Regularization by Denoising: Clarifications and New Interpretations

Edward T. Reehorst and Philip Schniter, *Fellow, IEEE*

***Abstract*—Regularization by denoising (RED), as recently proposed by Romano, Elad, and Milanfar, is powerful image-recovery framework that aims to minimize an explicit regularization objective constructed from a plug-in image-denoising function. Experimental evidence suggests that the RED algorithms are a state of the art. We claim, however, that explicit regularization does not explain the RED algorithms. In particular, we show that many of the expressions in the paper by Romano *et al.* hold only when the denoiser has a symmetric Jacobian, and we demonstrate that such symmetry does not occur with practical denoisers such as nonlocal means, BM3D, TNRD, and DnCNN. To explain the RED algorithms, we propose a new framework called Score-Matching by Denoising (SMD), which aims to match a "score" (i.e., the gradient of a log-prior). We then show tight connections between SMD, kernel density estimation, and constrained minimum mean-squared error denoising. Furthermore, we interpret the RED algorithms from Romano *et al.* and propose new algorithms with acceleration and convergence guarantees. Finally, we show that the RED algorithms seek a consensus equilibrium solution, which facilitates a comparison to plug-and-play ADMM.**

***Index Terms*—Equilibrium methods, image denoising, image reconstruction, kernel density estimation, score matching.**

I. Introduction

CONSIDER the problem of recovering a (vectorized) image $\boldsymbol{x}^0 \in \mathbb{R}^N$ from noisy linear measurements $\boldsymbol{y} \in \mathbb{R}^M$ of the form

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}^0 + \boldsymbol{e}, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ is a known linear transformation and $\boldsymbol{e}$ is noise. This problem is of great importance in many applications and has been studied for several decades.

One of the most popular approaches to image recovery is the "variational" approach, where one poses and solves an optimization problem of the form

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \big\{\ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho(\boldsymbol{x})\big\}. \tag{2}$$

In (2), $\ell(\boldsymbol{x};\boldsymbol{y})$ is a loss function that penalizes mismatch to the measurements, $\rho(\boldsymbol{x})$ is a regularization term that penalizes mismatch to the image class of interest, and $\lambda > 0$ is a design parameter that trades between loss and regularization. A prime advantage of the variational approach is that, in many cases, efficient optimization methods can be readily applied to (2).

A key question is: How should one choose the loss $\ell(\cdot;\boldsymbol{y})$ and regularization $\rho(\cdot)$ in (2)? As discussed in the sequel, the MAP-Bayesian interpretation suggests that they should be chosen in proportion to the negative log-likelihood and negative log-prior, respectively. The trouble is that accurate prior models for images are lacking.

Recently, a breakthrough was made by Romano, Elad, and Milanfar in [1]. Leveraging the long history (e.g., [2], [3]) and recent advances (e.g., [4], [5]) in image denoising algorithms, they proposed the *regularization by denoising* (RED) framework, where an explicit regularizer $\rho(\boldsymbol{x})$ is constructed from an image denoiser $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ using the simple and elegant rule

$$\rho_{\mathsf{red}}(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\top\big(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})\big). \tag{3}$$

Based on this framework, they proposed several recovery algorithms (based on steepest descent, ADMM, and fixed-point methods, respectively) that yield state-of-the-art performance in deblurring and super-resolution tasks.

In this paper, we provide some clarifications and new interpretations of the excellent RED algorithms from [1]. Our work was motivated by an interesting empirical observation: With many practical denoisers $\boldsymbol{f}(\cdot)$, the RED algorithms do not minimize the RED variational objective "$\ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho_{\mathsf{red}}(\boldsymbol{x})$." As we establish in the sequel, the RED regularization (3) is justified only for denoisers with symmetric Jacobians, which unfortunately does not cover many state-of-the-art methods such as non-local means (NLM) [6], BM3D [7], TNRD [4], and DnCNN [5]. In fact, we are able to establish a stronger result: For non-symmetric denoisers, there exists no regularization $\rho(\cdot)$ that explains the RED algorithms from [1].

In light of these (negative) results, there remains the question of how to explain/understand the RED algorithms from [1] when used with non-symmetric denoisers. In response, we propose a framework called *score-matching by denoising* (SMD), which aims to match the "score" (i.e., the gradient of the log-prior) rather than to design any explicit regularizer. We then show tight connections between SMD, kernel density estimation [8], and constrained minimum mean-squared error (MMSE) denoising. In addition, we provide new interpretations of the RED-ADMM and RED-FP algorithms proposed in [1], and we propose novel

Manuscript received July 17, 2018; revised September 25, 2018; accepted October 26, 2018. Date of publication November 9, 2018; date of current version February 7, 2019. This work was supported in part by the National Science Foundation under Grants CCF-1527162 and CCF-1716388 and in part by the National Institutes of Health under Grant R01HL135489. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Stanley Ho Chan. *(Corresponding author: Philip Schniter.)*

The authors are with the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210 USA (e-mail: reehorst.3@osu.edu; schniter.1@osu.edu).

Digital Object Identifier 10.1109/TCI.2018.2880326

RED algorithms with faster convergence. Inspired by [9], we show that the RED algorithms seek to satisfy a consensus equilibrium condition that allows a direct comparison to the plug-and-play ADMM algorithms from [10].

The remainder of the paper is organized as follows. In Section II we provide more background on RED and related algorithms such as plug-and-play ADMM [10]. In Section III, we discuss the impact of Jacobian symmetry on RED and test whether this property holds in practice. In Section IV, we propose the SMD framework. In Section V, we present new interpretations of the RED algorithms from [1] and new algorithms based on accelerated proximal gradient methods. In Section VI, we perform an equilibrium analysis of the RED algorithms, and, in Section VII, we conclude.

II. Background

A. The MAP-Bayesian Interpretation

For use in the sequel, we briefly discuss the Bayesian maximum a posteriori (MAP) estimation framework [11]. The MAP estimate of $\boldsymbol{x}$ from $\boldsymbol{y}$ is defined as

$$\widehat{\boldsymbol{x}}_{\mathsf{map}} = \arg\max_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}), \tag{4}$$

where $p(\boldsymbol{x}|\boldsymbol{y})$ denotes the probability density of $\boldsymbol{x}$ given $\boldsymbol{y}$. Notice that, from Bayes rule $p(\boldsymbol{x}|\boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x})/p(\boldsymbol{y})$ and the monotonically increasing nature of $\ln(\cdot)$, we can write

$$\widehat{\boldsymbol{x}}_{\mathsf{map}} = \arg\min_{\boldsymbol{x}} \big\{ -\ln p(\boldsymbol{y}|\boldsymbol{x}) - \ln p(\boldsymbol{x}) \big\}. \tag{5}$$

MAP estimation (5) has a direct connection to variational optimization (2): the log-likelihood term $-\ln p(\boldsymbol{y}|\boldsymbol{x})$ corresponds to the loss $\ell(\boldsymbol{x};\boldsymbol{y})$ and the log-prior term $-\ln p(\boldsymbol{x})$ corresponds to the regularization $\lambda\rho(\boldsymbol{x})$. For example, with additive white Gaussian noise (AWGN) $\boldsymbol{e} \sim \mathcal{N}(\boldsymbol{0}, \sigma_e^2 \boldsymbol{I})$, the log-likelihood implies a quadratic loss:

$$\ell(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2\sigma_e^2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|^2. \tag{6}$$

Equivalently, the normalized loss $\ell(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|^2$ could be used if σ_e^2 was absorbed into λ.

B. ADMM

A popular approach to solving (2) is through ADMM [12], which we now review. Using variable splitting, (2) becomes

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \big\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho(\boldsymbol{v}) \big\} \text{ s.t. } \boldsymbol{x} = \boldsymbol{v}. \tag{7}$$

Using the augmented Lagrangian, problem (7) can be reformulated as

$$\min_{\boldsymbol{x},\boldsymbol{v}} \max_{\boldsymbol{p}} \Big\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho(\boldsymbol{v}) + \boldsymbol{p}^\top(\boldsymbol{x} - \boldsymbol{v}) + \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{v}\|^2 \Big\} \tag{8}$$

using Lagrange multipliers (or "dual" variables) $\boldsymbol{p}$ and a design parameter $\beta > 0$. Using $\boldsymbol{u} \triangleq \boldsymbol{p}/\beta$, (8) can be simplified to

$$\min_{\boldsymbol{x},\boldsymbol{v}} \max_{\boldsymbol{u}} \Big\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho(\boldsymbol{v}) + \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{v} + \boldsymbol{u}\|^2 - \frac{\beta}{2}\|\boldsymbol{u}\|^2 \Big\}. \tag{9}$$

Algorithm 1: ADMM [12].

Require: $\ell(\cdot;\boldsymbol{y}), \rho(\cdot), \beta, \lambda, \boldsymbol{v}_0, \boldsymbol{u}_0$, and K
1: **for** $k = 1, 2, \ldots, K$ **do**
2: $\quad \boldsymbol{x}_k = \arg\min_{\boldsymbol{x}} \{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{v}_{k-1} + \boldsymbol{u}_{k-1}\|^2\}$
3: $\quad \boldsymbol{v}_k = \arg\min_{\boldsymbol{v}} \{\lambda\rho(\boldsymbol{v}) + \frac{\beta}{2}\|\boldsymbol{v} - \boldsymbol{x}_k - \boldsymbol{u}_{k-1}\|^2\}$
4: $\quad \boldsymbol{u}_k = \boldsymbol{u}_{k-1} + \boldsymbol{x}_k - \boldsymbol{v}_k$
5: **end for**
6: Return $\boldsymbol{x}_K$

The ADMM algorithm solves (9) by alternating the minimization of $\boldsymbol{x}$ and $\boldsymbol{v}$ with gradient ascent of $\boldsymbol{u}$, as specified in Algorithm 1. ADMM is known to converge under convex $\ell(\cdot;\boldsymbol{y})$ and $\rho(\cdot)$, and other mild conditions (see [12]).

C. Plug-and-Play ADMM

Importantly, line 3 of Algorithm 1 can be recognized as variational *denoising* of $\boldsymbol{x}_k + \boldsymbol{u}_{k-1}$ using regularization $\lambda\rho(\boldsymbol{x})$ and quadratic loss $\ell(\boldsymbol{x};\boldsymbol{r}) = \frac{1}{2\nu}\|\boldsymbol{x} - \boldsymbol{r}\|^2$, where $\boldsymbol{r} = \boldsymbol{x}_k + \boldsymbol{u}_{k-1}$ at iteration k. By "denoising," we mean recovering $\boldsymbol{x}^0$ from noisy measurements $\boldsymbol{r}$ of the form

$$\boldsymbol{r} = \boldsymbol{x}^0 + \boldsymbol{e}, \quad \boldsymbol{e} \sim \mathcal{N}(\boldsymbol{0}, \nu\boldsymbol{I}), \tag{10}$$

for some variance $\nu > 0$.

Image denoising has been studied for decades (see, e.g., the overviews [2], [3]), with the result that high performance methods are now readily available. Today's state-of-the-art denoisers include those based on image-dependent filtering algorithms (e.g., BM3D [7]) or deep neural networks (e.g., TNRD [4], DnCNN [5]). Most of these denoisers are not variational in nature, i.e., they are not based on any explicit regularizer $\lambda\rho(\boldsymbol{x})$.

Leveraging the denoising interpretation of ADMM, Venkatakrishnan, Bouman, and Wolhberg [10] proposed to replace line 3 of Algorithm 1 with a call to a sophisticated image denoiser, such as BM3D, and dubbed their approach *Plug-and-Play* (PnP) ADMM. Numerical experiments show that PnP-ADMM works very well in most cases. However, when the denoiser used in PnP-ADMM comes with no explicit regularization $\rho(\boldsymbol{x})$, it is not clear what objective PnP-ADMM is minimizing, making PnP-ADMM convergence more difficult to characterize. Similar PnP algorithms have been proposed using primal-dual methods [13] and FISTA [14] in place of ADMM.

Approximate message passing (AMP) algorithms [15] also perform denoising at each iteration. In fact, when $\boldsymbol{A}$ is large and i.i.d. Gaussian, AMP constructs an internal variable statistically equivalent to $\boldsymbol{r}$ in (10) [16]. While the earliest instances of AMP assumed separable denoising (i.e., $[\boldsymbol{f}(\boldsymbol{x})]_n = f(x_n)\ \forall n$ for some f) later instances, like [17], [18], considered non-separable denoising. The paper [19] by Metzler, Maleki, and Baraniuk proposed to plug an image-specific denoising algorithm, like BM3D, into AMP. Vector AMP, which extends AMP to the broader class of "right rotationally invariant" random matrices, was proposed in [20], and VAMP with image-specific denoising was proposed in [21]. Rigorous analyses of AMP and VAMP under non-separable denoisers were performed in [22] and [23], respectively.

D. Regularization by Denoising (RED)

As discussed in the Introduction, Romano, Elad, and Milanfar [1] proposed a radically new way to exploit an image denoiser, which they call *regularization by denoising* (RED). Given an arbitrary image denoiser $\boldsymbol{f}:\mathbb{R}^N\to\mathbb{R}^N$, they proposed to construct an explicit regularizer of the form

$$\rho_{\mathsf{red}}(\boldsymbol{x}) \triangleq \frac{1}{2}\boldsymbol{x}^\top(\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})) \tag{11}$$

to use within the variational framework (2). The advantage of using an explicit regularizer is that a wide variety of optimization algorithms can be used to solve (2) and their convergence can be tractably analyzed.

In [1], numerical evidence is presented to show that image denoisers $\boldsymbol{f}(\cdot)$ are *locally homogeneous* (LH), i.e.,

$$(1+\epsilon)\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}\big((1+\epsilon)\boldsymbol{x}\big)\ \forall \boldsymbol{x} \tag{12}$$

for sufficiently small $\epsilon\in\mathbb{R}\setminus 0$. For such denoisers, Romano *et al*. claim [1, Eq.(28)] that $\rho_{\mathsf{red}}(\cdot)$ obeys the gradient rule

$$\nabla\rho_{\mathsf{red}}(\boldsymbol{x}) = \boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x}). \tag{13}$$

If $\nabla\rho_{\mathsf{red}}(\boldsymbol{x}) = \boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})$, then any minimizer $\widehat{\boldsymbol{x}}$ of the variational objective under quadratic loss,

$$\frac{1}{2\sigma^2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}\|^2 + \lambda\rho_{\mathsf{red}}(\boldsymbol{x}) \triangleq C_{\mathsf{red}}(\boldsymbol{x}), \tag{14}$$

must yield $\nabla C_{\mathsf{red}}(\widehat{\boldsymbol{x}}) = \boldsymbol{0}$, i.e., must obey

$$\boldsymbol{0} = \frac{1}{\sigma^2}\boldsymbol{A}^\top(\boldsymbol{A}\widehat{\boldsymbol{x}}-\boldsymbol{y}) + \lambda(\widehat{\boldsymbol{x}}-\boldsymbol{f}(\widehat{\boldsymbol{x}})). \tag{15}$$

Based on this line of reasoning, Romano *et al*. proposed several iterative algorithms that find an $\widehat{\boldsymbol{x}}$ satisfying the fixed-point condition (15), which we will refer to henceforth as "RED algorithms."

