Regularized Algorithms for Dictionary Learning

Paul Irofti, Bogdan Dumitrescu*
University Politehnica of Bucharest
Department of Automatic Control and Computers
313 Spl. Independenţei, 060042 Bucharest, Romania
*Corresponding author (E-mail : bogdan.dumitrescu@acse.pub.ro)

***Abstract*—Dictionary learning (DL) for sparse representation is a difficult optimization problem for which several successful algorithms exist, although none can be claimed the best. A common problem is a possible stall in the evolution of the algorithm, due to nearly linearly dependent atoms. The proposed cure was to regularize the error criterion using either the norm of the representations or an atom coherence measure. However, only gradient-based algorithms have been proposed for the regularized problems. We give here regularized versions of Approximate K-SVD and other algorithms related to it and investigate numerically their behavior. The experiments show that the new regularized algorithms are able to reduce the representation error, and thus produce better dictionaries, when the imposed sparsity is not very high.**

***Key words* - sparse representation; dictionary learning; regularization.**

I. Introduction

Dictionary learning (DL) for sparse representations [1], [2] is currently an important topic in signal processing due to the ability of trained dictionaries to perform better than fixed dictionaries like those built from popular transforms.

Given a data set $\boldsymbol{Y} \in \mathbb{R}^{m\times N}$, made of N signals of size m, and a sparsity level s, the DL problem is

$$\begin{aligned} \underset{\boldsymbol{D},\boldsymbol{X}}{\text{minimize}} \quad & \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2 \\ \text{subject to} \quad & \|\boldsymbol{x}_\ell\|_0 \le s,\ \ell = 1:N \\ & \|\boldsymbol{d}_j\|_2 = 1,\ j = 1:n \end{aligned} \tag{1}$$

where the variables are the dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times n}$, whose columns are usually named atoms, and the sparse representations matrix $\boldsymbol{X} \in \mathbb{R}^{n\times N}$, whose columns have at most s nonzero elements. Notation: $\boldsymbol{d}_j$ is the j-th column (atom) of the dictionary $\boldsymbol{D}$, $\boldsymbol{x}_\ell$ is the ℓ-th column of the representation matrix $\boldsymbol{X}$, $\|\cdot\|_0$ is the number of nonzero elements of a vector (the so-called 0-norm, although not actually a norm) and $\|\cdot\|_F$ is the Frobenius norm of a matrix. The second constraint, atom normalization, is meant to remove the multiplicative indetermination between $\boldsymbol{D}$ and $\boldsymbol{X}$.

Besides the inherent difficulties caused by the non-convexity of the objective of (1) and the combinatorial character of the sparsity constraint, a particular obstacle was identified in [3] in the possible (almost) linear dependence of atoms that are used in the representation of the same signal. The proposed cure was to change the objective into

$$f_\mu(\boldsymbol{D},\boldsymbol{X}) = \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2 + \mu\|\boldsymbol{X}\|_F^2, \tag{2}$$

where $\mu > 0$ encourages small values of the representation coefficients and thus decreases the likelihood of atoms with similar directions (which typically cause large coefficients). This is a typical regularization for least-squares problem aiming to make the solution unique in cases of rank deficiency.

Although originally proposed [4] with another purpose, an alternative solution to the same problem is to reduce the total mutual coherence between atoms, thus decreasing the apparition of groups of almost linear dependent atoms. The optimization objective is

$$f_\gamma(\boldsymbol{D},\boldsymbol{X}) = \|\boldsymbol{Y}-\boldsymbol{D}\boldsymbol{X}\|_F^2 + \gamma\|\boldsymbol{D}^T\boldsymbol{D}-\boldsymbol{I}\|_F^2. \tag{3}$$

Until now, the regularized problems were solved with algorithms using gradient descent for the dictionary update. We propose here algorithms derived from Approximate K-SVD (AK-SVD) [5]. We also investigate numerically when regularization is more likely to bring benefits, in the sense of obtaining a better approximation error than when solving directly (1).

