Universal Denoising Networks : A Novel CNN-based Network Architecture for Image Denoising

Stamatios Lefkimmiatis
Skolkovo Institute of Science and Technology (Skoltech), Moscow, Russia
s.lefkimmiatis@skoltech.ru

Abstract

We design a novel network architecture for learning discriminative image models that are employed to efficiently tackle the problem of grayscale and color image denoising. Based on the proposed architecture, we introduce two different variants. The first network involves convolutional layers as a core component, while the second one relies instead on non-local filtering layers and thus it is able to exploit the inherent non-local self-similarity property of natural images. As opposed to most of the existing neural networks, which require the training of a specific model for each considered noise level, the proposed networks are able to handle a wide range of different noise levels, while they are very robust when the noise degrading the latent image does not match the statistics of the one used during training. The latter argument is supported by results that we report on publicly available images corrupted by unknown noise and which we compare against solutions obtained by alternative state-of-the-art methods. At the same time the introduced networks achieve excellent results under additive white Gaussian noise (AWGN), which are comparable to the current state-of-the-art network, while they depend on a more shallow architecture with the number of trained parameters being one order of magnitude smaller. These properties make the proposed networks ideal candidates to serve as sub-solvers on restoration methods that deal with general inverse imaging problems such as deblurring, demosaicking, superresolution, etc.

1. Introduction

Image denoising is among the basic low-level computer-vision problems and has received significant attention in both academic research as well as in practical digital imaging applications [9, 32]. However, during the past decade there was little progress in improving the state-of-the art denoising performance and it has been suggested that denoising algorithms have reached optimality and cannot be further improved [28]. Despite these beliefs, very recently and thanks to the advent of deep learning methods, several powerful image denoising algorithms that managed to significantly improve the state-of-the-art performance have been introduced [6, 23, 38, 40, 42, 43]. Nevertheless, their wide applicability in real-world applications is currently hindered mainly because the majority of them involves the training of a specific model for each considered noise level. Such requirement is rather impractical since it implies that a huge number of network parameters, analogous to the number of noise levels that the models are trained for, needs to be stored. This directly excludes the application of such methods on devices with limited memory capacity. Another important limitation of such deep-learning methods is that their denoising performance deteriorates very fast when the noise level distorting the input images deviates from the one that the model was originally trained for.

In this work, motivated by the recent advances in deep learning and relying on the rich body of algorithmic ideas developed in the past for dealing with image restoration problems, we introduce a novel network architecture specifically tailored to image denoising, which allows the training of image models that can handle a wide range of noise levels. Based on the proposed network architecture we introduce two different variants. The first network involves convolutional layers as a core component and behaves similarly to local variational methods, while the second one relies on a non-local filtering layer that allows us to exploit the inherent non-local self-similarity property of natural images. Both networks lead to very competitive results, which are directly comparable to the state-of-the art, while they involve considerably less parameters than the current best-performing network. At the same time they are robust and perform very well for inputs distorted by noise whose statistics differ from the ones of the noise model used for training.

2. Image Restoration

To restore a latent grayscale or color image $\mathbf{X}$ from a corrupted observation $\mathbf{Y}$, we rely on the linear model

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}. \tag{1}$$

In this setting, $\mathbf{y}, \mathbf{x} \in \mathbb{R}^{N \cdot C}$ are the vectorized versions of the observed and latent images, $\mathbf{Y}$ and $\mathbf{X}$ respectively, N is the number of pixel in each image channel, C is the number of channels, $\mathbf{H}$ is a linear operator that corresponds to the response of the imaging device, and $\mathbf{n}$ is the measurement noise that accounts for all possible errors during image acquisition, including stochastic noise and the possible mismatch between the observation model and the physical image acquisition process. For image denoising, which is the focus of this work, the linear operator $\mathbf{H}$ reduces to the identity matrix $\mathbf{I}$, since it is assumed that the imaging device does not introduce any other distortions to the latent signal. Regarding the term $\mathbf{n}$, the most common assumption in the literature, which we also adopt in this work, is that it is zero-mean i.i.d Gaussian noise with variance σ^2.

While the additive white Gaussian noise (AWGN) assumption is not frequently met in practice, an efficient solution of this problem is extremely valuable for two main reasons. The first one is that even in cases where the noise is signal dependent, there are several techniques available in the literature, such as variance stabilization transforms (VST) [1, 12, 30], which are able to transform the input data in a different domain so that the noise follows a Gaussian distribution with a fixed variance. Therefore, the solution can be obtained by first performing Gaussian denoising in the transform domain and then mapping the solution back to the original domain using the inverse VST. The second reason is that such a solution, in the context of convex optimization, can be interpreted as a proximal map [33] of a regularization function. Such proximal maps typically serve as building blocks of several powerful optimization schemes that have been proposed in the literature, including Majorization-Minimization [11, 18] and variable-splitting strategies [3]. These optimization strategies can address more general image restoration problems such as image deblurring, superresolution, demosaicking, inpainting, *etc*.

2.1. Image Priors

While Eq. (1) corresponds to a linear problem, the presence of the noise, whose exact realization is unknown, combined with the fact that usually the operator $\mathbf{H}$ is singular, makes it an ill-posed problem [2, 41]. This implies that a unique solution does not exist and therefore we cannot rely solely on the image evidence but we need to further exploit *a priori* information. In this case, the utilization of suitable prior models of image or scene properties plays an instrumental role in the success of image restoration methods.

While there are several ways of imposing prior knowledge on the solution, among the most popular strategies is the variational approach. In this framework, image recovery is cast as a minimization problem of an objective function of the form

$$f(\mathbf{x}) = d(\mathbf{x}; \mathbf{H}, \mathbf{y}) + \lambda\, r(\mathbf{x}), \tag{2}$$

where the minimizer corresponds to the recovered latent image. The role of the objective function is to quantify the quality of the solution and typically consists of two terms as shown in Eq. (2). The first term is the *data fidelity*, which measures the proximity of the solution to the observation, and the second one is the *regularizer*. The role of the regularizer is crucial since it encodes our prior knowledge by penalizing solutions that do not feature the desired properties. The parameter $\lambda \geq 0$, is used to combine the two terms and to adjust their contribution on the final result. Interestingly, the variational approach has direct links to Bayesian estimation methods and the derived solutions can be interpreted either as penalized maximum likelihood or as maximum a posteriori (MAP) estimates [2, 11].

As emphasized previously, a good choice for the regularizer is instrumental to the success of any variational-based image restoration method. A generic formulation that can be used to describe the majority of the most successful regularizers in the literature, is provided below

$$r(\mathbf{x}) = \sum_{k=1}^{K} \phi(\mathbf{L}_k \mathbf{x}), \tag{3}$$

where $\mathbf{L} : \mathbb{R}^N \mapsto \mathbb{R}^{K \times D}$ corresponds to the regularization operator ($\mathbf{L}_k \mathbf{x} \in \mathbb{R}^D$ denotes the D-dimensional k-th entry of the result obtained by applying $\mathbf{L}$ to the image $\mathbf{x}$) and $\phi : \mathbb{R}^D \mapsto \mathbb{R}$ is a potential function. Indeed, by varying the regularization operator $\mathbf{L}$ and the potential function ϕ we can derive several existing regularization functionals. Typical choices for the operator $\mathbf{L}$ are first or higher-order differential operators such as the gradient [4, 36], the structure tensor [26], the Laplacian and the Hessian [24, 27]. For the potential function there is also a wide variety of possible choices with the most popular ones being the ℓ_p vector norms and the Schatten matrix norms. The main reason for this is that their combination with linear operators leads to convex regularizers which are amenable to efficient optimization and provide certain convergence guarantees.