III. Clarifications on RED

In this section, we first show that the gradient expression (13) holds if and only if the denoiser $\boldsymbol{f}(\cdot)$ is LH and has Jacobian symmetry (JS). We then establish that many popular denoisers lack JS, such as the median filter (MF) [24], non-local means (NLM) [6], BM3D [7], TNRD [4], and DnCNN [5]. For such denoisers, the RED algorithms cannot be explained by $\rho_{\mathsf{red}}(\cdot)$ in (11). We also show a more general result: When a denoiser lacks JS, there exists no regularizer $\rho(\cdot)$ whose gradient expression matches (13). Thus, the problem is not the specific form of $\rho_{\mathsf{red}}(\cdot)$ in (11) but rather the broader pursuit of explicit regularization.

A. Preliminaries

We first state some definitions and assumptions. In the sequel, we denote the ith component of $\boldsymbol{f}(\boldsymbol{x})$ by $f_i(\boldsymbol{x})$, the gradient of $f_i(\cdot)$ at $\boldsymbol{x}$ by

$$\nabla f_i(\boldsymbol{x}) \triangleq \left[\begin{array}{ccc}\frac{\partial f_i(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f_i(\boldsymbol{x})}{\partial x_N}\end{array}\right]^\top, \tag{16}$$

and the Jacobian of $\boldsymbol{f}(\cdot)$ at $\boldsymbol{x}$ by

$$J\boldsymbol{f}(\boldsymbol{x}) \triangleq \left[\begin{array}{cccc}\frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_N}\\ \frac{\partial f_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial f_2(\boldsymbol{x})}{\partial x_N}\\ \vdots & \vdots & \ddots & \vdots\\ \frac{\partial f_N(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_N(\boldsymbol{x})}{\partial x_2} & \cdots & \frac{\partial f_N(\boldsymbol{x})}{\partial x_N}\end{array}\right]. \tag{17}$$

Without loss of generality, we take $[0,255]^N\subset\mathbb{R}^N$ to be the set of possible images. A given denoiser $\boldsymbol{f}(\cdot)$ may involve decision boundaries $\mathcal{D}\subset[0,255]^N$ at which its behavior changes suddenly. We assume that these boundaries are a closed set of measure zero and work instead with the open set $\mathcal{X}\triangleq(0,255)^N\setminus\mathcal{D}$, which contains almost all images.

We furthermore assume that $\boldsymbol{f}:\mathbb{R}^N\to\mathbb{R}^N$ is *differentiable* on $\mathcal{X}$, which means [25, p. 212] that, for any $\boldsymbol{x}\in\mathcal{X}$, there exists a matrix $\boldsymbol{J}\in\mathbb{R}^{N\times N}$ for which

$$\lim_{\boldsymbol{w}\to\boldsymbol{0}}\frac{\|\boldsymbol{f}(\boldsymbol{x}+\boldsymbol{w})-\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{J}\boldsymbol{w}\|}{\|\boldsymbol{w}\|} = 0. \tag{18}$$

When $\boldsymbol{J}$ exists, it can be shown [25, p. 216] that $\boldsymbol{J} = J\boldsymbol{f}(\boldsymbol{x})$.

B. The RED Gradient

We first recall a result that was established in [1].

Lemma 1 (Local homogeneity [1]): Suppose that denoiser $\boldsymbol{f}(\cdot)$ is locally homogeneous. Then $[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x})$.

Proof: Our proof is based on differentiability and avoids the need to define a directional derivative. From (18), we have

$$0 = \lim_{\epsilon\to 0}\frac{\|\boldsymbol{f}(\boldsymbol{x}+\epsilon\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x})-[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\epsilon\|}{\|\epsilon\boldsymbol{x}\|}\ \forall\boldsymbol{x}\in\mathcal{X} \tag{19}$$

$$= \lim_{\epsilon\to 0}\frac{\|(1+\epsilon)\boldsymbol{f}(\boldsymbol{x})-\boldsymbol{f}(\boldsymbol{x})-[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\epsilon\|}{\|\epsilon\boldsymbol{x}\|}\ \forall\boldsymbol{x}\in\mathcal{X} \tag{20}$$

$$= \lim_{\epsilon\to 0}\frac{\|\boldsymbol{f}(\boldsymbol{x})-[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}\|}{\|\boldsymbol{x}\|}\ \forall\boldsymbol{x}\in\mathcal{X}, \tag{21}$$

where (20) follows from local homogeneity (12). Equation (21) implies that $[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x} = \boldsymbol{f}(\boldsymbol{x})\ \forall\boldsymbol{x}\in\mathcal{X}$. ■

We now state one of the main results of this section.

Lemma 2 (RED gradient): For $\rho_{\mathsf{red}}(\cdot)$ defined in (11),

$$\nabla\rho_{\mathsf{red}}(\boldsymbol{x}) = \boldsymbol{x}-\frac{1}{2}\boldsymbol{f}(\boldsymbol{x})-\frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]^\top\boldsymbol{x}. \tag{22}$$

Proof: For any $\boldsymbol{x}\in\mathcal{X}$ and $n=1,\dots,N$,

$$\frac{\partial\rho_{\mathsf{red}}(\boldsymbol{x})}{\partial x_n} = \frac{\partial}{\partial x_n}\frac{1}{2}\sum_{i=1}^N\big(x_i^2 - x_i f_i(\boldsymbol{x})\big) \tag{23}$$

$$= \frac{1}{2}\frac{\partial}{\partial x_n}\left(x_n^2 - x_n f_n(\boldsymbol{x}) + \sum_{i\neq n} x_i^2 - \sum_{i\neq n} x_i f_i(\boldsymbol{x})\right) \quad (24)$$

$$= \frac{1}{2}\left(2x_n - f_n(\boldsymbol{x}) - x_n\frac{\partial f_n(\boldsymbol{x})}{\partial x_n} - \sum_{i\neq n} x_i \frac{\partial f_i(\boldsymbol{x})}{\partial x_n}\right) \quad (25)$$

$$= x_n - \frac{1}{2}f_n(\boldsymbol{x}) - \frac{1}{2}\sum_{i=1}^{N} x_i \frac{\partial f_i(\boldsymbol{x})}{\partial x_n} \quad (26)$$

$$= x_n - \frac{1}{2}f_n(\boldsymbol{x}) - \frac{1}{2}\left[[J\boldsymbol{f}(\boldsymbol{x})]^\top \boldsymbol{x}\right]_n, \quad (27)$$

using the definition of $J\boldsymbol{f}(\boldsymbol{x})$ from (17). Collecting $\{\frac{\partial \rho_{\mathsf{red}}(\boldsymbol{x})}{\partial x_n}\}_{n=1}^N$ into the gradient vector (13) yields (22). ■

Note that the gradient expression (22) differs from (13).

Lemma 3 (Clarification on (13)): Suppose that the denoiser $\boldsymbol{f}(\cdot)$ is locally homogeneous. Then the RED gradient expression (13) holds if and only if $J\boldsymbol{f}(\boldsymbol{x}) = [J\boldsymbol{f}(\boldsymbol{x})]^\top$.

Proof: If $J\boldsymbol{f}(\boldsymbol{x}) = [J\boldsymbol{f}(\boldsymbol{x})]^\top$, then the last term in (22) becomes $-\frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x}$, which equals $-\frac{1}{2}\boldsymbol{f}(x)$ by Lemma 1, in which case (22) agrees with (13). But if $J\boldsymbol{f}(\boldsymbol{x}) \neq [J\boldsymbol{f}(\boldsymbol{x})]^\top$, then (22) differs from (13). ■

C. Impossibility of Explicit Regularization

For denoisers $\boldsymbol{f}(\cdot)$ that lack Jacobian symmetry (JS), Lemma 3 establishes that the gradient expression (13) does not hold. Yet (13) leads to the fixed-point condition (15) on which all RED algorithms in [1] are based. The fact that these algorithms work well in practice suggests that "$\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$" is a desirable property for a regularizer $\rho(\boldsymbol{x})$ to have. But the regularization $\rho_{\mathsf{red}}(\boldsymbol{x})$ in (11) does not lead to this property when $\boldsymbol{f}(\cdot)$ lacks JS. Thus an important question is:

Does there exist some other regularization $\rho(\cdot)$ for which $\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$ when $\boldsymbol{f}(\cdot)$ is non-JS?

The following theorem provides the answer.

Theorem 1 (Impossibility): Suppose that denoiser $\boldsymbol{f}(\cdot)$ has a non-symmetric Jacobian. Then there exists no regularization $\rho(\cdot)$ for which $\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$.

Proof: To prove the theorem, we view $\boldsymbol{f} : \mathcal{X} \to \mathbb{R}^N$ as a vector field. Theorem 4.3.8 in [26] says that a vector field $\boldsymbol{f}$ is *conservative* if and only if there exists a continuously differentiable potential $\overline{\rho} : \mathcal{X} \to \mathbb{R}$ for which $\nabla\overline{\rho} = \boldsymbol{f}$. Furthermore, Theorem 4.3.10 in [26] says that if $\boldsymbol{f}$ is conservative, then the Jacobian $J\boldsymbol{f}$ is symmetric. Thus, by the contrapositive, if the Jacobian $J\boldsymbol{f}$ is *not* symmetric, then no such potential $\overline{\rho}$ exists.

To apply this result to our problem, we define

$$\rho(\boldsymbol{x}) \triangleq \frac{1}{2}\|\boldsymbol{x}\|^2 - \overline{\rho}(\boldsymbol{x}) \quad (28)$$

and notice that

$$\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \nabla\overline{\rho}(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}). \quad (29)$$

Thus, if $J\boldsymbol{f}(\boldsymbol{x})$ is non-symmetric, then $J[\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})] = \boldsymbol{I} - J\boldsymbol{f}(\boldsymbol{x})$ is non-symmetric, which means that there exists no ρ for which (29) holds. ■

Thus, the problem is not the specific form of $\rho_{\mathsf{red}}(\cdot)$ in (11) but rather the broader pursuit of explicit regularization. We note that the notion of conservative vector fields was discussed in [27, App. A] in the context of PnP algorithms, whereas here we discuss it in the context of RED.

D. Analysis of Jacobian Symmetry

The previous sections motivate an important question: Do commonly-used image denoisers have sufficient JS?

For some denoisers, JS can be studied analytically. For example, consider the "transform domain thresholding" (TDT) denoisers of the form

$$\boldsymbol{f}(\boldsymbol{x}) \triangleq \boldsymbol{W}^\top \boldsymbol{g}(\boldsymbol{W}\boldsymbol{x}), \quad (30)$$

where $\boldsymbol{g}(\cdot)$ performs componentwise (e.g., soft or hard) thresholding and $\boldsymbol{W}$ is some transform, as occurs in the context of wavelet shrinkage [28], with or without cycle-spinning [29]. Using $g_n'(\cdot)$ to denote the derivative of $g_n(\cdot)$, we have

$$\frac{\partial f_n(\boldsymbol{x})}{\partial x_q} = \sum_{i=1}^{N} w_{in} g_i'\left(\sum_{j=1}^{N} w_{ij}x_j\right) w_{iq} = \frac{\partial f_q(\boldsymbol{x})}{\partial x_n}, \quad (31)$$

and so the Jacobian of $\boldsymbol{f}(\cdot)$ is perfectly symmetric.

Another class of denoisers with perfectly symmetric Jacobians are those that produce MAP or MMSE optimal $\widehat{\boldsymbol{x}}$ under some assumed prior $\widehat{p}_{\mathsf{x}}$. In the MAP case, $\widehat{\boldsymbol{x}}$ minimizes (over $\boldsymbol{x}$) the cost $c(\boldsymbol{x};\boldsymbol{r}) = \frac{1}{2\nu}\|\boldsymbol{x}-\boldsymbol{r}\|^2 - \ln\widehat{p}_{\mathsf{x}}(\boldsymbol{x})$ for noisy input $\boldsymbol{r}$. If we define $\phi(\boldsymbol{r}) \triangleq \min_{\boldsymbol{x}} c(\boldsymbol{x};\boldsymbol{r})$, known as the Moreau-Yosida envelope of $-\ln\widehat{p}_{\mathsf{x}}$, then $\widehat{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{r}) = \boldsymbol{r} - \nu\nabla\phi(\boldsymbol{r})$, as discussed in [30]. (See also [31] for insightful discussions in the context of image denoising.) The elements in the Jacobian are therefore $[J\boldsymbol{f}(\boldsymbol{r})]_{n,q} = \frac{\partial f_n(\boldsymbol{r})}{\partial r_q} = \delta_{n-q} - \nu\frac{\partial^2\phi(\boldsymbol{r})}{\partial r_q \partial r_n}$, and so the Jacobian matrix is symmetric. In the MMSE case, we have that $\boldsymbol{f}(\boldsymbol{r}) = \boldsymbol{r} - \nabla\rho_{\mathsf{TR}}(\boldsymbol{r})$ for $\rho_{\mathsf{TR}}(\cdot)$ defined in (52) (see Lemma 4), and so $[J\boldsymbol{f}(\boldsymbol{r})]_{n,q} = \delta_{n-q} - \frac{\partial^2\rho_{\mathsf{TR}}(\boldsymbol{r})}{\partial r_q \partial r_n}$, again implying that the Jacobian is symmetric. But it is difficult to say anything about the Jacobian symmetry of *approximate* MAP or MMSE denoisers.

Now let us consider the more general class of denoisers

$$\boldsymbol{f}(x) = \boldsymbol{W}(\boldsymbol{x})\boldsymbol{x}, \quad (32)$$

sometimes called "pseudo-linear" [3]. For simplicity, we assume that $\boldsymbol{W}(\cdot)$ is differentiable on $\mathcal{X}$. In this case, using the chain rule, we have

$$\frac{\partial f_n(\boldsymbol{x})}{\partial x_q} = w_{nq}(\boldsymbol{x}) + \sum_{i=1}^{N}\frac{\partial w_{ni}(\boldsymbol{x})}{\partial x_q}x_i, \quad (33)$$

and so the following are sufficient conditions for Jacobian symmetry.

1) $\boldsymbol{W}(\boldsymbol{x})$ is symmetric $\forall\boldsymbol{x}\in\mathcal{X}$,
2) $\sum_{i=1}^N \frac{\partial w_{ni}(\boldsymbol{x})}{\partial x_q}x_i = \sum_{i=1}^N \frac{\partial w_{qi}(\boldsymbol{x})}{\partial x_n}x_i \ \forall\boldsymbol{x}\in\mathcal{X}$.

When $\boldsymbol{W}$ is $\boldsymbol{x}$-invariant (i.e., $\boldsymbol{f}(\cdot)$ is linear) and symmetric, both of these conditions are satisfied. This latter case was exploited for RED in [32]. The case of non-linear $\boldsymbol{W}(\cdot)$ is more compli-

TABLE I
AVERAGE JACOBIAN-SYMMETRY ERROR ON 16×16 IMAGES

	TDT	MF	NLM	BM3D	TNRD	DnCNN
$e^J_{\boldsymbol{f}}(\boldsymbol{x})$	5.36e-21	1.50	0.250	1.22	0.0378	0.0172

TABLE II
AVERAGE GRADIENT ERROR ON 16×16 IMAGES

$e^{\nabla}_{\boldsymbol{f}}(\boldsymbol{x})$	TDT	MF	NLM	BM3D	TNRD	DnCNN
$\nabla\rho_{\mathsf{red}}(\boldsymbol{x})$ from (13)	0.381	0.904	0.829	0.790	0.416	1.76
$\nabla\rho_{\mathsf{red}}(\boldsymbol{x})$ from (38)	0.381	1.78e-21	0.0446	0.447	0.356	1.69
$\nabla\rho_{\mathsf{red}}(\boldsymbol{x})$ from (22)	4.68e-19	1.75e-21	1.32e-20	4.80e-14	3.77e-19	6.76e-13

cated. Although $\boldsymbol{W}(\cdot)$ can be symmetrized (see [33], [34]), it is not clear whether the second condition above will be satisfied.

