LOW DIMENSIONAL SUBSPACE FINDING VIA SIZE-REDUCING DICTIONARY LEARNING

Bogdan Dumitrescu, Paul Irofti

Department of Automatic Control and Computers
University Politehnica of Bucharest
313 Spl. Independenţei, 060042 Bucharest, Romania
E-mails: bogdan.dumitrescu@acse.pub.ro, paul@irofti.net

ABSTRACT

We present a dictionary learning algorithm that aims to reduce the size of the dictionary to a parsimonious value during the learning process. The sparse coding step uses a weighted Orthogonal Matching Pursuit favoring atoms that enter more representations. The dictionary update step optimizes a regularized error, encouraging the apparition of zero rows in the representation matrix; the corresponding unused atoms are eliminated. The algorithm is extended to the case of incomplete data. Besides dictionary learning, the algorithm is also shown to be useful for finding low-dimensional subspaces. Such versatility is a feature with little precedent. Numerical examples show good convergence properties.

Index Terms— sparse representation, dictionary learning, low-rank representation

1. INTRODUCTION AND MAIN PROBLEMS

The starting point of our discussion is the problem of dictionary learning (DL) for sparse representations [1, 2]. Given the signals matrix $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$, we want to find a dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$ and a sparse matrix $\boldsymbol{X} \in \mathbb{R}^{n \times N}$ such that the representation error $\|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2$ is small. The learned dictionary $\boldsymbol{D}$ is the main outcome; its columns are named atoms and have norm equal to 1. The sparsity constraint is usually expressed by enforcing the representation matrix $\boldsymbol{X}$ to have at most s nonzero elements on each column. Since each signal is represented as a linear combination of at most s atoms, the underlying problem is that of sparse representation (SR). Typically, the dictionaries are overcomplete, i.e. $m < n$.

A related problem is that of low-rank representation (LRR) [3, 4], where the signals are known to lie in a single low-dimensional subspace, not in several, like in the SR case. The dictionary $\boldsymbol{D}$ is a tall matrix, with $m > n$, and n may even be very small; the subspace dimension n is usually not a priori known. The representation matrix $\boldsymbol{X}$ is now full.

In both the above problems we want to obtain a naturally parsimonious representation which ensures a small representation error. Parsimony is seen here as related to the dictionary size n. In the standard DL problem, we are interested in having the smallest number of atoms that otherwise satisfies representation requirements, like e.g. a preset error level, or other performance indicators depending on the application at hand. Too many atoms often do not improve representation, but certainly increase the complexity of operating with the dictionary. In the LRR problem, parsimony is more obviously stated: we want to find the number n of atoms which is the correct dimension of the subspace where the signals lie.

We treat both SR and LRR problems in the DL framework, by designing a size-reducing DL algorithm. The typically iterative DL design is initialized with a large dictionary and the number of atoms is encouraged to decrease in time, until reaching a low value that ideally is the exact subspace dimension in LRR, but has no precise meaning in SR, although it follows a general ideal of parsimony. The size-reducing DL algorithm, based on Approximated K-SVD (AK-SVD) [5, 6], is presented in Section 2. We adapt the algorithm to the case of incomplete data, where some randomly chosen elements of the matrix $\boldsymbol{Y}$ are missing, in Section 3. We show the good behavior of our algorithms in Section 4, insisting on the LRR problem with incomplete data.

Here are some relations with previous work. There are several algorithms that attempt finding an efficient dictionary size in the DL problem. Some [7, 8] start with a small dictionary and grow it until reaching a predefined error level. Others [9, 10, 11] start with a large dictionary and use diverse clustering and pruning techniques to reduce the number of atoms. In [12] a Bayesian approach is used to decide what atoms are significant. All these algorithms could be used in principle for the LRR problem, but there is no previous attempt in this direction, certainly not for incomplete data. Our algorithm is different as it alters the optimization objective

This work was supported by the Romanian National Authority for Scientific Research, CNCS - UEFISCDI, project number PN-II-ID-PCE-2011-3-0400.

and then derives an adapted dictionary update algorithm following exactly AK-SVD, the least complex among the popular DL algorithms. Also, the sparse coding stage is a minimally modified Orthogonal Matching Pursuit (OMP) [13]. So, our algorithm has low complexity; although starting with a larger than desired number of atoms, it typically reduces this number very quickly.

