The Little Engine that Could Regularization by Denoising (RED)

Yaniv Romano* Michael Elad‡ Peyman Milanfar‡

Abstract

Removal of noise from an image is an extensively studied problem in image processing. Indeed, the recent advent of sophisticated and highly effective denoising algorithms lead some to believe that existing methods are touching the ceiling in terms of noise removal performance. Can we leverage this impressive achievement to treat other tasks in image processing? Recent work has answered this question positively, in the form of the Plug-and-Play Prior (P^3) method, showing that any inverse problem can be handled by sequentially applying image denoising steps. This relies heavily on the ADMM optimization technique in order to obtain this chained denoising interpretation.

Is this the only way in which tasks in image processing can exploit the *image denoising engine*? In this paper we provide an alternative, more powerful and more flexible framework for achieving thef same goal. As opposed to the P^3 method, we offer Regularization by Denoising (RED): using the denoising engine in defining the regularization of the inverse problem. We propose an explicit image-adaptive Laplacian-based regularization functional, making the overall objective functional clearer and better defined. With a complete flexibility to choose the iterative optimization procedure for minimizing the above functional, RED is capable of incorporating any image denoising algorithm, treat general inverse problems very effectively, and is guaranteed to converge to the globally optimal result. We test this approach and demonstrate state-of-the-art results in the image deblurring and super-resolution problems.

keywords: Image Denoising Engine, Plug-and-Play Prior, Laplacian Regularization, Inverse Problems.

*The Electrical Engineering Department, The Technion - Israel Institute of Technology.

‡Google Research, Mountain View, California.

1 Introduction

We open this paper with a bold and possibly controversial statement: *To a large extent, removal of zero-mean white additive Gaussian noise from an image is a solved problem in image processing.*

Before justifying this statement, let us describe the basic building block that will be the star of this paper: *the image denoising engine*. From the humble linear Gaussian filter to the recently developed state-of-the-art methods using convolutional neural networks, there is no shortage of denoising approaches. In fact, these algorithms are so widely varied in their definition and underlying structure that a concise description will need to be made carefully. Our story begins with an image $\mathbf{x}$ corrupted by zero-mean white additive Gaussian noise,

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

In our notation, we consider an image as a vector of length n (after lexicographic ordering). In the above description, the noise vector is normally distributed, $\mathbf{e} \sim \mathcal{N}\left(\underline{0}, \sigma^2\mathbf{I}\right)$. In the most general terms, the image denoising engine is a function $f : [0,1]^n \longrightarrow [0,1]^n$ that maps an image $\mathbf{y}$ to another image of the same size $\widehat{\mathbf{x}} = f(\mathbf{y})$, with the hope to get as close as possible to the original image $\mathbf{x}$. Ideally, such functions operate on the input image $\mathbf{y}$ to remove the deleterious effect of the noise while maintaining edges and textures beneath.

The claim made above about the denoising problem being solved is based on the availability of algorithms proposed in the past decade that can treat this task extremely effectively and stably, getting very impressive results, which also tend to be quite concentrated (see for example the work reported in [3–18, 42]). Indeed, these documented results have led researchers to the educated guess that these methods are getting very close to the optimally possible denoising performance [21–23]. This aligns well with the unspoken observation in our community in recent years that investing more work to improve image denoising algorithms seems to lead to diminishing returns.

While the above may suggest that work on denoising algorithms is turning to a dead-end avenue, a new opportunity emerges from this trend: Seeking ways to leverage the vast progress made on the image denoising front in order to treat other tasks in image processing, bringing their solutions to new heights. One natural path towards addressing this goal is to take an existing and well-performing denoising algorithm, and generalize it to handle a new problem. This has been the logical path that has led to contributions such as [24–29], and many others. These papers, and others like them, offer an exhaustive manual adaptation of existing denoising algorithms, carefully re-tailored to handle specific alternative problems. This line of work, while often successful, is quite limited, as it offers no flexibility and no general scheme for diverting image denoising engines to treat new image processing tasks.

Could one offer a more systematic way to exploit the abundance of high-performing image-

denoising algorithms to treat a much broader family of problems? The recent work by Venkatakrishnan, Bouman and Wohlberg provides a positive and tantalizing answer to this question, in the form of the Plug-and-Play Prior (P^3) method [30–33]. This technique builds on the use of an implicit prior for regularizing general inverse problems. When handling the obtained optimization task via the ADMM optimization scheme [56–58], the overall problem decomposes into a sequence of image denoising tasks, coupled with simpler L_2-regularized inverse problems that are much easier to handle.

While the P^3 scheme may sound like the perfect answer to our prayers, reality is somewhat more complicated. First, this method is not accompanied by a clear definition of the objective function, since the regularization being effectively used is only implicitly implied by the denoising algorithm. Indeed, it is unclear at all that there is an underlying objective function behind the P^3 scheme, if arbitrary denoising engines are used [32]. Secondly, parameter tuning of the ADMM scheme is a delicate matter, and especially so under a non-provable convergence regime, as is the case when using sophisticated denoising algorithms. Third, being intimately coupled with the ADMM, the P^3 scheme does not offer easy and flexible ways for accelerating the convergence of the iterative procedure. Because of these reasons, the P^3 scheme is not a turn-key tool, nor is it free from emotional-involvement. Nevertheless, the P^3 method has drawn much attention (e.g., [31–38]), and rightfully so, as it offers a clear path towards harnessing a given image denoising engine for treating more general inverse problems, just as described above.

Is there a more general alternative to the P^3 method that could be simpler and more stable? This paper puts forward such a framework, offering a systematic use of such denoising engines for regularization of inverse problems. We term the proposed method "Regularization by Denosing" (RED), relying on a general structured smoothness penalty term plugged to regularize any desired inverse problem. More specifically, the regularization term we propose in this work is of the following

$$\rho(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}^T\left[\mathbf{x} - f(\mathbf{x})\right], \tag{2}$$

in which the denoising engine itself is applied on the candidate image $\mathbf{x}$, and the penalty induced is proportional to the inner-product between this image and its denoising residual. This defined smoothness regularization is effectively using an image-adaptive Laplacian, which in turn draws its definition from the arbitrary image denoising engine of choice, $f(\cdot)$. Surprisingly, under mild assumptions on $f(\cdot)$, it is shown that the gradient of the regularization term is manageable, given as the denoising residual, $\mathbf{x} - f(\mathbf{x})$. Therefore, armed with this regularization expression, we show that any inverse problem can be handled while calling the denoising engine iteratively.

RED, the newly proposed framework, is much more flexible in the choice of the optimization method to use, not being tightly coupled to one specific technique, as in the case of the P^3 scheme (relying on ADMM). Another key difference w.r.t. the P^3 method is that our adaptive Laplacian-based regularization functional is explicit, making the overall Bayesian objective

function clearer and better defined. RED is capable of incorporating any image denoising algorithm, and can treat general inverse problems very effectively, while resulting in an overall algorithm with very simple structure.

An important advantage of RED over the P^3 scheme is the flexibility with which one can choose the denoising engine $f(\cdot)$ to plug in the regularization term. While most of the discussion in this paper keeps focusing on White Gaussian Noise (WGN) removal, RED can actually deploy almost any denoising engine. Indeed, we define a set of four mild conditions that $f(\cdot)$ should satisfy, and show that many known denoising methods obey these properties. As an example, in our experiments we show how the median filter can become an effective regularizer. Last but not least, we show that the defined regularization term is a convex function, implying that in most cases, in which the log-likelihood term is convex too, the proposed algorithms are guaranteed to converge to a global optimum solution. We demonstrate this scheme, showing state-of-the-art results in image deblurring and single image super-resolution.

This paper is organized as follows: In the next section we present the background material for this work, discussing the general form of inverse problems as optimization tasks, and presenting the Plug-and-Play Prior scheme. Section 3 focuses on the image denoising engine, defining it and its properties clearly, so as to enable its use in the proposed Laplacian paradigm. Section 4 serves the main part of this work – introducing RED: a new way to use an image denoising engine to handle general structured inverse problems. In Section 5 we analyze the proposed scheme, discussing convexity, an alternative formulation, and a qualitative comparison to the P^3 scheme. Results on the image deblurring and single-image super-resolution problems are brought in Section 6, demonstrating the strength of the proposed scheme. We conclude the paper in Section 7 with a summary of the open questions that we identify for future work.

2 Preliminaries

In this section we provide background material that serves as the foundation to this work. We start by presenting the breed of optimization tasks we will work on throughout the paper for handling the inverse problems of interest. We then introduce the P^3 method and discuss its merits and weaknesses.

2.1 Inverse Problems as Optimization Tasks

Bayesian estimation of an unknown image $\mathbf{x}$ given its measured version $\mathbf{y}$ uses the posterior conditional probability, $P(\mathbf{x}|\mathbf{y})$, in order to infer $\mathbf{x}$. The most popular estimator in this regime is the Maximum a'posteriori Probability (MAP), which chooses the mode ($\mathbf{x}$ for which the maximum probability is obtained) of the posterior. Using the Bayes' rule, this implies that the

estimation task is turned into an optimization problem of the form

$$\begin{aligned}\widehat{\mathbf{x}}_{MAP} = \underset{\mathbf{x}}{\text{Argmax}}\ P(\mathbf{x}|\mathbf{y}) &= \underset{\mathbf{x}}{\text{Argmax}}\ \frac{P(\mathbf{y}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{y})} = \underset{\mathbf{x}}{\text{Argmax}}\ P(\mathbf{y}|\mathbf{x})P(\mathbf{x}) \\ &= \underset{\mathbf{x}}{\text{Argmin}}\ -log\{P(\mathbf{y}|\mathbf{x})\} - log\{P(\mathbf{x})\}.\end{aligned} \tag{3}$$

In the above derivations we exploited the fact that $P(\mathbf{y})$ is not a function of $\mathbf{x}$ and thus can be omitted. We also used the fact that the $-log$ function is monotonic decreasing, turning the maximization into a minimization problem.

The term $-log\{P(\mathbf{y}|\mathbf{x})\}$ is known as the log-likelihood term, and it encapsulates the probabilistic relationship between the desired image $\mathbf{x}$ and the measurements $\mathbf{y}$, under the assumption that the desired image is known. We shall rewrite this term as

$$\ell(\mathbf{y}, \mathbf{x}) = -log\{P(\mathbf{y}|\mathbf{x})\}. \tag{4}$$

As a classic example for the log-likelihood that will accompany us throughout this paper, the expression $\ell(\mathbf{y}, \mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{H}\mathbf{x} - \mathbf{y}\|_2^2$ refers to the case of $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{e}$, where $\mathbf{H}$ is a linear degradation operator and $\mathbf{e}$ is white Gaussian noise contamination of variance σ^2. Naturally, if the noise distribution changes, we depart form the comfortable L_2 form.

The second term in Equation (3), $-log\{P(\mathbf{x})\}$, refers to the *prior*, bringing in the influence of the unknown's statistics. This term is also referred to as the regularization, as it helps in better conditioning the overall optimization task in cases where the likelihood alone cannot lead to a unique or stable solution. We shall rewrite this term as

$$\lambda\rho(\mathbf{x}) = -log\{P(\mathbf{x})\}, \tag{5}$$

where λ is a scalar that encapsulates the confidence in this term.

Who is $\rho(\mathbf{x})$ and how is it chosen? This is the holy-grail of image processing, with a progressive advancement over the years of better modeling the image statistics and leveraging this for handling various tasks in image processing. Indeed, one could claim that almost everything done in our field surrounds this quest for choosing a proper prior, from the early smoothness prior $\rho(\mathbf{x}) = \lambda\mathbf{x}^T\mathbf{L}\mathbf{x}$ using the classic Laplacian [39], through total variation [40] and wavelet sparsity [41], all the way to recent proposals based on patch-based GMM [42, 43] and sparse-representation modeling [44]. Interestingly, the work we report here builds on the surprising comeback of the Laplacian regularization in a much more sophisticated form, as reported in [46–55].

Armed with a clear definition of the relation between the measurements and the unknown, and with a trusted prior, the MAP estimation boils down to the optimization problem of the form

$$\widehat{\mathbf{x}}_{MAP} = \underset{\mathbf{x}}{\text{Argmin}}\ \ell(\mathbf{y}, \mathbf{x}) + \lambda\rho(\mathbf{x}). \tag{6}$$

This defines a wide family of inverse problems that we aim to address in this work, which includes tasks such as denoising, deblurring, super-resolution, demosaicing, tomographic reconstruction, optical-flow estimation, segmentation, and many other problems. The randomness in these problems is typically due to noise contamination of the measurements, and this could be Gaussian, Laplacian, Gamma-distributed, Poisson, and other noise models.

2.2 The Plug-and-Play Prior (P^3) Approach [30]

For completeness of this exposition, we briefly review the P^3 approach. Aiming to solve the problem posed in Equation (6), the ADMM technique [56–58] suggests to handle this by variable splitting, leading to the equivalent problem

$$\{\widehat{\mathbf{x}}_{MAP}, \widehat{\mathbf{v}}\} = \underset{\mathbf{x},\mathbf{v}}{\text{Argmin}}\ \ell(\mathbf{y},\mathbf{x}) + \lambda\rho(\mathbf{v}) \quad s.t. \quad \mathbf{x} = \mathbf{v}. \tag{7}$$

The constraint is turned into an exact penalty term, relying on the augmented Lagrangian method, leading to

$$\{\widehat{\mathbf{x}}_{MAP}, \widehat{\mathbf{v}}, \widehat{\mathbf{u}}\} = \underset{\mathbf{x},\mathbf{v},\mathbf{u}}{\text{Argmin}}\ \ell(\mathbf{y},\mathbf{x}) + \lambda\rho(\mathbf{v}) + \frac{\beta}{2}\|\mathbf{x}-\mathbf{v}+\mathbf{u}\|_2^2, \tag{8}$$

where $\mathbf{u}$ serves as the Lagrange multiplier vector for the set of constraints. ADMM addresses the resulting problem by updating $\mathbf{x}$, $\mathbf{v}$, and $\mathbf{u}$ sequentially in a block-coordinate-descent fashion, leading to the following series of sub-problems:

1. **Update of x:** When considering $\mathbf{v}$ (and $\mathbf{u}$) as fixed, the term $\rho(\mathbf{v})$ is omitted, and our task becomes

   $$\widehat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}}\ \ell(\mathbf{y},\mathbf{x}) + \frac{\beta}{2}\|\mathbf{x}-\mathbf{v}+\mathbf{u}\|_2^2, \tag{9}$$

   which is a far simpler inverse problem, where the regularization is an L_2 proximity one, which is easy to solve in most cases.

