Controlling Neural Networks via Energy Dissipation

Michael Moeller
University of Siegen
michael.moeller@uni-siegen.de

Thomas Möllenhoff
TU Munich
thomas.moellenhoff@tum.de

Daniel Cremers
TU Munich
cremers@tum.de

Abstract

The last decade has shown a tremendous success in solving various computer vision problems with the help of deep learning techniques. Lately, many works have demonstrated that learning-based approaches with suitable network architectures even exhibit superior performance for the solution of (ill-posed) image reconstruction problems such as deblurring, super-resolution, or medical image reconstruction. The drawback of purely learning-based methods, however, is that they cannot provide provable guarantees for the trained network to follow a given data formation process during inference. In this work we propose energy dissipating networks that iteratively compute a descent direction with respect to a given cost function or energy at the currently estimated reconstruction. Therefore, an adaptive step size rule such as a line-search, along with a suitable number of iterations can guarantee the reconstruction to follow a given data formation model encoded in the energy to arbitrary precision, and hence control the model's behavior even during test time. We prove that under standard assumptions, descent using the direction predicted by the network converges (linearly) to the global minimum of the energy. We illustrate the effectiveness of the proposed approach in experiments on single image super resolution and computed tomography (CT) reconstruction, and further illustrate extensions to convex feasibility problems.

1. Introduction

In the overwhelming number of imaging applications, the quantity of interest cannot be observed directly, but rather has to be inferred from measurements that contain implicit information about it. For instance, color images have to be restored from the raw data captured through a color filter array (demosaicking), suboptimal foci or camera movements cause blurs that ought to be removed to obtain visually pleasing images (deblurring), and non-invasive medical imaging techniques such as magnetic resonance imaging (MRI) or computed tomography (CT) can faithfully be modeled as sampling the image's Fourier transform

and computing its Radon transform, respectively. Mathematically, the above problems can be phrased as *linear inverse problems* in which one tries to recover the desired quantity $\hat{u}$ from measurements f that arise from applying an application-dependent linear operator A to the unknown and contain additive noise ξ:

$$f = A\hat{u} + \xi. \tag{1}$$

Unfortunately, most practically relevant inverse problems are *ill-posed*, meaning that equation (1) either does not determine $\hat{u}$ uniquely even if $\xi = 0$, or tiny amounts of noise ξ can alter the naive prediction of $\hat{u}$ significantly. These phenomena have been well-investigated from a mathematical perspective with *regularization methods* being the tool to still obtain provably stable reconstructions.

At the heart of all regularization methods is the idea not to determine a naive estimate like $u = A^{\dagger} f$ for the pseudo-inverse $A^{\dagger}$. Instead, one determines an estimate u for which

$$\|Au - f\| \leq \delta, \tag{2}$$

holds for a suitable norm $\| \cdot \|$ and some $\delta \in \mathbb{R}$ proportional to $\|\xi\|$. The actual u is found by some iterative procedure that is stopped as soon as (2) is met, or by explicitly enforcing a desired regularity via a penalty term or constraint in an energy minimization method.

Motivated by the success of deep learning in classification and semantic segmentation tasks [30, 36], researchers have proposed to tackle a wide variety of different linear inverse problems with deep learning techniques, e.g., deblurring [52, 50], super-resolution [16], demosaicking [19], MRI- [53], or CT-reconstruction [26]. All the aforementioned results are based on acquiring or simulating exemplary pairs $(\hat{u}_i, f_i)$ of ground-truth images $\hat{u}_i$ and data f_i and training a network $\mathcal{G}$ which, for suitable parameters θ, accomplishes a mapping $\mathcal{G}(f_i; \theta) \approx \hat{u}_i$. While such methods show truly remarkable reconstruction quality in practical applications, there is no mechanism to provably guarantee that the solutions $\mathcal{G}(f; \theta)$ predicted by the network actually explain the measured data in accordance with the (known) data formation process (1), i.e., no a-priori proximity bound in the form of (2) can be guaranteed. The latter

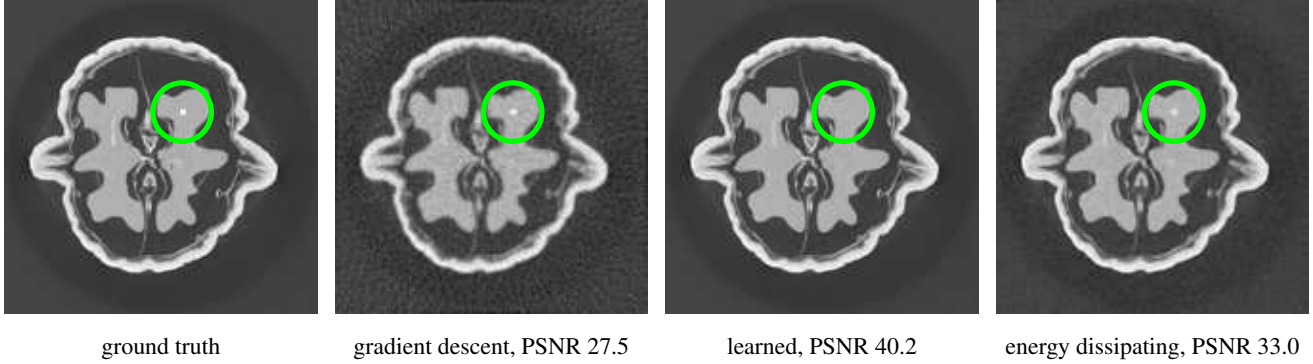| ground truth | gradient descent, PSNR 27.5 | learned, PSNR 40.2 | energy dissipating, PSNR 33.0 |

Figure 1: Illustrating the danger of learning safety-critical reconstruction algorithms: When overfitting a network to reconstruct a walnut image without a simulated pathology and using it to predict the reconstruction of a walnut with such an artifact (left image, white blob in the green circle), the reconstruction (middle right) is extremely clean, but the most important aspect is entirely lost, despite being contained in the data as illustrated in the plain gradient descent reconstruction (middle left). Our energy dissipating networks (right image) are able to benefit from the power of learning-based techniques while allowing to provably guarantee data fidelity constraints such as (2).

can pose a significant risk in trusting the networks prediction, particularly if little, or biased training data is provided. We illustrated such a risk in a toy example in Fig. 1: A CT scan is simulated on the ground truth image shown on the left, which is a slight modification of the walnut image from [21] with an additional small blob in the upper right. The simplest reconstruction, unregularized gradient descent, is very noisy, but the anomaly (our blob), is clearly visible. A network that has been trained on reconstructing the walnut without the blob is noise-free and has the highest peak-signal-to-noise-ratio (PSNR). However, it completely removed the important pathology (middle right). This is particularly disconcerting because the measured data clearly contains information about the blob as seen in the gradient descent image, and a purely learning-based approach may just decide to disrespect the data formation process (2) during inference.

