Deep Unfolding of a Proximal Interior Point Method for Image Restoration

Carla Bertocchi, Emilie Chouzenoux, Marie-Caroline Corbineau,
Jean-Christophe Pesquet, Marco Prato

Deep Unfolding of a Proximal Interior Point Method for Image Restoration†

C. Bertocchi[1], E. Chouzenoux[2], M.-C. Corbineau[2], J.-C. Pesquet[2], and M. Prato[1]

[1] Università di Modena e Reggio Emilia, Modena, Italy
[2] CVN, CentraleSupélec, INRIA Saclay, Université Paris-Saclay, Gif-Sur-Yvette, France

E-mail: carla.bertocchi@unimore.it, emilie.chouzenoux@centralesupelec.fr, marie-caroline.corbineau@centralesupelec.fr, jean-christophe.pesquet@centralesupelec.fr, marco.prato@unimore.it

Abstract. Variational methods are widely applied to ill-posed inverse problems for they have the ability to embed prior knowledge about the solution. However, the level of performance of these methods significantly depends on a set of parameters, which can be estimated through computationally expensive and time-consuming methods. In contrast, deep learning offers very generic and efficient architectures, at the expense of explainability, since it is often used as a black-box, without any fine control over its output. Deep unfolding provides a convenient approach to combine variational-based and deep learning approaches. Starting from a variational formulation for image restoration, we develop iRestNet, a neural network architecture obtained by unfolding a proximal interior point algorithm. Hard constraints, encoding desirable properties for the restored image, are incorporated into the network thanks to a logarithmic barrier, while the barrier parameter, the stepsize, and the penalization weight are learned by the network. We derive explicit expressions for the gradient of the proximity operator for various choices of constraints, which allows training iRestNet with gradient descent and backpropagation. In addition, we provide theoretical results regarding the stability of the network for a common inverse problem example. Numerical experiments on image deblurring problems show that the proposed approach compares favorably with both state-of-the-art variational and machine learning methods in terms of image quality.

Keywords: Interior point method, proximal algorithms, deep unfolding, neural network, image restoration, regularization.

† Contact author: M.-C. Corbineau, marie-caroline.corbineau@centralesupelec.fr.

1. Introduction

In this work we focus on inverse problems related to the following model:

$$y = \mathcal{D}(H\overline{x}), \tag{1}$$

where $y \in \mathbb{R}^m$ is the observed data, $\overline{x} \in \mathbb{R}^n$ is the sought signal or image, $H \in \mathbb{R}^{m \times n}$ is the observation operator, which is assumed linear, and $\mathcal{D}$ is the noise perturbation operator. The linear operator H is assumed to be known from a physical model or prior identification step [1, 2]. In this context, both variational and deep learning approaches provide efficient methods for delivering an estimate of $\overline{x}$, while offering different benefits and drawbacks, which are discussed hereafter.

In order to find an appropriate solution to an ill-posed inverse problem like (1), variational methods incorporate prior information on the sought variable $\overline{x}$, through constraints or regularization functions, such as the total variation and its various extensions [3] or sparsity-promoting functions [4]. This leads to the following minimization problem,

$$\min_{x \in \mathcal{C}} \; f(Hx, y) + \lambda \mathcal{R}(x) \tag{2}$$

where $f : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ is a data-fidelity function, which is convex with regards to its first variable and which is related to the degradation model, $\mathcal{R} : \mathbb{R}^n \to \mathbb{R}$ is a convex regularization function, $\lambda \in]0, +\infty[$ is a regularization parameter and $\mathcal{C}$ is a subset of $\mathbb{R}^n$. Although useful, this approach is sometimes limited by its complexity: solving (2) may require advanced algorithms that may be too slow for real-time applications. In addition, λ is a parameter that needs to be set and $\mathcal{R}$ is usually parametrized by one or several parameters, whose optimal choice may strongly depend on the data at hand. These parameters are often tuned manually or computed using, for instance, cross validation, the discrepancy principle [5], or methods based on Stein unbiased risk estimates (SURE) [6]. However, these methods are often time-consuming and their success is not always guaranteed. Furthermore, despite numerous efforts in designing sophisticated models, solving (2) does not necessarily lead to the best estimate for $\overline{x}$, hence the development of early stopping methods, where the iterative procedure is stopped before convergence [7]. Finding the optimal stopping time depends on the algorithm and requires the use of an oracle such as SURE, which may explain why these techniques are currently restricted to relatively simple cost functions.

Deep Neural Networks (DNNs), and in particular Convolutional Neural Networks (CNNs), provide good performance for various applications related to inverse problems, such as denoising [8], non-blind and blind deblurring [9, 10, 11], super-resolution [12], or CT reconstruction [13]. As detailed in [14], DNNs for inverse problems are very often preceded by a pre-processing step. Indeed, a rough estimation of $\overline{x}$ can be found by using the inverse or pseudoinverse of H. The latter tends, however, to strongly amplify noise. Hence, in this context, DNNs are used as denoisers and artifact-removers. However, since prior knowledge about its output can hardly be incorporated into a DNN, which in most of the cases is viewed as a black-box, the explainability and reliability [15] of such methods could be questioned. Furthermore, the pre-processing step, in itself, can include a penalty, thus amounting to solving a problem of the form (2), where the regularization weight strongly depends on the noise level, *e.g.* [10, 16]. One straightforward way to combine the benefits of both

variational-based methods and DNNs is to unfold an iterative method and untie the parameters of both the model and the algorithm across the layers of the network [17]. Interestingly, the fact that this approach makes use of a limited number of layers can be viewed as an analogue of early stopping methods. It is however worth mentioning that, in unfolded algorithms, the number of iterations (i.e., layers) is tuned during the off-line training step and is then fixed for all test images, which differs from early stopping strategies where the iteration number usually differs for each processed image.

In this paper, we propose a novel neural network architecture called iRestNet, which is obtained by unfolding a proximal interior point algorithm over a finite number of iterations. One key feature of this algorithm is that it produces only feasible iterates thanks to a logarithmic barrier. This barrier enables prior knowledge to be directly incorporated into iRestNet and, as opposed to a projection onto $\mathcal{C}$, it allows differentiation and gradient backpropagation throughout the network. Hence, gradient descent can be used for training. The stepsize, barrier parameter, and regularization weight are untied across the network and learned for each layer. Thus, once the network has been trained, its application on test images requires only a short execution time per image without any parameter search, as opposed to traditional variational methods.

Related works apply deep unfolding to probabilistic models, such as Markov random fields [17], topic models [18], and to different algorithms like primal-dual solvers [19] or the proximal gradient method [20, 21]. Classic optimization algorithms can be unfolded to perform many different tasks in image processing. For instance, FISTA and ISTA can be unfolded to perform sparse coding [22, 23], while the same ISTA and ADMM can be unrolled for image compressive sensing [24, 25]. However, in the aforementioned works, some functions and operators are learned, which weakens the link between the resulting network and the original algorithm. Deep unfolding is also used to learn shrinkage functions, which can be viewed as proximity operators of sparsity-promoting functions [26, 27], or to optimize hyperparameters in nonlinear reaction diffusion models [28]. Several recent works consider replacing handcrafted algorithms by learned iterative methods [29, 30]. In these approaches, the goal is to find the minimizer of a given objective function, whereas in the proposed method, the architecture is inspired by an optimization strategy applied to the minimization of an objective function. Hence, a better indicator of perceptual quality can be optimized during the training step. Only a few works so far have considered combining interior point methods (IPMs) with deep learning. Every layer of the network from [31] solves a small quadratic problem using an IPM, while in [32], hard constraints are enforced on weights by using the logarithmic barrier function during training. More recently, an interior point strategy was used to design a recurrent network, whose purpose is to solve a specific convex constrained problem [33]. In our case, the proposed network is not trained to output a minimizer of the constrained problem from which its architecture is inspired. Instead, a direct evaluation of the reconstruction error is used during training. In addition, iRestNet appears to have more flexibility since the regularization weight can vary among layers.

To the best of our knowledge, this paper presents the first architecture corresponding to a deep unfolded version of an interior point algorithm with untied stepsize and regularization parameter. As opposed to other unfolding methods like [20, 21], the proximity operator and the regularization term are kept explicit, which establishes a

direct relation between the original algorithm and the network. Other contributions of this work include the expression of the required proximity operator, and of its corresponding gradient, for three standard variational formulations, along with numerical experiments demonstrating the benefit of using the proposed approach over other machine learning and variational methods for image deblurring.

This paper is organized as follows: in Section 2, we describe the proximal interior point optimization method which is at the core of our approach, and we provide the proximity operator of the barrier for three useful cases in Section 3. In Section 4, we present the proposed neural network architecture and its associated backpropagation method. In Section 5, we conduct a stability analysis of the proposed network when the data fidelity term and the regularization function are quadratic. Section 6 is dedicated to numerical experiments and comparison to state-of-the-art methods for image deblurring; finally, some conclusions are drawn in Section 7.

2. Proximal interior point algorithm

2.1. Variational formulation and notation

As detailed in Section 1, the sought image $\overline{x}$ can be classically approximated by the minimizer of a penalized cost function expressed as the sum of a data-fitting term, which measures the fidelity of the solution to the observation model (1), and a regularization term, which is introduced so as to avoid meaningless solutions and improve stability to noise. This leads to problem (2) with $\lambda \in]0, +\infty[$ a regularization parameter. For every $q \in \mathbb{N}$, let $\Gamma_0(\mathbb{R}^q)$ denote the set of functions which take values in $\mathbb{R} \cup \{+\infty\}$ and are proper, convex, lower semicontinuous on $\mathbb{R}^q$. In the remaining of the paper, we will assume that, for every $y \in \mathbb{R}^m$, $f(\cdot, y) \in \Gamma_0(\mathbb{R}^m)$ is a twice-differentiable data-fidelity term, and $\mathcal{R} \in \Gamma_0(\mathbb{R}^n)$ is a twice-differentiable regularization function. Note that such assumption is necessary to define the derivative steps involved in the backpropagation procedure for the training of our network. The feasible set $\mathcal{C}$ is defined by p inequality constraints which enforce the fulfillment of some properties that are expected to be satisfied a priori by the image:

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid (\forall i \in \{1, \dots, p\})\ c_i(x) \geq 0\}, \tag{3}$$

where, for every $i \in \{1, \dots, p\}$, $-c_i \in \Gamma_0(\mathbb{R}^n)$. The strict interior of the feasible domain, $\mathrm{int}\mathcal{C}$, is equal to

$$\mathrm{int}\mathcal{C} = \{x \in \mathbb{R}^n \mid (\forall i \in \{1, \dots, p\})\ c_i(x) > 0\}, \tag{4}$$

and it is assumed to be nonempty. Inequality constraints are frequently used in image processing. For instance, they can be derived from the underlying geometry of the problem, like in [34], in the context of Poisson-noise denoising. We can also mention the work in [35], where inequality constraints are used in a problem of deformable image matching to ensure that the estimated image deformation is injective and preserves the topology. Constraints can also serve to enforce some a priori knowledge about the solution, as in the image segmentation approach in [36], where bound constraints are imposed on the segmented areas and their barycenters. Finally, we will assume that either $f(H\cdot, y) + \lambda\mathcal{R}$ is coercive, or $\mathcal{C}$ is bounded. Then the existence of solutions for (2) is guaranteed. It is worthy to emphasize that a large class of penalized formulations encountered in the literature of image restoration fulfills the above

requirements, see e.g. [37] and references therein.

Let us introduce additional notations, which will be useful in the rest of the paper. First, for every $g \in \Gamma_0(\mathbb{R}^n)$, $\gamma \in]0, +\infty[$, and $x \in \mathbb{R}^n$, the proximity operator [38] of γg at x is uniquely defined as

$$\text{prox}_{\gamma g}(x) = \underset{u \in \mathbb{R}^n}{\text{argmin}} \ \frac{1}{2}\|x - u\|^2 + \gamma g(u). \tag{5}$$

Finally, for all $(x, y, \lambda) \in \mathbb{R}^n \times \mathbb{R}^m \times]0, +\infty[$, we define

$$h(x, y, \lambda) = f(Hx, y) + \lambda \mathcal{R}(x), \tag{6}$$

and

$$\nabla_1 h(x, y, \lambda) = H^\top \nabla_1 f(Hx, y) + \lambda \nabla \mathcal{R}(x), \tag{7}$$

where $\nabla_1 f$ is the partial gradient of f with respect to its first variable.

