FAST UNSUPERVISED TENSOR RESTORATION VIA LOW-RANK DECONVOLUTION

David Reixach, Josep Ramon Morros

Universitat Politècnica de Catalunya, BarcelonaTech, Spain

ABSTRACT

Low-rank Deconvolution (LRD) has appeared as a new multi-dimensional representation model that enjoys important efficiency and flexibility properties. In this work we ask ourselves if this analytical model can compete against Deep Learning (DL) frameworks like Deep Image Prior (DIP) or Blind-Spot Networks (BSN) and other classical methods in the task of signal restoration. More specifically, we propose to extend LRD with differential regularization. This approach allows us to easily incorporate Total Variation (TV) and integral priors to the formulation leading to considerable performance tested on signal restoration tasks such image denoising and video enhancement, and at the same time benefiting from its small computational cost.

***Index Terms*—** Tensors, Restoration, Total Variation, Denoising, Enhancement

1. INTRODUCTION

The signal reconstruction problem generally relies on constraining the solution to be consistent with some prior knowledge. Traditional imaging priors include non-negativity, sparsity or self-similarity among others [1–5]. More recently, the field has strongly been influenced by Deep Learning (DL), which offers a wide range of models to be considered as priors but mostly limited to imaging problems [6–10].

Recently, Low-rank Deconvolution (LRD) [11] has been introduced as a new model for compressed tensor representation and completion with promising results. However, as we show in this work, LRD lacks noise rejection abilities. To address this issue we propose to combine LRD together with a regularization function to solve tensor restoration problems. More specifically, by advantaging of the DFT formulation of LRD we propose to use this framework with implicit differential regularization in the DFT domain. This approach allows us to easily incorporate squared Total Variation (TV) and integral regularization into consideration by just solving a linear expression.

TV has been used as a prior in the past, alone [12] or together with other priors such Deep Image Prior (DIP) [13], both achieving state-of-the-art results in the task of image denoising. In this work we demonstrate how our combination of LRD and TV obtains remarkable performance on image denoising problems, benefiting also from being much lighter computationally than some DL and other classical denoising methods, and also extending to multi-dimensional data like tensors. As a by-product, we also show how this method can be used for video enhancement tasks. The main contributions of this work are as follows:

- A novel method is proposed that extends the existing LRD framework with a new regularization function. It allows for tensor restoration tasks such denoising and detail enhancement in a fully unsupervised manner.
- A theoretical analysis is performed which allows incorporating the regularization function into the LRD framework by simply solving a linear expression in the DFT domain, allowing for efficient computation.
- The proposed method outperforms most of non-supervised state-of-the-art methods in both PSNR and execution time on image denoising. We also show that TV regularization based methods are still very competitive.

This work has been partially supported by the Spanish Ministry of Science and Innovation under the project DeeLight PID2020-117142GB-I00.

2. RELATED WORK

2.1. Image Denoising

Image denoising is a low-level computer vision topic that has been studied for decades and still remains open. It aims to obtain higher quality images from its corresponding noisy versions. Earlier approaches were based on modelling the signal or the noise and by formulating an optimization problem that tried to separate the two components. TV minimization [1,12] is a typical example of these approaches.

Following this idea, different solutions appeared that relied on signal decomposition in a sparse domain and filtering out the corresponding low-energy components [2]. Then the concept of image Non-local Self Similarity (NSS) appeared and boosted significantly image denoising performance, a prior assumption that is used by BM3D [3], EPLL [14] and WNNM [15] for example. A common thing that is shared by these classical methods in general is the fact that they are fully unsupervised, *i.e.*, they only require a single noisy image to obtain its corresponding clean pair. Even so, these methods are computationally expensive, as NSS involves fragmenting the input image into different patches and performing exhaustive analysis between them.

During the last decade, many approaches based on DL have emerged. Initially, the vast majority of the methods [16] were based on fully supervised processes, *i.e.*, requiring paired clean-noise image for training. However, to obtain such datasets that cover the whole range of noise possibilities is impractical. To get rid of this dependency, unsupervised methods based on DL priors such DIP [6] appeared. This method is able to learn natural image priors from large-scale datasets and requires only a single noisy image to obtain its corresponding clean pair. Similarly, Noise2Fast (N2F) [7] trains on a discrete set of clean images and does not require any prior knowledge of the noise distribution. However, these methods rely on solving a DL based optimization problem for each sample, which makes them computationally expensive too.

