

# Learned convex regularizers for inverse problems

Subhadip Mukherjee<sup>1</sup>, Sören Dittmer<sup>2</sup>, Zakhar Shumaylov<sup>1</sup>, Sebastian Lunz<sup>1</sup>, Ozan Öktem<sup>3</sup>, and Carola-Bibiane Schönlieb<sup>1</sup>

**Abstract**—We consider the variational reconstruction framework for inverse problems and propose to learn a data-adaptive input-convex neural network (ICNN) as the regularization functional. The ICNN-based convex regularizer is trained adversarially to discern ground-truth images from unregularized reconstructions. **Convexity of the regularizer is desirable since (i) one can establish analytical convergence guarantees for the corresponding variational reconstruction problem and (ii) devise efficient and provable algorithms for reconstruction.** In particular, we show that the optimal solution to the variational problem converges to the ground-truth if the penalty parameter decays sub-linearly with respect to the norm of the noise. Further, we prove the existence of a sub-gradient-based algorithm that leads to a monotonically decreasing error in the parameter space with iterations. **To demonstrate the performance of our approach for solving inverse problems, we consider the tasks of deblurring natural images and reconstructing images in computed tomography (CT), and show that the proposed convex regularizer is at least competitive with and sometimes superior to state-of-the-art data-driven techniques for inverse problems.**

**Index Terms**—Inverse problems, data-driven convex regularization, adversarial learning.

## I. INTRODUCTION

INVERSE problems arise in numerous scientific applications, e.g., in virtually every modern medical imaging modality, wherein the key objective is to estimate some parameters of interest based on an indirect and possibly noisy measurement.

An inverse problem is said to be ill-posed if it has no or multiple solutions, or if its solution is not continuous in the measurement. Many traditional approaches attempt to alleviate the issue of ill-posedness by involving hand-crafted prior information on possible reconstructions.

While such analytical priors usually lead to provable properties, they fall short in terms of data-adaptability; i.e., it is impossible to define formally, which, of all possible images, are *natural images*. In recent years, several solutions to this problem emerged with the rise of deep learning methods [1]. While the deep learning-based techniques often produce reconstructions of astonishingly high quality, they typically lack many, if not all, of the provable properties that the traditional variational approaches offer. In this work, we propose an approach that integrates deep learning into the classical regularization theory by learning a strongly convex regularizer to incorporate prior information into a variational reconstruction setting. Before explaining our contributions in more detail, we provide a brief overview of related works.

### A. Related works

Fully data-driven approaches for inverse problems aim to either map the measurement directly to the model parameter [2], or remove artifacts from an analytical reconstruction method [3] by training an over-parametrized neural network using pairs of appropriate input and target images. Such end-to-end fully trained methods are data-intensive and might generalize poorly if they are trained on limited amount of examples.

Another recent data-driven approach for inverse problems consists in *unrolling* of iterative model-based methods [4]–[7]. Iterative unrolling techniques are supervised and incorporate the forward operator in the learning model to achieve data-efficiency (thereby generalizing well from moderate amount of data), setting them apart from fully-trained [2] or post-processing-based approaches [3].

The idea of learning a sparsity-promoting data-driven regularizer has been a prominent research direction in recent years, the origin of which can be traced back to the pioneering work by Aharon et al. [8]. **The key concept in [8] was to learn a sparsifying synthesis dictionary leading to a parsimonious signal representation in terms of only a few dictionary elements. Sparsity in the learned basis can be promoted by penalizing the  $\ell_1$ -norm of the representation coefficients, resulting in a convex variational problem for a fixed pre-trained dictionary.** The linear synthesis sparsity model in [8] was subsequently extended to analysis- and transform-based linear sparsity models [9], [10]. **The surge of deep learning research led to a refinement of the classical sparsity models and attempts were made to draw connections between convolutional neural networks (CNNs) and multi-layer convolutional sparse modeling (see [11] and references therein).**

One of the recent approaches for solving inverse problems is based on the idea of using a denoiser inside an algorithm for minimizing the variational objective [12], [13]. The regularization by denoising (RED) algorithm proposed in [12] belongs to this class, wherein one constructs an explicit regularizer from an image denoiser by penalizing the inner product of the image with its denoising residual. Contrary to the claim in [12], it was shown in [14, Sec. III.G] that such a construction may fail to produce a convex regularizer even when the Jacobian matrix of the denoiser is symmetric and its eigenvalues lie in  $[0, 1]$ . However, it is argued in [14] that the RED algorithm in [12] does not necessarily minimize the variational loss arising from the RED-regularizer. Therefore, instead of relying on convexity of the regularizer, [14] develops a new framework based on *score-matching by denoising* (SMD) for the convergence analysis of the RED algorithm. The *regularization by artifact-removal* (RARE) approach proposed in [15] aims to learn an

<sup>1</sup>: Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK; emails: {sm2467, zs334, sl767, cbs31}@cam.ac.uk. <sup>2</sup>: Center for Industrial Mathematics, University of Bremen, Germany; email: sdittmer@math.uni-bremen.de. <sup>3</sup>: Department of Mathematics, KTH – Royal Institute of Technology, Sweden; email: ozan@kth.se.

artifact-removing CNN as the denoiser in the RED framework by using pairs of undersampled measurements. Although it might be possible, in principle, to learn the artifact-removing CNN in such a way that the resulting RED regularizer is convex, the constraints that one has to impose on the denoising CNN during training to ensure convexity of the regularizer are not straightforward to derive, especially in view of the analysis presented in [14]. A fixed-point convergence analysis of proximal gradient and alternating directions method of multipliers (ADMM) was developed in [16] by replacing the proximal operator with a plug-and-play (PnP) denoising operator of the form  $\mathcal{D} = \gamma \mathcal{I} + (1 - \gamma) \mathcal{D}_1$ , where  $\gamma \in (0, 1)$ ,  $\mathcal{I}$  denotes the identity operator, and  $\mathcal{D}_1$  is non-expansive. PnP methods are equivalent to regularized image reconstruction in spirit, since the PnP denoiser corresponds to an implicit prior.

The line of research that we build on directly uses a neural network to parametrize the regularization functional, allowing to reconstruct from the observed data by solving a variational optimization problem [17]–[20]. More specifically, our work relies upon the philosophy of learning an adversarial regularizer (AR) introduced in [17], which is parametrized using a neural network. Aside from [17], the idea of using a trained neural network as a regularizer was considered in [18] (referred to as network Tikhonov (NETT)) and more recently in [20] (referred to as total deep variation (TDV)). The key difference between the AR, and the TDV and NETT approaches lies in the training protocol for the regularizer. While the AR is trained with an objective to discriminate desired images from noisy ones, the NETT regularizer rests upon an encoder-decoder setup and the TDV approach trains the regularizer by differentiating through the minimization of the variational problem, similar to the unrolling schemes discussed before [4]. In all cases (AR, NETT, and TDV), gradient-descent is used for solving the variational problem for reconstruction and no convergence guarantees that match the classical results for convex regularizers can be derived in these settings. Further, since a penalty term may be interpreted as the inclusion of prior information, the idea of incorporating the prior knowledge by restricting the reconstruction to lie in the range of a generative model proposed in [21] is also closely related to our work. Under smoothness conditions, this approach can be shown to be equivalent to learning a penalty function via a Lagrangian argument [22].

In this work, we leverage strong convexity by generalizing the parametrization of input-convex neural networks (ICNNs) proposed in [23] while designing the regularizer. This not only allows us to show well-posedness and strong convergence of the variational reconstruction but also guides the development of a convergent sub-gradient descent algorithm for reconstruction. We state the specific contributions in the following subsection.

### B. Specific contributions

This work builds upon [17] that introduces the adversarial regularizer (AR) framework. The idea in AR is to replace a hand-crafted regularizer with a learned one, parametrized by a deep neural network. The parametric regularizer is first trained to separate ground-truth images from images containing artifacts. The loss function for training seeks to maximize

the output of AR for noisy images, while minimizing it for the ground-truth images. When the regularizer is constrained to be 1-Lipschitz, the optimal training loss corresponds to the Wasserstein distance [24] between the distributions of the ground-truth and that of the noisy images. One does not need paired images to approximate the training objective in AR, which makes the framework unsupervised in theory. Subsequently, the trained AR is deployed in a variational scheme for solving an ill-posed inverse problem.

To the best of our knowledge, this work makes the first attempt to enforce convexity on the learned regularizer by restricting the architecture in order to combine deep learning with convex regularization theory. This helps us establish stronger convergence results (proposition 3) and derive a precise stability estimate (Proposition 2) as opposed to the AR framework. Further, by exploiting convexity, we prove the existence of a convergent sub-gradient algorithm for minimizing the variational objective (Lemma 1).

By utilizing ICNNs [23], one can rigorously study regularizing properties of such variational schemes even when the regularizer is learned. The convergence guarantees and ICNN parametrization developed by us are applicable in the general setting where the underlying parameters and the measurement belong to Hilbert spaces. The resulting data-driven adversarial convex regularizer (ACR) offers reconstruction performance that is competitive with similar learned methods while being provably convergent simultaneously. In specific applications, we found that a convex regularizer even outperforms its non-convex variant, especially when the training dataset is small, and the forward operator is severely ill-conditioned.

## II. BACKGROUND ON INVERSE PROBLEMS

We denote function spaces by blackboard-bold letters (e.g.  $\mathbb{X}$ ) and functionals (mapping function spaces to the real line) by calligraphic letters (e.g.  $\mathcal{R}$ ). Generic elements of a space are denoted by boldface lowercase letters (e.g.  $\mathbf{x} \in \mathbb{X}$ ). Simple uppercase letters are used to denote random variables, e.g., a generic  $\mathbb{X}$ -valued random variable is denoted as  $X$ , with its distribution being denoted as  $\pi_X$ .

### A. Classical formulation

Inverse problems deal with reconstructing unknown model parameter  $\mathbf{x}^* \in \mathbb{X}$  from the indirect measurement

$$\mathbf{y}^\delta = \mathcal{A}(\mathbf{x}^*) + \mathbf{e} \in \mathbb{Y}, \quad (1)$$

where  $\mathcal{A} : \mathbb{X} \rightarrow \mathbb{Y}$  is the forward operator and  $\mathbf{e} \in \mathbb{Y}$ ,  $\|\mathbf{e}\|_2 \leq \delta$ , denotes measurement noise. Here  $\mathbb{X}$  and  $\mathbb{Y}$  are Hilbert spaces containing possible model parameters and data, respectively.

In the context of medical imaging, e.g., computed tomography (CT), the model parameter  $\mathbf{x}^* \in \mathbb{X}$  is the image of the interior structure one seeks to recover. The measurement  $\mathbf{y}^\delta \in \mathbb{Y}$  (data) represents indirect observations of  $\mathbf{x}^*$ .

