Distributed and Recursive Parameter Estimation

Srinivasan Sundhar Ram, Venugopal V. Veeravalli, and Angelina Nedić

Abstract Parametric estimation is a canonical problem in sensor networks. The intrinsic nature of sensor networks requires regression algorithms based on sensor data to be distributed and recursive. Such algorithms are studied in this chapter for the problem of (conditional) least squares regression when the data collected across sensors is homogeneous, i.e., each sensor observes samples of the dependent and independent variable in the regression problem. The chapter is divided into three parts. In the first part, distributed and recursive estimation algorithms are developed for the nonlinear regression problem. In the second part, a distributed and recursive algorithm is designed to estimate the unknown parameter in a parametrized state-space random process. In the third part, the problem of identifying the source of a diffusion field is discussed as a representative application for the algorithms developed in the first two parts.

1 Introduction

Sensor networks consist of spatially deployed sensors that sense their local environment across time to learn some feature of interest about the underlying field. In many applications, using a priori information, it is possible to build approximate parametrized model sets for the feature of interest. In such cases, the problem of learning the feature simplifies to the problem of estimating these unknown parameters that determine the model that best fits the observed data.

For example, consider sensors deployed to determine the source of a spatiotemporal temperature field. A model set for the heat source could be the set of all point sources with constant intensity. The model set is parametrized by the source location and the constant intensity, and the sensors measurements have to be used to estimate these values. The diffusion equation that governs the propagation of heat can then be

S.S. Ram (✉)
Electrical and Computer Engineering, University of Illinois at Urbana–Champaign, Urbana, IL, USA
e-mail: ssriniv5@illinois.edu

G. Ferrari (ed.), *Sensor Networks,* Signals and Communication Technology,
DOI 10.1007/978-3-642-01341-6_2,

used to map the model for the heat source to a model for the sensor measurements. Note that the actual source may have an arbitrary shape and have an exponentially decreasing intensity. In effect, the goal is to only determine the best approximation for the source among all point sources with constant intensity.

There are multiple advantages to estimating the unknown parameters jointly using data collected by a network of sensors, rather than individual sensors. First, measurements made by a single sensor may not be sufficient to uniquely resolve the parameter. For example, triangulation of sources require distance measurements format least three sensors that are not on a straight line. Second, there are fundamental limits on the sampling rate of a sensor. By using multiple sensors, it is possible to obtain a higher effective sampling rate. Finally, even when multiple sensors add redundancy, they serve to mitigate the effect of noise in observations.

There are two basic steps in estimation. The first step is to use physical laws to map the models for the features to models for the sensor measurements. In the heat source example, the diffusion equation can be used to map the model for the heat source to a model for the sensor measurements. In this chapter, it is assumed that a parametrized model for the sensor measurements is available and the focus is only on the second step, which is to use statistical techniques to determine the form of the estimator.

The intrinsic nature of sensor networks impose certain challenges. When there are a large number of sensors spread across a large operational area, it is not energy efficient to jointly estimate the parameter by collecting all the measurements from the sensors at a fusion center. Instead, algorithms have to be distributed and the estimate must be calculated by the network through local communication between sensors and their neighbors, without the aid of a central fusion center. Further, each sensor has a limited memory and measurements have to be purged periodically. Thus, the estimation procedures should also be recursive, and only a constant summary statistic must be stored and updated when a new measurement is made. Thus estimation algorithms that are to be used in sensor networks have to be distributed and recursive. In this chapter, recursive and distributed procedures for (conditional) least squares estimation are discussed. The chapter is divided into three parts. In the first part, distributed and recursive estimation algorithms are developed for the nonlinear regression problem. In the second part, a distributed and recursive algorithm is designed to estimate the unknown parameter in a parametrized statespace random process. In the third part, the problem of identifying the source of a diffusion field is discussed as a representative application for the algorithms developed in the first two parts.

2 Preliminaries

The following notational convention is used. For a matrix A, $[A]_{i,j}$ denotes its (i, j)-th entry. All vectors are column vectors and the set consisting of the first k elements of a sequence $\{r_\ell\}_{\ell\in\mathbb{N}}$ is denoted by r^k. A total of m, $m \in \mathbb{N}$, sensors are

spatially deployed and are indexed using the set $V = \{1, \ldots, m\}$. Throughout this chapter the variable $i \in V$ and the variable $k \in \mathbb{N}$. The sensors sequentially and synchronously sense the underlying field once every time unit. The k-th observation made by the i-th sensor is denoted by $z_{i,k}$. We will find it convenient to view $\{z_{i,k}\}$ as a sample path of a random process $\{Z_{i,k}\}$.

We denote by x, $x \in \Re^n$, the unknown vector that is to be estimated. To aid in the estimation, a priori information is used to develop a collection of models, called a *model set*, that is parametrized by x. Typically, the random process $\{Z_k\}$, rather than the actual observed sample path, is modeled. Thus a model set can be denoted by $\{\hat{Z}_k(x); x \in X\}_{k\in\mathbb{N}}$, where for each x $\hat{Z}_k(x)$ is a model, i.e., "guess", for Z_k, and different models are obtained as x takes different values in the parameter set X.

In some applications, it is possible that the a priori information available is not sufficient to determine complete model sets for the process $\{Z_k\}$. One such specific case is regression. In this application, z_k is divided into two parts: a dependent component r_k, and an independent component u_k. The model set *only specifies* how $\{R_k\}$ depends on $\{U_k\}$. We will refer to such model sets as *regression model sets*[1] and they can be represented by $\{\hat{R}_k(x, U_i^k); x \in X\}_{k\in\mathbb{N}}$. We will study regression model sets. Note that there is no loss in generality and the traditional estimation setting is simply the extreme case when $R_k = Z_k$ and $U_k = \emptyset$.

A model set captures the information in a "convenient" form [15]. In the heat source example discussed earlier, we could have taken the model set to be the set of all rectangular sources with random intensity. A priori information may tell us that this model set contains a model which is a better approximation to the actual source than any model in the point source model set. However, one may still work with the point source model set since it may not be easy to determine the specific model in the rectangular source model set that is the best. In the context of sensor networks, "convenient" is to be interpreted as having the structure to allow distributed and recursive estimation. Thus we might possibly need to discard some a priori information and work with less-accurate model sets, compared to a centralized and non-recursive setting. This is the cost that is to be paid to obtain a distributed implementation. Specifically, we will discard any information available about the joint statistics[2] of the measurements. Thus we will consider model sets that *only* specify how $\{R_{i,k}\}$ depends on $\{U_{i,k}\}$. They can be represented as $\{\hat{R}_{i,k}(x, U_i^k);\ x \in X, i \in V\}_{k\in\mathbb{N}}$. We will refer to such model sets as *separable regression model sets*.

The estimate of the parameter is the value that determines the model in the model set that "best fits" the observed data. In this chapter, we will define "best" as the *conditional least-squares criterion*, described next. Define

$$p_{i,k+1}(x, u_i^{k+1}, r_i^k) = \mathsf{E}\big[\hat{R}_{i,k+1}(x, U_i^k) \mid U_i^{k+1} = u_i^{k+1}, \{\hat{R}_{i,\ell}(x, U_i^\ell) = r_{i,\ell}\}\big].$$

[1] When $U_k = R_{k-1}$, the model set is *auto-regressive*.

[2] Not knowing anything about the joint statistics of the sensor measurements, should not be confused with knowing that they are independent.

Essentially, $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ is the best estimate for $R_{i,k+1}$ based on the past and present measurements of the independent variable, and all past values of the dependent variable, when it is known that the true description of $\{R_{i,k+1}\}_{k\in\mathbb{N}}$ is $\{\hat{R}_{i,k}(x, U_i^k)\}_{k\in\mathbb{N}}$. The conditional least squares estimate (CLSE) of [12] is given by:

$$x_N^* = \underset{x\in X}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{1}{N} \sum_{k=1}^{N} \left(r_{i,k+1} - p_{i,k+1}(x, u_i^{k+1}, r_i^k)\right)^2. \tag{1}$$

We refer the reader to [12] for a discussion of the properties of the CLSE.[3] The CLSE estimator has also been studied as the minimum prediction error estimate (MPEE) in [15].

