PAIRWISE APPROXIMATE K-SVD

Paul Irofti, Bogdan Dumitrescu

The Research Institute of the University of Bucharest (ICUB)
and Department of Computer Science
University of Bucharest, Romania

Department of Automatic Control and Computers
University Politehnica of Bucharest
313 Spl. Independenţei, 060042 Bucharest, Romania

ABSTRACT

Pairwise, or separable, dictionaries are suited for the sparse representation of 2D signals in their original form, without vectorization. They are equivalent with enforcing a Kronecker structure on a standard dictionary for 1D signals. We present a dictionary learning algorithm, in the coordinate descent style of Approximate K-SVD, for such dictionaries. The algorithm has the benefit of extremely low complexity, clearly lower than that of existing algorithms. Experimental evidence shows that the performance of the proposed algorithm is comparable to that of standard (unstructured) AK-SVD with the same number of atoms.

Index Terms— sparse representations, dictionary learning, separable dictionary, coordinate descent

1. INTRODUCTION

Sparse representations [1, 2] have numerous signal processing applications. They are especially effective when combined with dictionary learning (DL) [3]. A significant variation on this theme is that of structured dictionaries, where the representation matrix is endowed with a structure that, ideally, is both suited to the characteristics of the modeled signals and easier to compute than in the standard case.

We deal with separable (or pairwise) dictionaries, targeted to the representation of 2D signals. Given training signals $\boldsymbol{Y}_k \in \mathbb{R}^{m\times m}$, $k = 1 : N$, and target sparsity $s \in \mathbb{N}$, the pairwise DL problem is

$$\begin{aligned} \min_{\boldsymbol{D}_1,\boldsymbol{D}_2,\boldsymbol{X}} \quad & \sum_{k=1}^{N} \|\boldsymbol{Y}_k - \boldsymbol{D}_1\boldsymbol{X}_k\boldsymbol{D}_2^T\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{X}_k\|_0 \le s,\ k = 1 : N \\ & \|\boldsymbol{d}_{1i}\|_2 = 1,\ i = 1 : n_1 \\ & \|\boldsymbol{d}_{2j}\|_2 = 1,\ j = 1 : n_2, \end{aligned} \tag{1}$$

where $\boldsymbol{X}_k \in \mathbb{R}^{n_1\times n_2}$ are the sparse representations (the matrix $\boldsymbol{X}_k$ has at most s nonzero elements), $\boldsymbol{D}_1 \in \mathbb{R}^{m\times n_1}$ is the left dictionary and $\boldsymbol{D}_2 \in \mathbb{R}^{m\times n_2}$ is the right dictionary. By $\boldsymbol{d}_{1i}$ and $\boldsymbol{d}_{2j}$ we denote the columns of the two dictionaries, also named atoms, which are normalized.

Let vec$\boldsymbol{M}$ denote the column-order vectorization of matrix $\boldsymbol{M}$. We note that

$$\|\boldsymbol{Y}_k - \boldsymbol{D}_1\boldsymbol{X}_k\boldsymbol{D}_2^T\|_F = \|\text{vec}\boldsymbol{Y}_k - (\boldsymbol{D}_2\otimes\boldsymbol{D}_1)\text{vec}\boldsymbol{X}_k\|_2, \tag{2}$$

hence problem (1) is in fact the standard (1D) DL problem with dictionary $\boldsymbol{D}_2 \otimes \boldsymbol{D}_1$. The great advantage of the 2D formulation (1) is that it operates with two small matrices, while the large dictionary $\boldsymbol{D}_2 \otimes \boldsymbol{D}_1$ is never explicitly computed. Another advantage is the ability to work directly with the 2D signals, thus being more able to model the spatial structures.

Our contribution is a simple algorithm for (1) based on coordinate descent, in the style of Approximate K-SVD (AK-SVD) [4]. Its complexity is lower than that of the existing techniques listed in section 2.1. After reviewing sparse representation with 2D-OMP in section 2.2, the remainder of section 2 is dedicated to the presentation of our algorithm. Section 3 shows a behavior very close to that of the standard 1D AK-SVD, with much smaller resources in terms of dictionary size and computational complexity.