E. Jacobian Symmetry Experiments

For denoisers that do not admit a tractable analysis, we can still evaluate the Jacobian of $\boldsymbol{f}(\cdot)$ at $\boldsymbol{x}$ numerically via

$$\frac{f_i(\boldsymbol{x}+\epsilon\boldsymbol{e}_n)-f_i(\boldsymbol{x}-\epsilon\boldsymbol{e}_n)}{2\epsilon} \triangleq \big[\widehat{J\boldsymbol{f}}(\boldsymbol{x})\big]_{i,n}, \tag{34}$$

where $\boldsymbol{e}_n$ denotes the nth column of $\boldsymbol{I}_N$ and $\epsilon > 0$ is small ($\epsilon = 1 \times 10^{-3}$ in our experiments). For the purpose of quantifying JS, we define the normalized error metric

$$e^J_{\boldsymbol{f}}(\boldsymbol{x}) \triangleq \frac{\big\|\widehat{J\boldsymbol{f}}(\boldsymbol{x})-[\widehat{J\boldsymbol{f}}(\boldsymbol{x})]^\top\big\|_F^2}{\|\widehat{J\boldsymbol{f}}(\boldsymbol{x})\|_F^2}, \tag{35}$$

which should be nearly zero for a symmetric Jacobian.

Table I shows[1] the average value of $e^J_{\boldsymbol{f}}(\boldsymbol{x})$ for 17 different image patches[2] of size 16×16, using denoisers that assumed a noise variance of 25^2. The denoisers tested were the TDT from (30) with the 2D Haar wavelet transform and soft-thresholding, the median filter (MF) [24] with a 3×3 window, non-local means (NLM) [6], BM3D [7], TNRD [4], and DnCNN [5]. Table I shows that the Jacobians of all but the TDT denoiser are far from symmetric.

Jacobian symmetry is of secondary interest; what we really care about is the accuracy of the RED gradient expressions (13) and (22). To assess gradient accuracy, we numerically evaluated the gradient of $\rho_{\mathsf{red}}(\cdot)$ at $\boldsymbol{x}$ using

$$\frac{\rho_{\mathsf{red}}(\boldsymbol{x}+\epsilon\boldsymbol{e}_n)-\rho_{\mathsf{red}}(\boldsymbol{x}-\epsilon\boldsymbol{e}_n)}{2\epsilon} \triangleq \big[\widehat{\nabla\rho_{\mathsf{red}}}(\boldsymbol{x})\big]_n \tag{36}$$

and compared the result to the analytical expressions (13) and (22). Table II reports the normalized gradient error

$$e^{\nabla}_{\boldsymbol{f}}(\boldsymbol{x}) \triangleq \frac{\|\nabla\rho_{\mathsf{red}}(\boldsymbol{x})-\widehat{\nabla\rho_{\mathsf{red}}}(\boldsymbol{x})\|^2}{\|\widehat{\nabla\rho_{\mathsf{red}}}(\boldsymbol{x})\|^2} \tag{37}$$

for the same ϵ, images, and denoisers used in Table I. The results in Table II show that, for all tested denoisers, the numerical gradient $\widehat{\nabla\rho_{\mathsf{red}}}(\cdot)$ closely matches the analytical expression for $\nabla\rho_{\mathsf{red}}(\cdot)$ from (22), but not that from (13). The mismatch between $\widehat{\nabla\rho_{\mathsf{red}}}(\cdot)$ and $\nabla\rho_{\mathsf{red}}(\cdot)$ from (13) is partly due to insufficient JS and partly due to insufficient LH, as we establish below.

[1]Matlab code for the experiments is available at http://www2.ece.ohio-state.edu/∼schniter/RED/index.html.

[2]We used the center 16×16 patches of the standard Barbara, Bike, Boats, Butterfly, Cameraman, Flower, Girl, Hat, House, Leaves, Lena, Parrots, Parthenon, Peppers, Plants, Raccoon, and Starfish test images.

TABLE III
AVERAGE LOCAL-HOMOGENEITY ERROR ON 16×16 IMAGES

	TDT	MF	NLM	BM3D	TNRD	DnCNN
$e^{\mathsf{LH},1}_{\boldsymbol{f}}(\boldsymbol{x})$	2.05e-8	0	1.41e-8	7.37e-7	2.18e-8	1.63e-8
$e^{\mathsf{LH},2}_{\boldsymbol{f}}(\boldsymbol{x})$	0.0205	2.26e-23	0.0141	3.80e4	2.18e-2	0.0179

F. Local Homogeneity Experiments

Recall that the TDT denoiser has a symmetric Jacobian, both theoretically and empirically. Yet Table II reports a disagreement between the $\nabla\rho_{\mathsf{red}}(\cdot)$ expressions (13) and (22) for TDT. We now show that this disagreement is due to insufficient local homogeneity (LH).

To do this, we introduce yet another RED gradient expression,

$$\nabla\rho_{\mathsf{red}}(\boldsymbol{x}) \overset{\mathsf{LH}}{=} \boldsymbol{x} - \frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]\boldsymbol{x} - \frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]^\top\boldsymbol{x}, \tag{38}$$

which results from combining (22) with Lemma 1. Here, $\overset{\mathsf{LH}}{=}$ indicates that (38) holds under LH. In contrast, the gradient expression (13) holds under *both* LH and Jacobian symmetry, while the gradient expression (22) holds in general (i.e., even in the absence of LH and/or Jacobian symmetry). We also introduce two normalized error metrics for LH,

$$e^{\mathsf{LH},1}_{\boldsymbol{f}}(\boldsymbol{x}) \triangleq \frac{\big\|\boldsymbol{f}((1+\epsilon)\boldsymbol{x})-(1+\epsilon)\boldsymbol{f}(\boldsymbol{x})\big\|^2}{\|(1+\epsilon)\boldsymbol{f}(\boldsymbol{x})\|^2} \tag{39}$$

$$e^{\mathsf{LH},2}_{\boldsymbol{f}}(\boldsymbol{x}) \triangleq \frac{\big\|[\widehat{J\boldsymbol{f}}(\boldsymbol{x})]\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})\big\|^2}{\|\boldsymbol{f}(\boldsymbol{x})\|^2}. \tag{40}$$

which should both be nearly zero for LH $\boldsymbol{f}(\cdot)$. Note that $e^{\mathsf{LH},1}_{\boldsymbol{f}}$ quantifies LH according to definition (12) and closely matches the numerical analysis of LH in [1]. Meanwhile, $e^{\mathsf{LH},2}_{\boldsymbol{f}}$ quantifies LH according to Lemma 1 and to how LH is actually used in the gradient expressions (13) and (38).

The middle row of Table II reports the average gradient error of the gradient expression (38), and Table III reports average LH error for the metrics $e^{\mathsf{LH},1}_{\boldsymbol{f}}$ and $e^{\mathsf{LH},2}_{\boldsymbol{f}}$. There we see that the average $e^{\mathsf{LH},1}_{\boldsymbol{f}}$ error is small for all denoisers, consistent with the experiments in [1]. But the average $e^{\mathsf{LH},2}_{\boldsymbol{f}}$ error is several orders of magnitude larger (for all but the MF denoiser). We also note that the value of $e^{\mathsf{LH},2}_{\boldsymbol{f}}$ for BM3D is several orders of magnitude higher than for the other denoisers. This result is consistent with Fig. 2, which shows that the cost function associated with BM3D is much less smooth than that of the other denoisers. As discussed below, these seemingly small imperfections in LH have a significant effect on the RED gradient expressions (13) and (38).

Starting with the TDT denoiser, Table II shows that the gradient error on (38) is large, which can only be caused by insufficient LH. The insufficient LH is confirmed in Table III, which shows that the value of $e^{\mathsf{LH},2}_{\boldsymbol{f}}(\boldsymbol{x})$ for TDT is non-negligible, especially in comparison to the value for MF.

Continuing with the MF denoiser, Table I indicates that its Jacobian is far from symmetric, while Table III indicates that it is LH. The gradient results in Table II are consistent with these behaviors: the $\nabla\rho_{\mathsf{red}}(\boldsymbol{x})$ expression (38) is accurate on account of LH being satisfied, but the $\nabla\rho_{\mathsf{red}}(\boldsymbol{x})$ expression (13) is inaccurate on account of a lack of JS.

The results for the remaining denoisers NLM, BM3D, TNRD, and BM3D show a common trend: they have non-trivial levels of *both* JS error (see Table I) and LH error (see Table III). As a result, the gradient expressions (13) and (38) are *both* inaccurate (see Table II).

In conclusion, the experiments in this section show that the RED gradient expressions (13) and (38) are very sensitive to small imperfections in LH. Although the experiments in [1] suggested that many popular image denoisers are approximately LH, our experiments suggest that their levels of LH are insufficient to maintain the accuracy of the RED gradient expressions (13) and (38).

G. Hessian and Convexity

From (26), the (n,j)th element of the Hessian of $\rho_{\mathsf{red}}(\boldsymbol{x})$ equals

$$\frac{\partial^2\rho_{\mathsf{red}}(\boldsymbol{x})}{\partial x_n\partial x_j} = \frac{\partial}{\partial x_j}\left(x_n - \frac{1}{2}f_n(\boldsymbol{x}) - \frac{1}{2}\sum_{i=1}^{N} x_i\frac{\partial f_i(\boldsymbol{x})}{\partial x_n}\right) \tag{41}$$

$$= \delta_{n-j} - \frac{1}{2}\frac{\partial f_n(\boldsymbol{x})}{\partial x_j} - \frac{1}{2}\frac{\partial f_j(\boldsymbol{x})}{\partial x_n} - \frac{1}{2}x_j\frac{\partial^2 f_j(\boldsymbol{x})}{\partial x_n\partial x_j} - \frac{1}{2}\sum_{i\neq j} x_i\frac{\partial^2 f_i(\boldsymbol{x})}{\partial x_n\partial x_j} \tag{42}$$

$$= \delta_{n-j} - \frac{1}{2}\frac{\partial f_n(\boldsymbol{x})}{\partial x_j} - \frac{1}{2}\frac{\partial f_j(\boldsymbol{x})}{\partial x_n} - \frac{1}{2}\sum_{i=1}^{N} x_i\frac{\partial^2 f_i(\boldsymbol{x})}{\partial x_n\partial x_j}. \tag{43}$$

where $\delta_k = 1$ if $k = 0$ and otherwise $\delta_k = 0$. Thus, the Hessian of $\rho_{\mathsf{red}}(\cdot)$ at $\boldsymbol{x}$ equals

$$H\rho_{\mathsf{red}}(\boldsymbol{x}) = \boldsymbol{I} - \frac{1}{2}J\boldsymbol{f}(\boldsymbol{x}) - \frac{1}{2}[J\boldsymbol{f}(\boldsymbol{x})]^{\top} - \frac{1}{2}\sum_{i=1}^{N} x_i Hf_i(\boldsymbol{x}). \tag{44}$$

This expression can be contrasted with the Hessian expression from [1, (60)], which reads

$$\boldsymbol{I} - J\boldsymbol{f}(\boldsymbol{x}). \tag{45}$$

Interestingly, (44) differs from (45) even when the denoiser has a symmetric Jacobian $J\boldsymbol{f}(\boldsymbol{x})$. One implication is that, even if eigenvalues of $J\boldsymbol{f}(\boldsymbol{x})$ are limited to the interval $[0,1]$, the Hessian $H\rho_{\mathsf{red}}(\boldsymbol{x})$ may not be positive semi-definite due to the last term in (44), with possibly negative implications on the convexity of $\rho_{\mathsf{red}}(\cdot)$. That said, the RED algorithms do not actually minimize the variational objective $\ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho_{\mathsf{red}}(\boldsymbol{x})$ for common denoisers $\boldsymbol{f}(\cdot)$ (as established in Section III-H), and so the convexity of $\rho_{\mathsf{red}}(\cdot)$ may not be important in practice. We investigate the convexity of $\rho_{\mathsf{red}}(\cdot)$ numerically in Section III-I.

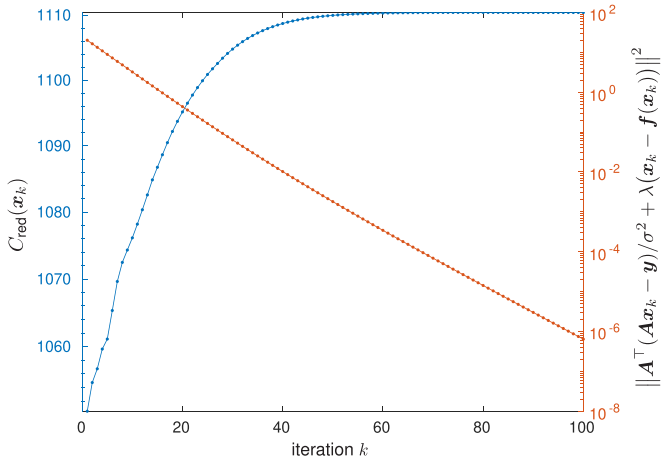

Fig. 1. RED cost $C_{\mathsf{red}}(\boldsymbol{x}_k)$ and fixed-point error $\|\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{x}_k-\boldsymbol{y})/\sigma^2 + \lambda(\boldsymbol{x}_k-\boldsymbol{f}(\boldsymbol{x}_k))\|^2$ versus iteration k for $\{\boldsymbol{x}_k\}_{k=1}^{K}$ produced by the RED-SD algorithm from [1]. Although the fixed-point condition is asymptotically satisfied, the RED cost does not decrease with k.

H. Example RED-SD Trajectory

We now provide an example of how the RED algorithms from [1] do not necessarily minimize the variational objective $\ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\rho_{\mathsf{red}}(\boldsymbol{x})$.

For a trajectory $\{\boldsymbol{x}_k\}_{k=1}^{K}$ produced by the steepest-descent (SD) RED algorithm from [1], Fig. 1 plots, versus iteration k, the RED Cost $C_{\mathsf{red}}(\boldsymbol{x}_k)$ from (14) and the error on the fixed-point condition (15), i.e., $\|\boldsymbol{g}(\boldsymbol{x}_k)\|^2$ with

$$\boldsymbol{g}(\boldsymbol{x}) \triangleq \frac{1}{\sigma^2}\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{x}-\boldsymbol{y}) + \lambda\big(\boldsymbol{x}-\boldsymbol{f}(\boldsymbol{x})\big). \tag{46}$$

For this experiment, we used the 3×3 median-filter for $\boldsymbol{f}(\cdot)$, the Starfish image, and noisy measurements $\boldsymbol{y} = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ with $\sigma^2 = 20$ (i.e., $\boldsymbol{A} = \boldsymbol{I}$ in (14)).

Fig. 1 shows that, although the RED-SD algorithm asymptotically satisfies the fixed-point condition (15), the RED cost function $C_{\mathsf{red}}(\boldsymbol{x}_k)$ does not decrease with k, as would be expected if the RED algorithms truly minimized the RED cost $C_{\mathsf{red}}(\cdot)$. This behavior implies that any optimization algorithm that monitors the objective value $C_{\mathsf{red}}(\boldsymbol{x}_k)$ for, say, backtracking line-search (e.g., the FASTA algorithm [35]), is difficult to apply in the context of RED.

