Distributed Optimization in Sensor Networks

Michael Rabbat and Robert Nowak*

Abstract

Wireless sensor networks are capable of collecting an enormous amount of data over space and time. Often, the ultimate objective is to derive an estimate of a parameter or function from these data. This paper investigates a general class of distributed algorithms for "in-network" data processing, eliminating the need to transmit raw data to a central point. This can provide significant reductions in the amount of communication and energy required to obtain an accurate estimate. The estimation problems we consider are expressed as the optimization of a cost function involving data from all sensor nodes. The distributed algorithms are based on an incremental optimization process. A parameter estimate is circulated through the network, and along the way each node makes a small adjustment to the estimate based on its local data. Applying results from the theory of incremental subgradient optimization, we show that for a broad class of estimation problems the distributed algorithms converge to within an ϵ-ball around the globally optimal value. Furthermore, bounds on the number incremental steps required for a particular level of accuracy provide insight into the trade-off between estimation performance and communication overhead. In many realistic scenarios, the distributed algorithms are much more efficient, in terms of energy and communications, than centralized estimation schemes. The theory is verified through simulated applications in robust estimation, source localization, cluster analysis and density estimation.

1 Introduction

Wireless sensor networks provide an attractive approach to spatially monitoring environments. Wireless technology makes these systems relatively easy to deploy, but also places heavy demands on energy consumption for communication.

A major challenge in developing sensor network systems and algorithms is that 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 amount of data collected by sensor networks. However, in many applications, the ultimate objective is not merely 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 paper considers an alternate approach based on distributed *in-network* processing which, in many cases, may significantly decrease the communication and energy resources consumed.

The basic idea is illustrated by a simple example. Consider a sensor network comprised of n nodes uniformly distributed over a square meter, each of which collects m measurements. Suppose that our objective is 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 $O(mn)$ bits need to be transmitted over an average of $O(1)$ meter.

*Supported by the National Science Foundation, grants CCR-0310889 and ANI-0099148, DOE SciDAC, and the Office of Naval Research, grant N00014-00-1-0390. M. Rabbat (rabbat@cae.wisc.edu) and R. Nowak (nowak@engr.wisc.edu) are with the Department of Electrical and Computer Engineering at the University of Wisconsin - Madison

2. Sensors first compute a local average and then transmit the local averages to a central processor which computes the global average. This requires only $O(n)$ bits to be transmitted over $O(1)$ meter.

3. Construct a path through the network which passes through all nodes and visits each node just once. The sequence of nodes can be constructed so that the path hops from neighbor to neighbor. 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 only $O(n^{-1/2})$ meters.

Clearly the last procedure could be much more communication efficient than the other two approaches. It is also clear that a similar procedure could be employed to compute any average quantity (*e.g.,* a least squares fit to any number of parameters). Are similar procedures possible for computing other sorts of estimates?

The answer is yes. 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 a simple accumulation process leads to a solution. More general optimization problems do not share this simple feature, but nonetheless 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 criteria possess the following important form:

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

where θ is the parameter of function to be estimated, and $f(\theta)$ is the cost function which can be expressed as a sum of n "local" functions $\{f_i(\theta)\}_{i=1}^n$ in which $f_i(\theta)$ only depends on the data measured at sensor i. For example, in the case of the average considered above,

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

and

$$f_i(\theta) = \frac{1}{m}\sum_{j=1}^{m}(x_{i,j}-\theta)^2,$$

where $x_{i,j}$ is the j-th measurement at the i-th 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 (e.g., reduce) its local cost and then passes the update to the next node. In the case of a quadratic cost function, one pass through the network is sufficient to solve the optimization. In more general cases, 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. A typical sort of result states that after K cycles, the distributed minimization procedure is guaranteed to produce an estimate $\widehat{\theta}$ satisfying $f(\widehat{\theta}) \leq f(\theta^*) + O(K^{-1/2})$, where θ^* is the minimizer of f. Also, the procedure only requires that $O(nK)$ bits be communicated over $O(n^{-1/2})$ meters. Alternatively, transmitting all data to a fusion center requires $O(mn)$ bits 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.

In addition to a theoretical analysis of distributed estimation algorithms of this sort, we also investigate their application in three problems:

Robust estimation: Robust estimates are often derived from criteria other than squared error. We will consider one such case, wherein the sum-of-squared errors criterion is replaced by the Huber loss function.

Source localization: Source localization algorithms are often based on a squared-error criterion (e.g., Gaussian noise model), but the location parameter of interest is usually nonlinearly related to the data (received signal strength is inversely proportional to the distance from source to sensor) leading to a nonlinear estimation problem.