Besides the local regularization methods mentioned above[1], the definition of Eq. (3) can also describe non-local regularization functionals such as those in [10, 14, 20, 25, 44]. In this case, $\mathbf{L}$ is designed so that it allows interactions between distant points in the image domain. This way it is possible to capture long-range dependencies between image points and thus model the so called *non-local self similarity* (NLSS) property that natural images exhibit. This property implies that images typically consist of localized patterns that are repeated in different and possibly distant locations in the image domain. NLSS is an important property and if properly exploited it can effectively distinguish the image content from noise and other types of distortions. This

[1] These methods are considered local in the sense that the regularization operator is localized and its influence is restricted in a small area around the pixel of interest.

has been demonstrated for several image restoration problems [8, 14, 25].

2.2. Constrained Optimization

In the variational framework the choice of the regularizer has an important effect on the quality of the restored image. Equally important is our ability to efficiently compute the minimizer of the overall objective function. Image denoising under AWGN, amounts to solving an unconstrained optimization problem of the form :

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right), \quad (4)$$

where for the regularizer we use the generic description of Eq. (3), while for the fidelity term we use a quadratic cost, in accordance with the Gaussian noise assumption.

As mentioned earlier, λ is a 'free' parameter that needs to be tuned by the user and different values lead to different restoration results of varying image quality. Therefore, among others one of the main challenges is to choose the value for the regularization parameter λ, that will lead to the best possible result under some image quality criterion. Unfortunately, there is not a direct way to *a priori* relate the strength of λ with the quality of the result. Therefore, in practice, λ is either tuned empirically or heuristic techniques such as the L-curve method [16] are employed, which involve solving Problem (4) for several values of λ.

One way to circumvent this difficulty, is to consider the following equivalent formulation

$$\mathbf{x}^* = \arg\min_{\|\mathbf{y}-\mathbf{x}\|_2 \leq \varepsilon} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right), \quad (5)$$

which transforms the original problem to a constrained optimization form. Problems (4) and (5) are equivalent in the sense that : for any $\varepsilon > 0$ such that Problem (5) is feasible, a solution of (5) is either the null vector or else it is a solution of Problem (4) for some $\lambda > 0$ [34]. To answer what is that we gain by pursuing such a reformulation, we note that while in Eq. (5) there is still a free parameter ε that needs to be tuned, this parameter is directly related to the noise level distorting the latent image $\mathbf{x}$. In particular, we observe that it holds $\|\mathbf{y} - \mathbf{x}\|_2 = \|\mathbf{n}\|_2 \propto \sigma$. Given that there are several methods available for estimating the standard deviation of the noise from the noisy input [13, 29], we now have a good indication about the range of values that the parameter ε should lie in, as opposed to the previous formulation.

2.3. Minimization Strategy

To attack the minimization problem of Eq. (5) we can rely on a splitting variable technique such as the Alternating Direction Method of Multipliers [3]. Here, however, we opt for a simpler approach that utilizes a gradient-descent algorithm. To do so, we first rewrite Eq. (5) as

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{k=1}^{K} \phi\left(\mathbf{L}_k \mathbf{x}\right) + \iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right), \quad (6)$$

where

$$\iota_{\mathcal{C}(\mathbf{y},\varepsilon)}\left(\mathbf{x}\right) = \begin{cases} 0, & \text{if } \|\mathbf{y} - \mathbf{x}\|_2 \leq \varepsilon \\ \infty, & \text{otherwise} \end{cases}, \quad (7)$$

is the indicator function of the convex set $\mathcal{C}$.

Next, we assume that the potential function ϕ is smooth and thus we can compute its partial derivatives. Since this is not the case for the indicator function, instead of the gradient descent algorithm we employ the proximal gradient method (PGM) [33]. This is a gradient descent variant that can deal with functions consisting of both smooth and non-smooth terms. According to PGM the function $f\left(\mathbf{x}\right)$ to be minimized is split into two terms, a smooth and a non-smooth one. In our case we naturally have $f\left(\mathbf{x}\right) = r\left(\mathbf{x}\right) + \iota_{\mathcal{C}}\left(\mathbf{x}\right)$, where based on the smoothness assumption for the potential function ϕ, the regularizer $r\left(\mathbf{x}\right)$ corresponds to the smooth term. Then, the solution is computed in an iterative fashion, using the update rule

$$\mathbf{x}^t = \text{prox}_{\gamma^t \iota_{\mathcal{C}}}\left(\mathbf{x}^{t-1} - \gamma^t \nabla_{\mathbf{x}} r\left(\mathbf{x}^{t-1}\right)\right), \quad (8)$$

where γ^t is a step-size and $\text{prox}_{\gamma^t \iota_{\mathcal{C}}}$ is the proximal operator [33] related to the indicator function $\iota_{\mathcal{C}}$.

The proximal map of the indicator function $\iota_{\mathcal{C}}$ in Eq. (8) corresponds to an orthogonal projection of the input onto the set $\mathcal{C}$. This can be computed in closed form as

$$\Pi_{\mathcal{C}}\left(\boldsymbol{v}\right) = \mathbf{y} + \varepsilon \frac{\boldsymbol{v} - \mathbf{y}}{\max\left(\|\boldsymbol{v} - \mathbf{y}\|_2, \varepsilon\right)}. \quad (9)$$

Given that the gradient of the regularizer is computed as

$$\nabla_{\mathbf{x}} r\left(\mathbf{x}\right) = \sum_{k=1}^{K} \mathbf{L}_k^{\mathsf{T}} \psi\left(\mathbf{L}_k \mathbf{x}\right) \equiv h\left(\mathbf{x}\right), \quad (10)$$

with $\psi\left(\mathbf{z}\right) = \nabla_{\mathbf{z}} \phi\left(\mathbf{z}\right)$, $\mathbf{z} \in \mathbb{R}^D$ and using Eq. (9), we rewrite Eq. (8) as

$$\mathbf{x}^t = \Pi_{\mathcal{C}}\left(\mathbf{x}^{t-1} - h^t\left(\mathbf{x}^{t-1}\right)\right) \text{ with } h^t\left(\mathbf{x}\right) = \gamma^t h\left(\mathbf{x}\right). \quad (11)$$

A careful inspection of Eqs. (9) and (11) leads us to the useful observation that under this approach the solution is obtained by recursively subtracting from the input refined estimates of the noise realization that distorts it. In particular, for the first iteration and given that $\mathbf{x}^0 = \mathbf{y}$ we have

$$\mathbf{x}^1 = \mathbf{y} - \varepsilon \frac{h^1\left(\mathbf{y}\right)}{\max\left(\|h^1\left(\mathbf{y}\right)\|_2, \varepsilon\right)} = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^1\right). \quad (12)$$

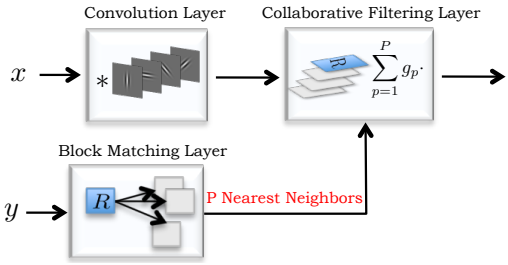


Figure 1. The architecture of the non-local filtering layer. The input of the layer is x while y is the network input based on which the block matching is computed.