2. **Update of v:** In this stage we freeze $\mathbf{x}$ (and $\mathbf{u}$), and thus the log-likelihood term drops, leading to

   $$\widehat{\mathbf{v}} = \underset{\mathbf{v}}{\text{Argmin}}\ \lambda\rho(\mathbf{v}) + \frac{\beta}{2}\|\mathbf{x}-\mathbf{v}+\mathbf{u}\|_2^2. \tag{10}$$

   This stage is nothing but a denoising of the image $\mathbf{x}+\mathbf{u}$, assumed to be contaminated by a white additive Gaussian noise of power $\sigma^2 = 1/\beta$. This is easily verified by returning to Equation (6) and plugging the log-likelihood term $\|\mathbf{v}-\mathbf{x}-\mathbf{u}\|_2^2/2\sigma^2$ referring to this case. Indeed, this is the prime observation in [30], as they suggest to replace the direct solution of (10) by activating an image denoising engine of choice. This way, we do not need to define explicitly the regularization $\rho(\cdot)$ to be used, as it is implicitly implied by the engine chosen.

3. **Update of u:** We complete the algorithm description by considering the update of the Lagrange multiplier vector $\mathbf{u}$, which is done by $\widehat{\mathbf{u}} = \mathbf{u} + \mathbf{x} - \mathbf{v}$.

Although the above algorithm has a clear mathematical formulation and only two parameters, denoted by β and λ, it turns out that tuning these is not a trivial task. The source of complexity emerges from the fact that the input noise-level to the denoiser is equal to $\sqrt{\lambda/\beta}$. The confidence in the prior is determined by λ, and the penalty on the distance between $\mathbf{x}$ and $\mathbf{v}$ is effected by β. Empirically, setting a fixed value of β does not seize the potential of this algorithm; following previous work (e.g. [33,38]), a common practical strategy to achieve a high-quality estimation is to increase the value of β as a function of the iterations: Starting from a relatively small value, i.e allowing an aggressive regularization, then proceeding to a more conservative one that limits the smoothing effect, up-to a point where β should be large enough to ensure convergence [33] and to avoid an undesired over-smoothed outcome. As one can imagine, it is cumbersome to choose the rate in which β should be increased, especially because the corrupted image $\mathbf{x} + \mathbf{u}$ is a function of the Lagrange multiplier, which varies through the iterations as well.

In terms of convergence, the P^3 scheme has been shown to be well-behaved under some conditions on the denoising algorithm used. While the work reported in [31] requires the denoiser to be a symmetric smoothing and non-expansive filter, the later work in [33] relaxes this condition to much simpler boundedness of the denoising effect. However, both these works prove at best a convergence to a steady-state outcome, which is very far from the desired claim of getting to the global minimizer of the overall objective function.

Indeed, in that respect, a delicate matter with the P^3 approach is the fact that given a choice of a denoising engine, it does not necessarily refer to a specific choice of a prior $\rho(\cdot)$, as not every such engine could have a MAP-oriented interpretation. This implies a fundamental difficulty in the P^3 scheme, as in this case we will be activating a denoising algorithm while departing from the original setting we have defined, and having no underlying cost function to serve. Indeed, the work reported in [32] addresses this very matter in a narrower setting, by studying the identity of the effective prior obtained from a chosen denoising engine. The author chooses to limit the answer to symmetric smoothing filters, showing that even in this special case, the outcome is far from being trivial. As we are about to see in the next section, this shortcoming can be removed by adopting a different regularization strategy.

3 The *Image Denoising Engine*

Image denoising is a special case of the inverse problem posed in Equation (6), referring to the case $\mathbf{y} = \mathbf{x} + \mathbf{e}$, where $\mathbf{e}$ is white Gaussian noise contamination of variance σ^2. In this case, the MAP problem becomes

$$\widehat{\mathbf{x}}_{Denoise} = \underset{\mathbf{x}}{\text{Argmin}} \frac{1}{2\sigma^2}\|\mathbf{y} - \mathbf{x}\|_2^2 + \lambda\rho(\mathbf{x}). \tag{11}$$

The *image denoising engine*, which is the focal point of this work, is any candidate solver to the above problem, under a specific choice of a prior. In fact, in this work we choose to widen the definition of the image denoising engine to be any function $f : [0,1]^n \longrightarrow [0,1]^n$ that maps an image $\mathbf{y}$ to another image $f(\mathbf{y})$ of the same size, and which aims to treat the denoising problem by the operation $\widehat{\mathbf{x}} = f(\mathbf{y})$, be it MAP-based, MMSE-based, or any other approach. We accompany the definition of a denoiser with several basic assumptions on the function f:

- **Condition 1: Constancy.** A denoiser $f(\mathbf{y})$ must not alter a constant image. More specifically, defining $\mathbb{1}$ as an n-dimensional column vector of all ones, for any scalar $c \geq 0$ we must have $f(c\mathbb{1}) = c\mathbb{1}$.

- **Condition 2: (Local) Homogeneity.** A denoiser applied to a positively scaled image $f(c\mathbf{y})$ should result in a scaled version of the original image. More specifically, for any scalar $c \geq 0$ we must have $f(c\mathbf{y}) = cf(\mathbf{y})$. In this work we shall relax this condition and demand its satisfaction for $|c-1| \leq \epsilon$ for a very small ϵ.

- **Condition 3: Directional Derivative.** The directional derivative of the denoiser $f(\mathbf{x})$ along $\mathbf{x}$ must exist[1], and can be evaluated as follows:

$$\nabla_{\mathbf{x}} f(\mathbf{x})\,\mathbf{x} = \frac{f(\mathbf{x}+\epsilon\mathbf{x}) - f(\mathbf{x})}{\epsilon} \tag{12}$$

  for a very small ϵ. Invoking the homogeneity condition this leads to

$$\begin{aligned} \nabla_{\mathbf{x}} f(\mathbf{x})\,\mathbf{x} &= \frac{(1+\epsilon)f(\mathbf{x}) - f(\mathbf{x})}{\epsilon} \\ &= f(\mathbf{x}). \end{aligned} \tag{13}$$

  Thus, the filter $f(\mathbf{x})$ can be written as[2]

$$f(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})\,\mathbf{x}. \tag{14}$$

- **Condition 4: Strong Passivity.** The Jacobian $\nabla_{\mathbf{x}} f(\mathbf{x})$ of the denoising algorithm is stable, satisfying the condition

$$\|\nabla_{\mathbf{x}} f(\mathbf{x})\| \leq 1. \tag{15}$$

  We interpret this condition as the restriction of the denoiser not to magnify the norm of an input image since

$$\|f(\mathbf{x})\| = \|\nabla_{\mathbf{x}} f(\mathbf{x})\mathbf{x}\| \leq \|\nabla_{\mathbf{x}} f(\mathbf{x})\| \cdot \|\mathbf{x}\| \leq \|\mathbf{x}\|. \tag{16}$$

[1] Note that this is a much weaker condition than a general differentiability of $f(\mathbf{x})$.

[2] This result is sometimes known as Euler's homogeneous function theorem [62].

Here we have relied on the relation $f(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x})\mathbf{x}$, that has been established in Equation (14). Note that we have chosen this specific condition over the natural weaker alternative, $\|f(\mathbf{x})\| \leq \|\mathbf{x}\|$, since the strong condition implies the weak one.

While the above condition may seem harsh, it is in fact satisfied if we require $\nabla_{\mathbf{x}} f(\mathbf{x})$ to be row-stochastic for all $\mathbf{x}$, since then its spectral norm (and it Ł$_\infty$-induced norm) is 1. For this to hold true, we should require that (i) this matrix is non-negative, $\nabla_{\mathbf{x}} f(\mathbf{x}) \geq 0$, and that (ii) the vector $\mathbb{1}$ is an eigenvector of $\nabla_{\mathbf{x}}$. The second condition has a close relation to Condition 1 posed above, since it demands this very property for $\nabla_{\mathbf{x}} f(c\mathbb{1})$. We note here that algorithms such as the NLM [1] and its many variants all lead to such a row-stochastic Jacobian.

A reformulation of the denoising engine that will be found useful throughout this work is the one suggested in [50], where we assume that the algorithm is built of two phases – a first in which highly non-linear decisions are made, and a second in which these decisions are used to adapt a linear filter to the raw noisy image in order to perform the actual noise removal. Algorithms such as the NLM, kernel-regression, K-SVD, and many others admit this structure, and for them we can write

$$\widehat{\mathbf{x}}_{Denoise} = f(\mathbf{y}) = \mathbf{W}(\mathbf{y})\mathbf{y}. \tag{17}$$

The matrix $\mathbf{W}$ is an $n \times n$ matrix, representing the (pseudo-) linear filter that multiplies the $n \times 1$ noisy image vector $\mathbf{y}$. This matrix is image dependent, as it draws its content from the pixels in $\mathbf{y}$. Nevertheless, this pseudo-linear format provides a convenient description of the denoising engine for our later derivations. We should emphasize that while this notation is true for only some of the denoising algorithms, the proposed framework we outline below is general and applies to *any* denoising filter that satisfies the four conditions posed above.

The careful reader will observe that this pseudo-linear form is closely related to the above-mentioned properties of the function f. First, observe that the constancy property implies that $\mathbf{W}(\mathbf{y})$ should be a row-stochastic matrix, or any other matrix with the vector $\mathbb{1}$ being its eigenvector. Secondly, considering the directional derivative condition shown above, we have seen that $f(\mathbf{y}) = \nabla_{\mathbf{y}} f(\mathbf{y})\,\mathbf{y}$, and in this form, the right hand side is now reminding us of the pseudo-linear form where matrix $\mathbf{W}(\mathbf{y})$ is replaced by the scaled Jacobian matrix $\nabla_{\mathbf{y}} f(\mathbf{y})$.

We cannot conclude this section without answering the key question: Who are those denoising engines we are constantly referring to? While these could include any of the thousands of denoising algorithms published over the years, we obviously focus on the best performing ones, such as the non-local means and its advanced variants [1–3], the K-SVD denoising method that relies on sparse representation modeling of image patches [4] and its non-local extension [7], the kernel-regression method that exploits local orientation [6], the well-known BM3D that combines sparsity and self-similarity of patches [5], the EPLL scheme that suggests patch-modeling based on the GMM model [42], CSR and NCSR, which cluster the patches and

sparsifies them jointly [8,9], the group-Wiener filtering applied on patches [10], the multi-layer Perceptron method trained to clean an image [11], more recent work that proposed low-rank modeling of patches and the use of the weighted nuclear-norm [16], non-local sparsity with GSM model [17], and the list goes on and on. Each and every one of these options (and many others) is a candidate engine that could be fit into our scheme.

4 Regularization by Denoising (RED)

4.1 The Image-Adaptive Laplacian

The new and alternative framework we propose relies on a form of an image-adaptive Laplacian which builds a powerful (empirical) prior that can be used to regularize a variety of inverse problems. As a place to start and motivate this definition, let's go back to the description of the denoiser given in Equation (17), namely[3] $\mathbf{W}(\mathbf{x})\mathbf{x}$. We may think of this linearized filter as one where a set of coefficients (depending on $\mathbf{x}$) are first computed in the matrix $\mathbf{W}$, and then applied to the image $\mathbf{x}$. From this we can construct the Laplacian form,

$$\rho_L(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}^T\mathbf{L}(\mathbf{x})\mathbf{x} = \frac{1}{2}\,\mathbf{x}^T(\mathbf{I}-\mathbf{W}(\mathbf{x}))\mathbf{x} = \frac{1}{2}\,\mathbf{x}^T\left[\mathbf{x}-\mathbf{W}(\mathbf{x})\mathbf{x}\right]. \tag{18}$$

This definition by itself is not novel, as it is similar to ideas brought up in a series of recent contributions [46–55]. This expression relies on using an image adaptive Laplacian – one that draws its definition from the image itself.

Observing the obtained expression, we note that it can be interpreted as the *unnormalized cross-correlation* between the image $\mathbf{x}$ and its corresponding residual $\mathbf{x}-\mathbf{W}(\mathbf{x})\mathbf{x}$. As a *prior* expression should give low values for likely images, in our case this would be achieved in one of two ways (or their combination):

- A small value is obtained for $\rho_L(\mathbf{x})$ if the residual is very small, implying that the image $\mathbf{x}$ serves as a near fixed-point of the denoising engine, $\mathbf{x} \approx \mathbf{W}(\mathbf{x})\mathbf{x}$.

- A small value is obtained for $\rho_L(\mathbf{x})$ if the cross-correlation of the residual to the image itself is small, a feature that implies that the residual behaves like white noise, or alternatively, if it does not contain elements from the image itself. Interestingly, this concept has been harnessed successfully by some denoising algorithms such as the Dantzig-Selector [59] and by image denoising boosting techniques [50, 60, 61]. Indeed, enforcing orthogonality between the signal and its treated residual is the underlying force behind the Normal equations in statistical estimation (e.g. Least Squares and Kalman Filtering).

[3]Note that we conveniently assume that the prior is applied to the clean image $\mathbf{x}$, a matter that will be clarified as we dive into our explanations.

Given the above prior, we return to the general inverse-problem posed in Equation (6), and define our new objective,

$$\widehat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}}\ \ell(\mathbf{y},\mathbf{x}) + \frac{\lambda}{2}\,\mathbf{x}^T\left[\mathbf{x} - \mathbf{W}(\mathbf{x})\mathbf{x}\right]. \tag{19}$$

The prior expression, while exhibiting a possibly complicated dependency on the unknown $\mathbf{x}$, is well-defined and clear. Nevertheless, an attempt to apply any gradient-based algorithm for solving the above minimization task encounters an immediate problem, due to the need to differentiate $\mathbf{W}(\mathbf{x})$ with respect to $\mathbf{x}$. We overcome this problem by observing that $\mathbf{W}(\mathbf{x})\mathbf{x}$ is in fact the activation of the image denoising engine on $\mathbf{x}$, i.e., $f(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$. This observation inspires the following more general definition of the Laplacian regularizer, which is the prime message of this paper:

$$\rho_L(\mathbf{x}) = \frac{1}{2}\,\mathbf{x}^T\left[\mathbf{x} - f(\mathbf{x})\right]. \tag{20}$$

This is the Regularization by Denoising (RED) paradigm that this work advocates. In this expression, the residual is defined more generally for any filter $f(\mathbf{x})$ even if it can not be written in the familiar (pseudo-)linear form. Note that all the preceding intuition about the meaning of this prior remains intact; namely, the value is low if the cross-correlation between the image and its denoising residual is small, or if the residual itself is small due to $\mathbf{x}$ being a fixed point of f.

