Learning Dictionaries as a sum of Kronecker products

Cassio Fraga Dantas, Michele N. da Costa, Renato da Rocha Lopes

Learning Dictionaries as a sum of Kronecker products

Cássio Fraga Dantas, Michele N. da Costa, Renato da Rocha Lopes, *Senior Member, IEEE*

***Abstract*—The choice of an appropriate frame, or dictionary, is a crucial step in the sparse representation of a given class of signals. Traditional dictionary learning techniques generally lead to unstructured dictionaries which are costly to deploy and train, and do not scale well to higher dimensional signals. In order to overcome such limitation, we propose a learning algorithm that constrains the dictionary to be a sum of Kronecker products of smaller sub-dictionaries. This approach, named SuKro, is demonstrated experimentally on an image denoising application.**

***Index Terms*—Kronecker product, dictionary learning, K-SVD, image denoising, separable dictionaries, ADMM, nuclear norm.**

I. Introduction

Sparse signal representation relies on the assumption that an input signal $\mathbf{y} \in \mathbb{R}^n$ can be represented as a linear combination of a small set of atoms from an appropriate, potentially over-complete, dictionary $\mathbf{D} \in \mathbb{R}^{n \times m}$ with $n \leq m$:

$$\mathbf{y} = \mathbf{D}\mathbf{x}$$

with $\mathbf{x} \in \mathbb{R}^m$ being the sparse representation vector with only $k \ll n$ non-zero elements.

The applicability of this model depends on the choice of a well suited dictionary $\mathbf{D}$. Originally, some analytical transformations were used, such as the Discrete Cosine Transform (DCT) [1], Wavelets [2] and Curvelets [3]. However, data-oriented approaches have recently been shown to achieve a better performance [4], due to the possibility of adapting the dictionary to a specific class of signals of interest.

Data-driven dictionaries are constructed by a training process over an input database. Let N be the total number of samples in the database, arranged as the columns of a training matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N] \in \mathbb{R}^{n \times N}$. Each data sample in $\mathbf{Y}$ is to be approximated over the overcomplete dictionary $\mathbf{D}$. This approximation should be based on no more than t columns of the dictionary and yield a small representation error, which results in the following optimization problem [4]:

$$\begin{aligned} \langle \mathbf{D}, \mathbf{X} \rangle = &\underset{\mathbf{D},\mathbf{X}}{\operatorname{argmin}} \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 \\ &\text{s.t.} \quad \forall i \ \|\mathbf{x}_i\|_0 \leq t, \quad \forall j \ \|\mathbf{d}_j\|_2 = 1 \end{aligned} \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{m \times N}$ is the sparse representation matrix with sparse columns $\mathbf{x}_i$, $i \in \{1, \dots, N\}$. The ℓ_0 pseudo norm, which counts the number of non-zero elements of a vector, was used as a sparsity inducing function, and the Frobenius norm was used to measure the error. Several iterative solutions have been proposed for optimization problems similar to (1), such as the Method of Optimal Directions (MOD) [5], the K-SVD [6] and an online approach [7]. For a more comprehensive survey of the area, we refer the reader to [4].

In this letter, we are mostly concerned with the computational complexity of the multiplications between the dictionary $\mathbf{D}$ (or its transpose $\mathbf{D}^T$) and the data vector, which arises in the design and application of the dictionary. Here, a tradeoff is found. Analytic dictionaries such as the DCT usually allow for fast implementations of these multiplications based on the underlying structure. However, they may yield a poor representation of certain datasets. On the other hand, data-driven dictionaries are usually represented by unstructured over-complete matrices which are costly to operate with, restricting their application to relatively small signals. However, they may provide a better and sparser match to the data.

To obtain computationally efficient dictionaries, some of the most recent works in the field employ parametric models in the training process, which produce structured dictionaries. The idea is to find a good compromise between the computational efficiency of the analytic dictionaries and the flexibility of the learned ones. As a byproduct, structured dictionaries also require fewer training symbols, since they have fewer parameters to be optimized. A promising structure in this regard is the separable dictionaries [8], which can be represented as the Kronecker product of two sub-dictionaries, i.e. $\mathbf{D} = \mathbf{B} \otimes \mathbf{C}$. This particular structure arises naturally when treating multi-dimensional data, such as images.