2.2. Interior point approaches

In general, problem (2) does not have a closed-form solution on account of the inequality constraints, even for simple regularizations, hence an iterative solver must be used. Several resolution approaches are available, either based on projected gradient strategies [39, 40], ADMM [41], primal-dual schemes [42], or interior point techniques [43]. Standard interior point methods require to invert several $n \times n$ linear systems, which leads to a high computational complexity for large scale problems. Nonetheless, it has recently been shown that combining the interior point framework with a proximal forward–backward strategy [44, 45] leads to very competitive solvers for inverse problems [46, 47, 48].

The idea behind IPMs is to replace the initial constrained optimization problem by a sequence of unconstrained subproblems of the form:

$$\min_{x \in \mathbb{R}^n} \ f(Hx, y) + \lambda \mathcal{R}(x) + \mu \mathcal{B}(x) \tag{8}$$

where $\mathcal{B} : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is the logarithmic barrier function with unbounded derivative at the boundary of the feasible domain:

$$(\forall x \in \mathbb{R}^n) \quad \mathcal{B}(x) = \begin{cases} -\sum_{i=1}^{p} \ln(c_i(x)) & \text{if } x \in \text{int}\mathcal{C} \\ +\infty & \text{otherwise,} \end{cases} \tag{9}$$

and $\mu \in]0, +\infty[$ is the so–called barrier parameter which vanishes along the minimization process. We assumed that either $f(H\cdot, y) + \lambda\mathcal{R}$ is coercive, or $\mathcal{C}$ is bounded, hence, the set of solutions to (2) is bounded. Since int$\mathcal{C}$ is not empty we can apply [49, Theorem 5(ii)] and the existence of solutions to (8) is guaranteed.

2.3. Proposed iterative schemes

Thanks to the proximity operator, the IPM from [50] does not require any matrix inversion. When the proximity operator is computed in an exact manner, the proposed

IPM can be rewritten as Algorithm 1, whose convergence has been proven under some assumptions [50, Theorem 4.1].

Algorithm 1 Exact version of the proximal IPM in [50] applied to problem (2).

Let $x_0 \in \mathrm{int}\mathcal{C}$, $\underline{\gamma} > 0$ and $(\gamma_k)_{k\in\mathbb{N}}$ be a sequence such that $(\forall k \in \mathbb{N})\ \underline{\gamma} \leq \gamma_k$;
for $k = 0, 1, \ldots$ **do**
$x_{k+1} = \mathrm{prox}_{\gamma_k(h(\cdot,y,\lambda)+\mu_k\mathcal{B})}(x_k)$
end for

Algorithm 1 requires evaluating the proximity operator of the sum of the barrier and the regularized cost function, which can be an issue since, in most of the cases, this operator does not have a closed-form solution. This is the reason why we propose to modify it by introducing a forward step, which leads to Algorithm 2.

Algorithm 2 Proposed forward–backward proximal IPM.

Let $x_0 \in \mathrm{int}\mathcal{C}$, $\underline{\gamma} > 0$ and $(\gamma_k)_{k\in\mathbb{N}}$ be a sequence such that $(\forall k \in \mathbb{N})\ \underline{\gamma} \leq \gamma_k$;
for $k = 0, 1, \ldots$ **do**
$x_{k+1} = \mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}(x_k - \gamma_k\nabla_1 h(x_k, y, \lambda))$
end for

To the best of our knowledge, there is no available convergence study for Algorithm 2 among the literature of interior-point methods. There exist links between the above algorithm and the diagonal or penalization method introduced in [51]. Indeed, taking $A \equiv 0$ and $\Psi_1 \equiv 0$ in [51] leads to Algorithm 2, whose convergence is proven. However, there are some key differences between both approaches, namely *i)* in [51], the barrier parameter tends to infinity while it goes to zero in our case, and *ii)* the algorithm in [51] solves a hierarchical minimization problem instead of the constrained optimization problem (2). It is worth noting that Algorithm 2 only requires computing the proximity operator of the logarithmic barrier. We will provide its expression in Section 3 for three different types of constraints.

2.4. Limitations

In IPMs, the barrier parameter and stepsize sequences, $(\mu_k)_{k\in\mathbb{N}}$ and $(\gamma_k)_{k\in\mathbb{N}}$, are usually set by following some heuristic rules, which ensure the convergence of the method to a minimizer of the considered objective function. However, handcrafted variational formulations do not necessarily capture well image quality. These heuristics can thus lead to a loss in terms of efficiency and versatility of the resulting restoration schemes. Moreover, as already mentioned, an accurate setting of the regularization weights is particularly critical in order to obtain a satisfactory image quality when using such penalized restoration approaches. Existing approaches for selecting λ, which are based on statistical considerations, are usually associated with a substantial increase of the computational cost.

To overcome these limitations, we propose to unfold Algorithm 2 over a given number of iterations and to learn the stepsize, the barrier and the regularization parameters for every iteration in a supervised fashion. Our machine learning method will make use of gradient backpropagation for its training step. The latter requires the derivatives of the proximity operator in Algorithm 2 with respect to its input and to the aforementioned parameters which are to be learned. Therefore, we first conduct an analysis of the proximity operator of the barrier and of its derivatives, for three examples of interest in Section 3.

3. Proximity operator of the barrier

Let $\mathcal{B}$ be defined as in (9) and for all $\mu > 0$, $\gamma > 0$ and $x \in \mathbb{R}^n$, let φ be defined as follows:

$$\varphi(x, \mu, \gamma) = \operatorname{prox}_{\gamma\mu\mathcal{B}}(x). \tag{10}$$

We provide in this section expressions of φ and of its derivatives with respect to its input variable x and the involved barrier and stepsize parameters (μ, γ), for three common types of constraints. The latter will be necessary for training the proposed neural network using a gradient backpropagation scheme.

3.1. Affine constraints

Let us first consider the following half-space constraint:

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid a^\top x \leq b\}, \tag{11}$$

with $a \in \mathbb{R}^n \setminus \{0\}$ and $b \in \mathbb{R}$.

Proposition 1 *Let $\gamma > 0$, $\mu > 0$, and let $\mathcal{B}$ be the function associated to* (11)*, defined as*

$$(\forall u \in \mathbb{R}^n) \quad \mathcal{B}(u) = \begin{cases} -\ln(b - a^\top u) & \text{if } a^\top u < b, \\ +\infty & \text{otherwise.} \end{cases} \tag{12}$$

Then, for every $x \in \mathbb{R}^n$, the proximity operator of $\gamma\mu\mathcal{B}$ at x is given by

$$\varphi(x, \mu, \gamma) = x + \frac{b - a^\top x - \sqrt{(b - a^\top x)^2 + 4\gamma\mu\|a\|^2}}{2\|a\|^2} a. \tag{13}$$

In addition, the Jacobian matrix of φ with respect to x and the gradients of φ with respect to μ and γ are given by

$$J_\varphi^{(x)}(x, \mu, \gamma) = \mathbb{I}_n - \frac{1}{2\|a\|^2} \left(1 + \frac{a^\top x - b}{\sqrt{(b - a^\top x)^2 + 4\gamma\mu\|a\|^2}}\right) aa^\top, \tag{14}$$

$$\nabla_\varphi^{(\mu)}(x, \mu, \gamma) = \frac{-\gamma}{\sqrt{(b - a^\top x)^2 + 4\gamma\mu\|a\|^2}} a, \tag{15}$$

and

$$\nabla_\varphi^{(\gamma)}(x, \mu, \gamma) = \frac{-\mu}{\sqrt{(b - a^\top x)^2 + 4\gamma\mu\|a\|^2}} a, \tag{16}$$

where $\mathbb{I}_n \in \mathbb{R}^{n \times n}$ denotes the identity matrix.

Proof. The expression for the proximity operator (13) directly follows from [38, Example 24.40], [38, Proposition 24.8 (v)] and [38, Corollary 24.15]. Taking the derivative of (13) with respect to x, μ and γ leads to (14)–(16). □

3.2. Hyperslab constraints

We now consider the following hyperslab set:

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid b_{\mathrm{m}} \leq a^\top x \leq b_{\mathrm{M}}\}, \tag{17}$$

where $a \in \mathbb{R}^n \setminus \{0\}$, $b_{\mathrm{m}} \in \mathbb{R}$ and $b_{\mathrm{M}} \in \mathbb{R}$ with $b_{\mathrm{m}} < b_{\mathrm{M}}$.

Proposition 2 *Let $\gamma > 0$, $\mu > 0$, and let $\mathcal{B}$ be the barrier function associated to* (17)*, defined as*

$$(\forall u \in \mathbb{R}^n) \quad \mathcal{B}(u) = \begin{cases} -\ln(b_{\mathrm{M}} - a^\top u) - \ln(a^\top u - b_{\mathrm{m}}) & \text{if } b_{\mathrm{m}} < a^\top u < b_{\mathrm{M}}, \\ +\infty & \text{otherwise.} \end{cases} \tag{18}$$

Then, for every $x \in \mathbb{R}^n$, the proximity operator of $\gamma\mu\mathcal{B}$ at x is given by

$$\varphi(x, \mu, \gamma) = x + \frac{\kappa(x, \mu, \gamma) - a^\top x}{\|a\|^2} a, \tag{19}$$

where $\kappa(x, \mu, \gamma)$ is the unique solution in $]b_{\mathrm{m}}, b_{\mathrm{M}}[$, of the following cubic equation:

$$\begin{aligned} 0 = {} & z^3 - (b_{\mathrm{m}} + b_{\mathrm{M}} + a^\top x)z^2 + (b_{\mathrm{m}} b_{\mathrm{M}} + a^\top x(b_{\mathrm{m}} + b_{\mathrm{M}}) - 2\gamma\mu\|a\|^2)z \\ & - b_{\mathrm{m}} b_{\mathrm{M}} a^\top x + \gamma\mu(b_{\mathrm{m}} + b_{\mathrm{M}})\|a\|^2. \end{aligned} \tag{20}$$

In addition, the Jacobian matrix of φ with respect to x and the gradients of φ with respect to μ and γ are given by

$$J_\varphi^{(x)}(x, \mu, \gamma) = \mathbb{I}_n + \frac{1}{\|a\|^2} \left(\frac{(b_{\mathrm{M}} - \kappa(x, \mu, \gamma))(b_{\mathrm{m}} - \kappa(x, \mu, \gamma))}{\eta(x, \mu, \gamma)} - 1 \right) aa^\top, \tag{21}$$

$$\nabla_\varphi^{(\mu)}(x, \mu, \gamma) = \frac{-\gamma(b_{\mathrm{m}} + b_{\mathrm{M}} - 2\kappa(x, \mu, \gamma))}{\eta(x, \mu, \gamma)} a, \tag{22}$$

and

$$\nabla_\varphi^{(\gamma)}(x, \mu, \gamma) = \frac{-\mu(b_{\mathrm{m}} + b_{\mathrm{M}} - 2\kappa(x, \mu, \gamma))}{\eta(x, \mu, \gamma)} a, \tag{23}$$

where

$$\begin{aligned} \eta(x, \mu, \gamma) = {} & (b_{\mathrm{M}} - \kappa(x, \mu, \gamma))(b_{\mathrm{m}} - \kappa(x, \mu, \gamma)) \\ & - (b_{\mathrm{m}} + b_{\mathrm{M}} - 2\kappa(x, \mu, \gamma))(\kappa(x, \mu, \gamma) - a^\top x) - 2\gamma\mu\|a\|^2. \end{aligned} \tag{24}$$