2. PAIRWISE DICTIONARY LEARNING

2.1. Existing Techniques

The usual approach is to alternate sparse coding and dictionary update steps. In the former, the representation matrices $\boldsymbol{X}_k$ are computed for fixed dictionaries; in the latter, the dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ and possibly the nonzero elements of $\boldsymbol{X}_k$ are updated with a fixed structure of zeros in $\boldsymbol{X}_k$.

Separable DL was introduced in [5], together with a highly complex algorithm based on manifold gradient search. In [6], the problem is tackled via alternating optimization; while we update atoms individually, there pairs of atoms are optimized together, which leads to a more complicated rank-1 CANDECOMP/PARAFAC (CP) decomposition. Tensor techniques are also used in DL context in [7] and [8], the latter proposing the TKSVD algorithm that will be analyzed below. Other Kronecker-like dictionary structures can be found in [6, 9].

Paul Irofti was supported by a grant of the Romanian Ministry of Research and Innovation, CCCDI-UEFISCDI, project number 17PCCDI/2018 within PNCDI III. E-mail: paul@irofti.net, bogdan.dumitrescu@acse.pub.ro.

2.2. Representation

Given patch $\boldsymbol{Y}$, left dictionary $\boldsymbol{D}_1$ and right dictionary $\boldsymbol{D}_2$, the error function is

$$\mathcal{E}(\boldsymbol{X}) = \boldsymbol{Y} - \boldsymbol{D}_1 \boldsymbol{X} \boldsymbol{D}_2^T, \tag{3}$$

where $\boldsymbol{X}$ is the s-sparse representation. Finding the best $\boldsymbol{X}$ given the sparsity level s requires solving the optimization problem

$$\boldsymbol{X} = \arg\min_{\boldsymbol{X}} \quad \|\mathcal{E}(\boldsymbol{X})\|_F \quad \text{subject to} \quad \|\boldsymbol{X}\|_0 \leq s. \tag{4}$$

We focus on simple methods having a greedy character. Let us assume that, at step ℓ of its execution, a method selects the left and right atoms whose indices are stored in the ordered sets $\mathcal{I} = \{i_1, \ldots, i_\ell\}$ and $\mathcal{J} = \{j_1, \ldots, j_\ell\}$, respectively; note that some atoms may be selected several times, hence the indices from a set may be repeated; however, the pairs $(i_1, j_1), \ldots, (i_\ell, j_\ell)$ are distinct.

Orthogonal matching pursuit (OMP) [10] can be easily adapted to a separable dictionary. Denoting $\tilde{\boldsymbol{X}}_\ell$ the representation at step ℓ, the corresponding residual is

$$\tilde{\boldsymbol{R}}_\ell = \boldsymbol{Y} - \boldsymbol{D}_1 \tilde{\boldsymbol{X}}_\ell \boldsymbol{D}_2^T. \tag{5}$$

The standard OMP projects the residual on the 1D separable dictionary, thus computing

$$(\boldsymbol{D}_2 \otimes \boldsymbol{D}_1)^T \text{vec} \tilde{\boldsymbol{R}}_\ell = \text{vec}(\boldsymbol{D}_1^T \tilde{\boldsymbol{R}}_\ell \boldsymbol{D}_2). \tag{6}$$

So, 2D OMP simply computes the matrix $\boldsymbol{P}_\ell = \boldsymbol{D}_1^T \tilde{\boldsymbol{R}}_\ell \boldsymbol{D}_2$. The indices $(i_{\ell+1}, j_{\ell+1})$ of the maximum absolute value in $\boldsymbol{P}_\ell$ give the next selected atoms from $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$.