More recently, a new category of methods has arisen, namely DL self-supervised. These methods do not require paired clean-noisy images, as they are able to learn from strictly noisy samples. Many of these approaches rely on a Blind-Spot Network (BSN) although in general they can only be applied under the assumption of pixel-wise independent noise which is unrealistic for a real world scenario. To tackle this issue, AP-BSN [8] propose asymmetric Pixel-shuffle Downsampling (PD) factors and post-refinement processing to make a better trade-off between noise removal and aliasing artifacts. SDAP [10] propose a new BSN framework in combination with a Random Sub-samples Generation (RSG) strategy that tries to break the pixel-wise independent noise assumption. Similarly, SASS [9] propose a new strategy to break such assumption by taking into account the respective characteristics of flat and textured regions in noisy images, and construct supervisions for them separately. However, self-supervised methods usually still require a large number of noisy images and in general they do not report good results when trained with scarce data.

2.2. Detail Enhancement

The goal of detail enhancement is to improve the visual appearance of images by increasing local contrast or amplifying details. Most existing techniques achieve it by modifying a decomposed version of the image. Such decomposition is typically achieved by filtering or optimization.

To that end, TV has been used in the past with great results [5]. Their idea is to decompose the image between two components: texture and cartoon. Texture is modelled as the signal which incorporates the main TV component, while cartoon holds the rest. Following this concept, our method, which is capable of such decomposition, eases the task of video enhancement (detail enhancement applied to tensors).

3. LOW-RANK DECONVOLUTION WITH DIFFERENTIAL REGULARIZATION

3.1. Notation

Let $\mathfrak{J} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ be a N-order tensor. The PARAFAC [17] decomposition (a.k.a. CANDECOMP [18]) is defined as:

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

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 $\mathfrak{J}$ 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 [19] 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 a special case of matricization known as n-mode matricization [19, 20]. 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 a tensor $\mathfrak{K}$ 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 $\mathfrak{K}$ to those of ${}^{(n)}\mathbf{K}$, defined as:

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

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 $\mathfrak{K} = [\![\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(N)}]\!]$, by following [19] we can obtain the n-mode matricization of the Kruskal operator as:

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

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.

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

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

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

3.2. Revisiting LRD

We recall the formulation of LRD [11]. Let $\mathcal{S} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ be a multi-dimensional signal. Our goal is to obtain a multi-dimensional 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 $\mathbf{X}_m^{(n)} \in \mathbb{R}^{I_n \times R}$, we can obtain a non-convex problem as:

$$\underset{\{\mathbf{X}_m^{(n)}\}}{\arg\min} \frac{1}{2} \left\| \sum_{m=1}^{M} \mathcal{D}_m * [\![\mathbf{X}_m^{(1)}, \ldots, \mathbf{X}_m^{(N)}]\!] - \mathcal{S} \right\|_2^2 + \Phi(\{\hat{\mathbf{X}}_m^{(n)}\}), \quad (8)$$

where $*$ indicates a N-dimensional convolution and $\Phi(\{\hat{\mathbf{X}}_m^{(n)}\})$ is a regularization term. According to [11] we propose to solve the problem for each Kruskal factor (n) alternately. Algorithm 1 provides with more details.

Following [11], we solve the LRD optimization problem in a DFT domain assuming that boundary effects are negligible (*i.e*. relying on the use of filters of small spatial support). To this end, we denote by $\hat{\mathbf{A}}$ an arbitrary variable $\mathbf{A}$ in the DFT domain. Let $\hat{\mathbf{D}}_m^{(n)} = \text{diag}\big(\text{vec}\big({}^{(n)}\hat{\boldsymbol{D}}_m\big)\big) \in \mathbb{R}^{\Lambda I_n \times \Lambda I_n}$ be a linear operator for computing convolution, and $\hat{\mathbf{x}}_m^{(n)} = \text{vec}(\hat{\mathbf{X}}_m^{(n)}) \in \mathbb{R}^{RI_n}$ be the vectorized Kruskal factor. We define $\hat{\mathbf{Q}}_m^{(n)} = \hat{\mathbf{X}}_m^{(N)} \odot \cdots \odot \hat{\mathbf{X}}_m^{(n+1)} \odot \hat{\mathbf{X}}_m^{(n-1)} \odot \cdots \odot \hat{\mathbf{X}}_m^{(1)} \in \mathbb{R}^{\Lambda \times R}$, as it was done in Eq. (6), with Λ defined in section 3.1. Then, by using Eq. (7) we define:

$$\hat{\mathbf{W}}_m^{(n)} = \hat{\mathbf{D}}_m^{(n)} \big[\hat{\mathbf{Q}}_m^{(n)} \otimes \mathbf{I}_{I_n}\big], \quad (9)$$

$$\hat{\mathbf{W}}^{(n)} = \big[\hat{\mathbf{W}}_0^{(n)}, \hat{\mathbf{W}}_1^{(n)}, \ldots, \hat{\mathbf{W}}_M^{(n)}\big], \quad (10)$$

$$\hat{\mathbf{x}}^{(n)} = \big[(\hat{\mathbf{x}}_0^{(n)})^\top, (\hat{\mathbf{x}}_1^{(n)})^\top, \ldots, (\hat{\mathbf{x}}_M^{(n)})^\top\big]^\top, \quad (11)$$

