x \in \mathbb{R}^d$ is not equivalent to directly projecting x to the nearest point in Θ_q. In particular when $x \notin \Theta$, the nearest point to x in Θ_q can be different from $Q[x]$, and may result in a larger error. We will study the quantized incremental subgradient algorithm where, upon receiving $\vartheta_{i-1,k}$ from its neighbor, node i computes and transmits an incremental update according to

$$\vartheta_{i,k} \quad = \quad Q[\vartheta_{i-1,k} - \alpha g_{i,k}/n], \tag{3}$$

where $g_{i,k} \in \partial f_i(\vartheta_{i-1,k})$, and $\alpha > 0$ is, again, a small constant step size. After a complete cycle through the network we obtain the iterate

$$\theta_k \equiv \vartheta_{n,k} \equiv \vartheta_{0,k+1}. \tag{4}$$

Each incremental step of the quantized algorithm amounts to computing the usual incremental step of the unquantized algorithm and then projecting this new value to the nearest quantization lattice point in Θ_q. Thus, applying the operator Q enforces the constraint $\theta \in \Theta$ at each incremental step and also ensures that $\vartheta_{i,k}$ lies on a point in Θ_q so that it can be transmitted using a finite number of bits. Since Θ is bounded there are a finite number of lattice points in Θ_q, and thus any point in Θ_q can be coded with a finite number of bits. We have the following theorem which summarizes the limiting behavior of this algorithm.

Theorem 3: Under assumptions (A1)-(A3), for the sequence $\{\theta_k\}$ generated by the quantized incremental subgradient algorithm (3)-(4), we have

$$\liminf_{k\to\infty} f(\theta_k) \quad \leq \quad f^* + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha}, \tag{5}$$

where d is the dimension of the variables θ_k and $\vartheta_{i,k}$, and B is as defined in assumption (A2). Furthermore, for any $\epsilon > 0$ we have

$$\min_{0\leq k\leq K} f(\theta_k) \quad \leq \quad f^* + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B + \alpha\epsilon}{2\alpha}, \tag{6}$$

where

$$K \quad = \quad \left\lfloor \frac{\left(\mathrm{dist}(\theta_0, \Theta^*)\right)^2}{\alpha\epsilon} \right\rfloor, \tag{7}$$

and $\mathrm{dist}(\theta_0, \Theta^*)$ is the minimum Euclidean distance from θ_0 to a point in Θ^*.
A proof is presented in the appendix.

The error terms in the limiting expression can be classified as follows. There is an error of $\alpha C_0^2/2$ attributed to the localized, myopic nature of the incremental subgradient update step, which we also observe in the unquantized algorithm (c.f., Theorem 1). An additional error of

$$\Delta \left(\frac{2n\sqrt{d}\alpha C_0 + n^2 d\Delta + 2n\sqrt{d}B}{2\alpha} \right)$$

arises from the fact that we are quantizing each incremental step before transmitting it. If the quantization is very precise then this term is negligible. However, for the highly resource-constrained scenarios envisioned as standard sensor network applications we expect that coarse quantization will be used to reduce the amount of energy expended in communication, and this error must be taken into consideration. Theorem 3 also indicates that although there is an additional error because we quantize the values before transmitting, the number of incremental steps, K, required to reach the limiting behavior is not affected. We emphasize that Theorem 3 describes the worst-case limiting behavior of the algorithm, and that experimentally we observe better average behavior.

In the previous section we saw that the error term in the unquantized algorithm is controlled by the step size parameter α. To summarize Theorem 2, the unquantized incremental subgradient algorithm is guaranteed to achieve a worst-case error proportional to α after, at most, a number of iterations inversely proportional to α^2. For a given problem specification (fixing B, d, and C_0), the error term in Theorem 3 above for the quantized algorithm depends both the step size α and the quantization bin width Δ. Consider the error term in (6). If we take α arbitrarily small without adjusting the bin width Δ, the error due to quantization will dominate the expression. Likewise, if the quantizer is infinitely precise then the we get the same expression as for the unquantized algorithm and the error depends solely on α. In order to balance the error bounds associated with these two parameters we set

$$\Delta = \frac{\alpha^2}{n\sqrt{d}}, \tag{8}$$

in which case the error term in (6) reduces to

$$\frac{(C_0^2 + 2B)\alpha + 2C_0\alpha^2 + \alpha^3 + \epsilon}{2}.$$

For α small (e.g., $\alpha \ll 1$) the α^2 and α^3 terms are negligible in comparison to the α term. Taking $\epsilon = \alpha$ we can summarize Theorem 3 as saying that the quantized incremental subgradient algorithm is guaranteed to achieve a worst-case error (roughly) proportional to α after no more than a number of iterations which is inversely proportional to α^2.