Given $\mathcal{I}$ and $\mathcal{J}$, the least-squares solution $\tilde{\boldsymbol{X}}_\ell$ is computed by noticing that

$$\begin{aligned} \|\mathcal{E}(\tilde{\boldsymbol{X}}_\ell)\|_F &= \|\boldsymbol{Y} - \sum_{\iota=1}^{\ell} \tilde{x}_{i_\iota, j_\iota} \boldsymbol{d}_{1 i_\iota} \boldsymbol{d}_{2 j_\iota}^T\|_F \\ &= \|\text{vec}\boldsymbol{Y} - \sum_{\iota=1}^{\ell} (\boldsymbol{d}_{2 j_\iota} \otimes \boldsymbol{d}_{1 i_\iota}) \tilde{x}_{i_\iota, j_\iota}\|. \end{aligned} \tag{7}$$

Let $\boldsymbol{A}$ be the matrix whose columns are the atom Kronecker products from above and let $\boldsymbol{x}$ be the coefficient vector formed by the nonzero elements of $\tilde{\boldsymbol{X}}_\ell$; then, the error (3) becomes $\|\mathcal{E}(\tilde{\boldsymbol{X}}_\ell)\|_F = \|\text{vec}\boldsymbol{Y} - \boldsymbol{A}\boldsymbol{x}\|_2$ and can be treated as a plain least-squares problem for finding the nonzero coefficients $\boldsymbol{x}$, like in standard OMP. A different and more thorough demonstration of the equivalence between pairwise and standard OMP is described in [11].

2.3. TKSVD Dictionary Update

Dictionary update is performed by TKSVD [8] by simultaneously updating each (i, j) dictionary atoms pair while keeping fixed the other atoms and the representations. Given i and j, let

$$\boldsymbol{R}_k = \mathcal{E}(\boldsymbol{X}_k) + x_{i,j}^{(k)} \boldsymbol{d}_{1i} \boldsymbol{d}_{2j}^T \tag{8}$$

denote the residual of the k-th sample without the contribution of the current atoms. With these atoms as variables, the optimization problem becomes

$$\begin{aligned} \min_{\boldsymbol{d}_{1i}, \boldsymbol{d}_{2j}} \quad & \sum_{k=1}^{N} \left\| \boldsymbol{R}_k - x_{i,j}^{(k)} \boldsymbol{d}_{1i} \boldsymbol{d}_{2j}^T \right\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{d}_{1i}\|_2 = 1, \|\boldsymbol{d}_{2j}\|_2 = 1. \end{aligned} \tag{9}$$

Typically, many representation coefficients $x_{i,j}^{(k)}$ are zero; the problem is solved via a rank-1 CP decomposition. This process is iterated by TKSVD for all atom pairs used by the sparse representations.

Now let us assume there is another pair in which the current atom $\boldsymbol{d}_{1i}$ from equation (9) participates: two atoms $\boldsymbol{d}_{2j}$ and $\boldsymbol{d}_{2j'}$ from $\boldsymbol{D}_2$ both pair with $\boldsymbol{d}_{1i}$. Note that the sets $\{k \mid x_{i,j}^{(k)} \neq 0\}$ and $\{k \mid x_{i,j'}^{(k)} \neq 0\}$ are usually different, possibly even disjoint. Hence, when the dictionary refinement process updates the (i, j') pair there is a good chance that not only the objective of (9) for the original (i, j) pair increases, because $\boldsymbol{d}_{1i}$ is not only no longer optimal for it, but also that the overall error (3) increases. This may ultimately lead to a lack of convergence and, in severe cases, even to divergence.

In fact we were able to see this exact behaviour in our numerical experiments on synthetic data, which is why we attacked the problem from a different angle.

2.4. AK-SVD Approach

Inspired by the AK-SVD [4] algorithm for the standard DL problem, we keep everything fixed in equation (1) except for either atom i of $\boldsymbol{D}_1$

$$\min_{\boldsymbol{d}_{1i}} \quad \sum_{k=1}^{N} \left\| \boldsymbol{R}_{1k} - \boldsymbol{d}_{1i} \boldsymbol{x}_{i,\mathcal{P}_k}^{(k)} \boldsymbol{D}_{2\mathcal{P}_k}^T \right\|_F^2 \quad \text{s.t.} \quad \|\boldsymbol{d}_{1i}\|_2 = 1 \tag{10}$$

