Parameter-Free Structure-Texture Image Decomposition by Unrolling

Parameter-free structure-texture image decomposition by unrolling

Laura Girometti[1], Jean-François Aujol[2], Antoine Guennec[2], and Yann Traonmilin[2]

[1] Department of Mathematics, University of Bologna
laura.girometti2@unibo.it
[2] Univ. Bordeaux, Bordeaux INP, CNRS, IMB, UMR 5251, F-33400, Talence, France
{jean-francois.aujol, antoine.guennec, yann.traonmilin}@math.u-bordeaux.fr

Abstract. In this work, we propose a parameter-free and efficient method to tackle the structure-texture image decomposition problem. In particular, we present a neural network LPR-NET based on the unrolling of the Low Patch Rank model. On the one hand, this allows us to automatically learn parameters from data, and on the other hand to be computationally faster while obtaining qualitatively similar results compared to traditional iterative model-based methods. Moreover, despite being trained on synthetic images, numerical experiments show the ability of our network to generalize well when applied to natural images.

Keywords: image decomposition · unrolling

1 Introduction

The problem of structure-texture image decomposition has been widely investigated in the last decades and successfully applied to tasks such as detail enhancement, image abstraction, texture transfer, HDR tone mapping [13,25]. In addition, it has also been exploited to solve inverse problems in imaging such as image fusion [24], inpainting [11] and super-resolution [5]. It aims to find an additive decomposition of an observed image f of the form

$$f = u + v, \tag{1}$$

where u represents the structure component of f containing the geometrical elements identified in edges and homogeneous regions, whereas the texture component v contains repeated patterns of small scale details. This problem is usually formulated as a constrained minimization problem as follows:

$$\min_{u,v\in\mathbb{R}^N \ s.t. f=u+v} \mu\mathcal{J}_1(u) + \mathcal{J}_2(v), \tag{2}$$

where $\mathcal{J}_1$ and $\mathcal{J}_2$ are regularization functions that enforce the desired properties on the components u and v, and μ is a positive weighting parameter that plays a crucial role in separating u from v, adapting to the image content. While for

the structure component the natural choice for $\mathcal{J}_1$ falls on functions promoting gradient-sparsity - Total Variation [19], Non convex Total Variation [12] - various proposals for $\mathcal{J}_2$ have been deeply investigated: high-oscillatory pattern promoting functions - G-norm [16,2] - low patch rank penalties that promote similarity between texture patches - [20,18,8], a generative texture prior inspired by generative convolutional neural networks [11], convolutional sparse coding with dictionary learning-based sparsity prior [23,5]. Another line of research focuses on optimal parameter selection based on separability between components-minimize the cross-correlation between pairs of them [6,3] - or on adjusting the tuning parameters depending on the right relationship between the gradient sparsity level of u and the rank of the texture patches [8]. Recently, motivated by progress in machine learning techniques, various data-driven approaches have been presented. The problem of lack of available datasets containing pairs of structure-texture components has been addressed by applying an unsupervised strategy [25]. The Plug-and-Play framework, where a denoiser is trained to learn a more significant regularizer for the structure u on natural image datasets [13] enables the learning of a joint regularizer $R_x(u, v)$ for structure and texture on a generated synthetic dataset [9]. Among the hybrid techniques that combine model-based methods and deep-learning approaches, algorithm unrolling is a recent promising paradigm to build efficient and interpretable neural networks. These techniques date back to Gregor et al.s paper [7], where they proposed to improve the computational efficiency of sparse coding algorithms through end-to-end training. Following their idea, the guiding principle in building unrolled neural networks is to represent each iteration of an iterative algorithm as one layer of the network. Consequently, mimicking model-based methods results in high interpretability of network layers, fewer parameters compared with popular neural networks and, therefore, less training data, as well as better generalization power than generic networks [17]. Recently, it has been applied to different signal and image processing tasks such as image denoising [14], blind image deblurring [15] and CT [1].

This paper proposes a novel strategy to achieve an efficient and automatic structure-texture image decomposition by leveraging the unrolling technique: model and algorithm parameters are learned in a supervised fashion making use of a synthetic generated dataset. In particular, in Section 2, we propose to unroll an extension of the Low Patch Rank model [20] and we present a numerical solution based on the Alternating Direction Method of Multipliers (ADMM). In Section 3, we propose our neural network LPR-NET and we investigate various design choices. Finally, in Section 4, numerical results validate that our proposal outperforms optimally hand-tuned classical variational models and gives a competitive end-to-end alternative to the Plug-and-Play framework where optimization of the learned regularization must be performed.

2 Proposed structure-texture decomposition model

In this section, we first describe an extended version of the Low Patch Rank model for structure-texture decomposition and we present an ADMM-based algorithm to solve it. Then, we propose our LPR-NET network based on the unrolling of the proposed ADMM iterative scheme.