V. An Energy-Accuracy Tradeoff

In this section we derive the tradeoff between the energy expended in running our algorithm and the accuracy achieved, and compare this performance to the approach where all data is quantized and transmitted to a fusion center for processing. It is commonly accepted that the amount of energy consumed for a single wireless communication of one bit is orders of magnitude greater than the energy required for a single local computation [6]. Accordingly, we

focus our analysis on the energy used for wireless communication and compare the energy used by our in-network processing algorithm to that required for every sensor to transmit its data to a fusion center for processing. We assume that the network employs multi-hop communication as opposed to direct point-to-point communication. Chakrabarti et al. have shown that multi-hop communication asymptotically achieves the optimal throughput (as the number of nodes in the network tends to infinity) for the fusion center accumulation setup [7], and our incremental algorithm lends itself to a multi-hop setup since all communication occurs between neighboring nodes in the network. We find it most interesting to study the asymptotic properties of the distributed and centralized algorithms as the number of nodes in the sensor network tends to infinity. Our main conclusion is that as the size of the sensor network increases, the amount of energy used by the quantized incremental algorithm in the entire network is $O\big(n \log n\big)$ bit-hops in comparison to $O\big(n^{3/2}/\sqrt{\log n}\big)$ bit-hops for the centralized algorithm.

For a given data-processing algorithm using multi-hop communication, and for a network of n nodes, let $c(n)$ denote the number of bits transmitted through the network, let $h(n)$ be the average number of hops traversed per bit, and let $e(n)$ denote the amount of energy expended when a node transmits one bit over one hop. The total energy used for in-network communication as a function of the number of nodes in the sensor network for any algorithm is given by

$$\mathcal{E}(n) \quad = \quad c(n) \times h(n) \times e(n).$$

In general, $e(n)$ depends on the density of nodes, the actual communication implementation employed, and physical layer channel properties. Rather than restricting our analysis to a particular setup, we express energy consumption in units of $e(n)$ (i.e., one unit, $e(n)$, per bit-hop).

Now let's take a look at what happens for the in-network algorithm described in Section IV. Each sensor makes a single communication to a neighboring node, so the number of hops per bit for the distributed algorithm is $h_d(n) = 1$. Using the uniform scalar quantization scheme described in the previous section, the number of bits which must be transmitted to reach the limiting behavior of the incremental subgradient algorithm is

$$c_d(n) \quad = \quad (\text{num. bits per transmission}) \times n \times K,$$

since each node transmits once in each cycle. Suppose that the uniform scalar quantizer Q has been designed to use b bits per component so that any $\theta \in \Theta_q$ can be represented using bd bits. Then, based on the assumed bound on the diameter of Θ, we know that the range of values taken by each component $\theta(j)$ of any $\theta \in \Theta$ is such that

$$\sup_{\theta_1, \theta_2 \in \Theta} |\theta_1(j) - \theta_2(j)| \quad \leq \quad B,$$

for $j = 1, \ldots, d$. Distributing 2^b points uniformly over an interval of length B gives us a maximum quantization error of $\Delta = B2^{-b}$ per coordinate. Setting Δ according to (8) to balance the error terms we find that the appropriate number of bits for a given number of sensors grows like

$$\begin{aligned} b \quad &= \quad \log_2 \left(\frac{n\sqrt{d}B}{\alpha^2} \right) \\ &= \quad O(\log n). \end{aligned}$$

Additionally, we set $\epsilon = \alpha$ and find that the number of cycles needed to reach the limiting behavior is

$$\begin{aligned} K &= \left\lfloor \frac{(\mathrm{dist}(\theta_0, \Theta^*))^2}{\alpha^2} \right\rfloor \\ &\leq \frac{B^2}{\alpha^2} \\ &= O(1), \end{aligned}$$

where the inequality follows from the assumed bound on the diameter of Θ. In other words, as the number of nodes in the network tends to infinity, the upper bound on the number of cycles necessary to achieve a specified level of accuracy remains constant. Thus, we have

$$c_d(n) = O\big(\log n\big) \times n \times O(1),$$

and the total energy required to run the distributed incremental algorithm grows with the number of nodes in the network according to

$$\mathcal{E}_d(n) = O(n \log n) \times 1 \times e(n).$$

In the centralized case, nodes must also quantize their data before transmitting it to the fusion center for processing, hence there will be some error incurred. It is difficult to say exactly what this error is without further specifying the functions $f_i(\theta; x_i)$. However, for the sake of making a point, suppose sensors do their best to minimize the amount of data to be transmitted to the fusion center without affecting the accuracy of the centralized algorithm. After they process their data (via local averaging, coding, quantizing, etc.), each of the n sensors transmits at least one bit so that $c_c(n) \geq n$. We model the deployment of nodes in the sensor network using a random geometric graph model where nodes are randomly distributed uniformly over a unit square region $[0, 1]^2$. For a network of n nodes in this model, it has been shown that nodes must set their transmit power such that the communication radius decays like $a\sqrt{\log n/n}$ for some positive constant a in order to guarantee that the network will be connected with high probability[3] [8]. Consequently, the expected radius of the network in hops (equivalently, the expected number of hops to the fusion center) grows roughly like the inverse of the communication radius, and $h_c(n) = a^{-1}\sqrt{n/\log n}$. Thus, the expected total energy for a centralized processing scheme grows at best like

$$\begin{aligned} \mathcal{E}_c(n) &\geq n \times a^{-1}\sqrt{n/\log n} \times e(n) \\ &\propto n^{3/2}/\sqrt{\log n} \times e(n). \end{aligned}$$

We emphasize that this is an extremely optimistic analysis, and that in general much more energy will be required for the centralized approach.

