SHIFT-INVARIANT DICTIONARY LEARNING FOR SPARSE REPRESENTATIONS: EXTENDING *K-SVD*

Boris Mailhé, Sylvain Lesage,Rémi Gribonval and Frédéric Bimbot

Projet METISS
Centre de Recherche INRIA Rennes - Bretagne Atlantique
IRISA, Campus de Beaulieu
F-35042 Rennes Cedex, France
E-mail: firstname.lastname@irisa.fr

Pierre Vandergheynst

Signal Processing Laboratory (LTS2)
School of Engineering, EPFL
Station 11, CH - 1015 Lausanne, Switzerland
E-mail: firstname.lastname@epfl.ch

ABSTRACT

Shift-invariant dictionaries are generated by taking all the possible shifts of a few short patterns. They are helpful to represent long signals where the same pattern can appear several times at different positions. We present an algorithm that learns shift invariant dictionaries from long training signals. This algorithm is an extension of *K-SVD*. It alternates a sparse decomposition step and a dictionary update step. The update is more difficult in the shift-invariant case because of occurrences of the same pattern that overlap. We propose and evaluate an *unbiased* extension of the method used in *K-SVD*, *i.e.* a method able to exactly retrieve the original dictionary in a noiseless case.

1. INTRODUCTION

Obtaining a sparse representation of a large signal is a key preprocessing step for many applications. Many common classes of signals are known to be well representable on some analytical basis: wavelets for natural images, Gabor bases for sounds [1],... For more complex classes, it might not be possible to find one basis that can represent the whole class efficiently. Then an overcomplete dictionary containing more vectors called *atoms* than the dimension of the signal space must be used. For example a musical signal often contains short and long notes, thus a multiscale Gabor dictionary would perform better than a simple Gabor basis.

When one does not know an analytic dictionary that fits a signal class, it can be useful to learn such a dictionary. To do this one usually relies on several examples of signals chosen among the class and searches the dictionary of a given size that minimizes the approximation error under a sparsity constraint. Roughly speaking, the *K-SVD* algorithm [2] attempts to perform this job by alternating a sparse approximation of the training signals on the current dictionary and the optimization of the dictionary according to the computed decomposition.

When the signals are known to be shift-invariant, one would also like to learn a shift-invariant dictionary generated by all the possible shifts of a set of *patterns*. A limited set of patterns can generate a huge dictionary while keeping the number of free parameters low. Standard *K-SVD* and many other learning algorithms do not allow this. The only way they have to deal with large signals is to split them into small frames and to consider those frames as the training set. While this enables the learning of time-localized atoms, their position is arbitrary and each shift of a pattern has to be learnt individually.

In this paper we present an extension of *K-SVD* that learns this set of patterns from a long signal.

2. PRINCIPLE

2.1 The problem

A dictionary D is a matrix whose columns d_k are called *atoms*. A signal s has a sparse representation on D if it can be decomposed as a linear combination $s = Dc$ where the number of non-zero coefficients in c, usually noted $\|c\|_0$, is much smaller than the length (*i.e.* the number of samples) of s.

Given a family of signals s_l, the dictionary learning problem can be expressed as the computation of a dictionary that minimizes the approximation error under a hard sparsity constraint:

$$\min_{\|c\|_0 \leq L} \sum_l \|s_l - Dc_l\|_2^2$$

where L is the maximum number of atoms allowed.

In the shift-invariant case, the learning is performed on one long signal s instead of a training set and the dictionary D is built by shifting a family M of *patterns* m: $M = (m_k)_{1 \leq k \leq K}$. Now it is this set of patterns that has to be learnt and the learning problem is defined by the new objective function:

$$\begin{aligned} &\min_{\|c\|_0 \leq L} \left\| s - \sum_k \sum_\tau c_{k,\tau} m_k(t-\tau) \right\|_2^2 \\ &= \min_{\|c\|_0 \leq L} \left\| s - \sum_k \sum_\tau c_{k,\tau} T_\tau m_k \right\|_2^2 \end{aligned} \quad (1)$$

where T_τ is the shift operator that takes a pattern m and returns an atom that is null everywhere except for a copy of m that starts at instant τ. So we have $D = (T_\tau m_k)_{k,\tau}$. For this work we only considered integer shifts τ.