Our proposed approach (illustrated in Fig. 1 on the right) is a learning-based iterative algorithm that can *guarantee* the solution u to meet the constraint (2) for any predefined (feasible) value of δ. Despite also training our network on reconstructing the walnut without the anomaly only, it is able to reconstruct the blob, allowing its use *even in safety critical applications like medical imaging*.

The key idea is to train a neural network $\mathcal{G}$ for predicting a descent direction d^k for a given model driven (differentiable) energy E such as $E(u) = \frac{1}{2}\|Au - f\|^2$. The network takes the current iterate u^k, the data f, and the gradient $\nabla E(u^k)$ as inputs and predicts a direction:

$$d^k = \mathcal{G}(u^k, f, \nabla E(u^k); \theta). \qquad (3)$$

The output of the network is constrained in such a way that for a fixed parameter $\zeta > 0$ the following condition prov-

ably holds for arbitrary points u^k:

$$\langle d^k, \nabla E(u^k)\rangle \geq \zeta\|\nabla E(u^k)\|. \qquad (4)$$

This property allows us to employ a descent algorithm

$$u^{k+1} = u^k - \tau_k\, d^k, \qquad (5)$$

in which a suitable (adaptive) step size rule for τ_k, such as a line-search, can guarantee the convergence to a minimizer of E under weak assumptions. Therefore, the proposed scheme (5) can provably enforce constraints like (2) on unseen test data, while also benefiting from training data and the inductive bias of modern deep architectures.

2. Related Work

2.1. Model-based solutions of inverse problems

Two of the most successful strategies for solving inverse problem in imaging are variational methods as well as (related) iterative regularization techniques. The former phrase the solution as a minimization problem of the form

$$\hat{u} = \arg\min_u H(u; f) + \alpha R(u), \qquad (6)$$

for a data fidelity term H, regularization term R and a trade-off parameter α that has to be chosen depending on the amount of noise in the data. As an example, a common choice for the data term is $H(u; f) = \frac{1}{2}\|Au - f\|^2$ and $R(u) = |\nabla u|$ is often the celebrated total variation (TV) regularizer [48]. Variants include minimizing the regularizer subject to a constraint on the fidelity, also know as Morozow regularization, or minimizing the fidelity term subject to a constraint on the regularizer, also known as Lavrentiev regularization. Other popular regularizers include extensions of the TV, for example introducing higher-order

derivatives [9, 10], sparsity priors for certain representations, such as wavelets or dictionaries [37], or application-driven penalties [23].

Closely related iterative approaches construct sequences $\{u_k\}$ that decrease the fidelity term monotonically along a suitable path (for instance steered by an appropriate regularizer as in [43]), and become a regularization method via stopping criteria such as the *discrepancy principle*, see [7].

2.2. Data-driven and hybrid methods

While the model-based reconstruction techniques of Sec. 2.1 admit a thorough mathematical understanding of their behavior with well defined regularization properties, see [7] for an overview, the solution quality on common benchmarks can often be enhanced significantly by turning to data-driven approaches.

Most frequently, convolutional neural networks (CNNs) [18, 31] are used to solve such image reconstruction problems, e.g. for deblurring [52], single image super resolution [42, 16], or CT reconstruction [25]. Another line of works pioneered in [20] is to take algorithms used to solve the model-based minimization problem (6), unroll them, and declare certain parts to be learnable [49, 56, 29, 13]. Although such architectures are well motivated and often yield excellent results with rather few learnable parameters, they do not come with any provable guarantees.

Alternatively, researchers have considered learning the regularizer in (6) [47, 2, 22, 12]. This typically results in difficult bilevel optimization problems or requires additional assumptions such as sparsity. Moreover, the considered penalties are rather simple, such that they do not quite reach the performance of fully data-driven models.

Other approaches have proposed algorithmic schemes that replace proximal operators of the regularizer by a CNN, but don't come with any provable guarantees [38, 11, 55] or impose restrictive conditions on the network [46].

In order to obtain guarantees, the recent approaches [34, 35] propose convergent energy minimization methods that incorporate update steps based on deep networks. The fundamental difference to our approach is, that in these works the data-driven updates are considered an error, which is then controlled by an interleaving with sufficiently many standard optimization steps. In our approach, every update is solely based on the neural network, and each update provably reduces the energy. Furthermore, the models in [34, 35] are trained on auxilliary tasks (such as image denoising), while we propose a training procedure that is more in line with the actual task of energy minimization. More general approaches on learning to optimize are based on recurrent neural networks [4] or reinforcement learning [33]. However, it still remains challenging to provide a rigorous convergence analysis.

Some recent approaches have considered to regularize

the reconstruction by the parametrization of a network, e.g., deep image priors in [51], or deep decoders [24]. This requires to solve a large nonconvex optimization problem which is rather expensive and due to local optima may also prevent a provable satisfaction of (2).

Finally, another line of works train generative models and represent the variable u to be reconstructed as the transformation of a latent variable z. While these approaches yield strong results, see e.g., [8], the nonconvex optimization can not necessarily guarantee a constraint like (2).