Surprisingly, while this expression is more general, it leads to a better-managed optimization problem due to the careful properties we have outlined in Section 3 on our denoising engines f. The overall energy functional to minimize is

$$E(\mathbf{x}) = \ell(\mathbf{y},\mathbf{x}) + \frac{\lambda}{2}\,\mathbf{x}^T\left(\mathbf{x} - f(\mathbf{x})\right), \tag{21}$$

and the gradient of this expression is readily available by

$$\begin{aligned}
\nabla_{\mathbf{x}} E(\mathbf{x}) &= \nabla_{\mathbf{x}}\ell(\mathbf{y},\mathbf{x}) + \frac{\lambda}{2}\,\nabla_{\mathbf{x}}\left\{\mathbf{x}^T\left(\mathbf{x} - f(\mathbf{x})\right)\right\} \\
&= \nabla_{\mathbf{x}}\ell(\mathbf{y},\mathbf{x}) + \frac{\lambda}{2}\,\nabla_{\mathbf{x}}\mathbf{x}^T\mathbf{x} - \frac{\lambda}{2}\,\nabla_{\mathbf{x}}\left[\mathbf{x}^T f(\mathbf{x})\right] \\
&= \nabla_{\mathbf{x}}\ell(\mathbf{y},\mathbf{x}) + \lambda\mathbf{x} - \frac{\lambda}{2}\left[f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x})\mathbf{x}\right].
\end{aligned} \tag{22}$$

Based on our prior assumption regarding the availability of a directional derivative for the denoising engine (Condition 3), the term $\nabla_{\mathbf{x}} f(\mathbf{x})\mathbf{x}$ can be replaced[4] by $\nabla_{\mathbf{x}} f(\mathbf{x})\,\mathbf{x} = f(\mathbf{x})$, based on Equation (14), implying that the gradient expression is further simplified to be

$$\nabla_{\mathbf{x}} E(\mathbf{x}) = \nabla_{\mathbf{x}}\ell(\mathbf{y},\mathbf{x}) + \lambda\mathbf{x} - \lambda f(\mathbf{x}) = \nabla_{\mathbf{x}}\ell(\mathbf{y},\mathbf{x}) + \lambda(\mathbf{x} - f(\mathbf{x})), \tag{23}$$

[4] A better approximation can be applied in which we replace $\nabla_{\mathbf{x}} f(\mathbf{x})\mathbf{x}$ by the difference $(f((1+\epsilon)\mathbf{x}) - f(\mathbf{x}))/\epsilon$, but this calls for two activations of the denoising engine per one gradient evaluation.

requiring only one activation of the denoising engine for the gradient evaluation. Interestingly, if we bring back now the pseudo-linear interpretation of the denoising engine, the gradient would be the residual, just as posed above, implying that

$$\nabla_{\mathbf{x}} E(\mathbf{x}) = \nabla_{\mathbf{x}} \ell(\mathbf{y}, \mathbf{x}) + \lambda(\mathbf{x} - \mathbf{W}(\mathbf{x})\mathbf{x}). \tag{24}$$

Observe that this is a non-trivial derivation of the gradient of the original penalty function posed in Equation (19).

4.2 Deploying the Denoising Engine for Solving Inverse Problems

In the discussion above we have seen that the gradient of the energy functional to minimize (given in Equation (21)) is easily computable, given in Equation (23). We now turn to show several options for using this in order to solve a general inverse problem. Common to all these methods is the fact that the eventual algorithm is iterative, in which each step is composed of applying the denoising engine (once or more), accompanied by other simpler calculations. In Figures 1, 2, and 3 we present pseudo-code for several such algorithms, all in the context of handling the case in which the likelihood function is given by $\ell(\mathbf{y}, \mathbf{x}) = \|\mathbf{H}\mathbf{x} - \mathbf{y}\|_2^2 / 2\sigma^2$.

- **Gradient Descent Methods:** Given the gradient of the energy function $E(\mathbf{x})$, the Steepest-Descent (SD) is the simplest option that can be considered, and it amounts to the update formula

$$\begin{aligned} \widehat{\mathbf{x}}_{k+1} &= \widehat{\mathbf{x}}_k - \mu \nabla_{\mathbf{x}} E(\mathbf{x})|_{\widehat{\mathbf{x}}_k} \\ &= \widehat{\mathbf{x}}_k - \mu \left[\nabla_{\mathbf{x}} \ell(\mathbf{y}, \mathbf{x})|_{\widehat{\mathbf{x}}_k} + \lambda(\widehat{\mathbf{x}}_k - f(\widehat{\mathbf{x}}_k)) \right]. \end{aligned} \tag{25}$$

  Figure 1 describes this algorithm in more details.

  A line-search can be proposed in order to set μ dynamically per iteration, but this is necessarily more involved. For example, in the case of the Armijo rule, it requires a computation of the above gradient $\mathbf{g}_k$ and then assessing the energy $E(\widehat{\mathbf{x}}_k - \mu \mathbf{g}_k)$ for different values of μ in a retracting fashion, each of which calling for a computation of the denoising engine once.

  One could envision using the Conjugate-Gradient (CG) to speed this method, or better yet, applying the Sequential Subspace Optimization (SESOP) algorithm [65]. SESOP holds the current gradient and the last several update directions as the columns of a matrix $\mathbf{V}_k$ (referring to the k^{th} iteration), and seeks the best linear combination of these columns as an update direction to the current solution, namely $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{V}_k \boldsymbol{\alpha}_k$. When restricted to have only one column, this reduces to a simple SD with line-search. When using two columns, it has the flavor (and strength) of CG, and when using more columns,

this method can lead to much faster convergence in non-quadratic problems. The key points of SESOP are (i) The matrix $\mathbf{V}$ is updated easily from one iteration to another by discarding the last direction, bringing in the last one, and adding the new gradient; and (ii) The unknown weights vector $\boldsymbol{\alpha}_k$ is low-dimensional, and thus updating it can be done using a Newton method. Naturally, one should evaluate the first and second derivatives of the penalty function w.r.t. $\boldsymbol{\alpha}_k$, and these will leverage the relations established above. We shall not dive deeper into this option because it will not be included in our experiments.

One possible shortcoming of the gradient approach (in all its manifestations) is the fact that per each activation of the denoising engine, the likelihood is updated rather mildly as a simple step toward the current log-likelihood gradient. This may imply that the overall algorithm will require many iterations to converge. The next two methods propose a way to overcome this limitation, by treating the log-likelihood term more "aggressively".

Objective: Minimize $E(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|_2^2 + \frac{\lambda}{2}\mathbf{x}^T(\mathbf{x}-f(\mathbf{x}))$

Input: Supply the following ingredients and parameters:

- Regularization parameter λ,
- Denoising algorithm $f(\cdot)$ and its noise level σ_f,
- Log-Likelihood parameters: $\mathbf{H}$ and σ, and
- Number of iterations N.

Initialization:

- Set $\widehat{\mathbf{x}}_0 = \mathbf{y}$.
- Set $\mu = \frac{2}{1/\sigma^2+\lambda}$.

Loop: For $k = 1, 2 \ \ldots \ , N$ do:

- Apply the denoising engine, $\tilde{\mathbf{x}}_k = f_{\sigma_f}(\widehat{\mathbf{x}}_{k-1})$.
- Update the solution by $\widehat{\mathbf{x}}_k = \widehat{\mathbf{x}}_{k-1} - \mu\left[\frac{1}{\sigma^2}\mathbf{H}^T(\mathbf{H}\widehat{\mathbf{x}}_{k-1}-\mathbf{y}) + \lambda(\widehat{\mathbf{x}}_{k-1}-\tilde{\mathbf{x}}_k)\right]$.

End of Loop

Result: The output of the above algorithm is $\widehat{\mathbf{x}}_N$.

Figure 1: The proposed scheme (RED) via the steepest descent method.

- **ADMM:** Addressing the optimization task given in Equation (21), we can imitate the path taken by the P^3 scheme, and apply variable splitting and ADMM. The steps would follow the description given in Section 2 almost exactly, with one major difference – the prior in our case is explicit, and therefore, the stage termed "update of $\mathbf{v}$" would become

$$\widehat{\mathbf{v}} = \underset{\mathbf{v}}{\text{Argmin}} \ \frac{\lambda}{2}\mathbf{v}^T(\mathbf{v}-f(\mathbf{v})) + \frac{\beta}{2}\|\mathbf{v}-\mathbf{x}-\mathbf{u}\|_2^2. \tag{26}$$

Rather than applying an arbitrary denoising engine to compute $\widehat{\mathbf{v}}$ as a replacement to the actual minimization, we should target this minimization directly by some iterative scheme. For example, setting the gradient of the above expression to zero leads to the equation

$$\lambda(\mathbf{v} - f(\mathbf{v})) + \beta(\mathbf{v} - \mathbf{x} - \mathbf{u}) = \mathbf{0}, \tag{27}$$

which can be solved iteratively using the fixed-point strategy, by

$$\begin{aligned} &\lambda(\mathbf{v}_j - f(\mathbf{v}_{j-1})) + \beta(\mathbf{v}_j - \mathbf{x} - \mathbf{u}) = \mathbf{0} \\ \rightarrow \quad &\mathbf{v}_j = \frac{1}{\beta + \lambda}\left(\lambda f(\mathbf{v}_{j-1} + \beta(\mathbf{x} + \mathbf{u})\right). \end{aligned} \tag{28}$$

This means that our approach in this case is computationally more expensive, as it will require several activations of the denoising engine. Figure 2 describes this specific algorithm in more details. If one changes all **Part 2** with the computation $\widehat{\mathbf{v}}_k = f_{1/\sqrt{\beta}}(\mathbf{z}^*)$, we obtain the P^3 scheme for the same choice of the denoising engine. While this difference is quite delicate, we should remind the reader that (i) this bridge between the two approaches is valid only when we deploy ADMM on our scheme, and (ii) as opposed to the P^3 method, our method is guaranteed to converge to the global optimum of the overall penalty function, as will be described hereafter.

We should point out that when using the ADMM, the update of $\mathbf{x}$ applies an aggressive inversion of the log-likelihood term, which is followed by the above optimization task. Thus, the shortcoming mentioned above regarding the lack of balance between the treatments given to the likelihood and the prior is mitigated.

- **Fixed-Point Strategy:** An appealing alternative to the above exists, obtained via the fixed-point strategy. As our aim is to find $\mathbf{x}$ that nulls the gradient, this could be posed as an implicit equation to be solved directly,

$$\nabla_{\mathbf{x}}\ell(\mathbf{y}, \mathbf{x}) + \lambda(\mathbf{x} - f(\mathbf{x})) = 0. \tag{29}$$

Using the fixed-point strategy, this could be handled by the iterative formula

$$\nabla_{\mathbf{x}}\ell(\mathbf{y}, \mathbf{x}_{k+1}) + \lambda(\mathbf{x}_{k+1} - f(\mathbf{x}_k)) = 0. \tag{30}$$

As an example, in the case of linear degradation model and Gaussian white additive noise, this equation would be

$$\frac{1}{\sigma^2}\mathbf{H}^T(\mathbf{y} - \mathbf{H}\mathbf{x}_{k+1}) + \lambda(\mathbf{x}_{k+1} - f(\mathbf{x}_k)) = 0, \tag{31}$$

Objective: Minimize $E(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|_2^2 + \frac{\lambda}{2}\mathbf{x}^T(\mathbf{x}-f(\mathbf{x}))$

Input: Supply the following ingredients and parameters:

- Regularization parameter λ,
- Denoising algorithm $f(\cdot)$ and its noise level σ_f,
- Log-Likelihood parameters: $\mathbf{H}$ and σ,
- Number of outer and inner iterations, N, m_1, and m_2, and
- ADMM coefficient β.

Initialization: Set $\widehat{\mathbf{x}}_0 = \mathbf{y}$, $\widehat{\mathbf{v}}_0 = \mathbf{y}$, and $\widehat{\mathbf{u}}_0 = \mathbf{0}$.

Outer Loop: For $k = 1, 2 \ \ldots \ , N$ do:

Part 1: Solve $\widehat{\mathbf{x}}_k = \underset{\mathbf{z}}{\text{Argmin}} \ \frac{1}{2\sigma^2}\|\mathbf{H}\mathbf{z}-\mathbf{y}\|_2^2 + \frac{\beta}{2}\|\mathbf{z}-\widehat{\mathbf{v}}_{k-1}+\widehat{\mathbf{u}}_{k-1}\|_2^2$ by

- Initialization: $\mathbf{z}_0 = \widehat{\mathbf{x}}_{k-1}$, and define $\mathbf{z}^* = \widehat{\mathbf{v}}_{k-1} + \widehat{\mathbf{u}}_{k-1}$.
- Inner Iteration: For $j = 1, 2 \ \ldots \ , m_1$ do:
  - Compute the gradient $\mathbf{e}_j = \frac{1}{\sigma^2}\mathbf{H}^T(\mathbf{H}\mathbf{z}_{j-1}-\mathbf{y}) + \beta(\mathbf{z}_{j-1}-\mathbf{z}^*)$.
  - Compute $\mathbf{r}_j = \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H}\mathbf{e}_j + \beta\mathbf{e}_j$.
  - Compute the step size $\mu = \mathbf{e}_j^T\mathbf{e}_j / \mathbf{e}_j^T\mathbf{r}_j$.
  - Update the solution by $\mathbf{z}_j = \mathbf{z}_{j-1} + \mu\mathbf{e}_j$.
  - Project the result to the interval $[0, 255]$.
- End of Inner Loop
- Set $\widehat{\mathbf{x}}_k = \mathbf{z}_{m_1}$.

Part 2: Solve $\widehat{\mathbf{v}}_k = \underset{\mathbf{z}}{\text{Argmin}} \ \lambda\mathbf{z}^T(\mathbf{z}-f_{\sigma_f}(\mathbf{z})) + \frac{\beta}{2}\|\mathbf{z}-\widehat{\mathbf{x}}_k-\widehat{\mathbf{u}}_{k-1}\|_2^2$ by

- Initialization: $\mathbf{z}_0 = \widehat{\mathbf{v}}_{k-1}$, and define $\mathbf{z}^* = \widehat{\mathbf{x}}_k + \widehat{\mathbf{u}}_{k-1}$.
- Inner Iteration: For $j = 1, 2 \ \ldots \ , m_2$ do:
  - Apply the denoising engine, $\tilde{\mathbf{z}}_j = f_{\sigma_f}(\widehat{\mathbf{z}}_{j-1})$.
  - Compute the gradient $\mathbf{z}_j = \frac{1}{\beta+\lambda}(\lambda\tilde{\mathbf{z}}_j + \beta\mathbf{z}^*)$.
- End of Inner Loop
- Set $\widehat{\mathbf{v}}_k = \mathbf{z}_{m_2}$.

