Non-negative Distributed Regression for Data Inference in Wireless Sensor Networks

Jie Chen[*,†], Cédric Richard[†], Paul Honeine[*], José Carlos M. Bermudez[‡]

[*] Institut Charles Delaunay (FRE CNRS 2848), Université de Technologie de Troyes,10010 Troyes, France
[†]Laboratoire Fizeau (UMR 6525 CNRS), Université de Nice Sophia-Antipolis, 06108 Nice, France
[‡] Department of Electrical Engineering, Federal University of Santa Catarina, 88040-900, Florianópolis, SC - Brazil

***Abstract*—Wireless sensor networks are designed to perform on inference the environment that they are sensing. Due to the inherent physical characteristics of systems under investigation, non-negativity is a desired constraint that must be imposed on the system parameters in some real-life phenomena sensing tasks. In this paper, we propose a kernel-based machine learning strategy to deal with regression problems. Multiplicative update rules are derived in this context to ensure the non-negativity constraints to be satisfied. Considering the tight energy and bandwidth resource, a distributed algorithm which requires only communication between neighbors is presented. Synthetic data managed by heat diffusion equations are used to test the algorithms and illustrate their tracking capacity.**

I. Introduction

Wireless sensor networks (WSNs) rely on sensor devices deployed in an environment to provide an inexpensive way to monitor physical phenomena, such as temperature, humidity, acoustic, etc. In traditional centralized solutions, the nodes in the network collect observations and send them to a central basic station for processing. This mode requires a powerful computation center, in addition to extensive amount of communication between the nodes and the center. In distributed strategies, estimates performed by nodes rely only on local data and on interactions with immediate neighbors. The burden of processing and communications is significantly reduced. Distributed learning in wireless sensor networks has been addressed in a variety of research works.

In many real-life phenomena, including biological and physical ones, physical characteristics inherent to the system under investigation require the imposition of non-negativity constraints on the parameters to estimate. For instance, observations in studies of concentration fields or thermal radiation fields are awlays described with non-negative values (in ppm or in Kelvin). Non-negativity as a physical constraint has received growing attention from the signal processing community during the last decade.

Non-parametric approach based on reproducing kernel methods have recently been successfully applied to distributed regression with collaborative networks. In [1], the authors present a general framework for distributed linear regression motivated by WSNs. In [2], a learning algorithm based on successive orthogonal projections is derived to solve the regularized kernel least-squares problem for regression in sensor networks. The work in [3] generalizes the model and algorithm discussed in [2]. [4] makes a detailed summarization of the distributed inference in the class of work presented in previous two literatures. In [5], the authors present a projection based kernel distributed learning strategy with reduced order models by using a sparsification criterion. Some distributed estimation algorithms have been also proposed in the context of distributed adaptive filtering, including incremental LMS [6], diffusion LMS [7] and diffusion RLS [8]. These works provide comprehensive studies in the functional regression and estimation for distributed learning in WSNs. Nevertheless, none of these algorithms could be used directly to solve the estimation problems in sensor networks under non-negativity constraints.

In this paper, we concentrate on the problem of modeling physical phenomena under non-negativity constraints, and of tracking its evolution. Firstly we formulate the non-negative regression with kernels in a centralized context. A simple multiplicative algorithm is derived to solve this problem. Then we show how the optimization problem can be relaxed to a problem of distributed regression in which nodes only need to communicate with neighbors.

II. Non-negative regression for inference

Within the context of learning in a wireless sensor network of N sensors, we often model a physical phenomenon as a function of the location. Consider a relationship $\psi(.)$ between the sensor's measurement and its position $\boldsymbol{x}_n$. We seek to estimate the function $\psi(\cdot)$ based on newly available position-measurement data y_n to minimize the summed square error

$$\min_{\psi\in\mathcal{H}} \sum_{n=1}^{N} E\left(\psi(\boldsymbol{x}_n) - y_n\right)^2. \tag{1}$$