When the sensors make a large number of measurements, a convenient abstraction is to consider this number to be infinite. Denote, by x^* the limit of x_N^*, i.e.,

$$x^* = \underset{x\in X}{\operatorname{argmin}} \sum_{i=1}^{m} \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left(r_{i,k+1} - p_{i,k+1}(x, u_i^{k+1}, r_i^k)\right)^2. \tag{2}$$

The goal is to generate a sequence $\{x_n\}$ in a recursive and distributed manner that converges to x^*. Traditionally, the convergence is proved only when there exists a $\bar{x} \in X$, for which $\{\hat{Z}_k(\bar{x})\}$ accurately describes $\{Z_k\}$. This is restrictive and may not be reasonable, especially considering the that we would like to work with "convenient" model sets. Thus, we will study the properties under more general conditions. In the following two sections, we discuss two different classes of model sets:

- *Simple non-linear regression*. This corresponds to the case when $\hat{R}_{i,k}(x, U_i^k)$ is of the form $g_i(x, U_{i,k}) + E_{i,k+1}$, where $\{E_{i,k+1}\}$ is an i.i.d. sequence.
- *Stationary Gaussian linear state space model sets*. This corresponds to the case when $\hat{R}_{i,k}(x, U_i^k)$ is a parametrized stationary Gaussian linear state space process with input process $\{U_{i,k}\}$.

3 Simple Non-linear Regression

In the simple nonlinear regression problem, the model sets have the following form:

$$\hat{R}_{i,k}(x, U_i^k) = g_i(x, U_{i,k}) + E_{i,k}, \tag{3}$$

[3] When the function in (1) has multiple minima, then each minimum is a CLSE estimate. For example, this would be the case if less than three sensors were used in the acoustic source localization. In most cases, by increasing the number of sensors, and thus the spatial diversity of the measurements, it is possible to ensure that the estimation problem is well defined and there is a unique least squares estimate. Such an assumption would be analogous the observability condition of [10].

where $\{E_{i,k}\}$ is zero mean noise. The statistics of $\{E_{i,k}\}$ are not of any concern in determining the CLSE. For the model set in (3),

$$p_{i,k+1}(x, u_i^{k+1}, r_i^k) = g_i(x, u_{i,k+1}),$$

and note that the CLSE in (1) reduces to the well known least squares estimate (LSE),

$$x^* = \operatorname*{argmin}_{x \in X} \lim_{N \to \infty} \frac{1}{N} \sum_{k=1}^{N} \sum_{i=1}^{m} \left(r_{i,k} - g_i(x, u_{i,k+1})\right)^2.$$

3.1 Algorithms

We next review three different distributed and recursive estimation algorithms that have been proposed in literature.

3.1.1 Cyclic Incremental Recursive Algorithm

Each sensor designates another sensor as an upstream neighbor and one as a downstream neighbor so that they form a cycle. See Fig. 1 for an example. Without loss of generality, we will index the upstream neighbor of sensor i as $i + 1$, with the understanding that the upstream neighbor of sensor m is a fictitious sensor, denote as sensor 0. Between any two consecutive sampling times, one iteration of the algorithm is performed. In iteration k, sensor i receives the current iterate $z_{i-1,k}$ from sensor $i - 1$, updates the iterate using its latest measurement and passes it to its upstream neighbor, sensor i. The update rule is

$$\begin{aligned} z_{0,k} &= z_{m,k-1} = x_{k-1}, \\ z_{i,k} &= P_X \left[z_{i-1,k} - 2\alpha_k \left(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k} \right) \nabla g_i(z_{i-1,k}, u_{i,k}) \right], \end{aligned} \tag{4}$$

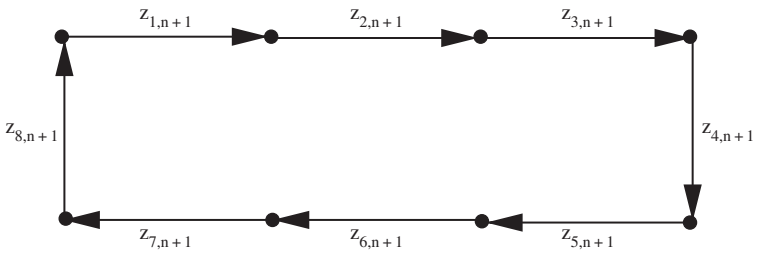

Fig. 1 A network of 8 sensors with cyclical incremental processing. The estimate is cycled through the network. The quantity $z_{i,k}$ is the intermediate value after sensor i updates at time k

where the initial iterate x_0 is chosen at random. Here, α_k is the stepsize, P_X denotes projection onto the set X and $\nabla g_i(x)$ denotes the gradient of the vector function g_i with respect to the vector parameter x. Thus, if x_j denotes the j-th component of the vector x,

$$\left[\nabla g_i(x, u_{i,k})\right]_{\ell,j} = \frac{\partial g_i^{(\ell)}(x, u_{i,k})}{\partial x_j}.$$

The cyclical incremental algorithm can be implemented only when each sensor identifies a suitable upstream and downstream neighbor. This could be achieved in a setup phase in a distributed manner using the algorithm proposed in [14]. Note the existence of a cycle is a stronger assumption than connectivity. We refer the reader to [23] for a discussion of other implementation issues.

3.1.2 Markov Incremental Recursive Algorithm

In this algorithm, the order in which the sensors update the iterate is not fixed and could be random. Let $N(i)$ denote the set of neighbors of sensor i and $|N(i)|$ denote the number of neighbors. Suppose at time $k-1$, sensor $s(k-1)$ updates and generates the estimate x_{k-1}. Then, agent $s(k-1)$ may either pass this estimate to a neighbor j, $j \in N(s(k-1))$, with probability

$$[P]_{s(k-1),s(k)} = \frac{1}{m},$$

or choose to keep the iterate with the remaining probability, in which case $s(k) = s(k-1)$. See Fig. 2 for an illustration. The update rule for this method is given by

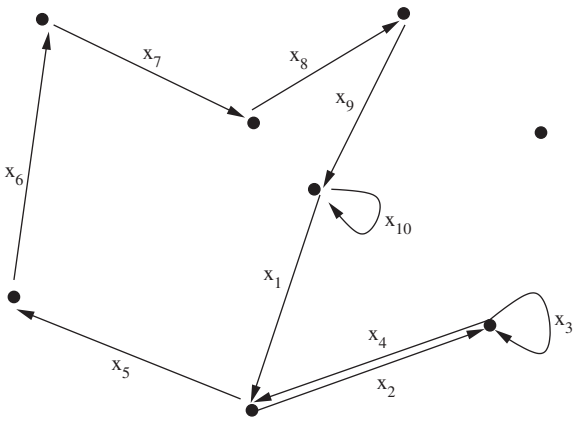

Fig. 2 A network of 8 sensors with incremental processing. The estimate is randomly passed through the network

$$x_k = P_X\left[x_{k-1} - 2\alpha_k\left(g_{s(k)}(x_{k-1}, u_{s(k),k}) - r_{s(k),k}\right)\nabla g_{s(k)}(x_{k-1}, u_{s(k),k})\right], \quad (5)$$

where $x_0 \in X$ is some random initial vector. Observe that the value of $s(k)$ depends only on the value of $s(k-1)$. Thus, the sequence of agents $\{s(k)\}$ can be viewed as a Markov chain with states $V = \{1, \ldots, m\}$ and transition matrix P. If the network topology changes with time, for example, in mobile sensor networks, then the transition matrix will also be time-varying [26]. Also note that the sensors need not sense in every slot, and a sensor needs to make a measurement only when it obtains the iterate. To save energy, sensors without an iterate may enter a sleep mode.

