Distributed dictionary learning over a sensor network

Pierre Chainais, Cédric Richard

Distributed dictionary learning over a sensor network

P. Chainais*[1] and C. Richard†[2]

[1]LAGIS UMR CNRS 8219, Ecole Centrale Lille, INRIA-Lille Nord Europe, SequeL, France
[2]Laboratoire Lagrange UMR CNRS 7293, University of Nice Sophia-Antipolis, France

May 31, 2013

Abstract

We consider the problem of distributed dictionary learning, where a set of nodes is required to collectively learn a common dictionary from noisy measurements. This approach may be useful in several contexts including sensor networks. Diffusion cooperation schemes have been proposed to solve the distributed linear regression problem. In this work we focus on a diffusion-based adaptive dictionary learning strategy: each node records observations and cooperates with its neighbors by sharing its local dictionary. The resulting algorithm corresponds to a distributed block coordinate descent (alternate optimization). Beyond dictionary learning, this strategy could be adapted to many matrix factorization problems and generalized to various settings. This article presents our approach and illustrates its efficiency on some numerical examples.

Keywords: dictionary learning, sparse coding, distributed estimation, diffusion, matrix factorization, adaptive networks, block coordinate descent.

1 Introduction

In a variety of contexts, huge amounts of high dimensional data are recorded from multiple sensors. When sensor networks are considered, it is desirable that computations be distributed over the network rather than centralized in some fusion unit. Indeed, centralizing all measurements lacks robustness - a failure of the central node is fatal - and scalability due to the needed energy and communication resources. In distributed computing, every node communicates with its neighbors only and processing is carried out by every node in the network. Another important remark is that relevant information from the data usually lives in a space of much reduced dimension compared to the physical space. The extraction of this relevant information calls for the identification of some adapted sparse representation of the data. Sparsity is an important property which favors the identification of the main components that are characteristic of the data. Each observation is then described by a sparse subset of atoms taken from a redundant dictionary. We study the problem of dictionary learning distributed over a sensor network in a setting where a set of nodes is required to collectively learn an adaptive sparse representation of independent observations.

Learning an adaptive representation of the data is useful for many tasks such as storing, transmitting or analyzing the data to understand its content. A basic dictionary can be obtained by using a Principal Component Analysis (PCA) also known as Karhunen-Loève decomposition in signal processing. However the number of atoms of such a decomposition is limited to the dimension of the data space. A more adapted representation is obtained by using a redundant dictionary where for instance no orthogonality property is imposed. While the number of potential characteristic sources is large, the number of effective sources which contribute to the signal observed by a sensor at a single moment is much smaller. Many recent works have shown the interest of learning a *redundant dictionary* allowing for a *sparse representation* of the data, see [TF11] for an up-to-date review. Furthermore, the problem of dictionary learning belongs to the more general family of matrix factorization problems that appear in a host of applications.

In this paper, we consider the situation where a set of connected nodes independently record data from observations of the same kind of physical system: each observation is assumed to be described by a sparse rep-

*pierre.chainais@ec-lille.fr
†cedric.richard@unice.fr

resentation using a common dictionary for all sensors. For instance, a set of cameras observe the same kind of scenes or a set of microphones records the same kind of sound environment.

The dictionary learning and the matrix factorization problems are connected to the linear regression problem. Let us consider that a set of observations of the system is described by a data matrix $\mathbf{S}$ where each column corresponds to one observation. Assume that $\mathbf{S} = \mathbf{DX}$. If either the coefficients $\mathbf{X}$ (resp. the dictionary $\mathbf{D}$) are known, the estimation of the dictionary (resp. the coefficients) knowing $\mathbf{S}$ is a linear regression problem. It appears that several recent works have proposed efficient solutions to the problem of least mean square (LMS) distributed linear regression, see [CS10] and references therein. The main idea is to use a so-called *diffusion strategy*: each node n carries out its own estimation $\mathbf{D}_n$ of the same underlying linear regression vector $\mathbf{D}$ but can communicate with its neighbors as well. The information provided to some node by its neighbors is taken into account according to some weights interpreted as diffusion coefficients. Under some mild conditions, the performance of such an approach in terms of mean squared error is similar to that of a centralized approach [ZS12]. Denoting by $\mathbf{D}_c$ the centralized estimate which uses all the observations at once, it can be shown that the error $\mathbb{E}\|\mathbf{D}_n - \mathbf{D}\|_2$ of the distributed estimate is of the same order as $\mathbb{E}\|\mathbf{D}_c - \mathbf{D}\|_2$: diffusion networks match the performance of the centralized solution.