2.2 Global overview

Shift-invariant *K-SVD* follows an iterative strategy described in Algorithm 1.

The *decomp* function tries to compute the best decomposition (*i.e.* the best c) of the signal on the current dictionary and the *update* function computes the dictionary that minimizes Criterion (1) with

Algorithm 1 $(M,c,\tau) \leftarrow$ *K-SVD*(s,L,M_{init})

$M \leftarrow M_{init}$
while the algorithm has not converged **do**
  $c \leftarrow \text{decomp}(s,M,L)$
  $(M,c) \leftarrow \text{update}(M,s,c)$
end while

fixed τ. This alternate optimization has already been widely used in dictionary learning [3][7][8]. The main difference in our approach is that we don't rely on the value of the amplitude coefficients, but only on the instants where they are not null.

Whereas the external loop described in Algorithm 1 is very similar to the *K-means* one, each inner step is more difficult. The sparse decomposition problem is combinatorial, so we will have to use a suboptimal algorithm. The dictionary update also becomes a hard problem. In *K-means*, each pattern can be updated independently from the others. This is not the case here: Objective function (1) cannot be separated in individual objectives for each pattern. In this work, we dealt with it by updating each pattern m_k successively and update the amplitude coefficients $c_{k,\tau}$ accordingly before updating the next pattern. Engan et al. [7] propose a way to update all the patterns at the same time, but they do so with fixed coefficients.

3. SPARSE DECOMPOSITION STEP

Finding the closest sparse approximation of a signal on a given dictionary is a combinatorial problem, but many sub-optimal algorithms are known. They can be sorted in 2 main groups: greedy algorithms which select atoms successively and optimisation algorithms that solve a constrained least-square problem. Both have been proven to give the right solution when the dictionary is incoherent [4], *i.e.* when different atoms of the dictionary are little correlated.

Unfortunately, this is not the case here as there is few difference between close shifts of a given pattern. We need an algorithm that does not perform too bad with highly coherent dictionaries, and that can deal with very large dictionaries: the size of our dictionaries will roughly equal the number of patterns times the length of the signal, generally several million atoms. For all these reasons we chose Matching Pursuit (MP).

3.1 Matching Pursuit (MP)

MP is a greedy algorithm. At each step, it selects the atom that has the highest correlation with the current residual, adds it to the book and subtracts it from the residual. The basic algorithm is explained in Algorithm 2. A fast C++ implementation has been developed for the shift invariant case [1]. It deals efficiently with long signals (up to memory capacity) without needing to split them thanks to the use of fast convolution for correlation computation, local update at each iteration and a tournament tree for the fast retrieval of the best atom. More explanations about this implementaton can be found in [5].

Algorithm 2 $c \leftarrow \text{decomp}(s,M,L)$

$r_0 = s$
$\forall k, \forall \tau, c_{k,\tau,0} = 0$
for $i = 1$ **to** I **do**
  $(k_i,\tau_i) = \text{argmax}_{(k,\tau)} \langle r_{i-1}, T_\tau m_k \rangle$ {select best atom}
  $\gamma_i = \langle r_{i-1}, T_{\tau_i} m_{k_i} \rangle$
  $c_{k_i,\tau_i,i} = c_{k_i,\tau_i,i-1} + \gamma_i$ {store best atom}
  $\forall (k,\tau) \neq (k_i,\tau_i), c_{k,\tau,i} = c_{k,\tau,i-1}$
  $r_i = r_{i-1} - \gamma_i T_{\tau_i} m_{k_i}$ {subtract from residual}
end for

[1] `http://mptk.gforge.inria.fr/`

3.2 MP stopping criterion

The main parameter to tune with this approach is the number of iterations I that bounds the decomposition length L. It can often be roughly estimated via simple preprocessing, such as spike detection. Moreover, we have observed that the algorithm is quite robust to overestimating the number of atoms, *i.e.* choosing L larger than really needed. Such an experiment is shown in Section 5.2.2.