In this letter, we propose a broader structure class, which consists in a sum of separable terms, where the number of components serves as a fine tuner for the complexity-adaptability tradeoff:

$$\mathbf{D} = \sum_{r=1}^{\alpha} \mathbf{B}^{(r)} \otimes \mathbf{C}^{(r)}. \tag{2}$$

Clearly, the separable structure is a special case, with $\alpha = 1$.

To design the dictionary, we use a mathematical result [9] that establishes a relation between the structure in (2) and a rank-α matrix. As is well-known, optimization problems involving a rank constraint are hard to solve. In this letter, we exploit the fact that good solutions can be obtained by relaxing this constraint using the nuclear norm [10]. As we will see, this optimization framework produces better separable dictionaries

The authors are with DSPCom Laboratory, University of Campinas, Brazil (e-mail: {cassio, nazareth, rlopes}@decom.fee.unicamp.br). This work was supported by FAPESP, under grant 2014/23936-4, and CAPES agency.

Code is available at https://github.com/cassiofragadantas/SuKro-DL.

than those from [8], which directly imposes the separable structure.

This letter is organized as follows. In Section II-A we shortly review some existing proposals on structured dictionary learning. The proposed methodology is formulated in Section II-B, followed by a complexity analysis in Section II-C. The proposed optimization approach is described in Section III. Finally, we present some image denoising results in Section IV and conclude in Section V.

Throughout the letter we use the vectorization operator $\mathrm{vec} : \mathbb{R}^{a\times b} \to \mathbb{R}^{ab\times 1}$, which stacks the columns of a matrix on top of each other. We denote unvec the inverse operation.

II. Structured Dictionary Learning

A. Related work

Besides separable dictionaries [8], other types of structure have been explored in the literature. In [11], each atom (column) of the dictionary is a sub-block of a compact *signature image*. The reduced number of parameters (only one per atom) makes the model overly restrictive. A more flexible approach is the search for sparse dictionaries [12], [13], which are written as the product of two matrices, one of which is sparse. In [12] the non-sparse matrix is a pre-specified *base dictionary*, such as the DCT, that has a fast implementation. In [14] this idea is taken further by replacing the fixed base dictionary by an adaptable multi-scale one, called *cropped* Wavelets. It provides more flexibility to the model, while maintaining its scalability. The sparsity idea is also extended in [15], constraining the dictionary to be the product of several sparse matrices. The total complexity in this case is determined by the number of non-zero values on the factor matrices. In [16], the dictionary atoms are the composition of several circular convolutions using sparse kernels. All the mentioned techniques obtain promising complexity-performance compromises, but their approaches are intrinsically different from the one proposed in this article.

B. Proposed technique

We begin by introducing a useful result that provides a way of transforming a Kronecker product into a rank-1 matrix [9]. To that end, consider a matrix $\mathbf{D} \in \mathbb{R}^{n_1 n_2 \times m_1 m_2}$ which is the Kronecker product of two sub-matrices $\mathbf{B} \in \mathbb{R}^{n_1 \times m_1}$ and $\mathbf{C} \in \mathbb{R}^{n_2 \times m_2}$ given by

$$\mathbf{D} = \mathbf{B} \otimes \mathbf{C}.$$

Now, define a *rearrangement operator*, which reorganizes the elements $d^{\text{in}}_{i,j}$ of $\mathbf{D}$ into a rearranged matrix $\mathcal{R}(\mathbf{D}) \in \mathbb{R}^{m_1 n_1 \times m_2 n_2}$, whose elements $d^{\text{out}}_{i,j}$ are given by

$$d^{\text{out}}_{i_1+(j_1-1)n_1, i_2+(j_2-1)n_2} = d^{\text{in}}_{i_2+(i_1-1)n_2, j_2+(j_1-1)m_2} \qquad (3)$$

with $i_l \in \{1, 2, \dots, n_l\}$, $j_l \in \{1, 2, \dots, m_l\}$ and $l \in \{1, 2\}$. This rearrangement leads to a rank-1 matrix [9], which can be written as the product of the vectorized versions of $\mathbf{B}$ and $\mathbf{C}$:

$$\mathcal{R}(\mathbf{D}) = \mathrm{vec}(\mathbf{B})\,\mathrm{vec}(\mathbf{C})^{\mathrm{T}}. \qquad (4)$$

This result is not unexpected, since the elements of both $\mathcal{R}(\mathbf{D})$ and $\mathbf{D}$ are all the possible products of a pair of elements from $\mathbf{B}$ and $\mathbf{C}$.