2.1 Low Patch Rank model for structure-texture decomposition

Given an image f possibly degraded by a linear operator $\mathcal{M}$ (e.g. deblurring, mask operator), the Low Patch Rank model [20] recovers the structure component u and the texture component v by enforcing u to be gradient sparse and v to be of low patch rank. Specifically, they consider the classical Total Variation as regularization function $\mathcal{J}_1$ and the nuclear norm of texture patches as $\mathcal{J}_2$. We propose to extend this model by considering, instead of the ℓ_1-norm, a non-convex non-smooth sparsity promoting function $\phi(\cdot; a)$ parametrized by $a \geq 0$. The resulting variational problem reads as:

$$\min_{u\in\mathbb{R}^N, v\in\mathbb{R}^N s.t. f=\mathcal{M}(u+v)} \mu \sum_{i=1}^{N} \phi(||(\mathrm{D}u)_i||_2; a) + ||\mathcal{P}v||_*, \tag{3}$$

where μ is a positive balancing parameter, $(\mathrm{D}u)_i$ represents the discrete gradient of u at pixel i and the nuclear norm $||\cdot||_*$ is chosen as a convex surrogate of the rank function. The operator $\mathcal{P}$ represents the patch operator characterized by the size p of the $p \times p$ patches and by the overlap o between two adjacent patches as shown in Figure 1:

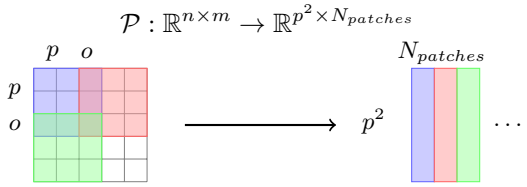

Fig. 1. Graphical action of the patch operator $\mathcal{P}$.

The considered non-convex penalty function $\phi(\cdot; a)$ is the minimax concave penalty (MCP) [22] defined as

$$\phi(t; a) = \begin{cases} |t| - \frac{a}{2}t^2 & \text{if } |t| \leq \frac{1}{a} \\ \frac{1}{2a} & \text{if } |t| \geq \frac{1}{a} \end{cases}. \tag{4}$$

Note that as $a \to 0$, we recover the absolute value function $|t|$ as a special case of the penalty function $\phi(\cdot; a)$ thus the classical Total Variation regularizer in (3). The MCP function was first introduced in [22] in the context of unbiased variable selection to overstep the bias introduced by the ℓ_1 penalty on large coefficients. Therefore, we aim to less penalize high gradients in the structure component u.

2.2 ADMM-based algorithm for LPR

To numerically solve the constrained minimization problem in (3), we equivalently reformulate the problem introducing two auxiliary variables $t := \mathrm{D}u \in \mathbb{R}^{2N}$ and $s := v \in \mathbb{R}^N$, which represent the discrete gradient of u and the texture v respectively. Problem (3) is rewritten as

$$\min_{\substack{t \in \mathbb{R}^{2N}, u, v, s \in \mathbb{R}^N \\ \text{s.t. } t = Du,\, s = v}} \mu \sum_{i=1}^{N} \phi(||t_i||_2; a) + ||\mathcal{P}s||_* + i_{\mathcal{C}}(u, v) \tag{5}$$

where the constraint $f = \mathcal{M}(u + v)$ is imposed by adding $i_{\mathcal{C}}$, the indicator function of the convex set $\mathcal{C} := \{x := (u, v) \in \mathbb{R}^{2N} \text{ s.t. } f = \mathcal{M}(u + v)\}$, to the cost function in (3). We propose the following Alternating Direction Method of Multipliers iterative scheme, where we denote by ρ_t and ρ_s the ADMM penalty parameters, by $x := \begin{pmatrix} u \\ v \end{pmatrix}$ and $y := \begin{pmatrix} y_t \\ y_s \end{pmatrix}$ the vector of Lagrange multipliers where $y_t \in \mathbb{R}^{2N}$ and $y_s \in \mathbb{R}^N$. Given the initial guesses $x^0 := (u^0, v^0), y^0 := (y_t^0, y_s^0)$, the k-th iteration reads as

$$t^k \in \arg\min_{t \in \mathbb{R}^{2N}} \sum_{i=1}^{N} \mu\phi(||t_i||_2; a) + \frac{\rho_t}{2} \left\| t - \left(\mathrm{D}u^{k-1} + \frac{y_t^{k-1}}{\rho_t} \right) \right\|^2 \tag{6}$$

$$s^k \in \arg\min_{s \in \mathbb{R}^N} \|\mathcal{P}s\|_* + \frac{\rho_s}{2} \left\| s - \left(v^{k-1} + \frac{y_s^{k-1}}{\rho_s} \right) \right\|^2 \tag{7}$$

$$x^k \in \arg\min_{x \,\in\, \mathcal{C}} \frac{\rho_t}{2} \left\| t^k - \frac{y_t^{k-1}}{\rho_t} - \begin{bmatrix} \mathrm{D} & 0_N \end{bmatrix} x \right\|^2 + \frac{\rho_s}{2} \left\| s^k - \frac{y_s^{k-1}}{\rho_s} - \begin{bmatrix} 0_N & \mathrm{I}_N \end{bmatrix} x \right\|^2 \tag{8}$$

$$y^k = y^{k-1} + \begin{bmatrix} \rho_t \mathrm{D} \\ \rho_s \mathrm{I}_N \end{bmatrix} x^k - \begin{pmatrix} \rho_t t^k \\ \rho_s s^k \end{pmatrix} \tag{9}$$

In the following, we describe the numerical solution of problems (6)-(8).