Our work gives strong indication that the classical dictionary learning technique based on block coordinate descent on the dictionary $\mathbf{D}$ and the coefficients $\mathbf{X}$ can be adapted to the distributed framework by adapting the diffusion strategy mentionned above. Our numerical experiments also strongly support this idea. The theoretical analysis is the subject of ongoing work. Note that solving this type of matrix factorization problems is really at stake since it corresponds to many inverse problems: denoising, adaptive compression, recommendation systems... A distributed approach is highly desirable both for use in sensor network and for parallelization of numerically expensive learning algorithms.

The paper is organized as follows. Section 2 formulates the problem we are considering. Section 3 recalls about dictionary learning techniques based on block coordinate descent approaches. Section 4 presents the diffusion strategy for distributed dictionary learning. Section 5 shows some numerical experiments and results. Section 6 points to main claims and prospects.

2 Problem formulation

Consider N nodes over some region. In the following, boldfaced letters denote column vectors, and capital letters denote matrices. The node n takes q_n measurements $\mathbf{y}_n(i)$, $1 \leq i \leq q_n$ from some physical system. All the observations are assumed to originate from independent realizations $\mathbf{s}_n(i)$ of the same underlying stochastic source process $\mathbf{s}$. Each measurement is a noisy measurement

$$\mathbf{y}_n(i) = \mathbf{s}_n(i) + \mathbf{z}_n(i) \tag{1}$$

where $\mathbf{z}$ denotes the usual i.i.d. Gaussian noise with covariance matrix $\Sigma_n = \sigma_n^2 \mathbb{I}$. Our purpose is to learn a redundant dictionary $\mathbf{D}$ which carries the characteristic properties of the data. This dictionary must yield a sparse representation of $\mathbf{s}$ so that:

$$\forall n, \quad \mathbf{y}_n(i) = \underbrace{\mathbf{D}\mathbf{x}_n(i)}_{\mathbf{s}_n(i)} + \mathbf{z}_n(i) \tag{2}$$

where $\mathbf{x}_n(i)$ features the coefficients $x_{nk}(i)$ associated to the contribution of atom $\mathbf{d}_k$, the k-th column in the dictionary matrix $\mathbf{D}$, to $\mathbf{s}_n(i)$. The sparsity of $\mathbf{x}_n(i)$ means that only few components of $\mathbf{x}_n(i)$ are non zero.

We are considering the situation where a unique dictionary $\mathbf{D}$ generates the observations at all nodes. On the contrary, observations will not be shared between nodes (this would be one potential generalization). Our purpose is to learn (estimate) this dictionary in a distributed manner thanks to in-network computing only, see section 4. As a consequence, each node will locally estimate a local dictionary $\mathbf{D}_n$ thanks to i) its observations $\mathbf{y}_n$ and ii) communication with its neighbors. The neighborhood of node n will be denoted by $\mathcal{N}_n$, including node n itself. The number of nodes connected to node n is the degree ν_n.

3 Dictionary learning strategies

3.1 Problem formulation

Various approaches to dictionary learning have been proposed [TF11]. Usually, in the centralized setting, the q observations are denoted by $\mathbf{y}(i) \in \mathbb{R}^p$ and grouped in a matrix $\mathbf{Y} = [\mathbf{y}(1), ..., \mathbf{y}(q)]$. As a consequence, $\mathbf{Y} \in \mathbb{R}^{p \times q}$. The dictionary (associated to some linear transform) is denoted by $\mathbf{D} \in \mathbb{R}^{p \times K}$: each column is one atom $\mathbf{d}_k$ of the dictionary. We gather the coefficients associated to observations in a single matrix $\mathbf{X} = [\mathbf{x}(1), ..., \mathbf{x}(q)]$. We will consider learning methods based on block coordinate descent or alternate

optimization on $\mathbf{D}$ and $\mathbf{X}$ with a sparsity constraint on $\mathbf{X}$ [Tse01, TF11].