There are very few DL algorithms for incomplete data, notably [14, 15] based on Bayesian dictionary learning, tuned for imaging applications.

For the LRR problem with incomplete data, we look at the big data context, where iterative algorithms with relatively low complexity per iteration are sought. Some are still based on singular value decomposition (SVD), associated with hard [3] or soft thresholding [4]; they can find the subspace dimension, but may have large complexity. Online algorithms are less complex, but use the true n [16] (otherwise they may diverge) or a fixed n [17]. Our algorithm combines the low complexity features of online algorithms (having however a batch character) with a good numerical performance and is different in character from all the above LRR algorithms.

2. SIZE-REDUCING DICTIONARY LEARNING

Many DL algorithms are built on an alternate optimization structure, in which the dictionary $\boldsymbol{D}$ and the representations $\boldsymbol{X}$ are optimized one at a time, the other being fixed. More precisely, we follow the framework of K-SVD [5] and AK-SVD [6], in which each iteration has two steps. In the *sparse coding* step, given the dictionary, the sparse representations $\boldsymbol{X}$ are computed; typically, OMP is used to this purpose. In the *dictionary update* step, the atoms and their corresponding representation coefficients are updated sequentially; the pattern of nonzeros is fixed. In this section we describe the modifications made to these steps such that our goals are attained. First we modify OMP such that atoms that are less used in the current representations are selected less than atoms that are "popular". Then we describe a dictionary update stage in the style of AK-SVD, but working with an objective that combines representation error with a term that favors the reduction of the number of used atoms.

2.1. Size-reducing OMP

Standard OMP represents a signal by greedily selecting atoms that have large projections on the signal. Denoting $\boldsymbol{y} \in \mathbb{R}^m$ a signal, $\boldsymbol{x} \in \mathbb{R}^n$ its current sparse representation (initially equal to zero) and $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{D}\boldsymbol{x}$ the current residual (initially equal to the signal), OMP adds to the representation the atom $\boldsymbol{d}_j$ for which the projection $|\boldsymbol{r}^T\boldsymbol{d}_j|$ is the largest. Denoting $\mathcal{I}$ the indices of the nonzeros in $\boldsymbol{x}$, the current representation is the least-squares (LS) solution

$$\boldsymbol{x}_{\mathcal{I}} = (\boldsymbol{D}_{\mathcal{I}}^T\boldsymbol{D}_{\mathcal{I}})^{-1}\boldsymbol{D}_{\mathcal{I}}^T\boldsymbol{y}. \tag{1}$$

(By $\boldsymbol{D}_{\mathcal{I}}$ we denote the restriction of $\boldsymbol{D}$ to the columns with indices in $\mathcal{I}$.)

OMP treats the atoms equally. If the probability $p(\boldsymbol{d_j})$ of an atom to enter the representation were known, it was proposed in [18] to use Bayes' rule and thus change the selection criterion according to

$$p(\boldsymbol{d}_j|\boldsymbol{r}) \sim p(\boldsymbol{r}|\boldsymbol{d}_j) \cdot p(\boldsymbol{d_j}), \tag{2}$$

where the probability of the residual given the atom can be naturally taken proportional with the projection, i.e.

$$p(\boldsymbol{r}|\boldsymbol{d}_j) \sim |\boldsymbol{r}^T\boldsymbol{d}_j|. \tag{3}$$

This choice leads to a weighted OMP. Another possibility is given in [19]; however, it needs the variance of the noise. An additive rule is proposed in [20] instead of (2), where a logit term proportional to $\log[p(\boldsymbol{d}_j)/(1-p(\boldsymbol{d}_j))]$ is added to the projection; again, the variance of the noise is needed.

Since our purpose is to reduce the number of atoms and considering the iterative nature of DL algorithms, a natural measure of the probability of an atom is given by the weight of its coefficients with respect to the whole current representation matrix $\boldsymbol{X}$. So, denoting $\boldsymbol{x}_j^T$ the j-th row of matrix $\boldsymbol{X}$, we propose to use (2) with prior probabilities

$$p(\boldsymbol{d_j}) \sim f(\|\boldsymbol{x}_j^T\|^2/\|\boldsymbol{X}\|_F^2), \tag{4}$$

where f is a function that allows adaptation to various situations. In the simplest case, f is the identity. For standard OMP, f is constant.