Cluster and density estimation: In the "discovery" process, one may have very little prior information about characteristics of the environment and distribution of the data. Clustering and density estimation are standard first-steps in data exploration and analysis and usually lead to non-quadratic optimizations.

All three problems can be tackled using our distributed algorithms, and simulation experiments in these applications will demonstrate the potential gains obtainable in practical settings.

1.1 Related Work

In the research community, an emphasis has been made on developing energy and bandwidth efficient algorithms for communication, routing, and data processing in wireless sensor network. In [4] Estrin et al. argue that there are significant robustness and scalability advantages to distributed coordination and present a general model for describing distributed algorithms. D'Costa and Sayeed analyze three schemes for distributed classification in [3], and in [13] Shin et al. present a scheme for tracking multiple targets in a distributed fashion. In [1], Boulis et al. describe a scheme for performing distributed estimation in the context of aggregation applications, and Moallemi and Van Roy present a distributed scheme for optimizing power consumption in [8]. All of the papers mentioned are related to the work presented here in that they describe distributed algorithms for accomplishing tasks in sensor networks and balance the same fundamental trade-offs.

In the remainder of the paper we develop a general framework for distributed optimization in wireless sensor networks. The basic theory and explanation of incremental subgradient methods for distributed optimization are discussed in Section 2. An analysis of the energy used for communication by these methods is presented in Section 3. In Section 4 we demonstrate three example applications using incremental subgradient algorithms. Finally, we conclude in Section 5.

2 Decentralized Incremental Optimization

In this section we formalize the proposed decentralized incremental algorithm for performing in-network optimization. We then review the basic theory, methods, and convergence behavior of incremental subgradient optimization presented by Nedić and Bertsekas in [9, 10]. These tools are useful for the energy-accuracy analysis presented in the next section.

We introduce the concept of a subgradient by first recalling an important property of the gradient of a convex differentiable function. For a convex differentiable function, $f : \Theta \rightarrow \mathbb{R}$, the following inequality for the gradient of f at a point θ_0 holds for all $\theta \in \Theta$:

$$f(\theta) \quad \geq \quad f(\theta_0) + (\theta - \theta_0)^T \nabla f(\theta_0).$$

In general, for a convex function f, a *subgradient* of f at θ_0 (observing that f may not be differentiable at θ_0) is any direction g such that

$$f(\theta) \quad \geq \quad f(\theta_0) + (\theta - \theta_0)^T g, \tag{1}$$

and the *subdifferential* of f at θ_0, denoted $\partial f(\theta_0)$, is the set of all subgradients of f at θ_0. Note that if f is differentiable at θ_0 then $\partial f(\theta_0) \equiv \{\nabla f(\theta_0)\}$; *i.e.*, the gradient of f at θ_0 is the only direction satisfying (1).

Recall that we are considering a network of n sensors in which each sensor collect m measurements. Let $x_{i,j}$ denote the j-th measurement taken at the i-th sensor. We would like to compute

$$\widehat{\theta} \quad = \quad \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} f_i(\{x_{i,j}\}_{j=1}^{m}, \theta), \tag{2}$$

where θ is a set of parameters which describe the global phenomena being sensed by the network. The functions $f_i : \mathbb{R}^d \rightarrow \mathbb{R}$ are convex (but not necessarily differentiable) and Θ is a nonempty, closed, and convex subset of $\mathbb{R}^d$. To simplify notation, we will write $f_i(\theta)$ instead of $f_i(\{x_{i,j}\}_{j=1}^{m}, \theta)$; that is, the function $f_i(\theta)$ depends on the data at the i-th sensor as well as the global parameter θ.

Gradient and subgradient methods (e.g., gradient descent) are a popular technique for iteratively solving optimization problems of this nature. The update equation for a centralized subgradient descent approach to solving (2) is

$$\widehat{\theta}^{(k+1)} \quad = \quad \widehat{\theta}^{(k)} - \alpha\sum_{i=1}^{n} g_{i,k}, \tag{3}$$

where $g_{i,k} \in \partial f_i(\widehat{\theta}^{(k)})$, α is a positive step size, and k is the iteration number. Note that each update step in this approach uses data from all of the sensors.