Current methods for imposing constraints directly in the network architecture [3, 17] are limited to low-dimensional linear inequality constraints or polyhedra with moderate numbers of vertices. Therefore, directly incorporating (and differentiating through) projections onto complex constraint sets as (2) currently seems infeasible. Nevertheless, the proposed technique, which we present in the following section, can ensure difficult constraints such as (2) within a data-driven approach.

3. Energy Dissipating Networks

3.1. Paths in the energy landscape

As presented in Eq. (5), we propose a simple technique that uses a data-driven neural network $\mathcal{G}$ to iteratively predict descent directions with respect to a given problem dependent cost function or *energy* E, which we assume to be continuously differentiable. While the idea of learning the gradient descent update in inverse problems has been considered under the framework of unrolled algorithms [1], our architecture is conceptually different: Instead of training an unrolled architecture in an end-to-end fashion, we merely train the network on separate iterations by minimizing an objective of the form

$$\mathbb{E}_{u^*\sim\mathcal{Y}}\mathbb{E}_{u\sim\mathcal{X}_\theta}\left[\ell\left(u-\mathcal{G}(u,f,\nabla E(u);\theta),u^*\right)\right], \quad (7)$$

with respect to the model parameters θ. In the above equation, ℓ is a loss function and $\mathcal{Y}$ the distribution of the "ground truth" data. A novelty in our approach is to consider an input distribution $\mathcal{X}_\theta$, which depends on the model parameters itself. We will make the above more precise later on, but $\mathcal{X}_\theta$ can be thought of as a distribution over iterates which are visited when running a descent method with directions provided by the model. While an objective of the form (7) seems daunting to optimize, we show that a simple "lagged" approach in which the parameters of $\mathcal{X}_\theta$ are updated sporadically and otherwise fixed works well in practice. Another novelty of our approach is to use the Euclidean projection onto the set

$$\mathcal{C}(\zeta,g) = \{d \mid \langle d,g\rangle \geq \zeta\|g\|\} \quad (8)$$

for g being the gradient of the energy (or Lyapunov function) E at the current iterate as a last layer of the network.
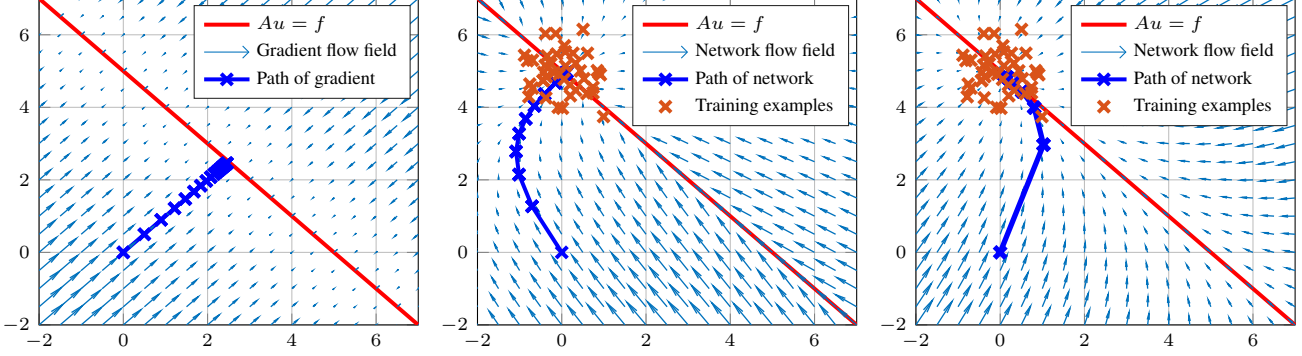
Figure 2: Two dimensional illustration of the proposed energy dissipation network for solving the (underdetermined) inverse problem of finding u such that $u_1 + u_2 = f$: The left plot shows the path gradient descent takes when initialized with zero, along with the magnitude and direction of the gradient at various points in the 2D plane. The middle plot shows the same visualization determined with our energy dissipating networks and additional training data. The right plot shows the results of an energy dissipating network with a more aggressive enforcement of descent directions via scaling ζ in (4) with an additional $\|\nabla E(u^k)\|$. As we can see, in both cases the network learned to create paths that lead towards the provided training data while provably guaranteeing non-increasing data-fidelity costs. The depicted networks even move solutions within the subspace $Au = f$ towards the clustered data points.

By additionally utilizing a backtracking line-search detailed in Alg. 1 to determine suitable step-sizes τ_k, we can state the following convergence result:

Proposition 1. *Consider the iterates given by Alg. 1 for an arbitrary starting point u^0, a continuously differentiable and coercive energy E, and a continuous (in the inputs) model $\mathcal{G}([u, f, \nabla E(u)]; \theta)$ that satisfies*

$$\mathcal{G}(u, f, \nabla E(u); \theta) \in \mathcal{C}(\zeta, \nabla E(u)) \qquad \forall u. \qquad (9)$$

Then the while-loop of Alg. 1 always terminates after finitely many iterations. The energy of the iterates is monotonically decreasing, i.e.

$$E(u^{k+1}) \leq E(u^k), \qquad (10)$$

and the sequence of $\|\nabla E(u^k)\|$ converges to zero, i.e.

$$\lim_{k \to \infty} \|\nabla E(u^k)\| = 0. \qquad (11)$$

Moreover, if E is strictly convex, then the sequence of u^k converges with

$$\lim_{k \to \infty} u^k = \arg\min_u E(u). \qquad (12)$$

Proof. The result is a conclusion of the descent direction, e.g. utilizing standard results like [41, Thm. 3.2]. □

Due to property (10) we refer to our approach as *energy dissipating networks*. Our intuition is that such networks, which are trained on (7) but provably satisfy (9), allow to predict paths in a given energy landscape that monotonically decrease the energy, but at the same time attract the

