Distributed and Recursive Parameter Estimation in Parametrized Linear State-Space Models

S. Sundhar Ram, V. V. Veeravalli, and A. Nedić

Abstract

We consider a network of sensors deployed to sense a spatio-temporal field and infer parameters of interest about the field. We are interested in the case where each sensor's observation sequence is modeled as a state-space process that is perturbed by random noise, and the models across sensors are parametrized by the *same* parameter vector. The sensors collaborate to estimate this parameter from their measurements, and to this end we propose a distributed and recursive estimation algorithm, which we refer to as the incremental recursive prediction error algorithm. This algorithm has the distributed property of incremental gradient algorithms and the on-line property of recursive prediction error algorithms. We study the convergence behavior of the algorithm and provide sufficient conditions for its convergence.

I. Introduction

A sensor network consists of sensors that are spatially deployed to make observations about a process or field of interest. If the process has a temporal variation, the sensors also obtain observations sequentially in time. An important problem in such networks is to use the spatially and temporally diverse measurements collected by the sensors locally to estimate something of interest about the underlying field. This estimation activity could either be the network's main objective, or could be an intermediate step such as in control applications where the sensors are also coupled with actuators.

The first two authors are with the Dept. of Electrical and Computer Engg., University of Illinois at Urbana-Champaign. The third author is with the Dept. of Industrial and Enterprise Systems Engg., University of Illinois at Urbana-Champaign. They can be contacted at {ssriniv5,vvv,angelia}@illinois.edu. This work was partially supported by a research grant from Vodafone and partially by a research grant from Motorola.

We propose a *distributed and recursive* estimation procedure, which is suitable for in-network processing. Each sensor locally processes its own data and shares only a summary of this data with other sensors in each time slot. The sensors form a cycle and update incrementally, whereby each sensor updates the estimate using its local information and the received estimate from its upstream neighbor, and passes the updated estimate to its downstream neighbor. Such an incremental computational model is a recognized technique to reduce the total in-network communication [1]–[3]. We refer the reader to [1] for a discussion of implementation issues. Furthermore, the sensor updates are generated recursively from every new measurement using only a summary statistic of the past measurements. This has two benefits. Firstly, the network has a (possibly coarse) estimate at all times, which is important in applications that require the network to react immediately to a stimulus and make decisions on-line. An additional benefit is that each sensor can purge its old measurements periodically, thereby reducing the memory requirements at the sensors.

We are interested in the case where each sensor's observation sequence is modeled as a state-space process that is perturbed by random noise, and the models across sensors are parametrized by the *same* unknown parameter vector. The network goal is to estimate this unknown parameter using the sensor observations. State-space models arise in many applications, directly, or as linear approximations to non-linear models [4]. The estimation criterion that we use is a direct extension of the recursive prediction error (RPE) criterion of [4] to the multi-sensor case. We call it the *incremental recursive prediction error* (IRPE) criterion. We propose an algorithm that builds on the incremental gradient algorithm in the same way the RPE algorithm of [4] builds on the standard gradient algorithm. We call the algorithm the IRPE algorithm.

Our main contribution is the study of the convergence properties of the IRPE algorithm. The algorithm is essentially an incremental stochastic gradient algorithm, in which the stochastic errors are due to the recursive approximation. The convergence of incremental stochastic gradient algorithms has been established only under a certain form of independence between the iterates ([5], and references therein). In the IRPE algorithms the stochastic errors are correlated across iteration and this correlation cannot be explicitly characterized. Thus, the results from existing literature are not directly applicable. To analyze the method, we consider a *hypothetical centralized system*, and prove that the IRPE criterion is essentially the RPE criterion for the hypothetical centralized system and the IRPE algorithm is the RPE algorithm applied to that hypothetical centralized system. The convergence then follows from the convergence results of the RPE algorithm in [4]. The connection between the IRPE algorithm and the RPE algorithm of the hypothetical system is not obvious, and the theoretical value of this work is in proving this connection.

The problem of centralized recursive estimation in linear state-space models is an old problem in system identification. We refer the interested reader to [4] for a survey of these methods for linear state-space models. The problem has also generated considerable interest in the neural networks community where the EM algorithm is used as a tool to learn the parameters [6]. A related algorithm is the *parallel recursive prediction error* algorithm proposed in [7] that updates the components of the parameter vector in parallel.

The literature on incremental and recursive estimation is limited. This paper extends our earlier work [5], where we considered the problem of recursive and incremental estimation for non auto-regressive stationary models. To the best of our knowledge, there is only one other related study [2] that deals with incremental and recursive estimation. In this, incremental versions of the least mean square (LMS) and recursive least squares (RLS) algorithms are proposed to estimate unknown parameters in a simple linear regression model. The algorithm in this paper can be viewed as an extension of the incremental LMS to the more general state space model. Equivalently stated, when the algorithm in this paper is applied to the linear regression model it simplifies to the incremental LMS.

The rest of the paper is organized as follows. We formulate the problem, and introduce our notation in Section II. We then discuss the standard recursive prediction error algorithm [4] and the incremental gradient algorithm of [8] in Section III. These algorithms are at the heart of our distributed algorithm presented in Section IV, where we also state our main convergence result. We discuss the proof for the convergence of the algorithm in Appendix A. We conclude in Section V.

II. Problem Formulation

We consider a network of m sensors, indexed $1, \ldots, m,$ deployed to sense a spatio-temporal diverse field to determine the value of some quantity of interest, denoted by x, $x \in \Re^d$. We will sometimes find it convenient to use $\mathcal{I}$ denote the set of sensors, i.e., $\mathcal{I} := \{1, \ldots, m\}$. We assume that time is slotted and each sensor sequentially senses the field once in every time slot.