II. Basic DL algorithms

Most DL algorithms iterate two basic operations: sparse coding and dictionary update. In sparse coding, the dictionary $\boldsymbol{D}$ is fixed and the representations $\boldsymbol{X}$ are computed, most often with the Orthogonal Matching Pursuit (OMP) algorithm [6]. Keeping now the support of the representations (the nonzero pattern of $\boldsymbol{X}$) fixed, the dictionary is updated such as the representation error is reduced. Some methods update also the representation coefficients.

AK-SVD [5] is a simpler and faster version of the K-SVD algorithm [7], based on the idea of alternate optimization of the atoms and their corresponding representation when the support is fixed. Assume all atoms but $\boldsymbol{d}_j$ are fixed and denote $\mathcal{I}_j$ the indices of the signals that use $\boldsymbol{d}_j$ in their representation. If atom $\boldsymbol{d}_j$ is ignored, then the representation error of these signals is

$$\boldsymbol{F} = \boldsymbol{Y}_{\mathcal{I}_j} - \sum_{i\ne j} \boldsymbol{d}_i \boldsymbol{X}_{i,\mathcal{I}_j}, \tag{4}$$

where the subscript $\mathcal{I}_j$ denotes the restriction of the matrix to the columns with indices in $\mathcal{I}_j$. AK-SVD aims to decrease the representation error

$$\|\boldsymbol{F} - \boldsymbol{d}_j \boldsymbol{X}_{j,\mathcal{I}_j}\|_F^2 \tag{5}$$

by optimizing alternatively the atom $\boldsymbol{d}_j$ and the corresponding representation coefficients $\boldsymbol{X}_{j,\mathcal{I}_j}$ (made of the nonzero elements in row j of $\boldsymbol{X}$). AK-SVD is described in Algorithm

1 and we detail below its most specific operations. To alleviate notation, but with obvious correspondence with (5) we consider the template problem

$$\min_{\|\boldsymbol{d}\|=1,\boldsymbol{x}} \quad \left\{\phi(\boldsymbol{d},\boldsymbol{x}) \triangleq \|\boldsymbol{F}-\boldsymbol{d}\boldsymbol{x}^T\|_F^2\right\} \tag{6}$$

When $\boldsymbol{d}$ is fixed, (6) is a simple least squares problem and the optimal representation is

$$\boldsymbol{x} = \boldsymbol{F}^T\boldsymbol{d}. \tag{7}$$

When $\boldsymbol{x}$ is fixed, the objective of (6) can be written as

$$\begin{aligned}\phi(\boldsymbol{d},\boldsymbol{x}) &= \mathrm{tr}[(\boldsymbol{F}^T-\boldsymbol{x}\boldsymbol{d}^T)(\boldsymbol{F}-\boldsymbol{d}\boldsymbol{x}^T)]\\ &= \mathrm{tr}(\boldsymbol{F}^T\boldsymbol{F}) - 2\mathrm{tr}(\boldsymbol{x}\boldsymbol{d}^T\boldsymbol{F}) + \mathrm{tr}(\boldsymbol{x}\boldsymbol{d}^T\boldsymbol{d}\boldsymbol{x}^T)\\ &= \|\boldsymbol{F}\|_F^2 + \|\boldsymbol{x}\|^2 - 2\boldsymbol{d}^T\boldsymbol{F}\boldsymbol{x}.\end{aligned}$$

To obtain the last equality we have used the unit norm constraint on the atom and the property of the trace operator that $\mathrm{tr}(\boldsymbol{A}\boldsymbol{B}\boldsymbol{C}) = \mathrm{tr}(\boldsymbol{B}\boldsymbol{C}\boldsymbol{A})$. In the last expression, the first two terms are constant. The third is a scalar product that becomes minimum when the unit norm atom $\boldsymbol{d}$ has the direction of the vector that multiplies it, hence the optimal atom is

$$\boldsymbol{d} = \boldsymbol{F}\boldsymbol{x}/\|\boldsymbol{F}\boldsymbol{x}\|. \tag{8}$$