In the beginning the dictionary has many atoms and starting with the identity function in (4) will favor atoms that contribute in reducing the error for many signals, even though they may be not the optimal choice. However, as the DL process advances and the number of atoms ideally settles, we want OMP to work in the standard mode, where it tries to find the combination of atoms that minimizes the error. So, denoting k the iteration number in the alternate DL process, we propose to use in (4) the function

$$f(\xi) = \xi + \frac{\alpha k}{n}, \tag{5}$$

where $\alpha \geq 0$ is a parameter. As k grows, the values of $f(\xi)$ tend to have similar values for all $\xi \in [0,1]$. It is also possible to switch, at a certain iteration or depending on other factors, from (5) to the constant function, which means switching from weighted to standard OMP.

We use OMP with a combined stopping criterion: either a sparsity level s or an error level ε (OMP stops when $\|\boldsymbol{r}\| \leq \varepsilon\sqrt{m}$, hence ε it the target RMSE per signal element). An artificially high value of the error level can be used to reduce the number of atoms, at least in the first DL iterations.

We denote OMP$(\boldsymbol{y}, \boldsymbol{D}, s, \varepsilon, \alpha)$ a call to our weighted OMP. Note that when the dictionary has at most s atoms and

$\varepsilon = 0$, a situation that may occur in LRR context (and in fact is desirable), then all atoms are involved in the representation, hence no support search is actually necessary. The LS solution (1) is immediately computed on the full support.

2.2. Size-reducing dictionary update

The number of atoms can be reduced indirectly by acting on the representation coefficients during the optimization process. The dictionary update step can be modified by adding a penalty term to the representation error. Note that we adopted the convention that the atoms are normalized, which makes useless any penalty put on their norm. Instead we can act on the representations by optimizing the combined objective

$$\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F^2 + \frac{\mu_1}{2}\|\boldsymbol{X}\|_F^2 + \mu_2 \sum_{i=1}^{n} \|\boldsymbol{x}_i^T\|_2, \qquad (6)$$

where $\boldsymbol{x}_i^T$ is the i-the row of $\boldsymbol{X}$.

The first penalty term is a simple regularization [21] and has the purpose to discourage almost linearly dependent atoms to enter the same representation, case in which their coefficients are typically large. It can also be seen as a rank minimizer, like in [17], via the nuclear norm, which obeys to

$$\|\boldsymbol{D}\boldsymbol{X}\|_* = \frac{1}{2} \min_{\boldsymbol{D},\boldsymbol{X}} \|\boldsymbol{D}\|_F^2 + \|\boldsymbol{X}\|_F^2.$$

In our case $\|\boldsymbol{D}\|_F$ is constant.

The second penalty term is a typical group sparsity inducer, acting on the rows of $\boldsymbol{X}$. If the entire row i of $\boldsymbol{X}$ is zero, then the atom $\boldsymbol{d}_i$ is unused and can be removed from the dictionary. If, as usual in DL, the number n of atoms is fixed, then replacement techniques are used, the simplest being to put a random atom instead of the unused one; on the contrary, here we just remove the atom, as reducing the number of useful atoms is our goal.

The first penalty term has a direct influence on the sparse coding step. The LS solution (1) should be replaced by its regularized form

$$\boldsymbol{x}_{\mathcal{I}} = (\boldsymbol{D}_{\mathcal{I}}^T \boldsymbol{D}_{\mathcal{I}} + \mu_1 \boldsymbol{I})^{-1} \boldsymbol{D}_{\mathcal{I}}^T \boldsymbol{y}. \qquad (7)$$

The second penalty term cannot be directly translated into OMP, as it is expressed in terms of the rows of $\boldsymbol{X}$, while OMP has the columns of $\boldsymbol{X}$ as variables. However, the particular choice of weights (4) favors creation of zero rows in $\boldsymbol{X}$, so it can be seen as a substitute of the second penalty term in the sparse coding step.