Part 3: Update $\widehat{\mathbf{u}}_k = \widehat{\mathbf{u}}_{k-1} + \widehat{\mathbf{x}}_k - \widehat{\mathbf{u}}_k$.

End of Outer Loop

Result: The output of the above algorithm is $\widehat{\mathbf{x}}_N$.

Figure 2: The proposed scheme (RED) via the ADMM method.

leading to the recursive update relation

$$\mathbf{x}_{k+1} = \left[\frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I}\right]^{-1}\left[\frac{1}{\sigma^2}\mathbf{H}^T\mathbf{y} + \lambda f(\mathbf{x}_k)\right]. \tag{32}$$

This formula suggests one activation of the denoising per iteration, followed by what seems to be a plain Wiener filtering computation[5]. The matrix inversion itself could be done in the Fourier domain for block-circulant $\mathbf{H}$, or iteratively using for example, the Richardson algorithm: Defining

$$\mathbf{A} = \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I} \quad \text{and} \quad \mathbf{b} = \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{y} + \lambda f(\mathbf{x}_k), \tag{33}$$

our goal is to solve the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$. This is achieved by a variant of the SD method[6], $\mathbf{x}_{j+1} = \mathbf{x}_j - \mu(\mathbf{A}\mathbf{x}_j - \mathbf{b}) = \mathbf{x}_j - \mu\mathbf{e}_j$, where we have defined $\mathbf{e}_j = \mathbf{A}\mathbf{x}_j - \mathbf{b}$. By setting the step size to be $\mu_j = \mathbf{e}_j^T\mathbf{A}\mathbf{e}_j / \mathbf{e}_j^T\mathbf{A}^T\mathbf{A}\mathbf{e}_j$, we greedily optimize the potential of each iteration.

Convergence of the above algorithm is guaranteed since

$$\left\| \left[\frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I}\right]^{-1} \lambda\nabla_{\mathbf{x}} f(\mathbf{x}_k) \right\| < 1. \tag{34}$$

This approach, similarly to the ADMM, has the desired balance mentioned above between the likelihood and the regularization terms, matching the efforts dedicated to both. A pseudo-code describing this algorithm appears in Figure 3.

A basic question that has not been discussed so far is how to set the parameters of $f(\mathbf{x})$ in defining the regularization term. More specifically, assuming that the denoising engine depends on one parameter – the noise standard-deviation σ_f – the question is which value to use. While one could envision using varying values as the iterations progress and the outcome improves, the approach we take in this work is to set this parameter to be a small and fixed value. Our intuition for this choice is the desire to have a clear and fixed regularization term, which in turn implies a clear cost function to work with. Furthermore, the prior we propose should encapsulate in it our desire to get to a final image that is a stable point of such a weak denoising engine, $\mathbf{x} \approx f(\mathbf{x})$. Clearly, more work is required to better understand the influence of this parameter and its automatic setting.

[5]Note that Equation (28) refers to the same problem posed here under the choice $\mathbf{H} = \mathbf{I}$ and $\beta = 1/\sigma^2$.

[6]All this refer to a specific iteration k within which we apply inner iterations to solve the linear system, and thus the use of the different index j.

Objective: Minimize $E(\mathbf{x}) = \frac{1}{2\sigma^2}\|\mathbf{H}\mathbf{x}-\mathbf{y}\|_2^2 + \frac{\lambda}{2}\mathbf{x}^T\left(\mathbf{x} - f(\mathbf{x})\right)$

Input: Supply the following ingredients and parameters:

- Regularization parameter λ,
- Denoising algorithm $f(\cdot)$ and its noise level σ_f,
- Log-Likelihood parameters: $\mathbf{H}$ and σ, and
- Number of outer and inner iterations, N and m.

Initialization: $\widehat{\mathbf{x}}_0 = \mathbf{y}$.

Outer Loop: For $k = 1, 2\ \ldots\ , N$ do:

- Apply the denoising engine, $\tilde{\mathbf{x}}_k = f_{\sigma_f}(\widehat{\mathbf{x}}_{k-1})$.
- Solve $\mathbf{A}\mathbf{z} = \mathbf{b}$ for $\mathbf{A} = \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I}$ and $\mathbf{b} = \frac{1}{\sigma^2}\mathbf{H}^T\mathbf{y} + \lambda f(\tilde{\mathbf{x}}_k)$

  - Initialization: $\mathbf{z}_0 = \tilde{\mathbf{x}}_k$.

  - Iterate: For $j = 1, 2\ \ldots\ , m$ do:

    - Compute the residual $\mathbf{r}_j = \mathbf{A}\mathbf{z}_{j-1} - \mathbf{b}$.
    - Compute the vector $\mathbf{e}_j = \mathbf{A}\mathbf{r}_j$.
    - Compute the optimal step $\mu = \mathbf{r}_j^T\mathbf{e}_j / \mathbf{e}_j^T\mathbf{e}_j$.
    - Update the solution by $\mathbf{z}_j = \mathbf{z}_{j-1} + \mu \cdot \mathbf{r}_j$.
    - Project the result to the interval $[0, 255]$.

  - End of Inner Loop

  - Set $\widehat{\mathbf{x}}_k = \mathbf{z}_m$.

End of Outer Loop

Result: The output of the above algorithm is $\widehat{\mathbf{x}}_N$.

Figure 3: The Proposed Scheme (RED) via the fixed-point method.

5 Analysis

5.1 Convexity

Is our proposed regularization function $\rho_L(\mathbf{x})$ convex? At first glance, this may seem like too much to expect. Nevertheless, it appears that for reasonably performing denoising engines that follow the conditions posed in Section 3, this is exactly the case. For the function $\rho_L(\mathbf{x}) = \mathbf{x}^T(\mathbf{x} - f(\mathbf{x}))$ to be convex, we should demand that the second derivative is a positive semi-definite matrix [63]. We have already seen that the first derivative is simply $\mathbf{x} - f(\mathbf{x})$, which leads to the conclusion that the second derivative is given by $\mathbf{I} - \nabla_{\mathbf{x}} f(\mathbf{x})$.

As already mentioned earlier, in the context of some algorithms such as the NLM and the K-SVD, this is associated with the Laplacian $\mathbf{I} - \mathbf{W}(\mathbf{x})$, and it is positive semi-definite if $\mathbf{W}$ has all its eigenvalues in the range[7] $[0, 1]$. This is indeed the case for the NLM filter [1], the K-SVD-denoising algorithm [55], and many other denoising engines.

In the wider context of general image denoising engines, convexity is assured if the Jacobian $\nabla_{\mathbf{x}} f(\mathbf{x})$ of the denoising algorithm is stable, as indeed required in Condition 4 in Section 3, $\|\nabla_{\mathbf{x}} f(\mathbf{x})\| \leq 1$. In this case we have that $\rho_L(\cdot)$ is convex, and this implies that if the log-likelihood expression is convex as well, the proposed scheme is guaranteed to converge to the global optimum of our cost function in Equation (6). This statement poses the proposed algorithm as superior to the P^3 scheme, which at best is known to get to a stable-point [30, 33, 34].

5.2 An Alternative Prior

In Section 4 we motivated the choice of the proposed prior by the desire to characterize the unknown image $\mathbf{x}$ as one that is not affected by the denoising algorithm, namely, $\mathbf{x} \approx f(\mathbf{x})$. Rather than taking the route proposed, we could have suggested a prior of the form

$$\rho_Q(\mathbf{x}) = \|\mathbf{x} - f(\mathbf{x})\|_2^2. \tag{35}$$

This prior term makes sense intuitively, being based on the same desire to see the denoising residual being small. Indeed, this choice is somewhat related to the option we chose since

$$\rho_Q(\mathbf{x}) = \|\mathbf{x} - f(\mathbf{x})\|_2^2 = \rho_L(\mathbf{x}) + f(\mathbf{x})^T(f(\mathbf{x}) - \mathbf{x}), \tag{36}$$

suggesting a symmetrization of our own expression.

In order to understand the deeper meaning of this alternative, we resort again to the pseudo-linear denoisers, for which this prior is nothing but

$$\rho_Q(\mathbf{x}) = \|\mathbf{x} - \mathbf{W}(\mathbf{x})\mathbf{x}\|_2^2 = \mathbf{x}^T(\mathbf{I} - \mathbf{W}(\mathbf{x}))^T(\mathbf{I} - \mathbf{W}(\mathbf{x}))\mathbf{x}. \tag{37}$$

[7] We could in fact allow negative eigenvalues for $\mathbf{W}$, but this is unnatural in the context of denoising.

This means that rather than regularizing with the Laplacian, we do so with its square. While this is a worthy possibility which has been considered in the literature under the term "fourth order regularization" [66], it is known to be more delicate. We leave this and other possibilities of formulating the regularization with the use of $f(\mathbf{x})$ for future work.

5.3 Relation to the P^3 Scheme

In Section 4 we described the use of ADMM as one of the possible avenues for handling our proposed regularization. When handling the inverse problem posed in Equation (21) with ADMM, we have shown that the only difference between this and the P^3 scheme resides in the update stage for $\mathbf{v}$. Here we aim to answer the following question: Assuming that the numerical algorithm used is indeed the ADMM, under which conditions would the two methods (P^3 and ours) become equivalent? The answer to this question resides in the optimization task for updating $\mathbf{v}$, which is a denoising task. Thus, purifying this question, our goal is to find conditions on $f(\cdot)$ and λ such that the two treatments of this update stage coincide. Starting from our approach, we would seek the solution of

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{Argmin}} \frac{\lambda}{2}\mathbf{x}^T(\mathbf{x} - f(\mathbf{x})) + \frac{\beta}{2}\|\mathbf{x} - \mathbf{y}\|_2^2, \tag{38}$$

or, putting it in terms of nulling the gradient of this energy, require

$$\lambda(\mathbf{x} - f(\mathbf{x})) + \beta(\mathbf{x} - \mathbf{y}) = \mathbf{0}. \tag{39}$$

The $\widehat{\mathbf{x}}$ that is the solution of this equation is our updated image. On the other hand, the P^3 scheme would propose to simply compute[8] $\widehat{\mathbf{x}} = f(\mathbf{y})$ as a replacement to this minimization task. Therefore, for the two methods to align, we should demand that the gradient expression posed above is solved for the choice of the P^3 scheme, namely,

$$\lambda(f(\mathbf{y}) - f(f(\mathbf{y}))) + \beta(f(\mathbf{y}) - \mathbf{y}) = \mathbf{0}, \tag{40}$$

or posed slightly different,

$$f(\mathbf{y}) - f(f(\mathbf{y})) = \frac{\beta}{\lambda}(\mathbf{y} - f(\mathbf{y})). \tag{41}$$

This means that the denoising residual should remain the same (up to a constant) for the first activation of the denoising engine $\mathbf{y} - f(\mathbf{y})$, and the second one applied on the filtered image $f(\mathbf{y})$.

[8] A delicate matter not considered here is that P^3 may apply $\frac{1}{c}f(c\mathbf{y})$ in order to tune to a specific noise level. We assume $c = 1$ for simplicity.

In order to get a better intuition towards this result, let's return to the pseudo-linear case, $f(\mathbf{y}) = \mathbf{W}\mathbf{y}$ with the assumption that $\mathbf{W}$ is a fixed and diagonalizable matrix. Plugged into the above condition, this gives

$$\mathbf{W}\mathbf{y} - \mathbf{W}^2\mathbf{y} = \frac{\beta}{\lambda}(\mathbf{y} - \mathbf{W}\mathbf{y}), \tag{42}$$

or posed differently,

$$\left(\frac{\beta}{\lambda}\mathbf{I} - \mathbf{W}\right)(\mathbf{I} - \mathbf{W})\,\mathbf{y} = \mathbf{0}. \tag{43}$$

As the above equation should hold true for any image $\mathbf{y}$, we require

$$\left(\frac{\beta}{\lambda}\mathbf{I} - \mathbf{W}\right)(\mathbf{I} - \mathbf{W}) = \mathbf{0}. \tag{44}$$

Without loss of generality, we can assume that $\mathbf{W}$ is diagonal, after multiplying the above equation from the left and right by the diagonalizing matrix. With this simplification in mind, we now consider the eigenvalues of $\mathbf{W}$, and the above equation implies that exact equivalence between our scheme and the P^3 one is obtained only if our denoising engine has eigenvalues that are purely 1-s, or β/λ. Clearly, this is a very limiting case, which suggests that for all other cases, the two methods are likely to differ.

Interestingly, the above analysis is somewhat related to the one given in [32]. Both [32] and our treatment assume that the actual applied denoising engine is $f(\mathbf{y}) = \mathbf{W}\mathbf{y}$ within the ADMM scheme. While we ask for the conditions on $\mathbf{W}$ to fit our regularization term $\mathbf{x}^T(\mathbf{x} - \mathbf{W}\mathbf{x})$, the author of [32] seeks the actual form of the prior to match this step, getting to the conclusion that the prior should be $\mathbf{x}^T(\mathbf{I} - \mathbf{W})\mathbf{W}^\dagger\mathbf{x}$. Bearing in mind that the conditions we get for the equivalence between the two methods are too restricting and rarely met, the result in [32] shows the actual gap between the two methods: While we regularize with the expression $\mathbf{x}^T(\mathbf{I} - \mathbf{W})\mathbf{x}$, an equivalence takes place only if the P^3 modifies this to involve $\mathbf{W}^\dagger$, getting a far more complicated and less natural term. Last but not least, note that all the discussion in [32] is shadowed by the restriction to symmetric denoising engines (symmetric smoothing), while our scheme has the flexibility to use any denoising engine.

6 Results

In this section we compare the performance of the proposed framework to the P^3 approach, along with various other leading algorithms that are designed to tackle the image deblurring and super-resolution problems. To this end, we plug two substantially different denoising algorithms into the proposed scheme. The first is the (simple) *median filter*, which surprisingly turns out to

act as a reasonable regularizer to the involved ill-posed inverse problem. This option is brought as a core demonstration of the idea that an arbitrary denoiser can be deployed in RED without difficulties. The second denoising engine we use is the state-of-the-art Trainable Nonlinear Reaction Diffusion (TNRD) [20] method. This algorithm trains a nonlinear reaction-diffusion model in a *supervised* manner. As such, in order to treat different restoration problems, one should re-train the underlying model for every specific task – something we aim to avoid. In the experiments below we build upon the published pre-trained model by the authors of TNRD, tailored to denoise images that are contaminated by white Gaussian noise with a fixed[9] noise-level, which is equal to 5. Leveraging this, we show how state-of-the-art deblurring and super-resolution results can be achieved simply by integrating the TNRD denoiser in RED.