Here $h^1(\mathbf{y})$ can be interpreted as an estimation module, which is equipped with a whitening mechanism so that it can infer from the input, the noise realization that distorts it. The noise realization estimate is further normalized to ensure that it has the correct variance and then it is subtracted from the noisy input. This leads to an output which consists of the latent image plus some residual noise, $\mathbf{n} - \mathbf{n}^1$. The subsequent updates refine the noise estimate and remove it from the original input as follows

$$\mathbf{x}^k = \mathbf{y} - \mathbf{n}^k = \mathbf{x} + \left(\mathbf{n} - \mathbf{n}^k\right), \quad k > 1 \tag{13}$$

where

$$\mathbf{n}^k = \varepsilon \frac{\mathbf{n}^{k-1} + h^k\left(\mathbf{x}^{k-1}\right)}{\max\left(\left\|\mathbf{n}^{k-1} + h^k\left(\mathbf{x}^{k-1}\right)\right\|_2, \varepsilon\right)}. \tag{14}$$

3. Proposed Network

From the previous analysis it is clear that the success of the iterative denoising scheme that we described in Section 2.3 depends exclusively on how well the function h, defined in Eq. (10), can estimate the realization of the noise. Designing such a function amounts to specifying the operator $\mathbf{L}$ and the gradient of the potential function ϕ. Manually selecting proper values for these parameters is a cumbersome task. For this reason, we pursue a machine learning approach and design a neural network that has the capacity to learn these parameters in a discriminative fashion from training data. In the same spirit with previous network architectures for restoration tasks [6,23,38], we consider each PGM update as a composition of network layers and construct our network as a cascade of such layers.

The remaining issue to be addressed is the parameterization of the operator $\mathbf{L}$ and function ψ in a way that will facilitate the learning of the network's parameters in an efficient and computationally tractable way.

3.1. Local and Non-local Operators

As mentioned in Section 2.1 common choices for the regularization operator $\mathbf{L}$ are local differential operators. In the discrete setting, image derivatives are typically computed as convolutions of the image with a filterbank consisting of several high-pass kernels. Naturally, this leads us to parametrize $\mathbf{L}$ as a convolutional layer, which is a widely used component in modern deep neural networks. One point however that requires our attention is that in order to learn a valid regularization operator, the filters utilized for its parametrization need to be zero-mean [7]. Moreover, since the function ψ will also be learned, if we inspect equation (10) we will notice that without further imposing a fixed scale to the operators $\mathbf{L}_k$, it is possible that two different sets of parameters $(\mathbf{L}_k, \psi)$ and $\left(\hat{\mathbf{L}}_k, \hat{\psi}\right)$ lead to the same result. Such a situation can occur if we choose $\hat{\mathbf{L}}_k = \beta \mathbf{L}_k$ and $\hat{\psi}(\beta \mathbf{z}) = \frac{1}{\beta}\psi(\mathbf{z})$, which gives $\mathbf{L}_k^\mathsf{T}\psi(\mathbf{L}_k\mathbf{x}) = \hat{\mathbf{L}}_k^\mathsf{T}\hat{\psi}\left(\hat{\mathbf{L}}_k\mathbf{x}\right)$. To ensure that the learned operators will satisfy both the zero-mean and the fixed-scale constraints, we parametrize the weights $\boldsymbol{w} \in R^L$ of each filter in the convolutional layer as

$$\boldsymbol{w} = s\left(\mathbf{v} - \bar{\mathbf{v}}\right) / \left\|\mathbf{v} - \bar{\mathbf{v}}\right\|, \tag{15}$$

where s is a scalar trainable parameter. Training the convolutional layers with this new parametrization can be done as usual using a stochastic gradient descent method. The gradient of the new parameters s and $\mathbf{v}$ w.r.t the loss function $\mathcal{L}$ can be computed as

$$\nabla_s \mathcal{L} = \langle \boldsymbol{w}/s, \nabla_{\boldsymbol{w}}\mathcal{L} \rangle \text{ and } \nabla_{\mathbf{v}}\mathcal{L} = \mathbf{M}_{\mathbf{v}}\nabla_{\boldsymbol{w}}\mathcal{L}, \tag{16}$$

where $\mathbf{M}_{\mathbf{v}} = \frac{s}{\|\mathbf{v}-\bar{\mathbf{v}}\|}\left(\mathbf{I} - \frac{\mathbf{1}\mathbf{1}^\mathsf{T}}{L}\right)\left(\mathbf{I} - \frac{(\mathbf{v}-\bar{\mathbf{v}})(\mathbf{v}-\bar{\mathbf{v}})^\mathsf{T}}{\|\mathbf{v}-\bar{\mathbf{v}}\|^2}\right)$, $\bar{\mathbf{v}}$ denotes the mean value of $\mathbf{v}$, and $\mathbf{1}$ is a column vector of ones.

Interestingly, the same parametrization of Eq. (15) but without the mean subtraction has been recently proposed in [37] as an alternative to batch-normalization [19]. However, in [37] the motivation is different and the aim is to make the training of deep networks more robust.

Besides the parametrization of $\mathbf{L}$ as a local operator, we further explore another option where we model $\mathbf{L}$ as a non-local operator. This leads to a second variant of the proposed network architecture. Our motivation for employing a non-local regularization operator is that the resulting network can take advantage of the NLSS property that we referred to in detail in the introduction. To this end, we adopt the parametrization that was proposed in [23]. Specifically, the non-local operator can be expressed as the composition of three layers : **1**) a convolution layer, where we use the

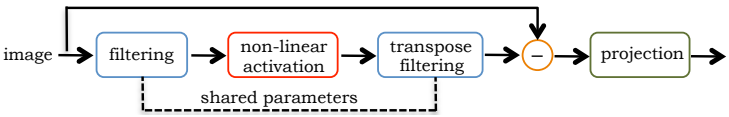


Figure 2. Schematic representation of the composite layer that serves as the core component of the proposed network architecture. Depending on the parametrization of the regularization operator, the filtering (transpose filtering) layer correspond to either a convolutional or a non-local filtering layer.

parametrization that we adopted earlier, that applies a linear transformation to every patch extracted from the image[2], **2**) a block-matching layer which forms a 3-D group for every valid patch and it consists of the P most similar patches to the reference patch (including the reference patch itself) and **3**) a collaborative-filtering layer that filters the grouped patches along the dimension of the group in a similar fashion as the Non-Local Means filter (NLM) [5]. In this last layer each 3-D group is projected to a single patch. A schematic representation of the described parametrization is provided in Fig. 1. This combination of layers leads to a non-local filtering approach similar to the one initially proposed in [8] but with two main differences. The first one is that the authors in [8] use fixed transforms while here the transforms are learned. The second one is that in the third step of the non-local operation an 1-D transformation along the group dimension is applied instead of the weighted sum that we use in this work. Therefore, in our case the output of the adopted non-local filtering layer leads to an output of the same size as the input, while in [8] the output is augmented in the third dimension by a factor of P. An additional remark relevant to the implementation of the described non-local filtering layer is that we constrain the weights of the third sub-layer to sum to one. This is consistent to the way that the NLM filter is defined. To impose this constraint we parametrize the weights $\boldsymbol{g} \in R^P$ as $\boldsymbol{g} = \nu^{-1}\boldsymbol{u}$ with $\nu = \langle \mathbf{1}\,,\, \boldsymbol{u} \rangle$. In this case the gradient of the new parameters $\boldsymbol{u}$ w.r.t the loss function $\mathcal{L}$ are computed as

$$\nabla_{\boldsymbol{u}} \mathcal{L} = \nu^{-1} \left(\mathbf{I} - \nu^{-1} \boldsymbol{u}^{\mathsf{T}} \right) \nabla_{\boldsymbol{g}} \mathcal{L}. \tag{17}$$