or atom j of $\boldsymbol{D}_2$

$$\min_{\boldsymbol{d}_{2j}} \quad \sum_{k=1}^{N} \left\| \boldsymbol{R}_{2k} - \boldsymbol{D}_{1\mathcal{Q}_k} \boldsymbol{x}_{\mathcal{Q}_k,j}^{(k)} \boldsymbol{d}_{2j}^T \right\|_F^2 \quad \text{s.t.} \quad \|\boldsymbol{d}_{2j}\|_2 = 1$$

where $\mathcal{P}_k$ is the set of atoms in $\boldsymbol{D}_2$ that pair with atom i in $\boldsymbol{D}_1$ and $\mathcal{Q}_k$ is the set of atoms from $\boldsymbol{D}_1$ pairing with $\boldsymbol{d}_{2j}$, when representing $\boldsymbol{Y}_k$. The residuals $\boldsymbol{R}_{1k}$ and $\boldsymbol{R}_{2k}$ are obtained via obvious modifications of (8). The k-th term of the sum from (10) becomes

$$\begin{aligned} \|\boldsymbol{R}_{1k}\|_F^2 - 2\text{tr}(\boldsymbol{R}_{1k}^T \boldsymbol{d}_{1i} \boldsymbol{x}_{i,\mathcal{P}_k}^{(k)} \boldsymbol{D}_{2\mathcal{P}_k}^T) + \\ + \boldsymbol{x}_{i,\mathcal{P}_k}^{(k)} \boldsymbol{D}_{2\mathcal{P}_k}^T \boldsymbol{D}_{2\mathcal{P}_k} (\boldsymbol{x}_{i,\mathcal{P}_k}^{(k)})^T, \end{aligned} \tag{11}$$

Require: initial dictionaries $\boldsymbol{D}_1 \in \mathbb{R}^{m\times n_1}$, $\boldsymbol{D}_2 \in \mathbb{R}^{m\times n_2}$
signals set $\boldsymbol{Y}_k \in \mathbb{R}^{m\times m}$, $k = 1 : N$
number of iterations $T \in \mathbb{N}$
Ensure: learned dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$
1: **for** $t = 1$ **to** T **do**
2: Sparse coding: keeping $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ fixed, compute sparse representations $\boldsymbol{X}$ using 2D OMP
Dictionary update:
3: **for** $i = 1$ **to** n_1 **do**
4: Compute the new atom $\boldsymbol{d}_{1i}$ using (13)
5: Update representations $\boldsymbol{x}^{(k)}_{i,\mathcal{P}_k}$ using (14), $k = 1 : N$
6: **end for**
7: **for** $j = 1$ **to** n_2 **do**
8: Compute the new atom $\boldsymbol{d}_{2j}$ using (15)
9: Update representations $\boldsymbol{x}^{(k)}_{\mathcal{Q}_k,j}$ using (16), $k = 1 : N$
10: **end for**
11: **end for**

Algorithm 1: Pairwise Approximate K-SVD algorithm

which is minimized when the trace is maximized. We rewrite the optimization problem (10) as

$$\max_{\boldsymbol{d}_{1i}} \quad \left(\sum_{k=1}^{N} \boldsymbol{x}^{(k)}_{i,\mathcal{P}_k} \boldsymbol{D}^T_{2\mathcal{P}_k} \boldsymbol{R}^T_{1k}\right) \boldsymbol{d}_{1i} \quad \text{s.t.} \quad \|\boldsymbol{d}_{1i}\|_2 = 1 \tag{12}$$

whose maximum value is achieved when $\boldsymbol{d}_{1i}$ is collinear to the sum vector, thus giving

$$\boldsymbol{d}_{1i} = \frac{\boldsymbol{t}_1}{\|\boldsymbol{t}_1\|}, \quad \boldsymbol{t}_1 = \sum_{k=1}^{N} \boldsymbol{R}_{1k} \boldsymbol{D}_{2\mathcal{P}_k} (\boldsymbol{x}^{(k)}_{i,\mathcal{P}_k})^T. \tag{13}$$

Mimicking AK-SVD, we now update the signals that use $\boldsymbol{d}_{1i}$ in their representation. The minimization of the quadratic form resulting from (11) gives the new optimal coefficient

$$(\boldsymbol{x}^{(k)}_{i,\mathcal{P}_k})^T = (\boldsymbol{D}^T_{2\mathcal{P}_k} \boldsymbol{D}_{2\mathcal{P}_k})^\dagger \boldsymbol{D}^T_{2\mathcal{P}_k} \boldsymbol{R}^T_{1k} \boldsymbol{d}_{1i}. \tag{14}$$