***t*-subproblem, estimation of gradients**: The minimization problem (6) is separable with respect to $t_i = (\mathrm{D}u)_i \in \mathbb{R}^2$ hence is equivalent to N optimization problems of the form:

$$\min_{t_i \in \mathbb{R}^2} \mu\phi(||t_i||_2; a) + \frac{\rho_t}{2} \left\| t_i - \left((\mathrm{D}u^{k-1})_i + \frac{(y_t^{k-1})_i}{\rho_t} \right) \right\|^2. \tag{10}$$

Sufficient conditions to ensure that problem (10) has a unique solution are presented in Proposition 1. For a detailed proof, see [12].

Proposition 1. *Given $v \in \mathbb{R}^2$, $a \geq 0$, $\mu, \rho_t > 0$, the function $\frac{\mu}{\rho_t}\phi(\cdot; a) + \frac{1}{2}||t_i - v||^2$ is strongly convex if $a \leq \frac{\rho_t}{\mu}$. Morever, it has a unique global minimizer that has the following closed-form expression:*

$$t_i = \begin{cases} \min\left(1, \frac{1}{1 - a\frac{\mu}{\rho_t}} \max\left(1 - \frac{\mu}{\rho_t ||v||_2}, 0\right)\right) v & \text{if } ||v||_2 > 0 \\ 0 & \text{if } ||v||_2 = 0 \end{cases} \tag{11}$$

Equation (11) is a generalization of the proximal operator of the ℓ_2-norm (recovered for $a \to 0$) where a second shrinkage value, that depends on a, is introduced.
s-subproblem, estimation of LR texture: The solution of the minimization problem in (7) is the evaluation of the proximal operator of the nuclear norm in $v^{k-1} + \frac{y_s^{k-1}}{\rho_s}$. This proximal operator has the following closed-form solution:

$$\text{prox}_{||\cdot||_*,\beta}(x) = U\max(\mathrm{S} - \beta\mathrm{I})V^T \tag{12}$$

where $x = \mathrm{USV}^{\mathrm{T}}$ is the Singular Value Decomposition (SVD) of x and the maximum is taken elementwise.
x-subproblem, enforcing the constraint: The minimization problem in (8) is a linearly constrained quadratic programming that does not admit a general closed-form solution but which can be efficiently solved by means of the Projected Gradient Descent. The cost function is differentiable with respect to x and the projection onto the set $\mathcal{C}$ can be computed as in [9]. Starting from x_0, the algorithm reads as

$$x_{i+1} = \mathcal{P}_{\mathcal{C}}\left(x_i - \tau\left(\mathrm{A}x_i - \mathrm{B}z\right)\right),\ i = 0, \ldots, n-1,$$

where τ is the gradient-step parameter, $\mathcal{P}_{\mathcal{C}}$ denotes the projection onto the set $\mathcal{C}$ and

$$\mathrm{A} = \begin{bmatrix} \rho_t \mathrm{D}^{\mathrm{T}}\mathrm{D} & \mathrm{O} \\ \mathrm{O} & \rho_s \mathrm{I} \end{bmatrix},\ \mathrm{B} = \begin{bmatrix} \rho_t \mathrm{D} & \mathrm{O} \\ \mathrm{O} & \rho_s \mathrm{I} \end{bmatrix} \text{ and } z = \begin{pmatrix} t^k - \frac{y_t^{k-1}}{\rho_t} \\ s^k - \frac{y_s^{k-1}}{\rho_s} \end{pmatrix}. \tag{13}$$

We remark that the proposed ADMM algorithm has no guarantee of convergence, except for $a = 0$, when the cost function is convex. For the general case, as detailed in [21], a Lipschitz-gradient term is required. This would imply the replacing of the strong constraint by some L^2 penalization, that could also handle noisy cases. Nevertheless, we prefer to hardly force u and v to satisfy the constraint.

3 LPR-NET architecture

Starting from the ADMM iterative scheme proposed in (6)-(9), we propose an unrolled neural network by stacking K blocks, each one resembling one iteration of the iterative algorithm considered. As described in Section 2, one iteration of the proposed ADMM consists of four subproblems to be solved, corresponding to the t, s, x and y updates. Hence, the generic k-th block of our network will be divided into four consecutive subblocks reproducing respectively the four subproblems. In Figure 2, the flow of a generic block of the network is shown.

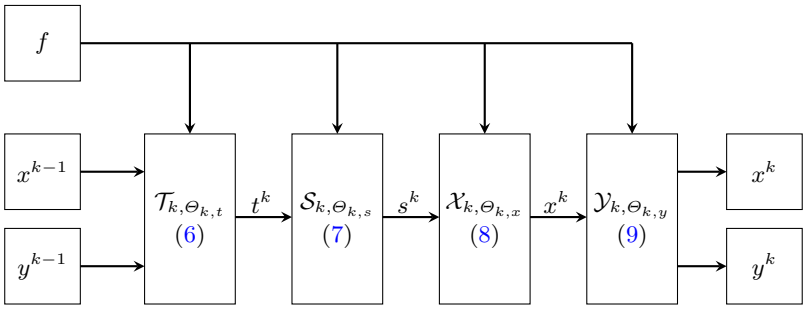

Fig. 2. Flow of the k-th block of the network. Each operation performed in a block corresponds to a step of the ADMM algorithm described in (6)-(9).