3.2. Parametrization of the Potential Function

Having defined the parametrization of the operator $\mathbf{L}$, we further need to model the function ψ (see Eq. (10)), which corresponds to the gradient of the potential function ϕ. To do so, first we assume that the potential function ϕ is separable, that is it can be expressed in the form

$$\phi\left(\mathbf{z}\right) = \sum_{d=1}^{D} \phi_d\left(\mathbf{z}_d\right), \tag{18}$$

and thus, $\psi\left(\mathbf{z}\right) = \left[\psi_1\left(\mathbf{z}_1\right)\ \psi_2\left(\mathbf{z}_2\right)\ \ldots\ \psi_D\left(\mathbf{z}_D\right)\right]^{\mathsf{T}} \equiv \nabla_{\mathbf{z}} \phi\left(\mathbf{z}\right)$, with $\psi_i\left(\mathbf{z}_i\right) = \partial \phi\left(\mathbf{z}\right) / \partial \mathbf{z}_i$. Next, we parametrize the partial derivatives ψ_i as a linear combination of Radial Basis Functions (RBFs), *i.e.*

$$\psi_i\left(x\right) = \sum_{j=1}^{M} \pi_{ij} \rho_j\left(\left|x - \mu_j\right|\right), \tag{19}$$

where π_{ij} are the expansion coefficients and μ_j are the centers of the basis functions ρ_j [17]. For our networks we use Gaussian RBFs, $\rho_j\left(r\right) = \exp\left(-a_j r^2\right)$, and we employ $M = 51$ Gaussian kernels whose centers are distributed equidistantly in the range [-100, 100] and they all share the same precision parameter a. To make sure that the input x lies in the specified range, a clipping layer is preceding the RBF-mixture layer. The representation of ψ_i using mixtures of RBFs is very powerful and allow us to approximate with high accuracy arbitrary non-linear functions. Details about the computation of the gradient of the parameters π_{ij} and of the input $\mathbf{z}$ w.r.t to the loss function $\mathcal{L}$ can be found in [23].

3.3. Trainable Projection Layer

The final component for the construction of the proposed network architecture is the projection layer, which is defined in Eq. (9). In this work we consider ε as a trainable parameter and we express it as $\varepsilon = e^{\alpha}\theta$ with $\theta = \sigma\sqrt{N \cdot C - 1}$, where σ is the standard deviation of the noise and $N \cdot C$ is the total number of pixel in the image.

Based on this parametrization and using the identity $\max\left(x, y\right) = 0.5\left(\left|x - y\right| + x + y\right)$, we compute the gradient of the input $\boldsymbol{v}$ w.r.t to the loss function $\mathcal{L}$ as

$$\nabla_{\boldsymbol{v}} \mathcal{L} = \varepsilon\gamma \left(\mathbf{I} - \beta^{+}\gamma^2 \left(\boldsymbol{v} - \mathbf{y}\right)\left(\boldsymbol{v} - \mathbf{y}\right)^{\mathsf{T}} \right) \nabla_{\boldsymbol{q}} \mathcal{L}, \tag{20}$$

where $\boldsymbol{q} = \Pi_{\mathcal{C}}\left(\boldsymbol{v}\right)$, $\beta^{+} = \left(1 + \operatorname{sign}\left(\left\|\boldsymbol{v} - \mathbf{y}\right\|_2 - \varepsilon\right)\right)/2$ and $\gamma = 1/\max\left(\left\|\boldsymbol{v} - \mathbf{y}\right\|_2, \varepsilon\right)$. Additionally, the gradient of the parameter α w.r.t the loss function $\mathcal{L}$ is computed as

$$\nabla_{\alpha} \mathcal{L} = \mu \left(\boldsymbol{v} - \mathbf{y}\right)^{\mathsf{T}} \nabla_{\boldsymbol{q}} \mathcal{L}, \tag{21}$$

where $\beta^{-} = \left(1 - \operatorname{sign}\left(\left\|\boldsymbol{v} - \mathbf{y}\right\|_2 - \varepsilon\right)\right)/2$ and $\mu = \varepsilon\gamma\left(1 - \varepsilon\gamma\beta^{-}\right)$. Note that for all the formulas above we are using the convention $\operatorname{sign}\left(0\right) = -1$.

4. Network Training

We train our networks for grayscale and color image denoising under i.i.d Gaussian noise. Each network consists of a cascade of S composite layers, as the one shown in Fig. 2, plus an additional clipping layer placed just before the output of the network. This last layer incorporates our prior knowledge about the valid range of image intensities and forces the pixel values of the restored image to lie in the range $[0, 255]$. The network parameters $\boldsymbol{\Theta} = \left[\boldsymbol{\Theta}^1, \ldots, \boldsymbol{\Theta}^S\right]$, where $\boldsymbol{\Theta}^t = \{s^t, \mathbf{v}^t, \boldsymbol{g}^t, \boldsymbol{\pi}^t, \alpha^t\}$[3] denotes the set of parameters for the t-th layer, are learned using a loss-minimization strategy given Q pairs of training data $\{\mathbf{y}_q, \mathbf{x}_q\}_{q=1}^{Q}$. Here $\mathbf{y}_q$ is a noisy input and $\mathbf{x}_q$ is the corresponding ground-truth image. To achieve an increased capacity for the network, we learn different parameters for each composite layer. However, the convolutional and non-local filtering layers (for the local and non-local version of

[2] Passing an image through a convolution layer of F filters whose support are $H \times W$, corresponds to applying a linear mapping $\mathbb{R}^{H \times W} \mapsto \mathbb{R}^F$ to every image patch of size $H \times W$. In addition, the stride of the convolutional layer determines the overlap between consecutive image patches.

[3] For the local variants of the proposed network, the parameters $\boldsymbol{g}^t$ are not present in the parameter set $\boldsymbol{\Theta}^t$.

(a) (b) (e) (d) (e)

Figure 3. Grayscale image denoising. (a) Original image, (b) Noisy image (AWGN with $\sigma = 20$) ; PSNR $= 22.10$ dB. (c) Denoised image using EPLL [45] ; PSNR $= 31.54$ dB. (d) Denoised image using DCNN [42] ; PSNR $= 31.83$ dB. (e) Denoised image using UNet_5; PSNR $= 31.71$ dB.

(a) (b) (e) (d) (e)

Figure 4. Color image denoising. (a) Original image, (b) Noisy image (AWGN with $\sigma = 30$) ; PSNR $= 18.57$ dB. (c) Denoised image using CBM3D [8] ; PSNR $= 28.55$ dB. (d) Denoised image using DCNN [42] ; PSNR $= 29.08$ dB. (e) Denoised image using CUNLNet_5; PSNR $= 29.13$ dB.

the network, respectively) in each composite layer share the same parameters $\{s^t, \mathbf{v}^t, \boldsymbol{g}^t\}$ with their transpose layers.