I. Visualization of RED Cost and RED-Algorithm Gradient

We now show visualizations of the RED cost $C_{\mathsf{red}}(\boldsymbol{x})$ from (14) and the RED algorithm's gradient field $\boldsymbol{g}(\boldsymbol{x})$ from (46), for various image denoisers. For this experiment, we used the Starfish image, noisy measurements $\boldsymbol{y} = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ with $\sigma^2 = 100$ (i.e., $\boldsymbol{A} = \boldsymbol{I}$ in (14) and (46)), and λ optimized over a grid (of 20 values logarithmically spaced between 0.0001 and 1) for each denoiser, so that the PSNR of the RED fixed-point $\widehat{\boldsymbol{x}}$ is maximized.

Fig. 2 plots the RED cost $C_{\mathsf{red}}(\boldsymbol{x})$ and the RED algorithm's gradient field $\boldsymbol{g}(\boldsymbol{x})$ for the TDT, MF, NLM, BM3D, TNRD, and DnCNN denoisers. To visualize these quantities in two dimensions, we plotted values of $\boldsymbol{x}$ centered at the RED fixed-

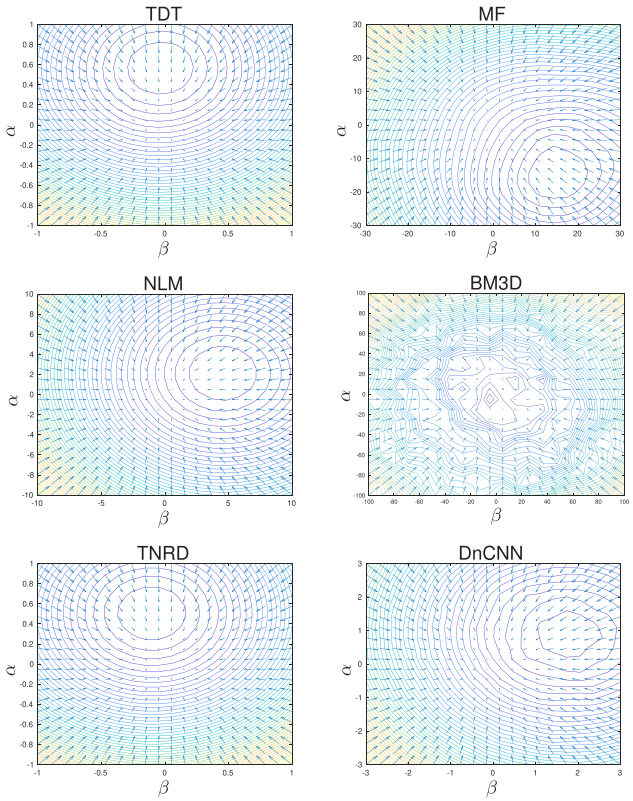

Fig. 2. Contours show RED cost $C_{\mathsf{red}}(\boldsymbol{x}_{\alpha,\beta})$ from (14) and arrows show RED-algorithm gradient field $\boldsymbol{g}(\boldsymbol{x}_{\alpha,\beta})$ from (46) versus (α,β), where $\boldsymbol{x}_{\alpha,\beta} = \widehat{\boldsymbol{x}} + \alpha\boldsymbol{e}_1 + \beta\boldsymbol{e}_2$ with randomly chosen $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. The subplots show that the minimizer of $C_{\mathsf{red}}(\boldsymbol{x}_{\alpha,\beta})$ is not the fixed-point $\widehat{\boldsymbol{x}}$, and that $C_{\mathsf{red}}(\cdot)$ may be non-smooth and/or non-convex.

point $\widehat{\boldsymbol{x}}$ and varying along two randomly chosen directions. The figure shows that the minimizer of $C_{\mathsf{red}}(\boldsymbol{x})$ does not coincide with the fixed-point $\widehat{\boldsymbol{x}}$, and that the RED cost $C_{\mathsf{red}}(\cdot)$ is not always smooth or convex.

IV. Score-Matching by Denoising

As discussed in Section II-D, the RED algorithms proposed in [1] are explicitly based on gradient rule

$$\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}). \tag{47}$$

This rule appears to be useful, since these algorithms work very well in practice. But Section III established that $\rho_{\mathsf{red}}(\cdot)$ from (11) does not usually satisfy (47). We are thus motived to seek an alternative explanation for the RED algorithms. In this section, we explain them through a framework that we call *score-matching by denoising* (SMD).

A. Tweedie Regularization

As a precursor to the SMD framework, we first propose a technique based on what we will call *Tweedie regularization*.

Recall the measurement model (10) used to define the "denoising" problem, repeated in (48) for convenience:

$$\boldsymbol{r} = \boldsymbol{x}^0 + \boldsymbol{e}, \quad \boldsymbol{e} \sim \mathcal{N}(\boldsymbol{0}, \nu\boldsymbol{I}). \tag{48}$$

To avoid confusion, we will refer to $\boldsymbol{r}$ as "pseudo-measurements" and $\boldsymbol{y}$ as "measurements." From (48), the likelihood of $\boldsymbol{x}^0$ is $p(\boldsymbol{r}|\boldsymbol{x}^0;\nu) = \mathcal{N}(\boldsymbol{r};\boldsymbol{x}^0,\nu\boldsymbol{I})$.

Now, suppose that we model the true image $\boldsymbol{x}^0$ as a realization of a random vector $\boldsymbol{x}$ with prior pdf $\widehat{p}_{\mathsf{x}}$. We write "$\widehat{p}_{\mathsf{x}}$" to emphasize that the model distribution may differ from the true distribution p_{x} (i.e., the distribution from which the image $\boldsymbol{x}$ is actually drawn). Under this prior model, the MMSE denoiser of $\boldsymbol{x}$ from $\boldsymbol{r}$ is

$$\mathbb{E}_{\widehat{p}_{\mathsf{x}}}\{\boldsymbol{x}|\boldsymbol{r}\} \triangleq \widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r}), \tag{49}$$

and the likelihood of observing $\boldsymbol{r}$ is

$$\widehat{p}_{\mathsf{r}}(\boldsymbol{r};\nu) \triangleq \int_{\mathbb{R}^N} p(\boldsymbol{r}|\boldsymbol{x};\nu)\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \tag{50}$$

$$= \int_{\mathbb{R}^N} \mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{51}$$

We will now define the *Tweedie regularizer* (TR) as

$$\rho_{\mathsf{TR}}(\boldsymbol{r};\nu) \triangleq -\nu \ln \widehat{p}_{\mathsf{r}}(\boldsymbol{r};\nu). \tag{52}$$

As we now show, $\rho_{\mathsf{TR}}(\cdot)$ has the desired property (47).

Lemma 4 (Tweedie): For $\rho_{\mathsf{TR}}(\boldsymbol{r};\nu)$ defined in (52),

$$\nabla\rho_{\mathsf{TR}}(\boldsymbol{r};\nu) = \boldsymbol{r} - \widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r}), \tag{53}$$

where $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\cdot)$ is the MMSE denoiser from (49).

Proof: Equation (53) is a direct consequence of a classical result known as Tweedie's formula [36], [37]. A short proof, from first principles, is now given for completeness.

$$\frac{\partial}{\partial r_n}\rho_{\mathsf{TR}}(\boldsymbol{r};\nu) = -\nu\frac{\partial}{\partial r_n}\ln\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})\,\mathrm{d}\boldsymbol{x} \tag{54}$$

$$= -\frac{\nu\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\frac{\partial}{\partial r_n}\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})\,\mathrm{d}\boldsymbol{x}}{\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})\,\mathrm{d}\boldsymbol{x}} \tag{55}$$

$$= \frac{\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})(r_n - x_n)\,\mathrm{d}\boldsymbol{x}}{\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})\,\mathrm{d}\boldsymbol{x}} \tag{56}$$

$$= r_n - \int_{\mathbb{R}^N} x_n \frac{\widehat{p}_{\mathsf{x}}(\boldsymbol{x})\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})}{\int_{\mathbb{R}^N}\widehat{p}_{\mathsf{x}}(\boldsymbol{x}')\mathcal{N}(\boldsymbol{r};\boldsymbol{x}',\nu\boldsymbol{I})\,\mathrm{d}\boldsymbol{x}'}\,\mathrm{d}\boldsymbol{x} \tag{57}$$

$$= r_n - \int_{\mathbb{R}^N} x_n\,\widehat{p_{\mathsf{x}|\mathsf{r}}}(\boldsymbol{x}|\boldsymbol{r};\nu)\,\mathrm{d}\boldsymbol{x} \tag{58}$$

$$= r_n - [\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r})]_n, \tag{59}$$

where (56) used $\frac{\partial}{\partial r_n}\mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I}) = \mathcal{N}(\boldsymbol{r};\boldsymbol{x},\nu\boldsymbol{I})(x_n - r_n)/\nu$. Stacking (59) for $n = 1,\ldots,N$ in a vector yields (53). ■

Thus, if the TR regularizer $\rho_{\mathsf{TR}}(\cdot;\nu)$ is used in the optimization problem (14), then the solution $\widehat{\boldsymbol{x}}$ must satisfy the fixed-point condition (15) associated with the RED algorithms from [1], albeit with an MMSE-type denoiser. This restriction will be removed using the SMD framework in Section IV-C.

It is interesting to note that the gradient property (53) holds even for non-homogeneous $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\cdot)$. This generality is important in applications under which $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\cdot)$ is known to lack LH. For example, with a binary image $\boldsymbol{x} \in \{0,1\}^N$ modeled by $\widehat{p}_{\mathsf{x}}(\boldsymbol{x}) = \prod_{n=1}^N 0.5(\delta(x_n) + \delta(x_n - 1))$, the MMSE denoiser

takes the form $[\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{x})]_n = 0.5 + 0.5\tanh(x_n/\nu)$, which is not LH.

B. Tweedie Regularization as Kernel Density Estimation

We now show that TR arises naturally in the data-driven, non-parametric context through kernel-density estimation (KDE) [8].

Recall that, in most imaging applications, the true prior p_{x} is unknown, as is the true MMSE denoiser $\boldsymbol{f}_{\mathsf{mmse},\nu}(\cdot)$. There are several ways to proceed. One way is to design "by hand" an approximate prior $\widehat{p}_{\mathsf{x}}$ that leads to a computationally efficient denoiser $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\cdot)$. But, because this denoiser is not MMSE for $\boldsymbol{x} \sim p_{\mathsf{x}}$, the performance of the resulting estimates $\widehat{\boldsymbol{x}}$ will suffer relative to $\boldsymbol{f}_{\mathsf{mmse},\nu}$.

Another way to proceed is to approximate the prior using a large corpus of training data $\{\boldsymbol{x}_t\}_{t=1}^T$. To this end, an approximate prior could be formed using the empirical estimate

$$\widehat{p}_{\mathsf{x}}(\boldsymbol{x}) = \frac{1}{T}\sum_{t=1}^{T}\delta(\boldsymbol{x}-\boldsymbol{x}_t), \tag{60}$$

but a more accurate match to the true prior p_{x} can be obtained using

$$\widetilde{p}_{\mathsf{x}}(\boldsymbol{x};\nu) = \frac{1}{T}\sum_{t=1}^{T}\mathcal{N}(\boldsymbol{x};\boldsymbol{x}_t,\nu\boldsymbol{I}) \tag{61}$$

with appropriately chosen $\nu > 0$, a technique known as kernel density estimation (KDE) or Parzen windowing [8]. Note that if $\widetilde{p}_{\mathsf{x}}$ is used as a surrogate for p_{x}, then the MAP optimization problem becomes

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{r}} \frac{1}{2\sigma^2}\|\boldsymbol{A}\boldsymbol{r}-\boldsymbol{y}\|^2 - \ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{r};\nu) \tag{62}$$

$$= \arg\min_{\boldsymbol{r}} \frac{1}{2\sigma^2}\|\boldsymbol{A}\boldsymbol{r}-\boldsymbol{y}\|^2 + \lambda\rho_{\mathsf{TR}}(\boldsymbol{r};\nu) \text{ for } \lambda = \frac{1}{\nu}, \tag{63}$$

with $\rho_{\mathsf{TR}}(\cdot;\nu)$ from (50)–(52) constructed using $\widehat{p}_{\mathsf{x}}$ from (60). In summary, TR arises naturally in the data-driven approach to image recovery when KDE is used to smooth the empirical prior.

C. Score-Matching by Denoising

A limitation of the above TR framework is that it results in denoisers $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}$ with symmetric Jacobians. (Recall the discussion of MMSE denoisers in Section III-D.) To justify the use of RED algorithms with non-symmetric Jacobians, we introduce the *score-matching by denoising* (SMD) framework in this section.

Let us continue with the KDE-based MAP estimation problem (62). Note that $\widehat{\boldsymbol{x}}$ from (62) zeros the gradient of the MAP optimization objective and thus obeys the fixed-point equation

$$\frac{1}{\sigma^2}\boldsymbol{A}^{\top}(\boldsymbol{A}\widehat{\boldsymbol{x}}-\boldsymbol{y}) - \nabla\ln\widetilde{p}_{\mathsf{x}}(\widehat{\boldsymbol{x}};\nu) = \boldsymbol{0}. \tag{64}$$

In principle, $\widehat{\boldsymbol{x}}$ in (64) could be found using gradient descent or similar techniques. However, computation of the gradient

$$\nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{r};\nu) = \frac{\nabla\widetilde{p}_{\mathsf{x}}(\boldsymbol{r};\nu)}{\widetilde{p}_{\mathsf{x}}(\boldsymbol{r};\nu)} = \frac{\sum_{t=1}^{T}(\boldsymbol{x}_t-\boldsymbol{r})\mathcal{N}(\boldsymbol{r};\boldsymbol{x}_t,\nu\boldsymbol{I})}{\nu\sum_{t=1}^{T}\mathcal{N}(\boldsymbol{r};\boldsymbol{x}_t,\nu\boldsymbol{I})} \tag{65}$$

is too expensive for the values of T typically needed to generate a good image prior $\widetilde{p}_{\mathsf{x}}$.

A tractable alternative is suggested by the fact that

$$\nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{r};\nu) = \frac{\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r})-\boldsymbol{r}}{\nu} \tag{66}$$

$$\text{for } \widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r}) = \frac{\sum_{t=1}^{T}\boldsymbol{x}_t\mathcal{N}(\boldsymbol{r};\boldsymbol{x}_t,\nu\boldsymbol{I})}{\sum_{t=1}^{T}\mathcal{N}(\boldsymbol{r};\boldsymbol{x}_t,\nu\boldsymbol{I})}, \tag{67}$$

where $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{r})$ is the MMSE estimator of $\boldsymbol{x}\sim\widehat{p}_{\mathsf{x}}$ from $\boldsymbol{r} = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0},\nu\boldsymbol{I})$. In particular, if we can construct a good approximation to $\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\cdot)$ using a denoiser $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$ in a computationally efficient function class $\mathcal{F} \triangleq \{\boldsymbol{f}_{\boldsymbol{\theta}} : \boldsymbol{\theta}\in\boldsymbol{\Theta}\}$, then we can efficiently approximate the MAP problem (62).