Among the related algorithms are SGK [8], which uses only the atom update $\boldsymbol{d} = \boldsymbol{F}\boldsymbol{x}/\|\boldsymbol{x}\|^2$, which is a least-squares solution with the same direction as (8) (normalization is made after all atoms are updated), and NSGK [9], which builds the matrix $\boldsymbol{F}$ in a different way, using differences of the dictionary with respect to its previous value. Parallel versions can be introduced like in [10] for AK-SVD, by updating all atoms simultaneously.

III. Regularized algorithms

To extend AK-SVD and the other algorithms cited at the end of the previous section to regularized criteria, we have to solve the basic problem (6) with the regularized functions that appear in (2) and (3). Consider first

$$\min_{\|\boldsymbol{d}\|=1,\boldsymbol{x}} \quad \left\{\phi_\mu(\boldsymbol{d},\boldsymbol{x}) \triangleq \|\boldsymbol{F}-\boldsymbol{d}\boldsymbol{x}^T\|_F^2 + \mu\|\boldsymbol{x}\|^2\right\} \tag{9}$$

(The norm of the other representations has been neglected, since it does not change the outcome.)

When the representation $\boldsymbol{x}$ is fixed, the optimal atom is obviously the same as for the basic problem and is given by (8). However, when the atom is fixed, the function

$$\phi_\mu(\boldsymbol{x}) = \|\boldsymbol{F}\|_F^2 + (1+\mu)\boldsymbol{x}^T\boldsymbol{x} - 2\boldsymbol{x}\boldsymbol{F}^T\boldsymbol{d}$$

is minimized by

$$\boldsymbol{x} = \frac{1}{1+\mu}\boldsymbol{F}^T\boldsymbol{d}. \tag{10}$$

So, the effect of regularization is that the representations are dampened.

Consider now the basic problem

$$\min_{\|\boldsymbol{d}\|=1,\boldsymbol{x}} \quad \left\{\phi_\gamma(\boldsymbol{d},\boldsymbol{x}) \triangleq \|\boldsymbol{F}-\boldsymbol{d}\boldsymbol{x}^T\|_F^2 + 2\gamma\|\bar{\boldsymbol{D}}^T\boldsymbol{d}\|^2\right\} \tag{11}$$

corresponding to the coherence regularized problem (3), where $\bar{\boldsymbol{D}}$ is the current dictionary from which atom $\boldsymbol{d}$ has been removed. (The factor 2 accounts for the symmetry of the matrix $\boldsymbol{D}^T\boldsymbol{D}$.)

Algorithm 1: A step of AK-SVD and regularized versions

Data: current dictionary $\boldsymbol{D} \in \mathbb{R}^{m\times n}$
signals set $\boldsymbol{Y} \in \mathbb{R}^{m\times N}$
Result: new dictionary $\boldsymbol{D}$

Sparse coding: keeping $\boldsymbol{D}$ fixed, compute sparse representations $\boldsymbol{X}$ using OMP
for $j = 1$ **to** n **do**
  Update atom $\boldsymbol{d}_j$ using (8) or (13)
  Update its representation using (7) or (10)

When the atom is fixed, the optimal representation is the same as in (7). When the representation is fixed, the optimization of the atom leads no longer to an analytic solution. We propose an efficient alternative that works very well in practice. If the optimal representation (7) (for the yet variable atom) is inserted in (11), the function becomes

$$\phi_\gamma(\boldsymbol{d}) = \|\boldsymbol{F}\|_F^2 - \boldsymbol{d}^T(\boldsymbol{F}\boldsymbol{F}^T - 2\gamma\bar{\boldsymbol{D}}\bar{\boldsymbol{D}}^T)\boldsymbol{d}. \tag{12}$$