Now, let us consider a sum of α Kronecker products

$$\mathbf{D} = \sum_{r=1}^{\alpha} \mathbf{B}^{(r)} \otimes \mathbf{C}^{(r)} = \sum_{r=1}^{\alpha} \mathbf{D}^{(r)}. \qquad (5)$$

After rearrangement, we obtain a rank-α matrix, since each term $\mathbf{D}^{(r)}$ leads to a rank-1 matrix as follows

$$\mathcal{R}(\mathbf{D}) = \sum_{r=1}^{\alpha} \mathcal{R}(\mathbf{D}^{(r)}) = \sum_{r=1}^{\alpha} \mathrm{vec}\Big(\mathbf{B}^{(r)}\Big)\,\mathrm{vec}\Big(\mathbf{C}^{(r)}\Big)^{\mathrm{T}}. \qquad (6)$$

Therefore, by using (6), we can introduce a low-rank regularization term to the original optimization problem in order to learn a dictionary as a sum of few Kronecker products:

$$\begin{aligned} &\min_{\mathbf{D},\mathbf{X}} \ \|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 + \lambda\,\mathrm{rank}(\mathcal{R}(\mathbf{D})) \qquad (7)\\ &\text{s.t.} \quad \forall i \ \|\mathbf{x}_i\|_0 \le t \ , \quad \forall j \ \|\mathbf{d}_j\|_2 = 1 \end{aligned}$$

where the parameter $\lambda \in \mathbb{R}^+$ controls the rank penalty: larger values of λ will lead to solutions with smaller ranks, and thus with fewer terms in (5). During the design, the value of λ must be tweaked until a desired value of α is obtained.

Note that any dictionary has an associated matrix $\mathcal{R}(\mathbf{D})$ with a certain rank, so our formulation may design an arbitrary dictionary. There is, however, no reason to expect that the resulting rank will be small, so there is no reason to expect that the proposed structure will be advantageous for all dictionaries. The simulation results in section IV, however, indicate that it does work well for several images, even for small values of α.

Finally, note that we do not explicitly impose the structure defined in (5) in our design criterion; instead, we limit the rank of the rearranged matrix $\mathcal{R}(\mathbf{D})$ through a rank penalization on the cost function. As we will see, this formulation allows us to benefit from powerful convex optimization tools that yield better dictionaries than those in [8], which explicitly impose the separable structure $\mathbf{D} = \mathbf{B} \otimes \mathbf{C}$.

C. Computational complexity analysis

Complexity savings are expected when operating with structured matrices. When it comes to a matrix-vector multiplication, the separable structure can be exploited by using the following Kronecker product property:

$$(\mathbf{B} \otimes \mathbf{C})\mathbf{x} = \mathrm{vec}(\mathbf{C}\,\mathrm{unvec}(\mathbf{x})\mathbf{B}^{\mathrm{T}}). \qquad (8)$$

The right-hand side expression contains a product of three matrices with sizes $(n_2 \times m_2)$, $(m_2 \times m_1)$ and $(m_1 \times n_1)$ respectively. If no particular structure is imposed to the sub-matrices $\mathbf{B}$ and $\mathbf{C}$, we obtain a complexity (in total number of multiplications and additions) of

$$2m_1 n_2 (n_1 + m_2). \qquad (9)$$

For instance, if we assume $n_1 = n_2 = \sqrt{n}$ and $m_1 = m_2 = \sqrt{m}$ the complexity becomes:

$$2(\sqrt{m}n + m\sqrt{n}) \qquad (10)$$

which is a considerable reduction when compared to the $2mn$ operations in the case of an unstructured matrix.