By virtue of the representer theorem, the function $\psi(\cdot)$ of reproducing Hilbert kernel space $\mathcal{H}$ can be written with a kernel expansion $\psi(\cdot) = \sum_{n=j}^{N} \alpha_j \kappa(\cdot, \boldsymbol{x}_n)$. Doing that, the cost function can be written as

$$\begin{aligned} J(\boldsymbol{\alpha}) &= \sum_{n=1}^{N} E(\sum_{j=1}^{N} \alpha_j \kappa(\boldsymbol{x}_n, \boldsymbol{x}_j) - y_n)^2 \\ &= \sum_{n=1}^{N} E\left(\boldsymbol{\alpha}^\top \boldsymbol{\kappa}_{\boldsymbol{x}_n} - y_n\right)^2. \end{aligned} \tag{2}$$

After determining the weight vector $\boldsymbol{\alpha}$, the field can be inferred at any points $\boldsymbol{x}$. One of the most widely used kernels is the Gaussian kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathrm{e}^{-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2}$. When a non-negative field is to be estimated, and considering that the Gaussian kernel is always positive, each component of the coefficient vector $\boldsymbol{\alpha}$ should be constrained to be non-negative to ensure a non-negative inference function $\psi(\boldsymbol{x})$ at any given position $\boldsymbol{x}$. The constrained optimization problem can be formalized as

$$\boldsymbol{\alpha}^o = \arg\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) \tag{3}$$

$$\text{subject to } \boldsymbol{\alpha} \geq \mathbf{0} \tag{4}$$

The gradient of $J(\boldsymbol{\alpha})$ is easily computed as follows

$$\nabla J(\boldsymbol{\alpha}) = \sum_{n=1}^{N} E\left(\boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^{\top}\boldsymbol{\alpha} - y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n}\right) \tag{5}$$

As the evaluation of the gradient usually cannot be achieved in many real-life applications, we use the instantaneous estimator

$$\widetilde{\nabla} J(\boldsymbol{\alpha}) = \sum_{n=1}^{N} \left(\boldsymbol{\kappa}_{\boldsymbol{x}_n}^{\top}\boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\alpha} - y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n}\right) \tag{6}$$

Note that $\psi(.)$ is linear with respect to the kernel functions $\kappa(., \boldsymbol{x}_n)$, although it is nonlinear with respect to $\boldsymbol{x}_n$.

A. Gradient projection algorithm

Gradient projection is a popular family of methods to solve this kind of optimization problem. They are based on successive projects on the feasible region, which are non-expensive operations when the constraints are simple. We move from $\boldsymbol{\alpha}(k)$ to iterate $\boldsymbol{\alpha}(k+1)$ as follows, first, we choose some scalar parameter $\eta(k) > 0$ and set

$$\boldsymbol{\beta}(k) = (\boldsymbol{\alpha}(k) - \eta(k)\nabla J(\boldsymbol{\alpha}(k)))_{+} \tag{7}$$

We then choose a second scalar $\mu(k) \in [0, 1]$ and set

$$\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k) + \mu(k)(\boldsymbol{\beta}(k) - \boldsymbol{\alpha}(k)) \tag{8}$$

Their low memory requirements and simplicity make them attractive for large scale problems. On the other hand, it is well known that these methods may exhibit very slow convergence if not combined with appropriate step length selection.

B. Multiplicative weight update algorithm

In this paper, we are more interested in another class of algorithm in multiplicative form. Let us now decompose the gradient $-\widetilde{\nabla} J(\boldsymbol{\alpha})$ as following

$$[-\nabla \widetilde{J}(\boldsymbol{\alpha}(k))]_i = [\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i - [\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i \tag{9}$$

where $[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i$ and $[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i$ are strictly positive components. Obviously, such a decomposition is not unique but always exists. Consider the update rule of gradient method

$$\boldsymbol{\alpha}_i(k+1) = \boldsymbol{\alpha}_i(k) + \eta_i(k)[-\nabla \widetilde{J}(\boldsymbol{\alpha}(k))]_i \tag{10}$$

If the step size is taken as

$$\eta_i(k) = \frac{\alpha_i(k)}{[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i} \tag{11}$$

The update equation for the i-th component can then be expressed as

$$\alpha_i(k+1) = \alpha_i(k)\frac{[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i}{[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i} \tag{12}$$

of which the vector form is

$$\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k)\mathrm{diag}\left(\frac{[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i}{[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i}\right) \tag{13}$$

