MULTI-DIMENSIONAL SIGNAL RECOVERY USING LOW-RANK DECONVOLUTION

David Reixach

Universitat Politècnica de Catalunya, BarcelonaTech, Spain

ABSTRACT

In this work we present Low-rank Deconvolution, a powerful framework for low-level feature-map learning for efficient signal representation with application to signal recovery. Its formulation in multi-linear algebra inherits properties from convolutional sparse coding and low-rank approximation methods as in this setting signals are decomposed in a set of filters convolved with a set of low-rank tensors. We show its advantages by learning compressed video representations and solving image in-painting problems.

Index Terms— Tensors, Sparse Coding, Low-rank Approximation, Tensor completion

1. INTRODUCTION

Convolutional Sparse Coding (CSC) aims to decompose an input signal $\mathbf{S}$ as a sum of few learned features (dictionary) $\{\mathbf{D}_m\}$ convolved with a set of sparse activation maps $\{\mathbf{X}_m\}$ such that $\mathbf{S} \approx \sum_m \mathbf{D}_m * \mathbf{X}_m$. It achieved certain relevance once it was proved that it could be efficiently formulated in the frequency domain in terms of the Convolutional Basis Pursuit Denoising (CBPDN) [1] as an optimization problem. As a result, this method has proven application in many areas such image denoising, in-painting, super-resolution among others [2, 3, 4, 5, 6]. Another technique commonly used to learn low/mid level features is Low-rank (LR) approximation, which relays on the fact that data obtained from many natural processes presents a low-rank structure. These methods are popular for denoising and completion [7, 8, 9]. And also in the multi-linear algebra domain as low-rank tensor applications [10, 11, 12].

Our approach considers a multi-linear convolutional model with sparse and low-rank activation maps that inherits properties from both CSC and LR. The idea is not new as we follow [13, 14]. However, our contribution goes further as we present a powerful linear-algebra formulation that allows for learning complex low-rank activations, for a given set of features, which permit us to represent multidimensional signals more efficiently than classical CSC approaches, also with direct application in tensor completion settings which we prove by learning compressed representations of video sequences and solving image in-painting problems. And finally we show that in this framework the sparsity constraint can be avoided allowing for simple algorithms to solve the optimization problem.

1.1. Notation

Let $\mathcal{K} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ be a N-order tensor. The PARAFAC [15] decomposition (a.k.a. CANDECOMP [16]) is defined as:

$$\mathcal{K} \approx \sum_{r=1}^{R} \mu_r \mathbf{v}_r^{(1)} \circ \mathbf{v}_r^{(2)} \circ \ldots \circ \mathbf{v}_r^{(N)}, \tag{1}$$

where $\mathbf{v}_r^{(n)} \in \mathbb{R}^{I_n}$ with $n = \{1, \ldots, N\}$ and $\mu_r \in \mathbb{R}$ with $r = \{1, \ldots, R\}$, represent an one-order tensor and a weight coefficient, respectively. $\circ$ denotes an outer product of vectors. Basically, Eq. (1) is a rank-R decomposition of $\mathcal{K}$ by means of a sum of R rank-1 tensors. If we group all these vectors per mode (n), as $\mathbf{X}^{(n)} = \left[\mathbf{v}_1^{(n)}, \mathbf{v}_2^{(n)}, \ldots, \mathbf{v}_R^{(n)}\right]$, we can define the Kruskal operator [17] as follows:

$$[\![\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(N)}]\!] = \sum_{r=1}^{R} \mathbf{v}_r^{(1)} \circ \mathbf{v}_r^{(2)} \circ \ldots \circ \mathbf{v}_r^{(N)}, \tag{2}$$

being the same expression as Eq. (1) with $\mu_r = 1$ for $\forall r$, i.e., depicting a rank-R decomposable tensor.