The problem can also be formulated as a constrained problem, trying to minimize

$$\|r_i\|_2^2 + \lambda \|c_i\|_0$$

over the iterations i instead of just performing a fixed number of iterations. Such a function is very easy to compute after each MP iteration. As the last selected atom is orthogonal to the last residual, the error update rule is given by

$$\|r_{i+1}\|_2^2 = \|r_i\|_2^2 - \gamma_{\max}^2$$

The l_0 norm of the coefficients can be computed greedily: it is equal to 0 at the beginning of the algorithm and is incremented each time a new atom is selected.

However, with such an approach there is still the λ parameter to tune. Parameter-free approaches have been proposed for other algorithms, such as choosing the variance of the residual as λ [3]. This leads to the objective function

$$\|r_i\|_2^2 (1 + \frac{\|c_i\|_0}{T})$$

where T is the number of samples of the signal.

Unfortunately this approach cannot be used for greedy algorithms. If the dictionary is quasi-incoherent, then the error is known to decrease exponentially [6]. In that case, as the l_0 norm of the coefficients is bounded by the number of iterations, the function converges to 0 and never reaches its minimum.

Even with coherent dictionaries, we have observed experimentally that the error is decreasing too fast (*i.e.* faster than $\frac{1}{i}$) and the objective function keeps decreasing even for large decompositions. This was observed for a decomposition of the signal used in Section 5.2 with 200 000 MP iterations, about half the dimension of the signal space.

In the experiments presented here we fixed the number of iterations I.

4. DICTIONARY UPDATE STEP

The patterns are updated to minimize the error according to the given supports $\sigma_k = \{\tau | c_{k,\tau} \neq 0\}$. As in standard *K-SVD*, this is performed successively on each pattern. For a given pattern m_κ, if we name $\hat{s}_\kappa = r + \sum_\tau c_{\kappa,\tau} T_\tau m_\kappa$ the signal without the contributions of the other patterns m_k where $k \neq \kappa$, then the best update pattern m_κ^{opt} is given by:

$$\left(m_\kappa^{opt}, c_\kappa^{opt}\right) = \text{argmin}_{\|m\|_2=1} \left\| \hat{s}_\kappa - \sum_{\tau \in \sigma_\kappa} c_\tau T_\tau m \right\|_2^2$$

4.1 Non-overlapping case

If the different occurrences of the pattern do not overlap, as the shift operators T_τ are unitary, we can easily show that:

$$\forall m, \left\| \hat{s}_\kappa - \sum_{\tau \in \sigma_\kappa} c_\tau T_\tau m \right\|_2^2 = \sum_{\tau \in \sigma_\kappa} \|T_\tau^* \hat{s}_\kappa - c_\tau m\|_2^2 + cst \qquad (2)$$

where T_τ^* is the adjoint of T_τ (*i.e.* the operator such that $\forall m, \forall s, \langle T_\tau m, s \rangle = \langle m, T_\tau^* s \rangle$). It takes a signal and extracts from it a *patch* that has the same length as a pattern and begins at sample τ.

4.1.1 Update with fixed coefficients

If we fix the coefficients c_τ, then the minimum of Expression (2) over m is a simple *weighted mean*:

$$m_\kappa \leftarrow \sum_{\tau \in \sigma_\kappa} c_\tau T_\tau^* \hat{s}_\kappa = \left(\sum_{\tau \in \sigma_\kappa} c_\tau^2 \right) m_\kappa + \sum_{\tau \in \sigma_\kappa} c_\tau T_\tau^* r$$
$$m_\kappa \leftarrow \frac{m_\kappa}{\|m_\kappa\|_2}$$

This update rule is very close to the *Bayesian rule* used in [3], except that the adaptation rate (*i.e.* the trade-off between the old pattern and the residual) here is chosen to minimize the error instead of being fixed.