In the classical function-analytic formulation,  $\mathbf{x}^*$  is modeled as deterministic and the standard practice is to approximate it from  $\mathbf{y}^\delta$  by solving a variational reconstruction problem:

$$\min_{\mathbf{x} \in \mathbb{X}} \mathcal{L}_{\mathbb{Y}}(\mathbf{y}^\delta, \mathcal{A}(\mathbf{x})) + \alpha \mathcal{R}(\mathbf{x}). \quad (2)$$

The loss functional  $\mathcal{L}_{\mathbb{Y}} : \mathbb{Y} \times \mathbb{Y} \rightarrow \mathbb{R}$  provides a measure of data-fidelity and is typically chosen based on the statistical properties of the measurement noise  $e$ , whereas the regularization functional  $\mathcal{R} : \mathbb{X} \rightarrow \mathbb{R}$  penalizes undesirable images. The penalty parameter  $\alpha > 0$  trades-off data-fidelity with the regularization penalty and is chosen depending on the noise strength. In the subsequent part, we consider the squared- $\ell_2$  loss to measure data-fidelity, i.e.,  $\mathcal{L}_{\mathbb{Y}}(\mathbf{y}_1, \mathbf{y}_2) = \|\mathbf{y}_1 - \mathbf{y}_2\|_{\mathbb{Y}}^2$ , unless otherwise specified, and correspondingly, (2) reduces to

$$\min_{\mathbf{x} \in \mathbb{X}} \|\mathbf{y}^\delta - \mathcal{A}(\mathbf{x})\|_{\mathbb{Y}}^2 + \alpha \mathcal{R}(\mathbf{x}). \quad (3)$$

### B. Statistical formulation

In the statistical formulation of inverse problems, one models the data as a single sample  $\mathbf{y}^\delta$  of the  $\mathbb{Y}$ -valued random variable

$$Y = \mathcal{A}(X) + e, \quad (4)$$

and aims to estimate the posterior distribution of  $X$  conditioned on  $Y = \mathbf{y}^\delta$ , denoted as  $\pi_{\text{post}}(X = \mathbf{x} | Y = \mathbf{y}^\delta)$ . Using the Bayes rule, the posterior distribution can be expressed in terms of the data-likelihood and the prior:

$$\pi_{\text{post}}(X = \mathbf{x} | Y = \mathbf{y}^\delta) = \frac{\pi_{\text{data}}(Y = \mathbf{y}^\delta | X = \mathbf{x}) \pi_X(X = \mathbf{x})}{Z(\mathbf{y}^\delta)}, \quad (5)$$

where  $Z(\mathbf{y}^\delta)$  is a normalizing constant independent of  $\mathbf{x}$ . While the data-likelihood is known in most inverse problems, the prior, which encodes a-priori belief about  $\mathbf{x}$ , is typically unknown. In the sequel, we write  $\pi_{\text{post}}(X = \mathbf{x} | Y = \mathbf{y}^\delta)$  as  $\pi_{\text{post}}(\mathbf{x} | \mathbf{y}^\delta)$ , and likewise for the other probability measures in (5) for simplicity. An approximation of the true image is typically obtained by summarizing the posterior distribution into a point-estimate, such as the mean. Among many choices available for extracting a point-estimate from the posterior, a particularly popular one is to compute the *mode*, leading to the so-called *maximum a-posteriori probability* (MAP) estimate:

$$\min_{\mathbf{x} \in \mathbb{X}} -\log \pi_{\text{data}}(\mathbf{y}^\delta | \mathbf{x}) - \log \pi_X(\mathbf{x}). \quad (6)$$

For a Gibbs-type prior  $\pi_X(\mathbf{x}) \propto \exp(-\lambda \mathcal{R}(\mathbf{x}))$ , the MAP estimation problem (6) is essentially equivalent to the variational reconstruction framework (2) in the classical setting.

## III. THEORETICAL RESULTS

In this section, we will first prove that the minimizer of the variational loss resulting from a strongly convex regularizer converges to the ground-truth. Subsequently, we develop a parametrization strategy using a neural network that achieves the properties required for convergence. We then explain the training protocol and describe the steps involved in learning the convex regularizer in a data-driven manner.

### A. Properties of strongly-convex regularizers

First, we demonstrate that the variational problem corresponding to a strongly-convex regularizer is well-posed, and derive the resulting convergence and stability estimates. The analysis serves as a motivation for the parametrization of the

regularizer discussed in the next section. In particular, we begin by investigating the analytical properties of a regularization functional of the form

$$\mathcal{R}(\mathbf{x}) = \mathcal{R}'(\mathbf{x}) + \rho_0 \|\mathbf{x}\|_{\mathbb{X}}^2, \quad (7)$$

where  $\|\cdot\|_{\mathbb{X}}$  is the norm induced by the inner product structure of  $\mathbb{X}$  and  $\mathcal{R}' : \mathbb{X} \rightarrow \mathbb{R}$  is 1-Lipschitz and convex in  $\mathbf{x}$ . No smoothness assumption is made on  $\mathcal{R}'(\mathbf{x})$ . The corresponding reconstruction problem consists in minimizing the variational objective  $J_\alpha(\mathbf{x}; \mathbf{y}^\delta)$  with respect to  $\mathbf{x}$ , where

$$J_\alpha(\mathbf{x}; \mathbf{y}) := \|\mathbf{y} - \mathcal{A}(\mathbf{x})\|_{\mathbb{Y}}^2 + \alpha \left( \mathcal{R}'(\mathbf{x}) + \rho_0 \|\mathbf{x}\|_{\mathbb{X}}^2 \right). \quad (8)$$

The forward operator is assumed to be bounded and linear, so its operator norm is bounded, i.e.,

$$\beta_1 := \sup_{\mathbf{x} \in \mathbb{X}} \frac{\|\mathcal{A}(\mathbf{x})\|_{\mathbb{Y}}}{\|\mathbf{x}\|_{\mathbb{X}}} < \infty.$$

In the following, we formally show the well-posedness of the variational problem corresponding to the objective in (8). Specifically, we show that the variational objective  $J_\alpha(\mathbf{x}; \mathbf{y})$  has a unique minimizer  $\hat{\mathbf{x}}_\alpha(\mathbf{y})$ , for any  $\mathbf{y}$  and  $\alpha > 0$ , varying continuously in the data  $\mathbf{y}$ . Further, when the noise level  $\delta \rightarrow 0$ , the minimizer of the variational loss  $J_\alpha(\mathbf{x}; \mathbf{y}^\delta)$  approaches the  $\mathcal{R}$ -minimizing solution  $\mathbf{x}^\dagger$  given by

$$\mathbf{x}^\dagger \in \arg \min_{\mathbf{x}} \mathcal{R}(\mathbf{x}) \text{ subject to } \mathcal{A}(\mathbf{x}) = \mathbf{y}^0, \quad (9)$$

where  $\mathbf{y}^0$  is the clean data. Convergence to  $\mathbf{x}^\dagger$  defined in (9) holds provided that the regularization penalty  $\alpha(\delta)$  is chosen appropriately as a function of  $\delta$ . These results can be derived as special cases from the general convex regularization theory [25, Theorems 3.22, 3.23, and 3.26; Proposition 3.32]. Here, we present their proofs, which, by directly using strong convexity, are simpler than their more general counterparts. That is why we include them here to make the theoretical treatment self-contained. An interested reader must refer to [25, Chapter 3] for more general proofs that do not require strong-convexity.

**Proposition 1.** (*Existence and uniqueness*)  $J_\alpha(\mathbf{x}; \mathbf{y})$  is strongly convex in  $\mathbf{x}$  with parameter  $2\alpha\rho_0$  and has a unique minimizer  $\hat{\mathbf{x}}_\alpha(\mathbf{y})$  for every  $\mathbf{y}$  and  $\alpha > 0$ . We also have

$$J_\alpha(\mathbf{x}; \mathbf{y}) \geq J_\alpha(\hat{\mathbf{x}}_\alpha(\mathbf{y}); \mathbf{y}) + \alpha\rho_0 \|\mathbf{x} - \hat{\mathbf{x}}_\alpha(\mathbf{y})\|_{\mathbb{X}}^2, \quad (10)$$

for any  $\mathbf{x} \in \mathbb{X}$ .

**Proof:** It follows from the definition of strong convexity that  $h_\mu(\mathbf{x}) = h(\mathbf{x}) + \frac{\mu}{2} \|\mathbf{x}\|_{\mathbb{X}}^2$  is  $\mu$ -strongly convex when  $h$  is convex. Since  $\|\mathbf{y} - \mathcal{A}(\mathbf{x})\|_{\mathbb{Y}}^2 + \alpha \mathcal{R}'(\mathbf{x})$  is convex when  $\mathcal{A}$  is linear, it follows that  $J_\alpha(\mathbf{x}; \mathbf{y})$  is  $2\alpha\rho_0$ -strongly convex in  $\mathbf{x}$ , and consequently, for any  $\mathbf{x}, \mathbf{v} \in \mathbb{X}$ , we have that

$$J_\alpha(\mathbf{x}; \mathbf{y}) \geq J_\alpha(\mathbf{v}; \mathbf{y}) + \langle (\mathbf{x} - \mathbf{v}), \mathbf{g}_{\mathbf{v}} \rangle + \alpha\rho_0 \|\mathbf{x} - \mathbf{v}\|_{\mathbb{X}}^2, \quad (11)$$

for all  $\mathbf{g}_{\mathbf{v}} \in \partial J_\alpha(\mathbf{v}; \mathbf{y})$ . In particular, when  $\mathbf{v} = \hat{\mathbf{x}}$  is a minimizer of  $J_\alpha(\cdot; \mathbf{y})$ , we have  $\mathbf{0} \in \partial J_\alpha(\mathbf{v}; \mathbf{y})$ , and therefore (11) leads to

$$J_\alpha(\mathbf{x}; \mathbf{y}) \geq J_\alpha(\hat{\mathbf{x}}; \mathbf{y}) + \alpha\rho_0 \|\mathbf{x} - \hat{\mathbf{x}}\|_{\mathbb{X}}^2. \quad (12)$$

(12) also ascertains that if there are two minimizers  $\hat{x}_1$  and  $\hat{x}_2$ , one must have  $\hat{x}_1 = \hat{x}_2$ , thereby guaranteeing uniqueness. The unique minimizer, denoted as  $\hat{x}_\alpha(\mathbf{y})$ , satisfies (10). ■

**Proposition 2. (Stability)** *The optimal solution  $\hat{x}_\alpha(\mathbf{y})$  is continuous in  $\mathbf{y}$ .*

**Proof:** Denote a perturbation of magnitude  $\delta_1$  on  $\mathbf{y}$  as

$$\mathbf{y}^{\delta_1} = \mathbf{y} + \delta_1, \text{ with } \|\delta_1\|_{\mathbb{Y}} \leq \delta_1.$$

Define for any  $\delta_1 > 0$

$$p_{\delta_1} := J_\alpha(\hat{x}_\alpha(\mathbf{y}^{\delta_1}); \mathbf{y}) - J_\alpha(\hat{x}_\alpha(\mathbf{y}^{\delta_1}); \mathbf{y}^{\delta_1}).$$

Clearly,  $\lim_{\delta_1 \rightarrow 0} p_{\delta_1} = 0$  since  $J_\alpha(\mathbf{x}; \mathbf{y})$  is continuous in  $\mathbf{y}$  for any  $\mathbf{x} \in \mathbb{X}$ . Further,  $p_{\delta_1}$  can be expressed as

$$\begin{aligned} p_{\delta_1} &= [J_\alpha(\hat{x}_\alpha(\mathbf{y}^{\delta_1}); \mathbf{y}) - J_\alpha(\hat{x}_\alpha(\mathbf{y}); \mathbf{y})] \\ &+ [J_\alpha(\hat{x}_\alpha(\mathbf{y}); \mathbf{y}) - J_\alpha(\hat{x}_\alpha(\mathbf{y}); \mathbf{y}^{\delta_1})] \\ &+ [J_\alpha(\hat{x}_\alpha(\mathbf{y}); \mathbf{y}^{\delta_1}) - J_\alpha(\hat{x}_\alpha(\mathbf{y}^{\delta_1}); \mathbf{y}^{\delta_1})]. \end{aligned} \quad (13)$$