For later computations, we also define a matricization transformation to express tensors in a matrix form. Particularly, we will use an special case of matricization known as n-mode matricization [18, 17]. To this end, let $\mathcal{C} = \{c_1, \ldots, c_G\} = \{1, \ldots, n-1, n+1, \ldots, N\}$ be the collection of ordered modes different than n, and $\Lambda = \prod_t {}^{I_t}/_{I_n}$ be the product of its correspondent dimensions; we can express then tensor $\mathcal{K}$ in a matricized array as ${}^{(n)}\mathbf{K} \in \mathbb{R}^{I_n \times \Lambda}$. Note that we represent the n-mode matricization by means of a left super-index. The n-mode matricization is a mapping from the indices of $\mathcal{K}$ to those of ${}^{(n)}\mathbf{K}$, defined as:

$$\left({}^{(n)}\mathbf{K}\right)_{i_n, j} = \mathcal{K}_{i_1, i_2, \ldots, i_N}, \tag{3}$$

with:

$$j = 1 + \sum_{g=1}^{G} \left[(i_{c_g} - 1) \prod_{g'=1}^{G-1} I_{c_{g'}}\right]. \tag{4}$$

With these ingredients, and defining $\mathcal{J}(\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)}) = [\![\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)}]\!]$, we can obtain the n-mode matricization

of the Kruskal operator as:

$$^{(n)}\mathbf{J} = \mathbf{X}^{(n)}(\mathbf{Q}^{(n)})^{\top}, \tag{5}$$

with:

$$\mathbf{Q}^{(n)} = \mathbf{X}^{(N)} \odot \ldots \odot \mathbf{X}^{(n+1)} \odot \mathbf{X}^{(n-1)} \odot \ldots \odot \mathbf{X}^{(1)}, \tag{6}$$

where $\odot$ denotes the Khatri-Rao product [17].

Finally, we can express the vectorized version of Eq. (5) as:

$$\mathrm{vec}\left(^{(n)}\mathbf{J}\right) = \left[\mathbf{Q}^{(n)} \otimes \mathbf{I}_{I_n}\right]\mathrm{vec}(\mathbf{X}^{(n)}), \tag{7}$$

where $\otimes$ indicates the Kronecker product, and $\mathrm{vec}(\cdot)$ is a vectorization operator. It is worth noting that doing so, the vectorized form of the Kruskal operator is represented by a linear expression.

2. LOW-RANK DECONVOLUTION

We now derive the formulation of our approach. Let $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be a multidimensional signal. Our goal is to obtain a multidimensional convolutional representation $\mathcal{S} \approx \sum_m \mathcal{D}_m * \mathcal{K}_m$, where $\mathcal{D}_m \in \mathbb{R}^{L_1 \times L_2 \times \cdots \times L_N}$ acts as a dictionary, and $\mathcal{K}_m \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the activation map, is a low-rank factored tensor (*i.e.* a Kruskal tensor). If we write $\mathcal{K}_m = [\![\mathbf{X}_m^{(1)}, \mathbf{X}_m^{(2)}, \ldots, \mathbf{X}_m^{(N)}]\!]$ with $\mathbf{X}_m^{(n)} \in \mathbb{R}^{I_n \times R}$, in the context of CBPDN we can obtain a non-convex problem as:

$$\begin{aligned}\underset{\{\mathbf{X}_m^{(n)}\}}{\arg\min}\ & \frac{1}{2}\left\|\sum_{m=1}^{M} \mathcal{D}_m * [\![\mathbf{X}_m^{(1)}, \ldots, \mathbf{X}_m^{(N)}]\!] - \mathcal{S}\right\|_2^2 \\ & + \sum_{m=1}^{M}\sum_{n=1}^{N} \lambda \left\|\mathbf{X}_m^{(n)}\right\|_1,\end{aligned} \tag{8}$$

where $*$ indicates a N-dimensional convolution. Following standard strategies to resolve the PARAFAC decomposition [17], we propose to solve for each Kruskal factor (n) alternately. We denoted this strategy as Low-rank Deconvolution (LRD).

2.1. ADMM Algorithm

In order to make the optimization problem in Eq. (8) tractable, we rewrite it in a form suitable for Alternating Direction Method of Multipliers (ADMM) [19] by using a couple of auxiliary variables $\{\mathbf{Y}_m^{(n)}\}$ and $\{\mathbf{U}_m^{(n)}\}$ of the same size as $\{\mathbf{X}_m^{(n)}\}$, obtaining the optimization as:

$$\begin{aligned}\underset{\{\mathbf{X}_m^{(n)}\},\{\mathbf{Y}_m^{(n)}\}}{\arg\min}\ & \frac{1}{2}\left\|\sum_{m=1}^{M} \mathcal{D}_m * [\![\ldots, \mathbf{X}_m^{(n)}, \ldots]\!] - \mathcal{S}\right\|_2^2 \\ & + \sum_{m=1}^{M} \lambda \left\|\mathbf{Y}_m^{(n)}\right\|_1 \\ \text{subject to}\quad & \mathbf{X}_m^{(n)} = \mathbf{Y}_m^{(n)}\ \forall\, m\end{aligned} \tag{9}$$