The function is minimized by the eigenvector $\boldsymbol{d}$ corresponding to the maximum eigenvalue of the matrix $\boldsymbol{H} = \boldsymbol{F}\boldsymbol{F}^T - 2\gamma\bar{\boldsymbol{D}}\bar{\boldsymbol{D}}^T$. In the spirit of AK-SVD, we propose to apply a single iteration of the power method on the matrix $\boldsymbol{H}$, starting with the previous version of the atom. So, we compute the new atom with

$$\boldsymbol{d} \leftarrow \boldsymbol{F}\boldsymbol{x} - 2\gamma\bar{\boldsymbol{D}}\bar{\boldsymbol{D}}^T\boldsymbol{d} \tag{13}$$

then normalize the atom.

IV. Numerical results

In this section we present numerical simulations for varied plain and regularized DL algorithms. When applying (9) we shrinked μ by 5% at each iteration. We measured similar execution times when using regularization.

As naming convention, we suffix the regularized methods with 'r' for representation damping and 'c' for coherence reduction. The parallel dictionary update algorithm variants are prefixed by the letter 'P'. Although we implemented all the methods mentioned above, for space reasons we report only the results of the best ones for each type of experiment.

In the following, we compare the resulting dictionary and sparse representations with the original signals by computing the root mean square error RMSE $= \frac{\|Y-DX\|_F}{\sqrt{mN}}$.

A. Synthetic Data

Our first set of experiments start with an arbitrary dictionary with $n = 50$ i.i.d. gaussian atoms of size $m = 20$ each, from which we construct a signal set of N vectors, every vector generated as a linear combination of s different atoms. We perturb the signals by adding white gaussian noise with SNR levels of 10, 20, 30 and ∞ dB. Next, we dismiss the original dictionary and perform DL on the noisy vectors.

In Table I we study the impact of representation damping (9) for $\mu = 0.01$ and varied sparsity constraints. We used

Table I
RMSE OF PLAIN VERSUS REGULARIZED REPRESENTATIONS ($\mu = 0.01$)

s	Method	SNR			
		10	20	30	∞
3	MOD	**0.1170**	**0.0637**	**0.0498**	0.0529
	MODr	0.1202	0.0659	0.0542	0.0505
	P-SGK	0.1227	0.0704	0.0574	0.0549
	P-SGKr	0.1213	0.0655	0.0589	**0.0504**
	NSGK	0.1242	0.0668	0.0594	0.0577
	NSGKr	0.1214	0.0691	0.0538	0.0560
	P-NSGK	0.1232	0.0733	0.0570	0.0585
	P-NSGKr	0.1237	0.0694	0.0584	0.0608
	AK-SVD	0.1199	0.0661	0.0556	0.0576
	AK-SVDr	0.1207	0.0670	0.0570	0.0531
6	MOD	0.1242	0.1114	0.1088	0.1084
	MODr	0.1240	0.1106	0.1079	0.1071
	P-SGK	0.1248	0.1113	0.1090	0.1083
	P-SGKr	0.1241	0.1103	0.1079	0.1075
	NSGK	0.1241	0.1106	0.1087	0.1075
	NSGKr	0.1234	**0.1099**	0.1082	0.1074
	P-NSGK	0.1236	0.1108	0.1086	0.1069
	P-NSGKr	0.1239	**0.1099**	**0.1074**	0.1069
	AK-SVD	0.1254	0.1110	0.1093	0.1083
	AK-SVDr	**0.1233**	0.1105	0.1078	**0.1065**
9	MOD	0.0834	0.0805	0.0786	0.0779
	MODr	0.0818	0.0782	0.0773	0.0759
	P-SGK	0.0830	0.0816	0.0794	0.0791
	P-SGKr	0.0806	0.0784	0.0769	0.0758
	NSGK	0.0811	0.0779	0.0760	0.0761
	NSGKr	0.0797	**0.0760**	0.0747	**0.0746**
	P-NSGK	0.0816	0.0766	0.0760	0.0757
	P-NSGKr	**0.0794**	0.0767	**0.0743**	**0.0746**
	AK-SVD	0.0843	0.0808	0.0782	0.0790
	AK-SVDr	0.0809	0.0784	0.0757	0.0765
12	MOD	0.0472	0.0458	0.0456	0.0463
	MODr	0.0423	0.0409	0.0408	0.0415
	P-SGK	0.0472	0.0461	0.0453	0.0469
	P-SGKr	0.0417	0.0404	0.0402	0.0412
	NSGK	0.0450	0.0432	0.0429	0.0432
	NSGKr	0.0410	0.0400	**0.0389**	**0.0396**
	P-NSGK	0.0449	0.0429	0.0421	0.0434
	P-NSGKr	**0.0408**	**0.0395**	0.0393	0.0399
	AK-SVD	0.0468	0.0461	0.0453	0.0468
	AK-SVDr	0.0413	0.0405	0.0397	0.0404