Algorithm 1 Learned energy dissipation with line-search

1: **Inputs:** Starting point u^0, energy dissipating network $\mathcal{G}(\cdot, \cdot, \nabla E(\cdot); \theta)$, constants $c \in (0, 0.5), \rho \in (0, 1)$
2: **while** not converged (e.g. based on $\|\nabla E(u^k)\|$) **do**
3: $\quad d^k = \mathcal{G}(u^k, f, \nabla E(u^k); \theta)$
4: $\quad \tau_k \leftarrow 1$
5: $\quad u^{k+1} \leftarrow u^k - \tau_k d^k$
6: $\quad$ **while** $E(u^{k+1}) > E(u^k) - c\tau_k \langle d^k, \nabla E(u^k) \rangle$ **do**
7: $\quad\quad \tau_k \leftarrow \rho\,\tau_k$
8: $\quad\quad u^{k+1} \leftarrow u^k - \tau_k d^k$
9: $\quad$ **end while**
10: **end while**

iterates to locations which are more probable according to the training examples. We have illustrated this concepts for $E(u) = \|Au - f\|^2$ and $A = [1\ \ 1]$ in the two-dimensional toy example shown in Fig. 2. While plain gradient descent merely drives the iterates towards the subspace of solutions in a direct way, the proposed energy dissipation networks have the freedom to exploit training data and learn a vector field that drives any initialization to solutions near the training data at $u = \begin{bmatrix} 0 & 5 \end{bmatrix}^\top$ for $f = 5$.

Besides the possibility of reconstructing solutions that reflect properties of the training examples in underdetermined cases as illustrated in Fig. 2, the intuition of taking a more favorable path extends to energies $E(u) = \|Au - f\|^2$ where A is invertible, but severely ill-conditioned: As we will illustrate in the results (Sec. 5), a typical gradient descent iteration on energies $E(u) = \|Au - f^\delta\|^2$ for noisy observations f^δ typically improves the results up to a cer-

tain iteration before the ill-conditioned matrix A leads to the introduction of heavy noise and a degradation of the results. Thus, early stopping, or discrepancy principles are used to stop the iteration once $\|Au - f^\delta\| \approx \|f - f^\delta\|$. Again, a favorable path hopefully allows to stop at a better overall solution. As discrepancy principles are frequently used in the analysis of (continuous / infinite dimensional) inverse problems, our framework possibly allows to derive a regularization method from (5) in the sense of [7, Def. 4.7].

3.2. Ensuring global linear convergence

While Prop. 1 establishes convergence, in applications an upper bound on the number of iterations required to reach a certain optimality gap can be desirable. To establish such a bound, we will make the following assumptions, which are standard in literature:

Assumption 1. *The energy E is L-Lipschitz differentiable, i.e., it satisfies the inequality*

$$E(v) \le E(u) + \langle \nabla E(u), v - u \rangle + \frac{L}{2}\|u - v\|^2, \quad (13)$$

and furthermore satisfies the Polyak-Łojaciewicz inequality with modulus $\mu > 0$,

$$\frac{1}{2}\|\nabla E(u)\|^2 \ge \mu(E(u) - E^*), \quad (14)$$

where $E^* = \min_u E(u)$ is the global minimum.

We remark that functions satisfying the Polyak-Łojaciewicz inequality (14) include strongly convex functions, possibly composed with a linear operator that has a non-trivial kernel. Notably, convexity is not a required assumption, but rather invexity [27, 39]. All examples we consider in the numerical experiments fulfill Asm. 1.

To establish a linear convergence result, we constrain the output of the network to a slightly different constraint set as the one considered in (8). For $\zeta_1 \ge \zeta_2 > 0$ we define it as:

$$\mathcal{C}(\zeta_1, \zeta_2, g) = \left\{ d \mid \langle d, g \rangle \ge \zeta_1\|g\|^2, \|d\| \le \zeta_2\|g\| \right\}. \quad (15)$$

To give an intuition about (15), note that the two conditions imply that the angle θ between d and g is bounded by $\cos\theta \ge \zeta_1/\zeta_2$. Therefore, by choosing appropriate ζ_1 and ζ_2 one can control the angular deviation between the predicted direction of the network and the actual gradient direction. The linear convergence result is made precise in the following proposition.

Proposition 2. *Assume that the energy E satisfies Asm. 1 and that the update directions given by the network meet $d^k \in \mathcal{C}(\zeta_1, \zeta_2, \nabla E(u^k))$. Then (5) with constant step size $\tau_k \equiv \zeta_1/((\zeta_2)^2 L)$ converges linearly,*

$$E(u^{k+1}) - E^* \le \left(1 - \frac{\gamma^2\mu}{L}\right)^k (E(u^0) - E^*), \quad (16)$$

where $\gamma = \zeta_1/\zeta_2$.

Proof. Combining (13) and the descent iteration (5) we have the following bound on the decrease of the energy:

$$\begin{aligned} E(u^{k+1}) - E(u^k) \le & -\tau\langle \nabla E(u^k), d^k \rangle \\ & + \frac{L\tau^2}{2}\|d^k\|^2. \end{aligned} \quad (17)$$

Using $d^k \in \mathcal{C}(\zeta_1, \zeta_2, \nabla E(u^k))$ we can further bound this

$$\begin{aligned} E(u^{k+1}) - E(u^k) & \le \left(\frac{L\tau^2(\zeta_2)^2}{2} - \tau\zeta_1\right)\|\nabla E(u^k)\|^2 \\ & = -\frac{\gamma^2}{2L}\|\nabla E(u^k)\|^2. \end{aligned} \quad (18)$$