For convenience, denote the terms within square brackets in (13) as  $t_1$ ,  $t_2$ , and  $t_3$ , respectively. By Proposition 1,

$$t_1, t_3 \geq \alpha \rho_0 \|\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})\|_{\mathbb{X}}^2,$$

and by continuity of  $J_\alpha(\cdot; \mathbf{y})$  in  $\mathbf{y}$ , we have  $\lim_{\delta_1 \rightarrow 0} t_2 = 0$ . Therefore  $\lim_{\delta_1 \rightarrow 0} (p_{\delta_1} - t_2) = 0$ , and (13) implies that

$$p_{\delta_1} - t_2 = t_1 + t_3 \geq 2\alpha \rho_0 \|\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})\|_{\mathbb{X}}^2. \quad (14)$$

Further, we have that

$$\begin{aligned} p_{\delta_1} - t_2 &= \|\mathcal{A}(\hat{x}_\alpha(\mathbf{y}^{\delta_1})) - \mathbf{y}\|_{\mathbb{Y}}^2 - \|\mathcal{A}(\hat{x}_\alpha(\mathbf{y}^{\delta_1})) - \mathbf{y}^{\delta_1}\|_{\mathbb{Y}}^2 \\ &+ \|\mathcal{A}(\hat{x}_\alpha(\mathbf{y})) - \mathbf{y}^{\delta_1}\|_{\mathbb{Y}}^2 - \|\mathcal{A}(\hat{x}_\alpha(\mathbf{y})) - \mathbf{y}\|_{\mathbb{Y}}^2. \end{aligned} \quad (15)$$

Now, substituting  $\mathbf{y}^{\delta_1} = \mathbf{y} + \delta_1$  in (15) and expanding further,

$$\begin{aligned} p_{\delta_1} - t_2 &= 2\langle \delta_1, \mathcal{A}(\hat{x}_\alpha(\mathbf{y}^{\delta_1})) \rangle - \|\delta_1\|_{\mathbb{Y}}^2 \\ &- 2\langle \delta_1, \mathcal{A}(\hat{x}_\alpha(\mathbf{y})) \rangle + \|\delta_1\|_{\mathbb{Y}}^2 \\ &= 2\langle \delta_1, \mathcal{A}(\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})) \rangle \\ &\leq 2\beta_1 \delta_1 \|\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})\|_{\mathbb{X}}, \end{aligned} \quad (16)$$

where the last inequality in (16) is due to Cauchy-Schwarz. Finally, combining (14) with (16), we have

$$\|\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})\|_{\mathbb{X}} \leq \frac{\beta_1 \delta_1}{\alpha \rho_0}. \quad (17)$$

(17) indicates that  $\lim_{\delta_1 \rightarrow 0} \|\hat{x}_\alpha(\mathbf{y}^{\delta_1}) - \hat{x}_\alpha(\mathbf{y})\|_{\mathbb{X}} = 0$ , confirming that  $\hat{x}_\alpha(\mathbf{y})$  is continuous in  $\mathbf{y}$  for a fixed  $\alpha$ . ■

Unlike Theorem 3 in [17], Proposition 2 establishes convergence of  $\hat{x}_\alpha(\mathbf{y}^{(k)}) = \arg \min_{\mathbf{x}} J_\alpha(\mathbf{x}; \mathbf{y}^{(k)})$  to  $\hat{x}_\alpha(\mathbf{y})$  with respect to the norm topology  $\|\cdot\|_{\mathbb{X}}$  on  $\mathbb{X}$  when  $\|\mathbf{y}^{(k)} - \mathbf{y}\|_{\mathbb{Y}} \rightarrow 0$ , and (17) yields a stability estimate, in addition.

**Proposition 3. (Convergence)** *For  $\delta \rightarrow 0$  and  $\alpha(\delta) \rightarrow 0$  such that  $\frac{\delta}{\alpha(\delta)} \rightarrow 0$ , we have that  $\hat{x}_\alpha(\mathbf{y}^\delta)$  converges to the  $\mathcal{R}$ -minimizing solution  $\mathbf{x}^\dagger$  given in (9).*

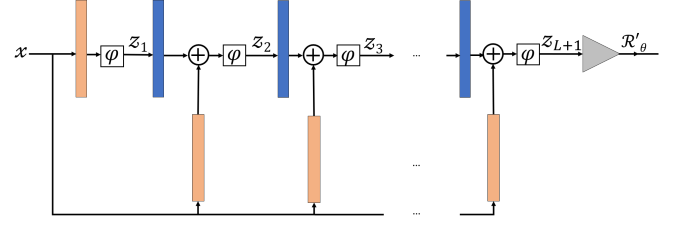


Fig. 1. Architecture of the convex regularizer. The blue rectangles indicate the layers  $\{\mathcal{B}_i\}_{i=1}^L$  that are allowed to contain only non-negative weights for convexity of the output, while the rectangles in orange denote layers with weights and biases  $\{\mathcal{W}_i, \mathbf{b}_i\}_{i=0}^L$  which are allowed to take any real value. The triangle in the end represents a global average-pooling layer. The activations  $\varphi_i$  in all layers are chosen to be leaky-ReLU and denoted by  $\varphi$ .

**Proof:** By (17), we have that

$$\|\hat{x}_\alpha(\mathbf{y}^\delta) - \hat{x}_\alpha(\mathbf{y}^0)\|_{\mathbb{X}} \leq \frac{\beta_1 \delta}{\alpha \rho_0}. \quad (18)$$

Since  $\mathcal{A}$  is linear and  $\mathcal{R}$  is strongly-convex, the solution  $\mathbf{x}^\dagger$  to (9) is unique and it can alternatively be expressed as

$$\begin{aligned} \mathbf{x}^\dagger &= \lim_{\alpha \rightarrow 0+} \left( \arg \min_{\mathbf{x}} \mathcal{R}(\mathbf{x}) + \frac{1}{\alpha} \|\mathcal{A}(\mathbf{x}) - \mathbf{y}^0\|_{\mathbb{Y}}^2 \right) \\ &= \lim_{\alpha \rightarrow 0+} \left( \arg \min_{\mathbf{x}} \alpha \mathcal{R}(\mathbf{x}) + \|\mathcal{A}(\mathbf{x}) - \mathbf{y}^0\|_{\mathbb{Y}}^2 \right) \\ &= \lim_{\alpha \rightarrow 0+} \hat{x}_\alpha(\mathbf{y}^0). \end{aligned} \quad (19)$$

Let  $\epsilon(\alpha) = \|\hat{x}_\alpha(\mathbf{y}^0) - \mathbf{x}^\dagger\|_{\mathbb{X}}$ . Then, by (19),  $\lim_{\alpha \rightarrow 0} \epsilon(\alpha) = 0$ . Thus, combining (18) and (19), and using the triangle inequality, one can argue that

$$\begin{aligned} \|\hat{x}_\alpha(\mathbf{y}^\delta) - \mathbf{x}^\dagger\|_{\mathbb{X}} &\leq \|\hat{x}_\alpha(\mathbf{y}^\delta) - \hat{x}_\alpha(\mathbf{y}^0)\|_{\mathbb{X}} \\ &+ \|\hat{x}_\alpha(\mathbf{y}^0) - \mathbf{x}^\dagger\|_{\mathbb{X}} \leq \frac{\beta_1 \delta}{\alpha \rho_0} + \epsilon(\alpha). \end{aligned}$$

Now, if  $\lim_{\delta \rightarrow 0} \alpha(\delta) \rightarrow 0$  and  $\lim_{\delta \rightarrow 0} \frac{\delta}{\alpha(\delta)} \rightarrow 0$ , the inequality above implies that  $\lim_{\delta \rightarrow 0} \|\hat{x}_\alpha(\mathbf{y}^\delta) - \mathbf{x}^\dagger\|_{\mathbb{X}} = 0$ . ■

## B. Input-convex neural networks

This section introduces ICNNs in the infinite dimensional setting, which boils down to the construction in [23] in the finite dimensional case. In particular, we need to implement the convex functional  $\mathcal{R}'$  in (7) on  $\mathbb{L}^2$  spaces, i.e.  $\mathbb{X} = \mathbb{L}^2$ . Discretizing this parameterized operator coincides with the convex network construction presented in [23]. More precisely, define the activation-spaces of our network to be  $\mathbb{V}_i := \mathbb{L}^2([0, 1]^{n_i})$  for  $i = 0, \dots, L-1$ , with  $\mathbb{V}_0 = \mathbb{X}$  and  $\mathbb{V}_L = \mathbb{R}$ . We assume the input  $\mathbf{x}$  of our network to be in  $\mathbb{V}_0$  and set  $\mathbf{0} := \mathbf{z}_0 \in \mathbb{V}_0$ . We then define the output of each layer  $i = 0, \dots, L$  to be

$$\mathbf{z}_{i+1}(\mathbf{x}) = \varphi_i(\mathcal{B}_i(\mathbf{z}_i(\mathbf{x}))) + \mathcal{W}_i(\mathbf{x}) + \mathbf{b}_i. \quad (20)$$

Here  $\mathcal{B}_i : \mathbb{V}_i \rightarrow \mathbb{V}_{i+1}$  and  $\mathcal{W}_i : \mathbb{V}_0 \rightarrow \mathbb{V}_{i+1}$  are bounded integral transforms and we assume the kernels of the  $\mathcal{B}_i$ 's to be pointwise non-negative. The activations  $\varphi_i : \mathbb{V}_{i+1} \rightarrow \mathbb{V}_{i+1}$  are given by the pointwise application of a convex monotone function from  $\mathbb{R}$  to  $\mathbb{R}$ . Further let the biases,  $\mathbf{b}_i$ , be in  $\mathbb{V}_{i+1}$ . The overall output of the network  $\mathcal{R}' : \mathbb{V}_0 \rightarrow \mathbb{R}$



---

**Algorithm 1** Training the ACR via (21).

---

**1. Input:** gradient penalty  $\lambda_{\text{gp}}$ , initial value of the network parameters  $\theta^{(0)} = \{\mathcal{B}_i \geq 0, \mathcal{W}_i, \mathbf{b}_i\}$ , mini-batch size  $n_b$ , parameters  $(\eta, \beta_1, \beta_2)$  for the Adam optimizer.

**2. for mini-batches**  $m = 1, 2, \dots$ , **do (until convergence):**

- Sample  $\mathbf{x}_j \sim \pi_X$ ,  $\mathbf{y}_j \sim \pi_Y$ , and  $\epsilon_j \sim \text{uniform}[0, 1]$ ; for  $1 \leq j \leq n_b$ . Compute  $\mathbf{x}_j^{(\epsilon)} = \epsilon_j \mathbf{x}_j + (1 - \epsilon_j) \mathcal{A}^\dagger \mathbf{y}_j$ .
- Compute the training loss for the  $m^{\text{th}}$  mini-batch:

$$\begin{aligned} \mathcal{L}(\theta) := & \frac{1}{n_b} \sum_{j=1}^{n_b} \mathcal{R}_\theta(\mathbf{x}_j) - \frac{1}{n_b} \sum_{j=1}^{n_b} \mathcal{R}_\theta(\mathcal{A}^\dagger \mathbf{y}_j) \\ & + \lambda_{\text{gp}} \cdot \frac{1}{n_b} \sum_{j=1}^{n_b} \left( \left\| \nabla \mathcal{R}_\theta \left( \mathbf{x}_j^{(\epsilon)} \right) \right\|_2 - 1 \right)^2. \end{aligned}$$

- Update  $\theta^{(m)} = \text{Adam}_{\eta, \beta_1, \beta_2}(\theta^{(m-1)}, \nabla_\theta \mathcal{L}(\theta^{(m-1)}))$ .
  - Zero-clip the negative weights in  $\mathcal{B}_i$  to preserve convexity.
- 3. Output:** Parameter  $\theta$  of the trained ACR.
- 

is obtained by applying a global-average pooling operator  $\mathcal{H}_{\text{avg}}$  on  $\mathbf{z}_{L+1}$ , i.e.,  $\mathcal{R}'(\mathbf{x}) := \mathcal{H}_{\text{avg}}(\mathbf{z}_{L+1}(\mathbf{x}))$ . In practice,  $\{\mathcal{B}_i, \mathcal{W}_i\}$ 's represent convolutional layers. The combined set of parameters  $\{\mathcal{B}_i, \mathcal{W}_i, \mathbf{b}_i\}_{i=0}^L$  of the convolutional layers in  $\mathcal{R}'$  is denoted by the shorthand notation  $\theta$  and the corresponding parametric regularizer is written as  $\mathcal{R}'_\theta$ . A schematic diagram of the architecture of  $\mathcal{R}'_\theta$  is shown in Figure 1.