Table II
RMSE OF PLAIN VERSUS COHERENCE REDUCTION ($\gamma = 3$)

s	Method	SNR			
		10	20	30	∞
4	P-SGK	0.1279	**0.0646**	0.0579	**0.0519**
	P-SGKc	0.1367	0.0700	0.0555	0.0556
	NSGK	0.1308	0.0684	0.0586	0.0545
	NSGKc	0.1412	0.0721	0.0636	0.0593
	P-NSGK	0.1336	0.0675	0.0608	0.0580
	P-NSGKc	0.1419	0.0781	0.0646	0.0615
	AK-SVD	0.1271	0.0668	**0.0550**	0.0543
	AK-SVDc	**0.1264**	0.0661	0.0583	0.0546
6	P-SGK	0.1431	0.1224	0.1208	0.1222
	P-SGKc	0.1419	0.1218	0.1221	0.1237
	NSGK	0.1426	0.1226	0.1210	0.1224
	NSGKc	0.1435	0.1239	0.1221	0.1237
	P-NSGK	0.1433	0.1226	0.1196	0.1226
	P-NSGKc	0.1433	0.1240	0.1221	0.1240
	AK-SVD	0.1427	0.1218	0.1214	0.1214
	AK-SVDc	**0.1405**	**0.1198**	**0.1183**	**0.1197**
8	P-SGK	0.1169	0.1089	0.1085	0.1068
	P-SGKc	**0.1108**	**0.1029**	**0.1014**	**0.1007**
	NSGK	0.1137	0.1067	0.1055	0.1043
	NSGKc	0.1119	0.1041	0.1025	0.1010
	P-NSGK	0.1138	0.1067	0.1056	0.1042
	P-NSGKc	0.1114	0.1041	0.1022	0.1012
	AK-SVD	0.1169	0.1093	0.1081	0.1070
	AK-SVDc	0.1111	0.1043	0.1027	0.1018
10	P-SGK	0.0846	0.0805	0.0809	0.0813
	P-SGKc	**0.0748**	**0.0705**	**0.0702**	**0.0708**
	NSGK	0.0811	0.0767	0.0780	0.0779
	NSGKc	0.0759	0.0715	0.0711	0.0713
	P-NSGK	0.0804	0.0767	0.0768	0.0770
	P-NSGKc	0.0759	0.0709	0.0707	0.0712
	AK-SVD	0.0841	0.0813	0.0811	0.0816
	AK-SVDc	0.0760	0.0713	0.0719	0.0728
12	P-SGK	0.0574	0.0548	0.0547	0.0553
	P-SGKc	**0.0452**	**0.0430**	**0.0427**	**0.0426**
	NSGK	0.0547	0.0523	0.0516	0.0523
	NSGKc	0.0458	0.0433	0.0428	0.0431
	P-NSGK	0.0538	0.0519	0.0516	0.0516
	P-NSGKc	0.0456	0.0434	0.0429	0.0430
	AK-SVD	0.0577	0.0556	0.0552	0.0556
	AK-SVDc	0.0461	0.0436	0.0432	0.0436

signal sets of $N = 512$ vectors and trained dictionaries for 50 iterations. The results are the average of 10 runs of each algorithm. MOD is the best for $s = 3$, but afterwards the regularized versions are the absolute winners.