Let us now describe the dictionary update step corresponding to (6), when the atoms and representations are optimized one by one. In fact, we adapt the AK-SVD method, where first an atom is optimized, while everything else is fixed, then the corresponding representation coefficients are optimized, again with everything else fixed. Let $\boldsymbol{d}$ be the generic atom that is optimized, namely the "current" column of $\boldsymbol{D}$, with index c. Denote $\mathcal{J}$ the set of the indices of the nonzeros elements in row c of $\boldsymbol{X}$; these are the indices of signals that use $\boldsymbol{d}$ in their representation. The representation error without atom $\boldsymbol{d}$ is

$$\boldsymbol{F} = \left[\boldsymbol{Y} - \sum_{i \neq c} \boldsymbol{d}_i \boldsymbol{x}_i^T\right]_{\mathcal{J}}. \qquad (8)$$

Let $\boldsymbol{x}^T = \boldsymbol{X}_{c,\mathcal{J}}$ be the compressed representation coefficients of the current atom. Then, the objective function corresponding to (6) when only $\boldsymbol{d}$ and $\boldsymbol{x}$ are variable is

$$\phi(\boldsymbol{d}, \boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{F} - \boldsymbol{d}\boldsymbol{x}^T\|_F^2 + \frac{\mu_1}{2}\|\boldsymbol{x}\|_2^2 + \mu_2\|\boldsymbol{x}\|_2. \qquad (9)$$

The penalty terms depend on the representation, so optimizing the atom with fixed representation, under the norm constraint $\|\boldsymbol{d}\| = 1$, leads to the standard AK-SVD formula

$$\boldsymbol{d} = \boldsymbol{F}\boldsymbol{x}/\|\boldsymbol{F}\boldsymbol{x}\|. \qquad (10)$$

To optimize the representation, we note that the gradient of (9) with respect to $\boldsymbol{x}$ is

$$\frac{\partial \phi}{\partial \boldsymbol{x}} = -\boldsymbol{F}^T\boldsymbol{d} + (1+\mu_1)\boldsymbol{x} + \mu_2 \frac{1}{\|\boldsymbol{x}\|}\boldsymbol{x}.$$

The optimal representation satisfies the relation

$$\boldsymbol{x} = \frac{\|\boldsymbol{x}\|}{(1+\mu_1)\|\boldsymbol{x}\| + \mu_2} \cdot \boldsymbol{F}^T\boldsymbol{d}. \qquad (11)$$

This leads to a soft thresholding relation for the optimal norm and to the following expression for the optimal representation

$$\boldsymbol{x} = \max\left(0, \frac{\|\boldsymbol{F}^T\boldsymbol{d}\| - \mu_2}{1+\mu_1}\right) \cdot \frac{1}{\|\boldsymbol{F}^T\boldsymbol{d}\|}\boldsymbol{F}^T\boldsymbol{d}. \qquad (12)$$

2.3. Size-reducing AK-SVD

Gathering all the above information, we obtain the size-reducing AK-SVD Algorithm 1.

For the SR problem, either the sparsity level s is given, case in which the error level is $\varepsilon = 0$, or the error level is given and s takes a value that is large enough (but smaller than m), which may be seen only as a safety measure for the possibly (few) badly represented signals. The regularization parameters μ_1 and μ_2 have relatively small values, while α can be rather large.

For the LRR problem, the sparsity level should be fixed to a value larger than the true subspace dimension. The error level ε should be larger than the noise level for preventing many atoms entering the representations. The parameters μ_1 and especially μ_2 need larger values, in order to reduce the dictionary size. For the same reason, α needs to be small.

In both problems, once the dictionary size settles, the optimization should focus on the approximation error. So, we can

Algorithm 1: An iteration of size-reducing AK-SVD.

Data: current dictionary $\boldsymbol{D} \in \mathbb{R}^{m \times n}$
signals set $\boldsymbol{Y} \in \mathbb{R}^{m \times N}$
sparse representations $\boldsymbol{X} \in \mathbb{R}^{n \times N}$
parameters s, ε, α, μ_1, μ_2
Result: next dictionary $\boldsymbol{D}$, next representations $\boldsymbol{X}$