It is clear from this analysis that the distributed incremental algorithm scales better than a centralized approach. Also, we emphasize that the analysis of the distributed incremental algorithm accounts for the worst-case behavior

[3]In [8], Gupta and Kumar analyze the case where nodes are uniformly distributed over the unit disc at random, however a similar analysis reveals that the radius must decay at the same rate for nodes uniformly, randomly distributed over the unit square. See, e.g., [2].

so that $O\big(n \log n\big)$ is an upper bound on the growth rate of expected total energy. To put it plainly, we have a lower bound on energy usage for the centralized processing approach which grows faster than the upper bound on energy usage for a distributed incremental algorithm.

The above analysis dealt with the total energy used over the entire network for both the in-network and centralized algorithms. It is also worth making a few remarks on per-node energy usage. In the distributed incremental scheme, each node transmits once per cycle so energy is used uniformly throughout the network. On the other hand, when all nodes transmit their data to a fusion center there is a greater demand on the resources of nodes which are close to the fusion center since they must transmit the data of other nodes which are further away in addition to their own data.

To further illustrate this point, we can compare the per-node energy usage for the distributed incremental algorithm

$$\frac{\mathcal{E}_d(n)}{n} \quad = \quad O(\log n) \times e(n),$$

with the per-node energy usage for the centralized approach

$$\frac{\mathcal{E}_c(n)}{n} \quad \geq \quad c\sqrt{\frac{n}{\log n}} \times e(n).$$

Thus, the number of transmissions per sensor grows logarithmically in the distributed case as opposed to polynomially in the centralized approach.

VI. Example: Robust Estimation

In parametric statistical inference the model underlying the inference procedure plays an important role in the overall performance of the procedure. If the model used in constructing the inference procedure does not exactly match the true model then the accuracy and variance of the estimator can suffer greatly. The field of statistics known as *robust statistics* is concerned with developing inference procedures which are insensitive to small deviations from the assumptions [9].

In sensor networks, there are many reasons one might want to consider using robust estimates rather than standard inference schemes. Consider the following illustrative example. Suppose that a sensor network has been deployed over an urban region for the purpose of monitoring pollution. Each sensor collects a set of pollution level measurements, $\{x_{i,j}\}_{i=1}^{m}$, $i = 1, \ldots, n$, over the course of a day, and at the end of the day the sample mean pollution level, $\hat{p} = \frac{1}{mn}\sum_{i,j} x_{i,j}$, is calculated. If the variance of each measurement is σ^2 then, assuming i.i.d. samples, the variance of the estimator is σ^2/mn. However, what if some ratio – say 10% – of the sensors are damaged or mis-calibrated so that they give readings with variance $100\sigma^2$? Then the estimator variance increases by a factor of roughly 10. From simulations of this simple example we will see that robust estimation techniques can be extremely useful in a practical system.

In robust estimation, the typical least squares loss function

$$f(\theta) \quad = \quad \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m} ||x_{i,j} - \theta||^2, \tag{9}$$

is replaced with a different loss function

$$f_{robust}(\theta) \quad = \quad \frac{1}{mn}\sum_{i=1}^{n}\sum_{j=1}^{m}\rho(x_{i,j},\theta), \tag{10}$$

with $\rho(x,\theta)$ typically chosen to give less weight to data points which deviate greatly from the parameter, θ. The ℓ-1 distance is one example of a robust loss function. Another standard example is the Huber loss functional for 1-dimensional data,

$$\rho_H(x;\theta) \quad = \quad \begin{cases} (x-\theta)^2/2, & \text{for } |x-\theta| \leq \gamma \\ \gamma|x-\theta| - \gamma^2/2, & \text{for } |x-\theta| > \gamma. \end{cases}$$

This choice of loss function acts as the usual squared error loss function if the data point x is close (within γ) to the parameter θ, but gives less weight to points outside a radius γ from the location θ [9].

A distributed robust estimation algorithm is easily achieved in the incremental subgradient framework by equating $f_i(\theta) \equiv \sum_{j=1}^{m}\rho(x_{i,j};\theta)/mn$. Consider an incremental subgradient algorithm using the Huber loss function. In order to fix a step size and determine the convergence rate of this algorithm, observe that

$$||\nabla f_i(\theta)|| \quad \leq \quad \gamma/n \equiv C.$$

Then, for a desired level of accuracy, ϵ, we need at most $O(1/\epsilon^2)$ cycles. We find that in practice this bound is very loose and much quicker convergence and a finer level of accuracy are achieved.

To demonstrate the usefulness of this procedure, we have simulated the scenario described above where sensors take i.i.d. one dimensional measurements corrupted by additive white Gaussian noise. In this example 100 sensors each make 10 measurements, however 10% of the sensors are damaged and give noisier readings than the other sensors. We use the notation $X \sim \mathcal{N}(\mu,\sigma^2)$ to denote a Gaussian distributed random variable X with mean μ and variance σ^2. A sensor which is working makes readings with distribution $x_{i,j} \sim \mathcal{N}(10.3,1)$, and a damaged sensor makes readings distributed according to $x_{i,j} \sim \mathcal{N}(10.3,100)$. We use the Huber loss function with $\gamma = 1$ and step size $\alpha = 0.1$. Examples showing convergence of quantized and unquantized incremental methods using both least squares (which corresponds the maximum likelihood estimate, in this case), and robust loss functions are shown in Figure 1. The plots depict the evolution of the residual error after each node makes an update. The gray horizontal dashed lines in each plot show the magnitude of the clairvoyant maximum likelihood estimate. That is, the dashed lines indicate the error which would be achieved if all of the unquantized, real-valued sensor data was processed at a fusion center. The figures on the left show the residual for each unquantized incremental algorithm, and the figures on the right show residuals for the quantized algorithms. The algorithms used to generate both figures on the right used quantization bins of width $\Delta = 1$. We repeated each scenario 100 times and found that the algorithm always converges after two cycles through the network which is much lower than the theoretical bound. We declare that the incremental procedure has converged if after successive cycles the change in estimate values is less than 0.1. Also, note that the true mean, 10.3, is not an integer, and thus does not lie on one of the quantization lattice points.