All these algebraic modifications together with $\hat{\mathbf{s}}^{(n)} = \text{vec}({}^{(n)}\hat{\mathbf{S}})$ allow us to transform the problem (8) into (12). And the solution will depend on the regularizer chosen $(\Phi(\{\hat{\mathbf{x}}_m^{(n)}\}))$:

$$\underset{\hat{\mathbf{x}}^{(n)}}{\arg\min} \frac{1}{2} \left\| \hat{\mathbf{W}}^{(n)} \hat{\mathbf{x}}^{(n)} - \hat{\mathbf{s}}^{(n)} \right\|_2^2 + \Phi(\{\hat{\mathbf{x}}_m^{(n)}\}). \quad (12)$$

3.3. Differential Regularization in the DFT Domain

Following the approach by [1], a squared Total Variation and an integral regularizer can be added to the global problem in

```
     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         $\mathbf{X}_m^{(n)} =$
             $\arg\min \frac{1}{2} \left\| \sum_{m=1}^M \mathcal{D}_m * [\![\mathbf{X}_m^{(1)}, \ldots, \mathbf{X}_m^{(N)}]\!] - \mathcal{S} \right\|_2^2 +$
             $\Phi(\{\mathbf{X}_m^{(n)}\})$
   5     end
   6 end
```

Algorithm 1: LRD algorithm solves the LRD problem by means of an alternated approach for every n-mode.

the following manner:

$$\underset{\{\mathbf{X}_m^{(n)}\}, \mathcal{U}}{\arg\min} \frac{1}{2} \|\mathcal{U} - \mathcal{S}\|_2^2 + \frac{\gamma}{2} \|\mathcal{U}\|_{TV}^2 + \frac{\zeta}{2} \|\mathcal{U}\|_{TI}^2 + \Psi(\{\mathbf{X}_m^{(n)}\}).$$

$$\text{subject to} \quad \mathcal{U} = \sum_{m=1}^{M} \mathcal{D}_m * [\![\mathbf{X}_m^{(1)}, \ldots, \mathbf{X}_m^{(N)}]\!] \quad (13)$$

Where γ, ζ are parameters, and $\Psi(\{\mathbf{X}_m^{(n)}\})$ is a regularization term. Then, with a little algebraic manipulation, and making use of the derivative and integral properties of the DFT, we can present the following proposition:

Proposition 3.1. *The problem presented in eq. (13) with* $\Psi(\{\mathbf{X}_m^{(n)}\}) = \sum_{m=1}^M \sum_{n=1}^N \frac{\alpha}{2} \left\|\mathbf{X}_m^{(n)}\right\|_2^2$ *has a solution given by a linear expression in the DFT domain given by:*

$$\big[(\hat{\mathbf{W}}^{(n)})^H \hat{\mathbf{W}}^{(n)} + \gamma(\hat{\Theta}^{(n)})^H \hat{\Theta}^{(n)} + \zeta(\hat{\Omega}^{(n)})^H \hat{\Omega}^{(n)} + \alpha \mathbf{I}_\beta\big] \hat{\mathbf{x}}^{(n)} = (\hat{\mathbf{W}}^{(n)})^H \hat{\mathbf{s}}^{(n)}, \quad (14)$$

where we have made use of $\hat{\mathbf{s}}^{(n)}$, $\hat{\mathbf{x}}^{(n)}$ *and* $\hat{\mathbf{W}}^{(n)}$ *defined in section 3.2. And defining:*

$$(\hat{\Theta}_i^{(n)})^T = 2\pi j \xi_i \oplus \hat{\mathbf{W}}^{(n)} \quad (15)$$

$$(\hat{\Omega}_i^{(n)})^T = (2\pi j \xi_i)^{-1} \oplus \hat{\mathbf{W}}^{(n)}, \quad (16)$$

$$\hat{\Theta} = \big[\hat{\Theta}_0^{(n)}, \hat{\Theta}_1^{(n)}, \ldots, \hat{\Theta}_N^{(n)}\big], \quad (17)$$

$$\hat{\Omega} = \big[\hat{\Omega}_0^{(n)}, \hat{\Omega}_1^{(n)}, \ldots, \hat{\Omega}_N^{(n)}\big] \quad (18)$$

with ξ_i *being the vector of frequencies for the* i*-dimension, and* $\oplus$ *denoting element-wise product. The problem is equivalent to the problem of eq. (8) with:*

$$\Phi(\{\mathbf{x}_m^{(n)}\}) = \frac{\gamma}{2} \left\| (\hat{\Theta}^{(n)})^T \hat{\mathbf{x}}^{(n)} \right\|_2^2 + \frac{\zeta}{2} \left\| (\hat{\Omega}^{(n)})^T \hat{\mathbf{x}}^{(n)} \right\|_2^2 + \Psi(\{\mathbf{X}_m^{(n)}\}). \quad (19)$$