while *not converged* **do**
  $\mathbf{X}_m^{(n)(k+1)} =$
  $\quad \arg\min \frac{1}{2}\left\|\sum_{m=1}^{M} \mathcal{D}_m * [\![\ldots, \mathbf{X}_m^{(n)}, \ldots]\!] - \mathcal{S}\right\|_2^2 +$
  $\quad \frac{\rho}{2}\sum_{m=1}^{M}\left\|\mathbf{X}_m^{(n)} - \mathbf{Y}_m^{(n)(k)} + \mathbf{U}_m^{(n)(k)}\right\|_2^2$
  $\mathbf{Y}_m^{(n)(k+1)} = \mathbf{prox}_{\mathbf{1}\frac{\lambda}{\rho}}(\mathbf{X}_m^{(n)(k+1)} + \mathbf{U}_m^{(k)})$
  $\mathbf{U}_m^{(n)(k+1)} = \mathbf{U}_m^{(n)(k)} + \mathbf{X}_m^{(n)(k+1)} - \mathbf{Y}_m^{(n)(k+1)}$
end
Not.: $\mathbf{prox}_{\mathbf{1}\gamma}(\mathbf{u}) = \mathrm{sign}(\mathbf{u}) \oplus \max(0, |\mathbf{u}| - \gamma)$.

Algorithm 1: ADMM algorithm for LRD for solving Eq. (9), considering a (n)-mode subproblem. **prox** is proximal operator to perform a shrinkage. $\mathrm{sign}(\cdot)$, $\max(\cdot)$ and $|\cdot|$ of a vector considered to be applied element-wise. $\oplus$ denotes the element-wise product. Step 2 is solved applying section 2.2.

The previous problem can be carried out efficiently by solving each Kruskal factor (n) alternately, updating one variable while fixing the others. Algorithm 1 explains the details. To initialize the auxiliary variables $\{\mathbf{Y}_m^{(n)}\}$ and $\{\mathbf{U}_m^{(n)}\}$ and the selection of the penalty coefficient ρ, we adopt the proposal in [20], with an adaptive strategy for the latter.

2.2. Formulation in the DFT Domain

As it was discussed in 1, the common practice to address CBPDN is to solve it in a DFT domain, achieving a solution both efficient and accurate. To this end, we denote by $\hat{\mathbf{A}}$ an arbitrary variable $\mathbf{A}$ in the DFT domain. Looking for a linear expression, let $\hat{\mathbf{D}}_m^{(n)} = \mathrm{diag}\big(\mathrm{vec}\big(^{(n)}\hat{\boldsymbol{D}}_m\big)\big) \in \mathbb{R}^{\Lambda I_n \times \Lambda I_n}$ be a linear operator for computing convolution, and $\hat{\mathbf{x}}_m^{(n)} = \mathrm{vec}(\hat{\mathbf{X}}_m^{(n)}) \in \mathbb{R}^{MRI_n}$ be the vectorized Kruskal factor. We can define now $\hat{\mathbf{Q}}_m^{(n)} = \hat{\mathbf{X}}_m^{(N)} \odot \cdots \odot \hat{\mathbf{X}}_m^{(n+1)} \odot \hat{\mathbf{X}}_m^{(n-1)} \odot \cdots \odot \hat{\mathbf{X}}_m^{(1)} \in \mathbb{R}^{\Lambda \times R}$, as it was done in Eq. (6), with Λ defined in section 1.1.