We propose a decentralized incremental approach for solving (2) in which each update iteration (3) is divided into a cycle of n subiterations, and each subiteration focuses on optimizing a single component $f_i(\theta)$. If $\widehat{\theta}^{(k)}$ is the vector obtained after k cycles then

$$\widehat{\theta}^{(k)} \quad = \quad \psi_n^{(k)}, \tag{4}$$

where $\psi_n^{(k)}$ is the result of n subiterations of the form

$$\psi_i^{(k)} = \psi_{i-1}^{(k)} - \alpha g_{i,k}, \qquad i = 1, \ldots, n \tag{5}$$

with $g_{i,k} \in \partial f_i(\psi_{i-1}^{(k)})$ and $\psi_0^{(k)} = \psi_n^{(k-1)}$. For the purposes of analyzing the rate of convergence we can assume that the algorithm is initialized to an arbitrary starting point $\widehat{\theta}^{(0)} \in \Theta$.

As stated above, our algorithm fits into the general framework of incremental subgradient optimization. Nedić and Bertsekas analyze the convergence behavior of incremental subgradient algorithms in [10]. Their results are based on two assumptions. First, they assume that an optimal solution, θ^*, exists. Additionally, they assume that there is a scalar, $\zeta > 0$, such that $||g_{i,k}|| \leq \zeta$ for all subgradients of the functions $f_i(\theta)$, $i = 1, \ldots, n$ and $\theta \in \Theta$. Both assumptions are reasonable for practical, realistic applications as illustrated later in this paper. Under these assumptions and for a constant positive step size, α, we then have that after K cycles, with

$$K \quad = \quad \left\lfloor \frac{||\,\widehat{\theta}^{(0)} - \theta^*||}{\alpha^2\zeta^2} \right\rfloor,$$

we are guaranteed that

$$\min_{0\leq k\leq K} f(\widehat{\theta}^{(k)}) \quad \leq \quad f(\theta^*) + \alpha\zeta^2, \tag{6}$$

where θ^* optimizes f. A proof of this result can be found in [10]. Assume that the distance between the starting point, $\widehat{\theta}^{(0)}$, and an optimal solution θ^* is bounded; $||\,\widehat{\theta}^{(0)} - \theta^*|| \leq c_0$. Setting

$$\epsilon = \alpha\zeta^2, \tag{7}$$

we can interpret (6) as guaranteeing convergence to a solution which brings the objective function within an ϵ-ball of the optimal value, $f(\theta^*)$, after at most

$$\begin{aligned} K &\leq \frac{c_0\zeta^2}{\epsilon^2} \\ &= O(\epsilon^{-2}) \end{aligned} \tag{8}$$

update cycles. The result also directly explains how to set the step size α given a desired level of accuracy ϵ, according to (7). In practice we find that (8) is a loose upper bound on the number of cycles required. Convergence to within a ball around the optimal value is generally the best we can hope to do using iterative algorithms with a fixed step size. Alternatively, one could consider using a diminishing sequence of step sizes $\alpha_k \to 0$ as $k \to \infty$. In this case, it is possible to show that $f(\widehat{\theta}^{(k)}) \to f(\theta^*)$ as $k \to \infty$. However, while the diminishing step size approach is guaranteed to converge to the optimal value, the rate of convergence generally becomes very slow as α_k gets small. In many applications of 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. Using a fixed step size allows the iterative algorithm to be more adaptive. For these reasons we advocate the use of a fixed step size.

3 Energy-Accuracy Tradeoffs

Energy consumption and bandwidth usage must be taken into consideration when developing algorithms for wireless sensor networks. For the incremental subgradient methods described in this paper, the amount of communication is directly proportional to the number of cycles required for the algorithm to converge. Using the theory described in the previous section we precisely quantify the amount of communication needed to guarantee a certain level of accuracy.

It is well-known that the amount of energy consumed for a single wireless communication of one bit can be many orders of magnitude greater than the energy required for a single local computation. Thus, we focus our analysis to the energy used for wireless communication. Specifically, we are interested in how the amount of communication scales with the size of the network. We assume a packet-based multi-hop communication model. Based on this model, the total energy used for in-network communication as a function of the number of nodes in the sensor network for a general data processing algorithm is

$$\mathcal{E}(n) \;=\; b(n) \times h(n) \times e(n),$$

where $b(n)$ is the number of packets transmitted, $h(n)$ is the average number of hops over which communications occur, and $e(n)$ is the average amount of energy required to transmit one packet over one hop.