Compute representations: $\boldsymbol{X} = \text{OMP}(\boldsymbol{y}, \boldsymbol{D}, s, \varepsilon, \alpha)$
Compute error matrix: $\boldsymbol{E} = \boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}$
for $j = 1$ **to** n **do**
  Error without atom j: $\boldsymbol{F} = \boldsymbol{E} + \boldsymbol{d}_j \boldsymbol{x}_j^T$
  Compute new atom $\boldsymbol{d}_j$ with (10)
  Compute new representation row $\boldsymbol{x}_j^T$ with (12)
  Update error: $\boldsymbol{E} = \boldsymbol{F} - \boldsymbol{d}_j \boldsymbol{x}_j^T$
  If $\|\boldsymbol{x}_j\| = 0$, remove atom j (and row j of $\boldsymbol{X}$)
Optional: update parameters ε, μ_1, μ_2

introduce a diminishing factor $\lambda \leq 1$ that multiplies ε, μ_1, μ_2 at the end of each iteration from Algorithm 1. (Although individual factors could be used for each parameter, we prefer a single one for the sake of simplicity.) Also, some parameters should be given default values whenever necessary; for example, in LRR context, when n becomes smaller than s and settles to a certain value, then it is natural to set $\varepsilon = 0$ and hence replace OMP with a regularized LS solution.

Regarding convergence, our algorithm inherits the properties of AK-SVD. In dictionary update, the combined objective is guaranteed to decrease; in sparse coding this no longer happens; however, when OMP is replaced by simple LS (as mentioned above for LRR), the error certainly decreases.

The algorithm has the complexity of AK-SVD; the extra operations due to regularization or weighting are not significant in this respect. Also, there are many operations that can be programmed in parallel. Sparse coding is inherently parallel, as each signal can be coded independently. Dictionary update contains mainly matrix-vector multiplications, again parallel (but with a lower grain).

3. MISSING DATA CASE

We present now a version of the algorithm for the case where the data are incomplete. More precisely, instead of $\boldsymbol{Y}$ we have $\boldsymbol{M} \odot \boldsymbol{Y}$, where the elements of the known mask matrix $\boldsymbol{M}$ are zero or one and $\odot$ is the elementwise product. A zero in $\boldsymbol{M}$ means the respective signal element is not available; the corresponding zero in $\boldsymbol{M} \odot \boldsymbol{Y}$ carries thus no meaning. We assume data are missing with probability ρ, independently of their position.

Somewhat surprisingly, DL with incomplete data has received little attention and was used in particular setups [14, 15, 22]. AK-SVD can be relatively easily adapted to this case. To simplify the exposition, we discuss here the necessary changes for the error objective

$$\|\boldsymbol{M} \odot (\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X})\|_F^2. \tag{13}$$

Adding weigths in OMP and regularization terms in dictionary update can be done like in the full-data versions from the previous section.

An algorithm for OMP with missing data was proposed in [23], providing recovery guarantees. Interestingly, it computes the support of the representation by just applying standard OMP to $\boldsymbol{M} \odot \boldsymbol{Y}$; only the final representation coefficients are computed in a different way, using the naturally available estimate of ρ, which is the zeros density in $\boldsymbol{M}$. In all our tests, this version of OMP was better than standard OMP. It is also tempting to adapt OMP by completely ignoring the missing elements (and the corresponding rows of the dictionary), hence running standard OMP for shorter vectors, which considerably decreases the execution time; for images, this masked OMP was actually the best; in other tests, masked OMP gave similar results with missing-data OMP, so we preferred it in the numerical experiments reported here. Note that when the dictionary has at most s atoms and $\varepsilon = 0$, then masked OMP reduces to the regularized LS solution from [17], all atoms being used in the representation.

The AK-SVD dictionary update adaptation was presented sketchily in [22]. With incomplete data and without penalty terms, the objective (9) becomes

$$\|\boldsymbol{M} \odot (\boldsymbol{F} - \boldsymbol{d}\boldsymbol{x}^T)\|_F^2. \tag{14}$$

Denoting $\mathcal{M} \subset \mathbb{N}^2$ the indices of available data in $\boldsymbol{Y}$, the LS problem corresponding to (14) is

$$d_i x_j = f_{ij}, \;\; (i,j) \in \mathcal{M}. \tag{15}$$

If $\boldsymbol{x}$ is fixed, then the optimal atom has the elements

$$d_i = \frac{\sum_{(i,j)\subset\mathcal{M}} f_{ij} x_j}{\sum_{(i,j)\subset\mathcal{M}} x_j^2}, \;\; i = 1:m. \tag{16}$$

Normalizing the atom, we obtain the missing data version of (10). The numerators in (16) are the elements of the vector $(\boldsymbol{M} \odot \boldsymbol{F})\boldsymbol{x}$. The denominators are the norms of the compressed representations corresponding to the available signals that use d_i in their representation.