We denote by $r_i(k)$ the actual measurement collected by sensor i at time slot k, and we assume that $r_i(k) \in \Re^p$. The goal is to use the sensor measurements to estimate x.

To aid in the estimation process each sensor has a model for the dependence between its measurements and the unknown parameter x. Usually, the model is determined from *a priori* information available about the system. We will use denote $R_i(k;x)$ to denote the model for $r_i(k)$ and consider stochastic models in which $\{R_i(k;x)\}$ has the following dynamics

$$\begin{aligned} \Theta_i(k+1;x) &= D_i(x)\Theta_i(k;x) + W_i(k;x), \\ R_i(k+1;x) &= H_i\Theta_i(k+1;x) + V_i(k+1). \end{aligned} \tag{1}$$

The state vector $\Theta_i(k+1;x)$ is a vector of dimension q. We impose the following assumptions on the system and observation models.

A.1 The processes $\{W_i(k;x)\}$ and $\{V_i(k)\}$ are zero mean i.i.d. random sequences and the matrix function $D_i(x)$ is twice differentiable.

A.2. The quantities $D_i(x)$, H_i, $\mathsf{E}[\Theta_i(0;x)]$, $\mathsf{Cov}(W_i(0;x))$, $\mathsf{Cov}(W_i(k;x))$ and $\mathsf{Cov}(V_i(k))$ are available at sensor i.

A.3. At all the sensors a closed and convex set X is available such that the system in (1) is stable, observable and controllable for all $x \in X$.

A.4. The sequence $\{r_{i,k}\}$ is asymptotically mean stationary and exponentially stable.

Note that X may even be the entire space $\Re^d$. Asymptotic stationarity means that if we view the sequence $\{r_i(k)\}$ as the realization of a random process then that process must in the limit exhibit stationarity. Exponential stability essentially implies that what happens at time slot s has very little influence on what happens at time slot t, when $t \gg s$. See page 170 of [4] for the precise mathematical definitions of asymptotically mean stationary and exponentially stable. A sufficient condition for (A.4) is the existence of a $x^* \in X$ such that $\{r_i(k)\}$ can be viewed as a sample path of $\{R_i(k;x^*)\}$, which additionally also has finite fourth moments (page 172 of [4]).

The problem is to estimate the parameter x from the collection of sensor measurements $\{r_i(k)\}$ with an algorithm that is:

1) *Distributed:* Sensor i does not share its raw measurements $\{r_i(k)\}$ with any other sensor.
2) *Recursive:* At all times, sensor i stores only a summary statistic of a constant size, i.e., the size of the statistic does not increase with the number of measurements collected by

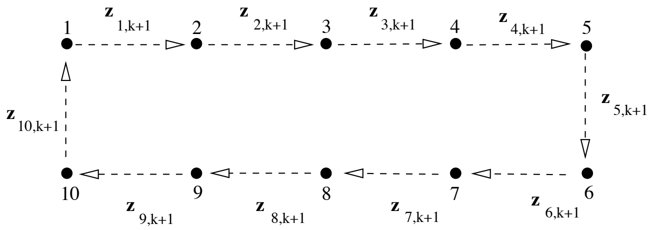

Fig. 1. A network of 10 sensors with incremental processing. The estimate is cycled through the network. The quantity $z_{i,k+1}$ is the intermediate value after the sensor i update at time slot $k+1$

the sensor.

We make the following remark. At this point, we do not assume any knowledge on the joint statistics of $\Theta_i(k+1;x)$ and $\Theta_j(k+1;x)$. They might be independent random processes or even the same random process. Even if information about the dependencies between the random processes is available we do not use it, with the understanding that this is the loss in efficiency that we suffer in order to obtain a distributed algorithm.

III. Preliminaries

To make the paper self contained we briefly discuss the incremental gradient algorithm of [8] and the RPE algorithm of [4].

A. Incremental gradient descent algorithm

The incremental gradient algorithm can be used to solve optimization problems of the form

$$\min_{x\in X}\sum_{i=1}^{m} f_i(x),$$

when the function f_i is known only to sensor i. In this algorithm, the iterates are generated according to

$$\begin{aligned} x_k &= z_{m,k} = z_{0,k+1}, \\ z_{i,k+1} &= \mathcal{P}_X\left[z_{i-1,k} - \alpha_{k+1}\nabla f_i(z_{i-1,k})\right]. \end{aligned} \tag{2}$$

Here, the scalar $\alpha_{k+1} > 0$ is the step-size, $\mathcal{P}_X$ denotes the projection onto the set X and ∇f_i denotes the gradient of the function f_i. In the k-th iteration sensor i receives the iterate $z_{i-1,k+1}$ from sensor $i-1$, incrementally updates it using the gradient of the locally available function f_i

and passes the updated iterate to the sensor $i+1$. See Fig. 1 for an illustration. The convergence of the incremental gradient method has been studied in [8], and references therein, under different assumptions on the functions $f_i(x)$ and the step-size rules.

B. Kalman predictor

We write $R_i^k(x)$ to denote the collection of random variables $\{R_i(1;x),\dots,R_i(k;x)\}$, which should be viewed as a collection of random variables parametrized by x and *not as a function of* x. Furthermore, in line with our notation, r_i^k denotes the collection $\{r_i(1),\dots,r_i(k)\}$, and r^k denotes the collection $\{r_1^k,\dots,r_m^k\}$.