For a matrix with α separable terms in the form of eq. (2), the total complexity becomes:

$$2\alpha(\sqrt{m}n + m\sqrt{n}). \tag{11}$$

III. Optimization framework

We solve the problem in (7) by alternately minimizing on the variables $\mathbf{D}$ and $\mathbf{X}$, as typically done on the literature [5]–[7]. The minimization on $\mathbf{X}$ is called the *sparse coding* step and the minimization on $\mathbf{D}$ is the *dictionary update* step. In this article, we use the Orthogonal Matching Pursuit (OMP) algorithm [17] for sub-optimally solving the NP-hard *sparse coding* problem.

The dictionary update step, in its turn, has been modified by the addition of the rank regularization term. Given the non-convexity of the rank operator, we use the nuclear norm (denoted $\|\cdot\|_*$) as its convex relaxation [10], which yields

$$\text{Dict. update:} \quad \min_{\mathbf{D}} \ \|\mathbf{DX}-\mathbf{Y}\|_F^2 + \lambda\|\mathcal{R}(\mathbf{D})\|_*. \tag{12}$$

The above problem cannot be addressed by a regular gradient descent, since the nuclear norm operator is not differentiable. However, the following variable introduction turns it into an approachable equality constrained problem:

$$\begin{aligned} &\min_{\mathbf{D},\widetilde{\mathbf{D}}} \ \|\mathbf{DX}-\mathbf{Y}\|_F^2 + \lambda\|\widetilde{\mathbf{D}}\|_* \\ &\text{s.t.} \quad \widetilde{\mathbf{D}} = \mathcal{R}(\mathbf{D}). \end{aligned} \tag{13}$$

which can be solved by the Alternating Direction Method of Multipliers (ADMM) [18]. It performs partial updates on the minimization variables $\mathbf{D}$ and $\widetilde{\mathbf{D}}$ before updating a Lagrangian multiplier matrix $\mathbf{Z}$, as shown in Algorithm 1. Note that the normalization constraint on the dictionary atoms is handled in a post-processing step.

Algorithm 1 Dictionary Update - ADMM

Initialize $\mathbf{D}_0$, $\widetilde{\mathbf{D}}_0$, $\mathbf{Z}_0$
while $\|\mathbf{Z}_{k+1}-\mathbf{Z}_k\|_F > tol$ **do**
$\mathbf{D}_{k+1} \approx \text{argmin}_{\mathbf{D}} \|\mathbf{DX}-\mathbf{Y}\|_F^2 + \frac{\mu}{2}\left\|\widetilde{\mathbf{D}}_k - \mathcal{R}(\mathbf{D}) - \mathbf{Z}_k\right\|_F^2$
$\widetilde{\mathbf{D}}_{k+1} = \text{argmin}_{\widetilde{\mathbf{D}}} \lambda\|\widetilde{\mathbf{D}}\|_* + \frac{\mu}{2}\left\|\widetilde{\mathbf{D}} - \mathcal{R}(\mathbf{D}_{k+1}) - \mathbf{Z}_k\right\|_F^2$
$\mathbf{Z}_{k+1} = \mathbf{Z}_k - \left(\widetilde{\mathbf{D}}_{k+1} - \mathcal{R}(\mathbf{D}_{k+1})\right)$
end while
Normalize columns of $\mathbf{D}$

The update with respect to the variable $\mathbf{D}$ (first step in Alg. 1) is just a partial solution of the associated minimization problem and corresponds to a single gradient step:

$$\mathbf{D}_{k+1} = \mathbf{D}_k - \gamma\nabla_{\mathbf{D}}(\mathrm{J}) \tag{14}$$

where $\mathrm{J} = \|\mathbf{DX}-\mathbf{Y}\|_F^2 + \frac{\mu}{2}\left\|\widetilde{\mathbf{D}}_k - \mathcal{R}(\mathbf{D}) - \mathbf{Z}_k\right\|_F^2$ and γ is the stepsize. In order to calculate $\nabla_{\mathbf{D}}(\mathrm{J})$ we use the fact that the Frobenius norm is indifferent to the elements ordering on a matrix. So, by applying the inverse of the rearrangement operation[1] $\mathcal{R}$, denoted $\mathcal{R}^{-1}$, the second term in J can be rewritten as $\left\|\mathcal{R}^{-1}(\widetilde{\mathbf{D}}) - \mathbf{D} - \mathcal{R}^{-1}(\mathbf{Z})\right\|_F^2$. The gradient is therefore given by:

$$\nabla_{\mathbf{D}}(\mathrm{J}) = 2(\mathbf{D}_k\mathbf{X}-\mathbf{Y})\mathbf{X}^T + \mu\Big(\mathbf{D}_k - \mathcal{R}^{-1}(\widetilde{\mathbf{D}}_k - \mathbf{Z}_k)\Big).$$