This expression is referred to as the multiplicative weight update algorithm. If we initialize the weight vector with a positive vector, the constraints will be always satisfied due to the non-negativity of $[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i$ and $[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i$. The gradient defined by (6) can be decomposed as in (9) by setting

$$\boldsymbol{U}(\boldsymbol{\alpha}(k)) = \sum_{n=1}^{N} y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n} + \xi \tag{14}$$

$$\boldsymbol{V}(\boldsymbol{\alpha}(k)) = \sum_{n=1}^{N} \boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^{\top}\boldsymbol{\alpha}(k) + \xi \tag{15}$$

with ξ positive to avoid $[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i$ to become negative due to the disturbance of some kind of additive observation noise. At each instant k, with the measure $y_{n,k}$, a centralized algorithm vector weight update is then

$$\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k)\mathrm{diag}\left(\frac{[\sum_{n=1}^{N} y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n} + \xi]_i}{[\sum_{n=1}^{N} \boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^{\top}\boldsymbol{\alpha}(k) + \xi]_i}\right) \tag{16}$$

III. Distributed regression with diffusion strategy in WSNs

Nevertheless the centralized algorithm defined by (16) is not suitable for distributed learning in the sensor networks as the order of models scales linearly with the number of deployed sensors. Moreover, each sensor should pass their measures to the center. In what follows, we show how the optimization problem in (2) can be relaxed for the problem of distributed inference.

A. Localized cost function

Let $\mathcal{N}_k \subseteq \{1, 2, ..., N\}$ denote the set of neighbors for sensor k. And we assume that each link can support the simple messages to be passed by our algorithm. Consider an $N \times N$ matrix $\mathbf{B}$ with with non-negative entries $\{b_{n,k}\}$ such that

$$b_{n,k} = 0 \quad \text{if } n \notin \mathcal{N}_k \qquad \mathbf{B}\mathbb{1} = \mathbb{1} \qquad \mathbb{1}^{\top}\mathbf{B} = \mathbb{1}^{\top} \tag{17}$$

where $\mathbb{1}$ denotes the $N \times 1$ vector with unit entries. With the constraint of communication range, the cost function of (2) is rewritten as follows

$$J(\boldsymbol{\alpha}) = J_k(\boldsymbol{\alpha}) + \sum_{n=1, n\neq k}^{N} J_n(\boldsymbol{\alpha}) \tag{18}$$

We define diagonal matrices $\mathbf{C_k}$ for each node k with elements $\boldsymbol{C}_k$: $c_{k,i,i} = 1$ if $i \in \mathcal{N}_k$ and $c_{k,i,i} = 0$ otherwise. The local cost function is defined as

$$J_k(\boldsymbol{\alpha}) = \sum_{n=1}^{N} b_{n,k} E\left(\boldsymbol{\alpha}^{\top}\mathbf{C}_n\boldsymbol{\kappa}_{\boldsymbol{x}_n} - y_n\right)^2 \tag{19}$$

which is actually equivalent to

$$J_k(\boldsymbol{\alpha}_k) = \sum_{n\in\mathcal{N}_k} b_{n,k} E\left(\boldsymbol{\alpha}_k^\top \boldsymbol{\kappa}_{\boldsymbol{x}_n} - y_n\right)^2 \tag{20}$$

Each node could only communicate with the nodes in the range of neighborhood, from the view of node k, the instantaneous gradient of the cost function (18) For each node k

$$[\nabla J(\boldsymbol{\alpha}_k)]_i = \left[\sum_{n\in\mathcal{N}_k} b_{n,k}\left(\boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^\top \boldsymbol{\alpha}_k - y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n}\right)\right]_i + \left[\sum_{n=1,n\neq k}^{N} \mathbf{C}_n \nabla J_n(\boldsymbol{\alpha})\right]_i \tag{21}$$

where $i \in \mathcal{N}_k$. For node k, to obtain the information of the second part of (21) two-hop transmission is needed, which introduces inconvenience to the learning process in networks. To relax the problem so that the sensors only need to get information from its neighbors, we use

$$[\nabla J(\boldsymbol{\alpha}_k)]_i = \left[\sum_{n\in\mathcal{N}_k} b_{n,k}\left(\boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^\top \boldsymbol{\alpha}_k - y_n\boldsymbol{\kappa}_{\boldsymbol{x}_n}\right)\right]_i + \left[\sum_{n\in\mathcal{N}_k,n\neq k}^{N} \mathbf{C}_n \nabla J_n(\boldsymbol{\alpha})\right]_i \tag{22}$$