Proof. The squared total variation regularization is given by

$$\frac{\gamma}{2}\left\|\mathbf{\mathcal{U}}\right\|_{TV}^{2}=\frac{\gamma}{2}\left\|\nabla\mathbf{\mathcal{U}}\right\|_{2}^{2}, \tag{20}$$

where,

$$\frac{\gamma}{2}\left\|\nabla\mathbf{\mathcal{U}}\right\|_{2}^{2}= \tag{21}$$
$$\frac{\gamma}{2}\left\|\left[\left(\frac{\partial\mathbf{u}^{(n)}}{\partial\mathbf{t}_0}\right)^T,\left(\frac{\partial\mathbf{u}^{(n)}}{\partial\mathbf{t}_1}\right)^T,\dots,\left(\frac{\partial\mathbf{u}^{(n)}}{\partial\mathbf{t}_N}\right)^T\right]^T\right\|_2^2.$$

By using the derivative property of the DFT transform,

$$\begin{aligned}\mathcal{F}\left\{\frac{\partial\mathbf{u}^{(n)}}{\partial\mathbf{t}_i}\right\}&=2\pi j\xi_i\oplus\mathcal{F}\{\mathbf{u}^{(n)}\}\\&=2\pi j\xi_i\oplus\hat{\mathbf{W}}^{(n)}\hat{\mathbf{x}}^{(n)},\end{aligned} \tag{22}$$

with ξ_i and $\oplus$ defined in section 3.2. Together with the definition of $\hat{\Theta}^{(n)}$ leads us to the following expression:

$$\mathcal{F}\left\{\frac{\gamma}{2}\left\|\sum_{m=1}^{M}\mathbf{\mathfrak{D}}_m*[\![\mathbf{X}_m^{(1)},\dots,\mathbf{X}_m^{(N)}]\!]\right\|_{TV}^2\right\}=$$
$$\frac{\gamma}{2}\left\|(\hat{\Theta}^{(n)})^T\hat{\mathbf{x}}^{(n)}\right\|_2^2, \tag{23}$$

which can be derived in the following manner (complex derivative),

$$\frac{\partial\frac{\gamma}{2}\left\|(\hat{\Theta}^{(n)})^T\hat{\mathbf{x}}^{(n)}\right\|_2^2}{\partial(\hat{\mathbf{x}}^{(n)})^H}=\frac{\gamma}{2}(\hat{\Theta}^{(n)})^H\hat{\Theta}^{(n)}. \tag{24}$$

The result above gives the solution for the derivative component. The proof for the squared integral component follows in the same manner and is given by:

$$\frac{\zeta}{2}\left\|\mathbf{\mathcal{U}}\right\|_{TI}^{2}= \tag{25}$$
$$\frac{\zeta}{2}\left\|\left[\left(\int\mathbf{u}^{(n)}\,d\mathbf{t}_0\right)^T,\dots,\left(\int\mathbf{u}^{(n)}\,d\mathbf{t}_N\right)^T\right]^T\right\|_2^2.$$

By using the integration property of the DFT transform,

$$\begin{aligned}\mathcal{F}\left\{\int\mathbf{u}^{(n)}\,d\mathbf{t}_i\right\}&=(2\pi j\xi_i)^{-1}\oplus\mathcal{F}\{\mathbf{u}^{(n)}\}\\&=(2\pi j\xi_i)^{-1}\oplus\hat{\mathbf{W}}^{(n)}\hat{\mathbf{x}}^{(n)},\end{aligned} \tag{26}$$

with ξ_i and $\oplus$ defined in section 3.2. Together with the definition of $\hat{\Omega}^{(n)}$ leads us to the following expression:

$$\mathcal{F}\left\{\frac{\zeta}{2}\left\|\sum_{m=1}^{M}\mathbf{\mathfrak{D}}_m*[\![\mathbf{X}_m^{(1)},\dots,\mathbf{X}_m^{(N)}]\!]\right\|_{TI}^2\right\}=$$
$$\frac{\zeta}{2}\left\|(\hat{\Omega}^{(n)})^T\hat{\mathbf{x}}^{(n)}\right\|_2^2, \tag{27}$$

which can be derived in the following manner (complex derivative),

$$\frac{\partial\frac{\zeta}{2}\left\|(\hat{\Omega}^{(n)})^T\hat{\mathbf{x}}^{(n)}\right\|_2^2}{\partial(\hat{\mathbf{x}}^{(n)})^H}=\frac{\zeta}{2}(\hat{\Omega}^{(n)})^H\hat{\Omega}^{(n)}. \tag{28}$$

The result above together with eq. (24) combined with the solution of eq. (8), brings us to eq. (14).