Since the objective function to be minimized is non-convex, to avoid getting stuck in poor local-minima we initialize our networks with the parameters that are learned following a greedy-training strategy. The same approach has been adopted in [6, 23, 38] and it amounts to learning the parameters of each composite layer by keeping all the preceding layers of the network fixed and minimizing the cost

$$\mathcal{L}\left(\boldsymbol{\Theta}^t\right) = \sum_{q=1}^{Q} \ell\left(\hat{\mathbf{x}}_q^t, \mathbf{x}_q\right). \tag{22}$$

In Eq. (22), $\hat{\mathbf{x}}_q^t$ is the output of the t-th composite layer and the loss function ℓ corresponds to the negative peak signal-to-noise-ratio (PSNR). This is computed as $\ell(\mathbf{y}, \mathbf{x}) = -20\log_{10}\left(p/\left\|\mathbf{y}-\mathbf{x}\right\|_2\right)$, where $p = 255\sqrt{N \cdot C}$. While these learned parameters are sub-optimal, we have experimentally observed that they serve as a good initialization for the joint optimization training that follows.

To minimize the objective function in Eq. (22) w.r.t the parameters $\boldsymbol{\Theta}^t$ we employ the Adam algorithm [21], which is a variant of the stochastic gradient descent (SGD) that involves adaptive normalization of the learning rate. Each layer is trained for 100 epochs using an initial learning rate 1e-2 (1e-3 for grayscale images), while the configuration parameters for Adam are chosen as `beta_1` = 0.9, `beta_2` = 0.999 and `eps` = 1e-4.

The final parameters of our network are obtained by using the previous learned parameters as initial values and by jointly minimizing the objective function

$$\mathcal{L}\left(\boldsymbol{\Theta}\right) = \sum_{q=1}^{Q} \ell\left(\hat{\mathbf{x}}_q^S, \mathbf{x}_q\right), \tag{23}$$

w.r.t to all network parameters $\boldsymbol{\Theta}$. This cost function does not take into account anymore the intermediate results (outputs of each composite layer) but only depends on the final output of the network $\hat{\mathbf{x}}_q^S$. In this case the training is performed by running 100 epochs using Adam optimization with the same configuration parameters as before.

5. Experiments and Results

To train our local and non-local models we generated the training data using the Berkeley segmentation dataset (BSDS) [31], which consists of 500 images. We split these images in two sets, a training set which consists of 400 images and the validation set which consists of the remaining 100 images. All the images were randomly cropped so that their size is 180×180 pixel. We note that the 68 BSDS images of [35] that are used for the comparisons reported in Tables 1 and 2 are strictly excluded from the training set and only cropped versions of them are used in the validation set. The proposed models were trained on a NVIDIA 1080 Ti GPU and the software we used for training and testing was built on top of MatConvnet [39].

Grayscale denoising Following the strategy described in Section 4, we have trained two variants of our proposed network. In the first network, we parametrize the regularization operator $\mathbf{L}$ as a local operator and in the second one as a non-local operator. Both networks consist of $S = 5$ composite layers each and we will refer to them as UNet_5 and UNLNet_5, respectively. For the local model, in order to parametrize the operator $\mathbf{L}$, in each composite layer we employ a convolution layer of 48 filters, which are zero-mean, fixed-norm and have a support 7×7. For the non-local model, instead of the convolution layer we utilize

Methods	Noise level - σ (std.)											
	5	10	15	20	25	30	35	40	45	50	55	avg.
BM3D [8]	37.57	33.30	31.06	29.60	28.55	27.74	27.07	26.45	25.99	25.60	25.26	28.93
EPLL [45]	37.55	33.36	31.18	29.73	28.67	27.84	27.16	26.58	26.09	25.71	25.34	29.02
WNMM [15]	37.76	33.55	31.31	29.83	28.73	27.94	27.28	26.72	26.26	25.85	25.49	29.16
DCNN [42]	37.68	33.72	31.60	30.19	29.15	28.33	27.66	27.10	26.62	26.21	25.80	29.46
UNet_5	37.59	33.54	31.38	29.93	28.84	28.01	27.38	26.85	26.38	25.95	25.53	29.22
UNLNet_5	37.62	33.62	31.47	30.04	28.96	28.13	27.50	26.96	26.48	26.04	25.64	29.32
$\mathrm{UNLNet}_5^{\mathrm{orc}}$	37.79	33.97	31.95	30.59	29.51	28.54	27.97	27.47	26.97	26.41	25.80	29.72

Table 1. Grayscale denoising comparisons for different noise levels over the standard set of 68 [35] Berkeley images. Performance is measured in terms of average PSNR (in dB). The highlighted results refer to those of the non-local model with oracle grouping.

Noise	Methods				
σ (std.)	CBM3D [8]	CDCNN [42]	CUNet_5	$\mathrm{CUNLNet}_5$	$\mathrm{CUNLNet}_5^{\mathrm{orc}}$
5	40.24	40.11	40.31	40.39	40.54
10	35.88	36.11	36.08	36.20	36.70
15	33.49	33.88	33.78	33.90	34.58
20	31.88	32.36	32.21	32.34	33.11
25	30.68	31.22	31.03	31.17	31.95
30	29.71	30.31	30.06	30.24	31.06
35	28.86	29.57	29.37	29.53	30.37
40	28.06	28.94	28.77	28.91	29.75
45	27.82	28.39	28.23	28.37	29.19
50	27.36	27.91	27.74	27.89	28.65
55	26.95	27.45	27.27	27.44	28.10
avg.	30.99	31.48	31.35	31.49	32.18

Table 2. Color denoising comparisons for different noise levels over the standard set of 68 [35] Berkeley images. Performance is measured in terms of average PSNR (in dB). The highlighted results refer to those of the non-local model with oracle grouping.

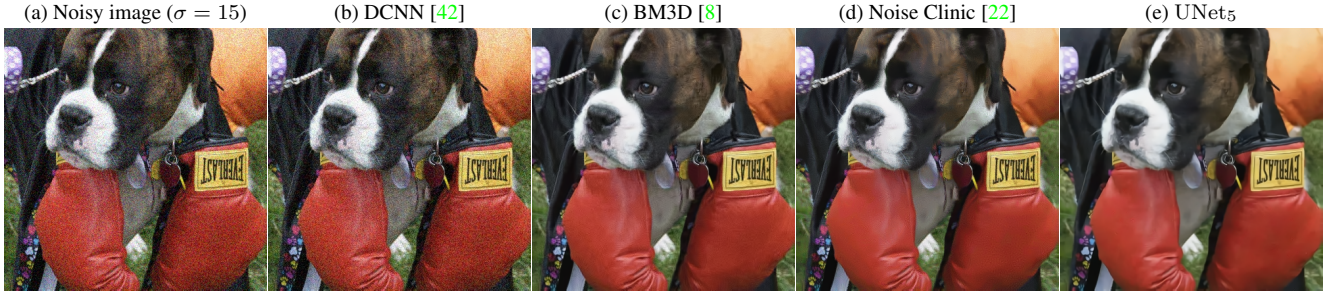
the non-local filtering layer as described in Section 3.1. In this case, similar to the local network, we utilize 48 filters of 7×7 support. As explained in Section 3.1 this corresponds to applying a non-redundant linear transformation, $\mathcal{T} : \mathbb{R}^{7\times 7} \mapsto \mathbb{R}^{48}$ on every image patch of size 7×7 extracted from the input image, excluding the DC component (the low-pass content) of the transform. Finally, in order to form the group of similar patches as required by the second step of the non-local filtering layer, we use the $P = 8$ closest neighbors (including the reference patch) while the similar patches are searched on the noisy input of the network in a window of 31×31 centered around each pixel. The same group indices are then used for all the composite layers of the network.