Finally, by (14) we have

$$E(u^{k+1}) - E(u^k) \le -\frac{\gamma^2\mu}{L}(E(u^k) - E^*), \quad (19)$$

and rearranging and subtracting E^* on both sides gives

$$E(u^{k+1}) - E^* \le \left(1 - \frac{\gamma^2\mu}{L}\right)(E(u^k) - E^*), \quad (20)$$

which yields the above result. $\qquad\square$

Therefore, the required number of iterations to reach an ε-accurate solution is in the order of $\mathcal{O}\left(\gamma^{-2}(L/\mu)\log(1/\varepsilon)\right)$. By giving the network more freedom to possibly deviate from the true gradient ($\theta \to \pm 90°$, i.e., $\gamma \to 0$), more iterations are required in the worst-case. As an example, the above analysis tells us that if we allow the directions predicted by the network to deviate by at most $45°$ from the true gradient, then in the worst case we might require twice as many ($\cos 45° = 1/\sqrt{2}$) iterations to find an ε-accurate solution as standard gradient descent. Nevertheless, we want to stress that there are also directions which dramatically improve the rate of convergence (for example the Newton direction), which is not captured by this worst-case analysis. As in practice the training data could coincide with the model, it is to be expected that the learned direction will lead to much faster initial convergence than the gradient direction. The above analysis should therefore be only seen as a bound of what could theoretically happen in case the network systematically picks the worst possible (in the sense of energy decay) direction. Therefore, we use Alg. 1 which performs a line-search instead of choosing the very conservative step-size from Prop. 2, to benefit from the scenario when the direction predicted by the model is good.

We remark that the factor in the above iteration complexity could possibly be improved using accelerated variants of gradient descent [40, 39].

4. Implementation

4.1. Satisfying the descent constraints

As discussed in the previous section, for the convergence results in Prop. 1 and Prop. 2 to hold, we either have to provably satisfy the constraints (9) or (15). The constraint (8) is a half-space, and can be satisfied by the projection:

$$z \mapsto z + \max\{\zeta - \langle z, n \rangle, 0\} \cdot n, \quad n = g/\|g\|, \quad (21)$$

which merely consists of linear operations and a rectified linear unit (ReLU), so that it is readily implemented in any deep learning framework. For simplicity, for the set (15) we propose to merely use a parametrization.

Proposition 3. *A surjective map onto* (15) *is given by*

$$z \mapsto \hat{\eta} g + \Pi_B(z - \eta g), \quad (22)$$

where $\eta = \langle z, g \rangle / \|g\|^2$, $\hat{\eta} = \Pi_{[\zeta_1, \zeta_2]}(\eta)$ *and* Π_B *is the projection onto* $B = \{d \mid \|d\| \le \sqrt{(\zeta_2)^2 - \hat{\eta}^2} \|g\|\}$.

Proof. To see this, first note that since $\langle z - \eta g, g \rangle = 0$ also holds after the projection, the first constraint in (15),

$$\langle \hat{\eta} g + \Pi_B(z - \eta g), g \rangle \ge \zeta_1 \|g\|^2, \quad (23)$$

is satisfied since $\hat{\eta} \ge \zeta_1$. For the second condition, we have due to orthogonality:

$$\|\hat{\eta} g + \Pi_B(z - \eta g)\|^2 = \hat{\eta}^2 \|g\|^2 + \|\Pi_B(z - \eta g)\|^2$$
$$\le (\zeta_2)^2 \|g\|^2, \quad (24)$$

so that $\hat{\eta} g + \Pi_B(z - \eta g) \in \mathcal{C}(\zeta_1, \zeta_2, g)$. $\square$

To avoid problems with the division by $\|g\|$, we approximate it using $\max\{\|g\|, 10^{-6}\}$, but also note that Alg. 1 stops once $\|g\|$ becomes small.

4.2. Lagged updating of the training distribution

For training, a first idea might be to incorporate a loss of the form (7) where $u \sim \mathcal{X}$ is sampled from the uniform distribution over the entire space of possible inputs. The latter is very high dimensional in most applications, and due to the curse of dimensionality it is unreasonable to aim at covering it exhaustively. Therefore we propose to introduce a dependency on the model parameters $u \sim \mathcal{X}_\theta$ and consider only inputs which are visited when running descent with the current model. To optimize such an objective, where the training distribution also depends on the model parameters, we propose the following iterative "lagged" strategy, which is an essential component of our approach.

The method is bootstrapped by fixing $\mathcal{X}_\theta$ to be the distribution of inputs obtained by running regular gradient descent on E (up to a maximum number of iterations). After
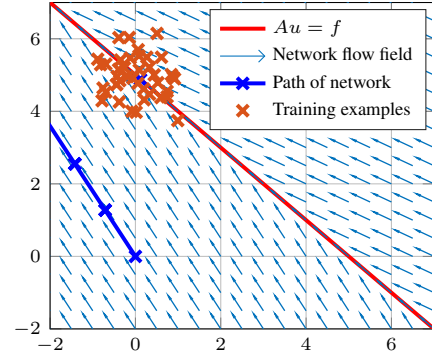

Figure 3: Motivation for the proposed "lagged" iterative scheme: Training only on iterates which are visited in gradient descent leads to a rough first guess. To arrive at a more refined result as shown on the right in Fig. 2, we propose an iterative scheme in which the training distribution is updated based on the current parameters.

a certain number of epochs, we update the distribution to also contain iterates obtained by the algorithm (5) with the current parameters to generate new (network-based) paths. As the path of energy dissipating networks can differ from plain gradient descent significantly, such an iterative strategy is vital for the success of the training.

Returning to our two dimensional toy example of Fig. 2 we show in Fig. 3 the network's descent vector field after only training on the gradient descent iterates (left plot in Fig. 2). The results are reasonable (as the network recognized that the iterations have to be further to the top left), but of course unsatisfactory for practical purposes as the iterates are pushed beyond the set of exemplary points (marked by red crosses). After only two additional iterations of updating the training distribution via the models' current path, we obtain the results shown in Fig. 2 on the right.

4.3. Choosing a suitable energy

In many imaging applications suitable data fidelity terms of model-driven approaches are well-known, or can be derived from maximum a-posteriori probability estimates for suitable noise models. While only using a suitable data fidelity term as an energy or Lyapunov function E allows to predict solutions that approximate the measured data to any desired accuracy, energy dissipating networks could further be safeguarded with a classical regularization term, making a classical regularized solution the lower bound on the performance of the method. We will demonstrate numerical results for both such approaches in Sec. 5. Moreover, although our main motivation stemmed from linear inverse problems in the form of (1), the idea of energy dissipating networks extends far beyond this field of application.