Similarly, if the atom is fixed in (15), the optimal representation coefficients are

$$x_j = \frac{\sum_{(i,j)\subset\mathcal{M}} f_{ij} d_i}{\sum_{(i,j)\subset\mathcal{M}} d_i^2}. \tag{17}$$

The numerators are the elements of the vector $(\boldsymbol{M} \odot \boldsymbol{F})^T \boldsymbol{d}$. This is the equivalent of the vector $\boldsymbol{F}^T \boldsymbol{d}$ from (12). Extending (17) to a regularized objective is thus immediate.

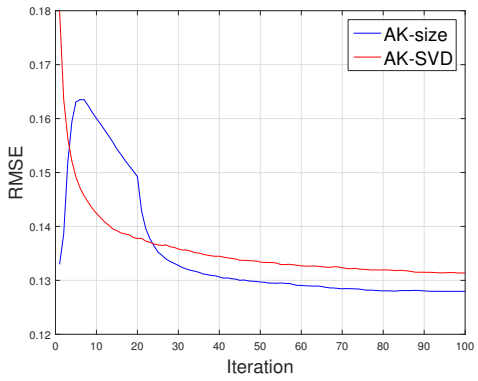

Fig. 1: Size-reducing vs plain AK-SVD.

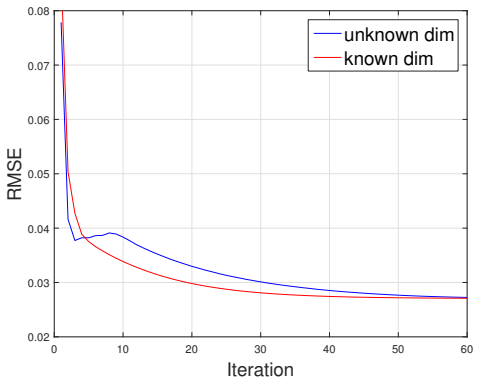

Fig. 2: Low-dimensional subspace finding, full data.

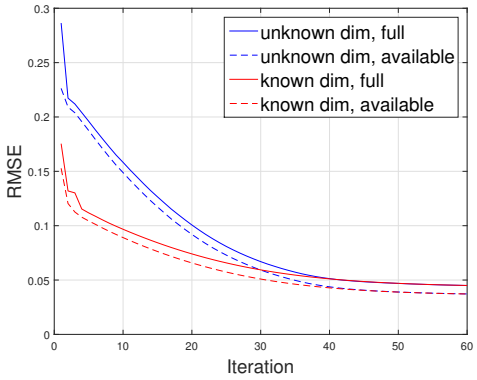

Fig. 3: Low-dimensional subspace finding, incomplete data.

4. NUMERICAL RESULTS

In all numerical experiments reported here we generate the signal matrix $\boldsymbol{Y}$ by multiplying random $\boldsymbol{D}$ and $\boldsymbol{X}$ (normally distributed elements and uniformly random sparsity pattern, if sparsity is required) and adding Gaussian noise at a SNR of 20 dB. We report RMSE values, defined as $\|\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X}\|_F/\sqrt{mN}$, averaged over 10 runs.

In DL, size-reducing AK-SVD can be used not only for finding an economical number of atoms, but also as an initialization method for plain AK-SVD, as show the results from Figure 1, where $m = 20$, $N = 500$, $s = 4$. Size-reducing AK-SVD starts with 100 atoms and uses $\varepsilon = 0.02$ and $\alpha = 0.2$; after 20 iterations it switches to standard OMP and hence to plain AK-SVD. (Regularization is not used, so $\mu_1 = \mu_2 = 0$.) Initially, the error grows due to the decrease in the number of atoms, but then decreases, with the sharpest decrease when switching to standard OMP. The average final number of atoms is 54. For comparison, plain AK-SVD is run with 54 atoms and initialized with a subset of the initialization of size-reducing AK-SVD. Similar relative behavior of the two algorithms can be seen for other parameter values, which lead to various number of atoms.