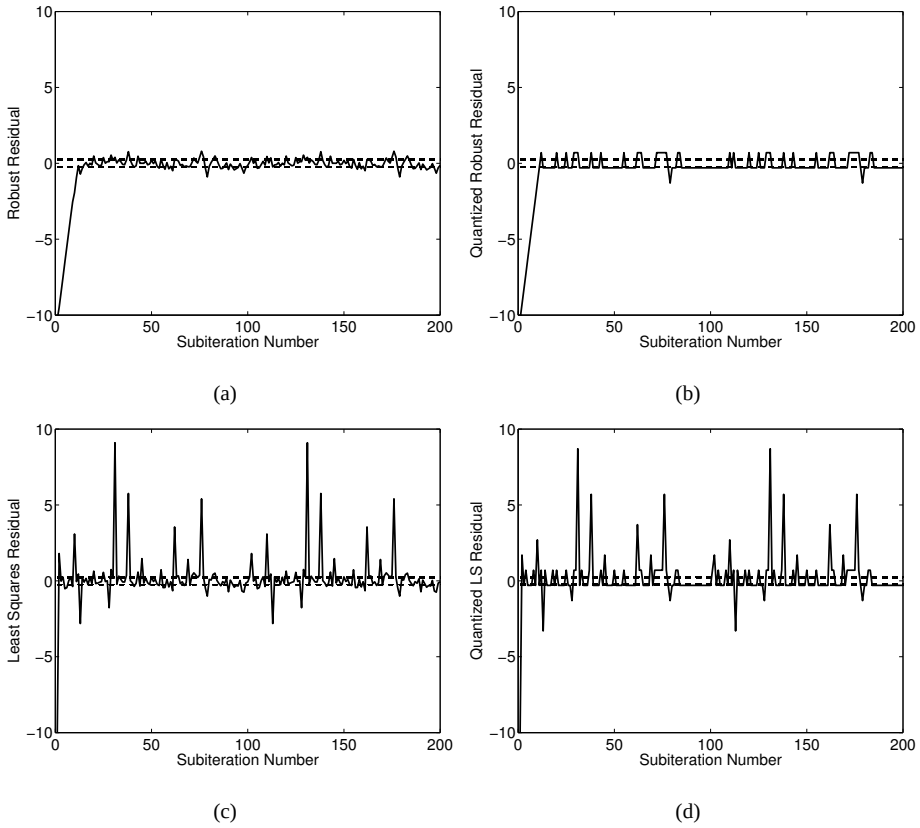

Fig. 1. Example of (a) unquantized and (b) quantized robust incremental estimation procedure using the Huber loss function when 10% of the sensors are damaged. The network consists of 100 sensors, each making 10 measurements. Good sensors make measurements from a $\mathcal{N}(10.3, 1)$ distribution, and measurements from damaged sensors are distributed according to $\mathcal{N}(10.3, 100)$. In comparison, the residuals for (c) unquantized and (d) quantized incremental least squares (in-network maximum likelihood) estimates are also depicted. In all figures, the horizontal dashed lines indicate the error that a clairvoyant centralized algorithm would produce for this data set. In each quantized algorithm integer precision is used.

VII. Discussion of Other Practical Issues

A. Cyclic Routing

For the purpose of analyzing routing schemes we represent the network by a graph where sensor nodes in the network correspond to nodes in the graph, and an edge is placed between two nodes if the two sensor nodes have a direct line of communication (they are neighbors). Our in-network processing algorithm hinges on finding a cycle through the network which touches each node once. In graph theoretic terms such a cycle is known as a Hamilton cycle, and the problem of determining whether a graph contains a Hamilton cycle is known to be NP-complete [10]. While this may seem discouraging, there are results pertaining to Hamilton cycles in random geometric graphs which are relevant in the context of this paper.

A random geometric graph is one where nodes are placed uniformly at random in the unit square[4], and two

[4]In more mathematical terms, the nodes of a random geometric graph are placed according to a planar Poisson point process

nodes are joined with a link if the distance between them is less than a radius $r > 0$. Let $\{r_n\}$ be a sequence of radii. A well known result states that if the radius decays according to $r_n = \Theta(\sqrt{\log n/n})$ as a function of the number of nodes in the network, then as $n \to \infty$ the resulting random geometric graph is connected[5] with high probability. Gupta and Kumar use this result as a condition on the rate of decay for the communication radius to ensure connectivity in a heterogeneous sensor network [8].