This approach can be formalized using the framework of *score matching* [38], which aims to approximate the "score" (i.e., the gradient of the log-prior) rather than the prior itself. For example, suppose that we want to want to approximate the score $\nabla\ln\widetilde{p}_{\mathsf{x}}(\cdot;\nu)$. For this, Hyvärinen [38] suggested to first find the best mean-square fit among a set of computationally efficient functions $\boldsymbol{\psi}(\cdot;\boldsymbol{\theta})$, i.e., find

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\widehat{p}_{\mathsf{x}}}\left\{\|\boldsymbol{\psi}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{x};\nu)\|^2\right\}, \tag{68}$$

and then to approximate the score $\nabla\ln\widetilde{p}_{\mathsf{x}}(\cdot;\nu)$ by $\boldsymbol{\psi}(\cdot;\widehat{\boldsymbol{\theta}})$. Later, in the context of denoising autoencoders, Vincent [39] showed that if one chooses

$$\boldsymbol{\psi}(\boldsymbol{x};\boldsymbol{\theta}) = \frac{\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x})-\boldsymbol{x}}{\nu} \tag{69}$$

for some function $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)\in\mathcal{F}$, then $\widehat{\boldsymbol{\theta}}$ from (68) can be equivalently written as

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{\widehat{p}_{\mathsf{x}}}\left\{\left\|\boldsymbol{f}_{\boldsymbol{\theta}}\big(\boldsymbol{x}+\mathcal{N}(0,\nu\boldsymbol{I})\big)-\boldsymbol{x}\right\|^2\right\}. \tag{70}$$

In this case, $\boldsymbol{f}_{\widehat{\boldsymbol{\theta}}}(\cdot)$ is the MSE-optimal denoiser, averaged over $\widehat{p}_{\mathsf{x}}$ and constrained to the function class $\mathcal{F}$.

Note that the denoiser approximation error can be directly connected to the score-matching error as follows. For any denoiser $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$ and any input $\boldsymbol{x}$,

$$\|\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) - \widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{x})\|^2$$

$$= \nu^2\left\|\frac{\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x})-\boldsymbol{x}}{\nu} - \nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{x};\nu)\right\|^2 \tag{71}$$

$$= \nu^2\|\boldsymbol{\psi}(\boldsymbol{x};\boldsymbol{\theta}) - \nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{x};\nu)\|^2 \tag{72}$$

where (71) follows from (66) and (72) follows from (69). Thus, matching the score is directly related to matching the MMSE denoiser.

Plugging the score approximation (69) into the fixed-point condition (64), we get

$$\frac{1}{\sigma^2}\boldsymbol{A}^{\top}(\boldsymbol{A}\widehat{\boldsymbol{x}}-\boldsymbol{y})+\lambda\big(\widehat{\boldsymbol{x}}-\boldsymbol{f}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{x}})\big)=\boldsymbol{0}\text{ for }\lambda=\frac{1}{\nu}, \quad (73)$$

which matches the fixed-point condition (15) of the RED algorithms from [1]. Here we emphasize that $\mathcal{F}$ may be constructed in such a way that $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$ has a non-symmetric Jacobian, which is the case for many state-of-the-art denoisers. Also, $\boldsymbol{\theta}$ does not need to be optimized for (73) to hold. Finally, $\widehat{p}_{\mathsf{x}}$ need not be the empirical prior (60); it can be any chosen prior [39]. Thus, the score-matching-by-denoising (SMD) framework offers an explanation of the RED algorithms from [1] that holds for generic denoisers $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$, whether or not they have symmetric Jacobians, are locally homogeneous, or MMSE. Furthermore, it suggests a rationale for choosing the regularization weight λ and, in the context of KDE, the denoiser variance ν.

D. Relation to Existing Work

Tweedie's formula (53) has connections to Stein's Unbiased Risk Estimation (SURE) [40], as discussed in, e.g., [41, Thm. 2] and [42, Eq. (2.4)]. SURE has been used for image denoising in, e.g., [43]. Tweedie's formula was also used in [44] to interpret autoencoding-based image priors. In our work, Tweedie's forumula is used to provide an interpretation for the RED algorithms through the construction of the explicit regularizer (52) and the approximation of the resulting fixed-point equation (64) via score matching.

Recently, Alain and Bengio [45] studied the contractive autoencoders, a type of autoencoder that minimizes squared reconstruction error plus a penalty that tries to make the autoencoder as simple as possible. While previous works such as [46] conjectured that such auto-encoders minimize an energy function, Alain and Bengio showed that they actually minimize the norm of a score (i.e., match a score to zero). Furthermore, they showed that, when the coder and decoder do not share the same weights, it is not possible to define a valid energy function because the Jacobian of the reconstruction function is not symmetric. The results in [45] parallel those in this paper, except that they focus on auto-encoders while we focus on variational image recovery. Another small difference is that [45] uses the small-ν approximation

$$\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{x})=\boldsymbol{x}+\nu\nabla\ln\widehat{p}_{\mathsf{x}}(\boldsymbol{x})+o(\nu), \quad (74)$$

whereas we use the exact (Tweedie's) relationship (53), i.e.,

$$\widehat{\boldsymbol{f}}_{\mathsf{mmse},\nu}(\boldsymbol{x})=\boldsymbol{x}+\nu\nabla\ln\widetilde{p}_{\mathsf{x}}(\boldsymbol{x}), \quad (75)$$

where is $\widetilde{p}_{\mathsf{x}}$ the "Gaussian blurred" version of $\widehat{p}_{\mathsf{x}}$ from (51).

V. Fast RED Algorithms

In [1], Romano *et al*. proposed several ways to solve the fixed-point equation (15). Throughout our paper, we have been referring to these methods as "RED algorithms." In this section, we provide new interpretations of the RED-ADMM and RED-FP algorithms from [1] and we propose new RED algorithms based on accelerated proximal gradient methods.

Algorithm 2: RED-ADMM with I Inner Iterations [1].

Require: $\ell(\cdot;\boldsymbol{y}),\boldsymbol{f}(\cdot),\beta,\lambda,\boldsymbol{v}_0,\boldsymbol{u}_0,K$, and I
1: **for** $k=1,2,\dots,K$ **do**
2: $\quad\boldsymbol{x}_k=\arg\min_{\boldsymbol{x}}\{\ell(\boldsymbol{x};\boldsymbol{y})+\frac{\beta}{2}\|\boldsymbol{x}-\boldsymbol{v}_{k-1}+\boldsymbol{u}_{k-1}\|^2\}$
3: $\quad\boldsymbol{z}_0=\boldsymbol{v}_{k-1}$
4: $\quad$**for** $i=1,2,\dots,I$ **do**
5: $\quad\quad\boldsymbol{z}_i=\frac{\lambda}{\lambda+\beta}\boldsymbol{f}(\boldsymbol{z}_{i-1})+\frac{\beta}{\lambda+\beta}(\boldsymbol{x}_k+\boldsymbol{u}_{k-1})$
6: $\quad$**end for**
7: $\quad\boldsymbol{v}_k=\boldsymbol{z}_I$
8: $\quad\boldsymbol{u}_k=\boldsymbol{u}_{k-1}+\boldsymbol{x}_k-\boldsymbol{v}_k$
9: **end for**
10: Return $\boldsymbol{x}_K$

A. RED-ADMM

The ADMM approach was summarized in Algorithm 1 for an arbitrary regularizer $\rho(\cdot)$. To apply ADMM to RED, line 3 of Algorithm 1, known as the "proximal update," must be specialized to the case where $\rho(\cdot)$ obeys (13) for some denoiser $\boldsymbol{f}(\cdot)$. To do this, Romano *et al*. [1] proposed the following. Because $\rho(\cdot)$ is differentiable, the proximal solution $\boldsymbol{v}_k$ must obey the fixed-point relationship

$$\boldsymbol{0}=\lambda\nabla\rho(\boldsymbol{v}_k)+\beta(\boldsymbol{v}_k-\boldsymbol{x}_k-\boldsymbol{u}_{k-1}) \quad (76)$$

$$=\lambda\big(\boldsymbol{v}_k-\boldsymbol{f}(\boldsymbol{v}_k)\big)+\beta(\boldsymbol{v}_k-\boldsymbol{x}_k-\boldsymbol{u}_{k-1}) \quad (77)$$

$$\Leftrightarrow\ \boldsymbol{v}_k=\frac{\lambda}{\lambda+\beta}\boldsymbol{f}(\boldsymbol{v}_k)+\frac{\beta}{\lambda+\beta}(\boldsymbol{x}_k+\boldsymbol{u}_{k-1}). \quad (78)$$

An approximation to $\boldsymbol{v}_k$ can thus be obtained by iterating

$$\boldsymbol{z}_i=\frac{\lambda}{\lambda+\beta}\boldsymbol{f}(\boldsymbol{z}_{i-1})+\frac{\beta}{\lambda+\beta}(\boldsymbol{x}_k+\boldsymbol{u}_{k-1}) \quad (79)$$

over $i=1,\dots,I$ with sufficiently large I, initialized at $\boldsymbol{z}_0=\boldsymbol{v}_{k-1}$. This procedure is detailed in lines 3-6 of Algorithm 2. The overall algorithm is known as RED-ADMM.

B. Inexact RED-ADMM

Algorithm 2 gives a faithful implementation of ADMM when the number of inner iterations, I, is large. But using many inner iterations may be impractical when the denoiser is computationally expensive, as in the case of BM3D or TNRD. Furthermore, the use of many inner iterations may not be necessary.

For example, Fig. 3 plots PSNR trajectories versus runtime for TNRD-based RED-ADMM with $I=1,2,3,4$ inner iterations. For this experiment, we used the deblurring task described in Section V-G, but similar behaviors can be observed in other applications of RED. Fig. 3 suggests that $I=1$ inner iterations gives the fastest convergence. Note that [1] also used $I=1$ when implementing RED-ADMM.

With $I=1$ inner iterations, RED-ADMM simplifies down to the 3-step iteration summarized in Algorithm 3. Since Algorithm 3 looks quite different than standard ADMM (recall Algorithm 1), one might wonder whether there exists another interpretation of Algorithm 3. Noting that line 3 can be rewritten

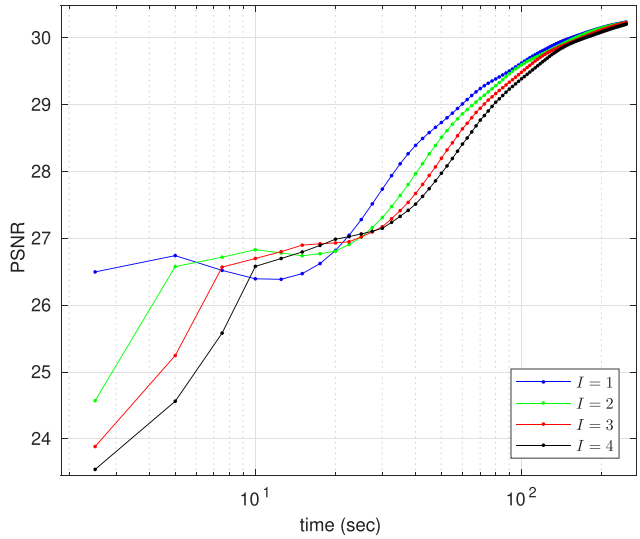

Fig. 3. PSNR versus runtime for RED-ADMM with TNRD denoising and I inner iterations.

Algorithm 3: RED-ADMM with $I = 1$.

Require: $\ell(\cdot;\boldsymbol{y})$, $\boldsymbol{f}(\cdot)$, $\beta, \lambda, \boldsymbol{v}_0, \boldsymbol{u}_0$, and K

for $k = 1, 2, \dots, K$ **do**
  $\boldsymbol{x}_k = \arg\min_{\boldsymbol{x}}\{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\beta}{2}\|\boldsymbol{x} - \boldsymbol{v}_{k-1} + \boldsymbol{u}_{k-1}\|^2\}$
  $\boldsymbol{v}_k = \frac{\lambda}{\lambda+\beta}\boldsymbol{f}(\boldsymbol{v}_{k-1}) + \frac{\beta}{\lambda+\beta}(\boldsymbol{x}_k + \boldsymbol{u}_{k-1})$
  $\boldsymbol{u}_k = \boldsymbol{u}_{k-1} + \boldsymbol{x}_k - \boldsymbol{v}_k$
end for
Return $\boldsymbol{x}_K$

as

$$\boldsymbol{v}_k = \boldsymbol{v}_{k-1} - \frac{1}{\lambda+\beta}\big[\lambda\nabla\rho(\boldsymbol{v}_{k-1}) + \beta(\boldsymbol{v}_{k-1} - \boldsymbol{x}_k - \boldsymbol{u}_{k-1})\big] \tag{80}$$

$$= \boldsymbol{v}_{k-1} - \frac{1}{\lambda+\beta}\nabla\left[\lambda\rho(\boldsymbol{v}) + \frac{\beta}{2}\|\boldsymbol{v} - \boldsymbol{x}_k - \boldsymbol{u}_{k-1}\|^2\right]_{\boldsymbol{v}=\boldsymbol{v}_{k-1}} \tag{81}$$

we see that the $I = 1$ version of inexact RED-ADMM replaces the proximal step with a gradient-descent step under stepsize $1/(\lambda+\beta)$. Thus the algorithm is reminiscent of the proximal gradient (PG) algorithm [47], [48]. We will discuss PG further in the sequel.

C. Majorization-Minimization and Proximal-Gradient RED

We now propose a proximal-gradient approach inspired by *majorization minimization* (MM) [49]. As proposed in [50], we use a quadratic upper-bound,

$$\overline{\rho}(\boldsymbol{x};\boldsymbol{x}_k) \triangleq \rho(\boldsymbol{x}_k) + [\nabla\rho(\boldsymbol{x}_k)]^\top(\boldsymbol{x} - \boldsymbol{x}_k) + \frac{L}{2}\|\boldsymbol{x} - \boldsymbol{x}_k\|_2^2, \tag{82}$$

on the regularizer $\rho(\boldsymbol{x})$, in place of $\rho(\boldsymbol{x})$ itself, at the kth algorithm iteration. Note that if $\rho(\cdot)$ is convex and $\nabla\rho(\cdot)$ is L_ρ-Lipschitz, then $\overline{\rho}(\boldsymbol{x};\boldsymbol{x}_k)$ "majorizes" $\rho(\boldsymbol{x})$ at $\boldsymbol{x}_k$ when $L \geq L_\rho$,

Algorithm 4: RED-PG Algorithm.

Require: $\ell(\cdot;\boldsymbol{y})$, $\boldsymbol{f}(\cdot)$, $\lambda, \boldsymbol{v}_0, L > 0$, and K

for $k = 1, 2, \dots, K$ **do**
  $\boldsymbol{x}_k = \arg\min_{\boldsymbol{x}}\{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\|\boldsymbol{x} - \boldsymbol{v}_{k-1}\|^2\}$
  $\boldsymbol{v}_k = \frac{1}{L}\boldsymbol{f}(\boldsymbol{x}_k) - \frac{1-L}{L}\boldsymbol{x}_k$
end for
Return $\boldsymbol{x}_K$

i.e.,

$$\overline{\rho}(\boldsymbol{x};\boldsymbol{x}_k) \geq \rho(\boldsymbol{x})\ \forall\boldsymbol{x} \in \mathcal{X} \tag{83}$$

$$\overline{\rho}(\boldsymbol{x}_k;\boldsymbol{x}_k) = \rho(\boldsymbol{x}_k). \tag{84}$$

The majorized objective can then be minimized using the *proximal gradient* (PG) algorithm [47], [48] (also known as forward-backward splitting) as follows. From (82), note that the majorized objective can be written as

$$\ell(\boldsymbol{x};\boldsymbol{y}) + \lambda\overline{\rho}(\boldsymbol{x};\boldsymbol{x}_k)$$

$$= \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\left\|\boldsymbol{x} - \left(\boldsymbol{x}_k - \frac{1}{L}\nabla\rho(\boldsymbol{x}_k)\right)\right\|^2 + \text{const} \tag{85}$$

$$= \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\Bigg\|\boldsymbol{x} - \underbrace{\left(\boldsymbol{x}_k - \frac{1}{L}\big(\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_k)\big)\right)}_{\triangleq\,\boldsymbol{v}_k}\Bigg\|^2 + \text{const}, \tag{86}$$

where (86) follows from assuming (47), which is the basis for all RED algorithms. The RED-PG algorithm then alternately updates $\boldsymbol{v}_k$ as per the gradient step in (86) and updates $\boldsymbol{x}_{k+1}$ according to the proximal step

$$\boldsymbol{x}_{k+1} = \arg\min_{\boldsymbol{x}}\left\{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\|\boldsymbol{x} - \boldsymbol{v}_k\|^2\right\}, \tag{87}$$

as summarized in Algorithm 4. Convergence is guaranteed if $L \geq L_\rho$; see [47], [48] for details.

