Summary of Distributed Optimization in Sensor Networks

Antti Rusanen 62830U
airusane@cc.hut.fi

19th April 2005

Abstract

Wireless sensor networks are able to collect large amounts of data. Often the objective is to get an estimate of some parameter or function of the data. In this case it may be beneficial to calculate these parameters in a distributed manner instead of sending the raw data to central node for processing. The paper[1] investigates a class of distributed algorithms that circulate the estimates of parameters through the network. Each node updates the parameters based on its own measurements. The paper uses theory of incremental subgradient optimization to create local update functions and to prove that the algorithm converges to within a ϵ-ball around the globally optimal value. The technique is then used for problems like robust estimation and source localization. Some simulation results are presented for these applications.

1 Introduction

A major issue in developing sensor network algorithms is that data trasmission from sensors to central processing location may demand a lot of energy and communication resources. In many cases, however, we are not interested in raw data as such, but rather in an estimate of parameters or functions on that data, for example source locations or average of the measurements. Instead of computing these parameters centrally, the paper investigates algorithms that creates estimates for them in the network.

The idea and motivation is illustrated with an example where we have a sensor network with n sensors uniformly distributed over a square meter, each taking m measurements and we want to calculate the average of the measurements. Three possible approaches presented are:

- Sending all the measurements to central node which calculates the average. $O(nm)$ bits are sent over an average of $O(1)$ meters.

- Counting average of the local measurements at each node and sending that to central node which calculates the average of those. Only $O(n)$ bits are sent over an average of $O(1)$ meters.
- Constructing a Hamilton cycle over the network through which the global average is accumulated each node adding their local average. Only O(n) bits are sent over an average of $O(n^{-1/2})$ meters.

The last approach can be generalized to other estimates besides average by creating cost functions to be minimized. The local averages can be viewed as values minimizing quadratic cost functions. Quadratic optimization problems have solutions that are linear functions of data, so simple accumulation process leads to a solution. This is not the case for more general optimization problems, but many can still be calculated using similar distributed approach. Specifically, many estimation criteria can be presented the following form:

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

where θ is the parameter to be estimated and $f(\theta)$ is the cost function that can be expressed as a sum of local functions f_i which depend only on local data at sensors. In the case of average,

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

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

where n is the number of sensors, m is the number of measurements taken at each sensor and $x_{i,j}$ is the j-th measure at sensor i.

The algorithms presented in the paper operate in a simple manner. A Hamilton cycle is constructed over the sensor network and current estimate of the parameter θ is passed from node to node along the cycle. Each node updates the estimate by improving its local cost. When calculating the average like above, one pass through the network is enough, but generally several cycles are needed.

2 Decentralized Incremental Optimization

The following inequality holds for the gradient of a convex differentiable function $f : \Theta \rightarrow \Re$ at point θ_0 for all $\theta \in \Theta$:

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

More generally, for a convex function f a subgradient of f at θ_0 is any direction g such that:

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

Subgradient $\partial f(\theta_0)$ is the set of all subgradients of f at θ_0.

Now given a network of n sensors and m measurements per node, if we let $x_{i,j}$denote the j-th measurement at i-th node, we want to compute

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

where θ is the set of parameters we want to measure.

Gradient descent is a popular technique for solving convex optimization tasks iteratively. In the central version the update rule is

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

where $g_{i,k} \in \partial f(\hat{\theta}^{(k)}$, α is a positive step size and k is the iteration number. For the distributed version, one update iteration is replaced with a cycle of n subiterations performed at each node.