□

3.4. Image Denoising & Detail Enhancement

The application to image denoising as a TV regularizer is straight-forward as it only involves solving (13) with a proper selection for γ and α, while ζ is left to 0.0. Regarding the application to detail enhancement, let us reformulate (13) into the following expression:

$$\begin{aligned}\underset{\{\mathbf{X}_m^{(n)}\},\{\mathbf{\mathcal{U}}_m\}}{\arg\min}\ &\frac{1}{2}\left\|\sum_{m=1}^{M}\mathbf{\mathcal{U}}_m-\mathbf{\mathcal{S}}\right\|_2^2+\sum_{m=1}^{M}\frac{\gamma_m}{2}\left\|\mathbf{\mathcal{U}}_m\right\|_{TV}^2\\&+\sum_{m=1}^{M}\frac{\zeta_m}{2}\left\|\mathbf{\mathcal{U}}_m\right\|_{TI}^2+\Psi(\{\mathbf{X}_m^{(n)}\}),\\\text{subject to}\quad&\mathbf{\mathcal{U}}_m=\mathbf{\mathfrak{D}}_m*[\![\mathbf{X}_m^{(1)},\dots,\mathbf{X}_m^{(N)}]\!]\end{aligned} \tag{29}$$

which correspond to the same equation but with explicit signal $\mathbf{\mathcal{U}}_m$ reconstruction for each filter m. The idea of enhancement is to solve (29) and reconstruct the enhanced signal as $\tilde{\mathbf{\mathcal{U}}}=\sum_{m=1}^{M}\delta_m\mathbf{\mathfrak{D}}_m*[\![\mathbf{X}_m^{(1)},\dots,\mathbf{X}_m^{(N)}]\!]+\mathbf{\mathcal{S}}$ with a proper set of parameters $\{\gamma_m\}_{m=1}^{M}$, $\{\zeta_m\}_{m=1}^{M}$, $\{\delta_m\}_{m=1}^{M}$ and α.

4. EXPERIMENTS

4.1. Image Denoising

In this section, we analyse the performance of LRD-TV for image denoising problems, and we compare it to the state of the art. To that end, we select twelve gray-scale images from [13] resized to $[256\times256]$ pixels presented in Table 1. The methods chosen to compare to our approach are the unsupervised methods BM3D [3], EPLL [14], TV [12], WNNM [15], DIP [6] and LRD [11], and the self-supervised N2F [7], AP-BSN [8], SDAP [10] and SASS [9]. For all the methods we use authors' code and tune their parameters to achieve the best result. For the self-supervised methods, we train them with the SIDD-Small dataset for our specific noise level and use it for the evaluation together with their supplied pre-trained models, keeping the best of each for everyone of them. We consider a range of Additive White Gaussian Noise (AWGN) with $\sigma=\{30,50,70,90\}$ and for each we compute the input signal quality as input PSNR (presented in the table). All experiments are carried out in a desktop computer using an INTEL® CORE™ i7-12700K and an NVIDIA® GeForce™ RTX 3090 Ti.