Assuming that boundary effects are negligible, i.e., relying on the use of filters of small spatial support L_n for $n = 1, \ldots, N$, we can formulate the problem in Alg. 1-step 2 in the DFT domain as:

$$\begin{aligned}\underset{\{\hat{\mathbf{x}}_m^{(n)}\}}{\arg\min}\ & \frac{1}{2}\left\|\sum_{m=1}^{M} \hat{\mathbf{D}}_m^{(n)}\left[\hat{\mathbf{Q}}_m^{(n)} \otimes \mathbf{I}_{I_n}\right]\hat{\mathbf{x}}_m^{(n)} - \hat{\mathbf{s}}^{(n)}\right\|_2^2 \\ & + \frac{\rho}{2}\sum_{m=1}^{M}\left\|\hat{\mathbf{x}}_m^{(n)} - \hat{\mathbf{z}}_m^{(n)}\right\|_2^2,\end{aligned} \tag{10}$$

where we have used a shortcut variable $\hat{\mathbf{z}}_m^{(n)} = \hat{\mathbf{y}}_m^{(n)} - \hat{\mathbf{u}}_m^{(n)}$ that encompasses vectorized versions of $\hat{\mathbf{Y}}_m^{(n)}$ and $\hat{\mathbf{U}}_m^{(n)}$, respectively. Moreover, $\hat{\mathbf{s}}^{(n)}$ is the vectorized version of $^{(n)}\hat{\mathbf{S}}$.

To solve the problem, we first define some supporting expressions:

$$\hat{\mathbf{W}}_m^{(n)} = \hat{\mathbf{D}}_m^{(n)}\big[\hat{\mathbf{Q}}_m^{(n)} \otimes \mathbf{I}_{I_n}\big], \tag{11}$$

$$\hat{\mathbf{W}}^{(n)} = \big[\hat{\mathbf{W}}_0^{(n)}, \hat{\mathbf{W}}_1^{(n)}, \ldots, \hat{\mathbf{W}}_M^{(n)}\big], \tag{12}$$

$$\hat{\mathbf{x}}^{(n)} = \big[(\hat{\mathbf{x}}_0^{(n)})^\top, (\hat{\mathbf{x}}_1^{(n)})^\top, \ldots, (\hat{\mathbf{x}}_M^{(n)})^\top\big]^\top, \tag{13}$$

$$\hat{\mathbf{z}}^{(n)} = \big[(\hat{\mathbf{z}}_0^{(n)})^\top, (\hat{\mathbf{z}}_1^{(n)})^\top, \ldots, (\hat{\mathbf{z}}_M^{(n)})^\top\big]^\top, \tag{14}$$

in order to finally transform the problem in Eq. (10) into:

$$\arg\min_{\hat{\mathbf{x}}^{(n)}} \frac{1}{2}\left\|\hat{\mathbf{W}}^{(n)}\hat{\mathbf{x}}^{(n)} - \hat{\mathbf{s}}^{(n)}\right\|_2^2 + \frac{\rho}{2}\left\|\hat{\mathbf{x}}^{(n)} - \hat{\mathbf{z}}^{(n)}\right\|_2^2. \tag{15}$$

Fortunately, Eq. (15) can be solved in closed form, by means of the next linear system:

$$\big[(\hat{\mathbf{W}}^{(n)})^H\hat{\mathbf{W}}^{(n)} + \rho\mathbf{I}_\beta\big]\hat{\mathbf{x}}^{(n)} = (\hat{\mathbf{W}}^{(n)})^H\hat{\mathbf{s}}^{(n)} + \rho\hat{\mathbf{z}}^{(n)}, \tag{16}$$

where $(\cdot)^H$ denotes a conjugate transpose matrix, and $\beta = MRI_n$. It is worth pointing out that the Kruskal tensor in Eq. (2) can be seen as a separable filter, and hence allows for the DFT transform to be computed independently for each factor.

2.2.1. *Choice of Regularization*

We suspect that in our case the sparsity regularization is not required as the low-rank constraint might be sufficient. One can observe that replacing the norm-1 term by a squared norm-2 term and λ by $\alpha/2$ in eq. (9), the solution is then given by:

$$\big[(\hat{\mathbf{W}}^{(n)})^H\hat{\mathbf{W}}^{(n)} + \alpha\mathbf{I}_\beta\big]\hat{\mathbf{x}}^{(n)} = (\hat{\mathbf{W}}^{(n)})^H\hat{\mathbf{s}}^{(n)}, \tag{17}$$

with β defined in section 2.2. Then, the full LRD method is summarized in algorithm 2. As it can be seen, we solve for every n-mode independently.