We give more results for the LRR problem, starting with the full data case, with $m = 50$, $N = 2000$ and true dimension $n = 4$. Size-reducing AK-SVD is run starting with an initial dictionary with 100 atoms; similar results are obtained with smaller numbers. Figure 2 shows the results for two values of the sparsity level s. First, we take $s = 8$, which accounts as an upper bound of the true dimension; to check the results, the algorithm is then run with $s = 4$, hence assuming the true dimension is known; still, the initial dictionary has 100 atoms. The parameter values are $\varepsilon = 0.05$, $\alpha = 0$, $\mu_1 = 0.1$, $\mu_2 = 1$, $\lambda = 0.95$. In all runs, the dictionary is reduced to 4 atoms in less than 10 iterations, thus reaching the true subspace dimension and hence the error level reaches an almost optimal value. In this example the RMSE reached by size-reducing AK-SVD after 60 iterations is 0.271 with $s = 8$ and 0.270 with $s = 4$; using the SVD decomposition of $\boldsymbol{Y}$ (which at this size is possible) and truncating it to the first 4 singular values gives RMSE $= 0.268$.

We go now to the incomplete data case, with the same dimensions as above and a missing ratio $\rho = 0.5$ (half of the data are available, in random positions). Figure 3 shows the results; dashed lines represent the RMSE $\|\boldsymbol{M} \odot (\boldsymbol{Y} - \boldsymbol{D}\boldsymbol{X})\|_F/\sqrt{(1-\rho)mN}$, i.e. the error on the available data only. Some parameters need larger values to force the decrease of dictionary size: $\varepsilon = 0.3$, $\alpha = 0.01$, $\mu_1 = 0.5$, $\mu_2 = 8$. The true subspace dimension is reached in about 5 iterations.

The last example shows the behavior with larger dimensions ($m = 500$, $N = 2000$, true size $n = 10$) and also more missing data ($\rho = 0.9$). Figure 4 shows two RMSE, one with $s = 20$, the other with $s = 10$. When signal size is much bigger than subspace dimension, then the initial dictionary size has little importance in the evolution of the algorithm; in terms of convergence, this problem is actually simpler than the previous, despite the higher missing data ratio; the evolution of the RMSE is almost the same for $s = 20$ and $s = 10$. The parameters can thus be tuned less aggressively: $\varepsilon = 0.02$, $\alpha = 0.01$, $\mu_1 = 0.5$, $\mu_2 = 5$. Again the true size is reached typically in about 5 iterations.

Of course, more work is necessary to automatically select the values of the parameters depending on the data. We note

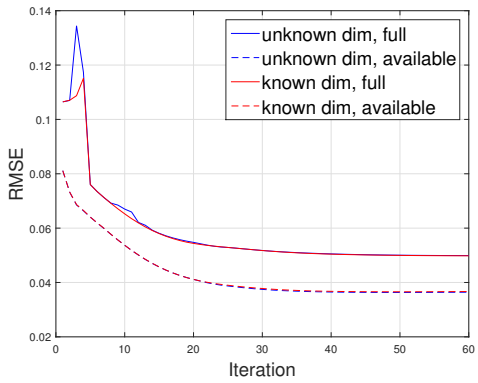

Fig. 4: Low-dimensional subspace finding, incomplete data.

that the values we used are relatively robust, in the sense that small changes do not significantly affect the outcome. They are also sub-optimal, as we did not systematically search for the best values; hence, improvements in convergence speed are possible.

5. CONCLUSIONS AND FUTURE WORK

We have presented a double-purpose dictionary learning algorithm that reduces the size of the dictionary and can also be successfully used to find low-rank representations. It also works well with incomplete data.

A main topic of future work is to design adaptive algorithms, thus grouping under the same form online dictionary learning and subspace tracking.

6. REFERENCES

[1] R. Rubinstein, A.M. Bruckstein, and M. Elad, "Dictionaries for Sparse Representations Modeling," *Proc. IEEE*, vol. 98, no. 6, pp. 1045–1057, June 2010.

[2] I. Tosic and P. Frossard, "Dictionary Learning," *IEEE Signal Proc. Mag.*, vol. 28, no. 2, pp. 27–38, Mar. 2011.

[3] E.J. Cai, J.F. Candes and Z. Shen, "A Singular Value Thresholding Algorithm for Matrix Completion," *SIAM J. Optim.*, vol. 20, no. 4, pp. 1956–1982, 2010.

[4] S. Ma, D. Goldfarb, and L. Chen, "Fixed Point and Bregman Iterative Methods for Matrix Rank Minimization," *Math. Prog., Ser. A*, vol. 128, pp. 321–353, 2011.