For $x \in X$, we assumed that the system in (1) is stable, observable and controllable. The Kalman gain for the system therefore converges to a finite time-invariant value [9]. Let $G_i(x)$ be the Kalman gain for the state-space system in (1), which is determined from $D_i(x), H_i$, $\mathsf{Cov}(W_i(k;x))$, and $\mathsf{Cov}(V_i(k))$ as the solution to the Riccati equation [4]. Define $F_i(x) = D_i(x) - G_i(x)H_i$. The Kalman predictor, $g_{i,k+1}(x;r_i^k)$, is defined as

$$\begin{aligned} \phi_{i,k+1}(x;r_i^k) =& F_i(x)\phi_{i,k}(x;r_i^{k-1}) + G_i(x)r_i(k), \\ g_{i,k+1}(x;r_i^k) =& H_i\phi_{i,k+1}(x;r_i^k), \end{aligned} \tag{3}$$

with $\phi_{i,0}(x;r_i^0) = \mathsf{E}[\Theta_i(0;x)]$. For later reference we next determine the gradient of $g_{i,k+1}(x;r_i^k)$. Let $x^{(\ell)}$ denote the ℓ-th component of x, and define

$$\begin{aligned} \zeta_{i,k}^{(\ell)}(x;r_i^{k-1}) &= \tfrac{\partial \phi_{i,k}(x;r_i^{k-1})}{\partial x^{(\ell)}}, & \nabla^{(\ell)}F_i(x) &= \frac{\partial F_i(x)}{\partial x^{(\ell)}}, \\ \eta_{i,k}^{(\ell)}(x;r_i^{k-1}) &= \tfrac{\partial g_{i,k}(x;r_i^{k-1})}{\partial x^{(\ell)}}, & \nabla^{(\ell)}G_i(x) &= \frac{\partial G_i(x)}{\partial x^{(\ell)}}. \end{aligned}$$

Thus the gradient $\nabla g_{i,k}(x;r_i^{k-1})$ is the $p \times d$ matrix,

$$\nabla g_{i,k}(x;r_i^{k-1}) = \begin{bmatrix} \eta_{i,k}^{(1)}(x;r_i^{k-1}) & \dots & \eta_{i,k}^{(d)}(x;r_i^{k-1}) \end{bmatrix}. \tag{4}$$

By differentiating in (3), we can immediately see that

$$\begin{aligned} \begin{bmatrix} \phi_{i,k+1}(x;r_i^k) \\ \zeta_{i,k+1}^{(\ell)}(x;r_i^k) \end{bmatrix} &= \begin{bmatrix} F_i(x) & 0 \\ \nabla^{(\ell)}F_i(x) & F_i(x) \end{bmatrix} \begin{bmatrix} \phi_{i,k}(x;r_i^{k-1}) \\ \zeta_{i,k}^{(\ell)}(x;r_i^{k-1}) \end{bmatrix} + \begin{bmatrix} G_i(x) \\ \nabla^{(\ell)}G_i(x) \end{bmatrix} r_i(k), \\ \begin{bmatrix} g_{i,k+1}(x;r_i^k) \\ \eta_{i,k+1}^{(\ell)}\left(x;r_i^k\right) \end{bmatrix} &= \begin{bmatrix} H_i(x) & 0 \\ 0 & H_i(x) \end{bmatrix} \begin{bmatrix} \phi_{i,k+1}(x;r_i^k) \\ \zeta_{i,k+1}^{(\ell)}(x;r_i^k) \end{bmatrix}. \end{aligned} \tag{5}$$

C. *RPE criterion and algorithm*

We illustrate the RPE criterion and algorithm of [4] by using it to estimate x *only* using sensor i's measurements $\{r_i(k)\}$. Thus, there is no collaboration with the other agents. For this system, the recursive prediction error criterion is

$$f_i(x) = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left\| r_i(k) - g_{i,k}(x; r_i^{k-1}) \right\|^2 . \tag{6}$$

Note that under assumption (A.4) on the observation sequence $\{r_i(k)\}$, the limit on the RHS of (6) depends only on x and not on $\{r_i(k)\}$. The RPE algorithm generates a sequence of iterates $\{x_k\}$ that converges to a local minimum of the function $f_i(x)$. The RPE algorithm is essentially a gradient projection algorithm with stochastic errors. Suppose the standard gradient projection algorithm is used to minimize $f_i(x)$, then the iterates are generated according to

$$x_{k+1} \;=\; \mathcal{P}_X \left[x_k - \alpha_{k+1} \nabla f_i(x_k) \right].$$

The iterates of the RPE algorithm are obtained by approximating $\nabla f_i(x_k)$ to make the algorithm recursive. The approximation involves: (a) an LMS-like approximation for the gradient, and (b) an approximation to make the LMS approximations recursive. If the model for the measurements is a simple regression model then the LMS approximation itself is recursive and approximation (b) is not recursive. Thus, the RPE generalizes the LMS algorithm to state-space systems. We refer the reader to [4] for the details of the algorithm. The final algorithm can be states as follows