The first part of (22) can be viewed as the gradient of local cost function $\nabla J_k(\boldsymbol{\alpha}_k)$. Using the proposed multiplicative algorithm developed in the section II-B, $[-\nabla J_k(\boldsymbol{\alpha}_k)]_i$ is decomposed into two positive components

$$[\boldsymbol{U_k}(\boldsymbol{\alpha}(k))]_i = \left[\sum_{n\in\mathcal{N}_k} b_{n,k} y_n \boldsymbol{\kappa}_{\boldsymbol{x}_n}\right]_i + \xi \tag{23}$$

$$[\boldsymbol{V_k}(\boldsymbol{\alpha}(k))]_i = \left[\sum_{n\in\mathcal{N}_k} b_{n,k} \boldsymbol{\kappa}_{\boldsymbol{x}_n}\boldsymbol{\kappa}_{\boldsymbol{x}_n}^\top \boldsymbol{\alpha}_k(k)\right]_i + \xi \tag{24}$$

The second part of (22) could be viewed as a regularization item for the local gradient decomposed using

$$\left[\widetilde{\boldsymbol{U}}_k(\boldsymbol{\alpha}(k))\right]_i = \sum_{n\in\mathcal{N}_k,n\neq k} [\boldsymbol{U}_n(\boldsymbol{\alpha}(k))]_i \tag{25}$$

$$\left[\widetilde{\boldsymbol{V}}_k(\boldsymbol{\alpha}(k))\right]_i = \sum_{n\in\mathcal{N}_k,n\neq k} [\boldsymbol{V}_n(\boldsymbol{\alpha}(k))]_i \tag{26}$$

where $[\boldsymbol{U}(\boldsymbol{\alpha}(k))]_i$ and $[\boldsymbol{V}(\boldsymbol{\alpha}(k))]_i$ are transferred from its neighbors. And they ensure the positivity of $\left[\widetilde{\boldsymbol{U}}_k(\boldsymbol{\alpha}(k))\right]_i$ and $\left[\widetilde{\boldsymbol{V}}_k(\boldsymbol{\alpha}(k))\right]_i$. Finally, the coefficients update rule for node k is written in multiplicative form as

$$\boldsymbol{\alpha}_i(k+1) = \boldsymbol{\alpha}_i(k)\frac{[\boldsymbol{U_k}(\boldsymbol{\alpha}(k))]_i + \left[\widetilde{\boldsymbol{U}}_k(\boldsymbol{\alpha}(k))\right]_i}{[\boldsymbol{V_k}(\boldsymbol{\alpha}(k))]_i + \left[\widetilde{\boldsymbol{V}}_k(\boldsymbol{\alpha}(k))\right]_i} \tag{27}$$

The algorithm is depicted pictorially in figure 1.

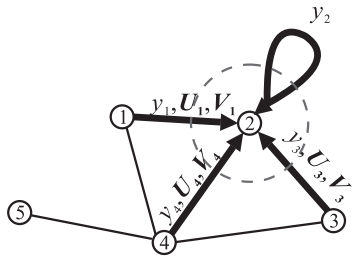

Fig. 1. The schema of the algorithm

B. Aggregation

When we wish to find the field at the position $\boldsymbol{x} \in \mathbb{R}^2$, it may employ one of the following strategies to aggregate the estimate of the network.

1) Single Sensor: The decision center will simply choose the sensor the most approached to the position $\boldsymbol{x}$, and use the estimate of field at $\boldsymbol{x}$ of this sensor.

2) m-Nearest-neighbor: The decision center will average the estimates provided by the m sensors nearest $\boldsymbol{x}$. The "single sensor" rule is a special case of this rule, corresponding to $m = 1$.