A similar result due to Petit states that if

$$r_n = \sqrt{a_n/n} \text{ where } r_n \to 0 \text{ and } a_n/\log n \to \infty,$$

then not only is the random geometric graph connected, but it also contains a Hamilton cycle with high probability [2]. This result is encouraging in that it tells us that we can obtain a network which contains a Hamilton cycle for the small price of increasing the communication radius by a negligible amount (e.g., $r_n \propto \sqrt{\log^2 n/n}$). In his paper, Petit also discusses a divide-and-conquer approach for finding a Hamilton cycle in a random geometric graph by dividing the unit square into smaller squares, finding a Hamilton cycle through the nodes in each of these smaller regions, and then patching the smaller Hamilton cycles together at the boundaries to get a Hamilton cycle for the entire graph. The intuition behind this procedure stems from the notion that nodes in a random geometric graph are nicely distributed over the unit square, making it possible to guarantee that there will be cycles in the smaller squares and that these cycles can be connected. Petit describes in further detail how to set the size of the smaller boxes appropriately. Additionally, Levy et al. describe a distributed algorithm which finds Hamilton cycles in more general random graphs with high probability when they exist [11]. Their algorithm runs in polynomial time (as a function of the number of nodes and edges in the graph) and may be useful for finding Hamilton cycles in smaller squares before patching them together according to Petit's scheme. Such algorithms could be used in an initialization phase to establish a cycle though the network, on which the distributed incremental algorithm could then be run.

B. Imperfect Transmission

Throughout this paper we have assumed that a node can transmit perfectly and reliably to its neighbor, however in reality this may not be the case. A thorough study of the effects of lossy channels and other physical layer communication issues is beyond the scope of this work, however we do have a few remarks. For the centralized data processing approach where all nodes transmit their data to a fusion center, if some data is corrupted with transmission noise or if some data is even lost in transit, because all of the data is being transmitted to the fusion center it is not likely that errors or losses will have a major effect on the computation. On the other hand, with in-network processing and cyclic routing, if a packet is dropped then the optimization parameter is effectively gone and the process must be restarted. If this happens too often then the algorithm will never run long enough to achieve the limiting behavior. This problem can be remedied if a reliable transmission scheme is used to avoid dropped packets,

[5]A graph is connected if there is a path between any two nodes

however reliable transmission generally comes at the cost of higher latency. If channel conditions are extremely unreliable transmissions may need to be rerouted which would also incur a delay and require additional energy and bandwidth resources. We are currently investigating ways of adaptively dealing with unreliable networking conditions in the context of decentralized incremental algorithms.

VIII. Conclusion

This paper is concerned with determining when in-network processing makes more efficient use of network resources than a centralized approach. We adopted a class of incremental subgradient methods from optimization theory which are well suited to the task of distributed optimization in sensor networks, and developed a quantized version of the algorithm which addresses the need, in practice, to transmit information in a finite number of bits. We found that the amount of energy required to run the quantized incremental algorithm (in the worst-case) is on the order of $n \log n$ bit-hops, whereas the amount of energy required for nodes to transmit a minimal amount of data to a fusion center grows at best like $n^{3/2}/\sqrt{\log n}$. Thus, the in-network algorithm scales much better than the centralized approach. An example application of robust estimation in sensor networks demonstrated the performance of our distributed incremental algorithm. Throughout this paper we have assumed that the local objective functions were convex. In our other work we have investigated the use of incremental algorithms for solving non-convex problems, and experimental results indicate that these algorithms behave well for minimizing some important non-convex applications, including source localization and tracking [1], [12].

The major drawbacks of the proposed distributed incremental algorithm revolve around the fact that routing occurs on a cycle through the network which touches each node exactly once. Because it is sequential in nature, the latency of this algorithm will generally be higher than that needed to accumulate data from all sensors at a fusion center. Additionally, since only one node makes an update at each point in time, it seems as though some energy is being wasted since many nodes may hear the broadcast of an updated value if they are in the neighborhood of the transmitter. Finally, maintaining a cycle through the network is a non-trivial task when faced with challenges such as faulty communication channels or failing nodes. This begs the question, *can similar algorithms be designed for other routing structures?* We plan to address all of these issues in the near future.

IX. Acknowledgements

The authors wish to thank Mr. Rui Castro for his thoughtful comments and insightful discussions which helped to improve this manuscript.

Appendix

Before proceeding with the proof of Theorem 3 we derive a few useful properties of the operator $Q[\cdot]$ described in Section IV.