We now show that RED-PG with $L = 1$ is identical to the "fixed point" (FP) RED algorithm proposed in [1]. First, notice from Algorithm 4 that $\boldsymbol{v}_k = \boldsymbol{f}(\boldsymbol{x}_k)$ when $L = 1$, in which case

$$\boldsymbol{x}_k = \arg\min_{\boldsymbol{x}}\left\{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}_{k-1})\|^2\right\}. \tag{88}$$

For the quadratic loss $\ell(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2\sigma^2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|^2$, (88) becomes

$$\boldsymbol{x}_k = \arg\min_{\boldsymbol{x}}\left\{\frac{1}{2\sigma^2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{y}\|^2 + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}_{k-1})\|^2\right\} \tag{89}$$

$$= \left(\frac{1}{\sigma^2}\boldsymbol{A}^\top\boldsymbol{A} + \lambda\boldsymbol{I}\right)^{-1}\left(\frac{1}{\sigma^2}\boldsymbol{A}^\top\boldsymbol{y} + \lambda\boldsymbol{f}(\boldsymbol{x}_{k-1})\right), \tag{90}$$

which is exactly the RED-FP update [1, (37)]. Thus, (88) generalizes [1, (37)] to possibly non-quadratic[3] loss $\ell(\cdot;\boldsymbol{y})$, and

[3]The extension to non-quadratic loss is important for applications like phase-retrieval, where RED has been successfully applied [51].

Algorithm 5: RED-DPG Algorithm.

Require: $\ell(\cdot;\boldsymbol{y}), \boldsymbol{f}(\cdot), \lambda, \boldsymbol{v}_0, L_0 > 0, L_\infty > 0$, and K

1: **for** $k = 1, 2, \dots, K$ **do**
2: $\quad \boldsymbol{x}_k = \arg\min_{\boldsymbol{x}} \{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L_{k-1}}{2}\|\boldsymbol{x} - \boldsymbol{v}_{k-1}\|^2\}$
3: $\quad L_k = \left(\frac{1}{L_\infty} + (\frac{1}{L_0} - \frac{1}{L_\infty})\frac{1}{\sqrt{k+1}}\right)^{-1}$
4: $\quad \boldsymbol{v}_k = \frac{1}{L_k}\boldsymbol{f}(\boldsymbol{x}_k) - \frac{1-L_k}{L_k}\boldsymbol{x}_k$
5: **end for**
6: Return $\boldsymbol{x}_K$

Algorithm 6: RED-APG Algorithm.

Require: $\ell(\cdot;\boldsymbol{y}), \boldsymbol{f}(\cdot), \lambda, \boldsymbol{v}_0, L > 0$, and K

1: $t_0 = 1$
2: **for** $k = 1, 2, \dots, K$ **do**
3: $\quad \boldsymbol{x}_k = \arg\min_{\boldsymbol{x}} \{\ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\|\boldsymbol{x} - \boldsymbol{v}_{k-1}\|^2\}$
4: $\quad t_k = \frac{1+\sqrt{1+4t_{k-1}^2}}{2}$
5: $\quad \boldsymbol{z}_k = \boldsymbol{x}_k + \frac{t_{k-1}-1}{t_k}(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})$
6: $\quad \boldsymbol{v}_k = \frac{1}{L}\boldsymbol{f}(\boldsymbol{z}_k) - \frac{1-L}{L}\boldsymbol{z}_k$
7: **end for**
8: Return $\boldsymbol{x}_K$

RED-PG generalizes RED-FP to arbitrary $L > 0$. More importantly, the PG framework facilitates algorithmic acceleration, as we describe below.

The RED-PG and inexact RED-ADMM-$I=1$ algorithms show interesting similarities: both alternate a proximal update on the loss with a gradient update on the regularization, where the latter term manifests as a convex combination between the denoiser output and another term. The difference is that RED-ADMM-$I=1$ includes an extra state variable, $\boldsymbol{u}_k$. The experiments in Section V-G suggest that this extra state variable is not necessarily advantageous.

D. Dynamic RED-PG

Recalling from (86) that $1/L$ acts as a stepsize in the PG gradient step, it may be possible to speed up PG by decreasing L, although making L too small can prevent convergence. If $\rho(\cdot)$ was known, then a line search could be used, at each iteration k, to find the smallest value of L that guarantees the majorization of $\rho(\boldsymbol{x})$ by $\overline{\rho}(\boldsymbol{x};\boldsymbol{x}_k)$ [47]. However, with a non-LH or non-JS denoiser, it is not possible to evaluate $\rho(\cdot)$, preventing such a line search.

We thus propose to vary L_k (i.e., the value of L at iteration k) according to a fixed schedule. In particular, we propose to select L_0 and L_∞, and smoothly interpolate between them at intermediate iterations k. One interpolation scheme that works well in practice is summarized in line 3 of Algorithm 5. We refer to this approach as "dynamic PG" (DPG). The numerical experiments in Section V-G suggest that, with appropriate selection of L_0 and L_∞, RED-DPG can be significantly faster than RED-FP.

E. Accelerated RED-PG

Another well-known approach to speeding up PG is to apply momentum to the $\boldsymbol{v}_k$ term in Algorithm 4 [47], often known as "acceleration." An accelerated PG (APG) approach to RED is detailed in Algorithm 6. There, the momentum in line 5 takes the same form as in FISTA [52]. The numerical experiments in Section V-G suggest that RED-APG is the fastest among the RED algorithms discussed above.

By leveraging the principle of vector extrapolation (VE) [53], a different approach to accelerating RED algorithms was recently proposed in [54]. Algorithmically, the approach in [54] is much more complicated than the PG-DPG and PG-APG methods proposed above. In fact, we have been unable to arrive at an implementation of [54] that reproduces the results in that paper, and the authors have not been willing to share their implementation with us. Thus, we cannot comment further on the difference in performance between our PG-DPG and PG-APG schemes and the one in [54].

F. Convergence of RED-PG

Recalling Theorem 1, the RED algorithms do not minimize an explicit cost function but rather seek fixed points of (15). Therefore, it is important to know whether they actually converge to any one fixed point. Below, we use the theory of non-expansive and α-averaged operators to establish the convergence of RED-PG to a fixed point under certain conditions.

First, an operator $\boldsymbol{B}(\cdot)$ is said to be *non-expansive* if its Lipschitz constant is at most 1 [55]. Next, for $\alpha \in (0,1)$, an operator $\boldsymbol{P}(\cdot)$ is said to be α-*averaged* if

$$\boldsymbol{P}(\boldsymbol{x}) = \alpha \boldsymbol{B}(\boldsymbol{x}) + (1-\alpha)\boldsymbol{x} \tag{91}$$

for some non-expansive $\boldsymbol{B}(\cdot)$. Furthermore, if $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ are α_1 and α_2-averaged, respectively, then [55, Prop. 4.32] establishes that the composition $\boldsymbol{P}_2 \circ \boldsymbol{P}_1$ is α-averaged with

$$\alpha = \frac{2}{1 + \frac{1}{\max\{\alpha_1, \alpha_2\}}}. \tag{92}$$

Recalling RED-PG from Algorithm 4, let us define an operator called $\boldsymbol{T}(\cdot)$ that summarizes one algorithm iteration:

$$\boldsymbol{T}(\boldsymbol{x}) \triangleq \arg\min_{\boldsymbol{z}} \left\{\ell(\boldsymbol{z};\boldsymbol{y}) + \tfrac{\lambda L}{2}\left\|\boldsymbol{z} - \left(\tfrac{1}{L}\boldsymbol{f}(\boldsymbol{x}) - \tfrac{1-L}{L}\boldsymbol{x}\right)\right\|^2\right\} \tag{93}$$

$$= \operatorname{prox}_{\ell/(\lambda L)}\left(\tfrac{1}{L}(\boldsymbol{f}(\boldsymbol{x}) - (1-L)\boldsymbol{x})\right) \tag{94}$$

Lemma 5: If $\ell(\cdot)$ is proper, convex, and continuous; $\boldsymbol{f}(\cdot)$ is non-expansive; and $L > 1$, then $\boldsymbol{T}(\cdot)$ from (94) is α-averaged with $\alpha = \max\{\frac{2}{1+L}, \frac{2}{3}\}$.

Proof: First, because $\ell(\cdot)$ is proper, convex, and continuous, we know that the proximal operator $\operatorname{prox}_{\ell/(\lambda L)}(\cdot)$ is α-averaged with $\alpha = 1/2$ [55]. Then, by definition, $\frac{1}{L}\boldsymbol{f}(\boldsymbol{z}) - \frac{1-L}{L}\boldsymbol{z}$ is α-averaged with $\alpha = 1/L$. From (94), $\boldsymbol{T}(\cdot)$ is the composition of these two α-averaged operators, and so from (92) we have that $\boldsymbol{T}(\cdot)$ is α-averaged with $\alpha = \max\{\frac{2}{1+L}, \frac{2}{3}\}$. ■

With Lemma 5, we can prove the convergence of RED-PG.

Theorem 2: If $\ell(\cdot)$ is proper, convex, and continuous; $\boldsymbol{f}(\cdot)$ is non-expansive; $L > 1$; and $\boldsymbol{T}(\cdot)$ from (94) has at least one fixed point, then RED-PG converges.

Proof: From (94), we have that Algorithm 4 is equivalent to

$$\boldsymbol{x}_{k+1} = \boldsymbol{T}(\boldsymbol{x}_k) \tag{95}$$

$$= \alpha \boldsymbol{B}(\boldsymbol{x}_k) + (1-\alpha)\boldsymbol{x}_k \tag{96}$$

where $\boldsymbol{B}(\cdot)$ is an implicit non-expansive operator that must exist under the definition of α-averaged operators from (91). The iteration (96) can be recognized as a Mann iteration [30], since $\alpha \in (0,1)$. Thus, from [55, Thm. 5.14], $\{\boldsymbol{x}_k\}$ is a convergent sequence, in that there exists a fixed point $\boldsymbol{x}_\star \in \mathbb{R}^N$ such that $\lim_{k\to\infty} \|\boldsymbol{x}_k - \boldsymbol{x}_\star\| = 0$. ■

We note that similar Mann-based techniques were used in [9], [56] to prove the convergence of PnP-based algorithms. Also, we conjecture that similar techniques may be used to prove the convergence of other RED algorithms, but we leave the details to future work. Experiments in Section V-G numerically study the convergence behavior of several RED algorithms with different image denoisers $\boldsymbol{f}(\cdot)$.

G. Algorithm Comparison: Image Deblurring

We now compare the performance of the RED algorithms discussed above (i.e., inexact ADMM, FP, DPG, APG, and PG) on the image deblurring problem considered in [1, Sec. 6.1]. For these experiments, the measurements $\boldsymbol{y}$ were constructed using a 9×9 uniform blur kernel for $\boldsymbol{A}$ and using AWGN with variance $\sigma^2 = 2$. As stated earlier, the image $\boldsymbol{x}$ is normalized to have pixel intensities in the range $[0, 255]$.

For the first experiment, we used the TNRD denoiser. The various algorithmic parameters were chosen based on the recommendations in [1]: the regularization weight was $\lambda = 0.02$, the ADMM penalty parameter was $\beta = 0.001$, and the noise variance assumed by the denoiser was $\nu = 3.25^2$. The proximal step on $\ell(\boldsymbol{x}; \boldsymbol{y})$, given in (90), was implemented with an FFT. For RED-DPG we used[4] $L_0 = 0.2$ and $L_\infty = 2$, for RED-APG we used $L = 1$, and for RED-PG we used $L = 1.01$ since Theorem 2 motivates $L > 1$.

Fig. 4 shows

$$\text{PSNR}_k \triangleq -10 \log_{10}\left(\frac{1}{N256^2}\|\boldsymbol{x} - \widehat{\boldsymbol{x}}_k\|^2\right)$$

versus iteration k for the starfish test image. In the figure, the proposed RED-DPG and RED-APG algorithms appear significantly faster than the RED-FP and RED-ADMM-$I=1$ algorithms proposed in [1]. For example, RED-APG reaches PSNR $= 30$ in 15 iterations whereas RED-FP and inexact RED-ADMM-$I = 1$ take about 50 iterations.

Fig. 5 shows the fixed-point error

$$\frac{1}{N}\left\|\frac{1}{\sigma^2}\boldsymbol{A}^H(\boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{y}) + \lambda(\boldsymbol{x}_k - \boldsymbol{f}(\boldsymbol{x}_k))\right\|^2$$

verus iteration k. All but the RED-APG and RED-ADMM algorithms appear to converge to the solution set of the fixed-point equation (15). The RED-APG and RED-ADMM algorithms appear to approximately satisfy the fixed-point equation (15), but not exactly satisfy (15), since the fixed-point error does not decay to zero.

[4]Matlab code for these experiments is available at http://www2.ece.ohio-state.edu/∼schniter/RED/index.html.

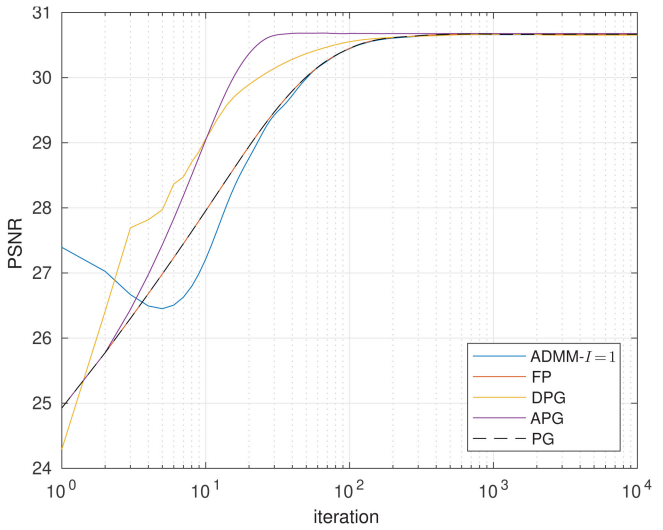

Fig. 4. PSNR versus iteration for RED algorithms with TNRD denoising when deblurring the starfish.

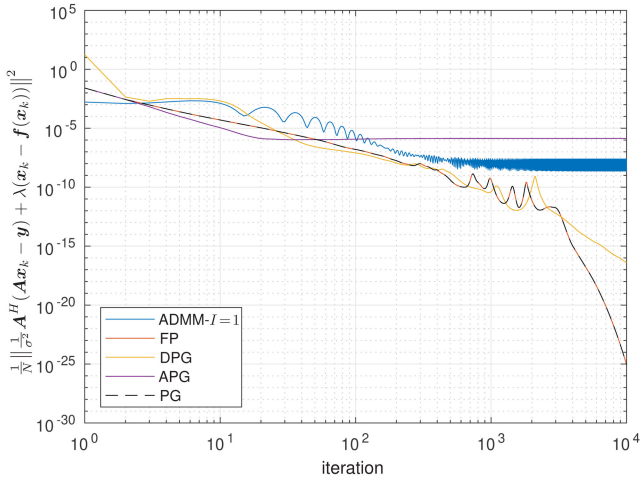

Fig. 5. Fixed-point error versus iteration for RED algorithms with TNRD denoising when deblurring the starfish.