The convexity of  $\mathcal{R}'_\theta(\mathbf{x})$  in  $\mathbf{x}$  can be argued using a recursive logic which uses the following two facts [26]:

- Non-negative combination of finitely many convex functions is another convex function, and
- the composition  $\psi_1 \circ \psi_2 : \mathbb{V} \rightarrow \mathbb{R}$  of two functions  $\psi_1$  and  $\psi_2$  is convex when  $\psi_2$  is convex and  $\psi_1$  is convex and monotonically non-decreasing.

We note that  $\mathbf{z}_1(\mathbf{x}) = \varphi_0(\mathcal{W}_0(\mathbf{x}) + \mathbf{b}_0)$  is convex in  $\mathbf{x}$ , since  $\mathbf{z}_1$  is the composition of a convex and monotonically non-decreasing activation  $\varphi_0$  (leaky-ReLU, in particular) with an affine (and hence convex) function of  $\mathbf{x}$ . Further, when  $\mathbf{z}_i$  is convex and the weights in  $\mathcal{B}_i$  are non-negative, the argument of  $\varphi_i$  in (20) is convex in  $\mathbf{x}$ , and so is  $\mathbf{z}_{i+1}$ , since  $\varphi_i$  is convex and monotonically non-decreasing (also chosen to be leaky-ReLU activation). Consequently,  $\mathbf{z}_{L+1}$  is convex in  $\mathbf{x}$  and so is the final output  $\mathcal{R}'_\theta$ , since the average-pooling operator  $\mathcal{H}_{\text{avg}}$  maintains convexity. To preserve convexity during training, one has to ensure that the weights in  $\mathcal{B}_i$  remain non-negative throughout training. This is achieved by applying a zero-clipping operation on the negative weights following each weight update.

The specific choice of the hyper-parameters in the architecture in the discretized implementation, such as the number of layers, kernel size, etc., is mentioned in the context of the specific experiments in Sec. V.

### C. Learning objective and protocol

The training objective and protocol adopted for ACR follow the same philosophy introduced in [17] for training AR. The regularizer is trained with the objective of favoring solutions

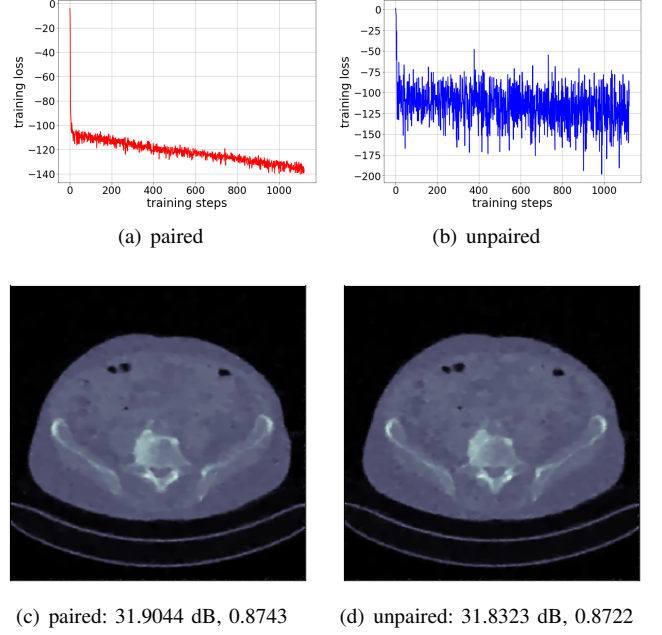


Fig. 2. Training ACR on paired vs. unpaired data for sparse-view CT, where each training step corresponds to update over 20 mini-batches. Training on paired data is more stable. The reconstruction produced by ACR trained on paired data is nearly indistinguishable from ACR trained on unpaired data, both qualitatively and in terms of objective measures of reconstruction quality, such as the peak signal-to-noise ratio (PSNR) (measured in dB) and the structural similarity index (SSIM).

that are similar to the ground-truth images in the training dataset and penalizing reconstructions with artifacts. Consequently, it should produce a small output when a true image is given as input and a large output when it is presented with an unregularized reconstruction. The distribution of unregularized reconstructions is denoted as  $\mathcal{A}^\dagger_{\#} \pi_Y$ , the push-forward of the measurement distribution  $\pi_Y$  by the pseudo-inverse of the forward operator. Naturally, during training, one seeks to minimize the average output of the regularizer  $\mathcal{R}_\theta$  when the input is sampled from the true image distribution  $\pi_X$ , while simultaneously maximizing the average output of  $\mathcal{R}_\theta$  when the input is an unregularized reconstruction. This is equivalent to minimizing the difference of the average output of  $\mathcal{R}_\theta$  over the true image distribution  $\pi_X$  and the distribution  $\mathcal{A}^\dagger_{\#} \pi_Y$  of unregularized solutions and can be posed as

$$\begin{aligned} \theta^* = & \arg \min_{\theta} \left( \mathbb{E}_{\pi_X} [\mathcal{R}_\theta(X)] - \mathbb{E}_{\mathcal{A}^\dagger_{\#} \pi_Y} [\mathcal{R}_\theta(X)] \right) \\ & \text{subject to } \mathcal{R}_\theta \in 1 - \text{Lipschitz}. \end{aligned} \quad (21)$$

The 1-Lipschitz condition in (21) encourages the output of  $\mathcal{R}_\theta$  to transition smoothly with respect to the input, thus making the corresponding variational loss stable. The 1-Lipschitz constraint is enforced by adding a gradient-penalty term [27] of the form

$$\mathcal{L}_{\text{gp}} = \lambda_{\text{gp}} \mathbb{E}_{\pi_{X^{(\epsilon)}}} \left[ \left( \left\| \nabla \mathcal{R}_\theta \left( X^{(\epsilon)} \right) \right\|_2 - 1 \right)^2 \right], \quad (22)$$

to the training objective in (21). Here,  $X^{(\epsilon)}$  is uniformly sampled on the line-segment between the images  $X$  and  $\mathcal{A}^\dagger Y$ . The actual steps involved in approximating the training

objective in (21) and imposing the gradient penalty (22) for training  $\mathcal{R}_\theta$  are listed in Algorithm 1.

The training protocol is unsupervised in principle, since it suffices to have  $N$  i.i.d. samples  $\{\mathbf{x}_i\}_{i=1}^N \in \mathbb{X}$  and  $\{\mathbf{y}_i\}_{i=1}^N \in \mathbb{Y}$  drawn from the marginal distributions  $\pi_X$  and  $\pi_Y$ , respectively, for approximating the loss in (21). Nevertheless, in practice, one has two options when it comes to estimating the loss in (21) by using empirical average over mini-batches: (i) using paired samples  $\{\mathbf{x}_i, \mathcal{A}^\dagger \mathbf{y}_i\}$ , where  $\mathbf{x}_i \sim \pi_X$  and

$$\mathbf{y}_i \sim \pi_{\text{data}}(\mathbf{y}_i | \mathbf{x}_i) = \pi_{\text{noise}}(\mathbf{y}_i - \mathcal{A}(\mathbf{x}_i)); \text{ and}$$

(ii) using unpaired samples  $\{\mathbf{x}_i, \mathcal{A}^\dagger \mathbf{y}_{(i)}\}$ , where the samples  $\{\mathbf{y}_{(i)}\}_{i=1}^N$  are obtained by applying a random permutation on  $\mathbf{y}_i \sim \pi_{\text{data}}(\mathbf{y}_i | \mathbf{x}_i)$ . The actual training Process would be supervised or unsupervised depending on whether one chooses the first or the second option, respectively. Notably, both options lead to identical outcomes in the limit of infinite amount of training data. However, for limited amount of data, we observe that supervised training leads to a more graceful evolution of the training loss with respect to training steps, while unsupervised training leads to considerably more fluctuations in the training loss. A comparison of the supervised and unsupervised approaches for training the ACR is shown in Fig. 2 for sparse-view CT reconstruction (see Sec. V-A for details). As the training objective evolves in a more stable manner when paired samples are used for approximation, we adopt a supervised learning process in the face of limited training data (similarly for AR as well to ensure a fair comparison of their performance). Nevertheless, the reconstruction metrics obtained using ACRs trained on paired and unpaired examples turn out to be nearly identical (see Figures 2(c) and 2(d)). A similar behavior during training and reconstruction was observed for the other two inverse problems considered in Section V, namely limited-view CT reconstruction and image deblurring.

#### IV. CONVERGENCE OF THE SUB-GRADIENT ALGORITHM

Although the regularizer is strongly-convex, it seems exceedingly difficult to come up with a closed-form expression for its proximal operator [28], thus preventing the use of proximal gradient-descent. We, however, show that a sub-gradient descent algorithm converges to the minimizer of the variational loss. We begin by rewriting the variational loss defined in (8) as

$$J_\alpha(\mathbf{x}; \mathbf{y}) = f_\alpha(\mathbf{x}; \mathbf{y}) + \alpha g(\mathbf{x}), \quad (23)$$

where  $f_\alpha(\mathbf{x}; \mathbf{y}) = \|\mathcal{A}(\mathbf{x}) - \mathbf{y}\|_{\mathbb{Y}}^2 + \alpha \rho_0 \|\mathbf{x}\|_{\mathbb{X}}^2$  is smooth, strongly-convex, and differentiable, while  $g(\mathbf{x}) = \mathcal{R}'(\mathbf{x})$  is convex and 1-Lipschitz. Note that  $g$  need not be differentiable. For brevity of notation, we drop the subscript  $\alpha$  in the loss functions and denote the Lipschitz constant of  $g$  by  $L_g$  for generality in the remainder of this section (the gradient penalty in Algorithm 1 enforces  $L_g$  to be close to one). Since the forward operator is linear,  $\nabla f(\cdot; \mathbf{y})$  is Lipschitz-continuous, with a Lipschitz constant which we denote as  $L_{\nabla f}$ .

The convergence of gradient-descent for convex optimization has been widely studied. To put our analysis into perspective and to outline the points of difference of our analysis with the classical convex optimization literature, we recall the following

well-known convergence results for gradient-descent applied to  $\min_{\mathbf{x}} J(\mathbf{x})$ :

- When  $J(\mathbf{x})$  is convex and has an  $L_{\nabla J}$ -Lipschitz gradient, gradient-descent with step-size  $\eta_k \leq \frac{2}{L_{\nabla J}}$  generates a sequence of iterates  $\mathbf{x}_k$  satisfying  $J(\mathbf{x}_k) - J(\hat{\mathbf{x}}) \leq \mathcal{O}(\frac{1}{k})$  [29, Thm. 2.1.13].
- When  $J(\mathbf{x})$  is  $\mu_J$ -strongly convex, with an  $L_{\nabla J}$ -Lipschitz gradient, one can show a linear convergence rate

$$\|\mathbf{x}_k - \hat{\mathbf{x}}\|_2^2 \leq \left( \frac{q_0 - 1}{q_0 + 1} \right)^k \|\mathbf{x}_0 - \hat{\mathbf{x}}\|_2^2,$$

for  $\eta_k \leq \frac{2}{\mu_J + L_{\nabla J}}$ , where  $q_0 = \frac{L_{\nabla J}}{\mu_J}$  [29, Thm. 2.1.14].