4.1.2 Joint update of the dictionary and coefficients

There are several reasons not to trust the coefficients given by the decomposition step. First they have been computed by a decomposition on the old dictionary, so we can hope to get better results with joint optimisation than with alternate one. More important, most sparse decomposition algorithms do not compute the orthogonal projection of the signal on the sub-dictionary. Constrained optimisation algorithms often use some other norm than the l_0 norm to get a continuous or convex objective function, so the coefficients they compute are biased by the constraint. There are greedy algorithms such as Orthogonal Matching Pursuit that compute the orthogonal projection of the signal on the sub-dictionary at each step, but current implementations require too much time or memory when used on long signals.

So we prefer to use only the support information from the decomposition, then jointly estimate the value of the coefficients and the new dictionary. This can be performed successively for each pattern. The pattern and coefficients that jointly minimize Criterion (2) are given by the SVD of the patch matrix. The pattern is the principal component of the patches:

$$m_\kappa \leftarrow \text{argmax}_{\|m\|_2=1} \sum_{\tau \in \sigma_\kappa} \langle m, T_\tau^* \hat{s}_\kappa \rangle^2 \tag{3}$$

$$(c_{\kappa,\tau})_{\tau \in \sigma_\kappa} \leftarrow \text{argmin} \left\| \hat{s}_\kappa - \sum_{\tau \in \sigma_\kappa} c_\tau T_\tau m \right\|_2^2 \tag{4}$$

4.2 Overlap handling

If the patches can overlap, then the decomposition (2) does not hold any more. If we still use the previous learning rules, then we introduce a bias in the learning. There have already been several approaches of this problem with fixed coefficients such as Engan et al. [7], Aharon's PhD thesis [8] and Skretting et al. [9].

One can also learn the new pattern only on patches that do not overlap with another, as done in [10], but this introduces a bias and a loss of robustness since the learning is performed on fewer training examples. That is the reason why we also propose an unbiased update that extends the previous one.

Now the vectors $T_\tau^* \hat{s}_\kappa$ cannot be used as patches because they are perturbed by the other overlapping occurrences. We propose to compute the patch family $P_\kappa = (p_{\kappa,\tau})_{\tau \in \sigma_\kappa}$ that minimizes the deviation of the current pattern direction:

$$P_\kappa^{opt} = \text{argmin}_{(p_\tau)} \sum_{\tau \in \sigma_\kappa} \|p_\tau - c_{\kappa,\tau} m_\kappa\|_2^2 \tag{5}$$

under the constraint that the patches rebuild the signal as perfectly as possible:

$$\forall \tau \in \sigma_\kappa, T_\tau^* \left(\sum_{\tau' \in \sigma_\kappa} T_{\tau'} p_{\tau'} \right) = T_\tau^* \hat{s}_\kappa$$

If the patches do not overlap, then the constraint can be reduced to $\forall \tau \in \sigma_\kappa, p_{\tau'} = T_\tau^* \hat{s}_\kappa$ and we get the previous case back. Otherwise it can still be computed easily: the solution is given by dividing the residual equally among the patches. If we define $w_{\kappa,\tau}$ as the number of patches that overlap on the τ sample (and ∞ if there is no patch, so that it can always be inverted) , then the solution can be written as:

$$\forall \tau \in \sigma_\kappa, p_{\kappa,\tau}^{opt} = c_{\kappa,\tau} m_\kappa + T_\tau^* \left(w_\kappa^{-1} \times r \right)$$

where $\times$ is the pointwise product of two signals and the powers of w_κ are also computed point by point. With these notations and l the length of the pattern we can compute the cost function (5):

$$\sum_{\tau \in \sigma_\kappa} \|p_\tau - c_{\kappa,\tau} m_\kappa\|_2^2 = \sum_{\tau \in \sigma_\kappa} \left\| T_\tau^* \left(w_\kappa^{-1} \times r \right) \right\|_2^2$$
$$= \sum_{\tau \in \sigma_\kappa} \sum_{\tau' \in [\tau, \tau+l-1]} \frac{r_{\tau'}^2}{w_{\kappa,\tau'}^2} = \sum_{\tau' / w_{\kappa,\tau'} < \infty} w_{\kappa,\tau'} \frac{r_{\tau'}^2}{w_{\kappa,\tau'}^2}$$
$$= \sum_{\tau' / w_{\kappa,\tau'} < \infty} \frac{r_{\tau'}^2}{w_{\kappa,\tau'}} = \left\| w_\kappa^{-\frac{1}{2}} \times r \right\|_2^2 + cst$$