The data is represented as the sum of a linear combination of atoms and a noise term $\mathbf{Z} \in \mathbb{R}^{p \times q}$:

$$\mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{Z} \tag{3}$$

Since dictionary learning is a *matrix factorization* problem, it is an ill-posed problem. The dictionary is potentially redundant and not necessarily orthogonal so that $K \gg p$. Some modeling is necessary to constrain the set of possible solutions. Various conditions can be considered (non-negative matrix factorization, orthogonal decomposition, ...). In general, a dictionary is considered as *adapted* to the data if each observation $\mathbf{y}(i)$ can be described by a little number of coefficients $\mathbf{x}(i)$. One usually searches for a *sparse representation* and imposes the sparsity of $\mathbf{X}$ [TF11].

3.2 Learning a redundant dictionary for sparse representation

The properties of redundancy of the dictionary and sparsity of the coefficients are complementary. The extreme case would be the one where the dictionary contains each one of the true data underlying the noisy observation so that only one non zero coefficient in $\mathbf{x}(i)$ would be sufficient to describe the observation $\mathbf{y}(i)$. Then we would have $K = q$ and $\mathbf{X}$ would be maximally sparse (only 1 non zero coefficient per observation). Of course this dictionary would not be very interesting since its generalization power would be very limited. A good dictionary must offer a compromise between its fidelity to the learning data set and its ability to generalize. The choice of the size of the dictionary is often made a priori so that $K > p$ to ensure some redundancy and $K < q$ to ensure it can capture some general information shared by the data. For instance, when working on image patches of size 8×8 (the data lives in dimension $p = 64$), it is typically proposed to learn dictionaries of size 256 or 512 [AEB06, TF11].

In the classical setting, the noise is usually assumed to be i.i.d. Gaussian noise so that the reconstruction error is measured by the L2-norm. Sparsity of the coefficient matrix is imposed through a L0 relaxed to L1-penalization in the mixed optimization problem:

$$(\mathbf{D}, \mathbf{X}) = \text{argmin}_{(\mathbf{D},\mathbf{X})} \quad \frac{1}{2}||\mathbf{Y} - \mathbf{D}\mathbf{X}||_2^2 + \lambda||\mathbf{X}||_1 \tag{4}$$

Under some mild conditions, this problem is known to provide a solution to L0-penalized problem (ideally we would prefer to directly solve the L0-penalized problem) [SMF10].

3.3 Block coordinate descent

One way to solve problem (4) is to use block coordinate descent [Tse01], that is alternate optimization on $\mathbf{X}$ and $\mathbf{D}$. There are several possibilities to do this, see e.g. [AEB06]. For instance, after some initialization, one may use gradient descents on $\mathbf{X}$ and $\mathbf{D}$ [OF96]. Such approaches are attractive since we know that linear regression by gradient descent can be translated in the distributed framework [CS10].

One possible choice is the Basis Pursuit algorithm. At each step, the forward-backward splitting (Basis Pursuit Denoising with Iterated Soft Thresholding, see [SMF10], p.161) iteratively estimates $\mathbf{X}$ by iterating the following steps over s and t:

1. $\mathbf{X}^{(s,t+1/2)} = \mathbf{X}^{(s,t)} + \lambda\mu\mathbf{D}^{(s,t)T}\left[\mathbf{Y} - \mathbf{D}^{(s)}\mathbf{X}^{(s,t)}\right]$ (gradient descent step, $\forall n$)
2. $\mathbf{X}^{(s,t+1)} = \text{SoftThreshold}_{\lambda\mu}(\mathbf{X}^{(s,t+1/2)})$ (soft thresholding step)