- When  $J$  is convex, non-smooth, the sub-gradients of  $J$  are bounded by  $L_J$  (i.e.,  $J$  is  $L_J$ -Lipschitz), the step-sizes satisfy  $\sum_{k=1}^{\infty} \eta_k^2 < \infty$ , and  $\sum_{k=1}^{\infty} \eta_k = \infty$ , the following convergence result holds for the sequence of iterates generated by the sub-gradient update rule (see [30, Sec. 3] for a proof):

$$\lim_{k \rightarrow \infty} J_{\text{best}}^{(k)} = J(\hat{\mathbf{x}}), \text{ where } J_{\text{best}}^{(k)} = \min_{0 \leq i \leq k} J(\mathbf{x}_i).$$

The structure of our variational loss in (23) is non-standard, in the sense that it does not fit into any of these classical results stated above. Therefore, we perform an independent analysis of the convergence of sub-gradient updates starting from the first principle, using the properties of  $J_\alpha$  in (23). More precisely, we establish a sub-linear rate of convergence of  $\|\mathbf{x}_k - \hat{\mathbf{x}}\|_2^2$  to 0, where  $\{\mathbf{x}_k\}_{k=1}^{\infty}$  is the sequence of updates generated by the sub-gradient algorithm.

The following lemma ascertains the existence of a step-size parameter that leads to a convergent sub-gradient algorithm. Nevertheless, in practice, we found that an appropriately chosen constant step-size works reasonably well.

**Lemma 1.** (Convergence of sub-gradient updates) Starting from any initial estimate  $\mathbf{x}_0$ , there exist optimal step-sizes

$$\eta_k^* = 2\alpha\rho_0 \frac{\|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2}{\|\mathbf{z}_k\|_{\mathbb{X}}^2}$$

such that the updates  $\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k^* \mathbf{z}_k$ , where  $\mathbf{z}_k = \nabla f(\mathbf{x}_k; \mathbf{y}) + \mathbf{u}_k$ , with  $\mathbf{u}_k \in \partial(\alpha g(\mathbf{x}_k))$ , converge to the minimizer  $\hat{\mathbf{x}}$  of  $J_\alpha(\mathbf{x}; \cdot)$  defined in (23), i.e.,  $\lim_{k \rightarrow \infty} \mathbf{x}_k = \hat{\mathbf{x}}$  in  $\mathbb{X}$  with respect to the norm topology.

**Proof:** Let  $e_k = \|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2$  be the squared estimation error in the  $k^{\text{th}}$  iteration, and let  $\mu = 2\alpha\rho_0$ . Omitting the argument  $\mathbf{y}$  in  $J(\mathbf{x}; \mathbf{y})$  for simplicity and using  $\mu$ -strong convexity of  $J(\mathbf{x})$ , we have

$$J(\mathbf{x}_k) \leq J(\hat{\mathbf{x}}) + \langle \mathbf{z}_k, \mathbf{x}_k - \hat{\mathbf{x}} \rangle - \frac{\mu}{2} \|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2. \quad (24)$$

From Proposition 1, we get  $J(\mathbf{x}_k) - J(\hat{\mathbf{x}}) \geq \frac{\mu}{2} \|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2$ , which, combined with (24), leads to

$$\langle \mathbf{z}_k, \mathbf{x}_k - \hat{\mathbf{x}} \rangle \geq \mu \|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2.$$

Therefore, we have the following bound on  $e_{k+1}$ :

$$\begin{aligned} e_{k+1} &= \|\mathbf{x}_{k+1} - \hat{\mathbf{x}}\|_{\mathbb{X}}^2 = \|\mathbf{x}_k - \eta_k \mathbf{z}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2 \\ &= \|\mathbf{x}_k - \hat{\mathbf{x}}\|_{\mathbb{X}}^2 - 2\eta_k \langle \mathbf{z}_k, \mathbf{x}_k - \hat{\mathbf{x}} \rangle + \eta_k^2 \|\mathbf{z}_k\|_{\mathbb{X}}^2 \\ &\leq (1 - 2\mu\eta_k) e_k + \eta_k^2 \|\mathbf{z}_k\|_{\mathbb{X}}^2. \end{aligned} \quad (25)$$

The bound in (25) becomes the tightest when

$$\eta_k = \eta_k^* = \frac{\mu e_k}{\|z_k\|_{\mathbb{X}}^2} = \mu \frac{\|x_k - \hat{x}\|_{\mathbb{X}}^2}{\|z_k\|_{\mathbb{X}}^2},$$

leading to  $e_{k+1} \leq e_k - \frac{\mu^2 e_k^2}{\|z_k\|_{\mathbb{X}}^2}$ . This indicates that the estimation error decreases monotonically with iteration. Further,  $\|z_k\|_{\mathbb{X}}^2$  can be bounded as

$$\begin{aligned} \|z_k\|_{\mathbb{X}}^2 &= \|\nabla f(x_k) + u_k\|_{\mathbb{X}}^2 \leq 2\|\nabla f(x_k)\|_{\mathbb{X}}^2 + 2\|u_k\|_{\mathbb{X}}^2 \\ &\leq 2\|\nabla f(x_k)\|_{\mathbb{X}}^2 + 2\alpha^2 L_g^2, \end{aligned} \quad (26)$$

as  $g$  is  $L_g$ -Lipschitz. Since  $\hat{x}$  is the unique minimizer of  $J(x)$ ,  $\exists \hat{u} \in \partial(\alpha g(\hat{x}))$  such that  $\nabla f(\hat{x}) + \hat{u} = 0$ . Using the triangle inequality and Lipschitz-continuity of  $\nabla f(x)$ ,

$$\begin{aligned} \|\nabla f(x_k)\|_{\mathbb{X}} - \|\hat{u}\|_{\mathbb{X}} &\leq \|\nabla f(x_k) + \hat{u}\|_{\mathbb{X}} \\ &= \|\nabla f(x_k) - \nabla f(\hat{x})\|_{\mathbb{X}} \\ &\leq L_{\nabla f} \|x_k - \hat{x}\|_{\mathbb{X}}, \end{aligned} \quad (27)$$

leading to  $\|\nabla f(x_k)\|_{\mathbb{X}} \leq L_{\nabla f} \sqrt{e_k} + \alpha L_g$ . Plugging this in (26) and substituting  $\|z_k\|_{\mathbb{X}}^2$  in (25) with its resulting upper bound, we arrive at

$$e_{k+1} \leq e_k - \frac{\mu^2 e_k^2}{2L_{\nabla f}^2 e_k + 4\alpha L_g L_{\nabla f} \sqrt{e_k} + 4\alpha^2 L_g^2}. \quad (28)$$

Finally, since  $0 \leq e_{k+1} < e_k$ ,  $\lim_{k \rightarrow \infty} e_k$  exists. Denote the limit by  $e$ . Taking limit as  $k \rightarrow \infty$  on both sides of (28), we get

$$e \leq e - \frac{\mu^2 e}{2L_{\nabla f}^2 e + 4\alpha L_g L_{\nabla f} \sqrt{e} + 4\alpha^2 L_g^2}, \quad (29)$$

implying that  $e = 0$ ; therefore proving  $\lim_{k \rightarrow \infty} x_k = \hat{x}$ . In other words, the sequence  $\{x_k\}_{k=1}^{\infty}$  converges to the unique optimal solution  $\hat{x}$  to the variational problem. ■

## V. NUMERICAL RESULTS

For performance evaluation and comparison with the state-of-the-art, we consider three applications, namely, CT reconstruction with (i) sparse-view and (ii) limited-angle projection, and (iii) natural image deblurring. For the CT experiments, human abdominal CT scans for 10 patients provided by Mayo Clinic for the low-dose CT grand challenge [31] are used. We simulate the projection data by using the ODL library [32] with a GPU-accelerated *astra* back-end. Our training dataset for the CT experiments consists of a total of 2250 2D slices, each of dimension  $512 \times 512$ , corresponding to 9 patients. The remaining 128 slices corresponding to one patient are used to evaluate the reconstruction performance. The deblurring experiments are conducted on the *STL-10* dataset [33]. In all cases, we measure the performance in terms of the peak signal-to-noise ratio (PSNR) and the structural similarity index (SSIM) [34]. The PSNR (in dB) is calculated as

$$\text{PSNR} = 10 \log_{10} \frac{x_{\max}^{*2}}{\frac{1}{P^2} \sum_{i,j=1}^P (x_{ij}^* - \hat{x}_{ij})^2},$$

where  $x_{\max}^* = 1.0$  is the maximum allowable intensity in our experiments (since the intensity values are normalized

methods	optimizer	$\alpha$	# iterations
AR	gradient-descent, $\eta = 0.5$	0.1	600
ACR	gradient-descent, $\eta = 0.8$	0.05	400

TABLE I  
THE OPTIMIZER SETTING FOR THE VARIATIONAL RECONSTRUCTION PROBLEMS CORRESPONDING TO AR AND ACR.

to  $[0, 1]$ ). Here,  $x^*$  and  $\hat{x}$  denote the ground-truth and the reconstructed images (of size  $P \times P$ ), respectively. In all settings, we compare our results also with total-variation (TV) reconstructions, computed by employing the ADMM-based solver in ODL. We choose the penalty parameter of each variational method to maximize the mean PSNR of its reconstructions. Besides the parametrization of ACR elucidated in Sec. III-B, we compare the performance of three simple variants of it. In the first variant, we incorporate an additional regularization term

$$\mathcal{R}'_{\text{sfb}}(x) = \|\mathcal{U}x\|_1, \quad (30)$$

where  $\mathcal{U}$  denotes a two-dimensional convolutional layer, into the regularizer. When the filters in  $\mathcal{U}$  have bounded norms, this additional term is also convex and Lipschitz-continuous, like  $\mathcal{R}'(x)$ . We refer to this term as *sparsifying filter-bank* (SFB) penalty, since the  $\ell_1$  penalty on  $\mathcal{U}x$  is tantamount to seeking sparsity of  $x$  in the learned filters  $\mathcal{U}$ . In other words, learning the filters  $\mathcal{U}$  in the SFB penalty is equivalent to learning a sparsifying convolutional transform proposed in [35], [36], albeit with a completely different training strategy. While the key characteristic of the adversarial learning strategy adopted in this paper is to tell apart true images from the noisy ones, traditional approaches for learning SFB-like priors entail minimizing an objective that comprises data representation error, sparsity prior, and suitable regularization terms on the transform to avoid degenerate solutions [10], [36]. The additional SFB term is motivated by the widely used wavelet-sparsity prior in image processing and essentially gives ACR a chance to learn a similar prior, and possibly a better one, during training. The other two variants correspond to only the ACR without any SFB term and the SFB term alone as a prior. We compare the performance of all three variants of ACR, namely with (i)  $\mathcal{R}'(x)$  and (ii)  $\mathcal{R}'(x) + \mathcal{R}'_{\text{sfb}}(x)$  as the Lipschitz convex part, and (iii)  $\mathcal{R}'_{\text{sfb}}(x) = \|\mathcal{U}x\|_1$  as the standalone Lipschitz-convex part to investigate if a simpler and more commonly used convex model indeed works on par or better than the ICNN model in Sec. III-B. While learning the SFB regularizer standalone or in combination with  $\mathcal{R}'(x)$ , we add an  $\ell_2$  penalty term on the SFB weights to control the Lipschitz constant of the resulting regularizer. For all of these variants of ACR, the same training strategy is adopted, as described in Sec. III-C.