The Markov incremental algorithm has some advantages over the cyclic incremental algorithm. First, the algorithm does not require the setup phase that is required in the cyclic incremental algorithm. Second, the algorithm requires only connectivity of the network, which is weaker than the condition imposed on the network in the cyclic incremental algorithm. However, one can expect the Markov incremental algorithm to take more time to converge than the cyclic incremental algorithm. In one sensing interval only one update of the iterate can be performed as a sensor may choose to retain the iterate for the next iteration and will have to wait until the new measurement to update the iterate. In contrast, in the cyclic incremental algorithm in each sampling interval m updates of the iterate are performed.

In the above discussion, the performance of the algorithms as a function of time was discussed. A better comparison would be to keep the communication costs a constant, and compare one iteration of the cyclic incremental algorithm with m iterations of the Markov incremental algorithm. Even in this case one can expect the cyclic incremental algorithm to converge faster. Since the sequence in which the agents update the agents in the Markov incremental algorithm is random, the iterate may be caught in some "corner" of the network and the other agents may receive the iterate only after a large number of iterations.

3.1.3 Diffusive Nonlinear Recursive Algorithm

In this setting, each agent maintains and updates an iterate sequence. This is fundamentally different from the incremental algorithms in which a single iterate sequence is incrementally updated by the agents. We will use $w_{i,k-1}$ to denote the iterate with agent i at the end of time slot $k-1$. One iteration of the algorithm is performed in each sampling interval. Each agent receives the current iterate of its present neighbors. See Fig. 3 for an illustration. Each agent then calculates the following weighted sum $v_{i,k-1}$ given by

$$v_{i,k-1} = w_{i,k-1} + \frac{1}{m}\sum_{j\in N(i)}(w_{j,k-1} - w_{i,k-1}).$$

Sensor i then obtains its new iterate $w_{i,k+1}$ from $v_{i,k}$ according to

$$w_{i,k} = P_X\left[v_{i,k-1} - 2\alpha_k\left(g_i(v_{i,k-1}, u_{i,k}) - r_{i,k}\right)\left(\nabla g_i(v_{i,k}, u_{i,k})\right)\right]. \quad (6)$$

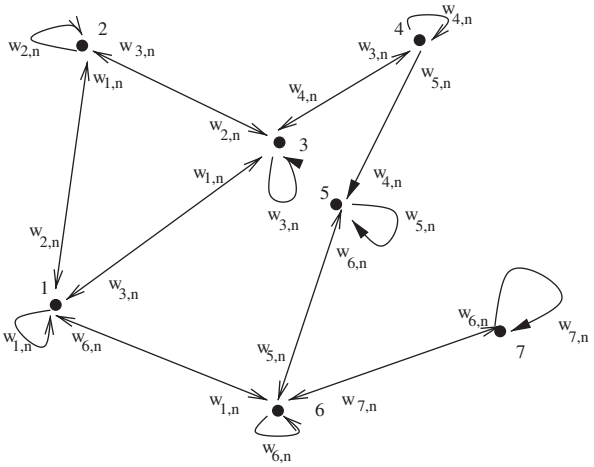

Fig. 3 A network of 8 sensors with parallel processing. Each agent shares its current iterate with its neighbors

The initial points $\{w_{i,0}\}$ are chosen randomly.

Similar to the Markov incremental algorithm, this algorithm requires the network to be only connected and does not require a setup phase. Similar to the cyclic incremental algorithm, each sensor uses its latest measurement to update in every iteration. However, note that each sensor updates a different iterate sequence. Therefore, unless the network is "densely connected" and each sensor has a large number of neighbors, one can expect the performance of the algorithm with time to be worse than that of the cyclic incremental algorithm. It is not immediately clear, however, if the performance of the distributed parallel algorithm as a function of communication costs would be better than the Markov incremental algorithm. The diffusive algorithms are closely related to consensus algorithms [9, 32], and they can be viewed as general consensus algorithms with stochastic errors.

3.2 Convergence of the Algorithms

The algorithms described above in Sect. 3.1 have been defined for arbitrary sequences $\{r_{i,k}\}$ and $\{u_{i,k}\}$. However, to study their convergence we would need to impose some restrictions on the actual measurement sequences.

Assumption 1 *The data* $\{z_k\}$ *is the sample path of an i.i.d. random process* $\{Z_k\}$.

In view of this assumption, it is possible to interpret the algorithms discussed above as stochastic approximation algorithms [3, 8, 22] and this will form the basis for the convergence results we state later. Due to Assumption 1,

$$x^* = \underset{X}{\operatorname{argmin}} \sum_{i=1}^{m} \mathsf{E}\Big[\big(R_{i,k} - g_i(x, U_{i,k})\big)^2\Big]. \tag{7}$$

Define $f_i(x) = \mathsf{E}\big[\big(R_{i,k} - g_i(x, U_{i,k})\big)\big]^2$ and $f(x) = \sum_{i=1}^{m} f_i(x)$. Observe that x^* is the minimum of $f(x)$. Suppose that the statistics of $R_{i,k}$ and $U_{i,k}$ are explicitly known to sensor i. Then, the problem simplifies to minimizing the sum of deterministic functions, where each function is known to a sensor. This distributed optimization problem has been studied in [18, 19, 25, 26] and different iterative distributed algorithms have been proposed. For example, consider the incremental gradient algorithm. In each iteration, the iterate is cycled through the network. Sensor i updates the iterate in the k-th iteration according to

$$\begin{aligned} z_{i,k} =& P_X\left[z_{i-1,k} - \alpha_k \nabla f_i(z_{i-1,k})\right] \\ =& P_X\left[z_{i-1,k} - 2\alpha_k \mathsf{E}\big[\big(g_i(z_{i-1,k}, U_{i,k}) - R_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, U_{i,k})\big)\big]\right], \end{aligned}$$

and communicates the updated iterate to sensor $i+1$. Note that algorithm requires sensor i to only know its local function f_i.

In the estimation problem, the statistics of $U_{i,k}$ and $R_{i,k}$ are not known. If they were known, the least squares estimate could have been obtained without making any measurements. Thus, sensor i cannot evaluate the gradient term $\nabla f_i(z_{i-1,k})$ that is required in the k-th iteration. Instead, if the k-th iteration is performed in the $(k+1)$-th sampling interval, i.e., after the k-th observation has been made, sensor i can approximate $\nabla f_i(z_{i-1,k})$ with an LMS style instantaneous empirical approximation, i.e,

$$\mathsf{E}\big[\big(g_i(z_{i-1,k}, U_{i,k}) - R_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, U_{i,k})\big)\big] \approx \big(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k}\big)\big(\nabla g_i(z_{i-1,k}, u_{i,k})\big).$$

Observe that the incremental algorithm with this approximation is the cyclic incremental recursive estimation algorithm in (4). Thus, in effect the cyclic incremental recursive estimation algorithm is a stochastic optimization algorithm.

In a similar manner, the Markov incremental recursive algorithm and the diffusive recursive algorithm are obtained from their deterministic counterparts discussed in [25, 26]. We next formally state the conditions that are required for the convergence of the algorithm. We refer the reader to [25, 26] for the proofs.