We compare the energy consumed by our incremental approach to a scheme where all of the sensors transmit their data to a fusion center for processing. Compressing the data at each sensor is an alternative approach (not considered here) that could be used to reduce the number of bits transmitted to the fusion center. See [6] and [7] for two examples of techniques which take this approach. In the centralized approach, the number of packets being transmitted is $b_{cen}(n) = c_1 mn$, where c_1 is the ratio of packet size to measurement size. The average number of hops from any node to the fusion center is $h_{cen}(n) = O(n^{1/d})$,

where d is the dimension of the space that the sensor network is deployed in (*i.e.*, $d = 2$ if the sensors are uniformly distributed over a plane, $d = 3$ for a cube). For the purpose of comparing the energy consumed by a centralized approach to the incremental approach we do not need to precisely quantify $e(n)$ since we assume a multi-hop communication model and the average hop distance is the same for either approach. Thus, the total energy required for all sensors to transmit their data to a fusion center is

$$\mathcal{E}_{cen}(n) \quad \geq \quad c_1 m n^{1+1/d} e(n).$$

According to this relation, the number of single-hop communications increases as either the number of sensors in the network increases or the number of measurements taken at each sensor increases.

Alternatively, in one cycle of our incremental approach each node makes a single communication – the current parameter estimate – to its nearest neighbor. Thus,

$$b_{incr}(n) \quad \propto \quad nK,$$

the number of packets transmitted is proportional to nK, where K is the number of cycles required for the incremental algorithm to be within $O(K^{-1/2})$ of the true solution. In the previous section we saw that the number of cycles required to achieve an estimate $f(\theta^{(K)}) \leq f(\theta^*) + \epsilon$ is bounded above by $K = O(\epsilon^{-2})$. Thus, the relationship between optimization accuracy and the number of packets transmitted is given by $\epsilon^2 b_{incr}(n) = O(n)$. Each communication to a nearest neighbor can be made in one hop, so $h_{incr} = 1$. Thus, for the incremental subgradient algorithm described in this paper, the total energy is

$$\mathcal{E}_{incr}(n) \quad \leq \quad c_2 n \epsilon^{-2} e(n),$$

where the positive constant c_2 depends on the ratio of the packet size to the number of bits required to describe parameter vector θ, as well as the constants in the numerator of (8). This quantity describes the energy-accuracy tradeoff for incremental subgradient algorithms. That is, for a fixed accuracy ϵ, the total energy used for communication grows linearly with the size of the network.

Next, to characterize the advantages of using incremental algorithms for processing, define the energy savings ratio to be

$$\begin{aligned} \mathcal{R} \quad &\equiv \quad \frac{\mathcal{E}_{cen}(n)}{\mathcal{E}_{incr}(n)} = \frac{c_1 m n^{1+1/d}}{c_2 n \epsilon^{-2}} \\ &= \quad c_3 m n^{1/d} \epsilon^2. \end{aligned}$$

This ratio describes how much energy is saved (by way of fewer wireless transmissions) when an incremental – rather than centralized – algorithm is used for computation. Specifically, when $\mathcal{R} > 1$ it is more efficient to use an incremental algorithm for processing. Observe that as either the size of the network or the amount of data collected at each node increase the energy savings ratio increases and it becomes more and more advantageous to use an incremental algorithm.

4 Applications

4.1 Robust Estimation

While statistical inference procedures are based on observations, the model underlying the inference procedure plays an equally important part in the accuracy of inference. If the model used in constructing the inference procedure does not exactly match the true model, then the accuracy and variance 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 modelling assumptions [5].

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 m 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 fraction – 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. Ideally, we would identify and discard these "bad" measurements from the estimation process. Robust estimation techniques attempt to do exactly this by modifying the cost function.

In robust estimation, the classical least squares loss function, $||x - \theta||^2$, is replaced with a different (robust) loss function, $\rho(x, \theta)$, with $\rho(x, \theta)$ typically chosen to give less weight to data points which deviate greatly from the parameter, θ. We then have a modified cost function,

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

to be optimized. The ℓ_1 distance is one example of a robust loss function. Another standard example is the Huber loss function,

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

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 [5].

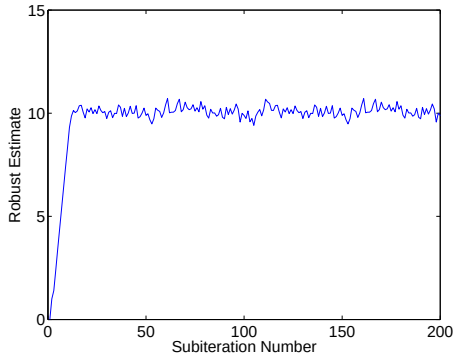
A distributed robust estimation algorithm is easily attained in the incremental subgradient framework by equating