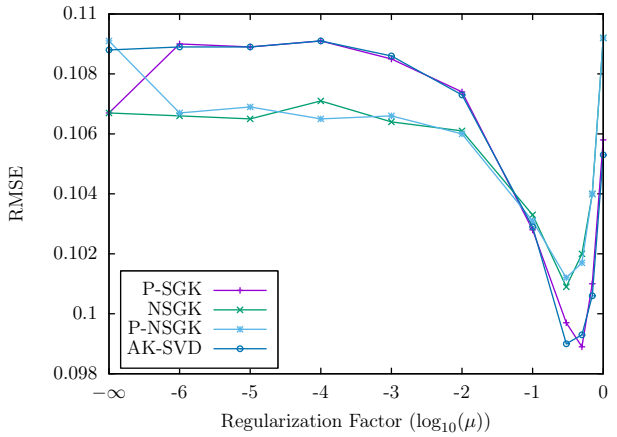
We follow with Table II where we depict the results of applying the coherence reduction (11) for $\gamma = 3$ at different sparsity levels, with $N = 1500$. Except for $s = 4$, the coherence reduction strategy is the ubiquitous winner, the best results being split between P-SGKc and AK-SVDc. Note that MOD cannot be adapted to this regularization.

Figure 1 explores the choice of the regularization constants μ and γ in the upper and the lower panel, respectively. The experiment was done on the same initial data, with $s = 8$, $N = 1500$ and the SNR at 20dB. Each point on the graph is an average of 10 runs. The upper panel shows that the optimal choice for μ is somewhere between 0.1 and 1. We tested in that interval with $\mu = \{0.3, 0.5, 0.7\}$. In the lower panel we see the approximation improvement as we increase the coherence factor until it stalls past $\gamma = 3$.

B. Images

In this section we present a few experiments on real data collected from the USC-SIPI database. We sample random 8×8 image patches that we vectorize as a dictionary training set with $N = 2048$ signals. We show in Table III the results, with and without regularization, when performing DL for a dictionary of $n = 128$ atoms on for varied sparsity constraints. Each method executes 50 iterations and each data point represents an average of 10 runs. We emphasised the winning variation of each row and for each sparsity level s we mark the overall winner with an extra † symbol.

Excepting $s = 6$ where the standard algorithms dominated, all the other tests clearly indicate the benefit of the regular-



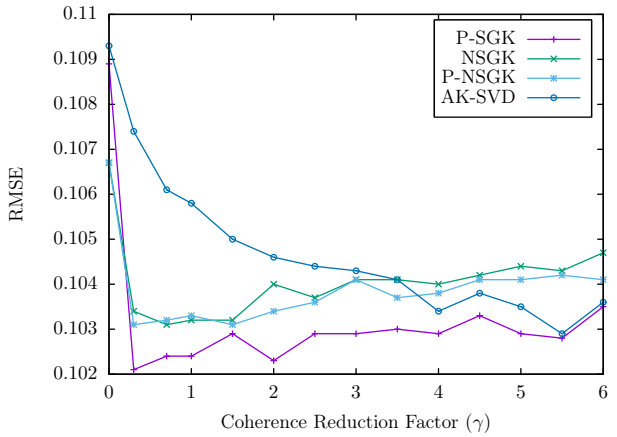


Figure 1. Final errors averaged over 10 runs for $s = 8$ with SNR= 20.