Theorem 1 *Consider the model set in (3). Let the set X be closed and convex, and let Assumption 1 hold. Further, let the functions $f_i(x)$ be convex and have bounded (sub)gradients on the set X. If $\{\alpha_k\}$ is chosen such that $\sum_k \alpha_k^2 < \infty$ and $\sum_k \alpha_k = \infty$. Then the following convergence results hold:*

1. *If the network topology allow a cycle that connects all the sensors, then iterates generated in (4), converge to a minimum of $f(x)$ with probability 1 and in mean square.*
2. *If the network is connected, then the iterates generated in (5) converge to a minimum of $f(x)$ in mean square.*
3. *If the network is connected, then the iterates generated in (6) converge to the same minimum of $f(x)$, for all $i \in V$, with probability 1 and in mean square.*

The convergence results in [25, 26] are more general and deal with general choices of probabilities in (5) and general weights in (6). While we have no proof, we believe that the convergence of the algorithms can be extended to the general case when $\{Z_k\}$ is required to satisfy only the conditions of exponential stability and stationarity (explained below). This has been verified for the specific case where $g_i(x, \cdot)$ is linear in x.

Theorem 1 requires the functions $f_i(x)$ to be convex. A simple case when this holds is when $g_i(x, \cdot)$ is linear in x. However, in general the condition is quite implicit and there seems to be no direct way to verify it. We next state another convergence result, which is applicable for a different class of functions. We only state the result and refer the reader to [24] for the essential ideas of the proofs.

Theorem 2 *Consider the model set (3) and let the network be connected. Additionally, if the incremental algorithm is used, let the network topology have a cycle. Let Assumption 1 hold, and let the set X be $\Re^n$, or compact and convex. Further, let the functions $f_i(x)$ be differentiable on the set X and let the gradient be Lipschitz continuous. If the set $X = \Re^n$, also assume that the gradients are bounded. If $\{\alpha_k\}$ is chosen such that $\sum_k \alpha_k^2 < \infty$ and $\sum_k \alpha_k = \infty$, then every cluster point of the iterate sequences generated in (4), (5) and (6) will be a stationary point.*

First note that the theorem guarantees only convergence to a stationary point. In addition, the conditions imposed on the set X are also stronger than the conditions in Theorem 1. However, note that the conditions imposed on the functions f_i are weak, and easier to verify. For example, a sufficient condition for convergence when X is compact is for the function $g_i(x, \cdot)$ to be twice continuously differentiable.

3.3 Effect of Quantization

Typically, the iterates are first quantized before they are communicated in wireless networks. We next study the effect of quantization on the estimation algorithms. The quantization lattice is the set

$$Q = \{(q_1\Delta, \ldots, q_n\Delta); q_j \in \mathbb{Z}\}.$$

For a vector $x \in \Re^n$, we use $Q[x]$ to denote the quantized value of x. We consider the following two types of quantizers:

- *Basic quantizer*. The quantized value of a vector $x \in X$, is the Euclidean projection of x. Thus, $Q[x] = P_Q[x]$. Observe that $\|Q[x] - x\|$ is deterministically bounded by $\frac{\sqrt{n}\Delta}{2}$ [23].
- *Dither quantizer*. The quantized value of a vector $x \in X$, is the Euclidean projection of x added with a dither signal. Thus, $Q[x] = P_Q[x + D]$, where D is a dither signal whose component are uniformly and independently chosen in $[\frac{-\Delta}{2}, \frac{\Delta}{2}]$. In this case, $\|Q[x] - x\|$, is random, statistically independent of x and uniformly distributed in $\left[\frac{-\sqrt{n}\Delta}{2}, \frac{\sqrt{n}\Delta}{2}\right]$[2, 10].

When there is quantization, it is better to use a constant stepsize than a diminishing stepsize. We motivate this through the following discussion. It seems reasonable to require the algorithms, with quantization, to converge to $Q[x^*,]$ i.e., the lattice point that is the closest to x^*. However, this is not the case even when there are no stochastic errors. To see this consider the standard gradient descent algorithm to minimize $f(x)$ over the set $X = \Re$ with a basic quantizer without any dither. The iterates are generated according,

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k).$$

Suppose $x_0 \neq Q[x^*]$, and $\alpha_k < \frac{\Delta}{2C}$, where C is the bound on the subgradient of f_i, then it can be immediately seen that $x_k = x_0$, for all k. More generally, one can conclude that stepsizes should always be large enough to push the iterate from a non-optimal lattice point, but small enough to ensure that the iterate gets caught in the optimal lattice point.

We will only discuss the quantized cyclic incremental algorithm here. Similar results can be easily obtained for quantized versions of the other two algorithms. Formally, the quantized cyclical incremental recursive estimation algorithm is given by:

$$\begin{aligned} z_{0,k} &= z_{m,k-1} = x_{k-1}, \\ z_{i,k} &= Q\left[P_X\left[z_{i-1,k} - 2\alpha_k \left(g_i(z_{i-1,k}, u_{i,k}) - r_{i,k}\right)\left(\nabla g_i(z_{i-1,k}, u_{i,k})\right)\right]\right]. \end{aligned} \quad (8)$$

It is possible to obtain bounds on that the asymptotic performance of the algorithm by extending the analysis for constant stepsize in [26] to handle quantization by using ideas similar to those in [23]. Define ν_i to be some constant such that

$$\nu_i \geq \sup_{x \in X} \mathsf{Var}\left[\left(R_i - g_i\left(x.U_i^T\right) -\right)\left(\nabla g_i\left(x, U_i\right)\right)\right].$$

Then ν_i is an upper bound on the variance of the empirical gradient estimate.

Theorem 3 *Let the Assumptions of Theorem 1 hold. Additionally, assume that X is bounded and a constant stepsize α is used. Then, with basic quantization the iterates* $\{x_k\}$ *in (8) satisfy*

$$\liminf_{k \to 0} \mathsf{E}[f(x_k)] \leq f^* + \frac{\sqrt{n}\Delta \max_{x,y \in X} \|x - y\|}{2\alpha} + \frac{\alpha}{2}\left(\sum_{i=1}^{m}\left(C_i + \nu_i + \frac{\sqrt{n}\Delta}{2\alpha}\right)\right)^2,$$

and with dither quantization the iterates satisfy

$$\liminf_{k \to 0} \mathsf{E}[f(x_k)] \leq f^* + \frac{\sqrt{n}\Delta \max_{x,y \in X} \|x - y\|}{2\alpha} + \frac{\alpha}{2}\left(\sum_{i=1}^{m}\left(C_i + \nu_i + \frac{\sqrt{n}\Delta}{\alpha\sqrt{6}}\right)\right)^2.$$

Furthermore, the bounds hold with probability 1 for $\inf_{k \geq 0} f(x_k)$.

Observe that with dither quantization it is possible to obtain a better bound. This indicates that dither quantization may result in smaller errors than basic quantization.

3.4 Special Case: Linear Regression

We next consider the special case when $g_i(x, U_{i,k}) = x^T U_{i,k}$, and $X = \Re^n$. In a centralized setting, it is possible to recursively evaluate the exact least squares, i.e, evaluate $\hat{x}_N$, from $\hat{x}_{N-1}$ and new measurements [15]. From a stochastic optimization perspective, the recursive least squares (RLS) algorithm approximates the standard Newton's gradient algorithm, where the gradient term is approximated using an instantaneous empirical approximation and a sample average for the Hessian term [15]. Attempts have been made to exploit this to develop distributed and recursive algorithms that perform better than the LMS-type algorithms discussed above. We discuss these next.