By choosing the distance to any desired constraint set as

Table 1: Quantitative evaluation on the task of single image super resolution ($4\times$ upscaling). The proposed energy dissipating approach simulatenously improves upon the baseline in terms of mean PSNR, SSIM and reconstruction error $\|Au - f\|^2$.

	Gradient Descent			Baseline Network			Energy Dissipating		
	PSNR	SSIM	$\|Au - f\|^2$	PSNR	SSIM	$\|Au - f\|^2$	PSNR	SSIM	$\|Au - f\|^2$
BSD100	25.09	0.6440	$\mathbf{9.5 \cdot 10^{-3}}$	26.67	0.7237	$2.8 \cdot 10^{-1}$	**27.12**	**0.7297**	$\mathbf{9.8 \cdot 10^{-3}}$
Urban100	22.19	0.6230	$\mathbf{6.8 \cdot 10^{-2}}$	24.44	0.7292	$1.9 \cdot 10^{-0}$	**24.83**	**0.7460**	$\mathbf{6.4 \cdot 10^{-2}}$
Set5	26.80	0.7448	$\mathbf{1.3 \cdot 10^{-2}}$	29.94	0.8548	$2.5 \cdot 10^{-1}$	**31.16**	**0.8726**	$\mathbf{8.5 \cdot 10^{-3}}$
Set14	24.74	0.6684	$\mathbf{1.6 \cdot 10^{-2}}$	27.22	0.7642	$4.3 \cdot 10^{-1}$	**27.74**	**0.7709**	$\mathbf{1.3 \cdot 10^{-2}}$

a surrogate energy, one can provably constrain the predicted solutions to any desired constrained set (at least if the some measure of distance to the desired set can be stated in closed form). We will illustrate such an extension to convex feasibility problems in Sec. 5.3 and Sec. 5.4.

5. Applications

We implemented the experiments from Sec. 5.1 and Sec. 5.4 using the PyTorch framework. The CT reconstruction experiments in Sec. 5.2 are implemented in Matlab. All models were trained using the Adam optimizer [28]. In all experiments ℓ in Eq. (7) was chosen as the square loss.

5.1. Single image super resolution

As a first application, we train an energy dissipating network on the task of single image super resolution. The linear operator in the energy $E(u) = \frac{1}{2}\|Au - f\|^2$ is chosen as a $4\times$ downsampling, implemented via average pooling.

The architecture both for the energy dissipating and baseline approach is based on [54], which consists of 20 blocks of 3×3 convolutional filters with 64 channels, ReLUs, and batch normalization layers, before a final 3×3 convolutional layer reduces the output to a single channel. For the energy dissipating approach, the result is fed into the layer given by (22) with $\zeta_1 = 25$, $\zeta_2 = 10000$. Following [54], we initialize with the "DnCNN-B" model pretrained on the task of image denoising, which we found important for the baseline to achieve good performance but had no effect on our approach. Starting from that initialization, we train on the specific task of $4\times$ super resolution on 52×52 patches from images in the BSD100 training set. For the energy dissipating network, the training data is updated every 100 mini-batches according to Sec. 4.2, choosing uniformly 0–10 descent steps to generate the iterates. As the data is generated in an online fashion, samples of a previous model or from the gradient descent initialization are discarded due to efficiency reasons.

During testing, we used a fixed number of 15 iterations in the descent with our network direction. For the gradient descent baseline (corresponding to simple up–sampling) we used 75 iterations to reach a similar reconstruction error.

For both methods, running more iterations did not significantly effect the result. Note that in principle one can run both convergent descent methods until the forward model is satisfied to an arbitrary accuracy. A quantitative evaluation on a standard super resolution benchmark is given in the above Table 1. Note that our goal was not necessarily to achieve high PSNR results, but rather demonstrate in a controlled experiment that our approach can be used to introduce provable guarantees into existing models without significantly worsening (here even improving) the quality.

5.2. CT Reconstruction

Next, we consider reconstructing CT images using the (sparse matrix) operator provided in [21]. The architecture is the same as for the super resolution experiment, but using 17 blocks and the projection layer given by (21). In the lack of highly accurate CT scans, we simulate training data using phantoms from [14, 15] (despite being MRI phantoms), also include random crops of natural images from BSD68 [47], and reconstruct in 2D (slices only). We use $E(u) = \frac{1}{2}\|Au - f\|^2 + \alpha TV_\epsilon(u)$ (with TV_ϵ being a Charbonnier total variation regularization) as our surrogate energy and start with the first 10 gradient descent iterates as the initial distribution of the training data for the lagged updating scheme described in Sec. 4.2.