The partial update with respect to the variable $\widetilde{\mathbf{D}}$ (second step in Alg. 1) is the proximal operator associated to the nuclear norm (denoted $\text{prox}_{\frac{\lambda}{\mu}\|\cdot\|_*}$). It consists in the singular value soft-thresholding operation, see [19] for details.

$$\widetilde{\mathbf{D}}_{k+1} = \text{prox}_{\frac{\lambda}{\mu}\|\cdot\|_*}\left(\mathcal{R}(\mathbf{D}_{k+1}) + \mathbf{Z}_k\right). \tag{15}$$

The variation on the multiplier matrix $\mathbf{Z}$ was used as a convergence criterion.

Once a rank-α $\mathcal{R}(\mathbf{D})$ is obtained, the sub-matrices $\mathbf{B}^{(r)}$ and $\mathbf{C}^{(r)}$ can be determined from any low-rank factorization $\mathcal{R}(\mathbf{D}) = \mathbf{LR}$. Each of the α columns of the left matrix $\mathbf{L}$ gives a $\mathbf{B}^{(r)}$ and each of the α rows of the right matrix $\mathbf{R}$ gives a $\mathbf{C}^{(r)}$ through an unvectorization operation.

IV. Simulation Results

We have chosen an image denoising application to validate the proposed algorithm, herein called SuKro (Sum of Kronecker products). We use the same simulation set-up used in [20], where five images (*barbara*, *boats*, *house*, *lena* and *peppers*) are corrupted by a white Gaussian noise with different standard deviations σ.

The training data is composed by vectorized versions of uniformly spaced and potentially overlapping (8×8)-pixel patches from the noisy image. The obtained dictionary $\mathbf{D}$ is used for reconstructing all the image patches. The recovered image is constructed by averaging the overlapping pixels.

The PSNR of the reconstructed images are evaluated, and all the reported results are averaged over 10 different noise realizations.

$$\text{PSNR} = 10\log\left(\frac{255^2 N_{px}}{\sum_{i=1}^{N_{px}}(y_i - \hat{y}_i)^2}\right)$$

where 255 is the maximum pixel value, N_{px} is the total number of pixels on the input image, y_i and $\hat{y}_i$ are respectively the i-th pixel value on the input and reconstructed image. By default, the simulation parameters are:

$n = 64$	$m = 256$	$N = 40000$	$\mathbf{D}_0$: ODCT[2]
$\mu = 10^7$	$\gamma = 6\times10^{-9}$	$N_{iter} = 100$	$tol = \|\mathbf{D}\|_F \times 10^{-4}$.
$n_1 = \sqrt{n}$	$n_2 = \sqrt{n}$	$m_1 = \sqrt{m}$	$m_2 = \sqrt{m}$

We have compared our results to the K-SVD [6] algorithm, which learns unstructured dictionaries, the SeDiL [8] algorithm for learning separable dictionaries (both using

[1] $\mathcal{R}^{-1}$ is obtained by reversing the input and output indexes on Eq. (3) $d^{\text{out}}_{i_2+(i_1-1)n_2, j_2+(j_1-1)m_2} = d^{\text{in}}_{i_1+(j_1-1)n_1, i_2+(j_2-1)n_2}$.

[2] The 1-D $n \times m$ overcomplete DCT dictionary, as defined in [12], is a cropped version of the orthogonal $m \times m$ DCT matrix. The 2-D ODCT is the Kronecker product of two 1-D ODCT dictionaries of size $\sqrt{n}\times\sqrt{m}$. Its complexity is that of a general separable matrix, since the fast implementation of each term is lost when the DCT is truncated and renormalized).