The individual details of the experimental setups for all three applications are described in the following.

### A. Reconstruction in sparse-view CT

The sparse-view projection data is simulated in ODL using a parallel-beam acquisition geometry with 200 angles and

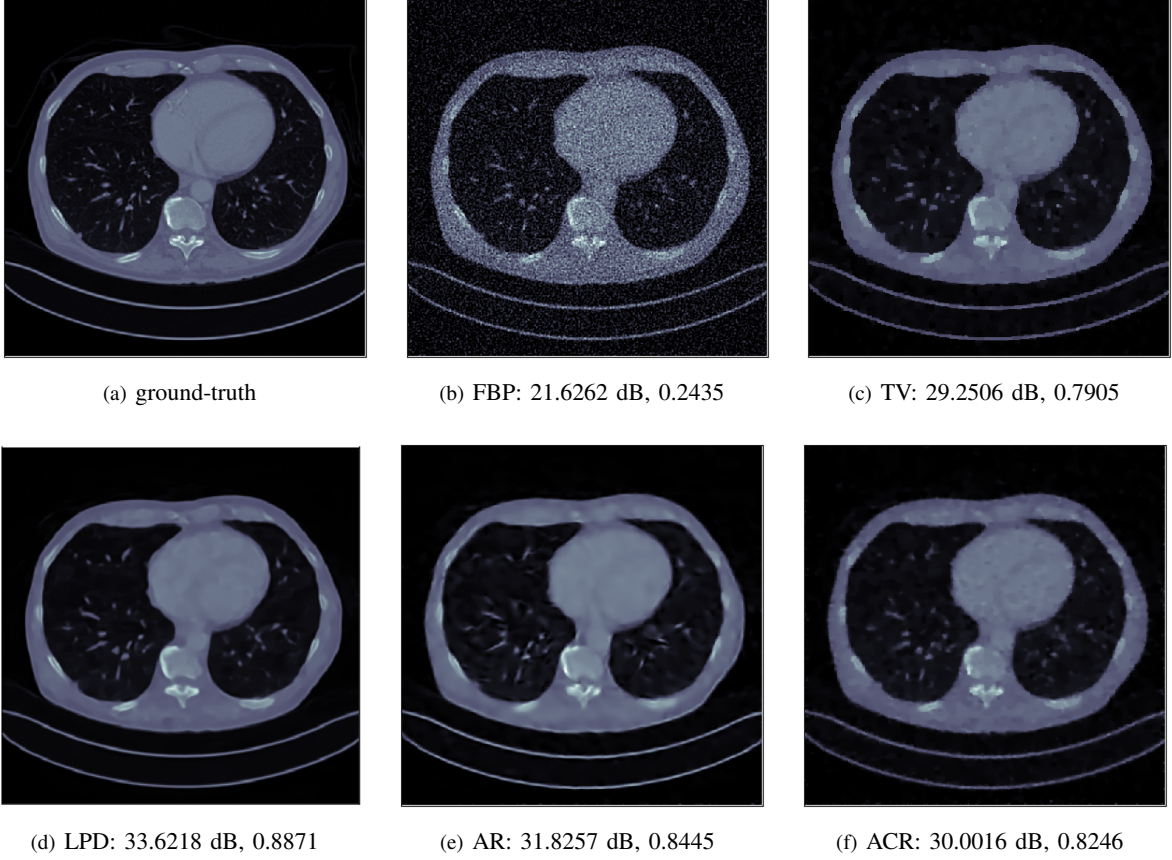


Fig. 3. Comparison of different reconstruction methods for sparse-view CT. The learned ACR approach outperforms the most widely used TV-based convex regularization. AR turns out to be better than ACR in this case, pointing to the limited expressive power of a convex regularizer. LPD trained on supervised data leads to the best performance.

methods	PSNR (dB)	SSIM	# param.	time (ms)
FBP	21.2866	0.2043	1	14.0
TV	30.3476	0.8110	1	21 315.0
LPD	35.1519	0.9073	1 138 720	184.4
FBP + U-Net	31.8008	0.7585	7 215 233	18.6
AR	33.6207	0.8750	19 347 890	41 058.0
ACR (no SFB)	31.2822	0.8468	590 928	242 414.3
ACR (with SFB)	31.4555	0.8644	606 609	245 366.5
SFB + $\rho_0 \ \mathbf{x}\ _2^2$	26.8001	0.5678	15 681	86 864.3

TABLE II  
AVERAGE PSNR, SSIM, AND RECONSTRUCTION TIME OVER TEST DATA FOR DIFFERENT RECONSTRUCTION METHODS FOR SPARSE-VIEW CT. THE NUMBER OF FREE PARAMETERS IN DIFFERENT RECONSTRUCTION TECHNIQUES, WHICH IS INDICATIVE OF THE COMPLEXITY OF THE MODEL, IS ALSO INDICATED.

400 rays/angle. White Gaussian noise with  $\sigma = 2.0$  is added to the projection data to simulate noisy measurement. The unregularized reconstructions  $\mathcal{A}^\dagger \mathbf{y}_i$  are taken as the output of the classical filtered back-projection method. The proposed ACR approach is compared with two model-based techniques, namely (i) FBP and (ii) TV regularization; and three data-driven methods, namely (i) the learned primal-dual (LPD)

method proposed in [5], (ii) the adversarial regularization (AR) approach introduced in [17], and the U-Net-based post-processing of FBP reconstruction [3]. The LPD method is trained on pairs of target image and projection data, while AR and the proposed ACR method are trained on ground-truth images and the corresponding FBP reconstructions. As explained before (Sec. III-C), using paired data for training ACR leads to a stable decay of the training loss as compared to the case where unpaired examples are used.

For LPD and AR, we develop *PyTorch* [37] implementations<sup>1</sup> based on their publicly available *TensorFlow* codes<sup>2</sup>. The LPD architecture was adapted to sparse-view projections by increasing the filter-size to  $5 \times 5$  and stacking 20 primal-dual layers (instead of  $3 \times 3$  filters and 10 layers, respectively, originally proposed in [5] for densely sampled projections) to increase its overall receptive field. This gives LPD a fair chance to reconstruct features that extend over a larger region in the image, in the face of sparse angular sampling. The optimizer setting used for AR and ACR while solving the variational problem is presented in Table I. The hyper-parameters in Table I are chosen empirically to maximize the reconstruction PSNR for our experiments.

<sup>1</sup>The ACR implementation is available upon request.

<sup>2</sup>LPD: [https://github.com/adler-j/learned\\_primal\\_dual](https://github.com/adler-j/learned_primal_dual), AR: <https://github.com/lunz-s/DeepAdversarialRegulariser>



The ACR architecture is constructed as described in Sec. III-B with  $L = 10$  layers.  $\mathcal{B}_i$  and  $\mathcal{W}_i$  are convolutional layers with  $5 \times 5$  kernels applied with a stride of 1 and consisting of 48 output channels. The layers  $\mathcal{B}_i$  are restricted to have non-negative weights to preserve convexity, while no such restriction is needed on  $\mathcal{W}_i$ 's. The activation at all layers except the final layer is chosen as the leaky-ReLU function with a negative slope of 0.2. The activation of the final layer is the identity function, and we apply a global average pooling on the output feature map to convert it to a scalar. The SFB term is composed of 10 2D convolution layers, each consisting of  $7 \times 7$  filters with 32 output channels. The *Adam* optimizer [38] is used for training the network, with a learning rate of  $2 \times 10^{-5}$  and  $(\beta_1, \beta_2) = (0.5, 0.99)$ . The gradient penalty term in (22) is chosen to be 5.0. The penalty parameter  $\rho_0$  corresponding to the squared- $\ell_2$  term in ACR is initialized at  $\rho_0 = \log(1 + \exp(-9.0))$  and then learned from training data. ACR and AR were trained for 5 and 10 epochs, respectively. We found that the performance of AR during reconstruction tends to deteriorate if the network is over-trained, while for ACR, we did not find any noticeable improvement or deterioration due to over-training, suggesting that AR is more susceptible to overfitting as compared to ACR.

The performance of ACR and the competing model- and data-based techniques, averaged over the test data, is reported in Table II. Representative reconstructed images for different methods are shown in Fig. 3. The average PSNR and SSIM of the reconstructed images indicate that the proposed ACR method leads to better reconstruction than TV (approximately 1 dB higher PSNR), which is by far the most popular analytical convex regularizer with well-studied convergence properties. Including the SFB term in conjunction with  $\mathcal{R}'$  leads to slight improvement in performance, although the SFB term alone turns out to be significantly inferior to TV. This indicates the need for a convex regularizer, like ours, that goes beyond linear sparsity-based convex prior models and learns more intricate structures specific to the application at hand.

The convex regularizers arising from linear sparsifying dictionaries (somewhat akin to the SFB-like priors in spirit) lead to a convex variational problem that can be formulated as a quadratic program (for the squared- $\ell_2$  data-fidelity measure) and solved efficiently. Nevertheless, convex priors emerging from linear sparsity might be limited in their ability to capture reasonably complicated signal priors. On the other extreme of the spectrum, a convex prior built by utilizing a parametric ICNN is substantially more expressive, although the corresponding variational objectives are relatively harder to minimize as compared to the linear sparsity priors. In particular, it was shown in [39, Theorem 1] that ICNNs with non-negative weights and ReLU activations are universal approximators in the class of Lipschitz convex functions over a compact domain. Therefore, the proposed ACR regularizers subsume the class of  $\ell_1$  sparsity based convex priors and are capable of representing more complicated image priors. The performance metrics reported in Table II indicate the performance-vs.-complexity trade-off in a quantitative manner. While the proposed ACR prior outperforms the SFB prior alone by a significant margin, it has approximately 40 times as many parameters as compared

to SFB and the reconstruction time is nearly three-fold higher. Thus, the proposed convex prior results in a variational problem that entails somewhat higher complexity in comparison with TV or the SFB prior, but one can still use a provably convergent sub-gradient method for optimization (as shown in Lemma 1). Unsurprisingly, the end-to-end trained networks on supervised data turn out to be more efficient in terms of reconstruction time as opposed to the methods that require solving a variational reconstruction problem in high dimension.

For the sparse-view CT experiment, the classical AR approach yields reconstruction that is superior to ACR, although our experiments in the sequel reveal that this improvement is not consistent across applications. As seen from the average PSNR and SSIM on test images reported in Table II, the performance of ACR is comparable (slightly worse in terms of PSNR, but better in SSIM) with a U-Net-based post-processing network [3] trained with FBP's as the input and the ground-truth images as the target. This indicates that the proposed ACR approach, along with offering the flexibility of unsupervised training, performs approximately on par with a fully data-driven supervised method. As one would expect, the LPD framework trained using paired examples performs the best among all the methods we compare, since it incorporates the acquisition physics into the architecture.