We test the resulting model by simulating a CT-scan of a walnut image from [21]. We compare our scheme 1 to plain gradient descent on the data term as well as to regularized reconstructions for different regularization parameters α. Fig. 4 shows the PSNR values as well as the decay of the data fidelity term over the number of iterations for two different noise levels. As we can see, after only 5-10 iterations, the energy dissipating networks have reached significantly higher PSNR values than their model-based counter parts, before converging to the same solution (at least in the regularized cases) as predicted by Prop. 1. The energy dissipating network with $\alpha = 0$ reaches $\|Au - f\| \leq \sigma$ after 8 (for $\sigma = 0.3$) and 15 (for $\sigma = 0.15$) iterations, indicating that a discrepancy principle can be useful. Moreover, the data term decays quicker than for the respective gradient methods, indicating that the trained networks can even represent fast optimizers. All plots have been generated with a single
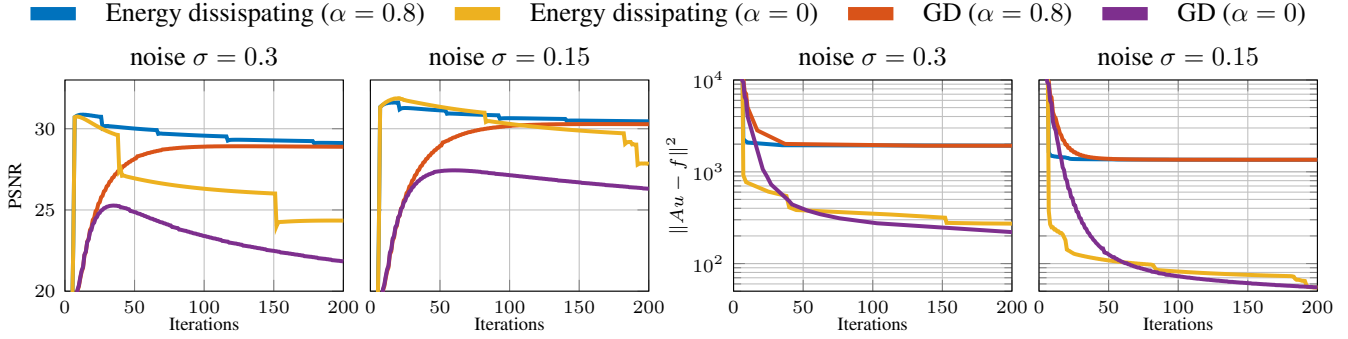
Figure 4: Comparing gradient descent (GD) with energy dissipating networks: the first two plots show the PSNR value over the number of iterations, and illustrate that the energy dissipating networks peak very quickly before (provably) converging to the same minimizer as their surrogate energies. As shown in the right plots, they quickly decrease the data discrepancy term and plateau in the regularized case, or keep decaying when using the data term as a surrogate energy only. Note that all curves of our approach were generated with the same energy dissipating networks, merely changing the surrogate energy during testing, which demonstrates an ability to generalize beyond the specific scenario used during training.

network, hinting at an ability to generalize to different noise levels and regularization parameters.

5.3. Imposing convex constraints

In this section we consider convex feasibility problems, in which the task is to find a point in the non-empty intersection of N closed convex sets $\{C_1, \ldots, C_N\}$. A formulation as a differentiable energy minimization problem is given by:

$$E(u) = \frac{1}{2N} \sum_{i=1}^{N} d_{C_i}^2(u), \qquad (25)$$

where $d_C^2(u) = \min_{v \in C} \|u - v\|^2$ is the squared Euclidean distance of the point u to the set C. The energy (25) is nonnegative and reaches zero if and only if u satisfies all constraints. By standard results in convex analysis [6, Corollary 12.30] it follows that $\nabla d_C^2 = 2(\text{id} - \Pi_C)$ so that the gradient of (25) is 1-Lipschitz and can be implemented if the projections onto the sets C_i are available. Under mild regularity assumptions (which even hold for some nonconvex sets) the energy (25) satisfies the Polyak-Łojaciewicz inequality, see [32, Prop. 8.6], and therefore also Asm. 1.

5.4. Sudoku

To demonstrate the previous application we tackle the problem of solving 9×9 Sudokus, which has emerged as a benchmark for learning based methods [3, 44]. Note that OptNet [3] only considers 4×4 due to scalability issues as remarked in [44]. We use a convex relaxation of the binary feasibility problem on $\{0, 1\}^{9 \times 9 \times 9}$ from [5]. Specifically, we have $N = 5$ in (25), where $C_1, \ldots C_3$ contain simplex constraints for each dimension of the tensor, C_4 constraints on the 3×3 fields in the Sudoku and C_5 encodes the givens.

Table 2: Evaluation on a test set of 50 easy Sudokus.

GD (100 it.)		Baseline Net.		En. Diss. (100 it.)	
Acc.	Solve	Acc.	Solve	Acc.	Solve
76.9%	2.0%	82.6%	4.0%	**87.1%**	**52.0%**

We consider the same architecture as for the previous experiments and train on a dataset of one million Sudokus [45]. For the energy dissipating approach, we constrain the output to be in the constraint set (15) and use the scheme from Sec. 4.2 with 15 iterations to update the training data.

In the above table we compare 100 iterations of gradient descent (equivalent to averaged projections), a baseline which performs a forward pass through a (seperately trained) network and 100 iterations of energy dissipation. We improve upon the baseline which indicates that it is possible to learn directions which reduce the energy of feasibility problems faster than the steepest descent direction.

6. Conclusion

We constrained the outputs of deep networks to be descent directions for suitably chosen energy functions. Combined with a line-serach algorithm, this yields (linear) convergence to minimizers of the underlying energy during inference. We further proposed an iterative training scheme to learn a model which provides descent directions that bias the optimization into the direction of the training data. In experiments, we demonstrated that this approach can be used to control networks via model-based energies, and at the same time improve over standard end-to-end baselines.

References

[1] Jonas Adler and Ozan Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12):124007, 2017. 3

[2] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006. 3

[3] Brandon Amos and J. Zico Kolter. OptNet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning (ICML)*, 2017. 3, 8

[4] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. 3

[5] Francisco J. Aragón Artacho, Jonathan M. Borwein, and Matthew K. Tam. Recent results on Douglas–Rachford methods for combinatorial optimization problems. *Journal of Optimization Theory and Applications*, 163(1):1–30, 2014. 8

[6] Heinz H. Bauschke and Patrick L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011. 8

[7] Martin Benning and Martin Burger. Modern regularization methods for inverse problems. *Acta Numerica*, 27:1111, 2018. 3, 5

[8] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models. In *International Conference on Machine Learning (ICML)*, 2017. 3

[9] Kristian Bredies, Karl Kunisch, and Thomas Pock. Total generalized variation. *SIAM Journal on Imaging Sciences*, 3(3):492–526, 2010. 3

[10] Antonin Chambolle and Pierre-Louis Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76(2):167–188, 1997. 3

[11] J.H. Rick Chang, Chun-Liang Li, Barnabas Poczos, B.V.K. Vijaya Kumar, and Aswin C. Sankaranarayanan. One network to solve them all — solving linear inverse problems using deep projection models. In *International Conference on Computer Vision (ICCV)*, 2017. 3