We define the output of the k-th block as

$$(x^k, y^k) = \mathcal{L}^k_{\Theta_k}(x^{k-1}, y^{k-1}) := \mathcal{Y}_{\Theta_{k,y}} \circ \mathcal{X}_{\Theta_{k,x}} \circ \mathcal{S}_{\Theta_{k,s}} \circ \mathcal{T}_{\Theta_{k,t}}(x^{k-1}, y^{k-1}) \quad (14)$$

where $\Theta_{k,t}$, $\Theta_{k,s}$, $\Theta_{k,x}$ and $\Theta_{k,y}$ denote the set of learnable parameters in $\mathcal{T}_{k,\Theta_{k,t}}$, $\mathcal{S}_{k,\Theta_{k,s}}$, $\mathcal{X}_{k,\Theta_{k,x}}$, $\mathcal{Y}_{k,\Theta_{k,y}}$ subblocks respectively. Finally, composing K blocks, we have the final relation between the input and the output of the network:

$$(x^K, y^K) = \mathcal{F}_\Theta(x^0, y^0) = \mathcal{L}^K_{\Theta_K} \circ \cdots \circ \mathcal{L}^1_{\Theta_1}(x^0, y^0), \quad (15)$$

where $\Theta_k = \{\Theta_{k,t}, \Theta_{k,s}, \Theta_{k,x}, \Theta_{k,y}\}$ and $\Theta := \cup_{k=1}^{\mathrm{K}} \Theta_k$ denotes the set of the learnable parameters of the network $\mathcal{F}_\Theta$. Various design options can be explored when constructing the subblocks $\mathcal{T}_{k,\Theta_{k,t}}$, $\mathcal{S}_{k,\Theta_{k,s}}$, $\mathcal{X}_{k,\Theta_{k,x}}$, $\mathcal{Y}_{k,\Theta_{k,y}}$ depending on the desired level of flexibility and complexity of $\mathcal{F}_\Theta$.

$\mathcal{T}_{\mathbf{k},\boldsymbol{\Theta}_{\mathbf{k,t}}}$-subblock, estimation of gradients: The closed form expression of (11) can be recast as a linear layer followed by a non linear function as follows:

$$t^k = \eta_1 \left(\mathrm{D}^{(k)} u^{k-1} + \frac{y_t^{k-1}}{\rho_t}; \mu^{(k)}, \rho_t^{(k)} \right) \quad (16)$$

where η_1 is the non linear function defined by the proximal operator of the function $\phi(\cdot; a)$ defined in (11). We replace the discrete operator D with a convolutional operator $\mathrm{D}^{(k)}_{\mathcal{T}}$ with kernel size 2×2 and 2 output channels. We also learn the model parameter $\mu^{(k)}$ that is free to vary in different blocks while the penalty parameter ρ_t is shared across them. To ensure the condition in Proposition 1 is satisfied, we parametrize $a^{(k)}$ as $a^{(k)} = \frac{\rho_t}{\mu^{(k)}} \left(1 - \frac{1}{b^{(k)}}\right)$ where $b^{(k)}$ is a learnable parameter constrained to be larger than 1.

$\mathcal{S}_{\mathbf{k},\boldsymbol{\Theta}_{\mathbf{k,s}}}$-subblock, estimation of LR texture: As described in Equation (12), updating s involves the computation of the SVD decomposition of the input since we need to apply a threshold on its singular values. In this work, we do not explore possible linear approximations of the SVD decomposition and we compute it exactly. Therefore, we preserve the same structure of (12) adding

flexibility by learning a shared ρ_s. Finally, the patch operator requires the size of the patch p and the overlap o between adjacent patches and a good choice is to fix $p = 4$ and $o = 2$.

$\mathcal{X}_{\mathbf{k},\boldsymbol{\Theta}_{\mathbf{k},\mathbf{x}}}$-subblock, enforcing the constraint: The algorithm described in (13) can be viewed as a composition of n layers, each one made of a linear layer followed by a non linear function, as illustrated in detail in Figure 3. To preserve the specific form of the algorithm while gaining flexibility, we replace the discrete gradient operator D and its transpose D^{T} with two convolutional operators with kernel size 2×2 and 2 and 1 output channel respectively. We denote them with $\mathrm{D}_{\mathcal{X}}^{(k)}$ and $\tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}$ allowing them to change in each block $\mathcal{X}_{k,\Theta_{k,x}}$ and relaxing the matching between $\tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}$ and $(\mathrm{D}_{\mathcal{X}}^{(k)})^{\mathrm{T}}$. Finally, the parameter τ is learned and shared across different blocks.