Images	1	2	3	4	5	6	7	8	9	10	11	12
Input PSNR = 15.36 dB / $\sigma = 30$												
BM3D	17.96	22.59	24.68	18.64	21.32	22.86	22.59	18.58	22.76	22.48	23.14	19.37
EPLL	20.76	24.36	24.67	**21.80**	24.13	23.84	22.38	20.98	24.34	22.72	24.33	20.52
TV	21.33	**27.81**	**26.47**	19.99	23.58	25.49	**24.48**	**21.39**	**26.66**	**25.34**	24.10	19.96
WNNM	21.52	**29.67**	**28.34**	**23.54**	**26.12**	**28.65**	**26.44**	**24.30**	**26.29**	**24.91**	**24.73**	**25.78**
DIP	**25.40**	24.78	23.53	18.91	19.16	20.30	20.08	13.46	18.15	18.11	19.09	16.66
N2F	20.19	17.72	19.59	17.78	15.78	16.79	17.82	15.65	13.55	10.74	18.22	16.62
AP-BSN	19.51	21.84	21.72	17.62	22.13	23.80	20.13	15.15	22.02	20.37	22.11	17.52
SDAP	15.17	17.96	10.75	16.78	16.52	13.00	15.66	16.68	17.66	17.65	14.26	16.65
SASS	17.82	15.99	18.66	15.76	18.00	19.60	17.46	11.04	17.80	16.98	18.10	13.74
LRD	16.86	15.69	18.53	16.55	17.95	18.61	18.10	13.09	18.48	17.43	18.60	16.11
LRD-TV (Ours)	**21.56**	25.57	23.00	20.56	**24.69**	**25.68**	23.74	20.33	25.80	24.73	**24.42**	**22.13**
Input PSNR = 12.18 dB / $\sigma = 50$												
BM3D	15.69	15.32	21.00	16.57	18.47	19.68	19.44	14.28	19.20	17.53	20.35	15.51
EPLL	19.17	21.50	20.68	18.52	20.48	19.81	18.75	18.52	20.30	20.02	19.93	18.82
TV	**20.41**	24.23	21.30	18.17	**23.41**	**23.80**	**22.25**	**18.86**	**23.48**	**22.69**	**21.32**	19.45
WNNM	**21.45**	**26.92**	**22.09**	**20.86**	21.26	**25.63**	**21.99**	**21.92**	22.41	21.73	18.00	**24.45**
DIP	12.62	20.82	19.56	17.38	20.88	22.10	20.13	15.97	19.72	19.27	19.35	16.79
N2F	17.83	13.71	15.17	12.76	13.19	13.28	14.04	12.99	12.12	8.52	14.45	13.99
AP-BSN	17.53	16.42	17.11	14.69	16.23	17.67	16.08	11.13	15.81	16.35	15.23	14.65
SDAP	12.47	14.24	8.37	11.95	13.34	11.50	13.25	12.93	13.80	15.96	12.16	12.69
SASS	15.78	13.63	14.83	13.49	14.19	15.46	14.01	8.95	13.83	14.55	13.08	12.39
LRD	14.08	13.21	14.49	13.68	13.75	14.91	14.35	9.00	13.61	14.10	14.17	12.98
LRD-TV (Ours)	19.60	**24.35**	**22.07**	**19.16**	**23.32**	23.75	**21.99**	17.87	**23.77**	**22.73**	**23.07**	**21.04**
Input PSNR = 10.49 dB / $\sigma = 70$												
BM3D	14.84	14.64	18.11	14.50	16.21	16.63	15.96	11.63	17.73	16.73	17.53	14.00
EPLL	17.32	19.37	17.92	16.82	18.97	18.46	16.38	16.90	18.04	17.30	18.62	17.65
TV	**19.12**	**24.87**	**20.16**	17.88	**22.91**	**23.37**	**20.82**	**19.72**	**22.86**	**21.66**	**20.43**	19.69
WNNM	15.02	21.82	15.51	**19.06**	18.65	**23.50**	**20.91**	**20.78**	19.11	19.46	16.49	**22.99**
DIP	13.19	19.74	19.39	17.08	19.47	21.16	18.41	14.16	18.31	17.52	16.87	15.40
N2F	16.47	12.13	14.03	11.11	12.28	10.90	12.05	10.76	10.96	7.35	11.76	10.88
AP-BSN	16.03	12.78	13.18	13.17	13.71	15.87	12.90	8.85	12.30	11.82	12.90	12.83
SDAP	10.36	13.37	6.94	10.58	11.15	10.50	10.13	11.39	11.16	14.44	11.22	10.41
SASS	15.08	11.48	12.01	12.22	12.49	14.17	11.61	7.66	11.32	10.94	11.68	11.39
LRD	12.53	11.02	12.91	11.92	12.25	12.79	12.13	7.96	11.81	11.62	12.01	11.00
LRD-TV (Ours)	**18.27**	**23.46**	**20.98**	**18.42**	**22.28**	23.13	20.58	17.43	**22.75**	**21.90**	**21.99**	**19.84**
Input PSNR = 9.49 dB / $\sigma = 90$												
BM3D	14.50	13.02	17.69	13.74	15.82	16.29	15.26	11.14	16.28	13.60	15.75	12.87
EPLL	17.01	18.16	16.18	15.46	17.83	16.61	15.26	16.00	16.80	16.68	17.44	16.32
TV	**18.67**	**22.92**	**20.04**	16.98	**22.62**	18.77	**20.77**	17.05	**21.85**	**21.33**	**20.58**	18.95
WNNM	14.26	19.23	15.11	**17.96**	18.16	**21.91**	**20.43**	**19.53**	19.06	17.74	15.68	**22.19**
DIP	10.32	17.00	15.74	16.21	20.74	21.01	17.34	13.17	18.25	16.75	16.40	15.50
N2F	15.95	10.76	12.69	10.03	10.76	9.74	11.12	9.81	9.52	6.19	10.05	10.33
AP-BSN	15.76	11.02	10.64	11.45	11.76	13.50	11.61	7.23	10.45	11.05	10.74	10.68
SDAP	9.21	11.76	5.80	9.19	10.28	9.09	9.65	10.10	9.34	14.08	10.02	9.12
SASS	15.07	10.11	9.86	10.51	10.81	12.12	10.44	6.56	9.72	10.19	10.01	9.73
LRD	12.55	9.90	10.95	10.88	10.60	11.97	10.70	7.03	10.11	10.09	9.92	9.96
LRD-TV (Ours)	**17.77**	**22.41**	**20.57**	**17.75**	**21.66**	**21.45**	19.79	**17.17**	**21.58**	**21.17**	**21.13**	**19.38**