We trained two UNet_5 and UNLNet_5 networks, one for low input noise levels ($\sigma < 30$) and one for high input noise levels ($30 \leq \sigma < 55$). For the low-noise network training, the training data were distorted with AWGN of standard deviation that varies from $\sigma = 5$ to $\sigma = 29$ with increments of 4 and for the high-noise network with AWGN of standard deviation that varies from $\sigma = 30$ to $\sigma = 55$ using the same increments. To evaluate the restoration performance of our proposed networks, in Table 1 we report comparisons with recent state-of-the-art denoising methods on the standard evaluation dataset of 68 images [35] for eleven different noise levels, where the standard deviation of the noise varies from $\sigma = 5$ to $\sigma = 55$ with increments of 5.

From these results we observe that the proposed networks perform better than all non-deep learning methods but they are outperformed by the deep network (DCNN) of [42]. Specifically, on average our local model UNet_5 leads to results that are 0.25 dB worse than DCNN, while our non-local network UNLNet_5 performs better than the local one but still falls behind DCNN around 0.15 dB on average. Nevertheless, the memory footprint of the proposed networks is about fourteen times smaller than that of DCNN (48K versus 666K parameters), which makes them ideal for deployment in mobile devices where memory storage is limited. More importantly, as we demonstrate later, our models show an excellent denoising performance under more realistic noise conditions, as opposed to DCNN which performs poorly. In Table 1 we further report the results obtained by our non-local network when the indices of similar patches are computed from the ground-truth images. In this case we observe that $\mathrm{UNLNet}_5^{\mathrm{orc}}$ outperforms DCNN and leads to an average increase of 0.25 dB. While this is not a practical scenario to consider, these results highlight the fact that a better grouping approach, which is out of the scope of the current work, can lead to further improvements in the restoration quality without any need to re-train the network. Representative grayscale denoising results that demonstrate visually the restoration quality of the proposed models are shown in Fig. 3.

Color denosing Similar to the grayscale case we trained two different network configurations, one using a local operator and one using a non-local operator. The only difference between the color denoising network architecture and the grayscale one is that the convolution layers used in the color-denoising networks consist of 74 filters of support $5 \times 5 \times 3$. The rest of the network parameters and the training setup remains the same as above. In Table 2 we report results for several noise levels and our compar-

(a) Noisy image ($\sigma = 15$) (b) DCNN [42] (c) BM3D [8] (d) Noise Clinic [22] (e) UNet_5



(a) Noisy image ($\sigma = 22$) (b) DCNN [42] (c) CBM3D [8] (d) Noise Clinic [22] (e) CUNLNet_5

Figure 5. Real grayscale and color image denoising. The details in the results are better seen by zooming in on a computer screen.

isons involve only methods that are specifically designed to handle color images. We refrain from reporting results obtained by grayscale methods applied on each image channel independently, since these results are not competitive to the ones obtained from methods specifically designed for color denoising.

From Table 2 we observe that our color networks outperform CBM3D [8], which has been the state-of-the-art method for almost a decade, by 0.35 dB for the local model and 0.5 dB for the non-local model and they are very competitive to CDCNN [42] which is the current state-of-the-art. Specifically, the proposed non-local network on average matches the performance of DCNN while it is considerably more shallow with 7 times less parameters (93K versus 668K). Another important advantage of the proposed networks, as it will be demonstrated next, is that similarly to the grayscale case they perform very well when the noise distorting the input is not AWGN, as opposed to DCNN that cannot handle successfully such cases. For a visual inspection of the restoration performance of the proposed color models we refer to Fig. 4.

Results on real images To demonstrate the practical significance of the proposed network architecture, we further report representative results on images obtained from [22], which are distorted by real noise and whose distribution and noise level are unknown. Since ground-truth images are not available, the evaluation of the different methods is only possible by visual comparisons. From Fig. 5 we observe that as opposed to the rest of the methods, DCNN method is completely incapable of removing the noise and of improving the image quality. Regarding the performance of the proposed networks, they lead to visually pleasing results with most of the noise being removed and without introducing any spurious artifacts, as those present in the rest of the methods under comparison. More results on real images can be found in the Appendix.

6. Conclusions and Future Work

In this work we proposed a novel network architecture for grayscale and color image denoising. The design of the resulting image models has been inspired by local and non-local variational methods and a constrained optimization formulation of the problem, which allows us to train our networks for a wide range of noise levels using a single set of parameters. While the architecture of the proposed networks is considerably more shallow than current state-of-the-art deep CNN-based approaches, the resulting models lead to very competitive results for AWGN distortions while they also appear to be very robust when the noise degrading the input deviates from the Gaussian assumption.

Based on the reported results using oracle grouping, a promising future research direction that has the potential to lead to further improvements in the restoration quality is to investigate different block-matching approaches for finding the similar patches used in the non-local variant of the proposed network. Another direction that we plan to explore is the use of the proposed networks as sub-solvers in restoration methods that deal with more general inverse imaging problems such as deblurring, inpainting, demosaicking, *etc*.

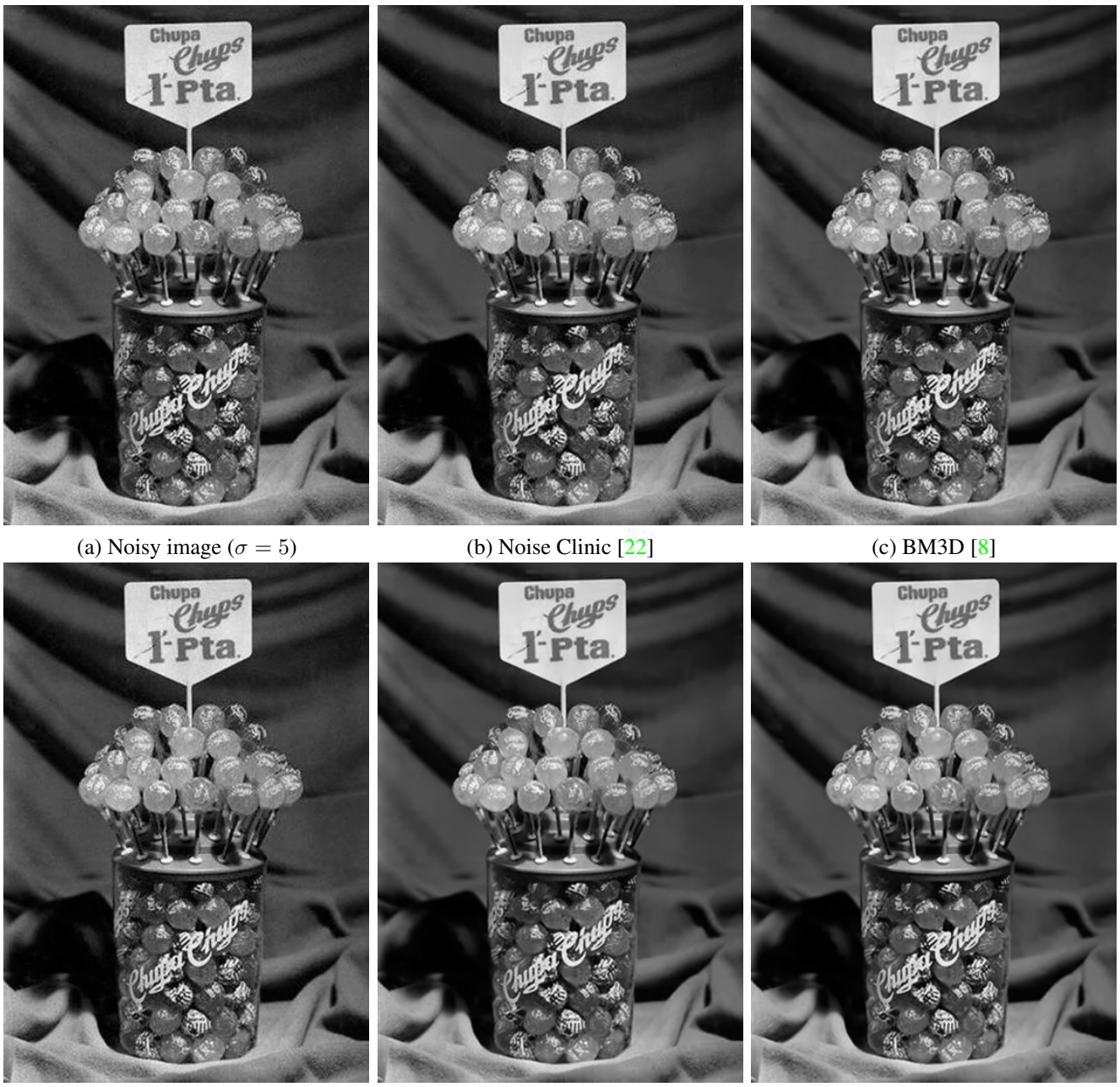
(a) Noisy image ($\sigma = 8$) (b) Noise Clinic [22] (c) BM3D [8]