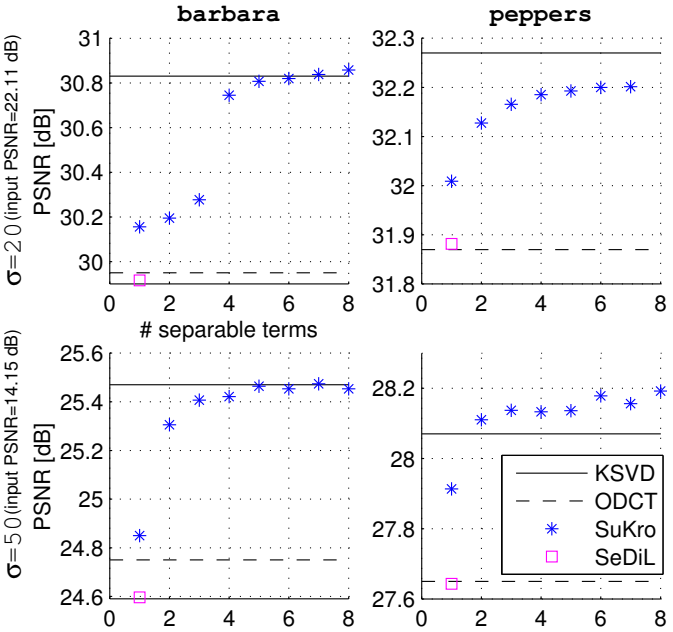

Fig. 1. PSNR vs. $\mathrm{rank}(\widetilde{\mathbf{D}})$ (i.e. the number of separable terms).

the code provided by the authors), and the ODCT analytic dictionary.

Originally in the SeDiL algorithm [8], the training data was extracted from different noiseless images, unlike the simulation set-up adopted here. In addition, we use the OMP algorithm instead of FISTA [21] for the sparse coding step. As indicated by the simulation results, the SeDiL algorithm may underperform the ODCT in this new configuration.

In Figure 1 we show the denoised image PSNR as a function of the number of separable terms in the dictionary, which can be indirectly controlled by the parameter λ in (12). Naturally, as the number of separable terms increases, so does the denoising performance, since the dictionary becomes more flexible. The drawback is the increase on the dictionary application complexity. Note, however, that even with very few separable terms, the results of SuKro are close to the K-SVD, which does not impose any structure on the dictionary.

Note, in Figure 1, that SuKro may even outperform K-SVD for high-noise scenarios. The reason is that, by reducing the flexibility of the dictionary, we may end up preventing overfitting. Finally, note that, even with a single separable term, SuKro outperforms SeDiL, which designs a dictionary with the same structure. This is a consequence of the difference between the design criteria in both methods.

Figure 2 illustrates the complexity-performance tradeoff. For one separable term, we obtain a considerably better performance than ODCT and SeDiL dictionaries while having exactly the same computational complexity for matrix-vector multiplication. Besides that, the proposed technique has the merit of providing a range of options on this tradeoff curve.

Finally, Figure 3 shows the performance of the methods as a function of the number of training samples. The performance is shown in terms of ΔPSNR, which is defined as the PNSR difference with respect to the PSNR obtained by ODCT. Note how the performance of K-SVD suffers for small datasets.

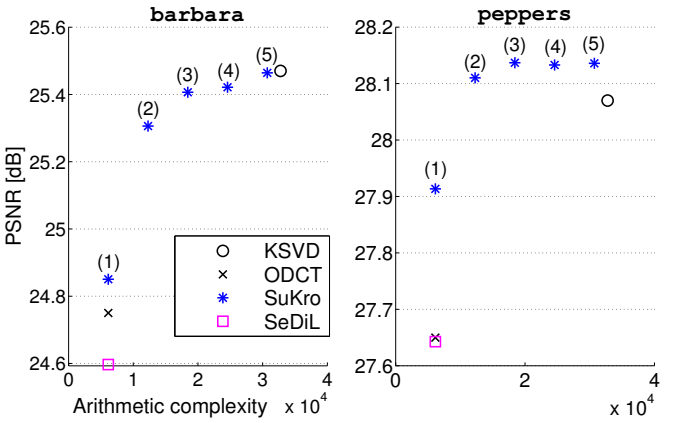

Fig. 2. Performance (PSNR) vs. Complexity, with $\sigma = 50$. The number of separable terms is displayed between parentheses.