$$\begin{aligned}
\begin{bmatrix} h_{k+1} \\ \xi^{(\ell)}_{k+1} \end{bmatrix} &= \begin{bmatrix} H_i & 0 \\ 0 & H_i \end{bmatrix} \begin{bmatrix} \psi_{k+1} \\ \chi^{(\ell)}_{k+1} \end{bmatrix}, \\
\epsilon_{k+1} &= r(k+1) - h_{k+1}, \\
\underline{x}^{(\ell)}_{k+1} &= x^{(\ell)}_{k} - \alpha_{k+1} \left(\xi^{(\ell)}_{k+1} \right)^T \epsilon_{k+1}, \\
\underline{x}_{k+1} &= \left[\underline{x}^{(1)}_{k+1} \;\; \cdots \;\; \underline{x}^{(d)}_{k+1} \right]^T, \\
x_{k+1} &= \mathcal{P}_X \left[\underline{x}_{k+1} \right], \\
\begin{bmatrix} \psi_{k+2} \\ \chi^{(\ell)}_{k+2} \end{bmatrix} &= \begin{bmatrix} F_i(x_{k+1}) & 0 \\ \nabla^{(\ell)} F_i(x_{k+1}) & F_i(x_{k+1}) \end{bmatrix} \begin{bmatrix} \psi_{k+1} \\ \chi^{(\ell)}_{k+1} \end{bmatrix} + \begin{bmatrix} G_i(x_{k+1}) \\ \nabla^{(\ell)} G_i(x_{k+1}) \end{bmatrix} r(k+1).
\end{aligned} \tag{7}$$

Here $l = 1, \ldots, d$. The algorithm is initialized with values for $\psi_1, \chi_1^{(\ell)}$ and x_0. Observe that to update x_k the algorithm requires only $r(k+1)$, $\chi^{(1)}_{k+1}, \ldots, \chi^{(d)}_{k+1}$ and ψ_{k+1}, and therefore, it is recursive.

The following theorem describes the convergence properties of the RPE algorithm (Theorem 4.3 of [4]).

Theorem 1: Let (A.1)—(A.4) hold. Moreover, let the step-size α_k *be such that* $k\alpha_k$ *converges. Then, the iterates* x_k *generated by the RPE in (7) converge to a local minimum of* $f_i(x)$ *in (6) over the set* X*, with probability* 1.

IV. IRPE Algorithm

The direct extension of the RPE criterion in (6) to the case when all the m sensors cooperate to estimate x is

$$f(x) = \sum_{i=1}^{m} f_i(x) = \sum_{i=1}^{m} \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \left\| r_i(k) - g_{i,k}(x; r_i^{k-1}) \right\|^2 . \tag{8}$$

We refer to this criterion as the *incremental recursive prediction error criterion*. The function f_i is potentially available only to sensor i. If, the incremental gradient descent algorithm is used to minimize the function $f(x)$ then the iterates are generated according to

$$\begin{aligned} x_k &= z_{m,k} = z_{0,k+1}, \\ z_{i,k+1} &= \mathcal{P}_X \left[z_{i-1,k} - \alpha_{k+1} \nabla f_i(z_{i-1,k}) \right] . \end{aligned} \tag{9}$$

The IRPE can be viewed as incremental gradient descent with stochastic errors that are generated when the term $\nabla f_i(z_{i-1,k})$ is approximated using the same two approximations that were used in the RPE algorithm. The first is the LMS like approximation, and the second is the recursive approximation to make the LMS approximation recursive. If only the LMS approximation is made, which would be the case in a simple linear regression problem, the IRPE algorithm simplifies to the incremental LMS algorithm of [2].

Formally, the iterates are generated according to the following relations for $i \in \mathcal{I}$ and $\ell =$

$1, \ldots, d$,

$$\begin{aligned}
x_k &= z_{m,k} = z_{0,k+1}, \\
\begin{bmatrix} h_{i,k+1} \\ \xi^{(\ell)}_{i,k+1} \end{bmatrix} &= \begin{bmatrix} H_i & 0 \\ 0 & H_i \end{bmatrix} \begin{bmatrix} \psi_{i,k+1} \\ \chi^{(\ell)}_{i,k+1} \end{bmatrix}, \qquad (10) \\
\epsilon_{i,k+1} &= r_i(k+1) - h_{i,k+1}, \qquad (11) \\
\underline{z}^{(\ell)}_{i,k+1} &= z^{(\ell)}_{i-1,k+1} - \alpha_{k+1} \left(\xi^{(\ell)}_{i,k+1} \right)^T \epsilon_{i,k+1}, \qquad (12) \\
\underline{z}_{i,k+1} &= \left[\underline{z}^{(1)}_{i,k+1} \quad \ldots \quad \underline{z}^{(d)}_{i,k+1} \right]^T, \qquad (13) \\
z_{i,k+1} &= \mathcal{P}_X[\underline{z}_{i,k+1}], \qquad (14) \\
\begin{bmatrix} \psi_{i,k+2} \\ \chi^{(\ell)}_{i,k+2} \end{bmatrix} &= \begin{bmatrix} F_i(z_{i,k+1}) & 0 \\ \nabla^{(\ell)} F_i(z_{i,k+1}) & F_i(z_{i,k+1}) \end{bmatrix} \begin{bmatrix} \psi_{i,k+1} \\ \chi^{(\ell)}_{i,k+1} \end{bmatrix} + \begin{bmatrix} G_i(z_{i,k+1}) \\ \nabla^{(\ell)} G_i(z_{i,k+1}) \end{bmatrix} r_i(k+1) \qquad (15)
\end{aligned}$$

The initial values for the recursion are fixed at $x_0 = x_s$, $\psi_{i,1} = \psi_{i,s}$ and $\chi^{(\ell)}_{i,1} = \chi^{(\ell)}_{i,s}$. To see that the algorithm has a distributed and recursive implementation assume sensor $i-1$ communicates $z_{i-1,k+1}$ to sensor i in slot $k+1$. Sensor i then uses[1] $r_i(k+1)$ to updates the iterate $z_{i-1,k+1}$ to generate $z_{i,k+1}$. This is then passed to the next sensor in the cycle. Observe that in updating $z_{i-1,k+1}$, sensor i requires only $\chi^{(1)}_{i,k+1}, \ldots \chi^{(d)}_{i,k+1}$ and $\psi_{i,k+1}$, which were calculated by sensor i in the previous time slot. Thus, the algorithm is recursive and distributed. Furthermore, note that sensor i only needs to know its own system matrices $H_i, F_i(x)$ and $G_i(x)$.