$$f_i(\theta) \quad = \quad \frac{1}{m}\sum_{j=1}^{m}\rho(x_{i,j};\theta).$$

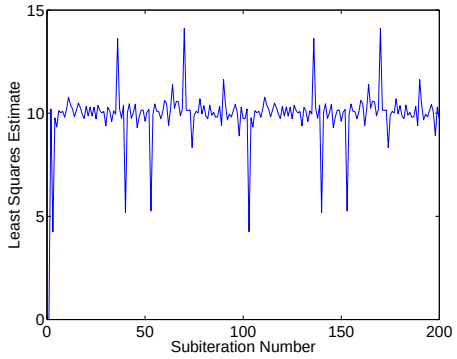
Consider an incremental subgradient algorithm using the Huber loss function. In order to fix a step size and determine the convergence rate of this algorithm, observe that

$$||\nabla f_i(\theta)|| \quad \leq \quad \gamma \equiv \zeta.$$

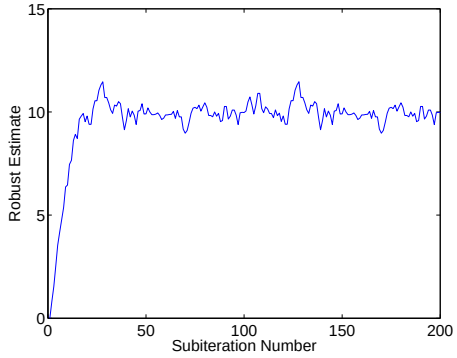
To demonstrate the efficacy of this procedure we have simulated the scenario described above where sensors are uniformly distributed over a homogeneous region, taking i.i.d. one dimensional measurements corrupted by additive white Gaussian noise. In this example 100 sensors each make 10 measurements, however some sensors are damaged and give noisier readings than the other sensors. Let $\mathcal{N}(\mu, \sigma^2)$ denote the Gaussian distribution with mean μ and variance σ^2. A sensor which is working makes readings with distribution $x_{i,j} \sim \mathcal{N}(10, 1)$, and a damaged sensor makes readings distributed according to $x_{i,j} \sim \mathcal{N}(10, 100)$. We use the Huber loss function with $\gamma = 1$ and step size $\alpha = 0.1$. An example illustrating the incremental robust estimate convergence rate is shown in Figure 1(a), with 10% of the sensors being damaged. In contrast, Figure 1(b) depicts the convergence behavior of an incremental subgradient implementation of the least squares algorithm using the same step size. Notice that the least squares estimate converges faster than the robust estimate, but the variance of the distributed least squares result is larger. As a more extreme example, Figures 1(c) and (d) illustrates a scenario where 50% of the sensors are damaged and give noisier readings. In both of these scenarios we repeated the simulation 100 times and found that the algorithm always converges after two cycles (200 subiterations), 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.



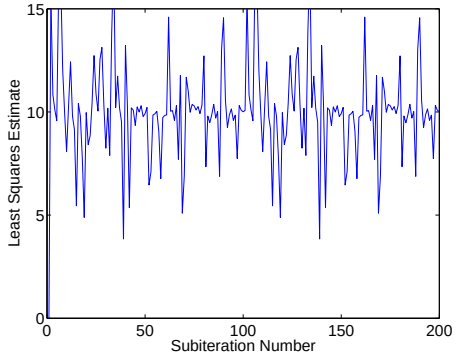
(a) 10% of the sensors are damaged



(b) 10% of the sensors are damaged



(c) 50% of the sensors are damaged



(d) 50% of the sensors are damaged

Figure 1: Example of a robust incremental estimation procedure using the Huber loss function when (a)10% and (c)50% of the sensors are damaged. The network consists of 100 sensors, each making 10 measurements. Good sensors make measurements from a $\mathcal{N}(10, 1)$ distribution, and measurements from damaged sensors are distributed according to $\mathcal{N}(10, 100)$. In comparison, least squares estimates are depicted in (b) and (d) with 10% and 50% of the sensors damaged.