The convergence of incremental subgradient optimization is analyzed in [2]. Assuming an optimal solution θ^* exists and for each subgradient $\|g_{i,k}\| \leq \zeta$ for some constant ζ, we have that after K cycles, with

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

we are guaranteed that

$$\min_{0 \leq k \leq K} f(\hat{\theta}^{(k)}) \quad \leq \quad f(\theta^*) + \alpha \zeta^2.$$

Bounding the distance between starting point $\hat{\theta}^{(0)}$ and optimal value $\theta^*$by $\left\| \hat{\theta}^{(0)} - \theta^* \right\| \leq c_0$ and setting desired level of accuracy $\epsilon = \alpha \zeta^2$, we get the minimum number of cycles

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

after which we are guaranteed to be within a ϵ-ball of the optimal value. Using diminishing sequence of step sizes $a_k \to 0,$ as $k \to \infty$would guarantee converging to optimal value, but authors recommend against it, because the rate of convergence generally gets very slow with small step sizes.

3 Energy-Accuracy Tradeoffs

Assuming a packet based multihop communication model, energy used by the network for wireless communication can be calculated with

$$E(n) = b(n) \times h(n) \times e(n),$$

where $b(n)$ is the number of packets transmitted, $h(n)$ is the average number of hops over which communication occurs, and $e(n)$ is the average energy consumed when transferring one packet through one hop. The paper compares the energy consumption of the incremental method with a naive method where all the nodes send all their data to the central node. In the central method, $b_{cen}(n) = O(mn)$ and $h_{cen}(n) = O(n^{1/d})$, where d is the dimension of space the sensors are distributed over. Because of the multi-hop model, we do not need to quantify $e(n)$, the average hop distance being the same for both methods. We get

$$E_{cen}(n) \geq c_1 m n^{1+1/d} e(n).$$

For the incremental method, $b_{incr}(n) = O(nK)$ and using the result $K = O(\epsilon^{-2})$ from the previous section, we get $b_{incr}(n) = O(n\epsilon^{-2})$. Given the hop count $h_{incr}(n) = 1$, we get

$$E_{incr}(n) \leq c_2 n \epsilon^{-2} e(n)$$

The energy savings ratio for the incremental method can now be calculated:

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

4 Applications

4.1 Robust Estimation

The accuracy of statistical interference procedures is depends heavily on how well the chosen model matches the true model.

In the paper's example a sensor network is set up for monitoring pollution level of a city. If sensor collects m measurements with variance of σ^2, then the sample mean pollution level $\hat{p} = \frac{1}{mn}\Sigma_{i,j}x_{i,j}$ has variance σ^2/mn. If, however, 10% of the sensors are broken and give readings with variance $100\sigma^2$, the variance of the estimator increases by factor of roughly 10.

Robust statistics is a field developing interference procedures insensitive to small deviations from the modelling assumptions. Robust estimation techniques attempt to discard the "bad" measurements by replacing the least squares loss function $\|x-\theta\|^2$with some robust loss function $\rho(x,\theta)$ which usually gives less weight for data points that differ much from the parameter θ. Then the modified cost function to be optimized is of form

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

One example of loss function is l_1-distance. Another one is Huber loss function,

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

A distributed robust estimation algorithm can be created by equating

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

Using Huber loss function we fix the step size and determine the convergence rate by observing

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

The performance of the procedure was tested by simulating a scenario where 100 sensors, part of which were damaged, made each 10 one dimensional measurements. Working sensors made readings with distribution $x_{i,j} \sim N(10,1)$ and damaged sensors with $x_{i,j} \sim N(10,100)$. Huber loss function with $\gamma = 1$ and step size $\alpha = 0.1$ was used. Figure 1(a) and (b) show the convergence behaviors of incremental robust estimate and incremental least squares algorithm respectively, when 10% of the sensors are damaged. Least squares algorithm converges faster, but has more variance than the robust algorithm. In a more extreme case in Figure(c) and (d) the amount of damaged sensors is 50%. In both scenarios the simulation was repeated 100 times and the authors claim that the robust algorithm always converged (change in estimate was less than 0.1) after two cycles or 200 subiterations.

4.2 Energy-Based Source Localization

Locating acoustic source is a popular problem in military and environmental applications. In this problem, a source at unknown location θ in the sensor field emits a signal isotropically and we want to locate it using signal energy measurements from the sensors.