The same reasoning is applied for atom j in $\boldsymbol{D}_2$. Atom $\boldsymbol{d}_{2j}$ is updated following

$$\boldsymbol{d}_{2j} = \frac{\boldsymbol{t}_2}{\|\boldsymbol{t}_2\|}, \quad \boldsymbol{t}_2 = \sum_{k=1}^{N} \boldsymbol{R}^T_{2k} \boldsymbol{D}_{1\mathcal{Q}_k} \boldsymbol{x}^{(k)}_{\mathcal{Q}_k,j}. \tag{15}$$

Then we update the representations via

$$\boldsymbol{x}^{(k)}_{\mathcal{Q}_k,j} = (\boldsymbol{D}^T_{1\mathcal{Q}_k} \boldsymbol{D}_{1\mathcal{Q}_k})^\dagger \boldsymbol{D}^T_{1\mathcal{Q}_k} \boldsymbol{R}_{2k} \boldsymbol{d}_{2j}. \tag{16}$$

Note that these dictionary update operations are guaranteed to decrease the objective of (1).

2.5. Algorithm

We describe the proposed DL process in Algorithm 1. Dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ are obtained by performing T rounds of sparse coding and dictionary update (step 1). In the sparse coding stage we keep the dictionaries fixed and update the representations, while during dictionary update we fix the representations and update one dictionary at a time. Our experiments showed that a good choice for T is usually somewhere below 100 iterations.

Step 2 starts the iteration by computing the sparse representations. Given current dictionaries $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ we use 2D OMP, described in Section 2.2, to obtain representations $\boldsymbol{X}_k$ with $k = 1 : N$. This produces results identical to standard OMP with reduced computational overhead.

In the dictionary update stage the representations are first fixed. We proceed by also fixing dictionary $\boldsymbol{D}_2$ and sequentially updating each of the atoms in $\boldsymbol{D}_1$ (step 3). Step 4 updates an atom, using relation (13). Following AK-SVD style, step 5 uses equation (14) to update the nonzero coefficients $\boldsymbol{x}^{(k)}_{i,\mathcal{P}_k}$ of the representations using atom $\boldsymbol{d}_{1i}$; of course, only signals $\boldsymbol{Y}_k$ using that atom are involved (meaning that $\mathcal{P}_k \neq \emptyset$).

Once all atoms of dictionary $\boldsymbol{D}_1$ are updated, the process is repeated for dictionary $\boldsymbol{D}_2$: we fix $\boldsymbol{D}_1$ and sequentially update each atom $\boldsymbol{d}_{2j}$ (step 7) and compute the new atom (step 8) and its associated representations (step 9). Note that we chose to perform dictionary update in order for clarity, but other strategies can be employed such as updating the atoms from both dictionaries in random order.

Let us analyze the number of operations necessary to perform a single iteration from step 1. For the sparse coding operation in step 2, it has been shown in [11] that 2D OMP has a complexity of $\mathcal{O}(mn_1n_2)$, which is $1/m$ that of standard OMP. The resulting representations $\boldsymbol{X}$ contain Ns nonzeros corresponding to just as many atom pairs from $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ respectively. Let us assume, without any loss of generality, that all atoms are used an equal number of times by the representations. Thus coefficients corresponding to atom $\boldsymbol{d}_{1i}$ appear Ns/n_1 times in $\boldsymbol{X}$. If we split equation (13) from step 4 based on the elements from $\boldsymbol{X}$, we have to compute Ns/n_1 products $\boldsymbol{R}_{1k}\boldsymbol{d}_{2j}x^{(k)}_{i,j}$, where i, j and k are fixed. This totals to sm^2N operations for updating $\boldsymbol{D}_1$. Updating the representations in step 5 requires m^2 operations for multiplying $\boldsymbol{R}^T_{1k}\boldsymbol{d}_{1i}$, hence $O(sm^2N)$ over one iteration, in the least favorable case where an atom from $\boldsymbol{D}_1$ pairs with a single atom from $\boldsymbol{D}_2$ for a signal $\boldsymbol{Y}_k$. The ensuing least squares solution needs $O(ms^2)$ operations if an atom for $\boldsymbol{D}_1$ is paired with s atoms from $\boldsymbol{D}_2$ for signal $\boldsymbol{Y}_k$, which is the maximum; otherwise the number of operations is smaller. So, again, the complexity over an iteration is $O(sm^2N)$. The same reasoning can be applied to steps 7–10. We note that AK-SVD has the same $O(sm^2N)$ complexity for the dictionary update (but is slower in sparse coding). Our algorithm has thus lower complexity than the algorithms from [6, 7] that use the CP decomposition as basic tools (in a 1D setup, our algorithm would relate to theirs as AK-SVD to K-SVD) and is considerably faster than SeDiL [5].