Fig. 6 shows the update distance $\frac{1}{N}\|\boldsymbol{x}_k - \boldsymbol{x}_{k-1}\|^2$ vs. iteration k for the algorithms under test. For most algorithms, the update distance appears to be converging to zero, but for RED-APG and RED-ADMM it does not. This suggests that the RED-APG and RED-ADMM algorithms are converging to a limit cycle rather than a unique limit point.

Next, we replace the TNRD denoiser with the TDT denoiser from (30) and repeat the previous experiments. For the TDT denoiser, we used a Haar-wavelet based orthogonal discrete wavelet transform (DWT) $\boldsymbol{W}$, with the maximum number of decomposition levels, and a soft-thresholding function $\boldsymbol{g}(\cdot)$ with threshold value 0.001. Unlike the TNRD denoiser, this TDT denoiser is the proximal operator associated with a convex cost function, and so we know that it is $\frac{1}{2}$-averaged and non-expansive.

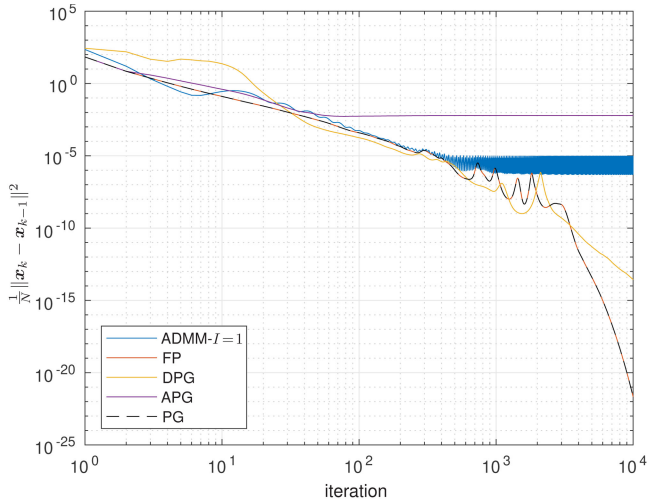

Fig. 6. Update distance versus iteration for RED algorithms with TNRD denoising when deblurring the starfish.

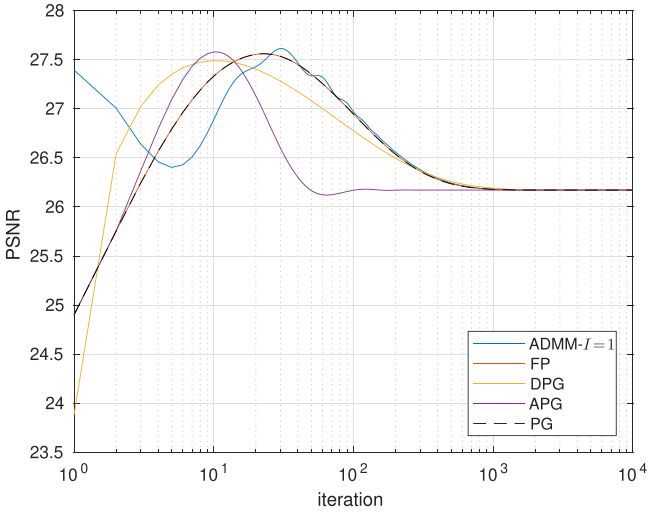

Fig. 7. PSNR versus iteration for RED algorithms with TDT denoising when deblurring the starfish.

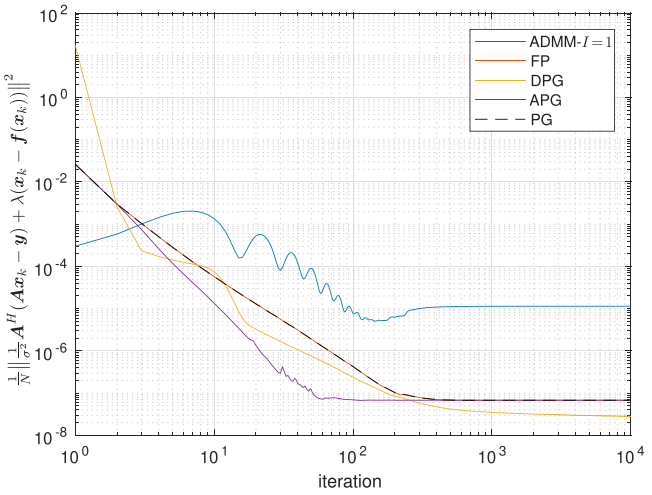

Fig. 8. Fixed-point error versus iteration for RED algorithms with TDT denoising when deblurring the starfish.

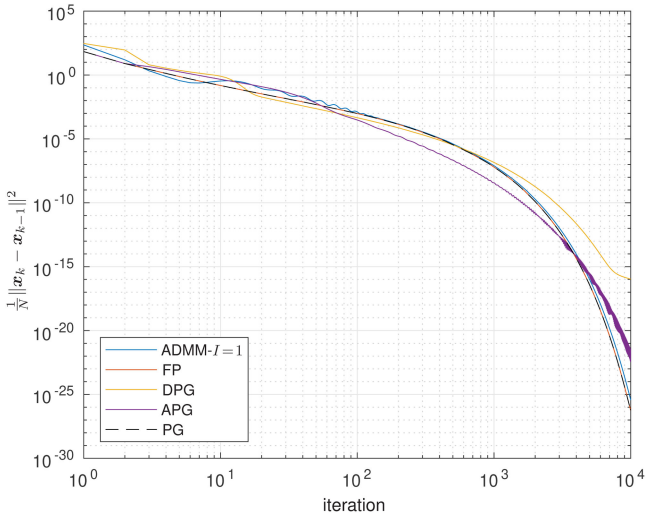

Fig. 9. Update distance versus iteration for RED algorithms with TDT denoising when deblurring the starfish.

Fig. 7 shows PSNR versus iteration with TDT denoising. Interestingly, the final PSNR values appear to be nearly identical among all algorithms under test, but more than 1 dB worse than the values around iteration 20. Fig. 8 shows the fixed-point error vs. iteration for this experiment. There, the errors of most algorithms converge to a value near 10^{-7}, but then remain at that value. Noting that RED-PG satisfies the conditions of Theorem 2 (i.e., convex loss, non-expansive denoiser, $L > 1$), it should converge to a fixed-point of (15). Therefore, we attribute the fixed-point error saturation in Fig. 8 to issues with numerical precision. Fig. 9 shows the normalized distance versus iteration with TDT denoising. There, the distance decreases to zero for all algorithms under test.

We emphasize that the proposed RED-DPG, RED-APG, and RED-PG algorithms seek to solve exactly the same fixed-point equation (15) sought by the RED-SD, RED-ADMM, and RED-FP algorithms proposed in [1]. The excellent quality of the RED fixed-points was firmly established in [1], both qualitatively and quantitatively, in comparison to existing state-of-the-art methods like PnP-ADMM [10]. For further details on these comparisons, including examples of images recovered by the RED algorithms, we refer the interested reader to [1].

VI. Equilibrium View of RED Algorithms

Like the RED algorithms, PnP-ADMM [10] repeatedly calls a denoiser $\boldsymbol{f}(\cdot)$ in order to solve an inverse problem. In [9], Buzzard, Sreehari, and Bouman show that PnP-ADMM finds a "consensus equilibrium" solution rather than a minimum of any explicit cost function. By consensus equilibrium, we mean a solution $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{u}})$ to

$$\widehat{\boldsymbol{x}} = F(\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{u}}) \tag{97a}$$

$$\widehat{\boldsymbol{x}} = G(\widehat{\boldsymbol{x}} - \widehat{\boldsymbol{u}}) \tag{97b}$$

for some functions $F, G : \mathbb{R}^N \to \mathbb{R}^N$. For PnP-ADMM, these functions are [9]

$$F_{\mathsf{pnp}}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\beta}{2}\|\boldsymbol{x}-\boldsymbol{v}\|^2 \right\} \tag{98}$$

$$G_{\mathsf{pnp}}(\boldsymbol{v}) = \boldsymbol{f}(\boldsymbol{v}). \tag{99}$$

A. RED Equilibrium Conditions

We now show that the RED algorithms also find consensus equilibrium solutions, but with $G \neq G_{\mathsf{pnp}}$. First, recall ADMM Algorithm 1 with explicit regularization $\rho(\cdot)$. By taking iteration $k \to \infty$, it becomes clear that the ADMM solutions must satisfy the equilibrium condition (97) with

$$F_{\mathsf{admm}}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\beta}{2}\|\boldsymbol{x}-\boldsymbol{v}\|^2 \right\} \tag{100}$$

$$G_{\mathsf{admm}}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left\{ \lambda\rho(\boldsymbol{x}) + \frac{\beta}{2}\|\boldsymbol{x}-\boldsymbol{v}\|^2 \right\}, \tag{101}$$

where we note that $F_{\mathsf{admm}} = F_{\mathsf{pnp}}$.

The RED-ADMM algorithm can be considered as a special case of ADMM Algorithm 1 under which $\rho(\cdot)$ is differentiable with $\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$, for a given denoiser $\boldsymbol{f}(\cdot)$. We can thus find $G_{\mathsf{red\text{-}admm}}(\cdot)$, i.e., the RED-ADMM version of $G(\cdot)$ satisfying the equilibrium condition (97b), by solving the right side of (101) under $\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$. Similarly, we see that the RED-ADMM version of $F(\cdot)$ is identical to the ADMM version of $F(\cdot)$ from (100). Now, the $\widehat{\boldsymbol{x}} = G_{\mathsf{red\text{-}admm}}(\boldsymbol{v})$ that solves the right side of (101) under differentiable $\rho(\cdot)$ with $\nabla\rho(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})$ must obey

$$\boldsymbol{0} = \lambda\nabla\rho(\widehat{\boldsymbol{x}}) + \beta(\widehat{\boldsymbol{x}} - \boldsymbol{v}) \tag{102}$$

$$= \lambda\big(\widehat{\boldsymbol{x}} - \boldsymbol{f}(\widehat{\boldsymbol{x}})\big) + \beta(\widehat{\boldsymbol{x}} - \boldsymbol{v}), \tag{103}$$

which we note is a special case of (15). Continuing, we find that

$$\boldsymbol{0} = \lambda\big(\widehat{\boldsymbol{x}} - \boldsymbol{f}(\widehat{\boldsymbol{x}})\big) + \beta(\widehat{\boldsymbol{x}} - \boldsymbol{v}) \tag{104}$$

$$\Leftrightarrow \boldsymbol{0} = \frac{\lambda+\beta}{\beta}\widehat{\boldsymbol{x}} - \frac{\lambda}{\beta}\boldsymbol{f}(\widehat{\boldsymbol{x}}) - \boldsymbol{v} \tag{105}$$

$$\Leftrightarrow \boldsymbol{v} = \left(\frac{\lambda+\beta}{\beta}\boldsymbol{I} - \frac{\lambda}{\beta}\boldsymbol{f}\right)(\widehat{\boldsymbol{x}}) \tag{106}$$

$$\Leftrightarrow \widehat{\boldsymbol{x}} = \left(\frac{\lambda+\beta}{\beta}\boldsymbol{I} - \frac{\lambda}{\beta}\boldsymbol{f}\right)^{-1}(\boldsymbol{v}) = G_{\mathsf{red\text{-}admm}}(\boldsymbol{v}), \tag{107}$$

where $\boldsymbol{I}$ represents the identity operator and $(\cdot)^{-1}$ represents the functional inverse. In summary, RED-ADMM with denoiser $\boldsymbol{f}(\cdot)$ solves the consensus equilibrium problem (97) with $F = F_{\mathsf{admm}}$ from (100) and $G = G_{\mathsf{red\text{-}admm}}$ from (107).

Next we establish an equilibrium result for RED-PG. Defining $\boldsymbol{u}_k = \boldsymbol{v}_k - \boldsymbol{x}_k$ and taking $k \to \infty$ in Algorithm 4, it can be seen that the fixed points of RED-PG obey (97a) for

$$F_{\mathsf{red\text{-}pg}}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\|\boldsymbol{x}-\boldsymbol{v}\|^2 \right\}. \tag{108}$$

Furthermore, from line 3 of Algorithm 4, it can be seen that the RED-PG fixed points also obey

$$\widehat{\boldsymbol{u}} = \frac{1}{L}\left(\boldsymbol{f}(\widehat{\boldsymbol{x}}) - \widehat{\boldsymbol{x}}\right) \tag{109}$$

$$\Leftrightarrow \widehat{\boldsymbol{x}} - \widehat{\boldsymbol{u}} = \widehat{\boldsymbol{x}} - \frac{1}{L}\left(\boldsymbol{f}(\widehat{\boldsymbol{x}}) - \widehat{\boldsymbol{x}}\right) \tag{110}$$

$$= \left(\frac{L+1}{L}\boldsymbol{I} - \frac{1}{L}\boldsymbol{f}\right)(\widehat{\boldsymbol{x}}) \tag{111}$$

$$\Leftrightarrow \widehat{\boldsymbol{x}} = \left(\frac{L+1}{L}\boldsymbol{I} - \frac{1}{L}\boldsymbol{f}\right)^{-1}(\widehat{\boldsymbol{x}} - \widehat{\boldsymbol{u}}), \tag{112}$$

which matches (97b) when $G = G_{\mathsf{red\text{-}pg}}$ for

$$G_{\mathsf{red\text{-}pg}}(\boldsymbol{v}) = \left(\frac{L+1}{L}\boldsymbol{I} - \frac{1}{L}\boldsymbol{f}\right)^{-1}(\boldsymbol{v}). \tag{113}$$

Note that $G_{\mathsf{red\text{-}pg}} = G_{\mathsf{red\text{-}admm}}$ when $L = \beta/\lambda$.

B. Interpreting the RED Equilibria

The equilibrium conditions provide additional interpretations of the RED algorithms. To see how, first recall that the RED equilibrium $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{u}})$ satisfies

$$\widehat{\boldsymbol{x}} = F_{\mathsf{red\text{-}pg}}(\widehat{\boldsymbol{x}} + \widehat{\boldsymbol{u}}) \tag{114a}$$

$$\widehat{\boldsymbol{x}} = G_{\mathsf{red\text{-}pg}}(\widehat{\boldsymbol{x}} - \widehat{\boldsymbol{u}}), \tag{114b}$$

or an analogous pair of equations involving $F_{\mathsf{red\text{-}admm}}$ and $G_{\mathsf{red\text{-}admm}}$. Thus, from (108), (109), and (114a), we have that

$$\widehat{\boldsymbol{x}} = F_{\mathsf{red\text{-}pg}}\left(\widehat{\boldsymbol{x}} + \frac{1}{L}(\boldsymbol{f}(\widehat{\boldsymbol{x}}) - \widehat{\boldsymbol{x}})\right) \tag{115}$$

$$= F_{\mathsf{red\text{-}pg}}\left(\frac{L-1}{L}\widehat{\boldsymbol{x}} + \frac{1}{L}\boldsymbol{f}(\widehat{\boldsymbol{x}})\right) \tag{116}$$

$$= \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda L}{2}\left\|\boldsymbol{x} - \frac{L-1}{L}\widehat{\boldsymbol{x}} - \frac{1}{L}\boldsymbol{f}(\widehat{\boldsymbol{x}})\right\|^2 \right\}. \tag{117}$$

When $L = 1$, this simplifies down to

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{f}(\widehat{\boldsymbol{x}})\|^2 \right\}. \tag{118}$$

Note that (118) is reminiscent of, although in general not equivalent to,

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \left\{ \ell(\boldsymbol{x};\boldsymbol{y}) + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x})\|^2 \right\}, \tag{119}$$

which was discussed as an "alternative" formulation of RED in [1, Sec. 5.2].