Table 1. **Quantitative evaluation on image denoising. Left:** From top to bottom, we display the collection of twelve images chosen for the experiment. **Right:** The table reports the PSNR in dB (higher is better) using ten state-of-the-art approaches and our method for the task of image denoising. Results are reported for different levels of input noise. Best viewed in color.

Following [11], the selection of parameters for this problem is set to $\{R = 3, M = 25, \alpha = 1 \cdot 10^{-16}\}$ and filters learned on the city and fruit datasets from [21] with dictionary dimensions set to $\mathfrak{D}_m \in \mathbb{R}^{L_1 \times L_2}$ for $m = \{1, \ldots, M\}$ with $\{L_n = 5\}_{n=1}^{2}$. Regarding the choice of γ, figure 3 reports a sensitiviy analysis from where an optimal choice can be made. For each method, we perform an image normalization after the main denoising step. The results are presented in the same table, where we have marked in blue and red color the best and second-best achievers. We observe how LRD-

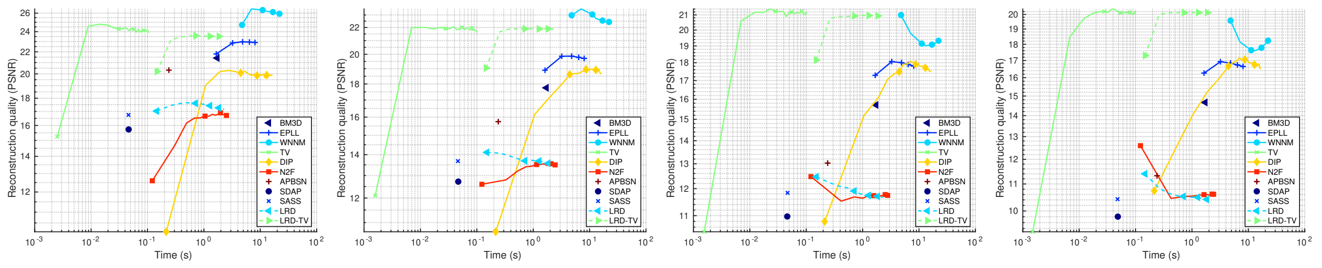

Fig. 1. **Quality of reconstruction (PSNR) vs. execution time (s).** We display overall results on the whole dataset for the four levels of input noise respectively. PSNR evolution as a function of time for a single image denoising for the chosen methods.

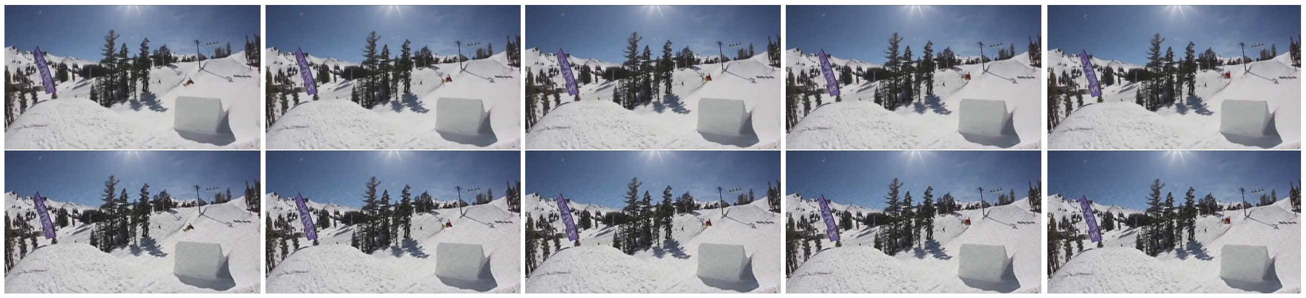

Fig. 2. **Qualitative evaluation on RGB video enhancement.** In all cases, we show five consecutive video frames. **Top.** Original color frames. **Bottom.** Video Enhancement. As it can be seen, our method can be used to enhance details in tensors, such video-sequences. Best viewed in color.