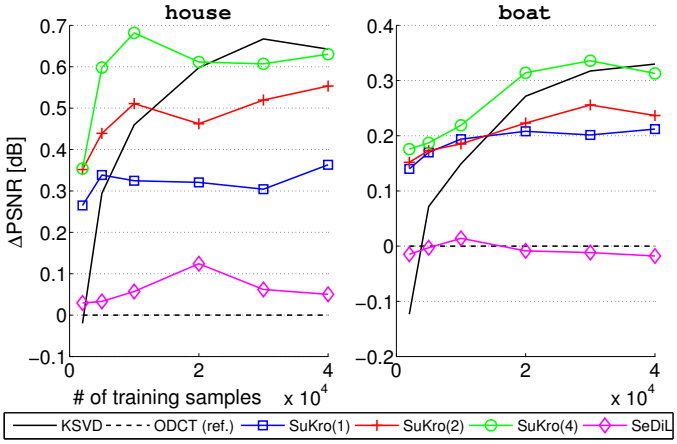

Fig. 3. Robustness to reduced training datasets, with $\sigma = 50$. The PNSR difference is taken with respect to the ODCT (reference). The number of separable terms is displayed between parentheses.

On the other hand, structured dictionaries like SuKro and SeDiL are more robust to reduced training sets, due to the decreased number of free parameters to estimate compared to unstructured dictionaries. Note also how, in SuKro, the advantage of adding more terms depends on the size of the training set. For instance, in the smallest training dataset in the house image, using two or four separable terms yield the same performance.

V. Conclusions

We have proposed a novel dictionary structure which consists on a sum of separable terms. It leads to fast operators while keeping a considerable degree of flexibility. Such tradeoff can be controlled through the number of terms in the summation. The proposed technique relies on a rank reduction constraint, which is handled via a nuclear norm relaxation. The image denoising simulations have shown very promising results, especially for higher noise and reduced training dataset scenarios. In such cases, the proposed structure manages to overcome the unstructured K-SVD dictionary in terms of both computational complexity and denoising performance.

REFERENCES

[1] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, Jan 1974.

[2] S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*, 3rd ed. Academic Press, 2008.

[3] J.-L. Starck, E. J. Candès, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions on image processing*, vol. 11, no. 6, pp. 670–684, 2002.

[4] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

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

[6] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, Nov 2006.

[7] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 689–696.

[8] 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.

[9] C. F. Van Loan and N. Pitsianis, "Approximation with kronecker products," in *Linear algebra for large scale and real-time applications*. Springer, 1993, pp. 293–314.

[10] B. Recht, M. Fazel, and P. A. Parrilo, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization," *SIAM Review*, vol. 52, no. 3, pp. 471–501, 2010. [Online]. Available: http://dx.doi.org/10.1137/070697835

[11] M. Aharon and M. Elad, "Sparse and redundant modeling of image content using an image-signature-dictionary," *SIAM Journal on Imaging Sciences*, vol. 1, no. 3, pp. 228–247, 2008. [Online]. Available: http://dx.doi.org/10.1137/07070156X

[12] R. Rubinstein, M. Zibulevsky, and M. Elad, "Double sparsity: Learning sparse dictionaries for sparse signal approximation," *Signal Processing, IEEE Transactions on*, vol. 58, no. 3, pp. 1553–1564, 2010.

[13] M. Yaghoobi and M. E. Davies, "Compressible dictionary learning for fast sparse approximations," in *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*, Aug 2009, pp. 662–665.

[14] J. Sulam, B. Ophir, M. Zibulevsky, and M. Elad, "Trainlets: Dictionary learning in high dimensions," *IEEE Transactions on Signal Processing*, vol. 64, no. 12, pp. 3180–3193, June 2016.

[15] L. L. Magoarou and R. Gribonval, "Flexible multilayer sparse approximations of matrices and applications," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 688–700, June 2016.

[16] O. Chabiron, F. Malgouyres, J.-Y. Tourneret, and N. Dobigeon, "Toward fast transform learning," *International Journal of Computer Vision*, vol. 114, no. 2-3, pp. 195–216, 2015.

[17] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, Nov 1993, pp. 40–44 vol.1.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jan 2011. [Online]. Available: http://dx.doi.org/10.1561/2200000016

[19] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[20] M. Elad and M. Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, Dec 2006.

[21] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009. [Online]. Available: http://dx.doi.org/10.1137/080716542