Proof. Let $x \in \mathbb{R}^n$, $\gamma > 0$, and $\mu > 0$. The expression for the proximity operator (19) follows from [52, Example 4.15] and [38, Corollary 24.15]. Let F be defined as follows:

$$F(x, \mu, \gamma, z) = (b_{\mathrm{M}} - z)(b_{\mathrm{m}} - z)(z - a^\top x) + \gamma\mu(b_{\mathrm{M}} + b_{\mathrm{m}} - 2z)\|a\|^2, \tag{25}$$

for $z \in]b_{\mathrm{m}}, b_{\mathrm{M}}[$. Expanding (25) gives the following:

$$\begin{aligned} F(x,\mu,\gamma,z) = z^3 - (a^\top x + b_{\mathrm{m}} + b_{\mathrm{M}})z^2 + (b_{\mathrm{m}}b_{\mathrm{M}} + a^\top x(b_{\mathrm{m}} + b_{\mathrm{M}}) - 2\gamma\mu\|a\|^2)z \\ - b_{\mathrm{m}}b_{\mathrm{M}}a^\top x + \gamma\mu(b_{\mathrm{m}} + b_{\mathrm{M}})\|a\|^2. \end{aligned} \tag{26}$$

Hence, by definition of $\kappa(x,\mu,\gamma)$, we have $F(x,\mu,\gamma,\kappa(x,\mu,\gamma)) = 0$. In addition, the derivative of F with respect to its last variable is equal to

$$\nabla F^{(z)}(x,\mu,\gamma,z) = (b_{\mathrm{M}} - z)(b_{\mathrm{m}} - z) - (b_{\mathrm{m}} + b_{\mathrm{M}} - 2z)(z - a^\top x) - 2\gamma\mu\|a\|^2. \tag{27}$$

By construction, $(b_{\mathrm{M}} - \kappa(x,\mu,\gamma))(b_{\mathrm{m}} - \kappa(x,\mu,\gamma)) < 0$. Moreover, $-2\gamma\mu\|a\|^2 < 0$ and, since $F(x,\mu,\gamma,\kappa(x,\mu,\gamma)) = 0$, it follows that $(b_{\mathrm{m}}+b_{\mathrm{M}}-2\kappa(x,\mu,\gamma))$ and $\kappa(x,\mu,\gamma)-a^\top x$ share the same sign. Hence,

$$\eta(x,\mu,\gamma) = \nabla F^{(z)}(x,\mu,\gamma,\kappa(x,\mu,\gamma)) \neq 0. \tag{28}$$

From the implicit function theorem [53, Theorem 1B.1], we deduce that the gradient of κ with respect to x and the partial derivatives of κ with respect to μ and γ exist and are equal to

$$\nabla\kappa^{(x)}(x,\mu,\gamma) = \frac{(b_{\mathrm{M}} - \kappa(x,\mu,\gamma))(b_{\mathrm{m}} - \kappa(x,\mu,\gamma))}{\eta(x,\mu,\gamma)}a, \tag{29}$$

$$\nabla\kappa^{(\mu)}(x,\mu,\gamma) = \frac{-\gamma\|a\|^2(b_{\mathrm{m}} + b_{\mathrm{M}} - 2\kappa(x,\mu,\gamma))}{\eta(x,\mu,\gamma)}, \tag{30}$$

and

$$\nabla\kappa^{(\gamma)}(x,\mu,\gamma) = \frac{-\mu\|a\|^2(b_{\mathrm{m}} + b_{\mathrm{M}} - 2\kappa(x,\mu,\gamma))}{\eta(x,\mu,\gamma)}. \tag{31}$$

Differentiating (19) with respect to x, μ and γ and using (29)–(31) yields (21)–(23). □

Note that the three roots of (20) can easily be computed using the Cardano formula. The graph of the resulting proximity operator is plotted on Figure 1 (left) for $n = 1$, $a = 1$, $b_{\mathrm{m}} = 0$, $b_{\mathrm{M}} = 1$, and various values for $\gamma\mu$.

3.3. Bounded ℓ_2-norm

We now consider the case when the feasible set in (2) is an Euclidean ball

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid \|x - c\|^2 \leq \alpha\}, \tag{32}$$

with $\alpha > 0$ and $c \in \mathbb{R}^n$.

Proposition 3 *Let $\gamma > 0$ and let $\mu > 0$. Let $\mathcal{B}$ be the barrier function associated to* (32)*, defined as*

$$(\forall u \in \mathbb{R}^n) \quad \mathcal{B}(u) = \begin{cases} -\ln(\alpha - \|u - c\|^2) & \text{if } \|u - c\|^2 < \alpha, \\ +\infty & \text{otherwise.} \end{cases} \tag{33}$$

Then, for every $x \in \mathbb{R}^n$, the proximity operator of $\gamma\mu\mathcal{B}$ at x is given by

$$\varphi(x,\mu,\gamma) = c + \frac{\alpha - \kappa(x,\mu,\gamma)^2}{\alpha - \kappa(x,\mu,\gamma)^2 + 2\gamma\mu}(x - c), \tag{34}$$

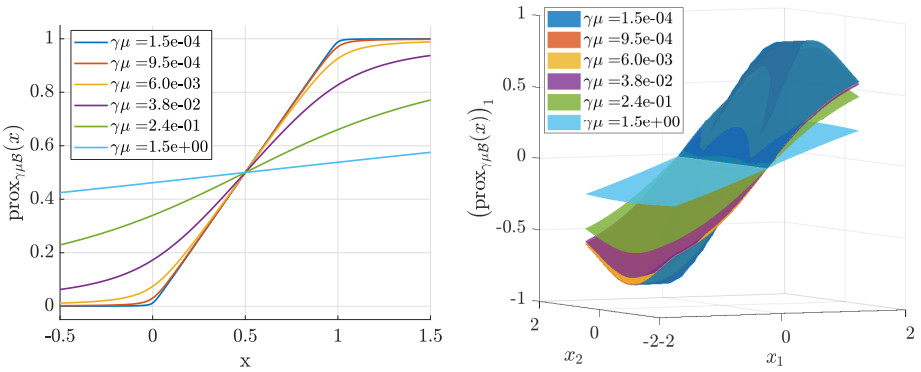

Figure 1. Proximity operator of the logarithmic barrier: $\mathrm{prox}_{\gamma\mu\mathcal{B}}(x)$ for hyperslab constraint as in Section 3.2 with $b_{\mathrm{m}} = 0$ and $b_{\mathrm{M}} = 1$ (*left*), $\left(\mathrm{prox}_{\gamma\mu\mathcal{B}}(x)\right)_1$ for a constraint on the ℓ_2-norm as in Section 3.3 with $\alpha = 0.7$ (*right*).

where $\kappa(x,\mu,\gamma)$ *is the unique solution in* $[0,\sqrt{\alpha}[$ *of the cubic equation:*

$$0 = z^3 - \|x-c\|z^2 - (\alpha + 2\gamma\mu)z + \alpha\|x-c\|. \tag{35}$$

In addition, the Jacobian matrix of φ *with respect to* x *and the gradients of* φ *with respect to* μ *and* γ *are given by*

$$J_{\varphi}^{(x)}(x,\mu,\gamma) = \frac{\alpha - \|\varphi(x,\mu,\gamma)-c\|^2}{\alpha - \|\varphi(x,\mu,\gamma)-c\|^2 + 2\gamma\mu} M(x,\mu,\gamma), \tag{36}$$

$$\nabla_{\varphi}^{(\mu)}(x,\mu,\gamma) = \frac{-2\gamma}{\alpha - \|\varphi(x,\mu,\gamma)-c\|^2 + 2\gamma\mu} M(x,\mu,\gamma)(\varphi(x,\mu,\gamma)-c), \tag{37}$$

and

$$\nabla_{\varphi}^{(\gamma)}(x,\mu,\gamma) = \frac{-2\mu}{\alpha - \|\varphi(x,\mu,\gamma)-c\|^2 + 2\gamma\mu} M(x,\mu,\gamma)(\varphi(x,\mu,\gamma)-c), \tag{38}$$

where

$$M(x,\mu,\gamma) = \mathbb{I}_n - \frac{2(x-\varphi(x,\mu,\gamma))(\varphi(x,\mu,\gamma)-c)^\top}{\alpha - 3\|\varphi(x,\mu,\gamma)-c\|^2 + 2\gamma\mu + 2(\varphi(x,\mu,\gamma)-c)^\top(x-c)}. \tag{39}$$

Proof. Let $x \in \mathbb{R}^n$, $\gamma > 0$, $\mu > 0$. Let us first consider the case when $c = 0$. We denote with φ_0 the following proximity operator:

$$\varphi_0(x,\mu,\gamma) = \underset{u \in \mathrm{int}\mathcal{C}}{\mathrm{argmin}} \ \frac{1}{2}\|x-u\|^2 - \gamma\mu \ln(\alpha - \|u\|^2). \tag{40}$$

Hence, $\|\varphi_0(x,\mu,\gamma)\|^2 < \alpha$ and $\varphi_0(x,\mu,\gamma)$ is a solution to the following equation:

$$0 = \varphi_0(x,\mu,\gamma) - x + \frac{2\gamma\mu}{\alpha - \|\varphi_0(x,\mu,\gamma)\|^2}\varphi_0(x,\mu,\gamma). \tag{41}$$

Since $\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu > 0$, (41) becomes

$$\varphi_0(x,\mu,\gamma) = \frac{\alpha - \|\varphi_0(x,\mu,\gamma)\|^2}{\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu} x. \tag{42}$$

By taking the norm in both sides of (42), we deduce that $\|\varphi_0(x,\mu,\gamma)\| = \kappa(x,\mu,\gamma)$ is a solution to the cubic equation (35). Since the proximity operator at a given x is uniquely defined, there exists only one real solution to (35) which belongs to $[0, \sqrt{\alpha}[$. Plugging the latter into (42) leads to (34). The analysis when $c \neq 0$ is deduced from the case $c = 0$ by using [38, Proposition 24.8 (v)]: the proximity operator of $\gamma\mu\mathcal{B}$ at x is given by

$$\varphi(x,\mu,\gamma) = c + \varphi_0(x - c, \mu, \gamma). \tag{43}$$

Let us study the derivatives of φ_0. For every $v \in \mathbb{R}^n$, let F be defined as

$$F(x,\mu,\gamma,v) = (\alpha - \|v\|^2)(v - x) + 2\gamma\mu v. \tag{44}$$

The Jacobian of F with respect to its last variable is equal to

$$J_F^{(v)}(x,\mu,\gamma,v) = (\alpha - \|v\|^2 + 2\gamma\mu)\mathbb{I}_n + 2(x - v)v^\top. \tag{45}$$

Since $\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 > 0$, according to the Sherman–Morrison Lemma [54], $J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))$ is invertible if and only if

$$\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu + 2\varphi_0(x,\mu,\gamma)^\top(x - \varphi_0(x,\mu,\gamma)) \neq 0. \tag{46}$$

Furthermore, it follows from (41) that

$$F(x,\mu,\gamma,\varphi_0(x,\mu,\gamma)) = 0. \tag{47}$$

Applying $\varphi_0(x,\mu,\gamma)^\top$ on (47) leads to $\varphi_0(x,\mu,\gamma)^\top(x - \varphi_0(x,\mu,\gamma)) \geq 0$. In addition, $\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu > 0$. Hence, $J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))$ is invertible and its inverse is given by the Sherman–Morrison formula:

$$\begin{aligned} J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))^{-1} =& \frac{1}{\alpha - \|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu} \times \\ & \left[\mathbb{I}_n - \frac{2(x - \varphi_0(x,\mu,\gamma))\varphi_0(x,\mu,\gamma)^\top}{\alpha - 3\|\varphi_0(x,\mu,\gamma)\|^2 + 2\gamma\mu + 2\varphi_0(x,\mu,\gamma)^\top x}\right]. \end{aligned} \tag{48}$$

From the implicit function theorem [53, Theorem 1B.1] we deduce that the Jacobian of φ_0 with respect to x and the gradients of φ_0 with respect to μ and γ exist and are equal to

$$J_{\varphi_0}^{(x)}(x,\mu,\gamma) = -J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))^{-1} J_F^{(x)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma)), \tag{49}$$

$$\nabla_{\varphi_0}^{(\mu)}(x,\mu,\gamma) = -J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))^{-1} \nabla_F^{(\mu)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma)), \tag{50}$$

and

$$\nabla_{\varphi_0}^{(\gamma)}(x,\mu,\gamma) = -J_F^{(v)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma))^{-1} \nabla_F^{(\gamma)}(x,\mu,\gamma,\varphi_0(x,\mu,\gamma)). \tag{51}$$

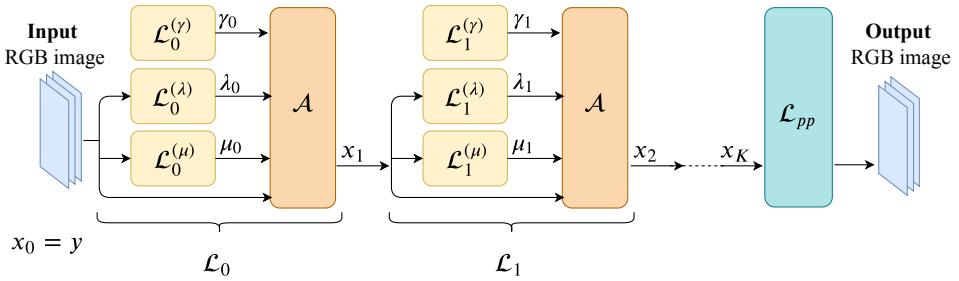

Figure 2. iRestNet global architecture.