(d) DCNN [42] (e) UDNet_5 (f) UNLDNet_5

Figure 6. Real grayscale image denoising. **Images are best viewed magnified on a computer screen.**

7. Appendix : Additional results on real images

In Figs. 6-10 we provide additional grayscale and color image denoising results on images that have been distorted by real noise, whose level and distribution are unknown. Further, these images are quantized and their values are in the range [0, 255]. All the images are publicly available and were obtained from [22], except to Fig. 6 which is available from https://en.wikipedia.org/wiki/David_Hilbert. In our reported results we compare 5 different methods that are all applicable both to grayscale and color images. In particular, we consider the method proposed in [22], which the authors refer to as "noise clinic" and it was developed so that it can be adapted to any signal dependent colored noise, the BM3D algorithm [8], which has been the state-of-the-art Gaussian denoising method for almost a decade and still leads to very competitive results, DCNN [42], which is a deep learning method that achieves the current state-of-the-art performance in Gaussian denoising, and the two variants (local and non-local) of our proposed denoising network. Since ground-truth images do not exist, we cannot provide any quantitative comparisons and the evaluation of the dif-



(a) Noisy image ($\sigma = 5$) (b) Noise Clinic [22] (c) BM3D [8]

(d) DCNN [42] (e) UDNet_5 (f) $\mathrm{UNLDNet}_5$

Figure 7. Real grayscale image denoising. **Images are best viewed magnified on a computer screen.**

ferent methods is only possible by a visual comparison of their restoration results. It is also worth mentioning that all the methods under comparison but the "noise clinic" have been originally designed to deal with additive white Gaussian noise (AWGN). Therefore, the main goal of our comparisons is to assess how robust each method is when the noise deviates significantly from the assumed noise model. Finally, we note that the noise clinic method and the DCNN network are equipped with an internal mechanism to estimate the noise level. On the other hand, the BM3D algorithm and our proposed networks apart from the noisy input, they accept a second input argument which corresponds to the standard deviation of the noise, σ. For these three methods, in our comparisons we have chosen empirically the value of σ (we indicate this value in the caption of each image) that led to the best restoration results.

(a) Noisy image ($\sigma = 25$) (b) Noise Clinic [22] (c) CBM3D [8]

(d) CDCNN [42] (e) CUDNet_5 (f) CUNLDNet_5

Figure 8. Real color image denoising. **Images are best viewed magnified on a computer screen.**

References

[1] F. J. Anscombe. The transformation of Poisson, binomial and negative-binomial data. *Biometrika*, 35(3):246–254, 1948. 2

[2] M. Bertero and P. Boccacci. *Introduction to Inverse Problems in Imaging*. IOP Publishing, 1998. 2

[3] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Now Publishers, 2011. 2, 3

[4] K. Bredies, K. Kunisch, and T. Pock. Total generalized variation. *SIAM J. Imaging Sci.*, 3:492–526, 2010. 2

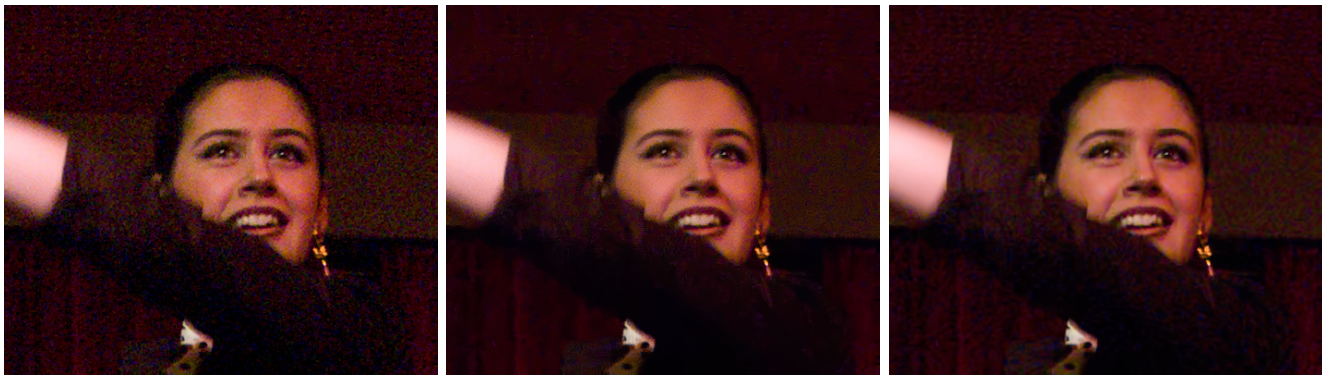
[5] A. Buades, B. Coll, and J.-M. Morel. Image denoising methods. A new nonlocal principle. *SIAM review*, 52:113–147, 2010. 5

[6] Y. Chen and T. Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE Trans. Pattern Anal. Mach. Intell*, 39:1256–1272, 2017. 1, 4, 6