As mentioned in Section I the convergence of the IRPE algorithm *cannot* be studied using the techniques that exist in literature. To establish convergence we will consider a hypothetical centralized system and prove that the iterates generated by the IRPE are identical to the iterates generated by the RPE algorithm when used on the hypothetical centralized system. We only state the final result here and discuss the proof in Appendix A.

Theorem 2: Let (A.1)—(A.4) hold. Moreover, let the step-size α_k be such that $k\alpha_k$ converges. Then, the iterates x_k generated by the IRPE algorithm in (10)–(15) converge to a local minimum of $f(x)$ in (8) over the set X, with probability 1.

We have not included an explicit input in modeling the system. The results immediately follow when there is a deterministic open-loop input $\{u_i(k)\}$ that drives the system in (1). Of course,

[1] We are assuming that sensor i obtains its measurement before it receives the iterate. From an implementation perspective, each time slot can be divided into two parts. In the first part, the sensors make measurements and in the second part they process.

$\{u_i(k)\}$ should be known to sensor i. Another immediate extension is to the case when the matrix H_i and noise $V_i(k)$ are also parametrized by x.

V. Discussion

The IRPE algorithm ignores any information about the parameter available in the joint statistics of the random process $\{\Theta_i(k;x)\}$ and $\{\Theta_j(k;x)\}$. A centralized system, on the other hand, can use the joint density information to generate better estimates. Thus, there is a tradeoff between the quality of the estimates and the 'distributedness' of the estimation scheme. For numerical simulations that capture this tradeoff and an application of the IRPE algorithm to localizing a diffusing source, we refer the reader to [10].

To truly understand the performance of the algorithm in practical settings, we need to obtain convergence results when there are communication errors. Further, we have considered a simple class of networks where the topology is fixed. It is important to obtain an algorithm that is similar to the IRPE for networks with a random and time-varying topologies.

References

[1] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Select Areas in Communications*, vol. 23, no. 4, pp. 798–808, 2005.

[2] C. G. Lopes and A. H. Sayeed, "Incremental adaptive strategies over distributed networks," *IEEE Transactions on Signal Processing*, vol. 55, no. 8, pp. 4064–4077, 2007.

[3] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with constant stepsize," *SIAM Journal of Optimization*, vol. 18, no. 1, pp. 29–51, 2007.

[4] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*, The MIT press, 1983.

[5] S. Sundhar Ram, A. Nedić, and V. V. Veeravalli, "Incremental stochastic subgradient algorithms for convex optimization," http://arxiv.org/abs/0806.1092, 2008.

[6] S. Roweis and Z. Ghahramani, "A unifying review of linear Gaussian models," *Neural Computation*, vol. 11, no. 2, pp. 305–345, 1999.

[7] S. Chen, C. Cowan, S. Billings, and P. Grant, "Parallel recursive prediction error algorithm for training layered neural networks," *International Journal of Control*, vol. 51, no. 6, pp. 1215–1228, 1990.

[8] A. Nedić and D. P. Bertsekas, "Incremental subgradient method for nondifferentiable optimization," *SIAM Journal of Optimization*, vol. 12, no. 1, pp. 109–138, 2001.

[9] P. Kumar and P. Varaiya, *Stochastic systems: Estimation, Identification and Adaptive Control*, Prentice-Hall, 1986.

[10] S. Sundhar Ram, V. V. Veeravalli, and A. Nedić, "Distributed and recursive parameter estimation in parametrized linear state-space models," Tech. report available at http://arxiv.org/abs/0804.1607, 2008.

Appendix

A. Proof of Theorem 2

In what follows, we extensively use the notion of block vectors and matrices. For positive integers a and b, let $\mathcal{M}_{a\times b}$ be the vector space of all real matrices of dimensions $a \times b$. A block vector in $\mathcal{M}_{a\times b}$ is a vector whose elements are from $\mathcal{M}_{a\times b}$. The length of a block vector is the number of block elements. In a similar manner, block matrices in $\mathcal{M}_{a\times b}$ are matrices where each element is itself a matrix from $\mathcal{M}_{a\times b}$. While writing block matrices we will allow for a slight abuse of notation and use 0 and I to denote the zero and identity matrices, respectively. Their dimensions can be unambiguously fixed from the dimensions of the other blocks in the block matrix. We will use $\boldsymbol{U}_b^a$, $b \le m$, to denote the unit block vector in $\mathcal{M}_{a\times a}$ of length m, with the b-th block equal to the identity matrix in $\mathcal{M}_{a\times a}$ and all the other blocks equal to the zero matrix in $\mathcal{M}_{a\times a}$. We allow i, j to take values in the set $\mathcal{I} = \{1, \dots, m\}$. We define $\delta[\cdot]$ as the Kronecker delta. Recall that the dimension of the matrices $\Theta_i(k; x)$ is q, the dimension of the measurement $r_i(k)$ is p, and the dimension of the parameter vector x is d.