6.1 Image Deblurring

In order to have a fair comparison to previous work, we follow the synthetic non-blind deblurring experiments conducted in the state-of-the-art work that introduced the NCSR algorithm [9], which combines the self-similarity assumption [1] with the sparsity-inspired model [64]. More specifically, we degrade the test images, supplied by the authors of NCSR, by convolving them with two commonly used point spread functions (PSF); the first is a 9×9 uniform blur, and the second is a 2D Gaussian function with a standard deviation of 1.6. In both cases, an additive Gaussian noise with $\sigma = \sqrt{2}$ is then added to the blurred images. Similarly to NCSR, restoring an RGB image is done by converting it to the YCbCr color-space, applying the deblurring algorithm on the luminance channel only, and then converting the result back to the RGB domain.

Table 1 provides the restoration performance of the three RED schemes – the steepest-descent (SD), the ADMM, and the fixed-point (FP) methods – along with the results of the P^3, the state-of-the-art NCSR and IDD-BM3D [27], and two additional baseline deblurring methods [67, 68]. For brevity, only the steepest-descent scheme is presented when considering the basic median filter as a denoiser. The performance is evaluated using the Peak Signal to Noise Ratio (PSNR) measure, higher is better, computed on the luminance channel of the ground-truth and the estimated image. The parameters of the proposed approach, as well as the ones of the P^3, are tuned to achieve the best performance on this dataset; in the case of the TNRD denoiser, these are depicted in Table 2 and 3, respectively. In the setting of the median filter, which extracts the median value of a 3×3 window, we choose to run the suggested steepest-descent scheme for $N = 400$ iterations with $\lambda = 0.12$ for the uniform PSF, and $N = 200$ with $\lambda = 0.225$ for the Gaussian PSF.

Several remarks are to be made with regard to the obtained results. When the image is

[9] In order to handle an arbitrary noise-level, σ_f, we rely on the following relation $f_{\sigma_f}(\mathbf{y}) = \frac{1}{c} f_5(c \cdot \mathbf{y})$, where $c = 5/\sigma_f$.

Image	Butterfly	Boats	C. Man	House	Parrot	Lena	Barbara	Starfish	Peppers	Leaves	Average
Deblurring: Uniform kernel, $\sigma = \sqrt{2}$											
Total Variation [67]	28.37	29.04	26.82	31.99	29.11	28.33	25.75	27.75	28.43	26.49	28.21
IDD-BM3D [27]	29.21	31.20	28.56	34.44	31.06	29.70	27.98	29.48	29.62	29.38	30.06
ASDS-Reg [68]	28.70	30.80	28.08	34.03	31.22	29.92	27.86	29.72	29.48	28.59	29.84
NCSR	29.68	31.08	28.62	34.31	31.95	29.96	28.10	30.28	29.66	29.98	30.36
P^3-TNRD	29.98	31.21	28.64	34.01	31.70	30.20	27.39	30.07	30.07	29.83	30.31
RED: SD-Median Filter	26.10	28.03	25.57	29.81	28.67	27.29	25.62	27.84	27.40	25.45	27.18
RED: SD-TNRD	30.20	31.20	28.67	33.83	31.62	29.98	27.35	30.47	30.10	29.72	30.31
RED: ADMM-TNRD	30.48	31.05	28.77	33.67	31.88	29.97	27.16	30.61	30.11	30.34	30.40
RED: FP-TNRD	30.41	31.12	28.76	33.76	31.83	30.02	27.27	30.57	30.12	30.13	30.40
Deblurring: Gaussian kernel, $\sigma = \sqrt{2}$											
Total Variation [67]	30.36	29.36	26.81	31.50	31.23	29.47	25.03	29.65	29.42	29.36	29.22
IDD-BM3D [27]	30.73	31.68	28.17	34.08	32.89	31.45	27.19	31.66	29.99	31.4	30.92
ASDS-Reg [68]	29.83	30.27	27.29	31.87	32.93	30.36	27.05	31.91	28.95	30.62	30.11
NCSR [9]	30.84	31.49	28.34	33.63	33.39	31.26	27.91	32.27	30.16	31.57	31.09
P^3-TNRD	31.54	31.74	28.11	34.07	33.37	31.61	27.49	32.53	30.78	32.00	31.32
RED: SD-Median Filter	29.02	30.01	26.45	31.59	31.32	30.00	25.02	30.29	28.53	28.69	29.09
RED: SD-TNRD	31.57	31.53	28.31	33.71	33.19	31.47	26.62	32.46	29.98	31.95	31.08
RED: ADMM-TNRD	31.65	31.56	28.36	33.73	33.33	31.37	26.76	32.49	30.48	31.91	31.16
RED: FP-TNRD	31.66	31.55	28.38	33.74	33.33	31.39	26.76	32.49	30.51	31.93	31.17

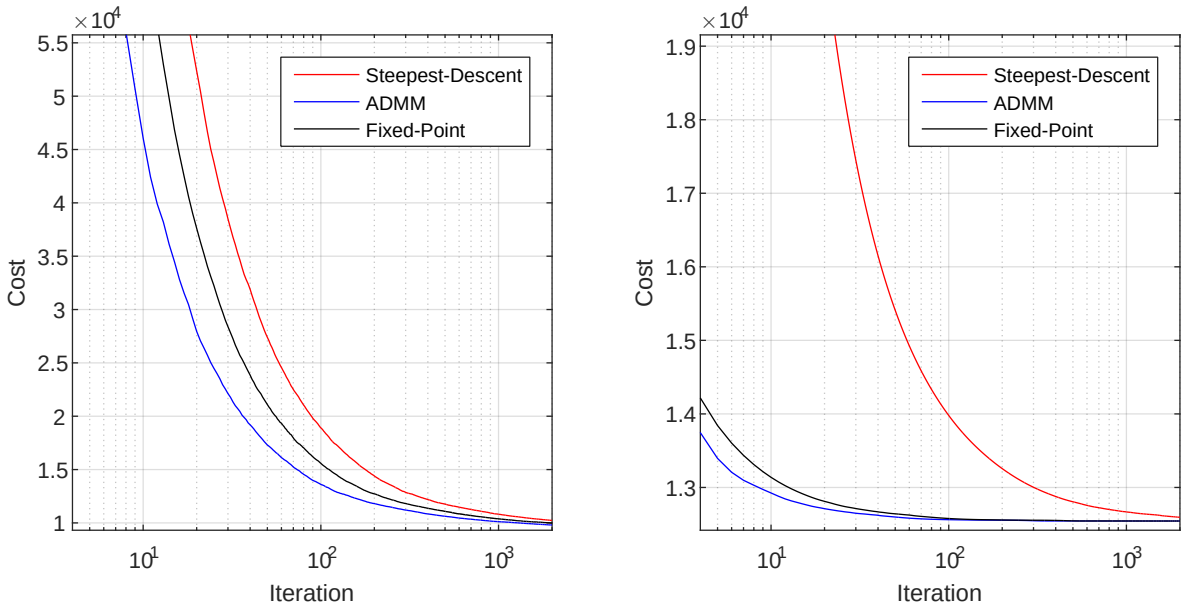
Table 1: Deblurring results measured in PSNR [dB] and evaluated on the set of images provided by the authors of NCSR [9]. The P^3 and RED build upon the TNRD [20] as the denoising engine. We also provide the results obtained by integrating the median filter with the steepest-descent RED scheme.

PSF	Proposed approach: Deblurring			
	Parameter	**Steepest Descent**	**Fixed Point**	**ADMM**
Uniform	N	1500	200	200
	σ_f	3.25	3.25	3.25
	λ	0.02	0.02	0.02
	m_1	–	100	100
	m_2	–	–	3
	β	–	–	0.001
Gaussian	N	1500	200	200
	σ_f	4.1	4.1	4.1
	λ	0.01	0.01	0.01
	m_1	–	100	100
	m_2	–	–	3
	β	–	–	0.0035

Table 2: The set of parameters being used in our framework, leading to the deblurring results reported in Table 1 when plugging the TNRD [20] denoiser.

PSF	P^3: Deblurring	
	Parameter	**Value**
Uniform	N	70
	α	1.0825
	β_0	0.0004
	β_k	$\alpha^k \cdot \beta_0$
	λ	$1024 \cdot \beta_0$
	σ_f	$\sqrt{\lambda/\beta_k}$
Gaussian	N	50
	α	1.105
	β_0	0.0005
	β_k	$\alpha^k \cdot \beta_0$
	λ	$512 \cdot \beta_0$
	σ_f	$\sqrt{\lambda/\beta_k}$

Table 3: The set of parameters being used by the P^3 method, leading to the deblurring results reported in Table 1 when plugging the TNRD [20] denoiser.



(a) Denoising Engine: Median Filter

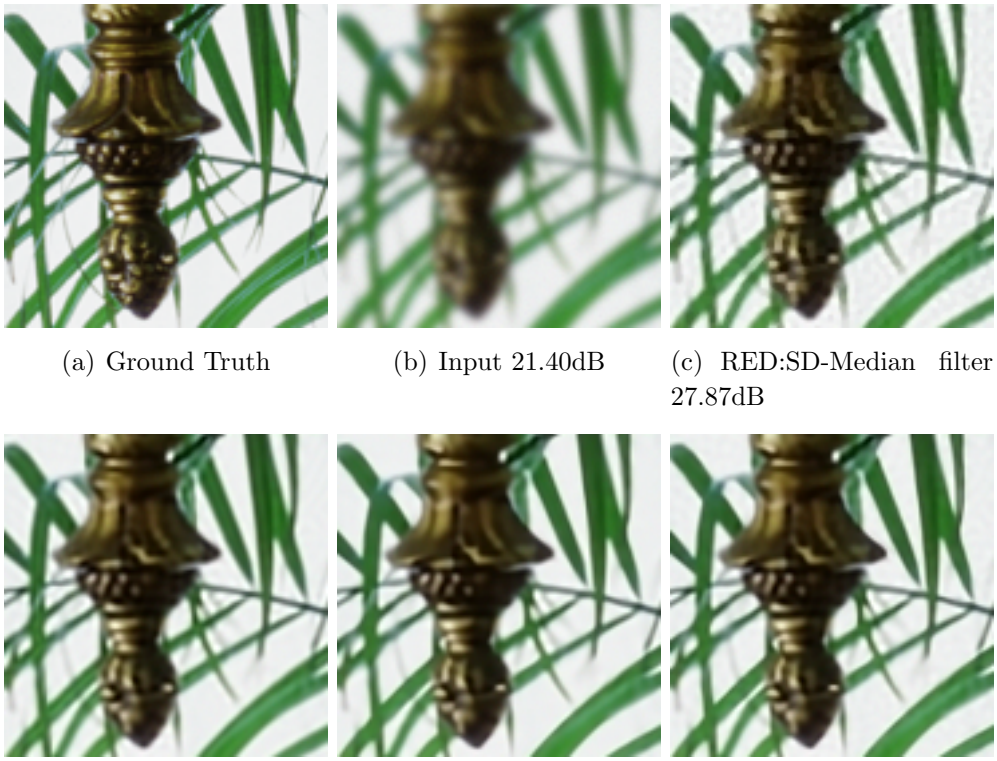
(b) Denoising Engine: TNRD

Figure 4: An illustration of the convergence of the three proposed numerical schemes – the steepest-descent (red), the ADMM (blue) and the fixed-point (black). These are applied on the image Leaves, degraded by a Gaussian PSF, when two denoising engines are tested: (a) the median filter, and (b) TNRD [20].

(a) Ground Truth (b) Input 20.83dB (c) RED: SD-Median filter 25.87dB

(d) NCSR 28.39dB (e) P^3-TNRD 28.20dB (f) RED: FP-TNRD 28.82dB

Figure 5: Visual comparison of deblurring a cropped area from the image Starfish, degraded by a uniform PSF, along with the corresponding PSNR [dB] score.



(a) Ground Truth (b) Input 21.40dB (c) RED:SD-Median filter 27.87dB

(d) NCSR 30.03dB (e) P^3-TNRD 30.35dB (f) RED: ADMM-TNRD 30.36dB

Figure 6: Visual comparison of deblurring a cropped area from the image Leaves, degraded by a Gaussian PSF, along with the corresponding PSNR [dB] score.

degraded by a Gaussian blur kernel, integrating the median filter in the proposed framework leads to a surprising restoration performance that is similar to the total variation deblurring [67]. Furthermore, by choosing the state-of-the-art TNRD to be our denoising engine we achieve results that are competitive with the impressive NCSR and IDD-BM3D methods, which are specifically designed to tackle the deblurring task. Notice that the three versions of the proposed framework obtain a similar PSNR score. However, while the ADMM and the fixed-point variants are of similar complexity, the steepest-descent requires much more steps to converge and thereby more applications of the denoiser. Last, and of most importance, based on the obtained results we conclude that the proposed approach is indeed a superior alternative to the P^3 framework. Specifically, tuning the parameters of the proposed algorithm is significantly simpler than the ones of the P^3; while in the P^3 the parameters should be modified throughout the iterations to achieve convergence to a saddle point (as done by several papers, e.g. [33, 38]), in the our approach these are fixed along the iterations.

An illustration of the convergence of the proposed approach using the three different numerical schemes (steepest-descent, ADMM, and fixed-point) and the two denoisers (median filter, and TNRD) is given in Figure 4. As can be observed, the three algorithms indeed converge, but at different rates; the steepest-descent is the slowest, while the ADMM is the fastest. The fixed-point strategy is slightly slower than the ADMM alternative, but requires only one application of the denoiser per iteration while the ADMM demands $m_2 = 3$ such operations[10].

The above discussion is also supported visually in Figure 5 and 6, comparing the proposed method to the P^3 and the NCSR both for uniform and Gaussian PSF. As can be seen, by plugging the median filter into RED we improve the quality of the blurry image, yet the gap in performance between this simplistic approach and the state-of-the-art is substantial. Once relying on the TNRD, we obtain an efficient deblurring machine that has similar results to the one of the P^3, and both are competitive or even slightly better than the NCSR.

6.2 Image Super-Resolution

Similarly to the previous subsection, we imitate the super-resolution experiments done in [9]. To this end, a low-resolution image is generated by blurring the ground-truth high-resolution one with a 7×7 Gaussian blur kernel with standard deviation 1.6, followed by down-sampling by a factor of 3 in each axis. Next, white Gaussian noise of standard deviation 5 is added to the low-resolution images. The upscaling of an RGB image is done by transforming it first to the YCbCr color-space, super-resolving the luminance channel using the proposed approach (or the baseline methods), while the chroma channels are upscaled by bicubic interpolation. Lastly,

[10] In practice, in the context of the ADMM scheme, we obtained a similar restoration performance for the setting $m_2 = 1$ which is preferred in terms of computations. However, we observed that the cost function does not necessarily reduce between two consecutive iterations, thereby we choose to avoid this configuration.

the outcome is converted back to the RGB color-space.