[5] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE Trans. Signal Proc.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[6] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient Implementation of the K-SVD Algorithm Using Batch Orthogonal Matching Pursuit," Tech. Rep. CS-2008-08, Technion Univ., Haifa, Israel, 2008.

[7] C. Rusu and B. Dumitrescu, "Stagewise K-SVD to Design Efficient Dictionaries for Sparse Representations," *IEEE Signal Proc. Letters*, vol. 19, no. 10, pp. 631–634, Oct. 2012.

[8] M. Marsousi, K. Abhari, P. Babyn, and J. Alirezaie, "An Adaptive Approach to Learn Overcomplete Dictionaries With Efficient Numbers of Elements," *IEEE Trans. Signal Proc.*, vol. 62, no. 12, pp. 3272–3283, June 2014.

[9] R. Mazhar and P. D. Gader, "EK-SVD: Optimized Dictionary Design for Sparse Representations," in *19th Int. Conf. Pattern Recognition*, 2008, pp. 1–4.

[10] J. Feng, L. Song, X. Yang, and W. Zhang, "Learning Dictionary via Subspace Segmentation for Sparse Representation," in *18th IEEE Int. Conf. Image Proc.*, 2011, pp. 1245–1248.

[11] N. Rao and F. Porikli, "A Clustering Approach to Optimize Online Dictionary Learning," in *Int. Conf. Acoustics Speech Signal Proc. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 1293–1296.

[12] H.P. Dang and P. Chainais, "A Bayesian Nonparametric Approach to Learn Dictionaries with Adapted Number of Atoms," in *IEEE Workshop Mach. Learning for Signal Proc.*, 2015.

[13] Y.C. Pati, R. Rezaiifar, and P.S. Krishnaprasad, "Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition," in *27th Asilomar Conf. Signals Systems Computers*, Nov. 1993, vol. 1, pp. 40–44.

[14] M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, and L. Carin, "Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images," *IEEE Trans. Image Proc.*, vol. 21, no. 1, pp. 130–144, Jan. 2012.

[15] Z. Xing, M. Zhou, A. Castrodad, G. Sapiro, and L. Carin, "Dictionary Learning for Noisy and Incomplete Hyperspectral Images," *SIAM J. Imaging Sciences*, vol. 5, no. 1, pp. 33–56, 2012.

[16] Y. Chi, Y.C. Eldar, and R. Calderbank, "PETRELS: Parallel Subspace Estimation and Tracking Using Recursive Least Squares from Partial Observations," *IEEE Trans. Signal Proc.*, vol. 61, no. 23, pp. 5947–5959, Nov. 2013.

[17] M. Mardani, G. Mateos, and G.B. Giannakis, "Subspace Learning and Imputation for Streaming Big Data Matrices and Tensors," *IEEE Trans. Signal Proc.*, vol. 63, no. 10, pp. 2663–2677, May 2015.

[18] O.D. Escoda, L. Granai, and P. Vandergheynst, "On the Use of A Priori Information for Sparse Signal Approximations," *IEEE Trans. Signal Proc.*, vol. 54, no. 9, pp. 3468–3482, Sep. 2006.

[19] M. Elad and I. Yavneh, "A Plurality of Sparse Representations Is Better Than the Sparsest One Alone," *IEEE Trans. Info. Th.*, vol. 55, no. 10, pp. 4701–4714, Oct. 2009.

[20] J. Scarlett, J.S. Evans, and S. Dey, "Compressed Sensing With Prior Information: Information-Theoretic Limits and Practical Decoders," *IEEE Trans. Signal Proc.*, vol. 61, no. 2, pp. 427–439, Jan. 2013.

[21] W. Dai, T. Xu, and W. Wang, "Simultaneous Codeword Optimization (SimCO) for Dictionary Update and Learning," *IEEE Trans. Signal Proc.*, vol. 60, no. 12, pp. 6340–6353, Dec. 2012.

[22] C. Guichaoua, "Dictionary Learning for Audio Inpainting," 2012, HAL Robotics [cs.RO]. 2012. http://dumas.ccsd.cnrs.fr/dumas-00725263.

[23] Y. Chen and C. Caramanis, "Noisy and Missing Data Regression: Distribution-Oblivious Support Recovery," in *Int. Conf. Mach. Learning*, Atlanta, Georgia, 2013.