### B. Reconstruction in limited-angle CT

CT reconstruction from limited-angle projection data, where no measurement is available in a particular angular region, is considered to be a particularly challenging ill-posed inverse problem. Limited-angle projections arise primarily because of limited scan-time or restricted scanner movement in certain applications. Due to the lack of projection data in an angular region, the reconstruction performance depends critically on the image prior. Akin to the sparse-view CT experiment, the projection data for the limited-angle CT experiment is simulated using ODL with a parallel-beam acquisition geometry. The projection data is corrupted using white Gaussian noise with  $\sigma = 3.2$  and reconstructed with 350 angles, 700 rays/angle, with a missing angular wedge of  $60^\circ$ . In this experiment, we compare our method with two model-based (FBP and TV) and three data-driven approaches (LPD, AR, and post-processing U-Net) as before. We found that unlike sparse-view CT, the LPD architecture had to be modified for limited-angle CT by reducing the overall number of layers and the number of feature channels in the convolutional layers, and by adding batch normalization after each convolutional layer in the primal and dual spaces. These modifications resulted in an LPD network with fewer learnable parameters and helped avoid overfitting while training on considerably ill-posed measurements. The LPD model was also trained with an increased batch size of 15 to ensure stability in terms of convergence during training.

The ACR architecture is constructed with  $L = 5$  layers, and  $\mathcal{B}_i$  and  $\mathcal{W}_i$  are all chosen as convolutional layers with  $5 \times 5$  kernels and 16 channels. Reducing the total number of layers and the number of feature channels in each layer of ACR as compared to the sparse-view reconstruction experiment helped avoid overfitting in the face of severely ill-posed data.

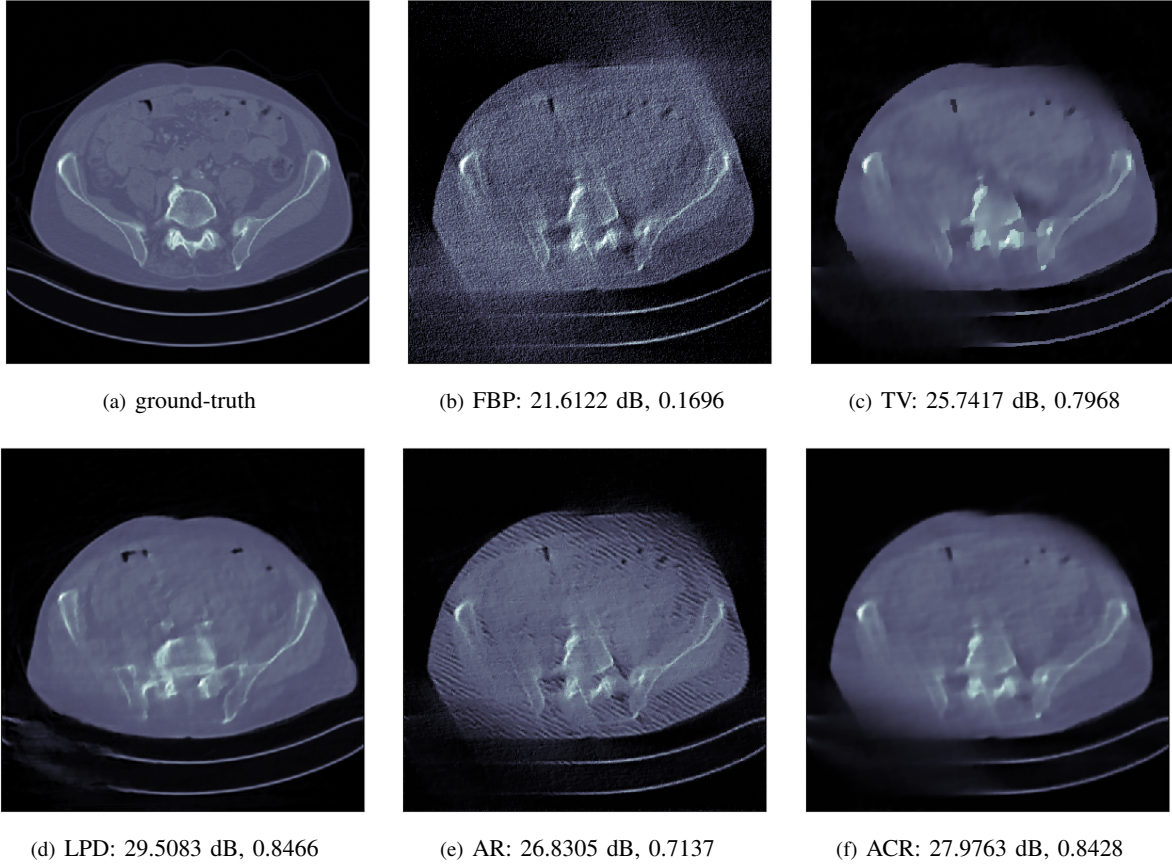


Fig. 4. Reconstructed images obtained using different methods, along with the associated PSNR and SSIM, for limited-angle CT. In this case, ACR outperforms TV and AR in terms of reconstruction quality.

The *RMSprop* optimizer (following the recommendation in [17], since *Adam* gave no improvement in convergence during training, unlike sparse-view CT) with a learning rate of  $5 \times 10^{-5}$  is used for training. The gradient penalty term in (22) is chosen to be 10.0 and the penalty parameter  $\rho_0$  is initialized the same way as in the sparse-view CT experiment. The convolutional layers for the SFB penalty is taken as  $5 \times 5$  filters with 32 output channels.

The reconstruction is performed by solving the variational problem via gradient-descent for 2000 iterations with a step size of  $10^{-5}$ . We observed that the reconstruction performance of AR can deteriorate if an early stopping is not applied. So, for a fair comparison, we report the highest PSNR achieved by AR during reconstruction. Such an early stopping was not needed for ACR. This phenomenon indicates a notable algorithmic advantage of a convex regularizer over a non-convex one.

The average PSNR and SSIM are reported in Table III for various competing methods. To facilitate visual comparison, an example of the reconstruction quality for a representative test image is shown in Fig. 4. From the average PSNR and SSIM reported in Table III we can see that ACR outperforms both model-based approaches and AR. As opposed to sparse-view CT, AR under-performs significantly in the limited-angle setting and produces streak-like artifacts that cover the whole image and are particularly concentrated near the missing angular region. The reason behind this phenomenon is that AR favors

oscillations in the direction of the blurring artifacts in the FBP reconstruction. Since the ground-truth images exhibit more high frequency components pointing in this spatial direction as compared to FBP reconstructions, this becomes a natural feature to use for telling apart the two distributions. Furthermore, the data fidelity term does not contain any information on these regions, and, therefore, it cannot steer AR away from over-regularizing. As a result, AR attempts to maximize the variation in the main blurring direction. This behavior does not arise in case of ACR, leading to better reconstructions. We hypothesize that this is due to the fact that a functional favoring high oscillation in a spatial direction is typically non-convex. In particular, if we consider the total variation in this spatial direction as a measure of oscillation, the negative total variation functional would emerge as an optimal functional to maximally favor high variation in the given spatial direction. This functional is concave, explaining why the convex structure of ACR prevents it from approximating this functional and hence from overly introducing a high amount of oscillation in the reconstruction. Thus, contrary to the conventional intuition, the restricted expressive power of a convex regularizer turns out to be advantageous, especially in a limited data scenario with a highly ill-conditioned forward operator. Similar to sparse-view CT, including the SFB term in ACR leads to a slightly better reconstruction performance and the SFB term as the standalone convex regularizer performs worse than both TV and ACR with

methods	PSNR (dB)	SSIM	# parameters
FBP	17.1949	0.1852	1
TV	25.6778	0.7934	1
LPD	28.9480	0.8394	127 370
FBP + U-Net	29.1103	0.8067	14 787 777
AR	23.6475	0.6257	133 792
ACR (no SFB)	26.4459	0.8184	34 897
ACR (with SFB)	26.7767	0.8266	42 898
SFB + $\rho_0 \ \mathbf{x}\ _2^2$	19.1876	0.1436	8 001

TABLE III

AVERAGE PSNR AND SSIM OVER TEST DATA FOR DIFFERENT RECONSTRUCTION METHODS FOR LIMITED-ANGLE CT.

methods	TV	SFB	ACR	AR	AR2
# parameters	1	4 706	142 594	2 562 242	142 594

TABLE IV

NUMBER OF PARAMETERS IN EACH OF THE DEBLURRING METHODS.

and without the SFB term. For limited-angle projection data, we found that LPD and the U-Net post-processor perform almost on par with each other. Since there is no data available for a considerable angular region, including the acquisition physics into the network does not bring in any significant advantage.

### C. Image deblurring

We will now describe the experimental setup and numerical results for the image deblurring task conducted on the STL-10 dataset [33]. The deblurring dataset is created by computing the noisy measurements,  $\mathbf{y}^\delta$ , by first smoothing the ground truth images,  $\mathbf{x} \in [0, 1]^{3 \times 96 \times 96}$ , from the STL-10 dataset, with a  $3 \times 3$  averaging filter in each of the RGB channels and subsequently adding zero-centered Gaussian noise with a standard deviation of 0.05. All variants of AR and ACR are trained on pairs  $(\mathbf{x}_i, \mathcal{A}^\dagger \mathbf{y}_i^\delta)$ , where the action of  $\mathcal{A}^\dagger$  is approximated via an overfitting Landweber reconstruction. More specifically, we approximate  $\mathcal{A}^\dagger \mathbf{y}^\delta$  by minimizing the data-fidelity error  $\|\mathcal{A} \mathbf{x} - \mathbf{y}^\delta\|_2^2$  via gradient-descent and terminating the iterations when the Morozov's discrepancy principle is satisfied (i.e.,  $\|\mathcal{A} \mathbf{x}_k - \mathbf{y}^\delta\|_2$  falls below the noise level  $\delta$ , where  $\mathbf{x}_k$  is the  $k^{\text{th}}$  iterate of gradient-descent.)

The architecture of the Lipschitz-convex component  $\mathcal{R}'$  in ACR is a discretized version of the architecture described in Section III-B and is built with  $L = 5$  layers. We set  $\varphi_i$  to

be the leaky-ReLU function with a negative slope of 0.2 for layers  $i = 0, \dots, L$ . All  $\mathcal{B}_i$  and  $\mathcal{W}_i$  are given by convolutional layers with kernels of size  $5 \times 5$  and 32 output channels. The dimensional reduction to a single real-valued output in the last layer occurs via a global average-pooling. The negative weights in the filters of  $\mathcal{B}_i$  are zero-clipped after each training step to preserve convexity. The convolutional layer in the SFB consists of a  $7 \times 7$  kernel and has 32 output channels. We compare the ACR and the SFB with the original AR and, due to its slightly sub-optimal performance, a second adversarial regularizer which we refer to as AR2. The AR2 regularizer has the same architecture as the ACR, but the  $\mathcal{B}_i$ 's are not restricted to be non-negative, thus allowing it to be non-convex. AR2 has significantly fewer parameters than AR and is less prone to overfitting as a consequence.

All models are trained for 8 epochs with a batch size of 100 using the *Adam* optimizer with a learning rate of  $5 \times 10^{-5}$  and  $\beta = (0.9, 0.99)$ . The number of trainable parameters in the networks are given in Table IV.

Subsequent to training, we compute the reconstructions by minimizing the corresponding variational functional. The optimization is carried out by using gradient descent for 4096 steps with a step size of  $10^4/27648 \approx 0.36$  (27648 is the size of the image/measurement). We chose the best reconstruction out of all the 4096 steps, which (with the PSNR maximizing penalty parameter) was, in all cases, approximately the last iteration. This, as before, confirms that the ACR does not require early stopping.