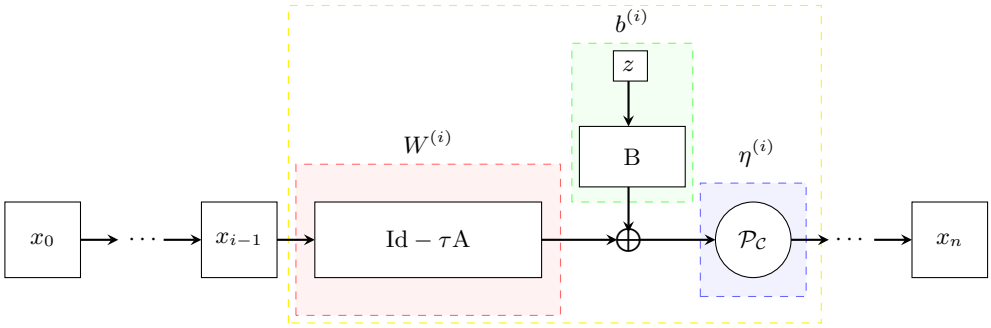

Fig. 3. Unrolling of the Projected Gradient Descent algorithm.

$\mathcal{Y}_{\mathbf{k},\boldsymbol{\Theta}_{\mathbf{k},\mathbf{y}}}$-subblock: in the updating of the Lagrange multipliers, we only replace the discrete gradient operator D with a convolutional kernel of size 2×2 with two channels denoted as $\mathrm{D}_{\mathcal{Y}}^{(k)}$.

Specifically, we evaluate and compare four distinct configurations of parameters with the twofold aim of studying the improvement gained with:

- a non-convex penalty ϕ (a>0) with respect to the Total Variation (a=0);
- a data-dependent $\mu^{(k)}$, estimated on x^k after each block $\mathcal{L}_{\Theta_k}^k$ with respect to a fixed $\mu^{(k)}$. In the former, the estimation is performed by adding a small CNN composed of three convolutional layers with kernel size 3×3 and 4,8,16 channels respectively, each one followed by a ReLU, and in the end, we have a fully connected layer and a Softplus to enforce the positivity of the estimated parameter $\mu^{(k)}$.

By combining a convex ($a = 0$) or non-convex Total Variation ($a > 0$) and a data-dependent or fixed $\mu^{(k)}$, we obtain four configurations. For each configuration, the learnable parameters are summarized as follows in Table 1: Configurations Θ_1 and Θ_2 contain less parameters and a lower representation power than Θ_3

	$\Theta_{k,t}$	$\Theta_{k,s}$	$\Theta_{k,x}$	$\Theta_{k,y}$	$\lvert\Theta\rvert$
Θ_1	$a^{(k)}=0, \mu^{(k)}, \rho_t, \mathrm{D}_{\mathcal{T}}^{(k)}$	ρ_s	$\mathrm{D}_{\mathcal{X}}^{(k)}, \tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}, \tau$	$\mathrm{D}_{\mathcal{Y}}^{(k)}$	$\approx 50\mathrm{K}$
Θ_2	$b^{(k)}, \mu^{(k)}, \rho_t, \mathrm{D}_{\mathcal{T}}^{(k)}$	ρ_s	$\mathrm{D}_{\mathcal{X}}^{(k)}, \tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}, \tau$	$\mathrm{D}_{\mathcal{Y}}^{(k)}$	$\approx 50\mathrm{K}$
Θ_3	$a^{(k)}=0, \mathrm{CNN}^{(k)}, \rho_t, \mathrm{D}_{\mathcal{T}}^{(k)}$	ρ_s	$\mathrm{D}_{\mathcal{X}}^{(k)}, \tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}, \tau$	$\mathrm{D}_{\mathcal{Y}}^{(k)}$	$\approx 2\cdot 10^3\mathrm{K}$
Θ_4	$b^{(k)}, \mathrm{CNN}^{(k)}, \rho_t, \mathrm{D}_{\mathcal{T}}^{(k)}$	ρ_s	$\mathrm{D}_{\mathcal{X}}^{(k)}, \tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}, \tau$	$\mathrm{D}_{\mathcal{Y}}^{(k)}$	$\approx 2\cdot 10^3\mathrm{K}$

Table 1. Four different parameter configurations with an estimate of the number of parameters.

and Θ_4, whereas configurations Θ_2 and Θ_4 should take advantage of the non-convex setting to better recover piecewise constant images. A detailed analysis will be carried out in Section 4.

4 Numerical Experiments

In this section, the training procedure of our LPR-NET is explained in detail and some numerical results are presented. In the first example, we will investigate the role that K, n and the four different configurations play in the final output results while in the second example, we present decomposition of synthetic and natural images compared with state-of-the-art methods.

Training strategy. We generate a synthetic image dataset composed of 400 images of dimension 64×64 that are the sum of a piecewise constant image u^{GT} and a texture image v^{GT} eventually degraded by a mask operator $\mathcal{M}$. The structure component is obtained by randomly positioning connected supports generated via the Lane-Riesenfeld algorithm while sparse Fourier texture generation has been used for the texture component. For further details, see [9]. Given our synthetic dataset and our architecture $\mathcal{F}_\Theta$, the estimate of the set of parameters Θ is carried out via the minimization of the Mean Square Error over the training dataset:

$$\widehat{\Theta} \in \arg\min_{\Theta} \frac{1}{N_{train}} \sum_{j=1}^{N_{train}} \left\| \mathcal{F}_\Theta(x_j^0, y_j^0) - x_j^{\mathrm{GT}} \right\|^2, \tag{17}$$