Incremental RLS algorithms can be obtained by scaling the gradient term in the incremental LMS with a matrix $L_{i,k}$. Formally,

$$z_{i,k} = z_{i-1,k} - \alpha_k L_{i,k} u_{i,k} (u_{i,k}^T z_{i-1,k} - r_{i,k}).$$

The matrix $L_{i,k}$ is to be viewed as an approximation to the Hessian in the Newton descent algorithm, which is determined recursively. If we allow the sensors to share a little more information to evaluate $L_{i,k}$, then it is possible to implement the least squares algorithm exactly, in a recursive and incremental manner [16]. Specifically, the matrix $L_{i,k}$ must have the form

$$\begin{aligned} L_{i,k} &= \frac{P_{i-1,k}}{1 + u_{i,k}^T P_{i-1,k} u_{i,k}} \\ P_{i,k} &= P_{i-1,k} - \alpha_k \frac{P_{i-1,k} u_{i,k} u_{i,k}^T P_{i-1,k}}{1 + u_{i,k}^T P_{i-1,k} u_{i,k}}. \end{aligned} \tag{9}$$

Note that the algorithm requires sensor $i-1$ to communicate to sensor i the statistic $P_{i-1,k}$, in addition to the iterate sequence. A more communication efficient incremental estimation algorithm can be obtained by replacing the $P_{i,k}$ in (9) with an approximation, $Q_{i,k}$, that is evaluated locally in the following manner

$$Q_{i,k} = Q_{i,k-1} - \frac{Q_{i,k-1} u_{i,k} u_{i,k}^T Q_{i,k-1}}{1 + u_{i,k}^T Q_{i,k-1} u_{i,k}}.$$

Note that the resulting algorithm requires the sensors to only cycle the iterates. The matrix $Q_{i,k}$ is recursively evaluated at sensor i using only local information. When α_k is fixed at 1, the estimates do not asymptotically converge to the least-squares estimate, but only to its neighborhood [30]. While it has not been verified, when

$\{\alpha_k\}$ diminishes at a suitable rate, one can expect the algorithm to converge to the least squares estimate.

Similarly, diffusive RLS algorithms can be obtained by scaling the gradient term in (6) with a matrix $L_{i,k}$ [5, 7, 33]. As stated previously, the matrix $L_{i,k}$ is the "equivalent" of the Hessian in the Newton direction, and can be evaluated in a distributed and recursive manner in different ways [7, 33]. A diffusive LMS algorithm that is quite different from the diffusive algorithms discussed so far is proposed in [31]. This algorithm uses a hierarchical network structure, in which a subset of sensors are designated as "bridge sensors" and a sensor is required to communicate only with neighbors, that are bridges. However, the algorithm converges only to a region around the optimal estimate.

3.5 Special Case: Accurate Model Sets

Next consider the case when the model set is accurate, i.e., there exists a $\bar{x}$ such that

$$R_{i,k} = g_i(\bar{x}, U_{i,k}) + E_{i,k}.$$

In this case it can be immediately seen that x^*, which is the asymptotic limit of the least squares estimators, will equal $\bar{x}$. For this case, the problem of estimating the parameter can be viewed as solving a system of equations, rather than an optimization problem. Specifically, observe that x^* is a solution to the following problem:

$$\text{Determine } x \text{ such that} \qquad \sum_{i=1}^{m} \mathsf{E}\big[R_{i,k} - g_i(x, U_{i,k}) - h_i(x)\big] = 0$$

The statistics of $R_{i,k}$ and $U_{i,k}$ are not known but we observe the samples $\{r_{i,k}\}$ and $\{u_{i,k}\}$. In a centralized setting, Robbins-Monro algorithm [28] can be used to solve the problem, by generating iterates according to:

$$x_k = x_{k-1} - \alpha_k \sum_{i=1}^{m} (r_{i,k} - g_i(x_{i,k}, u_{i,k})).$$

Distributed versions of these algorithm can developed. In the diffusive algorithms, proposed in [10], the iterates are generated according to

$$w_{i,k+1} = v_{i,k} - \alpha_k (v_{i,k} - g_i(v_{i,k}, u_{i,k})).$$

Similar cyclic and Markov incremental fixed point algorithms can also be developed. While this approach requires the stronger assumption that the model set be accurate, it is has the advantage that convergence can be obtained by imposing conditions directly on g_i and h_i [10].

4 Gaussian Linear State Space Model Sets

We next focus on the following class of model sets

$$\begin{aligned}\Theta_{i,k+1}(x) &= F_i(x)\Theta_{i,k}(x) + U_{i,k+1}^T G_i(x) + V_{i,k+1}(x)\\ \hat{R}_{i,k+1}(x, U^k) &= H_i(x)\Theta_{i,k+1}(x) + W_{i,k+1}(x),\end{aligned} \tag{10}$$

where $\{W_{i,k+1}(x)\}$ and $\{V_{i,k+1}(x)\}$ are i.i.d. random processes. The state vector $\Theta_{i,k+1}(x)$ is a vector of dimension q and the distribution of $\Theta_{i,0}(x)$ is such that the process $\{\hat{R}_{i,k}(x)\}_{k\in\mathbb{N}}$ is stationary. We will also assume that the random processes $\{W_{i,k}(x)\}_{k\in\mathbb{N}}$ and $\{V_{i,k}(x)\}_{k\in\mathbb{N}}$ are zero mean i.i.d. random sequences.[4]

Note that the (10) is the most general linear system description [15]. Linear state space models arise naturally, or as approximations to non-linear systems. For example, the auto-regressive moving average (ARMA) model set is an important special case, where

$$\hat{R}_{i,k+1}(x, R_i^k) = R_{i,k-1}g_i(x) + E_{i,k+1}(x).$$

Here $E_{i,k+1}(x) = E_{i,k}(x) + N_{i,k}(x)$, with $\{N_{i,k}(x)\}$ being an i.i.d. sequence. We refer the reader to [15] for further discussion.

For the model set in (10), the optimal predictor $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ will be the Kalman predictor. Let $K_i(x)$ be the Kalman gain for the state-space system in (10), which is determined from $D_i(x)$, $H_i(x)$, $\mathsf{Cov}\big(W_{i,k}(x)\big)$, and $\mathsf{Cov}\big(V_{i,k}(x)\big)$ as the solution to the Riccati equation [15]. Define $F_i(x) = D_i(x) - K_i(x)H_i(x)$. The Kalman predictor, $p_{i,k+1}(x, u_i^{k+1}r_i^k)$, is

$$\begin{aligned}\phi_{i,k+1}(x, u_i^{k+1}, r_i^k) =& F_i(x)\phi_{i,k}(x, u_i^k, r_i^{k-1}) + K_i(x)r_{i,k} + u_{i,k+1}^T G_i(x),\\ p_{i,k+1}(x, u_i^{k+1}, r_i^k) =& H_i(x)\phi_{i,k+1}(x, u_i^{k+1}, r_i^k).\end{aligned} \tag{11}$$

Note that to evaluate the gradient of $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ at x, u_i^{k+1} and r_i^k are required. In contrast, in the simple non-linear regression model, $p_{i,k+1}(x, u_i^{k+1}, r_i^k)$ was a function of only $u_{i,k+1}$ and we could make the algorithm recursive by using an LMS approximation. Thus, we need to make two approximations (a) an LMS-like approximation for the gradient, and (b) an approximation to make the LMS approximations recursive. These ideas are based on the recursive prediction error (RPE) algorithm of [15] and hence we call the incremental algorithm proposed below the incremental RPE (IRPE) algorithm [27]. Formally, the iterates are generated according to the following relations for $i \in V$ and $\ell = 1, \ldots, d$,

[4] We make the zero mean assumption to keep the presentation simple. The algorithm can be extended to the case when they are not zero mean.