Lemma 1: The quantization operator $Q : \mathbb{R}^d \to \Theta_q$ described in Section IV has the following properties:

1) For all $\theta \in \Theta$ the quantization error is bounded by

$$\|Q[\theta] - \theta\| \quad \leq \quad \sqrt{d}\Delta.$$

2) For all $x \in \mathbb{R}^d$, $\theta \in \Theta$ we have

$$\|Q[x] - \theta\| \quad \leq \quad \|x - \theta\| + \sqrt{d}\Delta.$$

Proof: Because $\theta \in \Theta$, the operation $Q[\theta]$ amounts to quantizing θ to the nearest lattice point in Θ_q. The first property follows from the design of the quantization lattice Λ and that assumption that Θ is a convex, compact set. For the second property observe that in general $Q\big[x\big] = Q\big[\mathcal{P}_\Theta[x]\big]$ and $\mathcal{P}_\Theta[x] \in \Theta$. Invoking the triangle inequality and using the first property from this lemma we have

$$\begin{aligned}
\|Q[x] - \theta\| &= \|Q[x] - \mathcal{P}_\Theta[x] + \mathcal{P}_\Theta[x] - \theta\| \\
&\leq \|\mathcal{P}_\Theta[x] - \theta\| + \|Q[x] - \mathcal{P}_\Theta[x]\| \\
&\leq \|\mathcal{P}_\Theta[x] - \theta\| + \sqrt{d}\Delta \\
&\leq \|x - \theta\| + \sqrt{d}\Delta,
\end{aligned}$$

where the last line follows since $\mathcal{P}_\Theta$ projects its argument onto Θ which is a convex set. ■

The next result characterizes the performance of a single cycle of the incremental algorithm and will be the basis of the proof of Theorem 3.

Lemma 2: Under assumptions (A1)-(A3) listed in Section II, for the sequence $\{\theta_k\}$ generated by the quantized incremental subgradient algorithm (3)-(4), for all $\theta \in \Theta$ and for all $k \geq 1$ we have

$$\|\theta_k - \theta\|^2 \quad \leq \quad \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) + (\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B. \tag{11}$$

Proof: This proof is similar to the proof of Lemma 2.1 in [4]. Without loss of generality, we assume that $\theta_0 \in \Theta_q$ so that $\vartheta_{i,k} \in \Theta_q$ for all $i = 0, \ldots, n$, and $k \geq 1$. Applying part 2 of Lemma 1 yields

$$\begin{aligned}
\|\vartheta_{i,k} - \theta\|^2 &= \|Q[\vartheta_{i-1,k} - \alpha g_{i,k}/n] - \theta\|^2 &(12)\\
&\leq \big(\|\vartheta_{i-1,k} - \alpha g_{i,k}/n - \theta\| + \sqrt{d}\Delta\big)^2 &(13)\\
&= \|\vartheta_{i-1,k} - \alpha g_{i,k}/n - \theta\|^2 + 2\sqrt{d}\Delta\|\vartheta_{i-1,k} - \alpha g_{i,k}/n - \theta\| + d\Delta^2. &(14)
\end{aligned}$$

Invoking the triangle inequality we obtain

$$\begin{aligned}
\|\vartheta_{i,k} - \theta\|^2 &\leq \|\vartheta_{i-1,k} - \alpha g_{i,k}/n - \theta\|^2 + 2\sqrt{d}\Delta\big(\|\vartheta_{i-1,k} - \theta\| + \|\alpha g_{i,k}/n\|\big) + d\Delta^2 \\
&= \|\vartheta_{i-1,k} - \theta\|^2 - \frac{2\alpha}{n}\left\langle g_{i,k}, \vartheta_{i-1,k} - \theta\right\rangle + \|\alpha g_{i,k}/n\|^2 + 2\sqrt{d}\Delta(\|\vartheta_{i-1,k} - \theta\| + \|\alpha g_{i,k}/n\|) + d\Delta^2 \\
&\leq \|\vartheta_{i-1,k} - \theta\|^2 - \frac{2\alpha}{n}\left\langle g_{i,k}, \vartheta_{i-1,k} - \theta\right\rangle + (\alpha C_0/n)^2 + 2\sqrt{d}\Delta(B + \alpha C_0/n) + d\Delta^2 \\
&= \|\vartheta_{i-1,k} - \theta\|^2 - \frac{2\alpha}{n}\left\langle g_{i,k}, \vartheta_{i-1,k} - \theta\right\rangle + (\alpha C_0/n + \sqrt{d}\Delta)^2 + 2\sqrt{d}\Delta B,
\end{aligned}$$

where the second to last line follows from the bounds described in assumptions (A1) and (A2). Since $g_{i,k}$ is a subgradient of the convex function f_i at $\vartheta_{i-1,k}$, by definition

$$\langle g_{i,k}, \theta - \vartheta_{i-1,k}\rangle \quad \leq \quad f_i(\theta) - f_i(\vartheta_{i-1,k}), \tag{15}$$

so that

$$\|\vartheta_{i,k} - \theta\|^2 \quad \leq \quad \|\vartheta_{i-1,k} - \theta\|^2 - \frac{2\alpha}{n}\big(f_i(\vartheta_{i-1,k}) - f_i(\theta)\big) + (\alpha C_0/n + \sqrt{d}\Delta)^2 + 2\sqrt{d}\Delta B. \tag{16}$$

Summing both sides of the above inequality over $i = 1, \ldots, n$ and rearranging terms we obtain

$$\begin{aligned}\|\vartheta_{n,k} - \theta\|^2 + \sum_{i=1}^{n-1}\|\vartheta_{i,k} - \theta\|^2 \quad \leq \quad & \sum_{i=2}^{n}\|\vartheta_{i-1,k} - \theta\|^2 + \|\vartheta_{0,k} - \theta\|^2 - \frac{2\alpha}{n}\sum_{i=1}^{n}\big(f_i(\vartheta_{i-1,k}) - f_i(\theta)\big) \\ & + n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B.\end{aligned}$$