4.2 Energy-Based Source Localization

Estimating the location of an acoustic source is an important problem in both environmental and military applications [2, 12]. In this problem an acoustic source is positioned at an unknown location, θ, in the sensor field. The source emits isotropically a signal, and we would like to estimate the source's location using received signal energy measurements take at each sensor. Again, suppose each sensor collects m measurements. In this example we assume that the sensors are uniformly distributed over either a square or cube with side of length $D \gg 1$, and that each sensor knows its own location, $r_i, i = 1, \ldots, n$, relative to a fixed reference point. Then we use an isotropic energy propagation model for the j-th received signal strength measurement at node i:

$$x_{i,j} \quad = \quad \frac{A}{||\theta - r_i||^\beta} + w_{i,j},$$

where $A > 0$ is a constant and

$$||\theta - r_i|| \quad > \quad 1, \tag{9}$$

for all i. The exponent $\beta \geq 1$ describes the attenuation characteristics of the medium through which the acoustic signal propagates, and $w_{i,j}$ are i.i.d. samples of a zero-mean Gaussian noise process with variance

σ^2. A maximum likelihood estimate for the source's location is found by solving

$$\hat{\theta} \;=\; \arg\min_{\theta} \frac{1}{mn} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right)^2 .$$

This non-linear least squares problem clearly fits into the general incremental subgradient framework. Taking

$$f_i(\theta) \;=\; \frac{1}{m} \sum_{j=1}^{m} \left(x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right)^2 ,$$

we find that the gradient of $f_i(\theta)$ is

$$\nabla f_i(\theta) \;=\; \frac{2\beta A}{m||\theta - r_i||^{\beta+2}} \sum_{j=1}^{m} \left(x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right) (\theta - r_i).$$

Next we bound the magnitude of the gradient by first observing that

$$||\nabla f_i(\theta)|| \;\leq\; \frac{2\beta A||\theta - r_i||}{m||\theta - r_i||^{\beta+2}} \sum_{j=1}^{m} \left| x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right| .$$

From the assumption (9),

$$\begin{aligned} ||\nabla f_i(\theta)|| \;&\leq\; \frac{2\beta A}{m} \sum_{j=1}^{m} \left| x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right| \\ &<\; \frac{2\beta A c_4 m}{m} \\ &=\; 2\beta A c_4, \end{aligned}$$

where the constant c_4 comes from an assumption on limitations of sensor measurement capabilities:

$$\left| x_{i,j} - \frac{A}{||\theta - r_i||^{\beta}} \right| \;<\; c_4. \tag{10}$$

That is, based on our Gaussian model for the noise, the measurements $x_{i,j}$ could theoretically take values over the entire support of the real line. However, we assume that the sensors are only capable of reporting values in a bounded range according to (10).

We have simulated this scenario with 100 sensors uniformly distributed in a 100×100 square, and the source location chosen randomly. The source emits a signal with strength $A = 100$ and each sensor makes 10 measurements at a signal-to-noise ratio of 3dB. Figure 2 depicts an example path taken by the algorithm plotted on top of contours of the log likelihood. Over a sample of 100 simulations with this configuration the algorithm converged to within a radius of 1 of the true source location after an average of 45 cycles.

4.3 Clustering and Density Estimation

The Distributed Expectation-Maximization (DEM) algorithm has previously been proposed in the context of density estimation and clustering for wireless sensor networks [11]. In this scenario, measurements are modelled as being samples drawn from a mixture of Gaussian distributions with unknown means and covariances, with mixture weights potentially being different at each sensor in the network. Such a scenario

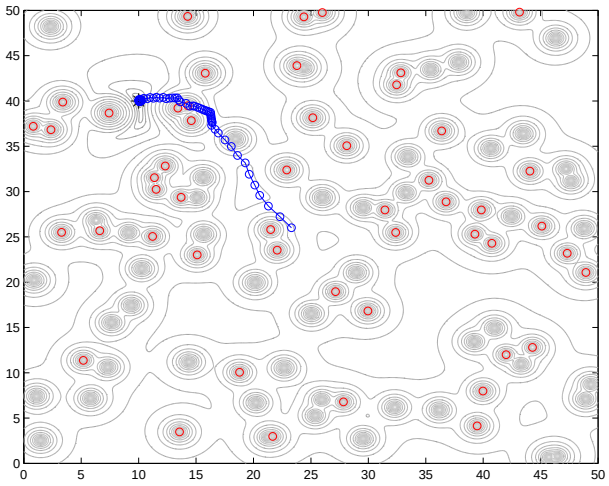


Figure 2: An example path taken by the incremental subgradient algorithm displayed on top of contours of log-likelihood function. The true source location in this example is at the point (10,40). There were 100 sensors in this simulation, each taking 10 measurements.

could arise in cases where the sensors are deployed over an inhomogeneous region. From the data analysis point-of-view, a first step in this setting would involve estimating the parameters of the global density. Once estimates have been obtained, the parameter estimates could be circulated allowing each node to determine which component of the density gives the best fit to its local data, thereby completing the clustering process.