TV systematically outperforms all DL approaches (DIP, N2F, AP-BSN, SDAP and SASS), performs on par with TV and is only beaten sometimes by WNNM. Our method is within the two best achievers in a majority of cases. Also, as stated in the introduction, we observe how LRD without TV regularization is not able to reject noise.

Figure 1 reports a time analysis for the single image denoising task in the whole dataset for the chosen methods. As it can be seen, TV, SDAP and SASS are the most computationally efficient methods obtaining their results well below 100 ms. These are followed by LRD, LRD-TV, N2F and AP-BSN which are able to converge below 1 s. The rest of the methods require between 1 to 10 s to converge to a solution.

4.2. Video Enhancement

In this section we analyse the performance of LRD-TV for video enhancement problems. We select the first 10 frames of the color sequence *Skiing* ($[360 \times 640]$ pixels) of the OTB50 dataset [22]. We represent them by means of three 3-order tensors, one for each channel. Dictionary dimensions are chosen as $\mathfrak{D}_{m,c} \in \mathbb{R}^{L_1 \times L_2 \times L_3}$ for $m = \{1, \dots, M\}$ and $c = \{1, \dots, C\}$, with $\{L_n = 11\}_{n=1}^{3}$, $M = 60$ and $C = 3$. We use the same data to learn the filters applying the algorithm from [23]. The rest of parameters for this problem are set to $\{R = 16, \alpha = 1 \cdot 10^{-16}, \gamma = 1 \cdot 10^{-3}, \zeta = 5 \cdot 10^{-3}\}$, and $\{\gamma_m = 0\}_{m=1}^{30}$, $\{\gamma_m = \gamma\}_{m=31}^{60}$ and $\{\zeta_m = \zeta\}_{m=1}^{30}$, $\{\zeta_m = 0\}_{m=31}^{60}$. The reconstruction parameters are: $\delta = 0.6$, $\{\delta_m = \delta\}_{m=1}^{30}$, $\{\delta_m = 0\}_{m=31}^{60}$.

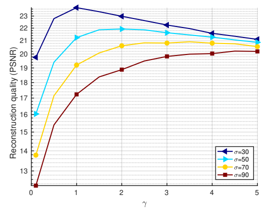

Fig. 3. **Quality of reconstruction (PSNR) vs. γ.** We display overall results on the whole dataset for our method and the four levels of input noise.

Figure 2 presents the qualitative results for the first five frames of the sequence. We can observe how in the ski slope snow details appear improved and in the sky, clouds texture is remarked, resulting in a visually more complex scene.

5. CONCLUSION

LRD is a powerful framework that eases the inclusion of priors to its formulation making it able to solve tensor restoration problems. The results regarding image denoising and video enhancement verify our claims and prove that TV regularization can still compete against the state of the art. The method takes advantage of being an analytical approach and does not require an extensive training stage like most DL approaches.

6. REFERENCES

[1] Leonid I Rudin, Stanley Osher, and Emad Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

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

[3] Kostadin Dabov, Alessandro Foi, Vladimir Katkovnik, and Karen Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on image processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

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

[6] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky, "Deep image prior," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9446–9454.

[7] Jason Lequyer, Reuben Philip, Amit Sharma, Wen-Hsin Hsu, and Laurence Pelletier, "A fast blind zero-shot denoiser," *Nature Machine Intelligence*, vol. 4, no. 11, pp. 953–963, 2022.

[8] Wooseok Lee, Sanghyun Son, and Kyoung Mu Lee, "Ap-bsn: Self-supervised denoising for real-world images via asymmetric pd and blind-spot network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[9] Junyi Li, Zhilu Zhang, Xiaoyu Liu, Chaoyu Feng, Xiaotao Wang, Lei Lei, and Wangmeng Zuo, "Spatially adaptive self-supervised learning for real-world image denoising," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[10] Yizhong Pan, Xiao Liu, Xiangyu Liao, Yuanzhouhan Cao, and Chao Ren, "Random sub-samples generation for self-supervised real image denoising," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 12150–12159.

[11] David Reixach, "Multi-dimensional signal recovery using low-rank deconvolution," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[12] Amir Beck and Marc Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE transactions on image processing*, vol. 18, no. 11, pp. 2419–2434, 2009.

[13] Jiaming Liu, Yu Sun, Xiaojian Xu, and Ulugbek S Kamilov, "Image restoration using total variation regularized deep image prior," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Ieee, 2019, pp. 7715–7719.

[14] Daniel Zoran and Yair Weiss, "From learning models of natural image patches to whole image restoration," in *2011 international conference on computer vision*. IEEE, 2011, pp. 479–486.

[15] 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.

[16] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang, "Learning deep cnn denoiser prior for image restoration," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3929–3938.

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

[18] 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.

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

[20] 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.

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

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

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