1) Hypothetical centralized system: Without loss of generality, assume that each time slot has duration of m time units. Consider a hypothetical centralized scheme where at time $mk+j$, sensor j communicates $r_j(k+1)$ to the fusion center over a perfect delayless link. For $i \neq j$, sensor i communicates a predetermined constant value, say 0, that does not convey any information about the value taken by the parameter x.

Denote the sequence communicated by a sensor i by $\{\bar{r}_i(mk+j)\}$, with

$$\bar{r}_i(mk+j) = r_i(k+1)\delta[i-j]. \tag{16}$$

Next, denote the observation sequence at the fusion center by $\{\tilde{r}(mk+j)\}$, where

$$\tilde{r}(mk+j) = [\bar{r}_1(mk+j) \ \dots \ \bar{r}_m(mk+j)]^T = \boldsymbol{U}_j^p r_j(k+1).$$

The model for $\{\tilde{r}(mk+j)\}$, which we denote by $\{\tilde{R}(mk+j; x)\}$, can be defined starting from $\{R_i(k; x)\}$ in an identical manner. We now consider the problem of estimating x from observation sequence $\{\tilde{r}(mk+j)\}$ using the RPE algorithm. To use the RPE algorithm, the random process $\{\tilde{R}(mk+j; x)\}$ has to be represented as the output vector of a suitably defined state-space system. We do this next using the model for $r_i(k)$ in (1).

2) State space model for $\{\tilde{R}(mk+j;x)\}$: First observe that to use the RPE algorithm the state and observation matrices must be fixed and not change with time. Note that from (16), we have

$$\bar{R}_i(mk+j;x) = R_i(k+1;x)\ \delta[i-j]. \tag{17}$$

Let $\bar{D}_i(x)$ be the following $m \times m$ block matrix in $\mathcal{M}_{q\times q}$:

$$\bar{D}_i(x) = \begin{bmatrix} 0 & I & 0 & \cdot & \cdot & 0 \\ 0 & 0 & I & \cdot & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot & I \\ D_i(x) & 0 & 0 & \cdot & \cdot & 0 \end{bmatrix}. \tag{18}$$

Also, define $\bar{H}_i = H_i\ (\boldsymbol{U}_1^q)^T$, and note that

$$\bar{H}_i \boldsymbol{U}_j^q = H_i\ \ (\boldsymbol{U}_1^q)^T\, \boldsymbol{U}_j^q = H_i \delta[j-1].$$

Define $\bar{\Theta}_i(0;x) = \boldsymbol{U}_i^q\ \Theta_i(0;x)$, and

$$\begin{aligned} \bar{\Theta}_i(mk+j;x) &= \begin{cases} \boldsymbol{U}_{i-j+1}^q\ \Theta_i(k+1;x) & \text{if } j \leq i \\ \boldsymbol{U}_{m+1-(j-i)}^q\ \Theta_i(k+2;x) & \text{if } j > i, \end{cases} \\ \bar{W}_i(mk+j;x) &= \boldsymbol{U}_m^q\ W_i(k+1;x)\ \delta[i-j], \\ \bar{V}_i(mk+j) &= V_i(k+1)\ \delta[i-j]. \end{aligned}$$

We next state the following result that describes the evolution of $\{\bar{R}_i(n+1;x)\}$. The result can be verified by directly substituting from the definitions defined above. For details, we refer the reader to [10].

Proposition 1: For all $n \geq 0$, we have

$$\bar{\Theta}_i(n+1;x) = \bar{D}_i(x)\bar{\Theta}_i(n;x) + \bar{W}_i(n;x), \tag{19}$$

$$\bar{R}_i(n+1;x) = \bar{H}_i\bar{\Theta}_i(n+1;x) + \bar{V}_i(n+1). \tag{20}$$

Now, by combining the equations in (19)–(20) for $i \in \mathcal{I}$, we provide evolution equations for $\{\tilde{R}(n;x)\}$. Define

$$\begin{aligned}
&\tilde{F}(x) = \text{diag}\left(\bar{F}_1(x), \ldots, \bar{F}_m(x)\right), \quad \tilde{H}(x) = \text{diag}\left(\bar{H}_1(x), \ldots, \bar{H}_m(x)\right), \\
&\tilde{\Theta}(n;x) = \left[\bar{\Theta}_1(n;x) \quad \ldots \quad \bar{\Theta}_m(n;x)\right]^T, \quad \tilde{W}(n;x) = \left[\bar{W}_1(n;x) \quad \ldots \quad \bar{W}_m(n;x)\right]^T, \\
&\tilde{V}(n;x) = \left[\bar{V}_1(n;x) \quad \ldots \quad \bar{V}_m(n;x)\right]. \qquad (21)
\end{aligned}$$

Using the relations in (19) and (20), we can write

$$\tilde{\Theta}(n+1;x) = \tilde{D}(x)\tilde{\Theta}(n;x) + \tilde{W}(n;x), \qquad (22)$$

$$\tilde{R}(n+1;x) = \tilde{H}\tilde{\Theta}_i(n+1;x) + \tilde{V}(n+1). \qquad (23)$$

Equations (22) and (23) describe the state-space model for the fusion centers observation sequence $\{\tilde{r}(n)$. To use the RPE algorithm we need to evaluate the Kalman predictor for the system in (23).