Recalling that $\vartheta_{n,k} = \theta_k = \vartheta_{0,k+1}$, the above expression is equivalent to

$$\begin{aligned}\|\theta_k - \theta\|^2 \quad \leq \quad & \|\theta_{k-1} - \theta\|^2 - \frac{2\alpha}{n}\sum_{i=1}^{n}\big(f_i(\vartheta_{i-1,k}) - f_i(\theta_{k-1}) + f_i(\theta_{k-1}) - f_i(\theta)\big) \\ & + n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B \qquad (17) \\ = \quad & \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) - \frac{2\alpha}{n}\sum_{i=2}^{n}\big(f_i(\vartheta_{i-1,k}) - f_i(\theta_{k-1})\big) \\ & + n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B. \qquad (18)\end{aligned}$$

Note that the summation in the last line is only over terms $i = 2, \ldots, n$ since $\vartheta_{0,k} = \theta_{k-1}$. Let $g_i(\theta_{k-1}) \in \partial f_i(\theta_{k-1})$ so that, similar to (15), we have

$$\langle g(\theta_{k-1}), \vartheta_{i-1,k} - \theta_{k-1}\rangle \quad \leq \quad f_i(\vartheta_{i-1,k}) - f_i(\theta_{k-1}). \tag{19}$$

Using this in (18) and applying the Cauchy-Schwarz inequality we obtain

$$\begin{aligned}\|\theta_k - \theta\|^2 \quad \leq \quad & \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) - \frac{2\alpha}{n}\sum_{i=2}^{n}\langle g(\theta_{k-1}), \vartheta_{i-1,k} - \theta_{k-1}\rangle \\ & + n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B \qquad (20) \\ \leq \quad & \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1} - f(\theta)\big) + \frac{2\alpha}{n}\sum_{i=2}^{n}\|g_i(\theta_{k-1})\| \cdot \|\vartheta_{i-1,k} - \theta_{k-1}\| \\ & + n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B. \qquad (21)\end{aligned}$$

Next, observe that using part 2 of Lemma 1 we can write

$$\begin{aligned}\|\vartheta_{i,k} - \vartheta_{i-1,k}\| \quad = \quad & \|Q[\vartheta_{i-1,k} - \alpha g_{i,k}/n] - \vartheta_{i-1,k}\| \\ \leq \quad & \|\vartheta_{i-1,k} - \alpha g_{i,k}/n - \vartheta_{i-1,k}\| + \sqrt{d}\Delta \\ \leq \quad & \alpha C_0/n + \sqrt{d}\Delta,\end{aligned}$$

where we use the bound from assumption (A1) to obtain the last line. Moreover,

$$\vartheta_{i-1,k} - \theta_{k-1} \quad = \quad \sum_{j=1}^{i-1}\vartheta_{j,k} - \vartheta_{j-1,k},$$

so that after taking the norm of both sides and applying the triangle inequality we obtain

$$\|\vartheta_{i-1,k} - \theta_{k-1}\| \quad \leq \quad (i-1)(\alpha C_0/n + \sqrt{d}\Delta).$$

Using this last result in (21) we find that

$$\begin{aligned}
\|\theta_k - \theta\|^2 \quad &\leq \quad \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) + 2\sum_{i=2}^{n}\sum_{j=1}(i-1)(\alpha C_0/n)(\alpha C_0/n + \sqrt{d}\Delta) \\
&\quad +n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B \\
&\leq \quad \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) + 2\sum_{i=2}^{n}(i-1)(\alpha C_0/n + d\Delta)^2 \\
&\quad +n(\alpha C_0/n + \sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B,
\end{aligned}$$

where the last line holds since $\alpha C_0/n + \sqrt{d}\Delta \geq \alpha C_0/n$. To finish off the proof, observe that

$$\begin{aligned}
2(\alpha C_0/n + \sqrt{d}\Delta)^2 \sum_{i=2}^{n}(i-1) + n(\alpha C_0/n + \sqrt{d}\Delta)^2 \quad &= \quad n^2(\alpha C_0/n + \sqrt{d}\Delta)^2 \\
&= \quad (\alpha C_0 + n\sqrt{d}\Delta)^2.
\end{aligned}$$

Then we have that for all $\theta \in \Theta$ and all $k \geq 1$,

$$\|\theta_k - \theta\|^2 \quad \leq \quad \|\theta_{k-1} - \theta\|^2 - 2\alpha\big(f(\theta_{k-1}) - f(\theta)\big) + (\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B.$$

■

Proof of Theorem 3: This proof is similar to Nedić and Bertsekas's proofs of Proposition 2.1 in [4] and Proposition 2.3 in [5], but we restate it here for completeness.