Table 1. Denoising PSNR(dB) and SSIM for standard images

σ_{noise} / PSNR	Method	lena		barbara		boat		peppers		house	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
5 / 34.15	1D	38.46	0.942	37.94	0.962	36.98	0.935	37.60	0.925	39.27	0.953
	2D	38.22	0.940	37.69	0.961	36.63	0.929	37.34	0.920	38.71	0.948
10 / 28.13	1D	35.30	0.907	34.21	0.931	33.47	0.878	34.63	0.876	35.62	0.901
	2D	35.21	0.905	34.10	0.930	33.42	0.875	34.56	0.873	35.43	0.897
20 / 22.11	1D	32.02	0.857	30.21	0.867	29.89	0.789	31.88	0.833	32.47	0.857
	2D	31.94	0.855	30.08	0.863	29.88	0.787	31.88	0.831	32.25	0.854
30 / 18.58	1D	29.96	0.817	27.75	0.803	27.90	0.727	30.12	0.804	30.44	0.828
	2D	29.91	0.815	27.65	0.799	27.91	0.725	30.10	0.802	30.18	0.824
50 / 14.15	1D	27.37	0.754	24.66	0.685	25.47	0.642	27.66	0.755	27.47	0.762
	2D	27.32	0.752	24.57	0.681	25.41	0.638	27.72	0.754	27.35	0.758

3. RESULTS[1]

Our first experiment shows the application of pairwise AK-SVD for image denoising. To this end we use images from the USC-SIPI database [12] to which we apply white gaussian noise of varied standard deviation levels σ_{noise}.

The denoising process starts by splitting the noisy image into overlapping patches of size $m = 8$ and randomly picking a set $N = 4000$ such patches for training a pair of dictionaries of $n_1 = n_2 = 16$ atoms. In our experiment we use just a few DL iterations, $T = 20$, with target sparsity $s = 6$. We repeat each experiment, using the same parametrization and training data, to obtain an AK-SVD dictionary of $n = n_1 n_2$ atoms. With the resulting dictionaries we perform sparse representation on the entire set of overlapping patches and reconstruct the image via pixel averaging.

Table 1 presents the results. We picked the images and noise levels that are often used for both standard [13] and separable [5] DL methods. Denoising quality is measured via the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) [14] between the original and denoised images. We note that the results for pairwise AK-SVD (denoted 2D in the table) and standard AK-SVD (denoted 1D) are similar if not identical. Although they are lower than those given by BM3D [15], they are comparable with those from other pairwise DL papers, like [6]; for example, the PSNRs reported there for barbara (better than those from [5]) are: 37.60, 34.03, 30.27, 27.94, 23.64.