3) Time-Invariant Kalman Predictor for Centralized System: Let us first obtain the time-invariant Kalman predictor for $\bar{R}_i(n;x)$ in (20). Fix $n = mk + j$ and define

$$\bar{\phi}_{i,n}(x;\bar{r}_i^{n-1}) = \begin{cases} \boldsymbol{U}^q_{i-j+1} \ \phi_{i,k+1}(x;r_i^k) & \text{if } j \le i \\ \boldsymbol{U}^q_{m+1-(j-i)} \ \phi_{i,k+2}(x;r_i^{k+1}) & \text{if } j > i, \end{cases}$$

$$\bar{g}_{i,n}(x;\bar{r}_i^{n-1}) = g_{i,k+1}(x;r_i^k) \ \delta[j-i].$$

Define $\bar{G}_i(x) = \boldsymbol{U}^p_m \ G_i(x)$, and $\bar{F}_i(x) = \bar{D}_i(x) - \bar{G}_i(x)\bar{H}_i$. The matrix $\bar{F}_i(x)$ will have the same form as $\bar{D}_i(x)$ in (18) but with $D_i(x)$ replaced by $F_i(x)$. Similar to Proposition 1, we can show

$$\begin{aligned}
\bar{\phi}_{i,n+1}(x;\bar{r}_i^n) &= \bar{F}_i(x)\bar{\phi}_{i,n}(x;\bar{r}_i^{n-1}) + \bar{G}_i(x)\bar{r}_i(n), \\
\bar{g}_{i,n+1}(x;\bar{r}_i^n) &= \bar{H}_i(x)\bar{\phi}_{i,n}(x;\bar{r}_i^n). \qquad (24)
\end{aligned}$$

We can now obtain a predictor family for $\tilde{\Theta}(n;x)$ and $\{\tilde{R}_n(x)\}$. Define,

$$\begin{aligned}
\tilde{\phi}_n(x;\tilde{r}^{n-1}) &= \left[\bar{\phi}_{1,n}(x;\bar{r}_1^{n-1}) \ldots \bar{\phi}_{m,n}(x;\bar{r}_m^{n-1})\right]^T, \\
\tilde{g}_n(x;\tilde{r}^{n-1}) &= \left[\bar{g}_{1,n}(x;\bar{r}_1^{n-1}) \ldots \bar{g}_{m,n}(x;\bar{r}_m^{n-1})\right]^T, \\
\tilde{G}(x) = \text{diag}\left(\bar{G}_1(x), \ldots, \bar{G}_m(x)\right) &\qquad \tilde{F}(x) = \text{diag}\left(\bar{F}_1(x), \ldots, \bar{F}_m(x)\right).
\end{aligned}$$

Furthermore, from (24) one can verify that

$$\begin{aligned}\tilde{\phi}_{n+1}(x;\tilde{r}^n) &= \tilde{F}(x)\tilde{\phi}_n(x;\tilde{r}^{n-1}) + \tilde{G}(x)\tilde{r}(n),\\ \tilde{g}_{n+1}(x;\tilde{r}^n) &= \tilde{H}\phi_{n+1}(x;\tilde{r}^n),\end{aligned} \tag{25}$$

Equation (25) is the Kalman predictor for the system in (23). For all components of $\tilde{r}(n+1)$ that are 0 the corresponding values in the predictor $\tilde{g}_{n+1}(x;\tilde{r}^n)$ will also be 0. For the one component of $\tilde{r}(n+1)$ that will equal $r_i(k+1)$, for some $i \in V$, the corresponding component will be $g_{i,k+1}(x;r_i^k)$.

4) RPE criterion for the centralized system in (23): We next evaluate the RPE criterion for the centralized system.

$$\begin{aligned}\tilde{f}(x) &= \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\mathsf{E}\left[\left\|\tilde{R}(n;x^*) - \tilde{g}_n(x,\tilde{R}^n(x^*))\right\|^2\right]\\ &= \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{m}\mathsf{E}\left[\left\|\bar{R}_i(n;x^*) - \bar{g}_{i,n}(x,\bar{R}_i^n(x^*))\right\|^2\right]\\ &= \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{m}\mathsf{E}\left[\left\|R_i(n;x^*) - g_{i,n}(x;R_i^n(x^*))\right\|^2\right] = f(x).\end{aligned}$$

We make the following important remark. As a consequence of the assumptions on the sequences $r_i(k)$ and the models in (1) the sequence $\{\tilde{r}(n)\}$ and its model in (23) satisfy the conditions of Theorem 1. *Thus, the sequence of iterates generated when RPE is applied to the system in (23) will converge to a local minimum of* $f(x)$. We will next show that the sequence generated by the RPE algorithm when applied to the system in (23) is identical to the sequence $\{z_{i,k}\}$ generated by the IRPE algorithm in (15).

5) RPE Algorithm for Centralized System: Here, we use the RPE algorithm to estimate x from $\{\tilde{r}(n)\}$. Define for $\ell = 1,\ldots,d$, $\nabla^{(\ell)}\tilde{F}(x) = \frac{\partial\tilde{F}(x)}{\partial x^{(\ell)}}$ and $\nabla^{(\ell)}G(x) = \frac{\partial\tilde{G}(x)}{\partial x^{(\ell)}}$.