Then we update $\mathbf{X}^{(s+1)} = \mathbf{X}^{(s,T)}$ after T (typically 30 or 40) iterated soft thresholding. Note that one must have $\mu \in \left(0, \frac{2}{|||\mathbf{D}|||^2}\right)$ where $|||\cdot|||$ denotes the spectral norm. Then the dictionary $\mathbf{D}^{(s+1)}$ can be updated knowing $\mathbf{X}^{(s+1)}$, using a simple gradient descent:

$$\tilde{\mathbf{D}}^{(s+1)} = \mathbf{D}^{(s)} + \eta\left[\mathbf{Y} - \mathbf{D}^{(s)}\mathbf{X}^{(s+1)}\right]\mathbf{X}^{(s+1)T} \tag{5}$$

which tends to minimize $||\mathbf{Y} - \mathbf{D}\mathbf{X}||_F^2$ with respect to $\mathbf{D}$ for $0 < \eta < 2/|||X|||^2$. The dictionary is then normalized:

$$\forall \leq k \leq K, \mathbf{d}_k = \frac{1}{\|\tilde{d}_k\|_2}\,\tilde{\mathbf{d}}_k. \tag{6}$$

One may also use Moore-Penrose pseudo-inverse following the MOD [EAHH99]:

$$\begin{aligned} \tilde{\mathbf{D}}^{(s+1)} &= \text{argmin}_{\mathbf{D}} \quad \frac{1}{2}||\mathbf{Y} - \mathbf{D}\mathbf{X}^{(s+1)}||_2^2 \\ &= \mathbf{Y}\mathbf{X}^{(s+1)T} \cdot \left(\mathbf{X}^{(s+1)}\mathbf{X}^{(s+1)T}\right)^{-1} \end{aligned} \tag{7}$$

again followed by a normalization step. Other more sophisticated methods have also been proposed like FOCUSS [MKD01], K-SVD [AEB06] or the majorization method [YBD09]. We do not discuss all these methods here for sake of briefness. In the following, it appears that methods rooted in the simple gradient descent update is the easiest to adapt to the distributed diffusion strategy. The comparison of performances of various methods is under study.

4 Distributed dictionary learning

4.1 Diffusion strategies for distributed estimation

This section presents one particular effective diffusion strategy to solve LMS distributed estimation problems, see [CS10, ZS12] for a detailed presentation. Here we focus on the Adapt-Then-Combine (ATC) strategy.

The Adapt-Then-Combine (ATC) strategy aims at solving the problem of a *scalar* least mean squares linear regression over a sensor network. Observations $y_n(i)$ are assumed to arrive sequentially at consecutive instants i. In the usual setting [CS10], each sensor records both a noisy *scalar* measurement $y_n(i) \in \mathbb{R}$ and a set of coefficients $\mathbf{x}_n(i)$ under the assumption

$$y_n(i) = \mathbf{w}_o^T \mathbf{x}_{n,i} + z_n(i). \tag{8}$$

The objective is to collectively estimate $\mathbf{w}_o$. The purpose of ATC is that the sensors $\{n : 1...N\}$ yield estimates $\mathbf{w}_n$ of the common underlying regression vector $\mathbf{w}_o$ from observations $\{y_n(i); \mathbf{x}_n(i)\}$ at time i. The cost function under the assumption of Gaussian noise is:

$$J(\mathbf{x}, \mathbf{w}) = \sum_{n=1}^{N} \underbrace{\mathbb{E}|y_n(i) - \mathbf{w}^T \mathbf{x}_{n,i}|^2}_{J_{loc}(\mathbf{x}_{n,i}, \mathbf{w})} \tag{9}$$

Let $\mathbf{A},\ \mathbf{C} \in (\mathbb{R}^+)^{N \times N}$ two matrices such that:

$$\begin{cases} c_{\ell,n} = a_{\ell,n} = 0 \text{ if } \ell \notin \mathcal{N}_n, \\ \mathbf{1}^T \mathbf{C} = \mathbf{1}^T, \mathbf{C}\mathbf{1} = \mathbf{1}, \mathbf{1}^T \mathbf{A} = \mathbf{1}^T \end{cases} \tag{10}$$

where $\mathbf{1}$ is column vector of ones. The ATC algorithm consists of 2 steps:

$$\begin{aligned} \boldsymbol{\psi}_{n,i} &= \mathbf{w}_{n,i-1} + \qquad (Adapt) \\ &\quad \mu_n^w \sum_{\ell \in \mathcal{N}_n} c_{\ell,n}^w \underbrace{\mathbf{x}_{\ell,i-1}[y_\ell(i) - \mathbf{w}_{n,i-1}^T \mathbf{x}_{\ell,i-1}]}_{\nabla_w J_{loc}(\mathbf{x}_{\ell,i-1}, \mathbf{w}_{n,i-1})} \end{aligned} \tag{11}$$

$$\mathbf{w}_{n,i} = \sum_{\ell \in \mathcal{N}_n} a_{\ell,n}^w \boldsymbol{\psi}_{\ell,i} \qquad (Combine) \tag{12}$$