input : $\mathcal{S}$, $\{\mathcal{D}_m\}_{m=1}^M$, $\{\mathbf{X}_{0,m}^{(n)}\}_{n=1,m=1}^{N,M}$, $R > 0$
output: $\{\mathbf{X}_m^{(n)}\}_{n=1,m=1}^{N,M}$
/* Initialize Kruskal Factors */
1 $\{\mathbf{X}_m^{(n)}\} = \{\mathbf{X}_{0,m}^{(n)}\}$
/* Main Loop, Eq. (8) */
2 **while** *not converged* **do**
3 **for** $n = 1, \ldots, N$ **do**
4 $\mathbf{X}_m^{(n)} =$
5 $\arg\min \frac{1}{2}\left\|\sum_{m=1}^M \mathcal{D}_m * [\![\mathbf{X}_m^{(1)}, \ldots, \mathbf{X}_m^{(N)}]\!] - \mathcal{S}\right\|_2^2 +$
6 $\Phi(\{\mathbf{X}_m^{(n)}\})$
7 **end**
end

Algorithm 2: LRD algorithm. Here $\Phi(\cdot)$ refers to the choice of regularization. If norm-1 is choosen, step 4 is solved by using Alg. 1, else if norm-2 is choosen, step 4 is performed by directly solving the linear system in eq. (17). The full algorithm solves the LRD problem by means of an alternated approach for every n-mode.

2.3. Linear Mask Decoupling for Tensor Completion

We continue by applying LRD to tensor completion problems. To do so, we are required to formulate the optimization problem masking out the unknowns which are given in the spatial domain. This requires us to include the DFT transform in our formulation and to consider the spatial version of the signal $\mathbf{s}^{(n)}$:

$$\arg\min_{\hat{\mathbf{x}}^{(n)}} \frac{1}{2}\left\|\mathbf{P}^{(n)}\hat{\mathbf{F}}^{(n)}\hat{\mathbf{W}}^{(n)}\hat{\mathbf{x}}^{(n)} - \mathbf{s}^{(n)}\right\|_2^2 + \frac{\alpha}{2}\left\|\hat{\mathbf{x}}^{(n)}\right\|_2^2. \tag{18}$$

Here, $\hat{\mathbf{F}}^{(n)} = \hat{\mathbf{F}}_N \otimes \cdots \otimes \hat{\mathbf{F}}_{n+1} \otimes \hat{\mathbf{F}}_{n-1} \otimes \cdots \otimes \hat{\mathbf{F}}_1 \otimes \hat{\mathbf{F}}_n$ is the matricization of the multilinear DFT inverse transform being $\hat{\mathbf{F}}_i$ the inverse transform for mode-i and $\mathbf{P}^{(n)}$ the mask matrix, which is diagonal for a tensor completion problem. Then, its solution is given by,

$$\big[(\hat{\mathbf{T}}^{(n)})^H\hat{\mathbf{T}}^{(n)} + \alpha\mathbf{I}_\beta\big]\hat{\mathbf{x}}^{(n)} = (\hat{\mathbf{T}}^{(n)})^H\mathbf{s}^{(n)}, \tag{19}$$

with $\hat{\mathbf{T}}^{(n)} = \mathbf{P}^{(n)}\hat{\mathbf{F}}^{(n)}\hat{\mathbf{W}}^{(n)}$ and β defined in section 2.2. One can see that algorithm 2 can be easily extended to consider such approach.

Fig. 1. **Qualitative evaluation on RGB Basketball video.** In all cases, we show ten consecutive video frames. **Top.** Ground truth color frames. **Middle.** Color Video reconstruction using [20]. **Bottom.** Our solution. As it can be seen, our method provides a more visually correct solution than [20] that includes a wide variety of artifacts. Best viewed in color.

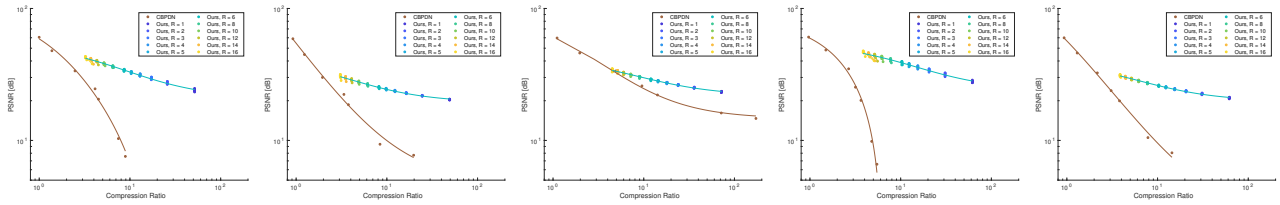


Fig. 2. **Quality of reconstruction (PSNR) vs. compression rate (CR).** We display results on the videos Basketball, Football1, Ironman, Skiing and Soccer, respectively. PSNR evolution as a function of CR for CBPDN [20] and our approach for different R values. Best viewed in color.