In terms of PSNR, Table 4 presents the restoration performance of the three variants of the proposed approach in addition to the ones of the P^3, the NCSR and the ASDS-Reg [68] algorithms. Similarly to the deblurring case, the PSNR is computed on the luminance channel only. In the case of the TNRD denoiser, the parameters that are used in our approach and in the P^3 are listed in Table 5 and 6, respectively. In the simpler case of the median filter (defined by a 3×3 window), the number of iterations of the proposed steepest-descent algorithm is set to $N = 50$ with a parameter $\lambda = 0.0325$.

Super-Resolution, scaling = 3, $\sigma = \sqrt{2}$										
Image	Butterfly	flower	Girl	Parth.	Parrot	Raccoon	Bike	Hat	Plants	**Average**
Bicubic	20.74	24.74	29.61	24.04	25.43	26.20	20.75	27.00	27.62	25.13
ASDS-Reg [68]	26.06	27.83	31.87	26.22	29.01	28.01	23.62	29.61	31.18	28.16
NCSR [9]	26.86	28.08	32.03	26.38	29.51	28.03	23.80	29.94	31.73	28.48
P^3-TNRD	26.69	28.25	32.08	26.46	29.57	27.96	23.92	30.20	31.79	28.55
RED: SD-Median Filter	24.44	27.24	31.13	25.80	27.76	27.65	22.89	28.69	30.24	27.32
RED: SD-TNRD	27.37	28.23	32.08	26.54	29.43	27.98	24.04	30.36	31.79	28.65
RED: ADMM-TNRD	27.38	28.23	32.08	26.54	29.45	27.98	24.04	30.37	31.79	28.65
RED: FP-TNRD	27.26	28.24	32.08	26.52	29.42	27.97	23.97	30.35	31.77	28.62

Table 4: Super-resolution results, measured in terms of PSNR [dB] and evaluated on the set of images provided by the authors of NCSR [9]. The P^3 and ours steepest-descent (SD), ADMM, and fixed-point (FP) methods build upon the TNRD [20] as the denoising engine. We also provide the results obtained by integrating the median filter in the steepest-descent scheme.

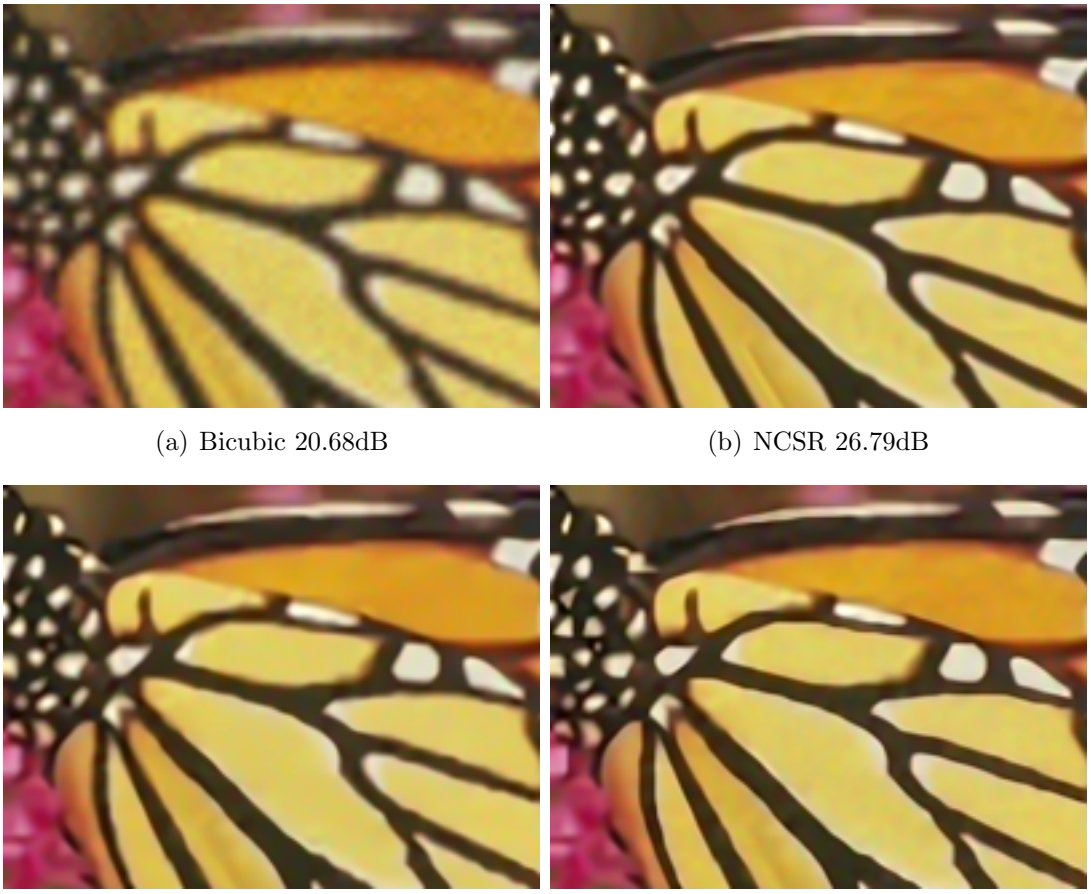
Proposed approach: Super-Resolution			
Parameter	**Steepest Descent**	**Fixed Point**	**ADMM**
N	1500	200	200
σ_f	3	3	3
λ	0.008	0.008	0.008
m_1	–	100	100
m_2	–	–	3
β	–	–	0.0003

Table 5: The set of parameters being used in our framework, leading to the super-resolution results reported in Table 4 when plugging the TNRD [20] denoiser.

P^3: Super-Resolution	
Parameter	**Value**
N	50
α	1.125
β_0	0.0001
β_k	$\alpha^k\ \beta_0$
λ	$320\ \beta_0$
σ_f	$\sqrt{\lambda/\beta_k}$

Table 6: The set of parameters being used in the P^3 method, leading to the super-resolution results reported in Table 4 when plugging the TNRD [20] denoiser.

Interestingly, when setting the median filter to be our denoising engine we get a 2.19dB improvement (on average) over the bicubic interpolation. Alternatively, when choosing a stronger

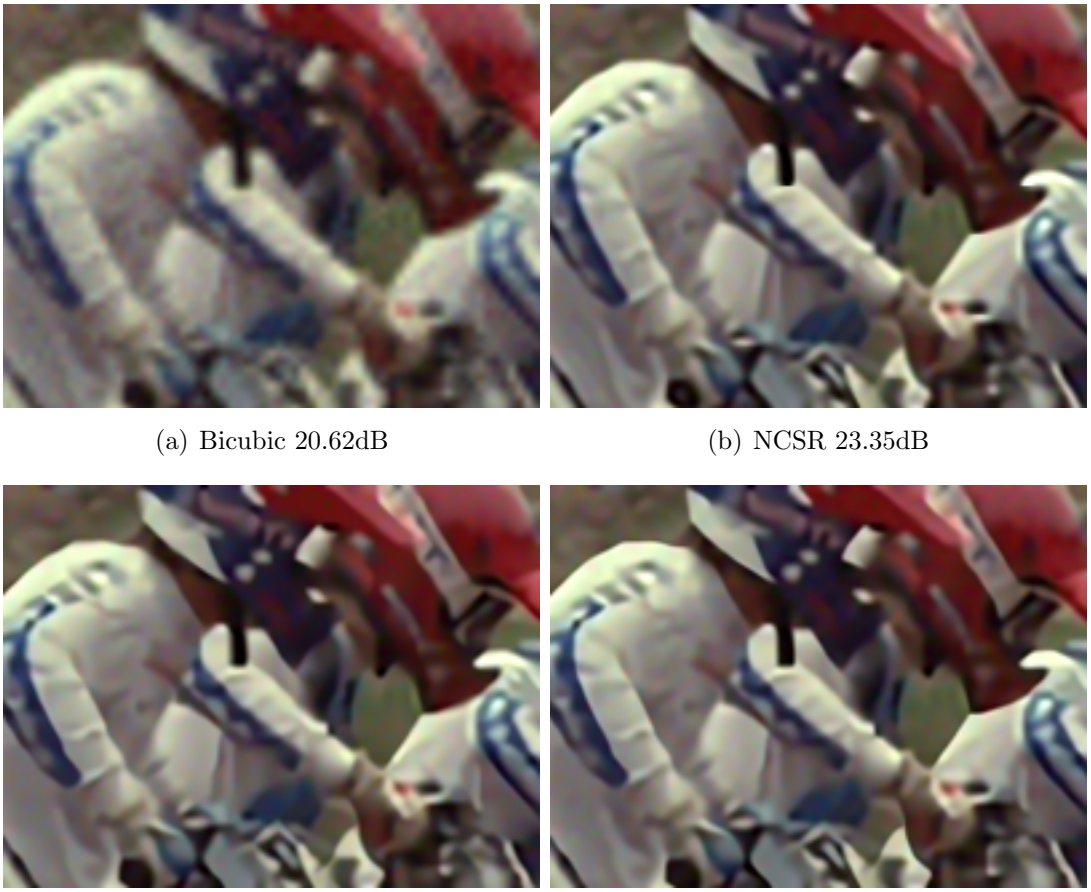


(a) Bicubic 20.68dB (b) NCSR 26.79dB

(c) P^3-TNRD 26.61dB (d) Ours: SD-TNRD 27.39dB

Figure 7: Visual comparison for upscaling by a factor of 3 for the image Butterfly, along with the corresponding PSNR [dB] score.

denoiser such as the TNRD, we achieve state-of-the-art results. Notably, the P^3 and the three variants of the proposed approach lead to similar restoration performance, similar to the observation that was made in the context of the deblurring problem. These support once again the claim that our framework is a tangible alternative to the P^3. Figure 7 and 8 visually compare the proposed method to the P^3 and also to the state-of-the-art NCSR. As shown, the three algorithms offer an impressive restoration with sharp and clear edges, complying with the quantitative results which are given in Table 4.



(a) Bicubic 20.62dB (b) NCSR 23.35dB

(c) P^3-TNRD 23.58dB (d) Ours: ADMM-TNRD 23.66dB

Figure 8: Visual comparison for upscaling by a factor of 3 for the image Bike, along with the corresponding PSNR [dB] score.

7 Conclusions

The idea put forward in this paper is strange – using a denoiser engine within the regularization term in order to handle general inverse problems. A surprising outcome of this proposal is the fact that differentiation of the regularization terms remains trackable, still using the very same denoiser engine and *not* its derivative. This led us to the proposed scheme, termed Regularization by Denoising (RED). We have shown and discussed various appealing properties of this approach, such as convexity and its relation to advanced Laplacian smoothing. We have contrasted this scheme with the plug-and-play-prior method [30], and we have provided experiments that demonstrate the validity of the regularization strategy and the resulting competitive performance.

Is there a wider message in this work? Could it be that one could use more general f functions and still form the proposed regularization? We have seen that all that it takes is the availability of the directional derivative of this function in order to follow all through. Strangely enough, we have shown how even a median filter could fit into this scheme, despite the fact that it does not have any relation to Gaussian denoising. More work is required to investigate the ability to further generalize RED to other and perhaps more daring regularization functionals.

A key question we have left open at this stage is the setting of the parameter σ_f. We chose this to be a fixed value, but we did not address the question of its influence on the overall performance, or whether a varying value strategy could lead to a benefit. More work is required here as well.

A Can We Mimic any Prior?

So far we have shown that denoisers $f(\mathbf{x})$ can be used to define the powerful family of priors $\mathbf{x}^T(\mathbf{x} - f(\mathbf{x}))$ that can regularize a wide variety of inverse problem. Now, let's consider the reverse question: For a given $\rho(x)$, what is the effective $f(\mathbf{x})$ behind it (if any)? More specifically, given $\rho(\mathbf{x})$, we wish to find $f(\mathbf{x})$ such that

$$\frac{1}{2}\mathbf{x}^T(\mathbf{x} - f(\mathbf{x})) = \rho(\mathbf{x}) \tag{45}$$

Recall that one of the key conditions we assumed for the denoiser $f(\mathbf{x})$ is that it is homogenous of degree 1,

$$f(c\mathbf{x}) = cf(\mathbf{x}). \tag{46}$$

This immediately notifies us that the $\rho(\mathbf{x})$ we wish to mimic in (45) must be 2-homogenous because

$$\rho(c\mathbf{x}) = \frac{1}{2}(c\mathbf{x})^T(c\mathbf{x} - f(c\mathbf{x})) = \frac{1}{2}c^2\mathbf{x}^T(\mathbf{x} - f(\mathbf{x})) = c^2\rho(\mathbf{x}) \tag{47}$$

Of course, not all priors satisfy this condition, but the class of priors that do is wide. For instance, while the total-variation function $\mathrm{TV}(\mathbf{x}) = \|\nabla x\|_1$ is only 1-homogeneous, its square $\|\nabla x\|_1^2$ keeps us in business. More generally, since any norm is *by definition* 1-homogenous, all regularizers of the type $\rho(\mathbf{x}) = \|\mathbf{A}x\|_q^2$ for $q = 1, 2, \cdots, \infty$ are 2-homogeneous[11]:

$$\rho(c\mathbf{x}) = \|\mathbf{A}(c\mathbf{x})\|_q^2 = (c\|\mathbf{A}\mathbf{x}\|_q)^2 = c^2\|\mathbf{A}\mathbf{x}\|_q^2 = c^2\rho(\mathbf{x}) \tag{48}$$

Squared norms are not the only functions at our disposal. Beyond norms, any k-homogenous function (such as homogeneous polynomials of degree k) can be made 2-homogeneous by raising to power $2/k$. As well, order-statistics such as maximum, minimum and median are 1-homonegous, and can be squared to the same end.