The ATC algorithm can be seen as a distributed gradient descent where each sensor tries to estimate $\mathbf{w}^o$ as $\mathbf{w}_{n,i}$ by exploiting its own measurement $y_n(i)$ as well as information shared with its neighbors. Eq. (11) is the *Adapt* or *incremental* step, eq. (12) is the *Combine* or *diffusion* step which averages estimates from neighbors of node n. As a consequence, a local (possibly averaged if $\mathbf{C} \neq \mathbb{I}$) gradient with respect to $\mathbf{w}$ is computed at each node. An intermediate updated version of the local estimate of $\mathbf{w}_o$ denoted by $\psi_{n,i}$ is then obtained. The final estimate at each node is a local average of neighboring intermediate estimates.

In the sequel, we will focus on the case where observations are not shared between nodes so that matrix $\mathbf{C} = (c_{\ell,n})$ is simply identity $\mathbf{C} = \mathbb{I}$. Various choices can be considered for $\mathbf{A}$. In the numerical experiments below we typically work with either some a priori fixed matrix $\mathbf{A}$ or with the relative degree variance:

$$a_{\ell,n} = \frac{\nu_\ell \sigma_\ell^2}{\sum_{m \in \mathcal{N}_n} \nu_m \sigma_m^2} \tag{13}$$

The performance analysis of this ATC diffusion strategies and some other variants can be found in [ZS12]. The mean-square error of the ATC estimate of $\mathbf{w}_o$ is similar to that of the centralized version (which would see all the observations at once). As a conclusion, this diffusion strategy is very powerful to deal with a distributed solution to a linear regression problem. Let us emphasize that in this setting each observation is made of a couple $(y_n, \mathbf{x}_n)$ where y_n is a scalar. In the dictionary learning problem, only the *vector* $\mathbf{y}_n$ will be observed and both the dictionary (therefore $\mathbf{D}$ in place of $\mathbf{w}_o$) and the coefficient $\mathbf{x}_n$ are to be jointly estimated: this is a factorization problem.

4.2 Distributed alternate optimization for dictionary learning

The ATC diffusion strategy for distributed estimation described above originates the following approach to distributed block-coordinate descent (alternate optimization) for dictionary learning. We will mainly keep the concept of *diffusion* to ensure communication between nodes: every node will share its dictionary estimate with its neighbors in $\mathcal{N}_n$. Let us remark some differences in our setting compared to setting of section 4.1. Observations will be the *vectors* (not only scalar) $\mathbf{y}_n(i), i = 1...q_n$ at node n. Observations are taken simultaneously at each node, not sequentially, so that a whole data matrix $\mathbf{Y}_n$ is assumed to be available at node n. Here index i stands for iterations. The case where data arrive sequentially at each node can also be dealt with at the price of a natural adaptation of the present approach. Note that the $\mathbf{x}_{n,i}$ are not known anymore: each node must estimate both its local dictionary $\mathbf{D}_n$ and the coefficients $\mathbf{X}_n$ which describe observations $\mathbf{Y}_n = \mathbf{D}_n \mathbf{X}_n + \mathbf{Z}_n$. At each iteration i, only the local dictionary estimates $\mathbf{D}_{n,i}$ are assumed to be shared between neighbors, not observations, so that $\mathbf{C} = \mathbb{I}$ in eq. (10). The algorithm goes

Algorithm 1: ATC for sparse dictionary learning

Initialize $\mathbf{D}_{n,0}$, $\forall n$ (see in the text for various options).
Given a matrix $\mathbf{A}$ satisfying (10), $i = 0$,