When $c \neq 0$, the derivatives of φ are deduced from those of φ_0 using (43):

$$J_{\varphi}^{(x)}(x,\mu,\gamma) = -J_F^{(v)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c)^{-1} J_F^{(x)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c), \quad (52)$$

$$\nabla_{\varphi}^{(\mu)}(x,\mu,\gamma) = -J_F^{(v)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c)^{-1} \nabla_F^{(\mu)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c), \quad (53)$$

and

$$\nabla_{\varphi}^{(\gamma)}(x,\mu,\gamma) = -J_F^{(v)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c)^{-1} \nabla_F^{(\gamma)}(x-c,\mu,\gamma,\varphi(x,\mu,\gamma)-c), \quad (54)$$

which lead to (36)-(38). □

Similarly to the previous case, the three solutions to (35) can be obtained by using the Cardano formula. The form of the resulting proximity operator for $n = 2$ is plotted on Figure 1 (right) for $\alpha = 0.7$, $c = 0$, and several values of $\gamma\mu$ and x; for symmetry reasons, only the first component $\left(\text{prox}_{\gamma\mu\mathcal{B}}(x)\right)_1$ is represented.

As shown in this section, the proximity operator of the barrier is easily computable and differentiable for several classic types of constraints. Next, we detail the proposed approach in Section 4.

4. iRestNet architecture

4.1. Overview

Our proposal is to adopt a supervised learning strategy in order to determine, from a training set of images, an optimal setting for the parameters of Algorithm 2, which should lead to an optimal image restoration quality. To this aim, Algorithm 2 is unfolded over K iterations and the regularization parameter λ is untied across the network, so as to provide more flexibility to the approach [17]. The update rule at a given iteration $k \in \{0,\dots,K-1\}$ reads

$$x_{k+1} = \mathcal{A}\left(x_k, \mu_k, \gamma_k, \lambda_k\right) \quad (55)$$

with

$$\mathcal{A}\left(x_k, \mu_k, \gamma_k, \lambda_k\right) = \text{prox}_{\gamma_k \mu_k \mathcal{B}}\left(x_k - \gamma_k \nabla_1 h\left(x_k, y, \lambda_k\right)\right). \quad (56)$$

For every $k \in \{0,\dots,K-1\}$, we build the k-th *layer* $\mathcal{L}_k$ as the association of three hidden structures, $\mathcal{L}_k^{(\mu)}$, $\mathcal{L}_k^{(\gamma)}$ and $\mathcal{L}_k^{(\lambda)}$, followed by the update $\mathcal{A}$. Structures $\mathcal{L}_k^{(\mu)}$,

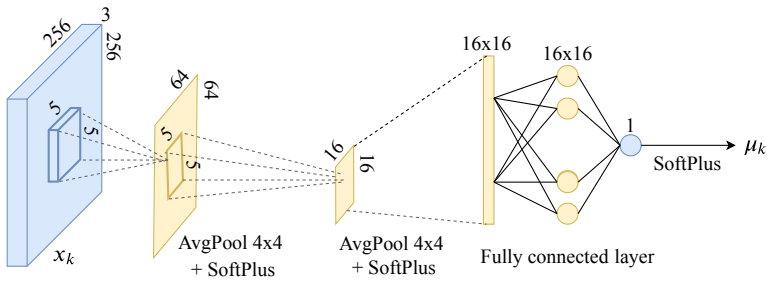

Figure 3. Architecture of $\mathcal{L}_k^{(\mu)}$.

$\mathcal{L}_k^{(\gamma)}$, and $\mathcal{L}_k^{(\lambda)}$ aim at inferring the barrier parameter μ_k, the stepsize γ_k and the regularization weight λ_k, respectively. Since a finite number K of layers (i.e., updates) is used, the convergence of the resulting scheme is not an issue. Note that we also allow in our framework the use of a post–processing step after going through the K layers, that will be denoted as $\mathcal{L}_{\text{pp}}$. The resulting architecture is depicted in Figure 2.

4.2. Hidden structures

Let us now provide more details about the hidden structures. For every $k \in \{0, \ldots, K-1\}$, the outputs $(\mu_k, \gamma_k, \lambda_k)$ of the structures $\mathcal{L}_k^{(\mu)}$, $\mathcal{L}_k^{(\gamma)}$, and $\mathcal{L}_k^{(\lambda)}$ must be positive. To enforce such constraint, we use the Softplus function [55], defined below, which can be viewed as a smooth approximation of the ReLU activation function:

$$(\forall z \in \mathbb{R}) \quad \text{Softplus}(z) = \ln(1 + \exp(z)). \tag{57}$$

Unlike the ReLU, the gradient of Softplus is never strictly equal to zero, which, given our architecture, helps to propagate the gradient through the network. The stepsize is estimated as follows,

$$\gamma_k = \mathcal{L}_k^{(\gamma)} = \text{Softplus}\left(a_k\right), \tag{58}$$

where a_k is a scalar parameter of the network learned during training. The barrier parameter is obtained using two convolutional and average pooling layers followed by a fully connected layer. The detailed architecture of $\mathcal{L}_k^{(\mu)}$ is depicted in Figure 3.
Traditional methods for estimating the regularization parameter generally depend on the signal-to-noise ratio and on the image statistics [56]. For most applications the noise level is unknown and can be estimated, for instance, by applying a median filter over the wavelet diagonal coefficients of the image [57]. This strategy is used in the numerical experiments presented in Section 6. The advantage is to yield a network which can handle datasets for which the signal-to-noise ratio is unknown and can vary within a reasonable range. The expression of $\mathcal{L}_k^{(\lambda)}$ is then problem–dependent since it is built upon the regularization strategy. A specific example is given in Section 6 for the total variation regularization function.
Regarding the post-processing step $\mathcal{L}_{\text{pp}}$, its detailed architecture also depends on the task to be performed. An example is provided in Section 6 for the case of deblurring: the purpose of $\mathcal{L}_{\text{pp}}$ is then to remove remaining artifacts using convolutional layers, residual learning, batch normalization, and dilation.

4.3. Differential calculus

To train the neural network presented in Figure 2 using gradient descent, one needs to compute the gradient of x_K with respect to the different parameters of the network. The chain rule can be applied since most of the steps in the network correspond to operators having straightforward derivatives. However, particular care should be taken when differentiating $\mathcal{A}$. Since f and $\mathcal{R}$ are assumed to be twice differentiable, the only area of concern is related to $\mathrm{prox}_{\gamma\mu\mathcal{B}}$. If $\mathrm{prox}_{\gamma\mu\mathcal{B}}$ is simple enough, automatic differentiation [58] can be used. Otherwise, as shown in Section 3, for common examples of barrier functions, the differential of this term is well-defined. The corresponding expressions for the derivatives are provided in Propositions 1–3.

5. Network stability

One critical issue concerning neural networks is to guarantee that their performance remains acceptable when the input is perturbed. For example, the authors of [15] show that the class prediction made by AlexNet can be arbitrarily changed by using small nonrandom perturbations on the test image. A recent work [59] provides a theoretical framework which enables to evaluate the robustness of a network. In this section, we will focus on a subclass of problem (2) where both $f(\cdot, y)$ and $\mathcal{R}$ are quadratic functions. After highlighting the similarities between the proposed architecture and generic feedforward networks in that case, we will give explicit conditions under which the robustness of the proposed architecture is ensured.

5.1. Relation to generic deep neural networks

Although the proposed architecture may seem specific to Algorithm 2, it is actually very similar to generic feedforward neural networks. Classical feedforward (acyclic) architectures [60] can be expressed as $R_{K-1}\circ(W_{K-1}\cdot+b_{K-1})\circ\cdots\circ R_0\circ(W_0\cdot+b_0)$, where $(R_k)_{0\leq k\leq K-1}$ are nonlinear activation functions, $(W_k)_{0\leq k\leq K-1}$ are weight operators and $(b_k)_{0\leq k\leq K-1}$ are bias parameters. Let us show that iRestNet actually shares a similar structure. For the sake of simplicity, we will consider the variational problem,

$$\underset{x\in\mathcal{C}}{\text{minimize}}\ \frac{1}{2}\|Hx-y\|^2+\frac{\lambda}{2}\|Dx\|^2, \tag{59}$$

where $y\in\mathbb{R}^n$, $H\in\mathbb{R}^{n\times n}$, $D\in\mathbb{R}^{n\times n}$, and $\mathcal{C}$ is defined as in (3). Moreover, we assume that no post–processing layer $\mathcal{L}_{\mathrm{pp}}$ is used. Following the notation of Section 4, $(\forall k\in\{0,\ldots,K-1\})$ $(\mu_k,\gamma_k,\lambda_k)$ are given positive real numbers, K being the number of layers of the network. Then, for every $k\in\{0,\ldots,K-1\}$, layer $\mathcal{L}_k$ corresponds to the following update,

$$\begin{aligned} x_{k+1} &= \mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}\left(x_k-\gamma_k\left(H^\top(Hx_k-y)+\lambda_k D^\top Dx_x\right)\right)\\ &= \mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}\left(\left[\mathbb{I}_n-\gamma_k\left(H^\top H+\lambda_k D^\top D\right)\right]x_k+\gamma_k H^\top y\right), \end{aligned} \tag{60}$$

where $\mathcal{B}$ is defined as in (9). For every $k\in\{0,\ldots,K-1\}$, we set

$$W_k=\mathbb{I}_n-\gamma_k\left(H^\top H+\lambda_k D^\top D\right),\quad b_k=\gamma_k H^\top y,\quad\text{and}\quad R_k=\mathrm{prox}_{\gamma_k\mu_k\mathcal{B}}. \tag{61}$$

Then, the K-layer network $\mathcal{L}_{K-1}\circ\cdots\circ\mathcal{L}_0$ is equivalent to $R_{K-1}\circ(W_{K-1}\cdot+b_{K-1})\circ\cdots\circ R_0\circ(W_0\cdot+b_0)$, where $(W_k)_{0\leq k\leq K-1}$ and $(b_k)_{0\leq k\leq K-1}$ are interpreted as weight

operators and bias parameters, respectively. The operators $(R_k)_{0\leq k\leq K-1}$ defined in (61) can be viewed as specific activation functions since, as shown in [59], every standard activation function can be derived from a proximity operator. In addition, using [38, Proposition 24.8(iii)], for every $k \in \{0,\dots,K-1\}$, R_k can be re-written as the sum of a *proximal activation operator* [59, Definition 2.20] and a bias.