With the above discussion, we move forward with the assumption that we have a 2-homogeous prior $\rho(\mathbf{x})$ in our hands. Let's recall that the task is to find $f(\mathbf{x})$ so that

$$\frac{1}{2}\mathbf{x}^T(\mathbf{x} - f(\mathbf{x})) = \rho(\mathbf{x}) \tag{49}$$

We proceed by differentiating both sides:

$$\begin{aligned}
\mathbf{x} - \frac{1}{2}\nabla\left(\mathbf{x}^T f(\mathbf{x})\right) &= \nabla\rho(\mathbf{x}) \\
\mathbf{x} - \frac{1}{2}\left(f(\mathbf{x}) + \nabla f(\mathbf{x})\,\mathbf{x}\right) &= \nabla\rho(\mathbf{x}) \\
\mathbf{x} - \frac{1}{2}\left(f(\mathbf{x}) + f(\mathbf{x})\right) &= \nabla\rho(\mathbf{x}) \\
\mathbf{x} - f(\mathbf{x}) &= \nabla\rho(\mathbf{x})
\end{aligned}$$

where in the last step we have invoked the directional derivative expression developed in (14). The solution, $f(\mathbf{x})$, is a denoiser explicitly defined in terms of the prior,

$$f(\mathbf{x}) = \mathbf{x} - \nabla\rho(\mathbf{x}). \tag{50}$$

This is quite a natural result in retrospect – it resembles a steepest descent step. To see this, consider the denoising problem regularized by $\rho(\mathbf{x})$:

$$\underset{\mathbf{x}}{\mathrm{Argmin}}\ \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2 + \rho(\mathbf{x}). \tag{51}$$

One step of steepest descent with step-size of 1 and initial condition $\mathbf{x}_0 = \mathbf{y}$, would read

$$\mathbf{x}_1 = \mathbf{x}_0 - (\mathbf{x}_0 - \mathbf{y} + \nabla\rho(\mathbf{x}_0)) = \mathbf{y} - \nabla\rho(\mathbf{y}) \tag{52}$$

just as shown above as a denoising on $\mathbf{y}$.

[11] The L_0-norm (not really a norm) is an exception that can not be treated this way.

B Kernelizing Priors

We showed in the previous section that a prior with the right properties implies a denoiser beneath; and that this denoiser is directly given by the Jacobian of the prior. Next, let's address a related question: Given a prior $\rho(\mathbf{x})$, does it imply a denoising filter of the pseudo-linear form $f(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$? These types of filters are of course of great interest because they reveal the kind of weighted averaging done by the denoiser. Given $\mathbf{x}$, we can compute the weights $\mathbf{W}(\mathbf{x})$ in one step, and apply them as $\mathbf{W}(\mathbf{x})\mathbf{x}$ to the image in another step. Some of the most popular filters to date, such as bilateral, NLM, and K-SVD, are of this convenient form.

Before we go about finding the hidden filter matrix $\mathbf{W}(\mathbf{x})$, let's illustrate a useful property of the pseudo-linear filters. Take $f(\mathbf{x}) = \mathbf{W}(\mathbf{x})\mathbf{x}$, and again invoke the expression $f(\mathbf{x}) = \nabla f(\mathbf{x})\mathbf{x}$ developed in (14). Substitution gives

$$\mathbf{W}(\mathbf{x})\mathbf{x} = \nabla f(\mathbf{x})\mathbf{x} \qquad \Longrightarrow \qquad (\nabla f(\mathbf{x}) - \mathbf{W}(\mathbf{x}))\mathbf{x} = 0 \tag{53}$$

for all $\mathbf{x}$. From this we conclude that the gradient of pseudo-linear filters is in fact the weight matrix

$$\nabla f(\mathbf{x}) = \mathbf{W}(\mathbf{x}). \tag{54}$$

Now we can go after the weight matrix by taking the analysis from the previous section one step further and computing the second derivative of (45). Starting with the expression (50) that arose from the first derivative, we differentiate again,

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{x} - \nabla\rho(\mathbf{x}) \\ \nabla f(\mathbf{x}) &= \mathbf{I} - \nabla(\nabla\rho(\mathbf{x})) \\ \nabla f(\mathbf{x}) &= \mathbf{I} - \mathbf{H}(\rho(\mathbf{x})). \end{aligned}$$

Replacing $\nabla f(\mathbf{x}) = \mathbf{W}(\mathbf{x})$, we obtain the pleasing result that the weight matrix implied by the prior is the identity matrix minus the Hessian of the prior,

$$\mathbf{W}(\mathbf{x}) = \mathbf{I} - \mathbf{H}(\rho(\mathbf{x})), \tag{55}$$

or posed another way, $\mathbf{L}(\mathbf{x}) = \mathbf{I} - \mathbf{W}(\mathbf{x}) = \mathbf{H}(\rho(\mathbf{x}))$ (i.e. the Laplacian filter is directly given by the Hessian of the prior, which is not surprising, bearing in mind that we seek a relation of the form $\rho(\mathbf{x}) = \mathbf{x}^T\mathbf{L}(\mathbf{x})\mathbf{x}$). What we have done here is to "kernelize" the regularizer and find an explicit expression for the weights of the implied filter. Several observations about this result are in order:

- **Convexity:** If $\rho(\mathbf{x})$ is convex, then its Hessian is symmetric positive semi-definite (PSD), and therefore $\mathbf{L}(\mathbf{x})$ is PSD. Furthermore, if $\|\mathbf{L}(\mathbf{x})\| \leq 1$, we get that (i) $\mathbf{W}(\mathbf{x})$ has spectral radius equal to 1, implying that strong passivity condition (Condition 4) is guaranteed; and (ii) $\mathbf{W}(\mathbf{x}) = \mathbf{I} - \mathbf{L}(\mathbf{x})$ is also PSD – a desirable property [19].

- **Homogeneity:** Since $\rho(\mathbf{x})$ is 2-homogenous, its gradient is 1-homogeneous. We can show this by differentiating:

$$\begin{aligned}
\rho(c\mathbf{x}) &= c^2\rho(\mathbf{x}) \\
\nabla_{\mathbf{x}}\rho(c\mathbf{x}) &= c^2\nabla_{\mathbf{x}}\rho(\mathbf{x}) \\
\frac{\partial c\mathbf{x}}{\partial \mathbf{x}}\nabla_{c\mathbf{x}}\rho(c\mathbf{x}) &= c^2\nabla_{\mathbf{x}}\rho(\mathbf{x}) \\
c\nabla_{c\mathbf{x}}\rho(c\mathbf{x}) &= c^2\nabla_{\mathbf{x}}\rho(\mathbf{x}) \\
\nabla_{c\mathbf{x}}\rho(c\mathbf{x}) &= c\nabla_{\mathbf{x}}\rho(\mathbf{x})
\end{aligned}$$

  Similarly the Hessian $\mathbf{H}(\rho(\mathbf{x}))$ is *invariant*[12] to scaling of the image $\mathbf{x}$, and so is the implied filter matrix $\mathbf{W}(\mathbf{x})$. Consequently, the applied filter $\mathbf{W}(\mathbf{x})\mathbf{x}$ is 1-homogenous, which again is consistent with our earlier conditions.

- **Row Stochasticness:** Let's recall the expression we derived above for the filter in terms of the Hessian of the prior,

$$\mathbf{W}(\mathbf{x}) = \mathbf{I} - \mathbf{H}(\rho(\mathbf{x})). \tag{56}$$

  If a filter defined this way is to be row-stochastic, we would have (for every $\mathbf{x}$), $\mathbf{W}(\mathbf{x})\mathbb{1} = (\mathbf{I} - \mathbf{H}(\rho(\mathbf{x})))\,\mathbb{1} = \mathbb{1}$, or equivalently,

$$\mathbf{H}(\rho(\mathbf{x}))\mathbb{1} = \mathbb{0}. \tag{57}$$

  This relation does not hold in general. However, consider defining the prior in terms of the gradient of the image instead of the image itself. Namely, this involves a change of variables in the prior $\rho(\mathbf{x})$ from $\mathbf{x}$ to $\mathbf{D}\mathbf{x}$, where $\mathbf{D}$ is the gradient (e.g. difference) operator. For instance, instead of $\rho(\mathbf{x}) = \|\mathbf{x}\|_1^2$, consider $\rho_{\mathbf{D}}(\mathbf{x}) = \|\mathbf{D}\mathbf{x}\|_1^2$. The Hessian of the prior under this linear transformation is given by the chain rule as

$$\mathbf{H}(\rho_{\mathbf{D}}(\mathbf{x})) = \mathbf{D}^T\mathbf{H}(\rho(\mathbf{x}))\mathbf{D}. \tag{58}$$

  This Hessian, when applied to the constant vector $\mathbb{1}$ will vanish for all $\mathbf{x}$ since $\mathbf{D}\mathbb{1} = \mathbb{0}$:

$$\mathbf{D}^T\mathbf{H}(\rho(\mathbf{x}))\mathbf{D}\,\mathbb{1} = \mathbb{0}, \tag{59}$$

  so the filter resulting from this Hessian is row-stochastic. In fact, the same can be said for column-stochasticness, since $\mathbb{1}^T\mathbf{D}^T = 0$.

[12] Or 0-homogenous

C More on Homogeneity

The concept of homogeneity of a filter played an important role in the development of the key results of the paper. So it is worth saying a bit more about it and to question whether this condition is satisfied for some popular and familiar filters. A general construction of a denoising filter could be based on a symmetric positive semi-definite kernel $\mathbf{K}_{i,j}(\mathbf{x}) = \mathbf{K}(x_i, x_j) \geq 0$ from which the filter matrix $\mathbf{W}(\mathbf{x})$ is constructed by normalization. More specifically,

$$\mathbf{W}_{i,j}(\mathbf{x}) = \frac{\mathbf{K}_{i,j}(\mathbf{x})}{\sum_{i=1}^{n} \mathbf{K}_{i,j}(\mathbf{x})}. \tag{60}$$

Whether such a denoiser is homogenous very much depends on the choice of the kernel $\mathbf{K}$. For instance, if the kernel is homogeneous of *any* degree p, then the resulting filter matrix is invariant to scaling through cancellation,

$$\mathbf{W}_{i,j}(c\mathbf{x}) = \frac{\mathbf{K}_{i,j}(c\mathbf{x})}{\sum_{i=1}^{n} \mathbf{K}_{i,j}(c\mathbf{x})} = \frac{c^p \mathbf{K}_{i,j}(\mathbf{x})}{\sum_{i=1}^{n} c^p \mathbf{K}_{i,j}(\mathbf{x})} = \mathbf{W}_{i,j}(\mathbf{x}). \tag{61}$$

Examples of homogeneous kernels include (homogeneous) polynomials, and several others [45] which are not in common use in image processing. Most commonly used kernels are the exponentials (Gaussian to be exact), which are used in the bilateral or non-local means (NLM) cases. The Gaussian function is not homogeneous, but as we will show below, the resulting pseudo-linear filter are nearly so. We will show that for $c = 1 + \epsilon$ with very small ϵ we have 1-homogeneity for the NLM-type filters, namely:

$$f(c\mathbf{x}) = \mathbf{W}(c\mathbf{x})(c\mathbf{x}) = c\mathbf{W}(\mathbf{x})\,\mathbf{x} = cf(\mathbf{x}). \tag{62}$$

The i, j-th element of the filter weight matrix for the NLM filter[13] is

$$\mathbf{W}_{i,j}(\mathbf{x}; \sigma) = \frac{e_{ij}(\sigma)}{d_j(\sigma)} \tag{63}$$

where

$$\begin{aligned} e_{ij}(\sigma) &= \exp(-\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2/2\sigma^2), \\ d_j(\sigma) &= \sum_{i=1}^{n} \exp(-\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2/2\sigma^2). \end{aligned}$$

where $\mathbf{R}_i\mathbf{x}$ is a patch centred at pixel position i, extracted from the image; the normalization constant $d_j(\sigma)$ is given by summing across the rows of the kernel matrix.

[13]The bilateral filter, which includes a spatial distance weight is similarly treatable, since the spatial weights are invariant to scaling of the image in any case.

Do these weights change much when the image $\mathbf{x}$ is replaced by a scaled version $c\mathbf{x}$? First, we note that the scaling in $\mathbf{x}$ can be absorbed in the parameter σ as follows:

$$\exp(-\|c\mathbf{R}_i\mathbf{x} - c\mathbf{R}_j\mathbf{x}\|^2/2\sigma^2) = \exp(-\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2/2(\sigma/c)^2). \tag{64}$$

Second, the effect of this (multiplicative) scaling can be approximated by an additive perturbation,

$$\frac{\sigma}{c} = \frac{\sigma}{1+\epsilon} \approx \sigma - \epsilon\sigma = \sigma + \delta. \tag{65}$$

We now compute an approximation to $\mathbf{W}_{i,j}(\mathbf{x}, \sigma + \delta)$ using a Taylor series:

$$\begin{aligned}
\mathbf{W}_{i,j}((1+\epsilon)\mathbf{x}; \sigma) &\approx \mathbf{W}_{i,j}(\mathbf{x}; \sigma + \delta) \\
&\approx \mathbf{W}_{i,j}(\mathbf{x}; \sigma) + \delta \, \frac{\partial \mathbf{W}_{i,j}(\mathbf{x}; \sigma)}{\partial \sigma}.
\end{aligned}$$

The derivative of the weight values will be calculated in terms of the functions $e_i(\sigma)$ and $d_j(\sigma)$ as follows:

$$\begin{aligned}
\frac{\partial \mathbf{W}_{i,j}(\mathbf{x}; \sigma)}{\partial \sigma} &= \frac{\partial}{\partial \sigma}\left(\frac{e_i(\sigma)}{d_j(\sigma)}\right) \\
&= \frac{e_i'(\sigma)d_j(\sigma) - e_i(\sigma)d_j'(\sigma)}{d_j^2(\sigma)} \\
&= \frac{e_i'(\sigma)}{d_j(\sigma)} - \frac{e_i(\sigma)}{d_j(\sigma)}\frac{d_j'(\sigma)}{d_j(\sigma)} \\
&= \frac{\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2}{\sigma^3}\frac{e_i(\sigma)}{d_j(\sigma)} - \frac{e_i(\sigma)}{d_j(\sigma)}\frac{d_j'(\sigma)}{d_j(\sigma)} \\
&= \frac{\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2}{\sigma^3}\mathbf{W}_{i,j}(\mathbf{x}; \sigma) - \frac{d_j'(\sigma)}{d_j(\sigma)}\mathbf{W}_{i,j}(\mathbf{x}; \sigma) \\
&= \left(\frac{\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2}{\sigma^3} - \frac{d_j'(\sigma)}{d_j(\sigma)}\right)\mathbf{W}_{i,j}(\mathbf{x}; \sigma).
\end{aligned}$$

Therefore,

$$\mathbf{W}_{i,j}(\mathbf{x}; \sigma + \delta) \approx \left[1 + \delta \, \left(\frac{\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2}{\sigma^3} - \frac{d_j'(\sigma)}{d_j(\sigma)}\right)\right]\mathbf{W}_{i,j}(\mathbf{x}; \sigma). \tag{66}$$

Replacing $\delta = -\epsilon\sigma$, we obtain

$$\begin{aligned}
\mathbf{W}_{i,j}((1+\epsilon)\mathbf{x}; \sigma) &\approx \left[1 - \epsilon \, \left(\frac{\|\mathbf{R}_i\mathbf{x} - \mathbf{R}_j\mathbf{x}\|^2}{\sigma^2} - \sigma\frac{d_j'(\sigma)}{d_j(\sigma)}\right)\right]\mathbf{W}_{i,j}(\mathbf{x}; \sigma) \\
&= (1 - \epsilon \, \phi(\sigma))\, \mathbf{W}_{i,j}(\mathbf{x}; \sigma).
\end{aligned}$$

We can simplify further, but this is not necessary since $\phi(\sigma)$ does not depend on ϵ. What we have shown is that the filter weights change very little as a result of the scaling $(1+\epsilon)\mathbf{x}$ as long as ϵ is very small. Therefore the NLM (and bilateral) filters are (almost exactly) 1-homogeneous as we had hoped.