$$
\begin{aligned}
x_{k-1} &= z_{m,k-1} = z_{0,k}, \qquad (12)\\
\begin{bmatrix} \psi_{i,k} \\ \chi_{i,k}^{(\ell)} \end{bmatrix} &= \begin{bmatrix} F_i(z_{i,k-1}) & 0 \\ \nabla^{(\ell)} F_i(z_{i,k-1}) & F_i(z_{i,k-1}) \end{bmatrix} \begin{bmatrix} \psi_{i,k-1} \\ \chi_{i,k-1}^{(\ell)} \end{bmatrix} + \begin{bmatrix} K_i(z_{i,k-1}) \\ \nabla^{(\ell)} K_i(z_{i,k-1}) \end{bmatrix} r_{i,k-1} \\
&\quad + \begin{bmatrix} G_i(z_{i,k-1}) \\ \nabla^{(\ell)} G_i(z_{i,k-1}) \end{bmatrix} u_{i,k} \\
\begin{bmatrix} \zeta_{i,k} \\ \xi_{i,k}^{(\ell)} \end{bmatrix} &= \begin{bmatrix} H_i(z_{i-1,k}) & 0 \\ \nabla H_i^{(l)}(z_{i-1,k}) & H_i(z_{i-1,k}) \end{bmatrix} \begin{bmatrix} \psi_{i,k} \\ \chi_{i,k}^{(\ell)} \end{bmatrix}, \\
\varepsilon_{i,k} &= r_{i,k} - \zeta_{i,k}, \\
z_{i,k}^{(\ell)} &= z_{i-1,k}^{(\ell)} - \alpha_k \left(\xi_{i,k}^{(\ell)}\right)^T \varepsilon_{i,k}, \\
z_{i,k} &= \begin{bmatrix} z_{i,k}^{(1)} & \dots & z_{i,k}^{(d)} \end{bmatrix}^T.
\end{aligned}
$$

The initial values for the recursion are fixed at $x_0 = x_s$, $\psi_{i,0} = \psi_{i,s}$ and $\chi_{i,0}^{(\ell)} = \chi_{i,s}^{(\ell)}$. Observe that the algorithm has a distributed and recursive implementation. In iteration k, sensor i first uses its iterates from the previous iteration, i.e., $z_{i,k-1}$, and $u_{i,k}$ to evaluate $\chi_{i,k}^{(1)}, \dots \chi_{i,k}^{(d)}$ and $\psi_{i,k}$. Sensor i then receives $z_{i-1,k}$ from sensor $i-1$ and updates it using $z_{i,k-1}$ to generate $z_{i,k}$. This is then passed to the next sensor in the cycle. Thus, sensor i only needs to store $z_{i,k}$, $\chi_{i,k}^{(1)}, \dots \chi_{i,k}^{(d)}$ and $\psi_{i,k}$ for the next iteration. The algorithm is therefore recursive and distributed. Furthermore, note that sensor i only needs to know its own system matrices $H_i(x)$, $F_i(x)$ and $G_i(x)$.

4.1 Convergence of the Algorithm

The data is assumed to satisfy the following assumption

Assumption 2 *For each $i \in V$, the sequence $\{z_k\}_{k\in\mathbb{N}}$ is the sample path of stationary and exponentially stable random process $\{Z_k\}_{k\in\mathbb{N}}$.*

Exponential stability requires z_k and z_s to be weakly related when $t \gg s$. Both these conditions are quite weak and are satisfied in practice. The precise mathematical definitions and other details are available on page 170 of [15]. We next state the convergence result. The proof is available in [27].

Theorem 4 *Let Assumption 2 hold and let the network topology allow a cycle between the sensors. For each $x \in X$, let the system in (10) be stable and let the matrix functions $F_i(x)$, $H_i(x)$ and $K_i(x)$ be twice continuously differentiable. Let the step-size α_k be such that $k\alpha_k$ converges to a positive constant. Then, the iterates $\{x_k\}$ generated by the IRPE algorithm in (12) converge to a local minimum of $f(x)$ in (2) with probability 1.*

5 Application: Determining the Source of a Diffusion Field

A diffusion field is a spatio-temporal field that follows the diffusion partial differential equation. The diffusion equation models diverse physical phenomena like the conduction of heat, dispersion of plumes in air, dispersion of odors through a medium and the migration of living organisms. We study the problem of determining the source of a diffusion field (specifically, temperature field) generated in a room.

We briefly review the literature on source identification in the diffusion equation. The point source model is a common model for diffusing sources and has been extensively used [1, 13, 17, 34]. Usually the intensity is assumed to be a constant [1, 13, 17] or instantaneous. Localization of sources with time-varying intensity have been studied in a centralized and non-recursive setting in [6, 20]. These studies consider a deterministic evolution of the leak intensity and use a continuous observation model. Most papers study that case when the medium is infinite or semi-infinite since the diffusion equation has a closed form solution in that case [13, 17]. An exception is [11] where two dimensional shaped medium is considered.

While centralized recursive source localization has received much interest [13, 17, 20, 21] there are very few papers that discuss a distributed solution. A recursive and distributed solution to the problem in a Bayesian setting is discussed in [34]. A related paper is [29] that deals with the problem of estimating the diffusion coefficient in a distributed and recursive manner. We are not aware of any prior work that solves the source localization problem using a distributed and recursive approach in a non-Bayesian setting. We refer the reader to [34] for a more detailed discussion of the literature.

We will consider two different parametrized model sets for the source (the feature of interest) and use the algorithms developed so far to determine the parameters.

5.1 Point Source and Constant Intensity Model Sets

We first consider the case when based on a priori information the model set for the source can be chosen to be the set of all point sources with constant intensity. The model set is parametrized by the source location $x = (x_1, x_2)$ and intensity I. Thus the problem of learning the source simplifies to estimating x and I. Additionally, it is also known that the warehouse is large, the initial temperature is a constant throughout the warehouse, and the thermal conductivity is large and uniform throughout the medium, and known.

To map the model set for the source to a model set for the sensor measurements we will use the diffusion equation. Let $C(s, t; x, I)$ denote the temperature at a point s at time t when the source is at x and intensity is I. For the source model set and medium, $C(s, t; x, I)$ can be approximated using the steady-state value given by [17]:

$$C(s; x, I) = \frac{I}{2\nu\pi \|s - x\|},$$

where ν is the diffusion coefficient. If s_i is the location of the i-th sensor, then the model set for its k-th measurement is

$$\hat{R}_{i,k}(x, I) = C(s_i; x, I) + N_{i,k},$$

where $N_{i,k}$ is a zero mean i.i.d. measurement noise. Thus the estimation problem is a simple nonlinear regression problem with no measurable inputs and can be solved using the algorithms developed in Sect. 3.

5.1.1 Numerical Results

We illustrate the performance of the algorithms with simulation results. In the simulation experiments the diffusion coefficient $\nu = 1$. The actual location of the source is $x^* = (37, 48)$ and the actual intensity value is 10. A network of 27 sensors is randomly deployed. The initial iterate value is fixed at (50, 50) for the source location and 5 for the intensity. The results are plotted in Figs. 4 and 5. Observe that about 200 iterations are sufficient to obtain a good estimate of the source location and 1000 iterations to obtain a good estimate of the intensity using the cyclic incremental, Markov incremental and diffusion algorithms. In addition, we observed that the convergence speed of the algorithm is affected by the initial point. If the initial points are taken very far from the actual value then there is convergence to other stationary points in the problem.

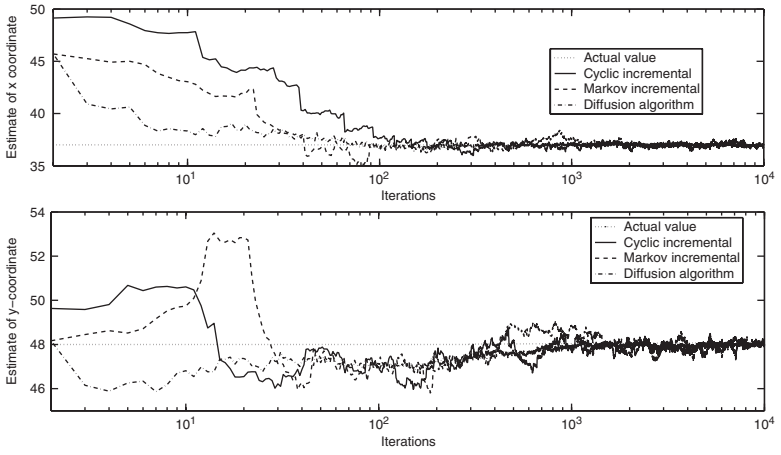

Fig. 4 Estimates of the x and y coordinates generated by the cyclic incremental, Markov incremental and diffusion gradient algorithms