Repeat until convergence of $(\mathbf{D}_{n,i}, \mathbf{X}_{n,i})_{n=1:N}$

For each node n **repeat:**

1) Optimization w.r.t. $\mathbf{X}_{n,i}$ (sparse rep.):
Given the dictionary, the coefficients are iteratively updated through
For $t = 1 : M$ (typically $M = 30$)
i) $\mathbf{X}_{n,i}^{(t+1/2)} = \mathbf{X}_{n,i}^{(t)} + \lambda_n \mu_n^X \mathbf{D}_{n,i}^T (\mathbf{Y}_n - \mathbf{D}_{n,i}\mathbf{X}_{n,i})$ (gradient descent step)
ii) $\mathbf{X}_{n,i}^{(t+1)} = \text{SoftThreshold}_{\lambda_n \mu_n^X}(\mathbf{X}_{n,i}^{(t+1/2)})$
EndFor (t)

2) Optimization w.r.t. $\mathbf{D}_{n,i}$ (dictionary):

$$\begin{cases} \boldsymbol{\psi}_{n,i+1} &= \mathbf{D}_{n,i} + \mu_n^D (\mathbf{Y}_n - \mathbf{D}_{n,i}\mathbf{X}_{n,i})\mathbf{X}_{n,i}^T \\ \mathbf{D}_{n,i+1} &= \sum_{\ell \in \mathcal{N}_k} a_{\ell,n}^D \boldsymbol{\psi}_{\ell,i} \text{ (diffusion)} \end{cases}$$

EndFor (n)

$i \leftarrow i + 1$
EndRepeat

as follows. First the local dictionaries $\mathbf{D}_{n,0}$ are initialized to a random set of K observations (columns) from $\mathbf{Y}_n$ at node n. Then we iteratively solve the sparse representation problem (4) at each node, for instance using the forward-backward splitting method over a large number M of iterations, see section 3.3. The penalty parameter λ_n may be adjusted for each node according to the local noise level σ_n^2. Note that positive learning rates μ_n^X (resp. μ_n^D) must obey the condition $\mu_n^X < \frac{2}{|||\mathbf{D}_{n,i}|||^2}$ (resp. $\mu_n^D < \frac{2}{|||\mathbf{X}_{n,i}|||^2}$).

In summary, each node updates its dictionary as a function of its local observations $\mathbf{Y}_n$ (Adapt step) and its neighbors' dictionaries (Combine step). Sparse representations are computed locally. Based on known results for the ATC strategy in its usual setting, we expect the present Algorithm 1 above converges to an accurate estimate of the common underlying dictionary $\mathbf{D}$. Next section supports this intuition thanks to numerical experiments on images.

5 Numerical experiments & results

We present some numerical experiments to illustrate the relevance and efficiency of our approach. In the spirit of the seminal work by Olshausen & Field [OF96] our algorithm was tested on datasets containing controlled forms of sparse structure. We consider a set of $r \times r$ image patches composed of sparse pixels. Each pixel was activated independently according to an exponential distribution, $P(x) \propto e^{-|x|}$.

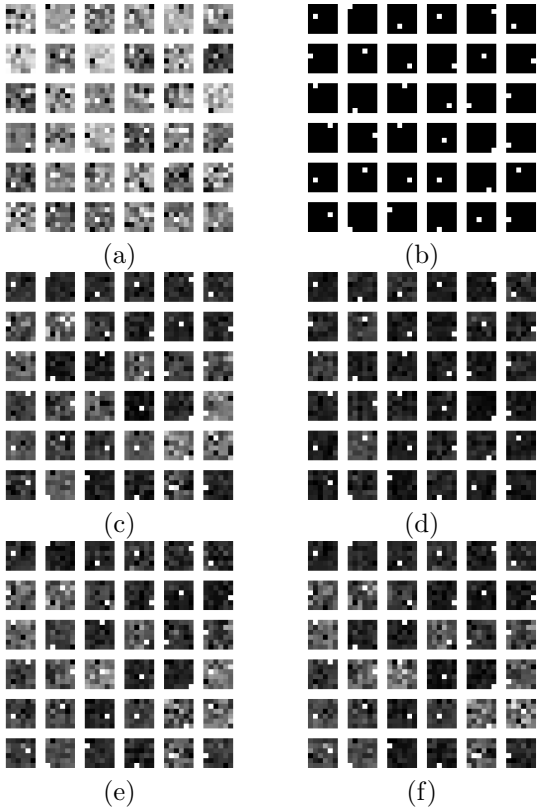

Figure 1: (a) Examples of patches, (b) true dictionary used for synthesis, (c) & (e) local dictionaries for 2 different nodes, (d) dictionary learnt from the usual centralized learning, (f) dictionary averaged over nodes' estimates. Atoms have been reordered to make comparisons easier.