The results of the experiments are reported in Table V, while the representative deblurred images are shown in Figure 5 for a visual comparison. One can see that all learned reconstruction methods outperform TV. However, while the ACR and the AR2 outperform TV by a significant margin, e.g., approximately 1 dB in terms the PSNR values, AR only outperforms it by a somewhat smaller margin, and the SFB outperforms TV only by half a dB. The results lead us to two conclusions for this deblurring setting: (i) The convexity of the ACR does not seem to be a significant constraint in terms of performance and (ii) one seems to benefit from using more powerful and expressive convex models than the SFB prior.

## VI. CONCLUSIONS

We proposed a novel data-driven strongly-convex regularization approach for inverse problems and established analytical convergence guarantee and stability estimate. Moreover, we showed the existence of a sub-gradient descent algorithm for minimizing the variational loss, leading to a reconstruction error that decays monotonically to zero with iterations. The proposed ACR approach brings together the power of data-driven inference and the provability of analytical convex regularization. Numerical performance evaluation on sparse-view CT suggests that ACR is superior to the classical TV reconstruction, but gets outperformed by the non-convex AR method, albeit with significantly fewer parameters in the regularizer network as compared to AR. The image deblurring experiment indicates that the ACR model is at least on par and sometimes better than AR in terms of reconstruction quality,

	stats	TV	SFB	ACR	AR	AR2
PSNR (dB)	mean	25.50	26.05	26.55	26.35	26.57
	median	25.01	25.82	26.12	25.99	26.21
	std. dev.	2.08	1.71	2.05	1.98	1.88
SSIM	mean	0.80	0.81	0.83	0.82	0.83
	median	0.80	0.81	0.83	0.82	0.83
	std. dev.	0.05	0.04	0.03	0.03	0.03

TABLE V

AVERAGE TEST PERFORMANCE FOR DEBLURRING ON THE STL-10 DATASET FOR DIFFERENT METHODS IN TERMS OF PSNR AND SSIM.



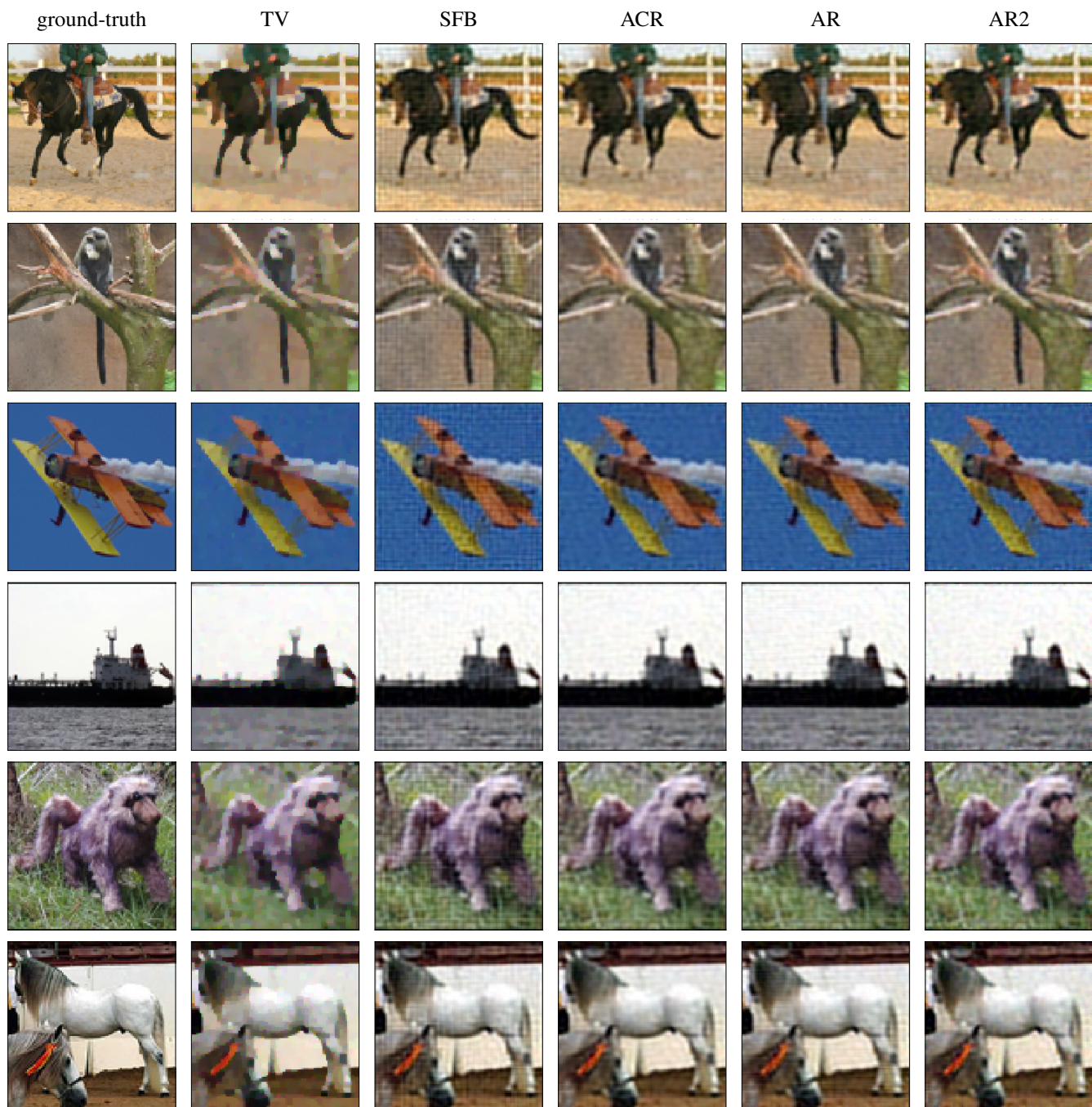


Fig. 5. A visual comparison of different methods for deblurring some representative images in the STL10 dataset.

thus showing a clear advantage both in terms of theoretical guarantees and numerical performance. We also noted that ACR could be parametrized more parsimoniously as compared to its non-convex counterpart without significantly affecting its performance, while avoiding overfitting in a limited-data scenario and thereby leading to visibly fewer artifacts in the reconstructed image. Further, we noted that, unlike AR, ACR did not require early stopping during training to avoid overfitting; or during reconstruction to prevent the gradient-descent updates from diverging.

## REFERENCES

- [1] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb, “Solving inverse problems using data-driven models,” *Acta Numerica*, vol. 28, pp. 1–174, 2019.
- [2] B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, and M. S. Rosen, “Image reconstruction by domain-transform manifold learning,” *Nature*, vol. 555, pp. 487–492, 2018.
- [3] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, “Deep convolutional neural network for inverse problems in imaging,” *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, 2017.
- [4] J. Adler and O. Öktem, “Solving ill-posed inverse problems using iterative deep neural networks,” *Inverse Problems*, vol. 33, no. 12, 2009.
- [5] J. Adler and O. Öktem, “Learned primal-dual reconstruction,” *IEEE transactions on medical imaging*, vol. 37, no. 6, pp. 1322–1332, 2018.



- [6] E. Kobler, T. Klatzer, K. Hammernik, and T. Pock, "Variational networks: connecting variational methods and deep learning," in *German conference on pattern recognition*. Springer, 2017, pp. 281–293.
- [7] T. Meinhardt, M. Moller, C. Hazirbas, and D. Cremers, "Learning proximal operators: Using denoising networks for regularizing inverse imaging problems," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1781–1790.
- [8] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.
- [9] R. Rubinstein, T. Peleg, and M. Elad, "Analysis k-svd: A dictionary-learning algorithm for the analysis sparse model," *IEEE Transactions on Signal Processing*, vol. 61, no. 3, pp. 661–677, 2013.
- [10] S. Ravishanker and Y. Bresler, "Learning sparsifying transforms," *IEEE Transactions on Signal Processing*, vol. 61, no. 5, pp. 1072–1086, 2012.
- [11] J. Sulam, V. Pappas, Y. Romano, and M. Elad, "Multilayer convolutional sparse modeling: pursuit and dictionary learning," *IEEE Transactions on Signal Processing*, vol. 5, no. 15, pp. 4090–4104, 2018.
- [12] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (red)," *SIAM Journal on Imaging Sciences*, vol. 10, no. 4, pp. 1804–1844, 2017.
- [13] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play admm for image restoration: Fixed-point convergence and applications," *IEEE Transactions on Computational Imaging*, vol. 3, no. 1, pp. 84–98, 2016.
- [14] E. T. Reehorst and P. Schniter, "Regularization by denoising: clarifications and new interpretations," *IEEE Transactions on Computational Imaging*, vol. 5, no. 1, pp. 52–67, 2019.
- [15] J. Liu, Y. Sun, C. Eldeniz, W. Gan, A. H., and K. U. S., "Rare: image reconstruction using deep priors learned without groundtruth," *IEEE J. Selected Topics in Signal Processing*, vol. 14, no. 6, pp. 1088–1099, 2020.
- [16] Y. Sun, B. Wohlberg, and K. U. S., "An online plug-and-play algorithm for regularized image reconstruction," *IEEE Transactions on Computational Imaging*, vol. 5, no. 3, pp. 395–408, 2019.
- [17] S. Lunz, O. Öktem, and C.-B. Schönlieb, "Adversarial regularizers in inverse problems," in *Advances in Neural Information Processing Systems*, 2018, pp. 8507–8516.
- [18] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier, "NETT: Solving inverse problems with deep neural networks," *arXiv preprint arXiv:1803.00092v3*, Dec. 2019.
- [19] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.
- [20] E. Kobler, A. Effland, K. Kunisch, and T. Pock, "Total deep variation for linear inverse problems," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7549–7558.
- [21] P. Peng, S. Jalali, and X. Yuan, "Auto-encoders for compressed sensing," 2019.
- [22] S. Dittmer, T. Kluth, P. Maass, and D. O. Bager, "Regularization by architecture: A deep prior approach for inverse problems," *Journal of Mathematical Imaging and Vision*, pp. 1–15, 2019.
- [23] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *International Conference on Machine Learning*, 2017, pp. 146–155.
- [24] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875v3*, Dec. 2017.
- [25] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen, *Variational methods in imaging*. Springer, 2009.
- [26] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [27] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv preprint arXiv:1704.00028v3*, Dec. 2017.
- [28] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.
- [29] Y. Nesterov, "Primal-dual subgradient methods for convex problems," *Mathematical programming*, vol. 120, no. 1, pp. 221–259, 2009.
- [30] S. Boyd, *Subgradient methods*, May 2014 (accessed July 27, 2020). [Online]. Available: [https://stanford.edu/class/ee364b/lectures/subgrad\\_method\\_notes.pdf](https://stanford.edu/class/ee364b/lectures/subgrad_method_notes.pdf)
- [31] C. McCollough, "Tfg-207a-04: Overview of the low dose ct grand challenge," *Medical Physics*, vol. 43, no. 6, pp. 3759–3760, 2014.
- [32] J. Adler, H. Kohr, and O. Öktem, "Operator discretization library (odl)," *Software available from https://github.com/odlgroup/odl*, 2017.
- [33] A. Coates, A. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011, pp. 215–223.
- [34] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [35] L. Pfister and Y. Bresler, "Learning filter-bank sparsifying transforms," *arXiv preprint arXiv:1803.01980v1*, 2018.
- [36] J. Maggu, E. Chouzenoux, G. Chierchia, and A. Majumdar, "Convolutional transform learning," in *International Conference on Neural Information Processing*, 2018, pp. 391–398.
- [37] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.
- [38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [39] Y. Chen, Y. Shi, and B. Zhang, "Optimal control via neural networks: A convex approach," *arXiv preprint arXiv:1805.11835v5*, 2019.