IV. Simulation experiments

To illustrate the relevance of the proposed technique, we consider a classical application of estimating a heat diffusion field governed by the partial differential equation

$$\frac{\partial T(\boldsymbol{x},t)}{\partial t} - c\nabla_{\boldsymbol{x}}^2 T(\boldsymbol{x},t) = Q(\boldsymbol{x},t). \tag{28}$$

Here $T(\boldsymbol{x},t)$ denotes the temperature as a function of space and time, c is a medium-specific parameter, $\nabla_{\boldsymbol{x}}^2$ is the Laplace spatial operator, and $Q(\boldsymbol{x},t)$ is the heat added. The temperature field generated here is a non-negative field. With the Gaussian kernel function $\phi_i(\boldsymbol{x})$ are always positive, to ensure that the estimate of any position is non-negative, it is required that non-negativity constraints are imposed on the coefficients $\boldsymbol{\alpha}$.

We studied the problem of monitoring the evolution of the heat in a 2-by-2 square region with open boundaries and conductivity $c = 0.1$, using $N = 100$ random positions with known measurement. Two heat sources of intensity 200 W were placed within the region, the first one was activated from $t = 1$to $t = 100$, and the second one from $t = 100$ to $t = 200$. The measurement is corrupted by a additive Gaussian noise with variance of 0.01. Preliminary experiments were conducted to tune the parameters, yielding the bandwidth of Gaussian function $\sigma = 0.1826$. The convergence of the proposed algorithm is illustrated in where we show the evolution overtime of the normalized mean-square prediction error, defined on all the measure positions by

$$\frac{\sum_{n=1}^{N}(d_n - \psi(\boldsymbol{x}_n))^2}{\sum_{n=1}^{N} d_n^2}.$$

The following experiments are conducted for the purpose of comparison:

1) Centralized multiplicative algorithm described in section II-B: Due to the unscalability centralized algorithms, this centralized method may not be appropriate if adopted directly into the wireless sensor networks; however, we implement it as a optimal solution for comparison.
2) Proposed distributed multiplicative algorithm: Our proposed distributed algorithm derived in section III is tested to show its ability of modeling such a field and its pursuing capacity of the environment change;
3) Centralized gradient projection algorithm in section II-A: One centralized gradient projection algorithm is tested here to compared with the multiplicative algorithm. The performance of algorithms in this class is highly dependent on the step length selection strategy. Considerable attention has been paid to an approach due Barzilai-Borwein gradient projection method.
4) Distributed gradient projection algorithm: In order compare with the distributed multiplicative algorithm, we adapt the centralized gradient projection algorithm of *3)* to minimize local cost functions within the neighborhood nodes.

Two random scenarios, respectively depicted in Figure 2 and Figure 3, are taken into simulations to show the convergence performance. In the 1st scenario, two sources are located at positions relatively "poor" of sensors; otherwise, in the 2nd scenario, the two sources are located at better positions.

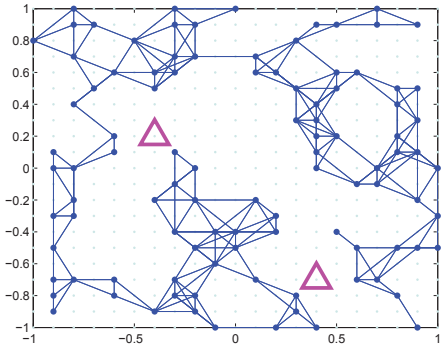

Fig. 2. 1st simulation scenario with 100 randomly distributed sensors. The edges between two nodes show the neighborhood relation. Two magenta △ represent the positions of two sources.

The abrupt change in heat sources at $t = 100$ is clearly visible, and hightlights the convergence behavior of these algorithms. In Figure 4, there's slight difference between multiplicative algorithm and gradient projection with Barzilai-Borwein step size selection, which shows efficiency and simplicity of the proposed algorithm. With all available information of the network the centralized methods perform better than distributed ones. In Figure 5, with better positions of sensors relative to even sources, the estimation error of the network

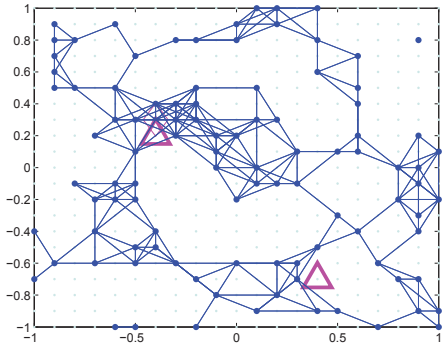

Fig. 3. 2nd simulation scenario with 100 randomly distributed sensors. The edges between two nodes show the neighborhood relation. Two magenta △ represent the positions of two sources. In this scenario, the positions of two sources are more close to sensors than that in scenario1.