5.2. Preliminary results

Before stating our main stability theorem, we recall the result from [59, Lemma 3.3] in Proposition 4 below. We then derive Proposition 5, which will appear useful when addressing the robustness of the global network. In the following, $\mathcal{S}_n$ denotes the set of symmetric matrices in $\mathbb{R}^{n\times n}$ and, for every $W \in \mathbb{R}^{n\times n}$, $\|W\|$ denotes its spectral norm.

Proposition 4 [59] *Let $K \geq 1$ be an integer and set $\theta_{-1} = 1$. For every $k \in \{0,\dots,K-1\}$, let $W_k \in \mathbb{R}^{n\times n}$ and let θ_k be defined by*

$$\theta_k = \|W_k \circ \cdots \circ W_0\| + \sum_{\ell=0}^{k-1} \sum_{0\leq j_0<\cdots<j_\ell\leq k-1} \|W_k \circ \cdots \circ W_{j_\ell+1}\| \times \|W_{j_\ell} \circ \cdots \circ W_{j_{\ell-1}+1}\| \cdots \|W_{j_0} \circ \cdots \circ W_0\|. \tag{62}$$

Then, for every $k \in \{0,\dots,K-1\}$, $\theta_k = \sum_{\ell=0}^{k} \theta_{\ell-1} \|W_k \circ \ldots \circ W_l\|$.

Proposition 5 *Let $K \geq 1$, $\theta > 0$, and $\alpha \in [1/2, 1]$. Let $W \in \mathcal{S}_n$ and let β_- and β_+ denote the smallest and largest eigenvalues of W, respectively. Then, the condition*

$$\|W - 2^K(1-\alpha)\mathbb{I}_n\| - \|W\| + 2\theta \leq 2^K\alpha \tag{63}$$

is satisfied if and only if one of the following conditions holds:

(i) $\beta_+ + \beta_- \leq 0$ *and* $\theta \leq 2^{K-1}(2\alpha - 1)$*;*

(ii) $0 \leq \beta_+ + \beta_- \leq 2^{K+1}(1-\alpha)$ *and* $2\theta \leq \beta_+ + \beta_- + 2^K(2\alpha-1)$*;*

(iii) $2^{K+1}(1-\alpha) \leq \beta_+ + \beta_-$ *and* $\theta \leq 2^{K-1}$*.*

Proof. Let $\alpha \in [1/2, 1]$. Since $W \in \mathcal{S}_n$, we have, $\|W\| = \max\{\beta_+, -\beta_-\}$, and

$$\|W - 2^K(1-\alpha)\mathbb{I}_n\| = \max\left\{\beta_+ - 2^K(1-\alpha), -\beta_- + 2^K(1-\alpha)\right\}. \tag{64}$$

Three different cases arise that we review below.

(i) If $\beta_+ + \beta_- \leq 0$ then $\|W\| = -\beta_-$ and

$$\beta_+ - 2^K(1-\alpha) \leq -\beta_- + 2^K(1-\alpha). \tag{65}$$

From (64) and (65), we deduce that $\|W - 2^K(1-\alpha)\mathbb{I}_n\| = -\beta_- + 2^K(1-\alpha)$. Replacing $\|W\|$ and $\|W - 2^K(1-\alpha)\mathbb{I}_n\|$ by their value in (63) leads to Proposition 5(i).

(ii) If $0 \leq \beta_+ + \beta_- \leq 2^{K+1}(1-\alpha)$ then $\|W\| = \beta_+$ and (65) is satisfied. Hence, $\|W - 2^K(1-\alpha)\mathbb{I}_n\| = -\beta_- + 2^K(1-\alpha)$. Replacing $\|W\|$ and $\|W - 2^K(1-\alpha)\mathbb{I}_n\|$ by their value in (63) leads to Proposition 5(ii).

(iii) If $2^{K+1}(1-\alpha) \leq \beta_+ + \beta_-$ then $\|W\| = \beta_+$ and

$$\beta_+ - 2^K(1-\alpha) \geq -\beta_- + 2^K(1-\alpha). \tag{66}$$

From (64) and (66), we deduce that $\|W - 2^K(1-\alpha)\mathbb{I}_n\| = \beta_+ - 2^K(1-\alpha)$. Replacing $\|W\|$ and $\|W - 2^K(1-\alpha)\mathbb{I}_n\|$ by their value in (63) leads to Proposition 5(iii), which completes the proof.

□

5.3. Averaged operator

The notion of nonexpansiveness, whose definition is recalled below, plays a central role in the analysis of the robustness of nonlinear operators. We racall that $T : \mathbb{R}^n \to \mathbb{R}^n$ is nonexpansive if it is Lipschitz continuous with constant 1, i.e.,

$$(\forall x \in \mathbb{R}^n)(\forall y \in \mathbb{R}^n) \quad \|T(x) - T(y)\| \leq \|x - y\|. \tag{67}$$

In the present study we make use of the notion of averaged operator [38], which is stronger than nonexpansiveness. T is α–averaged with $\alpha \in [0, 1]$, if there exists a nonexpansive operator $R : \mathbb{R}^n \to \mathbb{R}^n$ such that $T = (1-\alpha)I_n + \alpha R$, where I_n denotes the identity operator of $\mathbb{R}^n$.
The following property provides an upper bound of the effect of an input perturbation, which depends on the averageness constant α. In particular, the smaller α is, the more stable the operator is.

Proposition 6 [38, Remark 4.34, Proposition 4.35] *Let $T : \mathbb{R}^n \to \mathbb{R}^n$.*

(i) If T is averaged, then it is nonexpansive.

(ii) Let $\alpha \in]0, 1]$. T is α–averaged if and only if for every $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$,

$$\|T(x) - T(y)\|^2 \leq \|x - y\|^2 - \frac{1-\alpha}{\alpha}\|(I_n - T)(x) - (I_n - T)(y)\|^2. \tag{68}$$

5.4. Robustness of iRestNet to an input perturbation

Let us consider problem (59), where we assume additionally that $H^\top H$ and $D^\top D$ are diagonalizable in a same basis denoted $\mathcal{P}$. The latter is satisfied for instance if H and D are the results of cyclic convolutive operators. Theorem 1 below gives sufficient conditions under which the proposed network applied to problem (59) is averaged.

Theorem 1 *Let $\alpha \in [1/2, 1]$, $(W_k, b_k, R_k)_{0 \leq k \leq K-1}$ be defined by (61), and $(\theta_k)_{-1 \leq k \leq K-1}$ be defined as in Proposition 4. Let β_- and β_+ be the smallest and largest eigenvalues of $W = W_{K-1} \circ \cdots \circ W_0$, respectively. For every $p \in \{1, \ldots, n\}$ and every $k \in \{0, \ldots, K-1\}$, let $\beta_k^{(p)} = 1 - \gamma_k \left(\beta_H^{(p)} + \lambda_k \beta_D^{(p)}\right)$, where $\beta_H^{(p)}$ and $\beta_D^{(p)}$ denote the p^{th} eigenvalue of $H^\top H$ and $D^\top D$ in $\mathcal{P}$, respectively. Then, β_-, β_+, and $(\forall k \in \{0, \ldots, K-1\})$ θ_k can be computed as follows:*

$$\beta_- = \min_{1 \leq p \leq n} \prod_{k=0}^{K-1} \beta_k^{(p)}, \ \beta_+ = \max_{1 \leq p \leq n} \prod_{k=0}^{K-1} \beta_k^{(p)} \ \text{ and } \ \theta_k = \sum_{l=0}^{k} \theta_{l-1} \max_{1 \leq q_l \leq n} \left|\beta_k^{(q_l)} \ldots \beta_l^{(q_l)}\right|. \tag{69}$$

In addition, if one of the following conditions is satisfied

(i) $\beta_+ + \beta_- \leq 0$ *and* $\theta_{K-1} \leq 2^{K-1}(2\alpha - 1)$*;*
(ii) $0 \leq \beta_+ + \beta_- \leq 2^{K+1}(1-\alpha)$ *and* $2\theta_{K-1} \leq \beta_+ + \beta_- + 2^K(2\alpha - 1)$*;*
(iii) $2^{K+1}(1-\alpha) \leq \beta_+ + \beta_-$ *and* $\theta_{K-1} \leq 2^{K-1}$*,*

then the operator $R_{K-1} \circ (W_{K-1} \cdot + b_{K-1}) \circ \cdots \circ R_0 \circ (W_0 \cdot + b_0)$ *is* α*–averaged.*

Proof. If $H^\top H$ and $D^\top D$ are diagonalizable in the same basis then $W \in \mathcal{S}_n$, which, combined with Proposition 4, leads to (69). If one of the conditions (i)–(iii) is satisfied, then we deduce from Proposition 5 that W satisfies [59, Proposition 3.6(iii)]. and [59, Condition 3.1]. In addition, for every $k \in \{0, \ldots, K-1\}$, $R_k(\cdot + b_k)$ is firmly nonexpansive [38, Proposition 12.28]. Finally, [59, Theorem 3.8] completes the proof. □

The conditions provided by Theorem 1 can be easily checked using (69). Theorem 1 provides a framework under which iRestNet is robust to a perturbation of its input: the upper bound of the output perturbation can then be derived from Proposition 6.

6. Numerical experiments

In this section, we present numerical experiments on a set of problems of image restoration, demonstrating that in many cases the proposed approach yields a better reconstruction quality than standard variational and machine learning methods.

6.1. Problem formulation

We consider the non-blind color image deblurring problem, whose degradation model reads

$$y = H\overline{x} + \omega, \tag{70}$$

where n is the number of pixels, $y = (y^{(j)})_{1 \leq j \leq 3} \in \mathbb{R}^{3n}$ is the blurred RGB image, $\overline{x} = (\overline{x}^{(j)})_{1 \leq j \leq 3} \in \mathbb{R}^{3n}$ is the ground-truth, $H \in R^{3n \times 3n}$ is a linear operator that models the circular convolution of a known blur kernel with each channel of the color image, and $\omega \in \mathbb{R}^{3n}$ is a realization of an additive white Gaussian noise with standard deviation σ. An estimate of $\overline{x}$ can be derived from the following penalized formulation, which includes a smoothed total variation regularization,

$$\underset{x \in \mathcal{C}}{\text{minimize}} \quad \frac{1}{2}\|Hx - y\|^2 + \lambda \sum_{i=1}^{3n} \sqrt{\frac{(D_{\text{v}}x)_i^2 + (D_{\text{h}}x)_i^2}{\delta^2} + 1}, \tag{71}$$

where the feasible set C is the hypercube $[x_{\min}, x_{\max}]^{3n}$, $x_{\min}$ and $x_{\max}$ are a lower and an upper bound on the pixel intensity, respectively, $D_{\text{v}} \in \mathbb{R}^{3n \times 3n}$ and $D_{\text{h}} \in \mathbb{R}^{3n \times 3n}$ are the vertical and horizontal gradient operators, respectively, $\delta > 0$ is a smoothing parameter and $\lambda > 0$ is the regularization parameter. Here, $x_{\min} = 0$, $x_{\max} = 1$ and we set $\delta = 0.01$ in all experiments, which appears as an appropriate order of magnitude. To find this value for δ, we solved Problem 2 for a small set of images of the database and used the simplex method to find the best values for δ and λ in terms of image quality. It is worth noting that the value for δ has not been fine-tuned, but that the proposed architecture could also be easily modified to include the inference of δ. The update $\mathcal{A}$, defined in (56), is derived from (71), and is unfolded over K

iterations, as it is described in Section 4. The bound constraints in problem (71) fall under the framework studied in Section 3.2, which provides us with the expression for the proximity operator of the barrier and its gradient.