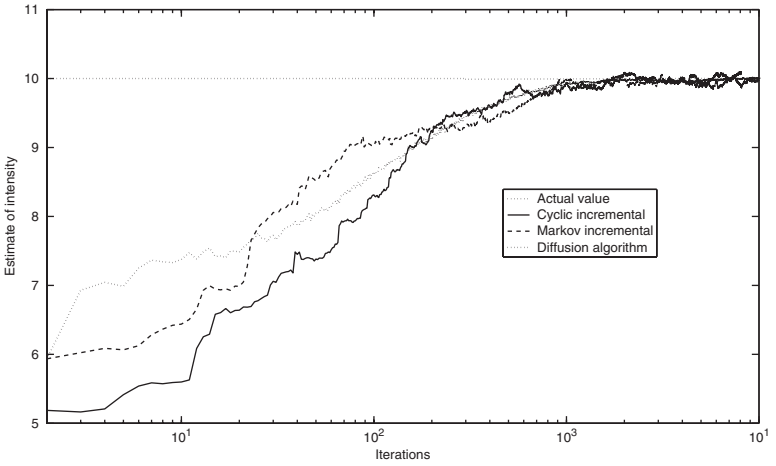

Fig. 5 Estimate of the intensity generated by the cyclic incremental, Markov incremental and diffusion gradient algorithms

5.2 Point Source and Time-Varying Intensity Model Sets

Suppose that based on a priori information, the model set that is chosen consists of all point sources that switch on at time $t = 0$ with intensity values $I(t) = I_k$, for $t \in [k-1, k)$, where $I_{k+1} = \rho I_k + S_k$. Here, ρ is a known scalar and $\{S_k\}$ is a sequence of i.i.d. Gaussian random variables with mean $(1-\rho)\mu$ and variance $(1-\rho^2)\sigma_s^2$, and $I(0)$ is Gaussian with mean μ and variance σ_s^2. Thus the model set for the source, which is the feature of interest, is parametrized by the location $x = (x_1, x_2)$. Note that $\{I_k\}$ does not parametrize the model set and is an incidental random process whose statistics are specified. Thus we only need to estimate the source location.

Additionally the following is known about the medium. Let D denote the room and ∂D denote the boundary of the room. The medium is uniform throughout the room. The temperature at the boundaries of the room is always a constant and at time $t = 0$, the temperature is modeled to be a constant everywhere in the room. Without loss of generality, we fix the constant temperature to be 0, i.e., $C(\cdot, 0; \cdot) = 0$.

For the above source model set and medium, the temperature at a point y at time t is

$$\frac{\partial C(y,t;x)}{\partial t} = \nu \nabla^2 C(y,t;x) + I(t)\bar{\delta}(y-x), \tag{13}$$

with the initial and boundary conditions

$$C(s,0;x) = 0 \text{ for all } s \in D,$$
$$C(s,t;x) = 0 \text{ for all } t \geq 0 \text{ and } s \in \partial D.$$

Here, ν is the medium conductivity, ∇^2 is the Laplacian differential operator and $\bar{\delta}$ is the Dirac delta function. Let s_i be the location of the i-th sensor. Then the model set for the sensor measurements is

$$\hat{R}_{i,k}(x) = C(s_i, k; x) + N_{i,k},$$

where $N_{i,k}$ is a zero mean i.i.d. Gaussian measurement noise with known variance σ_n^2. By using Green's technique to solve the differential equations, it is possible to obtain an approximate state-space description for each sensor's observation process. The estimation problem can now be solved using the techniques developed in Sect. 4. We refer the reader to [27] for a detailed analysis.

5.2.1 Numerical Results

We illustrate the performance of the algorithm with simulation results for the problem discussed above. In the simulation experiments, $l_1 = l_2 = 100$ and diffusion coefficient $\nu = 1$. The actual location of the source is $x^* = (37, 48)$. The value of $\mu = 1000$, $\rho = 0.95$ and $\sigma_s^2 = 1$. A network of 27 sensors is deployed. To ensure complete coverage of the sensing area, we first placed 9 sensors on a grid and then randomly deployed 3 sensors in the immediate neighborhood of each of the 9 sensors. The network is shown in Fig. 6. The sampling interval is 10 time units and the measurement noise variance is set to 0.1.

Observe from Fig. 7 that the IRPE algorithm does not converge to the correct parameter value. This is because, in the absence of convexity, the algorithms in general are only guaranteed to converge to a stationary point and necessarily the global minimum of the cost function.

One way to improve the structure of the cost function is to partially model the dependence between sensor measurements. The idea is to group sensors in clusters. The dependence between the sensor measurements within a cluster is explicitly modeled, while across clusters no model is assumed. Each sensor could then pass its measurements to a cluster head and the distributed algorithms discussed could be run between the cluster heads. Note that in this case the $p_{i,k+1}$ term in the conditional least squares term in (1) should be the best estimate for $R_{i,k+1}$ based on the past and present measurements of the independent variable, and all past values of the dependent variable, of *all the sensors in the cluster*. Therefore, to improve the topology of the cost function the network is divided into 9 clusters of size 3. The cluster heads are the sensors marked by circles in Fig. 6. At the beginning of each slot, a cluster head collects the measurements from the sensors in the cluster. A total of 1000 iterations of the IRPE is run between the cluster heads and the performance is compared with 1000 iterations of the RPE. Observe from Fig. 7 that the algorithm now converges to the correct parameter value.

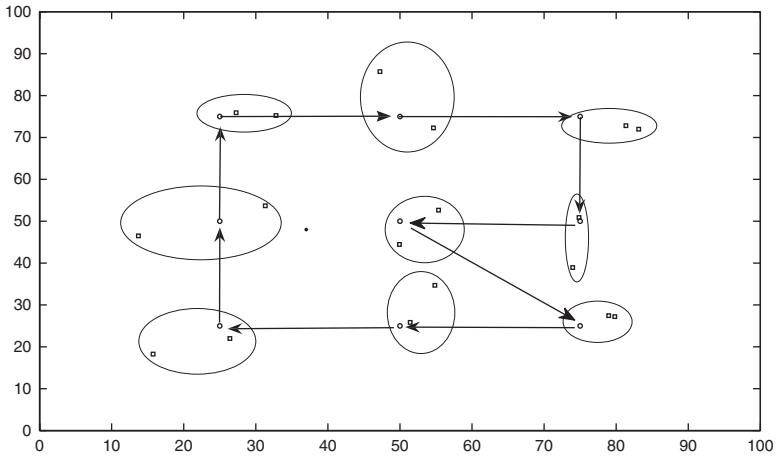

Fig. 6 A network of 27 sensors. The *small circles* denote the cluster heads and the *squares* denote the sensors. The source is represented by a *dot*. The *arrows* indicate the order in which the iterates are passed in the cluster IRPE

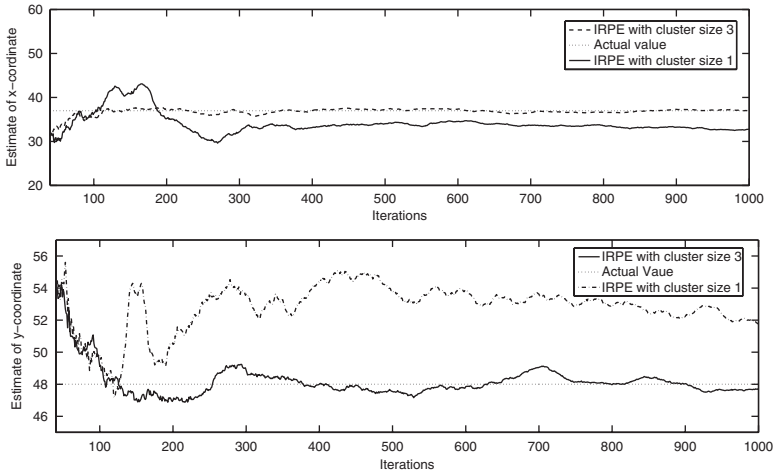

Fig. 7 Estimates of the x and y coordinates generated by the standard IRPE and the cluster IRPE. Observe that the standard IRPE converges to a stationary point while the cluster IRPE converges to the correct source location

6 Discussion

In this chapter we have focused on least squares parameter estimation. All of the estimation algorithms were based on stochastic gradient optimization algorithms. Similar ideas can be used to obtain weighted, and robust least-squares estimators, for the case where the criterion is not necessarily quadratic. If the sensor measurements are independent across sensors, the ideas presented here can be extended to

obtain distributed and recursive maximum likelihood estimators; the independence across sensors provides the additive form for the estimation cost function, which makes it possible to use the distributed stochastic algorithms. Related work, which we did not discuss in this chapter, is that involving recursive Bayesian estimation over networks. In [4], the asymptotic convergence of optimal Bayesian estimates is investigated, under the premise that the network is not very complicated and each sensor can keep track of the total information flow. In [34], cyclical incremental Bayesian estimation algorithms are discussed.