We assume that sensors are distributed unformly over square or cube with side of length $D \gg 1$ and each sensor knows its location. Isotropic energy propagation model gives the j-th measurement at node i:

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

where $A > 0$ is constant,$\|\theta - r_i\| > 1$, $\beta \geq 1$ describes the attenuation characteristics of the medium and $w_{i,j}$are samples of zero-mean Gaussian noise process with variance σ^2. The maximum likelihood estimate for source location can be found by solving:

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

This fits into the incremental subgradient framework. By letting

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

and limiting the measurement capabilities of the sensors with $\left| x_{i,j} - A \|\theta - r_i\|^{-\beta} \right| < c_4$, we get bound for the gradient:

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

The algorithm was simulated with 100 sensors uniformly distributed over 100×100 square and the source placed at random. Source had signal strength $A = 100$ and each sensor made 10 measurements with signal-to-noise ratio of 3dB. Figure 2 shows an example path taken by the algorithm. The algorithm converged to a within radius 1 of the source location after an average of 45 cycles.

Conclusions and personal observations

The paper investigated a simple family of distributed algorithms for sensor networks that operate by circulating parameter estimates through the network, each node making small updates for the estimates. The algorithms were variants of incremental subgradient optimization procedures, which was used to get theoretical results for their performance. The algorithms were shown to produce estimates of parameters within ϵ-ball of true values with $K = O(\epsilon^{-2})$ cycles. The amount of communication needed was shown to be about a factor of $K/(mn^{1/d})$ less than with a naive scheme that sends all the data to central node for processing.

The problem of creating the Hamilton cycle in the network, which generally is a NP-complete problem, was addressed in the paper. However the authors present in[3] techiques to do this efficiently in random geometric graphs.

Assuming the distance between neighboring nodes in the cycle does not vary greatly, the consumption of energy should be pretty even between the nodes increasing the lifetime of the network.

The cyclic routing is unfortunately very sensitive to node failures. When one node goes down, the whole network breaks until a new cycle is built.

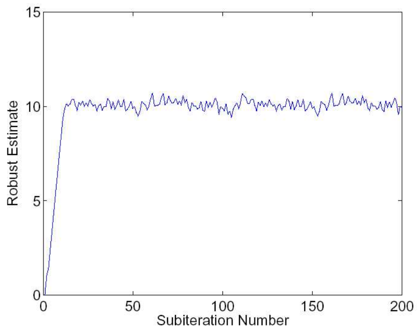
In the source localization problem, the number of cycles required is quite large and every node needs to participate even if they are far from the source and do not contribute much to the localization. Having smaller local cycles and choosing between them based on the energy levels measured by the sensors could perhaps lead to a faster algorithm that uses less energy.

References

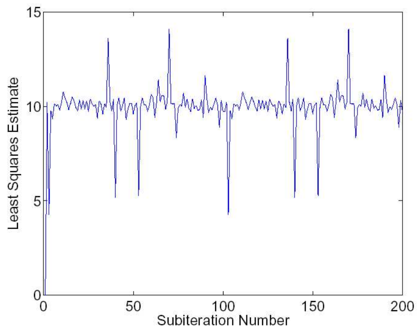
[1] M. Rabbat and R. Nowak. Distributed Optimization in Sensor Networks

[2] A. Nédic and D. Bertsekas. Stochastic Optimization: Algorithms and Applications, chapter Convergence Rate of Incremental Subgradient Algorithms, p. 263-304. Kluwer Academic Publishers, 2000.