While DEM is not exactly an incremental subgradient method, in this section we relate it to incremental subgradient methods and study its convergence in this context. The goal of DEM is to minimize the function

$$f(\theta) \quad = \quad -\sum_{i=1}^{n}\sum_{j=1}^{m}\log\left(\sum_{k=1}^{J}\alpha_{i,k}\mathcal{N}(y_{i,j}|\mu_k,\Sigma_k)\right),$$

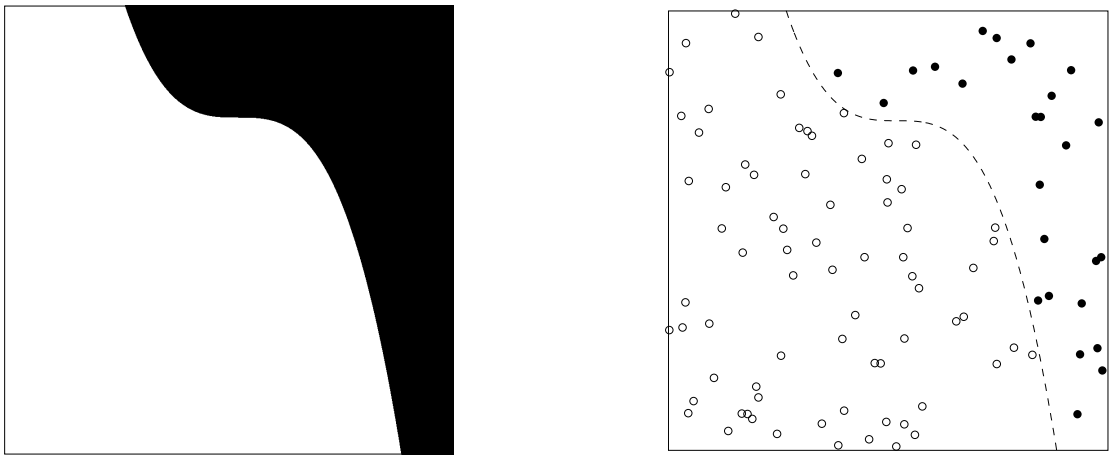
where $\theta = \{\alpha_{i,k}\} \cup \{\mu_k\} \cup \{\Sigma_k\}$, and $\mathcal{N}(y|\mu,\Sigma)$ denotes the multivariate Gaussian density with mean μ and covariance Σ evaluated at the point y. There are a few well-known results about the EM algorithm such as the monotonicity and guaranteed convergence to a fixed point. Less is know about convergence of the incremental EM algorithm, of which DEM is a variant. However, roughly speaking, in each step DEM is behaving like the regular EM algorithm acting locally. In [14], Xu and Jordan discuss convergence properties of the EM algorithm for Gaussian mixture models. A key result which we will use relates the EM steps to the gradient of the log-likelihood function. Corollary 1 of [14] states that

$$\theta^{(k+1)} \quad = \quad \theta^{(k)} + \mathcal{P}(\theta^{(k)})\left.\frac{\partial l_y}{\partial\theta}\right|_{\theta=\theta^{(k)}},$$

where, with probability one, the matrix $\mathcal{P}(\theta^{(k)})$ is positive-definite. That is, the search direction, $\theta^{(k+1)} - \theta^{(k)}$ has a positive projection on the gradient of the log-likelihood function. Because projection may not be exactly aligned along the gradient of the log-likelihood, strictly speaking, DEM is not an incremental subgradient method. However, suppose that the parameters $\{\alpha_{i,k}\}$ and $\{\Sigma_k\}$ are known. Xu and Jordan show that for this case, the projection matrix tends to a diagonal so that EM is equivalent to a gradient method. Thus, under these conditions DEM behaves like an incremental subgradient method and the convergence theory described above applies.

As an example application of the DEM algorithm, suppose that a sensor network is deployed over an inhomogeneous region like the one depicted in Figure 3(a). Sensors located in the light region measure

i.i.d. samples from a $\mathcal{N}(5,5)$ distribution, and sensors in the dark region measure i.i.d. samples from a $\mathcal{N}(15,15)$ distribution. We assume that the number of components (2 in this case) is already known, as the model-order selection problem is a separate, challenging issue in itself. Then, in an initial phase, the sensors run DEM to determine the means and variances of each distribution. Next, each sensor executes a hypothesis test based on its local data and the estimated global parameters in order to determine which class it belongs to (region it is located in). Finally, each sensor transmits it's decision (region 1 or region 2 – a single bit) back to the fusion center which then has a rough image of the field.