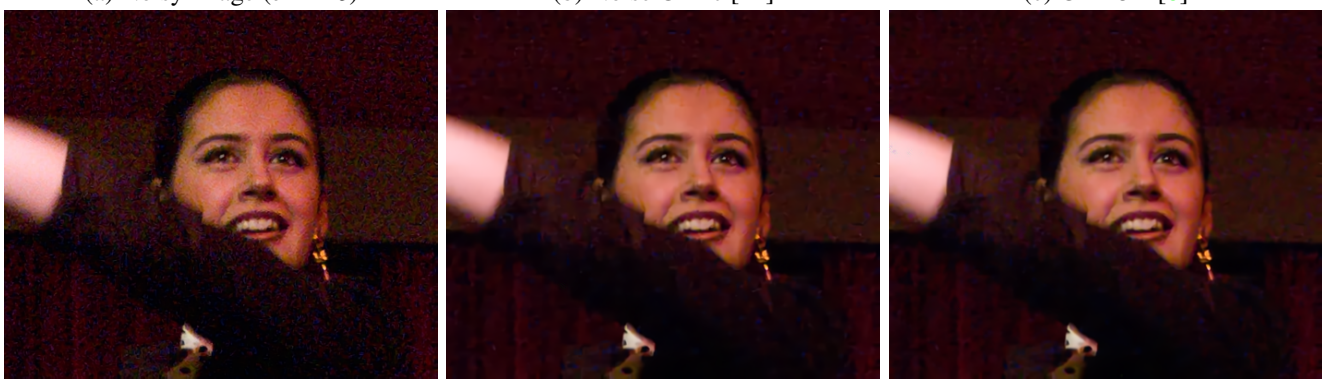
[7] Y. Chen, R. Rantfl, and T. Pock. Insights into analysis operator learning: From patch-based sparse models to higher order MRFs. *IEEE Trans. Image Process.*, 23:1060–1072, 2014. 4

[8] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007. 3, 5, 7, 8, 9, 10

[9] DxO Labs. Using DxO Prime. http://www.dxo.com/us/photography/photo-software/dxo-photolab, 2017. 1



(a) Noisy image ($\sigma = 15$) (b) Noise Clinic [22] (c) CBM3D [8]



(d) CDCNN [42] (e) CUDNet_5 (f) $\mathrm{CUNLDNet}_5$

Figure 9. Real color image denoising. **Images are best viewed magnified on a computer screen.**

(a) Noisy image ($\sigma = 12$) (b) Noise Clinic [22] (c) CBM3D [8]

(d) CDCNN [42] (e) CUDNet_5 (f) $\mathrm{CUNLDNet}_5$

Figure 10. Real color image denoising. Some important differences in the restoration quality between the methods under comparison can be spotted in the highlighted image region. **Images are best viewed magnified on a computer screen.**

[10] A. Elmoataz, O. Lezoray, and S. Bougleux. Nonlocal discrete regularization on weighted graphs: a framework for

image and manifold processing. *IEEE Trans. Image Proces.*, 17:1047–1060, 2008. 2

[11] M. Figueiredo, J. Bioucas-Dias, and R. Nowak. Majorization–minimization algorithms for wavelet-based image restoration. *IEEE Trans. Image Process.*, 16:2980–2991, 2007. 2

[12] M. Fisz. The limiting distribution of a function of two independent random variables and its statistical application. *Colloquium Mathematicum*, 3:138–146, 1955. 2

[13] A. Foi. Clipped noisy images: Heteroskedastic modeling and practical denoising. *Signal Processing*, 89(12):2609–2629, 2009. 3

[14] G. Gilboa and S. Osher. Nonlocal operators with applications to image processing. *Multiscale Model. Simul.*, 7:1005–1028, 2008. 2, 3

[15] S. Gu, L. Zhang, W. Zuo, and X. Feng. Weighted nuclear norm minimization with application to image denoising. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 2862–2869, 2014. 8

[16] P. C. Hansen. The L-curve and its use in the numerical treatment of inverse problems. In *Computational Inverse Problems in Electrocardiology, Advances in Computational Bioengineering*, pages 119–142. WIT Press, 2000. 3

[17] Y. H. Hu and J.-N. Hwang. *Handbook of neural network signal processing*. CRC press, 2001. 5

[18] D. Hunter and K. Lange. A tutorial on MM algorithms. *The American Statistician*, 58:30–37, 2004. 2

[19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456. PMLR, 2015. 5

[20] S. Kindermann, S. Osher, and P. W. Jones. Deblurring and denoising of images by nonlocal functionals. *Multiscale Model. Simul.*, 4:1091–1115, 2005. 2

[21] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[22] M. Lebrun, M. Colom, and J.-M. Morel. The noise clinic: a blind image denoising algorithm. *Image Processing On Line*, 5:1–54, 2015. 7, 8, 9, 10

[23] S. Lefkimmiatis. Non-local color image denoising with convolutional neural networks. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 3587–3596, 2017. 1, 4, 5, 6

[24] S. Lefkimmiatis, A. Bourquard, and M. Unser. Hessian-based norm regularization for image restoration with biomedical applications. *IEEE Trans. Image Process.*, 21(3):983–995, 2012. 2

[25] S. Lefkimmiatis and S. Osher. Non-local Structure Tensor functionals for image regularization. *IEEE Trans. Comput. Imaging*, 1:16–29, 2015. 2, 3

[26] S. Lefkimmiatis, A. Roussos, P. Maragos, and M. Unser. Structure tensor total variation. *SIAM J. Imaging Sci.*, 8:1090–1122, 2015. 2

[27] S. Lefkimmiatis, J. Ward, and M. Unser. Hessian Schatten-norm regularization for linear inverse problems. *IEEE Trans. Image Process.*, 22(5):1873–1888, 2013. 2

[28] A. Levin and B. Nadler. Natural image denoising: Optimality and inherent bounds. In *IEEE Conf. Comput. Vision and Patt. Recogn. (CVPR)*, pages 2833–2840. IEEE, 2011. 1

[29] X. Liu, M. Tanaka, and M. Okutomi. Single-image noise level estimation for blind denoising. *IEEE Trans. Image Process.*, 22(12):5226–5237, 2013. 3

[30] M. Mäkitalo and A. Foi. Noise parameter mismatch in variance stabilization, with an application to poisson–gaussian noise estimation. *IEEE Trans. Image Process.*, 23(12):5348–5359, 2014. 2

[31] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. IEEE Int. Conf. Computer Vision*, pages 416–423, 2001. 6

[32] Neat Image team. Neat Image. https://ni.neatvideo.com/user-guides/ni8/NISLMUG.pdf, 2017. 1

[33] N. Parikh and S. Boyd. *Proximal Algorithms*. Now Publishers, 2013. 2, 3

[34] R. T. Rokcafellar. *Convex Analysis*. Princeton, NJ: Princeton Univ. Press, 1970. 3

[35] S. Roth and M. J. Black. Fields of experts. *International Journal of Computer Vision*, 82(2):205–229, 2009. 6, 7, 8

[36] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992. 2

[37] T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29*, pages 901–909. 2016. 5

[38] U. Schmidt and S. Roth. Shrinkage fields for effective image restoration. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 2774–2781, 2014. 1, 4, 6

[39] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015. 6

[40] R. Vemulapalli, O. Tuzel, and M.-Y. Liu. Deep Gaussian conditional random field network: A model-based deep network for discriminative denoising. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, pages 4801–4809, 2016. 1

[41] C. R. Vogel. *Computational Methods for Inverse Problems*. SIAM, 2002. 2

[42] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Process.*, 2017. 1, 7, 8, 9, 10

[43] K. Zhang, W. Zuo, S. Gu, and L. Zhang. Learning deep CNN denoiser prior for image restoration. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, July 2017. 1

[44] D. Zhou and B. Schölkopf. Regularization on discrete spaces. In *Pattern Recognition*, pages 361–368. Springer, 2005. 2

[45] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *Proc. IEEE Int. Conf. Computer Vision*, pages 479–486. IEEE, 2011. 7, 8