We first establish the limiting behavior of the sequence generated by the algorithm (3)-(4). Suppose, for the sake of a contradiction, that there exists a $\delta > 0$ such that

$$\liminf_{k\to\infty} f(\theta_k) > f^* + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha} + 2\delta.$$

Let $\tilde{\theta}$ be a point in Θ such that

$$\liminf_{k\to\infty} f(\theta_k) \geq f(\tilde{\theta}) + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha} + 2\delta. \tag{22}$$

Additionally, for all $k \geq 0$ we have

$$\begin{aligned}
f(\theta_k) \quad &\geq \quad \liminf_{k\to\infty} f(\theta_k) \\
&\geq \quad \liminf_{k\to\infty} f(\theta_k) - \delta.
\end{aligned} \tag{23}$$

Combining the statements (22) and (23) we get that for all $k \geq 0$

$$f(\theta_k) - f(\tilde{\theta}) \geq \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha} + \delta.$$

In particular, this equation holds for $k-1 \geq 0$, and applying it in equation (11) from Lemma 2 with $\theta = \tilde{\theta}$ we find that

$$\begin{aligned}
\|\theta_k - \tilde{\theta}\|^2 &\leq \|\theta_{k-1} - \tilde{\theta}\|^2 - 2\alpha\left(\frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha} + \delta\right) + (\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B \\
&= \|\theta_{k-1} - \tilde{\theta}\|^2 - 2\alpha\delta \\
&\leq \|\theta_{k-2} - \tilde{\theta}\|^2 - 4\alpha\delta \\
&\vdots \\
&\leq \|\theta_0 - \tilde{\theta}\|^2 - 2k\alpha\delta.
\end{aligned}$$

For large enough k the right hand side can be made negative contradicting the definition of a norm as being non-negative. Hence, we have

$$\liminf_{k\to\infty} f(\theta_k) \geq f^* + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B}{2\alpha}.$$

Next, we examine the number of steps required to approximate this limiting behavior to within a factor of $\epsilon > 0$. Let $\theta^* \in \Theta^*$ be an optimal solution which achieves $\|\theta_0 - \theta^*\| = \operatorname{dist}(\theta_0, \Theta^*)$. Assume, for the sake of a contradiction that for all $k \leq K = \lfloor \|\theta_0 - \theta^*\|^2/\alpha\epsilon \rfloor$

$$\begin{aligned}
f(\theta_k) &> f(\theta^*) + \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B + \alpha\epsilon}{2\alpha} \\
f(\theta_k) - f(\theta^*) &> \frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B + \alpha\epsilon}{2\alpha}.
\end{aligned}$$

Using this inequality in equation (11) from Lemma 2 above with $\theta = \theta^*$ now, we find that

$$\begin{aligned}
\|\theta_{K+1} - \theta^*\|^2 &\leq \|\theta_K - \theta^*\|^2 - 2\alpha\left(\frac{(\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B + \alpha\epsilon}{2\alpha}\right) + (\alpha C_0 + n\sqrt{d}\Delta)^2 + 2n\sqrt{d}\Delta B \\
&= \|\theta_K - \theta^*\|^2 - \alpha\epsilon \\
&\leq \|\theta_{K-1} - \theta^*\|^2 - 2\alpha\epsilon \\
&\vdots \\
&\leq \|\theta_0 - \theta^*\|^2 - (K+1)\alpha\epsilon
\end{aligned}$$

The left hand side is a normed quantity, implying that the right hand side is greater than or equal to zero, but if

$$\|\theta_0 - \theta^*\|^2 - (K+1)\alpha\epsilon \geq 0$$

then

$$K \leq \frac{\|\theta_0 - \theta^*\|^2}{\alpha\epsilon} - 1,$$

which contradicts the definition of K, as desired. ■

REFERENCES

[1] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proc. Information Processing in Sensor Networks*, Berkeley, California, April 2004.

[2] J. Petit, "Hamiltonian cycles in faulty random geometric networks," in *Proc. 2nd Intl. Wksp. Approximation and Randomization in Communication Networks*, 2001.

[3] V. Kibardin, "Decomposition into functions in the minimization problem," *Automation and Remote Control*, vol. 40, pp. 1311–1323, 1980.

[4] A. Nedić and D. Bertsekas, "Incremental subgradient methods for nondifferentiable optimization," Massachusetts Institute of Technology, Cambridge, MA, Tech. Rep. LIDS-P-2460, 1999.

[5] ——, "Convergence rate of incremental subgradient algorithms," in *Stochastic Optimization: Algorithms and Applications*, S. Uryasev and P. Pardalos, Eds. Kluwer Academic Publishers, 2000, pp. 263–304.

[6] L. Doherty, B. Warneke, B. Boser, and K. Pister, "Energy and performance considerations for smart dust," *International Journal on Parallel and Distributed Systems and Networks*, vol. 4, no. 3, pp. 121–133, 2001.

[7] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Multi-hop communication is order-optimal for homogeneous sensor networks," in *Proc. Information Processing in Sensor Networks*, Berkeley, California, April 2004.

[8] P. Gupta and P. R. Kumar, "Critical power for asymptotic connectivity in wireless networks," in *Stochastic Analysis, Control, Optimization, and Applications*, Boston, 1998, pp. 1106–1110.

[9] P. J. Huber, *Robust Statistics*. John Wiley & Sons, 1981.

[10] C. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1998.

[11] E. Levy, G. Louchard, and J. Petit, "A distributed algorithm to find hamiltonian cycles in $\mathcal{G}(n,p)$ random graphs," in *Proc. Wksp. Combinatorial and Algorithmic Aspects of Networking*, Banff, Alberta, Canada, August 2004, to appear.

[12] M. Rabbat and R. Nowak, "Decentralized source localization and tracking," in *Proc. IEEE ICASSP*, Montreal, May 2004.