[12] Yunjin Chen, Rene Ranftl, and Thomas Pock. Insights into analysis operator learning: From patch-based sparse models to higher order mrfs. *IEEE Transactions on Image Processing*, 23(3):1060–1072, 2014. 3

[13] Yunjin Chen, Wei Yu, and Thomas Pock. On learning optimized reaction diffusion processes for effective image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[14] Chris A. Cocosco, Vasken Kollokian, Remi K.S. Kwan, and Alan C. Evans. BrainWeb Phantom. http://brainweb.bic.mni.mcgill.ca/selection_normal.html. Online; accessed 19-March-2019. 7

[15] Chris A. Cocosco, Vasken Kollokian, Remi K.S. Kwan, and Alan C. Evans. BrainWeb: Online Interface to a 3D MRI Simulated Brain Database. *NeuroImage*, 5(4), 1997. 7

[16] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016. 1, 3

[17] Thomas Frerix, Matthias Nießner, and Daniel Cremers. Linear inequality constraints for neural network activations. *arXiv:1902.01785*, 2019. 3

[18] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202, 1980. 3

[19] Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. Deep joint demosaicking and denoising. *ACM Trans. Graph.*, 35(6):191:1–191:12, 2016. 1

[20] Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *International Conference on Machine Learning (ICML)*, 2010. 3

[21] Keijo Hämäläinen, Lauri Harhanen, Aki Kallonen, Antti Kujanpää, Esa Niemi, and Samuli Siltanen. Tomographic X-ray data of a walnut (version 1.0.0). http://doi.org/10.5281/zenodo.1254206. Online; accessed 19-March-2019. 2, 7

[22] Simon Hawe, Martin Kleinsteuber, and Klaus Diepold. Analysis operator learning and its application to image reconstruction. *IEEE Transactions on Image Processing*, 22(6):2138–2150, 2013. 3

[23] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(12):2341–2353, 2011. 3

[24] Reinhard Heckel and Paul Hand. Deep decoder: Concise image representations from untrained non-convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2019. 3

[25] Kyong Hwan Jin, Michael T. McCann, Emmanuel Froustey, and Michael Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Transactions on Image Processing*, 26(9):4509–4522, 2017. 3

[26] Eunhee Kan, Junhong Min, and Jong Chul Ye. WaveNet: a deep convolutional neural network using directional wavelets for low-dose X-ray CT reconstruction. *Medical Physics*, 44:e360–e375, 10 2016. 1

[27] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition. In *European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, pages 795–811, 2016. 5

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 7

[29] Erich Kobler, Teresa Klatzer, Kerstin Hammernik, and Thomas Pock. Variational networks: Connecting variational methods and deep learning. In *German Conference on Pattern Recognition (GCPR)*, 2017. 3

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012. 1

[31] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989. 3

[32] Adrian Lewis, Russell Luke, and Jerome Malick. Local convergence for alternating and averaged nonconvex projections. *Foundations of Computational Mathematics*, 9:485–513, 2009. 8

[33] Ke Li and Jitendra Malik. Learning to optimize. *arXiv:1606.01885*, 2016. 3

[34] Risheng Liu, Shichao Cheng, Xiaokun Liu, Long Ma, Xin Fan, and Zhongxuan Luo. A bridging framework for model optimization and deep propagation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 3

[35] Risheng Liu, Long Ma, Yiyang Wang, and Lei Zhang. Learning converged propagations with deep prior ensemble for image enhancement. *IEEE Transactions on Image Processing*, 28(3):1528–1543, 2019. 3

[36] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1

[37] Julien Mairal, Francis Bach, Jean Ponce, et al. Sparse modeling for image and vision processing. *Foundations and Trends in Computer Graphics and Vision*, 8(2-3):85–283, 2014. 3

[38] Tim Meinhardt, Michael Moeller, Caner Hazirbas, and Daniel Cremers. Learning proximal operators: Using denoising networks for regularizing inverse imaging problems. In *International Conference on Computer Vision (ICCV)*, 2017. 3

[39] Ion Necoara, Yu Nesterov, and Francois Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Mathematical Programming*, pages 1–39, 2018. 5

[40] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018. 5

[41] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer, New York, second edition, 2006. 4

[42] Christian Osendorfer, Hubert Soyer, and Patrick Van der Smagt. Image super-resolution with fast approximate convolutional sparse coding. In *International Conference on Neural Information Processing (ICONIP)*, 2014. 3

[43] Stanley Osher, Martin Burger, Donald Goldfarb, Jinjun Xu, and Wotao Yin. An iterative regularization method for total variation-based image restoration. *Multiscale Modeling & Simulation*, 4(2):460–489, 2005. 3

[44] Rasmus Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks. In *Advances in Neural Information Processing Systems*, pages 3372–3382, 2018. 8

[45] Kyubyong Park. Can neural networks crack Sudoku? https://github.com/Kyubyong/sudoku. Online; accessed 19-March-2019. 8

[46] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Sciences*, 10:1804–1844, 2017. 3

[47] Stefan Roth and Michael J. Black. Fields of experts. *International Journal of Computer Vision*, 82(205), 2009. 3, 7

[48] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Phys. D*, 60(1-4):259–268, 1992. 2

[49] Uwe Schmidt and Stefan Roth. Shrinkage fields for effective image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 3

[50] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[51] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Deep image prior. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[52] Li Xu, Jimmy S.J. Ren, Ce Liu, and Jiaya Jia. Deep convolutional neural network for image deconvolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 1, 3

[53] Guang Yang, Simiao Yu, Hao Dong, Greg Slabaugh, Pier Luigi Dragotti, Xujiong Ye, Fangde Liu, Simon Arridge, Jennifer Keegan, Yike Guo, and David Firmin. DAGAN: Deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1310–1321, 2018. 1

[54] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 7

[55] Kai Zhang, Wangmeng Zuo, Shuhang Gu, and Lei Zhang. Learning deep CNN denoiser prior for image restoration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[56] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H.S. Torr. Conditional random fields as recurrent neural networks. In *International Conference on Computer Vision (ICCV)*, 2015. 3