where the constant only depends on the parts of the signal with no occurrence of m_κ. So at each step the new pattern is the vector of canonical norm 1 that minimizes the elliptic euclidean norm of the residual $\|w_\kappa^{-\frac{1}{2}} \times r\|_2^2$. The less overlap there is at a sample τ (to a minimum of 1), the larger $w_{\kappa,\tau}^{-\frac{1}{2}}$. So, compared to the canonical norm, this elliptic norm penalizes more the error committed on parts of the signal with few overlap, thus giving them more weight in the computation of the pattern. It makes sense to trust more the observed parts that are supposed to be less perturbed.

Then we take the new pattern as the principal component of the patches, as in Equation (3). Finally the coefficients that reduce the patchwise error (5) the most are given by

$$\forall \tau \in \sigma_\kappa, c_{\kappa,\tau} \leftarrow \langle p_{\kappa,\tau}, m_\kappa \rangle$$

If the patches do not overlap, then these coefficients are the same as the ones given by the orthogonal projection update (4).

5. EXPERIMENTAL RESULTS

5.1 Exact recovery on synthetic data

This experiment is aimed at checking that the update method we propose is unbiased, *i.e.* that it can recover the original dictionary if we know the right decomposition support $(\sigma_k)_k$. We also measured the influence of a bad estimation of the amplitude coefficients $c_{k,\tau}$ on the recovered dictionary. To do that we used a synthetic toy example.

The dictionary contains only one random Gaussian pattern m_0 of length 1024 and the signal is generated with only 2 overlapping occurrences of that pattern ($\sigma_0 = \{0, 256\}$) with Gaussian amplitudes $(c_{0,0}, c_{0,256})$. Then we iterated all the different update methods (without alternating with decomposition) with another random dictionary and the exact decomposition support σ_0 as an input. The input amplitude coefficients were $(c_{0,0} + \varepsilon_{0,0}, c_{0,256} + \varepsilon_{0,256})$ where ε is an additive Gaussian noise. For the Bayesian, we used an adaptation rate decreasing as $\frac{1}{i}$, with i the iteration number, to ensure convergence.

Figure 1 shows the dictionary SNR defined as $-20 \log_{10} \|m - m_0\|$ where m is the learnt pattern and m_0 the original one, depending on the number of successive updates. It shows that the principal component method with overlap handling converges to the exact pattern with exponential speed. The SNR threshold at 300dB corresponds to machine precision. Without overlap handling, the principal component method converges very fast to a sub-optimal solution.

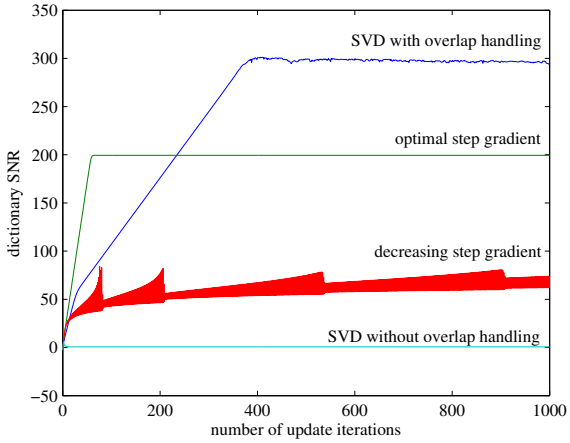

Figure 1: Dictionary SNR reached by the different update methods on a toy signal with only 2 occurrences of the same pattern and a 200dB noise on the initial amplitude coefficients.

The optimal rate weighted mean update converges to a sub-optimal solution with SNR roughly equal to the input coefficient SNR equal to $-20\log_{10}\frac{\|\varepsilon\|}{\|c\|}$. This was observed for several different coefficient SNR between 0 and 250dB. If the input coefficients are the exact ones, then this method converges faster than the principal component one to the optimal solution. Bayesian update oscillates because of a too big adaptation rate and converges much more slowly to the same limit as weighted mean update (more than 1 million iterations, not shown on the plot). The final result of principal component methods is not affected by the initial coefficients, but convergence speed might slightly change.