where $x_j^0 = (u_j^0, v_j^0)$ and y_j^0 are the initial guesses of the ADMM set as $u_j^0 = f_j$, $v_j^0 = 0$ and $y_j^0 = 0$. At the k-th block of the network, the initial guess for x_j in the Projected Gradient Descent is chosen as x_j^{k-1}. The network has been implemented on Pytorch and trained using Adam optimizer with learning rate $= 10^{-3}$ halved after forty epochs, training batch size $= 16$ and for a total of 250 epochs. We initialized the parameters as: $(\mu^{(k)})^0 = 0.05$, $\rho_t^0 = 1$, $\rho_s^0 = 1$, $\tau^0 = 0.1$, $(b^{(k)})^0 = 1.1$, $\mathrm{D}_{\mathcal{T}}^{(k)}, \mathrm{D}_{\mathcal{X}}^{(k)}, \mathrm{D}_{\mathcal{Y}}^{(k)}$ initialized as the discrete gradient, $\tilde{\mathrm{D}}_{\mathcal{X}}^{(k)}$ initialized as the transpose of the discrete gradient and the parameters of the small CNN are initialized such that the output of the network is equal to $(\mu^{(k)})^0 = 0.05$ (all zeros except the bias of the last fully connected layer).

Example 1: Ablation study of the LPR-NET. In this example, we trained

our LPR-NET for different values of the number of unrolled iterations of the ADMM K and of the Projected Gradient Descent n over the training dataset of 400 images considering no degradation $\mathcal{M}$. In particular, we chose K $= 5, 10, 15, 20$ and $n = 5, 10, 15, 20$ and quantitatively compared the different networks on a test dataset of 1000 images generated from the synthetic generator of structure and texture images available at [10]. In Table 2, the mean PSNR values of the dataset are reported for the four configurations of parameters.

	Θ_1					Θ_2				
	n	5	10	15	20	n	5	10	15	20
K	5	38.91	39.03	39.11	39.23	5	39.14	39.13	38.93	39.09
	10	40.44	40.38	40.48	40.49	10	40.78	40.61	40.60	40.61
	15	41.08	41.11	41.01	40.94	15	41.44	41.34	41.28	41.24
	20	41.60	41.44	41.35	41.38	20	42.00	41.75	41.68	41.74
	Θ_3					Θ_4				
	n	5	10	15	20	n	5	10	15	20
K	5	39.23	39.34	39.38	39.40	5	39.52	39.51	39.55	39.54
	10	40.64	40.49	40.41	40.39	10	41.02	40.93	40.82	40.93
	15	41.33	41.19	41.13	41.00	15	41.69	41.69	41.60	41.46
	20	41.81	41.51	41.43	41.41	20	42.13	41.96	41.76	41.65

Table 2. mean PSNR over the test dataset for different values of K and n and for the four different configurations.

As expected, increasing the number of blocks of the network K leads to better quality decomposition results while higher values of n show a small deterioration. Among the four configurations, adding a non-convex penalty ϕ improved the decomposition with respect to $a = 0$ while adding a small CNN to make $\mu^{(k)}$ data-dependent brings a slight improvement when compared to a variable $\mu^{(k)}$ shared for all images. Note that we limit our ablation study to K and n but it could be extended to the size of filters, on matching and untied adjoints and that more flexibility could be achieved by allowing larger filters and more channels.

Example 2: comparison with state-of-the-art methods. In this example, we compare the best configuration of our LPR-NET with three variational models - TV-L_2, TV-G, LPR - and with the joint structure-texture decomposition model $R_x(u, v)$ proposed in [9], reporting the mean PSNR over the same test dataset in Table 3. The best configuration of our LPR-NET outperforms clas-

	Θ_4	TV-L_2 [4]	TV-G [2]	LPR [20]	$R_x(u, v)$ [13]
PSNR	42.13	38.84	38.86	41.61	42.69

Table 3. Comparison of mean PSNR values with classical variational models and [9]. Notice that the regularization parameter μ was set by trial and error for the 3 variational models to achieve the best possible PSNR.

sical variational models whose parameters are selected in order to get the best PSNR values for each distinct image. This means that, despite, on average, our results are worse than the $R_x(u, v)$ model, our network is able to automatically set the optimal parameters adapting to the image content avoiding having to set-up the parameters of the Plug-and-Play optimization algorithm. We also remark that, with respect to [9], our network has a lighter architecture (around 4×10^4 parameters) that requires a smaller training dataset and less training hours (in the worst case, around 4 hours). In Figure 4 and Figure 5, we illustrate decomposition results of synthetic and natural images compared with $R_x(u, v)$.