3. EXPERIMENTS

3.1. Compressed Video Reconstruction

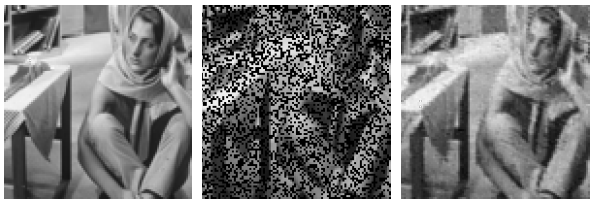
We select five color sequences of the OTB50 dataset [21], denoted as *Basketball*, *Football1*, *Ironman*, *Soccer* and *Skiing*. For every video, we consider the first 78 frames, with a minimum resolution of $[30 \times 30]$ pixels, and we represent it by means of a 3-order tensor with 3 channels. Dictionary dimensions are chosen as $\mathcal{D}_{m,c} \in \mathbb{R}^{L_1 \times L_2 \times L_3}$ for $m = \{1, \ldots, M\}$ and $c = \{1, \ldots, C\}$, with $\{L_n = 5\}_{n=1}^{3}$, $M = 25$ and $C = 3$. We split every sequence in two: we use the first half for learning the filters applying the algorithm from [22], and the second half for testing.

To show the reconstruction quality we use a Peak Signal-to-Noise Ratio (PSNR), together with a Compression Ratio (CR) defined as $CR = \Pi_{n=1}^{N} {}^{I_n}/_{N_{NZ}}$ where $N_{NZ} = \sum_m \|\mathbf{X}_m\|_1$ for the CBPDN approach [20] and $N_{NZ} = \sum_m \sum_n \left\|\mathbf{X}_m^{(n)}\right\|_1$ for our method. This measure is equivalent to the inverse of the number of features relative to the size of the signal which states the efficiency of the representation. Additionally, we also report results as a function of λ for all methods, and evaluate for different rank values R in our case.

Figure 2 shows that our approach obtains a good reconstruction even for high levels of compression. While CBPDN [20] obtains good results for small compression, quickly dropping as more sparsity is demanded (by increasing λ). Recall that the low-rank factorization is an important source of compression as according to section 1.1 a rank-R tensor is equivalent to the sum of R rank-1 tensors, which in turn can be expressed as $\sum_n I_n$ instead of $\prod_n I_n$ values. Finally, we present a qualitative comparison on the Basketball sequence in Fig. 1 for a similar $CR \approx 9$, the obtained accuracy is $PSNR = 10.79$ and $PSNR = 28.32$ for the CBPDN [20] and our method, respectively. Again, we can observe how our estimation is visually more accurate than that provided by competing techniques.

3.2. Image In-painting

For this problem, we consider ten grey-scale images from [23] resized to $[100 \times 100]$ pixels, named *Barbara*, *Boat*, *Cameraman*, *Couple*, *Fingerprint*, *Hill*, *House*, *Man*, *Montage* and *Peppers*. For each of them we mask out a random sample of the pixels with a proportion of $\{30\%, 50\%, 60\%\}$ relative to the total number of pixels of the image and we use the method from section 2.3 to recover the original signal, with parameters set to $\{R = 3, M = 15, \alpha = 10^{-4}\}$ and filters learned on the city and fruit datasets from [1]. Results are presented in Table 1. Both the qualitative and quantitative results show that our method is capable of recovering the original signal even for an important number of missing entries.



Missing \ Image	Barbara	Boat	C.Man	Couple	F.Print	Hill	House	Man	Montage	Peppers	Av.
30%	26.80	23.64	26.96	24.29	20.20	25.86	30.28	22.17	27.74	23.13	25.11
50%	23.76	23.10	24.83	22.84	18.17	23.41	27.32	21.36	23.33	20.76	22.89
60%	22.48	22.54	24.35	22.25	17.60	22.93	25.52	20.62	22.76	20.39	22.14

Table 1. **Qualitative and quantitative evaluation on image in-painting. Top:** From left to right, we display ground truth, input and result for the Barbara image for a 50% missing pixels. **Bottom:** The table reports the PSNR in dB (higher is better) using our approach for 10 images. We indicate the solution for a missing pixel rate of $\{30\%, 50\%, 60\%\}$.