6.2. Network characteristics

The tuning of the number of unfolded iterations K must achieve a compromise between training time, memory requirement, and performance. In order to determine a suitable setting for K, we trained individually several layers of iRestNet and increased the number of layers until the performance of the network did not improve significantly. Using this procedure, the depth of iRestNet is taken equal to $K = 40$. Regarding the hidden structures $(\mathcal{L}_k^{(\lambda)})_{0\leq k\leq K-1}$, which estimate the regularization parameter, they are chosen in view of the regularization function used in problem (71) and have the following expression,

$$(\forall k \in \{0, \ldots, K-1\}) \quad \lambda_k = \mathcal{L}_k^{(\lambda)}(x_k) = \frac{\text{Softplus}(b_k)\,\widehat{\sigma}(y)}{\eta(x_k) + \text{Softplus}(c_k)}, \tag{72}$$

where (b_k, c_k) is a pair of scalars learned by the network, $\eta(x_k)$ is the standard deviation of the concatenated spatial gradients of x_k, $[(D_{\text{v}}x_k)^\top (D_{\text{h}}x_k)^\top]$, and $\widehat{\sigma}(y)$ is an approximation of the noise level in the blurred image. The noise level is estimated as in [61, Section 11.3.1]

$$\widehat{\sigma}(y) = \text{median}(|W_{\text{H}}y|)/0.6745, \tag{73}$$

where $|W_{\text{H}}y|$ is the vector gathering the absolute value of the diagonal coefficients of the first level Haar wavelet decomposition of y. It is worth noticing that the proposed architecture does not require any prior knowledge about the noise level, in particular the noise standard deviation does not have to be the same for all input images.
The architecture of the post-processing layer $\mathcal{L}_{\text{pp}}$ is inspired from [62]: it is made of 9 convolutional layers with filters of size 3×3. The dilation factor changes from one layer to another, so as to widen the receptive field without creating memory issues. There is little correlation between the artifacts that remain in the image after going through the 40 blocks of iRestNet and the ground-truth image. Hence, it is easier for the network to learn the residual mapping instead of the image itself since pushing the residual to zero is easier than fitting an identity mapping by a stack of layers [8, 62, 63]. Therefore, we add a skip connection between the input of $\mathcal{L}_{\text{pp}}$ and its output. Finally, a ReLU activation function is used after each convolution, the final activation function is chosen as the Sigmoid function, and residual learning is combined with batch normalization, a technique which is widely used in deep learning to fasten and stabilize the training process [62]. The final architecture of $\mathcal{L}_{\text{pp}}$ can be found in Figure 4.

6.3. Dataset and experimental settings

The training set is made of 1200 RGB images: 200 images stem from the Berkeley segmentation (BSD500) training set, while the remaining 1000 images are taken from the COCO training set. We use the BSD500 validation set, which is made of 100 images, to monitor the training and check if there is overfitting. The performance of the proposed method is evaluated on two different test sets: the BSD500 test set,

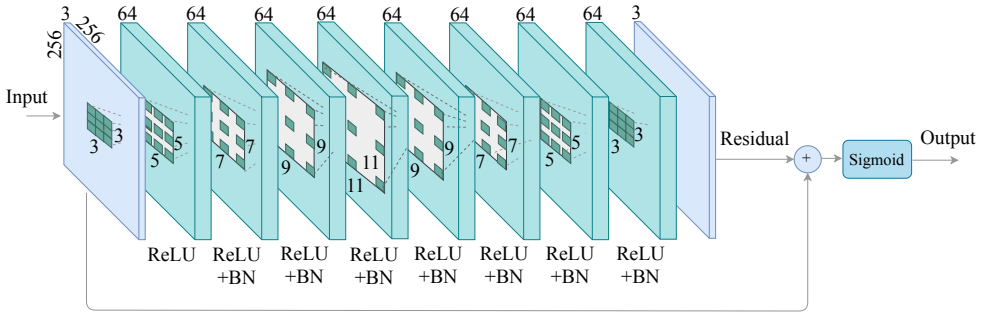

Figure 4. Architecture of $\mathcal{L}_{\text{pp}}$. BN: batch normalization.

which is made of 200 RGB images, and the Flickr30 test set used in [9], which is made of 30 RGB images. The test images have been center-cropped using a window of size 256×256. Blurry images are produced using the following 25×25 blur kernels and noise levels:

- A Gaussian kernel, which models atmospheric turbulence, with a standard deviation of 1.6 pixels, and a Gaussian noise standard deviation of $\sigma = 0.008$. This configuration is denoted as GaussianA. To evaluate the robustness of the proposed method with respect to the noise level, the same kernel is used with a Gaussian noise whose standard deviation is uniformly distributed between 0.01 and 0.05. The latter is denoted as GaussianB.
- The Gaussian kernel with a standard deviation of 3 pixels, and a Gaussian noise standard deviation of $\sigma = 0.04$, denoted as GaussianC.
- The eighth and third motion test kernels from [64], which are a real-world camera shake kernels, with a Gaussian noise standard deviation of $\sigma = 0.01$. These settings are denoted as MotionA and MotionB, respectively.
- The square uniform kernel of size 7×7, with a Gaussian noise standard deviation of $\sigma = 0.01$. This configuration is referred to as Square.

6.4. Training

For each degradation model, one iRestNet network is trained. We use a greedy approach for training the first 30 layers. For $\mathcal{L}_0$, a minibatch of 10 images is selected at every iteration, randomly cropped using a window of size 256×256, blurred with the given kernel, and degraded with Gaussian noise; the training of $\mathcal{L}_0$ stops after a fixed number of epochs. Then, for each image of the training set, a random crop of size 256×256 is selected, blurred, corrupted with noise and passed through $\mathcal{L}_0$, the output is saved and used as an input to train $\mathcal{L}_1$. When the training of $\mathcal{L}_1$ is complete, its output is used to train the next layer, etc... This training strategy is chosen with regards to its low memory requirement: the number of layers is not limited by the hardware. The rest of the network, $\mathcal{L}_{\text{pp}} \circ \mathcal{L}_{39} \circ \ldots \circ \mathcal{L}_{30}$, is trained as one block and the learning rate is multiplied by 0.9 every 50 epochs. To accelerate the training, for every $k \in \{1, \ldots, K-1\}$, the weights of $\mathcal{L}_k$ are initialized with those of $\mathcal{L}_{k-1}$. Detailed information about learning rates and number of epochs can be found in Table 1 below.

	GaussianA	GaussianB	GaussianC	MotionA	MotionB	Square
Rates	(0.01,0.001)	(0.01,0.001)	(0.001,0.001)	(0.01,0.002)	(0.01,0.001)	(0.01,0.005)
Epochs	(40,393)	(40,340)	(40,300)	(40,1200)	(40,1250)	(40,740)

Table 1. Training information. First row: initial learning rates, second row: number of epochs. For every couple, the first and second numbers correspond to the training of $(\mathcal{L}_k)_{0\leq k\leq 29}$ and $\mathcal{L}_{\text{pp}} \circ \mathcal{L}_{39} \circ \ldots \circ \mathcal{L}_{30}$, respectively.

The validation set is used to monitor this last step of the training. In particular, the parameter configuration that gives the best performance on the validation set during the training is the one saved and used for the tests. Note that for the first 30 layers, after each layer the quality of the restored training images should improve. This property comes from the training strategy, it is not encoded in the network: if memory was not an issue, then iRestNet could be trained in an end-to-end fashion.
We use the Adam optimizer [65] to minimize the training loss, which is taken as the negative of the structural similarity measure (SSIM) [66] defined below

$$\text{SSIM}(x,\overline{x}) = \frac{(2\mu_x\mu_{\overline{x}} + c_1)(2\sigma_x\sigma_{\overline{x}} + c_2)(2\text{cov}_{x\overline{x}} + c_3)}{(\mu_x^2 + \mu_{\overline{x}}^2 + c_1)(\sigma_x^2 + \sigma_{\overline{x}}^2 + c_2)(\sigma_x\sigma_{\overline{x}} + c_3)}, \tag{74}$$

where $\overline{x}$ is the ground truth, x is the restored image, (μ_x, σ_x) and $(\mu_{\overline{x}}, \sigma_{\overline{x}})$ are mean and standard deviation of x and $\overline{x}$, respectively, $\text{cov}_{x\overline{x}}$ is the cross–covariance of x and $\overline{x}$, and c_1, c_2 and c_3 are constants. As the SSIM captures features that are related to the human eye perception, it is more discriminative with regards to artifacts than the mean square error for instance. The gradient of the SSIM loss with respect to the trainable parameters of the network is computed using the code provided by the authors of [66], the chain rule, automatic differentiation [58], and the expression given in Section 3.2 for the derivatives of the barrier proximity operator.
Codes are implemented in Pytorch. Some hidden layers make use of functions that are not differentiable everywhere, like ReLU or the absolute value for instance. Since this nondifferentiability happens only at specific points for which the left and right derivatives are well–defined, Pytorch can handle it as explained in [67]. All trainings are conducted using a GeForce GTX 1080 GPU or a Tesla V100 GPU. The training, which can be performed off-line, takes approximately 3 to 4 days for each blur kernel, while the execution time is only about 1.4 sec per image on a GeForce GTX 1080 GPU.

6.5. Evaluation metrics and competitors

The restoration is evaluated in terms of the SSIM metric. The reconstruction given by the proposed approach is compared with a solution to problem (71) obtained using the projected gradient algorithm [39]. For every blurred image, the pair (λ, δ) which leads to the best SSIM is selected using the simplex method. The solution given by this variational approach is referred to as VAR. The latter is an unrealistic scenario since it assumes that there is a perfect estimator of the error, but it gives an upper bound on the image quality that one can expect by solving (71). We also use the following deep learning image restoration methods for comparison: EPLL [68], MLP [10], and IRCNN [62]. Finally, we include comparisons with two unfolded-based methods, namely FCNN [69], where the authors unfold a half-quadratic splitting algorithm and use a network to

learn an effective regularization function, and the method from [70], which is referred to as PDHG, where the authors perform a maximum of 30 iterations of a primal dual hybrid gradient algorithm and the proximity operator of the second regularization function is replaced by a neural network.
For FCNN, we use the code that is available online, in which the authors provide a model that has only been trained for motion blurs. For completeness, we also test this model on the Gaussian and uniform kernels of our test configurations. Similarly, for MLP and PDHG, the authors do not provide models that were trained specifically for MotionB and Square, so, in order to test these methods on MotionB we use the same models as for MotionA, and for Square we use models that were trained for a larger uniform blur.
Since MLP, EPLL and IRCNN require the knowledge of the noise level, for the GaussianB degradation model, we make use of the estimation of the noise standard deviation given by the method in [57]. In addition, since some comparison methods, like EPLL for instance, do not estimate well the borders of the images, the SSIM index is computed excluding a 6-pixel-wide frame for all images and all tested methods.

6.6. Results and discussion

The average SSIM obtained with the different methods for the various blur kernels and noise levels on the BSD500 test set can be found in Table 2. The mean SSIM achieved with iRestNet on this test set is greater than those obtained with the other methods for all degradation models except MotionA. For this kernel, the average SSIM achieved with iRestNet is the second highest value after IRCNN, which appears as the most competitive method. IRCNN involves two steps: first, a Wiener filter is applied to the blurred image, then, a neural network is used to predict the residual and denoise the image. These two steps are repeated 30 times, for 30 different manually tuned regularization parameters. In contrast, iRestNet does not require any tuning from the user regarding the regularization parameters during training. For completeness, the SSIM of all images of the BSD500 test set are plotted in Figure 5 for the 6 different degradation models. As one can see, iRestNet performs well in terms of SSIM on most of the images.

	GaussianA	GaussianB	GaussianC	MotionA	MotionB	Square
Blurred	0.676	0.526	0.326	0.383	0.549	0.544
VAR	0.804	0.723	0.587	0.819	0.829	0.756
EPLL [68]	0.800	0.708	0.565	0.816	0.839	0.755
MLP [10]	0.821	0.734	0.608	0.854	0.832	0.701
PDHG [70]	0.796	0.716	0.563	0.801	0.805	0.656
IRCNN [62]	0.841	0.768	0.619	**0.902**	0.907	0.834
FCNN [69]	0.782	0.711	0.569	0.794	0.847	0.752
iRestNet	**0.853**	**0.787**	**0.641**	0.898	**0.910**	**0.840**

Table 2. SSIM results on the BSD500 test set.