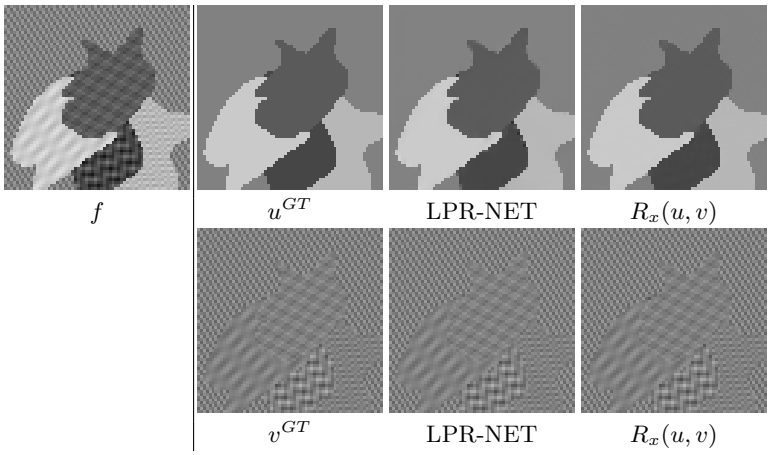

Fig. 4. Decomposition results of a synthetic image obtained with our LPR-NET with $(\mathrm{K}, n) = (20, 5)$ and with the joint structure-texture model proposed in [9](PSNR$(u_{\mathrm{LPR-NET}}, u^{GT}) = 41.93$, PSNR$(u_{R_x(u,v)}, u^{GT}) = 40.53$, PSNR$(v_{\mathrm{LPR-NET}}, v^{GT}) = 41.80$, PSNR$(v_{R_x(u,v)}, v^{GT}) = 40.25$).

While for the synthetic image the results are visually comparable, we note how on natural images the behaviour is slightly different: the LPR-NET removes most of the texture but also some structural edges (e.g. the face of the pirate) while the $R_x(u, v)$ model better preserves the geometric of the image but removes few tiles in the mosaic.

Extension to inverse problems. Finally, we also carry out some inpainting experiments. With this aim, we trained our LPR-NET ($(\mathrm{K}, n) = (20, 20)$) following the training strategy described in Section 4 except for the training dataset where each image is masked with some random shapes. In Figure 6, we compare our model with [9] on Barbara where we randomly remove 65% of pixels. The results are promising when the measurement operator is fixed and well described by $\mathcal{M}$ as our network achieves a more "natural" decomposition with respect to the Plug-and-Play method $R_x(u, v)$ where some artifacts are visible in Barbara's arm (recall that $R_x(u, v)$ is trained independently of the measurement method).

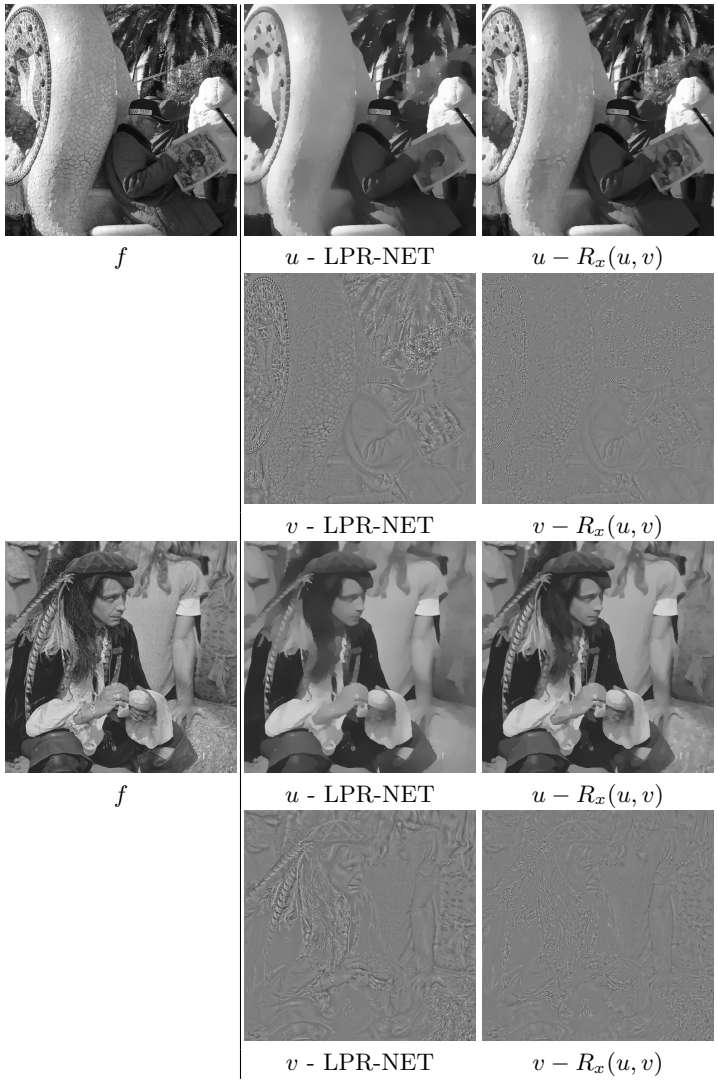

Fig. 5. Decomposition results on natural images obtained with our LPR-NET with $(\mathrm{K}, n) = (20, 5)$ (left) and with the model proposed in [9] (right).

Fig. 6. Inpainting results on Barbara with 65 % of pixels randomly removed, and comparison with the model proposed in [9].

5 Conclusion

In this paper, we presented a fully automatic way of separating images into structure and texture components based on the unrolling of a generalized low patch rank model. We proposed a novel unrolled network LPR-NET and investigated various architectural choices. Numerical experiments validate the potentiality of our network to achieve high quality decomposition results without tuning parameters.