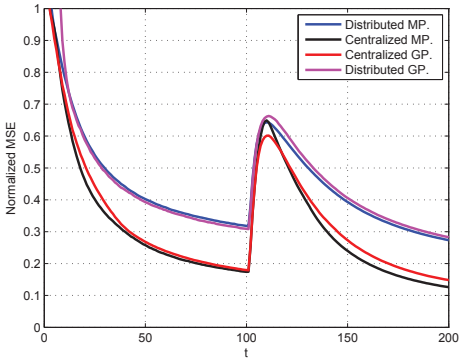

Fig. 4. Convergence comparison among four methods of the 1st scenario. In the legend MP represents for multiplicative method; GP represents for gradient projection.

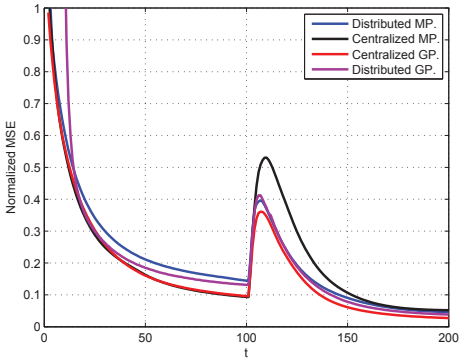

Fig. 5. Convergence comparison among four methods of the 2nd scenario. In the legend MP represents for multiplicative method; GP represents for gradient projection.

is apparently lower than that in the 1st scenario. However, the

centralized multiplicative shows a slower convergence rate as the abrupt change after $t = 100$ than the distributed algorithm. This might be caused by the longer filter length (N=100). Whereas the Barzilai-Borwein gradient projection performs better here at the cost of manipulating the Gramme matrix of large dimension.

V. Conclusion

In many real-life phenomena non-negative is a desired constraint that must be imposed on the parameters to estimate due to the inherent physical characteristics of systems.. In this paper, we proposed a multiplicative method for data inference under non-negativity constraints. Under the context of wireless sensor networks, we developed a distributed learning algorithm to enable each sensor to estimate the non-negative field with the help of neighbor information. The proposed algorithm also shows a good performance in its tracking capacity.

References

[1] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM New York, NY, USA, 2004, pp. 1–10.

[2] J. Predd, S. Kulkarni, and H. Poor, "Regression in sensor networks: Training distributively with alternating projections," in *Proc. SPIE*, vol. 5910. Citeseer, 2005, pp. 42–56.

[3] ——, "Distributed Kernel Regression: An Algorithm for Training Collaboratively," in *IEEE Information Theory Workshop, 2006. ITW'06 Punta del Este*, 2006, pp. 332–336.

[4] ——, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56–69, 2006.

[5] P. Honeine, C. Richard, J. Bermudez, H. Snoussi, M. Essoloh, and F. Vincent, "Functional estimation in Hilbert space for distributed learning in wireless sensor networks," in *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing-Volume 00*. IEEE Computer Society, 2009, pp. 2861–2864.

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

[7] ——, "Diffusion least-mean squares over adaptive networks: Formulation and performance analysis," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3122–3136, 2008.

[8] F. Cattivelli, C. Lopes, A. Sayed *et al.*, "Diffusion recursive least-squares for distributed estimation over adaptive networks," *IEEE Transactions on Signal Processing*, vol. 56, pp. 1865–1877, 2008.

[9] A. De Pierro, "On the convergence of the iterative image space reconstruction algorithm for volume ECT." *IEEE TRANS. MED. IMAG.*, vol. 6, no. 2, pp. 174–175, 1987.

[10] F. Benvenuto, R. Zanella, L. Zanni, and M. Bertero, "Nonnegative least-squares image deblurring: improved gradient projection approaches," *Inverse Problems*, vol. 26, p. 025004, 2010.