Since no image was taken from Flickr for training iRestNet, the results on the Flickr30 test set show how well the performance of the trained networks are transferable on test sets with statistics that are different from those of the training set. Table 3

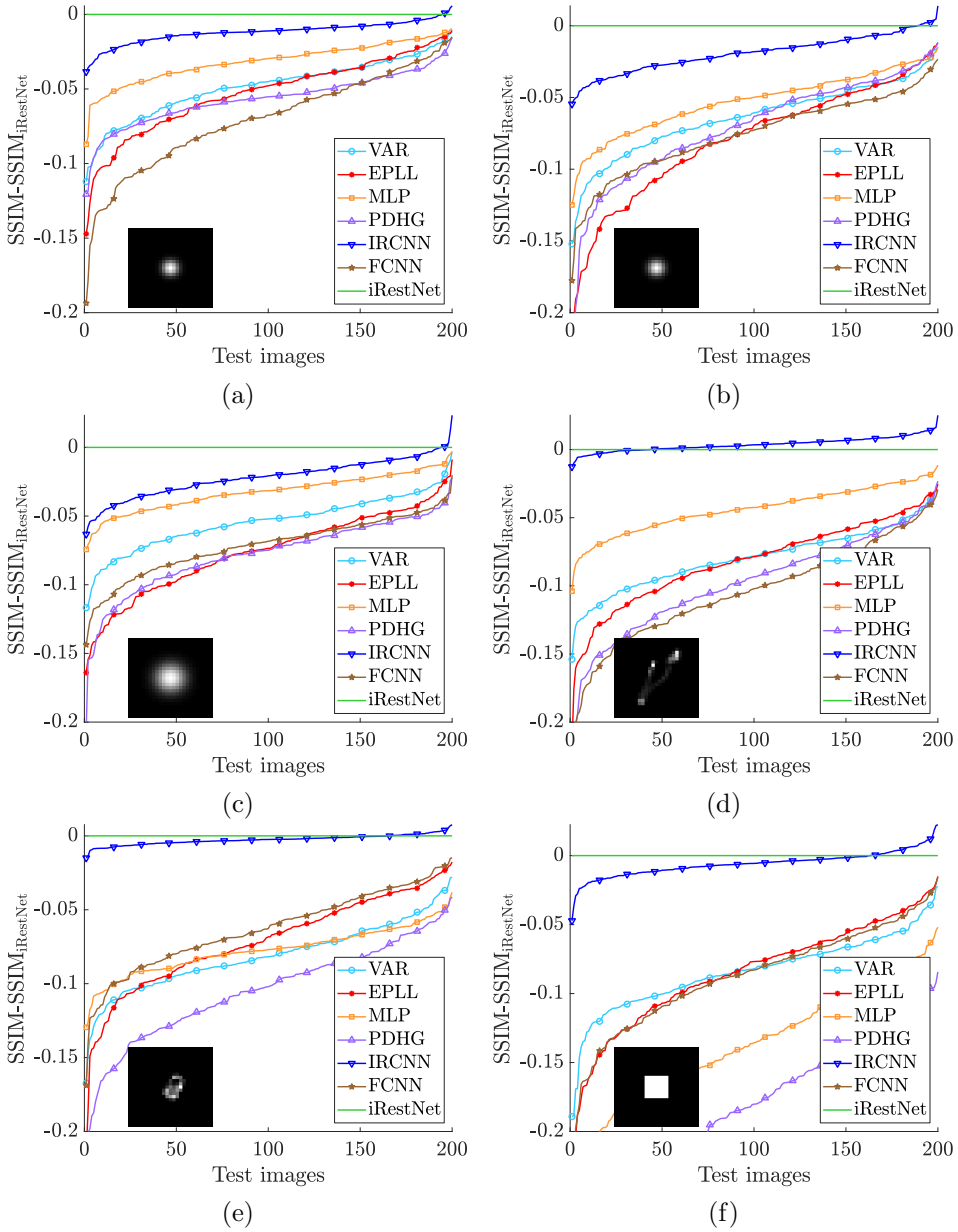

Figure 5. Sorted improvement of iRestNet with regards to other methods on the BSD500 test set using the SSIM metric: a negative value indicates a better performance of iRestNet. (a): GaussianA, (b): GaussianB, (c): GaussianC, (d): MotionA, (e): MotionB, (f): Square.

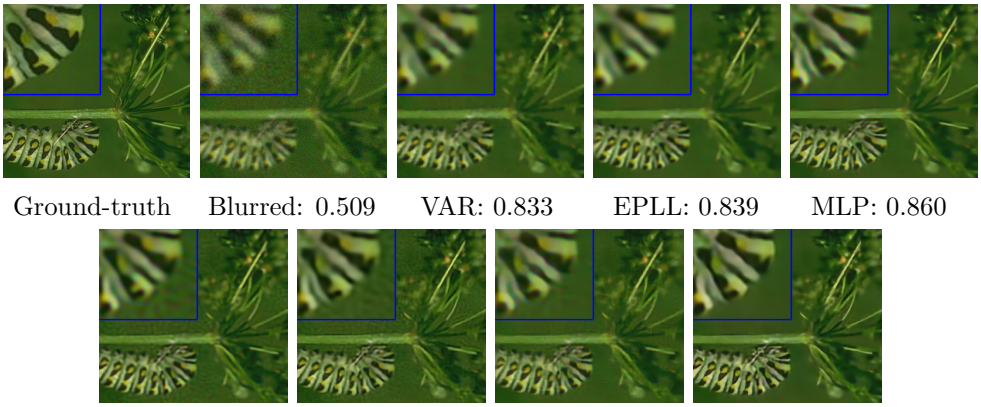

Figure 6. Visual results and SSIM obtained with the different methods on one image from the BSD500 test set degraded with GaussianB.

contains the average SSIM obtained with the different methods on the Flickr30 test set. Similarly to the BSD500 test set, iRestNet compares favorably with the other approaches on the Flickr30 test set.

	GaussianA	GaussianB	GaussianC	MotionA	MotionB	Square
Blurred	0.723	0.545	0.355	0.376	0.590	0.579
VAR	0.857	0.776	0.639	0.856	0.869	0.818
EPLL [68]	0.860	0.770	0.616	0.857	0.887	0.827
MLP [10]	0.874	0.798	0.668	0.891	0.873	0.787
PDHG [70]	0.853	0.781	0.623	0.855	0.854	0.721
IRCNN [62]	0.885	0.819	0.676	**0.927**	**0.930**	**0.886**
FCNN [69]	0.846	0.766	0.614	0.801	0.890	0.822
iRestNet	**0.892**	**0.833**	**0.696**	0.919	**0.930**	**0.886**

Table 3. SSIM results on the Flickr30 test set.

Examples of visual results obtained with the different methods can be found in Figures 6 and 7 for two images from the BSD500 test set and the blur kernels GaussianB and Square, respectively. We also provide the results obtained for one image from the Flickr30 test set that has been degraded with MotionB. As one can see from inspecting these pictures, details from the snake's and caterpillar's skin patterns are better retrieved with iRestNet, which provides more visually-satisfactory results than competitors. Similarly, on Figure 8, competitors tend to smooth too much the details on the leaves as it can be seen in the top left-hand corner. Regarding Figure 6, which belongs to the test set with a level-varying noise, it is worth noting that, on the result obtained with the proposed method, the green background is free from artifacts, which is not the case for PDHG and IRCNN. This suggests that those two competitors are not robust to a small change in the noise level.

Figure 9 shows the barrier parameter, stepsize and regularization weight sequences obtained by passing the image from Figure 6 through the 40 layers of iRestNet. The oscillations in these plots are most likely due to the fact that the first 30 layers are

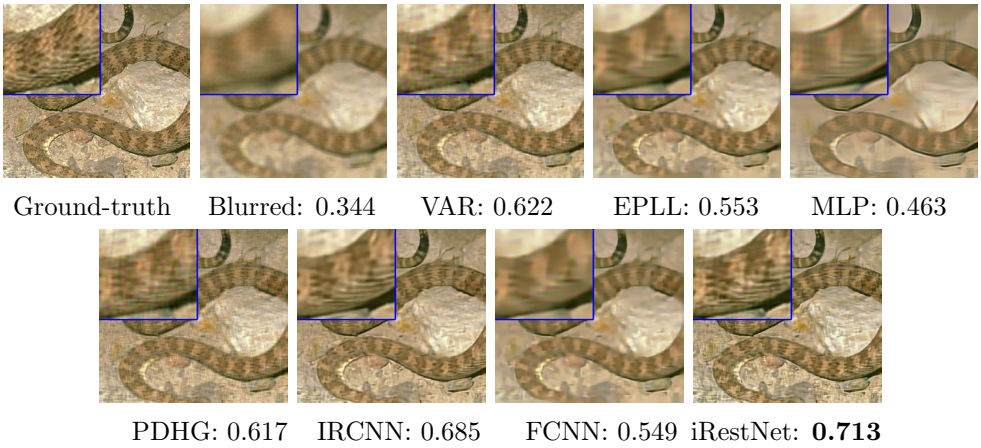

Figure 7. Visual results and SSIM obtained with the different methods on one image from the BSD500 test set degraded with Square.

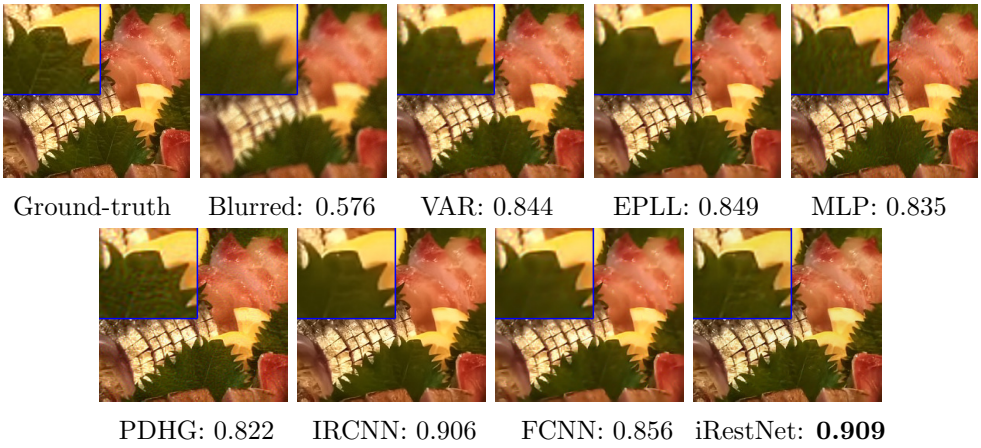

Figure 8. Visual results and SSIM obtained with the different methods on one image from the Flickr30 test set degraded with MotionB.

trained individually, while the increasing values for the stepsize in the last 10 layers are probably a consequence of the effect of the post-processing layer during the second part of the training. It is worth noting that the largest value for the barrier parameter is reached at the first layer, which was expected since, in interior point methods, the barrier parameter vanishes along the iterations.