These results were confirmed on more complex synthetic data with several patterns in the dictionary and several random occurrences of each pattern in the signal (plot not shown).

5.2 Learning on a music track

We also ran these algorithms on real data to check that found atoms are morphologically sound, to compare their performances for sparse approximation and to observe the effect of a bad estimation of the decomposition length L. These experiments were performed on an excerpt from the RWC base [2]. It is a 1 minute long jazz guitar piece down-sampled to 8000Hz, so 480000 sample long. All the learnt dictionaries contain 40 patterns of 1024 samples.

For all the learnt dictionaries, the learnt patterns were mostly sinusoidal, sometimes with modulated amplitude. Some atoms had a strong harmonic structure and some looked more noisy. Figure 2 shows some examples.

5.2.1 The different update methods

For the first experiment, we focused on the influence of the dictionary update method chosen. All the dictionaries were learnt on 10000 atom decompositions with different update rules:

- the Bayesian rule used in [3]
- the weighted mean of the patches
- the principal component of the patches
- the principal component of the patches modified to handle overlap

Performing 100 learning iterations took about 3h 40min for the two mean-based methods and about 4h 20min for the two SVD-based methods. The difference remains quite low because most of the time is spent in the decomposition step. At each iteration, one step of mean-based update costs about 20s, one SVD-based update about 45s and one MP decomposition about 1min 50s. However, the SVD cost might explode when dealing with very large patterns: to our knowledge, the most efficient SVD algorithms have quadratic complexity.

[2] http://staff.aist.go.jp/m.goto/RWC-MDB/

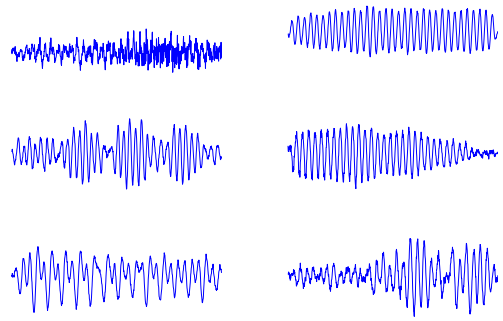

Figure 2: Some learnt patterns

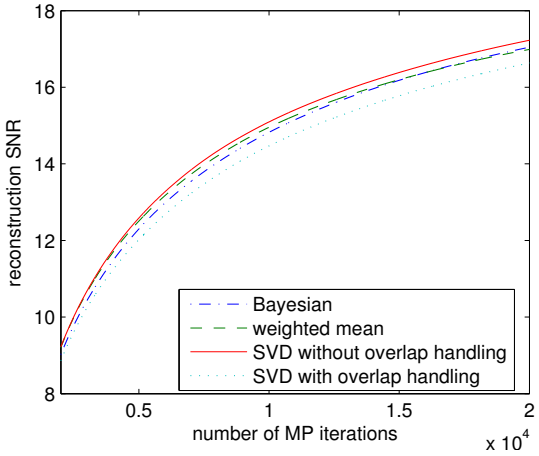

Figure 4: reconstruction SNR of a music signal with dictionaries learnt with different update methods

Figure 4 shows the reconstruction SNR defined as $-20\log_{10}\frac{\|r\|}{\|s\|}$ as a function of the l_0 norm of the coefficients. It represents the trade-off between the quality of an approximation and its sparsity. All the update methods give quite close results, with a maximal difference of 1.3dB. SVD without residual split performs better than the weighted mean, and weighted mean performs better than the Bayesian update with a fixed adaptation rate.

The overlap handling causes a loss of performance that could be explained by the minimized error function. As seen in section 4.2, when correcting the overlap we minimize the weighted euclidean norm of the residual $\|w_\kappa^{-\frac{1}{2}} \times r\|_2^2$. The fact that we don't minimize the cost function used for the performance measurements could explain the loss. This might be corrected in future works by looking for another way to split the residual that leads to minimizing the canonical norm.