(a) (b)

Figure 3: (a)Example of a region composed of two areas. Sensors located in the white area make i.i.d. measurements distributed according to $\mathcal{N}(5,5)$ and sensors in the black measure i.i.d. samples from a $\mathcal{N}(15,15)$ distribution. (b) The outcome of one simulation with 100 sensors each making 100 measurements. The sensors first run DEM to estimate the means and variances of each Gaussian component in the mixture model. Next, based on their local estimates of the mixing probabilities α_1 and α_2, each sensor decides whether it is located in the dark region or the light region. In the figure, open circles correspond to nodes deciding that they are in the light region and filled circles correspond to nodes deciding they are in the dark region. The dotted line shows the true boundary.

We have simulated the procedure described above for a network with 100 sensors uniformly distributed over the region, each making 100 measurements. Figure 3(b) depicts the outcome of one such simulation. The dotted line shows the true boundary separating region 1 from region 2, and sensors are shown in their locations as either open or filled circles. The type of circle indicates the region chosen by the sensor in the hypothesis test. We repeated the simulation 100 times and found that each node always classified itself correctly. Additionally, the DEM procedure converged in an average of 3 cycles, with the maximum being 10.

5 Conclusions

This paper investigated a family of simple, distributed algorithms for sensor network data processing. The basic operation involves circulating a parameter estimate through the network, and making small adjustments to the estimate at each node based on its local data. 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. In particular, we showed that the convergence theory can be applied to gauge the potential benefits (in terms of communication and energy savings) of the distributed algorithms

in comparison with centralized approaches. The theory predicts that K cycles of the distributed algorithm procedure will produce an estimate $\widehat{\theta}$ satisfying $f(\widehat{\theta}) \leq f(\theta^*) + O(K^{-1/2})$, where f is the underlying cost function and θ^* is the minimizer of f. For a network comprised of n nodes uniformly distributed over the unit square or cube and m measurements per node, the number of communications required for the distributed algorithm is roughly a factor of $K/(mn^{1/d})$ less than the number required to transmit all the data to a centralized location for processing. As the size or density of the sensor network increases, the savings provided by the distributed approach can be enormous. Simulated experiments demonstrated the potential of the algorithms in three applications of practical interest in sensor networking.

References

[1] A. Boulis, S. Ganeriwal, and M. Srivastava. Aggregation in sensor networks: An energy-accuracy trade-off. In *Proc. IEEE SNPA*, Anchorage, AK, May 2003.

[2] J. C. Chen, K. Yao, and R. E. Hudson. Source localization and beamforming. *IEEE Sig. Proc. Magazine*, March 2002.

[3] A. D'Costa and A. M. Sayeed. Collaborative signal processing for distributed classification in sensor networks. In *Proc. IPSN*, Palo Alto, CA, April 2003.

[4] D. Estrin, R. Govindan, and J. Heidemann. Scalable coordination in sensor networks. Technical Report USC Tech Report 99-692, USC/ISI, 1999.

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

[6] P. Ishwar, R. Puri, S. Pradhan, and K. Ramchandran. On compression for robust estimation in sensor networks. In *Proc. of IEEE ISIT*, Yokohama, Japan, September 2003.

[7] C. Kreucher, K. Kastella, and A. Hero. Sensor management using relevance feedback learning. Submitted to IEEE Transactions on Signal Processing, June 2003.

[8] C. Moallemi and B. Van Roy. Distributed optimization in adaptive networks. In *Proc. NIPS*, Vancouver, BC, Canada, December 2003.

[9] A. Nedić and D. Bertsekas. Incremental subgradient methods for nondifferentiable optimization. Technical Report LIDS-P-2460, Massachusetts Institute of Technology, Cambridge, MA, 1999.

[10] A. Nedić and D. Bertsekas. *Stochastic Optimization: Algorithms and Applications*, chapter Convergence Rate of Incremental Subgradient Algorithms, pages 263–304. Kluwer Academic Publishers, 2000.

[11] R. Nowak. Distributed EM algorithms for density estimation and clustering in sensor networks. *IEEE Trans. on Sig. Proc.*, 51(8), August 2003.

[12] X. Sheng and Y.-H. Hu. Energy based acoust source localization. In *Proc. IPSN*, Palo Alto, CA, April 2003.

[13] J. Shin, L. Guibas, and F. Zhao. A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks. In *Proc. IPSN*, Palo Alto, CA, April 2003.

[14] L. Xu and M. Jordan. On convergence properties of the EM algorithm for gaussian mixtures. *Neural Computation*, 8:129–151, 1996.