7. Conclusion

From a variational formulation of an inverse problem, we have derived in this paper a novel neural network architecture by unfolding a proximal interior point algorithm. It can be noted that the proposed approach can be extended to a set of regularization functions, or to penalizations which are parametrized by several variables. Useful constraints on the sought solution can be enforced thanks to a logarithmic barrier, so providing more control over the output of the network. We have

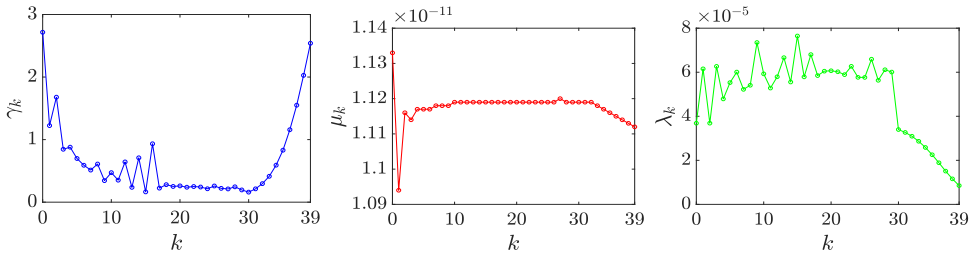

Figure 9. Left to right: estimated parameters $(\gamma_k)_{0\leq k\leq K-1}$, $(\mu_k)_{0\leq k\leq K-1}$ and $(\lambda_k)_{0\leq k\leq K-1}$ for the image from Figure 6 passed through the network layers.

shown for three standard types of constraints that the involved proximity operator can easily be computed, and that its derivatives are well-defined and computable. In the case of a quadratic cost function, the theoretical result of Section 5 regarding the robustness of the network with respect to an input perturbation, ensures the reliability of the proposed method, which is crucial for many applications. It would be interesting to extend the scope of this study to a wider class of problems, and to illustrate this stability result by numerical experiments on different applications like classification. As demonstrated by our experiments in image restoration, iRestNet performs favorably compared to state-of-the-art variational and machine learning methods. An advantage of the proposed approach is that, in contrast with its evaluated competitors, it does not require any knowledge about the noise level and it does not involve any hand-selection of the regularization parameters. One limitation of iRestNet is that the network needs to be trained for a given blur kernel. A direction for future works is to extend the method to situations in which the observation model is not fully known, so as to address blind or semi-blind deconvolution problems.

References

[1] R L Lagendijk and J Biemond. *The Handbook of Image and Video Processing*. Academic Press, New-York, N. J., USA, 2005.

[2] L Xu and J Jia. Two-phase kernel estimation for robust motion deblurring. In *Proceedings of the 11th European Conference on Computer Vision (ECCV)*, pages 157–170. Springer, 5-11 September 2010.

[3] J-F Aujol. Some first-order algorithms for total variation based image restoration. *Journal of Mathematical Imaging and Vision*, 34(3):307–327, July 2009.

[4] N Pustelnik, A Benazza-Benhayia, Y Zheng, and J-C Pesquet. *Wavelet-based image deconvolution and reconstruction*. John Wiley & Son, February 2016.

[5] O Scherzer. The use of Morozov's discrepancy principle for Tikhonov regularization for solving nonlinear ill-posed problems. *Computing*, 51(1):45–60, 1993.

[6] C-A Deledalle, S Vaiter, J Fadili, and G Peyré. Stein Unbiased GrAdient estimator of the Risk (SUGAR) for multiple parameter selection. *SIAM Journal on Imaging Sciences*, 7(4):2448–2487, 2014.

[7] Y Yao, L Rosasco, and A Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[8] K Zhang, W Zuo, Y Chen, D Meng, and L Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[9] L Xu, J S J Ren, C Liu, and J Jia. Deep convolutional neural network for image deconvolution. In *Proceedings of the International Conference on Advances in Neural Information Processing Systems (NIPS)*, pages 1790–1798, 2014.

[10] C Schuler, H C Burger, S Harmeling, and B Schölkopf. A machine learning approach for non-

blind image deconvolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1067–1074. IEEE, 2013.

[11] C J Schuler, M Hirsch, S Harmeling, and B Scholkopf. Learning to deblur. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 38(7):1439, 2016.

[12] C Ledig, L Theis, F Huszár, J Caballero, A Cunningham, A Acosta, A Aitken, A Tejani, J Totz, Z Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017.

[13] K H Jin, M T McCann, E Froustey, and M Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017.

[14] M T McCann, K H Jin, and M Unser. Convolutional neural networks for inverse problems in imaging: A review. *IEEE Signal Processing Magazine*, 34(6):85–95, 2017.

[15] C Szegedy, W Zaremba, I Sutskever, J Bruna, D Erhan, I Goodfellow, and R Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[16] D Boublil, M Elad, J Shtok, and M Zibulevsky. Spatially-adaptive reconstruction in computed tomography using neural networks. *IEEE transactions on medical imaging*, 34(7):1474–1485, 2015.

[17] J R Hershey, J Le Roux, and F Weninger. Deep unfolding: model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*, 2014.

[18] J-T Chien and C-H Lee. Deep unfolding for topic models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):318–331, 2018.

[19] S Wang, S Fidler, and R Urtasun. Proximal deep structured models. In *Advances in Neural Information Processing Systems*, pages 865–873, 2016.

[20] M Mardani, H Monajemi, V Papyan, S Vasanawala, D Donoho, and J Pauly. Recurrent generative adversarial networks for proximal learning and automated compressive image recovery. *arXiv preprint arXiv:1711.10046*, 2017.

[21] S Diamond, V Sitzmann, F Heide, and G Wetzstein. Unrolled optimization with deep priors. *arXiv preprint arXiv:1705.08041*, 2017.

[22] K Gregor and Y LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML)*, pages 399–406, Haifa, Israel, 21-24 June 2010. Omnipress.

[23] U S Kamilov and H Mansour. Learning optimal nonlinearities for iterative thresholding algorithms. *IEEE Signal Processing Letters*, 23(5):747–751, 2016.

[24] J. Zhang and B. Ghanem. ISTA-Net: Interpretable optimization–inspired deep network for image compressive sensing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1828–1837, Jun 2018.

[25] J Sun, H Li, Z Xu, and Y Yang. Deep ADMM-Net for compressive sensing MRI. In *Advances in Neural Information Processing Systems*, pages 10–18, 2016.

[26] U Schmidt and S Roth. Shrinkage fields for effective image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2774–2781, Columbus, USA, 24-27 June 2014.

[27] J Sun and Z Xu. Color image denoising via discriminatively learned iterative shrinkage. *IEEE Transactions on Image Processing*, 24(11):4148–4159, 2015.

[28] Y Chen and T Pock. Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1256–1272, 2017.

[29] M Andrychowicz, M Denil, S Gomez, M W Hoffman, D Pfau, T Schaul, B Shillingford, and N De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.

[30] K Li and J Malik. Learning to optimize. *arXiv preprint arXiv:1606.01885*, 2016.

[31] B Amos and J Z Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 136–145, 6-11 August 2017.

[32] T B Trafalis, T A Tutunji, and N P Couëllan. Interior point methods for supervised training of artificial neural networks with bounded weights. In P M Pardalos, D W Hearn, and W W Hager, editors, *Network Optimization*, pages 441–470, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[33] P T Krasopoulos and N G Maratos. *An Interior Point Recurrent Neural Network for Convex Optimization Problems*, pages 409–427. Springer New York, New York, NY, 2014.

[34] S. Harizanov, J.-C. Pesquet, and G. Steidl. Epigraphical projection for solving least squares Anscombe transformed constrained optimization problems. In *Proceedings of the 4th*

International Conference on Scale Space and Variational Methods in Computer Vision (SSVM), pages 125–136, Schloss Seggau, Graz, Austria, Jun 2013. Springer.

[35] O. Musse, F. Heitz, and J.-P. Armspach. Topology preserving deformable image matching using constrained hierarchical parametric models. *IEEE Transactions on Image Processing*, 10(7):1081–1093, 2001.

[36] M. Klodt and D. Cremers. A convex framework for image segmentation with moment constraints. In *Proceeding of the 13th International Conference on Computer Vision (ICCV)*, pages 2236–2243. IEEE, Dec 2011.

[37] S Durand and M Nikolova. Stability of minimizers of regularized least squares objective functions I: study of the local behaviour. *Applied Mathematics and Optimization (Springer-Verlag New York)*, 53:185–208, 2006.

[38] H H Bauschke and P L Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2017.

[39] A N Iusem. On the convergence properties of the projected gradient method for convex optimization. *Computational Applied Mathematics*, 22(1):37–52, 2003.

[40] S Bonettini and M Prato. New convergence results for the scaled gradient projection method. *Inverse Problems*, 31(9):1–20, 2015.

[41] S Boyd, N Parikh, E Chu, B Peleato, and J Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–222, 2011.

[42] N Komodakis and J-C Pesquet. Playing with duality: An overview of recent primal-dual approaches for solving large-scale optimization problems. *IEEE Signal Processing Magazine*, 32(6):31–54, Nov. 2014.

[43] S Bonettini and T Serafini. Non-negatively constrained image deblurring with an inexact interior point method. *Journal of Computational and Applied Mathematics*, 231(1):236 – 248, 2009.

[44] P L Combettes and V R Wajs. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling and Simulation*, 4(4):1168–1200, Nov 2005.

[45] P L Combettes and J-C Pesquet. Proximal splitting methods in signal processing. In H H Bauschke, R Burachik, P L Combettes, V Elser, D R Luke, and H Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer-Verlag, New York, 2010.

[46] E. Chouzenoux, M.-C. Corbineau, and J.-C. Pesquet. A proximal interior point algorithm with applications to image processing. *HAL preprint hal-02120005*, 2019.

[47] M-C Corbineau, E Chouzenoux, and J-C Pesquet. PIPA: a new proximal interior point algorithm for large-scale convex optimization. In *Proceedings of the 43rd IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, Calgary, Canada, 15-20 April 2018.

[48] M-C Corbineau, E Chouzenoux, and J-C Pesquet. Geometry-texture decomposition/reconstruction using a proximal interior point algorithm. In *Proceedings of the 10th IEEE Sensor Array and Multichannel Signal processing Workshop (SAM 2018)*, 8-11 July 2018.

[49] M H Wright. Interior methods for constrained optimization. *Acta numerica*, 1:341–407, 1992.

[50] A Kaplan and R Tichatschke. Proximal methods in view of interior-point strategies. *Journal of Optimization Theory and Applications*, 98(2):399–429, 1998.

[51] M.-O. Czarnecki, N. Noun, and J. Peypouquet. Splitting forward-backward penalty scheme for constrained variational problems. *Journal of Convex Analysis*, 23(2):531–565, 2016.

[52] C Chaux, P L Combettes, J-C Pesquet, and V R Wajs. A variational formulation for frame-based inverse problems. *Inverse Problems*, 23(4):1495–1518, June 2007.

[53] A L Dontchev and R T Rockafellar. Implicit functions and solution mappings. *Springer Monogr. Math.*, 2009.

[54] M S Bartlett. An inverse matrix adjustment arising in discriminant analysis. *The Annals of Mathematical Statistics*, 22(1):107–111, 1951.

[55] C Dugas, Y Bengio, F Bélisle, C Nadeau, and R Garcia. Incorporating second-order functional knowledge for better option pricing. In *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, pages 472–478, 2001.

[56] C R Vogel. *Computational methods for inverse problems*, volume 23. SIAM, 2002.

[57] A Ramadhan, F Mahmood, and A Elci. Image denoising by median filter in wavelet domain. *arXiv preprint arXiv:1703.06499*, 2017.

[58] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. De Vito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop: The Future of Gradient-based Machine Learning Software and Techniques*, Long Beach, CA,

USA, Dec 2017.

[59] P L Combettes and J-C Pesquet. Deep neural network structures solving variational inequalities. *arXiv preprint arXiv:1808.07526*, 2018.

[60] J Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[61] S. Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[62] K Zhang, W Zuo, S Gu, and L Zhang. Learning deep CNN denoiser prior for image restoration. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[63] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[64] A Levin, Y Weiss, F Durand, and W T Freeman. Understanding and evaluating blind deconvolution algorithms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1964–1971, Miami, USA, 20-25 June 2009.

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

[66] Z Wang, A C Bovik, H R Sheikh, and E P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[67] I Goodfellow, Y Bengio, and A Courville. *Deep Learning*. MIT Press, 2016.

[68] D Zoran and Y Weiss. From learning models of natural image patches to whole image restoration. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 479–486. IEEE, 2011.

[69] J. Zhang, J. Pan, W.-S. Lai, R. W. H. Lau, and M.-H. Yang. Learning fully convolutional networks for iterative non-blind deconvolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3817–3825, 2017.

[70] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1781–1790, Venice, Italy, Oct 2017.