4. CONCLUSION

LRD is a powerful framework that provides a sufficient prior to learn the latent structure of data in multidimensional settings. The results obtained regarding the compressed video reconstruction verify our claims. Moreover its formulation is flexible enough to deal with incomplete data allowing its application in tensor completion problems as we verified with the experiments regarding image in-painting. As a future work would be interesting to evaluate how this approach deals with increasing data dimensions and large datasets, as in this situations data compression is of an important matter.

5. REFERENCES

[1] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *CVPR*, 2010, pp. 2528–2535.

[2] K. Kavukcuoglu, P. Sermanet, Y. L. Boureau, K. Gregor, M. Mathieu, and Y. L. Cun, "Learning convolutional feature hierarchies for visual recognition," in *NIPS*, 2010, pp. 1090–1098.

[3] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning.," in *ICCV*, 2011, vol. 1, p. 6.

[4] F. Heide, W. Heidrich, and G. Wetzstein, "Fast and flexible convolutional sparse coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5135–5143.

[5] Shuhang Gu, Wangmeng Zuo, Qi Xie, Deyu Meng, Xiangchu Feng, and Lei Zhang, "Convolutional sparse coding for image super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1823–1831.

[6] V. Papyan, Y. Romano, J. Sulam, and M. Elad, "Convolutional dictionary learning via local processing," in *ICCV*, 2017, pp. 5296–5304.

[7] Hui Ji, Chaoqiang Liu, Zuowei Shen, and Yuhong Xu, "Robust video denoising using low rank matrix completion," in *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 1791–1798.

[8] Emmanuel Candes and Benjamin Recht, "Exact matrix completion via convex optimization," *Communications of the ACM*, vol. 55, no. 6, pp. 111–119, 2012.

[9] Emmanuel J Candes and Yaniv Plan, "Matrix completion with noise," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

[10] Zhen Long, Yipeng Liu, Longxi Chen, and Ce Zhu, "Low rank tensor completion for multiway visual data," *Signal processing*, vol. 155, pp. 301–316, 2019.

[11] Lefei Zhang, Liangchen Song, Bo Du, and Yipeng Zhang, "Nonlocal low-rank tensor completion for visual data," *IEEE transactions on cybernetics*, vol. 51, no. 2, pp. 673–685, 2019.

[12] Changxiao Cai, Gen Li, H Vincent Poor, and Yuxin Chen, "Nonconvex low-rank tensor completion from noisy data," *Advances in neural information processing systems*, vol. 32, 2019.

[13] Pierre Humbert, Julien Audiffren, Laurent Oudre, and Nicolas Vayatis, "Low rank activations for tensor-based convolutional sparse coding," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3252–3256.

[14] Pierre Humbert, Laurent Oudre, Nicolas Vayatis, and Julien Audiffren, "Tensor convolutional dictionary learning with cp low-rank activations," *IEEE Transactions on Signal Processing*, vol. 70, pp. 785–796, 2021.

[15] R. A. Harshman et al., "Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis," 1970.

[16] J. D. Carroll and J. J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[17] T. G. Kolda, "Multilinear operators for higher-order decompositions.," Tech. Rep., Sandia National Laboratories, 2006.

[18] B. W. Bader and T. G. Kolda, "Algorithm 862: MATLAB tensor classes for fast algorithm prototyping," *TOMS*, vol. 32, no. 4, pp. 635–653, 2006.

[19] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[20] B. Wohlberg, "SPORCO: A python package for standard and convolutional sparse representations," in *Proceedings of the 15th Python in Science Conference, Austin, TX, USA*, 2017, pp. 1–8.

[21] Y. Wu, J. Lim, and M. H. Yang, "Online object tracking: A benchmark," in *CVPR*, 2013, pp. 2411–2418.

[22] C. Garcia-Cardona and B. Wohlberg, "Convolutional dictionary learning: A comparative review and new algorithms," *TCI*, vol. 4, no. 3, pp. 366–381, 2018.

[23] Shuhang Gu, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng, "Weighted nuclear norm minimization with application to image denoising," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 2862–2869.