There are a number of avenues for future extensions. Perhaps, the most immediate is the development and analysis of Markov incremental and diffusive algorithms for Gaussian state space model set. Another avenue involves relaxing the assumption that the sensors sense and process in a synchronous manner, since this assumption may not be valid in large networks. At a broader level, it is of interest to develop recursive and distributed estimation algorithms for estimation in other model sets that the two specific model sets that we described in the chapter.

Acknowledgement This work was supported in part by a Vodafone Fellowship, Motorola, and the National Science Foundation, under CAREER grant CMMI 07-42538.

References

1. Alpay, M.E., Shor, M.H.: Model-based solution techniques for the source localization problem. IEEE Transactions on Control Systems Technology **8**(6), (2000)
2. Aysal, T., Coates, M., Rabbat, M.: Distributed average consensus with dithered quantization. IEEE Transactions on Signal Processing **56**(10), 4905–4918 (2008)
3. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and Distributed Computation: Numerical Methods. Athena Scientific, Belmont, MA (1997)
4. Borkar, V., Varaiya, P.P.: Asymptotic agreement in distributed estimation. IEEE Transactions on Automatic Control **27**, 650–655 (1982)
5. Calafiore, G., Abrate, F.: Distributed linear estimation over sensor networks. International Journal of Control **82**(5), 868–882 (2009)
6. Cannon, J.R., DuChateau, P.: Structural identification of an unknown source term in a heat equation. Inverse Problems **14**, 535–551 (1998)
7. Cattivelli, F.: Diffusion recursive least-squares for distributed estimation over adaptive networks. IEEE Transactions on Signal Processing **56**(5), 1865–1877 (2008)
8. Ermoliev, Y.: Stochastic quasi-gradient methods and their application to system optimization. Stochastics **9**(1), 1–36 (1983)
9. Jadbabaie, A., Lin, J., Morse, S.: Coordination of groups of mobile autonomous agents using nearest neighbor rules. IEEE Transactions on Automatic Control **48**(6), 998–1001 (2003)
10. Kar, S., Moura, J., Ramanan, K.: Distributed parameter estimation in sensor networks: nonlinear observation models and imperfect communication. Submitted, available at http://arxiv.org/abs/0809.0009 (2008)
11. Khachfe, A.R., Jarny, Y.: Estimation of heat sources within two dimensional shaped bodies. In: Proceedings of 3rd International Conference on Inverse Problems in Engineering (1999)
12. Klimko, L., Nelson, P.: On conditional least squares estimation for stochastic processes. The Annals of Statistics **6**(3), 629–642 (1978)
13. Levinbook, Y., Wong, T.F.: Maximum likelihood diffusive source localization based on binary observations. In: Conference Record of the Thirty-Eighth Asilomar Conference on Signal, Systems and Computers (2004)

14. Levy, E., Louchard, G., Petit, J.: A distributed algorithm to find Hamiltonian cycles in G(n,p) random graphs. Lecture Notes in Computer Science **3405**, 63–74 (2005)
15. Ljung, L., Söderström, T.: Theory and Practice of Recursive Identification. The MIT Press, Cambridge, MA (1983)
16. Lopes, C.G., Sayeed, A.H.: Incremental adaptive strategies over distributed networks. IEEE Transactions on Signal Processing **55**(8), 4064–4077 (2007)
17. Matthes, J., Gröll, L., Keller, H.B.: Source localization for spatially distributed electronic noses for advection and diffusion. IEEE Transactions on Signal Processing **53**(5), 1711–1719 (2005)
18. Nedić, A., Bertsekas, D.P.: Incremental subgradient method for nondifferentiable optimization. SIAM Journal of Optimization **12**, 109–138 (2001)
19. Nedić, A., Ozdaglar, A.: Distributed sub-gradient methods for multi-agent optimization. Transactions on Automatic Control **54**(1), 48–61 (2009)
20. Niliot, C.L., Lefévre, F.: Multiple transient point heat sources identification in heat diffusion: application to numerical two- and three-dimensional problems. Numerical Heat Transfer **39**, 277–301 (2001)
21. Piterbarg, L.I., Rozovskii, B.L.: On asymptotic problems of parameter estimation in stochastic PDE's: the case of discrete time sampling. Mathematical Methods of Statistics **6**, 200–223 (1997)
22. Polyak, B.T.: Introduction to Optimization. Optimization Software Inc., New York (1987)
23. Rabbat, M.G., Nowak, R.D.: Quantized incremental algorithms for distributed optimization. IEEE Journal on Select Areas in Communications **23**(4), 798–808 (2005)
24. Ram, S.S., Nedić, A., Veeravalli, V.V.: Asynchronous gossip algorithms for stochastic optimization. Submitted to IEEE CDC (2000)
25. Ram, S.S., Nedić, A., Veeravalli, V.V.: Distributed stochastic subgradient algorithm for convex optimization. Submitted to SIAM Journal on Optimization. Available at http://arxiv.org/abs/0811.2595 (2008)
26. Ram, S.S., Nedić, A., Veeravalli, V.V.: Incremental stochastic sub-gradient algorithms for convex optimization. SIAM Journal on Optimization **20**(2), 691–717 (2009)
27. Ram, S.S., Veeravalli, V.V., Nedić, A.: Distributed and recursive parameter estimation in parametrized linear state-space models. To appear in IEEE Transactions on Automatic Control
28. Robbins, H., Monro, S.: A stochastic approximation method. Annals of Mathematical Statistics **22**(3), 400–407 (1951)
29. Rossi, L., Krishnamachari, B., Kuo, C.C.J.: Distributed parameter estimation for monitoring diffusion phenomena using physical models. In: First IEEE International Conference on Sensor and Ad Hoc Communications and Networks (2004)
30. Sayed, A., Lopes, C.: Distributed recursive least-squares strategies over adaptive networks. In: 40th Asilomar Conference on Signals, Systems and Computers, pp. 233–237 (2006)
31. Schizas, I., Mateos, G., Giannakis, G.: Stability analysis of the consensus-based distributed LMS algorithm. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3289–3292 (2008)
32. Tsitsiklis, J.N.: Problems in decentralized decision making and computation. Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA (1984)
33. Xiao, L., Boyd, S., Lall, S.: A space-time diffusion scheme for peer-to-peer least-square estimation. In: Fifth International Conference on Information Processing in Sensor Networks, pp. 168–176 (2006)
34. Zhao, T., Nehorai, A.: Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks. IEEE Transactions on Signal Processing **55**, 4669–4682 (2007)

http://www.springer.com/978-3-642-01340-9

Sensor Networks
Where Theory Meets Practice
Ferrari, G. (Ed.)
2009, XIV, 394 p. 144 illus., Hardcover
ISBN: 978-3-642-01340-9