References

1. J. Adler and O. Öktem, "Learned Primal-Dual Reconstruction", in IEEE Transactions on Medical Imaging, vol. 37, no. 6, pp. 1322-1332, 2018.
2. J-F. Aujol, G. Aubert, L. Blanc-Feraud, and A. Chambolle, "Image decomposition into a bounded variation component and an oscillating component", Journal of Mathematical Imaging and Vision, 22 571, pp. 7188, 2005.
3. J-F. Aujol, G. Giboa, T. Chan, and S. Osher, "Structure-texture image decomposition - Modeling, Algorithms, and Parameter selection", International Journal of Computer Vision, 67, pp. 11-136, 2006.
4. A. Chambolle, "An algorithm for total variation minimization and applications", Journal of Mathematical 586 imaging and vision, 20, pp. 8997, 2004.
5. S. Ducotterd, S. Neumayer and M. Unser, "Learning of Patch-Based Smooth-Plus-Sparse Models for Image Reconstruction", arXiv preprint arXiv:2412.13070, 2024.
6. L. Girometti, M. Huska, A. Lanza and S. Morigi, "Quaternary image decomposition with crosscorrelation-based multi-parameter selection", In: Scale Space and Variational Methods in Computer Vision: 9th International Conference, SSVM 2023, pp. 120133, 2023.
7. K. Gregor and Y. LeCun, Learning Fast Approximations of Sparse Coding, in Proc. Int. Conf. Machine Learning, 2010.
8. A. Guennec, J.-F. Aujol, Y. Traonmilin, "Adaptive parameter selection for gradient-sparse plus low patch-rank recovery: application to image decomposition", 32nd European Signal Processing Conference (EUSIPCO), Lyon, France, pp. 2672-2676, 2024.

9. A. Guennec, J-F. Aujol and Y. Traonmilin, "Joint structure-texture low dimensional modeling for image decomposition with a plug and play framework", ffhal-04648963, 2024.
10. A. Guennec, J-F. Aujol and Y. Traonmilin, https://github.com/aguennecjacq/joint_decomposition, 2024.
11. A. Habring and M. Holler, "A Generative Variational Model for Inverse Problems in Imaging", SIAM Journal on Mathematics of Data Science, vol. 4, no 1, pp. 306-335, 2022.
12. M. Huska, A. Lanza, S. Morigi, and I. Selesnick, "A convex-nonconvex variational method for the additive decomposition of functions on surfaces", Inverse Problems, 35, 124008, 2019.
13. Y. Kim, B. Ham, M. N. Do, and K. Sohn, Structuretexture image decomposition using deep variational priors, IEEE Trans. Image Process., vol. 28, no. 6, pp. 26922704, 2019.
14. H. T. V. Le, N. Pustelnik and M. Foare, "The faster proximal algorithm, the better unfolded deep learning architecture ? The study case of image denoising," 30th European Signal Processing Conference (EUSIPCO), Belgrade, Serbia, pp. 947-951, 2022.
15. Y. Li, M. Tofighi, J. Geng, V. Monga, and Y. C Eldar, Efficient and Interpretable Deep Blind Image Deblurring via Algorithm Unrolling, IEEE Trans. Comput. Imaging, vol. 6, pp. 666681, Jan. 2020
16. Y. Meyer and D. Lewis, "Oscillating Patterns in Image Processing and Nonlinear Evolution Equations: The Fifteenth Dean Jacqueline B. Lewis Memorial Lectures", Memoirs of the American Mathematical Society, American Mathematical Society, 2001.
17. V. Monga, Y. Li, and Y. C. Eldar, Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing, IEEE Signal Processing Magazine, vol. 38, no. 2, pp. 1844, 2021.
18. S. Ono, T. Miyata, and I. Yamada, Cartoon-texture image decomposition using blockwise low-rank texture characterization, IEEE Transactions on Image Processing, vol. 23, no. 3, pp. 11281142, March 2014.
19. L. Rudin, S. Osher, E. Fatemi. "Nonlinear total variation based noise removal algorithms", Physica D, 60, 259268, 1992.
20. H. Schaeffer and S. Osher, A low patch-rank interpretation of texture, SIAM Journal on Imaging Sciences, vol. 6, no. 1, pp. 226262, 2013.
21. A. Themelis, P. Patrinos. "DouglasRachford splitting and ADMM for nonconvex optimization: tight convergence results", SIAM J. Optim. 30(1), 149181, 2020.
22. C. H. Zhang. "Nearly unbiased variable selection under minimax concave penalty", Ann. Stat. 38 (2), pp. 894942, 2010.
23. H. Zhang and V. M. Patel, "Convolutional Sparse and Low-Rank Coding-Based Image Decomposition," in IEEE Transactions on Image Processing, vol. 27, no. 5, pp. 2121-2133, 2018.
24. Y. Zhang, M. Yang, N. Li and Z. Yu, "Analysis-synthesis dictionary pair learning and patch saliency measure for image fusion" Signal Process. 167, pp. 107327, 2020.
25. F. Zhou, Q. Chen, B. Liu, and G. Qiu, Structure and texture-aware image decomposition via training a neural network, IEEE Transactions on Image Processing, vol. 29, pp. 34583473, 2019.