Even though the number of atoms in the AK-SVD dictionary is the same as the total number of pairs in the 2D-case, the standard method has the advantage of specializing each atom individually, whereas in the separable case that is not possible as one atom from the left dictionary can pair with more than one atom in the right dictionary and vice-versa. Though masked in the denoising case, this effect is clearly visible if we focus only on learning performance. As depicted in Figure 1, our experiments showed that, when varying n_1 and n_2, for the same input data and parametrization

[1]Code available at `https://github.com/pirofti/pair_ksvd`

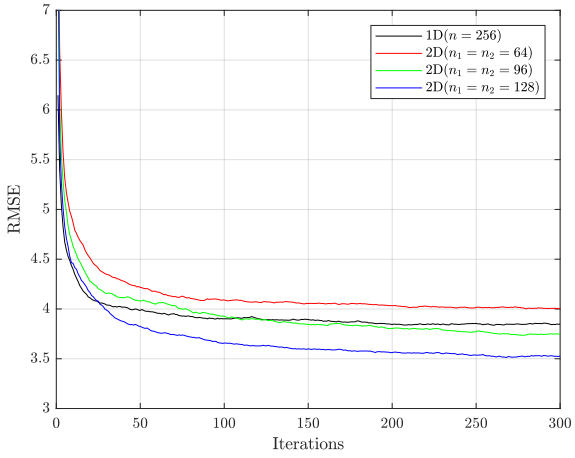

Fig. 1. RMSE evolution over 300 iterations for varied dictionary sizes.

as in the denoising case, learning through standard AK-SVD is outperformed by pairwise AK-SVD somewhere between $n_1 = n_2 = 64$ and $n_1 = n_2 = 96$. Note that we can afford the time-debt incurred by using larger dictionaries due to the improved complexity of pairwise AK-SVD. Here we measure learning performance in terms of RMSE between the training signals and the representations $\sqrt{\sum_{k=1}^{N} \|\mathcal{E}(\boldsymbol{X}_k)\|_F^2}/\sqrt{m^2 N}$. See (2) for adapting RMSE for standard AK-SVD.

4. CONCLUSIONS

We have proposed a coordinate descent algorithm for solving the pairwise dictionary learning problem (1). The dictionary update stage is based on coordinate descent: the atoms of both dictionaries and the corresponding representations are successively optimized while all other variables are fixed. This AK-SVD style algorithm has low complexity and gives results that are comparable with those of more complex algorithms.

REFERENCES

[1] A.M. Bruckstein, D.L. Donoho, and M. Elad, "From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images," *SIAM Rev.*, vol. 51, no. 1, pp. 34–81, 2009.

[2] Z. Zhang, Y. Xu, J. Yang, X. Li, and D. Zhang, "A Survey of Sparse Representation: Algorithms and Applications," *IEEE Access*, vol. 3, pp. 490–530, 2015.

[3] B. Dumitrescu and P. Irofti, *Dictionary Learning Algorithms and Applications*, Springer, 2018.

[4] R. Rubinstein, M. Zibulevsky, and M. Elad, "Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit," *Technical Report - CS Technion*, 2008.

[5] S. Hawe, M. Seibert, and M. Kleinsteuber, "Separable dictionary learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 438–445.

[6] S.H. Hsieh, C.S. Lu, and S.C. Pei, "2D sparse dictionary learning via tensor decomposition," in *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*. IEEE, 2014, pp. 492–496.

[7] Z. Zhang and S. Aeron, "Denoising and completion of 3d data via multidimensional dictionary learning," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2016, pp. 2371–2377.

[8] Y. Fu, J. Gao, Y. Sun, and X. Hong, "Joint multiple dictionary learning for tensor sparse coding," in *Neural Networks (IJCNN), 2014 International Joint Conference on*. IEEE, 2014, pp. 2957–2964.

[9] C.F. Dantas, M.N. da Costa, and R. da Rocha Lopes, "Learning Dictionaries as a Sum of Kronecker Products," *IEEE Signal Processing Letters*, vol. 24, no. 5, pp. 559–563, 2017.

[10] 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.

[11] Y. Fang, J. Wu, and B. Huang, "2D sparse signal recovery via 2D orthogonal matching pursuit," *Science China Information Sciences*, vol. 55, no. 4, pp. 889–897, 2012.

[12] A.G. Weber, "The USC-SIPI Image Database," 1997.

[13] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Trans. Image Proc.*, vol. 15, no. 12, pp. 3736–3745, 2006.

[14] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Proc.*, vol. 13, no. 4, pp. 600–612, 2004.

[15] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Trans. Image Proc.*, vol. 16, no. 8, pp. 2080–2095, 2007.