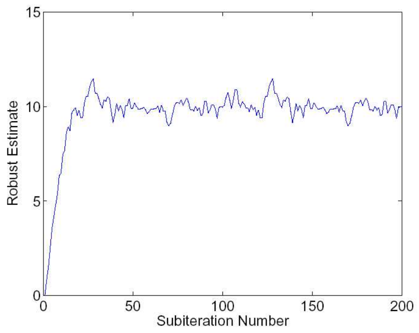
[3] M. Rabbat and R. Nowak. Quantized Incremental Algorithms for Distributed Optimization



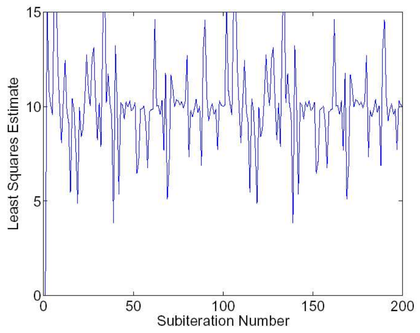
(a) 10% of the sensors are damaged



(b) 10% of the sensors are damaged



(c) 50% of the sensors are damaged



(d) 50% of the sensors are damaged

Figure 1: Robust incremental estimation procedure using Huber loss function (a), (c) and least squares estimates (b), (d) in the scenario with damaged sensors.

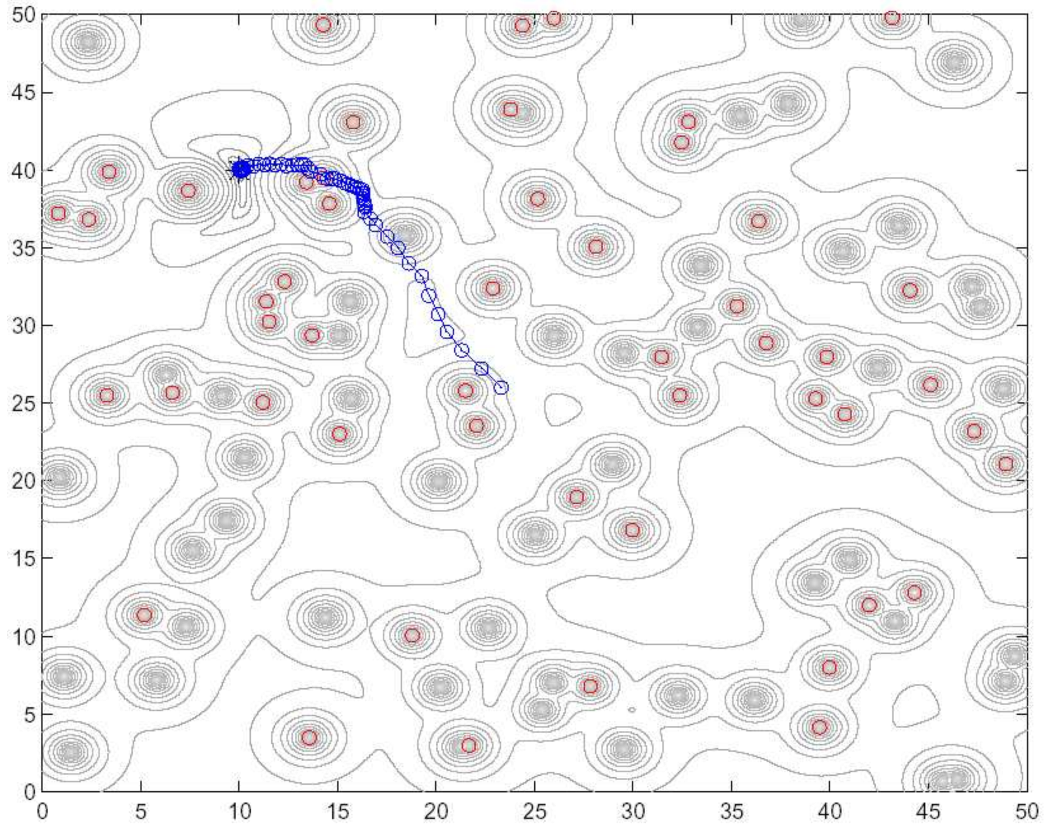


Figure 2: An example path taken by incremental subgradient algorithm finding the acoustic source. True source location is at (10,40)