The RPE algorithm applied to the system in (23) generates the iterates $\{\tilde{x}_n\}$ as follows

$$\begin{bmatrix}\tilde{h}_{n+1}\\ \tilde{\xi}_{n+1}^{(\ell)}\end{bmatrix} = \begin{bmatrix}\tilde{H} & 0\\ 0 & \tilde{H}\end{bmatrix}\begin{bmatrix}\tilde{\psi}_{n+1}\\ \tilde{\chi}_{n+1}^{(\ell)}\end{bmatrix},$$

$$\begin{aligned}
\tilde{\epsilon}_{n+1} &= \tilde{r}(n+1) - \tilde{h}_{n+1}, \\
\underline{\tilde{x}}^{(\ell)}_{n+1} &= \tilde{x}^{(\ell)}_{n} - \tilde{\alpha}_{n+1} \left(\tilde{\xi}^{(\ell)}_{n+1}\right)^T \tilde{\epsilon}_{n+1}, \\
\underline{\tilde{x}}_{n+1} &= \left[\underline{\tilde{x}}^{(1)}_{n+1} \ldots \underline{\tilde{x}}^{(d)}_{n+1}\right]^T, \\
\tilde{x}_{n+1} &= \mathcal{P}_X\left[\underline{\tilde{x}}_{n+1}\right], \\
\begin{bmatrix} \tilde{\psi}_{n+2} \\ \tilde{\chi}^{(\ell)}_{n+2} \end{bmatrix} &= \begin{bmatrix} \tilde{F}(\tilde{x}_{n+1}) & 0 \\ \nabla^{(\ell)}\tilde{F}(\tilde{x}_{n+1}) & \tilde{F}(\tilde{x}_{n+1}) \end{bmatrix} \begin{bmatrix} \tilde{\psi}_{n+1} \\ \tilde{\chi}^{(\ell)}_{n+1} \end{bmatrix} + \begin{bmatrix} \tilde{G}(\tilde{x}_{n+1}) \\ \nabla^{(\ell)}\tilde{G}(\tilde{x}_{n+1}) \end{bmatrix} \tilde{r}(n+1).
\end{aligned}$$

Here, $\alpha(n) = \alpha_{k+1}$ for $n = mk + j$ for $j = 1, \ldots, m-1$. Next, we assign the initial values for the recursion. Recall that the IRPE algorithm in (15) is initialized with the values $\psi_{i,1} = \psi_{i,s}$, $\xi^{(\ell)}_{i,1} = \xi^{(\ell)}_{i,s}$ for all i and ℓ, and $x_0 = x_s$. We let $\tilde{x}_0 = x_s$, and

$$\tilde{\psi}_0 = \left[\bar{\psi}_{1,s} \quad \ldots \quad \bar{\psi}_{m,s}\right]^T, \qquad \tilde{\xi}^{(\ell)}_0 = \left[\bar{\xi}^{(\ell)}_{1,s} \quad \ldots \quad \bar{\xi}^{(\ell)}_{m,s}\right]^T,$$

where $\bar{\psi}_{i,s} = \boldsymbol{U}^q_i \psi_{i,s}$ and $\bar{\xi}^{(\ell)}_{i,s} = \boldsymbol{U}^q_i \xi^{(\ell)}_{i,s}$ for all i and l.

6) Proof that $\tilde{x}_{mk+j} = z_{j,k+1}$: Fix $n = mk + j$. Recall that $\psi_{i,k}$ and $\chi^{(\ell)}_{i,k}$ are generated in the IRPE algorithm (10)–(15). Define for $l = 1, \ldots, d$,

$$\bar{\psi}_{i,n} = \begin{cases} \boldsymbol{U}^q_{i-j+1}\ \psi_{i,k+1} & \text{if } j \leq i \\ \boldsymbol{U}^q_{m+1-(j-i)}\ \psi_{i,k+2} & \text{if } j > i, \end{cases}$$

$$\bar{\chi}^{(\ell)}_{i,n} = \begin{cases} \boldsymbol{U}^q_{i-j+1}\ \chi^{(\ell)}_{i,k+1} & \text{if } j \leq i \\ \boldsymbol{U}^q_{m+1-(j-i)}\ \chi^{(\ell)}_{i,k+2} & \text{if } j > i. \end{cases}$$

We next state a key lemma. We refer the reader to [10] for a detailed proof.

Lemma 1: Let $n = mk + j$. If $\tilde{x}_n = z_{j,k+1}$ and

$$\tilde{\psi}_{n+1} = \left[\bar{\psi}_{1,n+1} \ldots \bar{\psi}_{m,n+1}\right]^T, \qquad \tilde{\chi}^{(\ell)}_{n+1} \quad = \left[\bar{\chi}^{(\ell)}_{1,n+1} \ldots \bar{\chi}^{(\ell)}_{m,n+1}\right]^T \qquad \textit{for } \ell = 1, \ldots, d,$$

then $\tilde{x}_n = z_{j+1,k+1}$, and

$$\tilde{\psi}_{n+2} = \left[\bar{\psi}_{1,n+2} \ldots \bar{\psi}_{m,n+2}\right]^T, \qquad \tilde{\chi}^{(\ell)}_{n+2} \quad = \left[\bar{\chi}^{(\ell)}_{1,n+2} \ldots \bar{\chi}^{(\ell)}_{m,n+2}\right]^T \qquad \textit{for } \ell = 1, \ldots, d.$$

We now prove Theorem 2 using the preceding lemma. Observe that the initial values for the RPE algorithm in (26) and the IRPE algorithm in (15) are chosen such that $\tilde{x}_0 = z_{0,1}$, $\tilde{\psi}_1 = \left[\bar{\psi}_{1,1} \ldots \bar{\psi}_{m,1}\right]^T$, and $\tilde{\chi}^{(\ell)}_1 = \left[\bar{\chi}^{(\ell)}_{1,1} \ldots \bar{\chi}^{(\ell)}_{m,1}\right]^T$ for all ℓ. By using Lemma 1 and the induction on k, we can conclude that $\tilde{x}_{mk+i} = z_{i,k+1}$ for all $k \geq 1$ and $i \in \mathcal{I}$.