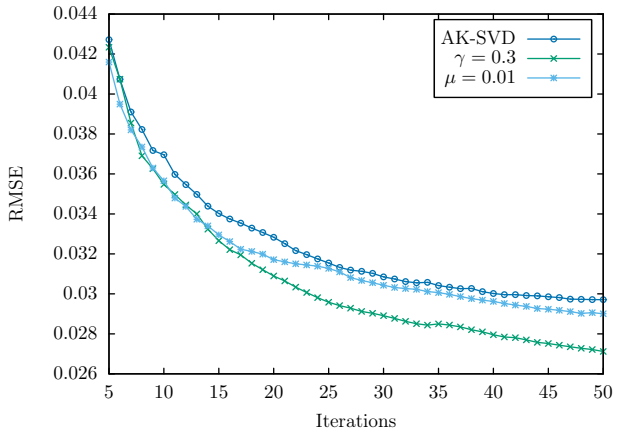


Figure 2. AK-SVD error evolution for $s = 8$ averaged over 10 runs.

ization for all algorithms but P-NSGKc.

In Figure 2 we show the error evolution of the AK-SVD algorithm along with its regularized variants for the $s = 8$ case from Table III. We can see that the curves are similar in terms of smoothness and descent with the regularization versions always ahead of the regular AK-SVD.

V. Conclusions

We have presented regularized versions of AK-SVD and related algorithms. Extensive numerical experiments have shown that, except for very high sparsity, the regularized versions are able to reach lower values of the representation error and thus confirm their usefulness in the family of dictionary learning algorithms.

Table III
RMSE of plain versus regularized DL

s	Method	$\gamma = \mu = 0$	$\mu = 0.01$	$\gamma = 0.3$
4	NSGK	0.0367	0.0364	**0.0340**
	P-NSGK	0.0360	**0.0358**	0.0401
	AK-SVD	0.0388	0.0383	**0.0349**
	PAK-SVD	0.0353	0.0352	**0.0338**†
6	NSGK	**0.0269**	0.0306	0.0309
	P-NSGK	**0.0242**†	0.0289	0.0392
	AK-SVD	**0.0296**	0.0321	**0.0296**
	PAK-SVD	**0.0251**	0.0287	0.0285
8	NSGK	0.0269	**0.0264**	0.0276
	P-NSGK	0.0242	**0.0241**†	0.0462
	AK-SVD	0.0296	0.0293	**0.0271**
	PAK-SVD	0.0251	0.0248	**0.0246**
10	NSGK	0.0236	**0.0235**	0.0261
	P-NSGK	0.0220	**0.0215**†	0.0551
	AK-SVD	0.0270	0.0268	**0.0246**
	PAK-SVD	0.0218	**0.0217**	0.0220
12	NSGK	0.0220	**0.0219**	0.0237
	P-NSGK	0.0199	**0.0193**†	0.0486
	AK-SVD	0.0257	0.0254	**0.0231**
	PAK-SVD	0.0210	0.0205	**0.0202**

Acknowledgements

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

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] 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.

[4] C.D. Sigg, T. Dikk, and J.D. Buhmann, "Learning Dictionaries With Bounded Self-Coherence," *IEEE Signal Proc. Letters*, vol. 19, no. 19, pp. 861–865, Dec. 2012.

[5] 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.

[6] 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.

[7] 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.

[8] S. K. Sahoo and A. Makur, "Dictionary training for sparse representation as generalization of K-Means clustering," *Signal Processing Letters, IEEE*, vol. 20, no. 6, pp. 587–590, June 2013.

[9] M. Sadeghi, M. Babaie-Zadeh, and C. Jutten, "Dictionary Learning for Sparse Representation: a Novel Approach," *IEEE Signal Proc. Letter*, vol. 20, no. 12, pp. 1195–1198, Dec. 2013.

[10] P. Irofti and B. Dumitrescu, "GPU Parallel Implementation of the Approximate K-SVD Algorithm Using OpenCL," in *EUSIPCO*, Lisbon, Portugal, 2014.