5.2.2 Influence of the decomposition length

We also measured the effect of bad estimation of the number of MP iterations I during the learning. In that goal we learnt several dictionaries with several choices for I between 100 and 10 000 atoms,

number of MP iterations used during learning	100	200	500	1000	2000	5000	10000
reconstruction SNR with 2000 atoms (dB)	7.5	8.0	8.6	9.4	9.9	9.5	9.2

Figure 3: reconstruction SNR for dictionaries learnt on several decomposition lengths.

then we measured the reconstruction SNR $-20\log_{10}\frac{\|r\|}{\|s\|}$ obtained by each dictionary for a decomposition of fixed length. Figure 3 shows the reconstruction SNR for a decomposition of 2 000 MP iterations on dictionaries learnt with an I parameter of 100, 200, 500, 1 000, 2 000, 5 000 or 10 000. We can see that the best dictionary is the one learnt with the same decomposition length that is used for measure. However, the SNR decreases slowly for dictionaries learnt on bigger sizes. This leads us to believe that the algorithm is quite robust to the overestimation of the decomposition length.

These results were also observed by comparing the SNR at all the values of I used for learning one of the dictionaries (data not shown).

6. CONCLUSION

We have presented a way to extend the *K-SVD* algorithm to learn shift invariant dictionaries for sparse representation. The update method we propose is unbiased and does not rely on the exact value of the decomposition coefficients. We have observed on synthetic data that this method is able to retrieve the dictionary used for synthesis if provided the correct support of the decomposition. We have also checked on real music data that the algorithm was able to find sinusoidal and harmonic patterns modulated patterns.

Although not demonstrated here, the same algorithm can be used to learn patterns of several sizes or on multidimensional signals. The invariance to any group of unitary operators could also be easily formally derived, but practical tractability would require an efficient implementation of the corresponding decomposition.

This algorithm could be much improved by being able to learn parameters such as the decomposition length, the number of patterns or their length.

REFERENCES

[1] S.A. Abdallah and M.D. Plumbley, "If edges are the independent components of natural images, what are the independent components of natural sounds?," in *Proc. of the Int'l Workshop on ICA and Blind Signal Separation (ICA 2001)*, December 2001, pp. 534–539.

[2] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: design of dictionaries for sparse representation.," in *In* Proc. SPARS'05*, IRISA, Rennes, France*, November 2005, http://spars05.irisa.fr.

[3] T. Blumensath and M. Davies, "Sparse and shift-invariant representations of music," *Audio, Speech, and Language Processing, IEEE Transactions on [see also Speech and Audio Processing, IEEE Transactions on]*, vol. 14, no. 1, pp. 50–57, Jan. 2006.

[4] J.A. Tropp, "Greed is good: algorithmic results for sparse approximation," *IEEE Transactions on Information theory*, vol. 50, pp. 2231–2242, October 2004.

[5] S. Krstulovic and R. Gribonval, "MPTK: Matching pursuit made tractable," *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 3, pp. III–III, 14-19 May 2006.

[6] R. Gribonval and P. Vandergheynst, "On the exponential convergence of matching pursuits in quasi-incoherent dictionaries," *Information Theory, IEEE Transactions on*, vol. 52, no. 1, pp. 255–261, Jan. 2006.

[7] K. Engan, S.O. Aase, and J. Hakon Husoy, "Method of optimal directions for frame design," *Acoustics, Speech, and Signal Processing, 1999. ICASSP '99. Proceedings., 1999 IEEE International Conference on*, vol. 5, pp. 2443–2446 vol.5, 1999.

[8] Michal Aharon, *Sparseness and Over-completeness in Signal Representation*, Ph.D. thesis, Technion, Haïfa, Israel, 2006.

[9] Karl Skretting, John Håkon Husøy, and Sven Ole Aase, "General design algorithm for sparse frame expansions," *Signal Process.*, vol. 86, no. 1, pp. 117–126, 2006.

[10] Sylvain Lesage, *Apprentissage de dictionnaires structurés pour la modélisation parcimonieuse de signaux multicanaux*, Ph.D. thesis, Université de Rennes 1, 2007.