We consider the simple situation of a set of 4 nodes in a symmetrically connected network. Thus we used a symmetric matrix $\mathbf{A}$ such that:

$$\mathbf{A} = \begin{bmatrix} 0.6 & 0.2 & 0 & 0.2 \\ 0.2 & 0.6 & 0.2 & 0 \\ 0 & 0.2 & 0.6 & 0.2 \\ 0.2 & 0 & 0.2 & 0.6 \end{bmatrix} \tag{14}$$

Note that nodes are not even directly connected one to all the others.

Fig. 1 shows that all the nodes have consistently learnt the same dictionary. Let us emphasize that these dictionaries are consistent, in the sense that no local

reordering was necessary at any step. All the nodal dictionaries $\mathbf{D}_n$ are close to the same common dictionary $\mathbf{D}$. It appears that the mean-square error over all estimates is similar to that obtained from the centralized dictionary learning procedure of section 3. Therefore even though each nodeslocally solves a matrix factorization problem from a particular disjoint subset of observations, the same common dictionary is (approximately) identified. This is made possible by the diffusion principle which relies on a simple communication between neighbors only.

6 Conclusion & Prospects

As a conclusion, we have presented an original algorithm which solves the problem of distributed dictionary learning over a sensor network. This is made possible thanks to a diffusion strategy which permits some local communication between neighbors. Connected nodes can exchange their local dictionaries which are estimated from disjoint subsets of data. This algorithm is the adaptation of usual dictionary learning techniques for sparse representation to the context of in-network computing. Some numerical experiments illustrate the relevance of our approach. The theoretical study of the algorithm is the subject of ongoing work. Several improvements and generalizations can also be considered. Many methods are available for sparse coding. This choice is crucial to get better dictionary estimates. We will study which method is most adapted to this distributed setting. The optimization of communication coefficients may be of some help as well.

We believe that this approach to the general problem of distributed matrix factorization opens the way towards many prospects and applications. Moreover, as far as computational complexity is concerned, distributed parallel implementations are a potentially interesting alternative to online learning techniques [MBPS10].

References

[AEB06] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311 –4322, nov. 2006.

[CS10] F.S. Cattivelli and A.H. Sayed. Diffusion lms strategies for distributed estimation. *Signal Processing, IEEE Transactions on*, 58(3):1035 –1048, march 2010.

[EAHH99] K. Engan, S.O. Aase, and J. Hakon Husoy. Method of optimal directions for frame design. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 5, pages 2443–2446 vol.5, 1999.

[MBPS10] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, March 2010.

[MKD01] J.F. Murray and K. Kreutz-Delgado. An improved focuss-based learning algorithm for solving sparse linear inverse problems. In *Proc. of 35th Asilomar Conf. on Sig. Sys. & Comp.*, volume 1, pages 347–351, nov. 2001.

[OF96] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[SMF10] J.-L. Starck, F. Murtagh, and J. Fadili. *Sparse Image and Signal Processing: Wavelets, Curvelets, Morphological Diversity*. Cambridge University Press, 2010.

[TF11] I. Tosic and P. Frossard. Dictionary learning. *IEEE Signal Processing Magazine*, 28(2):27 –38, march 2011.

[Tse01] P. Tseng. Convergence of a block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:475–494, 2001.

[YBD09] M. Yaghoobi, T. Blumensath, and M.E. Davies. Dictionary learning for sparse approximations with the majorization method. *Signal Processing, IEEE Transactions on*, 57(6):2178 –2191, june 2009.

[ZS12] Xiaochuan Zhao and Ali H. Sayed. Performance limits for distributed estimation over lms adaptive networks. *IEEE Transactions on Signal Processing*, 60(10):5107–5124, 2012.