References

[1] A. Buades, B. Coll, and J.M. Morel, A Non-Local Algorithm for Image Denoising, *CVPR*, 2005.

[2] C. Kervrann and J. Boulanger, Optimal Spatial Adaptation for Patch-Based Image Denoising, *IEEE Trans. on Image Processing*, Vol. 15, No. 10, pp. 2866–2878, October 2006.

[3] T. Tasdizen, Principal Neighborhood Dictionaries for Nonlocal Means Image Denoising, *IEEE Trans. on Image Processing*, Vol. 18, No. 12, pp. 2649–2660, 2009.

[4] M. Elad and M. Aharon, Image Denoising via Sparse and Redundant Representations Over Learned Dictionaries, *IEEE Trans. on Image Processing*, Vol. 15, No. 12, pp. 3736–3745, 2006.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, Image Denoising by Sparse 3-D Transform-Domain Collaborative Filtering, *, IEEE Trans. on Image Processing*, Vol. 16, No. 8, pp. 2080–2095, 2007.

[6] H. Takeda, S. Farsiu, and P. Milanfar, Kernel Regression for Image Processing and Reconstruction, *IEEE Trans. on image processing*, Vol. 16, No. 2, 349–366, 2007.

[7] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, Non-Local Sparse Models for Image Restoration, ICCV, 2009.

[8] W. Dong, X. Li, L. Zhang, and G. Shi, Sparsity-Based Image Denoising via Dictionary Learning and Structural Clustering, *CVPR* 2011.

[9] W. Dong, L. Zhang, G. Shi, and X. Li, Nonlocally Centralized Sparse Representation for Image Restoration, *IEEE Trans. on Image Processing*, Vol. 22, No. 4, pp. 1620–1630, 2013.

[10] P. Chatterjee and P. Milanfar, Patch-Based Near-Optimal Image Denoising, *IEEE Trans. on Image Processing*, Vol. 21, No. 4, pp. 1635–1649, 2012.

[11] H.C. Burger, C.J. Schuler, and S. Harmeling, Image denoising: Can Plain Neural Networks Compete With BM3D?, *CVPR* 2012.

[12] M. Lebrun, M. Colom, A. Buades, and J.-M. Morel, Secrets of Image Denoising Cuisine, *Acta Numerica*, Vol. 21, No. 4, pp. 475–576, 2012.

[13] M. Lebrun, A. Buades, and J.-M. Morel, Implementation of the "Non- Local Bayes" (NL-Bayes) Image Denoising Algorithm, *IPOL*, Vol. 3, pp, 1–42, 2013.

[14] C. Knaus and M. Zwicker, Dual-Domain Image Denoising, *ICIP*, 2013.

[15] H. Talebi and P. Milanfar, Global image denoising, *IEEE Transactions on Image Processing*, Vol. 23, No. 2, pp. 755–768, 2014.

[16] S. Gu, L. Zhang, W. Zuo, and X.Feng, Weighted Nuclear Norm Minimization with Application to Image Denoising, *CVPR* 2014.

[17] W. Dong, G. Shi, Y. Ma, and X. Li, Image Restoration via Simultaneous Sparse Coding: Where Structured Sparsity Meets Gaussian Scale Mixture, *Int. Journal of Computer Vision (IJCV)*, Vol. 114, No. 2, pp. 217–232, 2015.

[18] N. Pierazzo, M. Rais, J.-M. Morel, and G. Facciolo, DA3D: Fast and Data Adaptive Dual Domain Denoising, *ICIP*, 2015.

[19] P. Milanfar, Symmetrizing Smoothing Filters, *SIAM Journal on Imaging Science*, Vol. 6, No. 1, pp. 263284, 2013.

[20] Y. Chen and T. Pock, Trainable Nonlinear Reaction Diffusion: A Flexible Framework for Fast and Effective Image Restoration, *arXiv*, 1508.02848, 2015.

[21] P. Chatterjee and P. Milanfar, Is Denoising Dead? *IEEE Trans. on Image Processing*, Vol. 19, No. 4, pp. 895–911, 2010.

[22] A. Levin and B. Nadler, Natural Image Denoising: Optimality and Inherent Bounds, *CVPR*, 2011.

[23] A. Levin, B. Nadler, F. Durand, and W.T. Freeman, Patch Complexity, Finite Pixel Correlations and Optimal Denoising, *ECCV*, 2012.

[24] M. Protter, M. Elad, H. Takeda, and P. Milanfar, Generalizing the Nonlocal-Means to Superresolution Reconstruction, *IEEE Trans. Image Processing*, Vol. 18, No. 1, pp. 36–51, 2009.

[25] S. Ravishankar and Y. Bresler, MR Image Reconstruction From Highly Undersampled k-Space Data by Dictionary Learning, *IEEE Trans. on Medical Imaging*, Vol. 30, No. 5, pp. 1028–1041, 2011.

[26] M. Akcakaya, T.A. Basha, B. Goddu, L.A. Goepfert, K.V. Kissinger, V. Tarokh, W.J. Manning, and R. Nezafat, Low-Dimensional Structure Self-Learning and Thresholding: Regularization Beyond Compressed Sensing for MRI Reconstruction, *Magnetic Resonance in Medicine*, Vol. 66, No. 3, pp. 756–767, 2011.

[27] A. Danielyan, V. Katkovnik, and K. Egiazarian, BM3D Frames and Variational Image Deblurring, *IEEE Trans. on Image Processing*, Vol. 21, No. 4, pp. 1715–1728, 2012.

[28] W. Dong, L. Zhang, R. Lukac, and G. Shi, Sparse Representation based Image Interpolation with Nonlocal Autoregressive Modeling, *IEEE Trans. on Image Processing*, Vol. 22, No. 4, pp. 1382–1394, 2013.

[29] C.A. Metzler, A. Maleki, and R.G. Baraniuk, Optimal Recovery From Compressive Measurements via Denoising-Based Approximate Message Passing, *SAMPTA*, 2015.

[30] S.V. Venkatakrishnan, C.A. Bouman, and B. Wohlberg, Plug-and-Play Priors for Model Based Reconstruction, *GlobalSIP*, 2013.

[31] S. Sreehari, S.V. Venkatakrishnan, B. Wohlberg, L.F. Drummy, J.P. Simmons, and C.A. Bouman Plug-and-Play Priors for Bright Field Electron Tomography and Sparse Interpolation, *arXiv*, 1512.07331, 2015.

[32] S.H. Chan, Algorithm-Induced Prior for Image Restoration, *ArXiv*, 1602.00715, 2016.

[33] S.H. Chan, X. Wang, and O.A. Elgendy, Plug-and-Play ADMM for Image Restoration: Fixed Point Convergence and Applications, *ArXiv*, 1605.01710, 2016.

[34] S. Sreehari, S.V. Venkatakrishnan, B. Wohlberg, L.F. Drummy, J.P. Simmons, and C.A. Bouman, Plug-and-Play Priors for Bright Field Electron Tomography and Sparse Interpolation, *ArXiv*, 1512.07331, 2015.

[35] Y. Dar, A.M. Bruckstein, M. Elad, and R. Giryes, Postprocessing of Compressed Images via Sequential Denoising, *IEEE Trans. on Image Processing*, Vol. 25, No. 7, pp. 3044 - 3058, 2016.

[36] M.U. Sadiq , J.P. Simmons, and C.A. Bouman, Model Based Image Reconstruction with Physics Based Priors, *ICIP*, 2016.

[37] A.M. Teodoro, J.M. Bioucas-Dias, and M.A.T. Figueiredo, Image Restoration with Locally Selected Class-Adapted Models, *ICIP*, 2016.

[38] A. Brifman, Y. Romano, and M. Elad, Turning a Denoiser into a Super-Resolver Using Plug and Play Priors, *ICIP*, 2016.

[39] R.L. Lagendijk and J. Biemond, Iterative Identification and Restoration of Images, *Springer*, 1990.

[40] L. I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, Phys. D, vol. 60, no. 1-4, pp. 259268, Nov. 1992.

[41] S Mallat, A Wavelet Tour of Signal Processing, *Academic Press*, 1999.

[42] D. Zoran and Y. Weiss, From Learning Models of Natural Image Patches to Whole Image Restoration, *CVPR* 2011.

[43] G. Yu, G. Sapiro, and S. Mallat, Solving Inverse Problems with Piecewise Linear Estimators: From Gaussian Mixture Models to Structured Sparsity, *IEEE Trans. Image Processing*, Vol. 21, No. 5, pp. 2481–2499, 2012.

[44] A.M. Bruckstein, D.L. Donoho, and M. Elad, From Sparse Solutions of Systems of Equations to Sparse Modeling of Signals and Images, *SIAM Review*, Vol. 51, No. 1, pp. 34–81, 2009.

[45] A. Vedaldi and A. Zisserman, Efficient Additive Kernels via Explicit Feature Maps, IEEE Trans Pattern Anal Mach Intell. 2012 Mar vo. 34, no. 3, pp. 480-92

[46] A. Elmoataz, O. Lezoray, and S. Bougleux, Nonlocal Discrete Regularization on Weighted Graphs: A Framework for Image and Manifold Processing, *IEEE Trans. Image Processing*, Vol. 17, No. 7, pp. 1047–1060, 2008.

[47] A.D. Szlam, M. Maggioni, and R.R. Coifman, Regularization on Graphs with Function-Adapted Diffusion Processes, *Journal on Machine Learning Research*, Vol. 9, pp. 1711–1739, 2008.

[48] S. Bougleux, A. Elmoataz, and M. Melkemi, Local and Nonlocal Discrete Regularization on Weighted Graphs for Image and Mesh Processing, *Int. Journal on Computer Vision*, Vol. 84, pp. 220–236, 2009.

[49] G. Peyre, S. Bougleux, and L.D. Cohen, Non-local Regularization of Inverse Problems, *Inverse Problems and Imaging*, Vol. 5, No. 2, pp. 511–530, 2011.

[50] P. Milanfar, A Tour of Modern Image Filtering: New Insights and Methods, both Practical and Theoretical, *IEEE Signal Processing Magazine*, Vol. 30, No. 1, pp. 106–128, 2013.

[51] F.G. Meyer and X. Shen, Perturbation of the Eigenvectors of the Graph Laplacian: Application to Image Denoising, *Applied Computational Harmonic Analysis*, Vol. 36, No. 2, pp. 326–334, 2014.

[52] X. Liu, D. Zhai, D. Zhao, G. Zhai, and W. Gao, Progressive Image Denoising Through Hybrid Graph Laplacian Regularization: A Unified Framework, *IEEE Trans. Image Processing*, Vol. 23, No. 4, pp. 1491–1503, 2014.

[53] S.M. Haque, G. Pai, and V.M. Govindu, Symmetric Smoothing Filters From Global Consistency Constraints, *IEEE Trans. Image Processing*, Vol. 24, No. 5, pp. 1536–1548, 2014.

[54] A. Kheradmand and P. Milanfar, A General Framework for Regularized, Similarity-Based Image Restoration, *IEEE Trans. Image Processing*, Vol. 23, No. 12, pp. 5136–5151, 2014.

[55] Y. Romano and M. Elad, Boosting of Image Denoising Algorithms, *SIAM Journal on Imaging Sciences*, Vol. 8, No. 2, pp 1187–1219, June 2015.

[56] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers, *Foundations and Trends in Machine Learning*, Vol. 3, No. 1, July 2011.

[57] M. Afonso, J. Bioucas-Dias, and M.A.T. Figueiredo, Fast Image Recovery Using Variable Splitting and Constrained Optimization, *IEEE Trans. on Image Processing*, Vol. 19, No. 9, pp. 2345–2356, 2010.

[58] M. Afonso, J. Bioucas-Dias, and M.A.T. Figueiredo, An Augmented Lagrangian Approach to the Constrained Optimization Formulation of Imaging Inverse Problems, *IEEE Trans on Image Processing*, Vol. 20, No. 3, pp. 681–695, 2011.

[59] E. Candes and T. Tao, The Dantzig Selector: Statistical Estimation When p is Much Larger than n, *Annals of Statistics*, Vol. 35, No. 6, pp. 2313–2351, 2007.

[60] Y. Romano and M. Elad, Improving K-SVD Denoising by Post-Processing its Method-Noise, *ICIP*, 2013.

[61] H. Talebi, X. Zhu, and P. Milanfar, How to SAIF-ly Boost Denoising Performance, *IEEE Trans. on Image Processing*, Vol. 22, No. 4, pp. 1470–1485, 2013.

[62] T.M. Apostol, CALCULUS, 2-nd Edition, Volume 1, *Waltham*, Massachusetts – Blaisdell, 1967.

[63] S. Boyd and L. Vandenberghe, Convex Optimization, *Cambridge University Press*, 2004.

[64] M. Elad, Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing, *Springer*, 1st Edition, 2010.

[65] M. Elad, B. Matalon, and M. Zibulevsky, Coordinate and Subspace Optimization Methods for Linear Least Squares with Non-Quadratic Regularization, *Applied and Computational Harmonic Analysis*, Vol. 23, No. 3, pp. 346–367, 2007.

[66] M. Droske and A. Bertozzi, Higher-Order Feature-Preserving Geometric Regularization, *SIAM Journal on Imaging Sciences*, Vol. 3, No. 1, pp. 21–51, 2010.

[67] A. Beck and M. Teboulle, Fast Gradient-Based Algorithms for Constrained Total Variation Image Denoising and Deblurring Problems, *IEEE Transactions on Image Processing*, Vol. 18, No. 11, pp. 2419–2434, 2009.

[68] W. Dong, L. Zhang, G. Shi, and X. Wu, Image Deblurring and Super-Resolution by Adaptive Sparse Domain Selection and Adaptive Regularization, *IEEE Transactions on Image Processing*, Vol. 20, No. 7, pp. 1838–1857, 2011.