Insights into the relationship between RED and PnP-ADMM can be obtained by focusing on the simple case of

$$\ell(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{2\sigma^2}\|\boldsymbol{x}-\boldsymbol{y}\|^2, \tag{120}$$

where the overall goal of variational image recovery would be the denoising of $\boldsymbol{y}$. For PnP-ADMM, (90) and (98) imply

$$F_{\mathsf{pnp}}(\boldsymbol{v}) = \frac{1}{1+\lambda\sigma^2}\boldsymbol{y} + \frac{\lambda\sigma^2}{1+\lambda\sigma^2}\boldsymbol{v}, \tag{121}$$

and so the equilibrium condition (97a) implies

$$\widehat{\boldsymbol{x}}_{\mathsf{pnp}} = \frac{1}{1+\lambda\sigma^2}\boldsymbol{y} + \frac{\lambda\sigma^2}{1+\lambda\sigma^2}(\widehat{\boldsymbol{x}}_{\mathsf{pnp}} + \widehat{\boldsymbol{u}}_{\mathsf{pnp}}) \tag{122}$$

$$\Leftrightarrow \widehat{\boldsymbol{u}}_{\mathsf{pnp}} = \frac{\widehat{\boldsymbol{x}}_{\mathsf{pnp}} - \boldsymbol{y}}{\lambda\sigma^2}. \tag{123}$$

Meanwhile, (99) and the equilibrium condition (97b) imply

$$\widehat{\boldsymbol{x}}_{\mathsf{pnp}} = \boldsymbol{f}(\widehat{\boldsymbol{x}}_{\mathsf{pnp}} - \widehat{\boldsymbol{u}}_{\mathsf{pnp}}) \tag{124}$$

$$= \boldsymbol{f}\left(\frac{\lambda\sigma^2-1}{\lambda\sigma^2}\widehat{\boldsymbol{x}}_{\mathsf{pnp}} + \frac{1}{\lambda\sigma^2}\boldsymbol{y}\right). \tag{125}$$

In the case that $\lambda = 1/\sigma^2$, we have the intuitive result that

$$\widehat{\boldsymbol{x}}_{\mathsf{pnp}} = \boldsymbol{f}(\boldsymbol{y}), \tag{126}$$

which corresponds to direct denoising of $\boldsymbol{y}$. For RED, $\widehat{\boldsymbol{u}}_{\mathsf{red}}$ is algorithm dependent, but $\widehat{\boldsymbol{x}}_{\mathsf{red}}$ is always the solution to (15), where now $\boldsymbol{A} = \boldsymbol{I}$ due to (120). That is,

$$\boldsymbol{y} - \widehat{\boldsymbol{x}}_{\mathsf{red}} = \lambda\sigma^2\big(\widehat{\boldsymbol{x}}_{\mathsf{red}} - \boldsymbol{f}(\widehat{\boldsymbol{x}}_{\mathsf{red}})\big). \tag{127}$$

Taking $\lambda = 1/\sigma^2$ for direct comparison to (126), we find

$$\boldsymbol{y} - \widehat{\boldsymbol{x}}_{\mathsf{red}} = \widehat{\boldsymbol{x}}_{\mathsf{red}} - \boldsymbol{f}(\widehat{\boldsymbol{x}}_{\mathsf{red}}). \tag{128}$$

Thus, whereas PnP-ADMM reports the denoiser output $\boldsymbol{f}(\boldsymbol{y})$, *RED reports the* $\widehat{\boldsymbol{x}}$ *for which the denoiser residual* $\boldsymbol{f}(\widehat{\boldsymbol{x}}) - \widehat{\boldsymbol{x}}$ *negates the measurement residual* $\boldsymbol{y} - \widehat{\boldsymbol{x}}$. This $\widehat{\boldsymbol{x}}$ can be expressed concisely as

$$\widehat{\boldsymbol{x}} = (2\boldsymbol{I} - \boldsymbol{f})^{-1}(\boldsymbol{y}) = G_{\mathsf{red\text{-}pg}}(\boldsymbol{y})\big|_{L=1}. \tag{129}$$

VII. Conclusion

The RED paper [1] proposed a powerful new way to exploit plug-in denoisers when solving imaging inverse-problems. In fact, experiments in [1] suggest that the RED algorithms are state-of-the-art. Although [1] claimed that the RED algorithms minimize an optimization objective containing an explicit regularizer of the form $\rho_{\mathsf{red}}(\boldsymbol{x}) \triangleq \frac{1}{2}\boldsymbol{x}^\top(\boldsymbol{x} - \boldsymbol{f}(\boldsymbol{x}))$ when the denoiser is LH, we showed that the denoiser must also be Jacobian symmetric for this explanation to hold. We then provided extensive numerical evidence that practical denoisers like the median filter, non-local means, BM3D, TNRD, or DnCNN lack sufficient Jacobian symmetry. Furthermore, we established that, with non-JS denoisers, the RED algorithms cannot be explained by explicit regularization of any form.

None of our negative results dispute the fact that the RED algorithms work very well in practice. But they do motivate the need for a better understanding of RED. In response, we showed that the RED algorithms can be explained by a novel framework called *score-matching by denoising* (SMD), which aims to match the "score" (i.e., the gradient of the log-prior) rather than design any explicit regularizer. We then established tight connections between SMD, kernel density estimation, and constrained MMSE denoising.

On the algorithmic front, we provided new interpretations of the RED-ADMM and RED-FP algorithms proposed in [1], and we proposed novel RED algorithms with much faster convergence. Finally, we performed a consensus-equilibrium analysis of the RED algorithms that lead to additional interpretations of RED and its relation to PnP-ADMM.

Acknowledgment

The authors thank P. Milanfar, M. Elad, G. Buzzard, and C. Bouman for insightful discussions.

References

[1] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imag. Sci.*, vol. 10, no. 4, pp. 1804–1844, 2017.

[2] A. Buades, B. Coll, and J.-M. Morel, "A review of image denoising algorithms, with a new one," *Multiscale Model. Simul.*, vol. 4, no. 2, pp. 490–530, 2005.

[3] P. Milanfar, "A tour of modern image filtering: New insights and methods, both practical and theoretical," *IEEE Signal Process. Mag.*, vol. 30, no. 1, pp. 106–128, Jan. 2013.

[4] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, Jun. 2017.

[5] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3142–3155, Jul. 2017.

[6] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, vol. 2, pp. 60–65.

[7] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[8] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, vol. 33, no. 3, pp. 1065–1076, 1962.

[9] G. T. Buzzard, S. H. Chan, S. Sreehari, and C. A. Bouman, "Plug-and-play unplugged: Optimization-free reconstruction using consensus equilibrium," *SIAM J. Imag. Sci.*, vol. 11, no. 3, pp. 2001–2020, 2018.

[10] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Inf. Process.*, 2013, pp. 945–948.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2007.

[12] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[13] S. Ono, "Primal-dual plug-and-play image restoration," *IEEE Signal Process. Lett.*, vol. 24, no. 8, pp. 1108–1112, Aug. 2017.

[14] U. Kamilov, H. Mansour, and B. Wohlberg, "A plug-and-play priors approach for solving nonlinear imaging inverse problems," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1872–1876, May 2017.

[15] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci.*, vol. 106, pp. 18914–18919, Nov. 2009.

[16] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Trans. Inf. Theory*, vol. 57, no. 2, pp. 764–785, Feb. 2011.

[17] S. Som and P. Schniter, "Compressive imaging using approximate message passing and a Markov-tree prior," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3439–3448, Jul. 2012.

[18] D. L. Donoho, I. M. Johnstone, and A. Montanari, "Accurate prediction of phase transitions in compressed sensing via a connection to minimax denoising," *IEEE Trans. Inf. Theory*, vol. 59, no. 6, pp. 3396–3433, Jun. 2013.

[19] C. A. Metzler, A. Maleki, and R. G. Baraniuk, "From denoising to compressed sensing," *IEEE Trans. Inf. Theory*, vol. 62, no. 9, pp. 5117–5144, Sep. 2016.

[20] S. Rangan, P. Schniter, and A. K. Fletcher, "Vector approximate message passing," in *Proc. IEEE Int. Symp. Inf. Theory*, 2017, pp. 1588–1592.
[21] P. Schniter, S. Rangan, and A. K. Fletcher, "Denoising-based vector approximate message passing," in *Proc. Int. Biomed. Astron. Signal Process. Frontiers Workshop*, 2017, 77 pages.
[22] R. Berthier, A. Montanari, and P.-M. Nguyen, "State evolution for approximate message passing with non-separable functions," arXiv:1708.03950, 2017.
[23] A. K. Fletcher, S. Rangan, S. Sarkar, and P. Schniter, "Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis," in *Proc. Neural Inf. Process. Syst. Conf.*, to be published, 2018.
[24] T. S. Huang, G. J. Yang, and Y. T. Tang, "A fast two-dimensional median filtering algorithm," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-27, no. 1, pp. 13–18, Feb. 1979.
[25] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1976.
[26] S. Kantorovitz, *Several Real Variables*. New York, NY, USA: Springer, 2016.
[27] S. Sreehari *et al.*, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. Comput. Imag.*, vol. 2, no. 4, pp. 408–423, Dec. 2016.
[28] D. L. Donoho and I. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.
[29] R. R. Coifman and D. L. Donoho, "Translation-invariant de-noising," in *Wavelets and Statistics*, A. Antoniadis and G. Oppenheim, Eds. New York, NY, USA: Springer, 1995, pp. 125–150.
[30] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 3, no. 1, pp. 123–231, 2013.
[31] F. Ong, P. Milanfar, and P. Getreurer, "Local kernels that approximate Bayesian regularization and proximal operators," arXiv:1803.03711, 2018.
[32] A. Teodoro, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Scene-adapted plug-and-play algorithm with guaranteed convergence: Applications to data fusion in imaging," arXiv:1801.00605, 2018.
[33] P. Milanfar, "Symmetrizing smoothing filters," *SIAM J. Imag. Sci.*, vol. 30, no. 1, pp. 263–284, 2013.
[34] P. Milanfar and H. Talebi, "A new class of image filters without normalization," in *Proc. IEEE Int. Conf. Image Process.*, 2016, pp. 3294–3298.
[35] T. Goldstein, C. Studer, and R. Baraniuk, "Forward-backward splitting with a FASTA implementation," arXiv:1411.3406, 2014.
[36] H. Robbins, "An empirical Bayes approach to statistics," in *Proc. Berkeley Symp. Math. Statist. Prob.*, 1956, pp. 157–163.
[37] B. Efron, "Tweedie's formula and selection bias," *J. Amer. Statist. Assoc.*, vol. 106, no. 496, pp. 1602–1614, 2011.
[38] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *J. Mach. Learn. Res.*, vol. 6, pp. 695–709, 2005.
[39] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, 2011.
[40] C. M. Stein, "Estimation of the mean of a multivariate normal distribution," *Ann. Statist.*, vol. 9, pp. 1135–1151, 1981.
[41] F. Luisier, "The SURE-LET approach to image denoising," Ph.D. thesis, EPFL, Lausanne, Switzerland, 2010.
[42] M. Raphan and E. P. Simoncelli, "Least squares estimation without priors or supervision," *Neural Comput.*, vol. 23, pp. 374–420, Feb. 2011.
[43] T. Blu and F. Luisier, "The SURE-LET approach to image denoising," *IEEE Trans. Image Process.*, vol. 16, no. 11, pp. 2778–2786, Nov. 2007.
[44] S. A. Bigdeli and M. Zwicker, "Image restoration using autoencoding priors," arXiv:1703.09964, 2017.
[45] G. Alain and Y. Bengio, "What regularized auto-encoders learn from the data-generating distribution," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 3563–3593, 2014.
[46] M. A. Ranzato, Y.-L. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," in *Proc. Neural Inf. Process. Syst. Conf.*, 2008, pp. 1185–1192.
[47] A. Beck and M. Teboulle, "Gradient-based algorithms with applications to signal recovery," in *Convex Optimization in Signal Processing and Communications*, D. P. Palomar and Y. C. Eldar, Eds. Cambridge, U.K.: Cambridge Univ. Press, 2009, pp. 42–88.
[48] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, H. Bauschke, R. Burachik, P. Combettes, V. Elser, D. Luke, and H. Wolkowicz, Eds. New York, NY, USA: Springer, 2011, pp. 185–212.
[49] Y. Sun, P. Babu, and D. P. Palomar, "Majorization-minimization algorithms in signal processing, communications, and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 3, pp. 794–816, Feb. 2017.
[50] M. A. T. Figueiredo, J. M. Bioucas-Dias, and R. D. Nowak, "Majorization-minimization algorithms for wavelet-based image restoration," *IEEE Trans. Image Process.*, vol. 16, no. 12, pp. 2980–2991, Dec. 2007.
[51] C. A. Metzler, P. Schniter, A. Veeraraghavan, and R. G. Baraniuk, "prDeep: Robust phase retrieval with flexible deep neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3501–3510.
[52] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.
[53] A. Sidi, *Vector Extrapolation Methods With Applications*. Philadelphia, PA, USA: SIAM, 2017.
[54] T. Hong, Y. Romano, and M. Elad, "Acceleration of RED via vector extrapolation," arXiv:1805:02158, 2018.
[55] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics, 1st ed. New York, NY, USA: Springer, 2011.
[56] Y. Sun, B. Wohlberg, and U. S. Kamilov, "An online plug-and-play algorithm for regularized image reconstruction," arXiv:1809.04693, 2018.

Edward T. Reehorst received the B.S. degree in electrical engineering from The Ohio State University, Columbus, OH, USA, in 2016. He is currently working toward the Ph.D. degree in electrical and computer engineering with The Ohio State University. He has completed several internship experiences with the NASA Glenn Research Center, Cleveland OH, USA, between 2012 and 2016, including the NASA Academy and Space communication and navigation Internship Program (SIP). His research interests are in imaging and machine learning.

Philip Schniter (S'92–M'93–SM'05–F'14) received the B.S. and M.S. degrees in electrical engineering from the University of Illinois at Urbana-Champaign, Champaign, IL, USA, in 1992 and 1993, respectively, and the Ph.D. degree in electrical engineering from Cornell University, Ithaca, NY, USA, in 2000. From 1993 to 1996, he was with Tektronix Inc., Beaverton, OR, USA, as a Systems Engineer. After receiving the Ph.D. degree, he joined the Department of Electrical and Computer Engineering, Ohio State University, Columbus, OH, USA, where he is currently a Professor. In 2008–2009, he was a Visiting Professor with Eurecom, Sophia Antipolis, France, and with Supelec, Gif-sur-Yvette, France. In 2016–2017, he was a Visiting Professor with Duke University, Durham, NC, USA. His areas of interests currently include signal processing, wireless communications, and machine learning. In 2002, he was the recipient of the NSF CAREER Award, in 2016, the IEEE Signal Processing Society Best Paper Award, and in 2018, the Qualcomm Faculty Award. He currently serves on the IEEE Sensor Array and Multichannel Technical Committee and the IEEE Computational Imaging Technical Committee.