Solving Inverse Problems with Model Mismatch using Untrained Neural Networks within Model-based Architectures

Anonymous authors
Paper under double-blind review

Abstract

Model-based deep learning methods such as *loop unrolling* (LU) and *deep equilibrium model* (DEQ) extensions offer outstanding performance in solving inverse problems (IP). These methods unroll the optimization iterations into a sequence of neural networks that in effect learn a regularization function from data. While these architectures are currently state-of-the-art in numerous applications, their success heavily relies on the accuracy of the forward model. This assumption can be limiting in many physical applications due to model simplifications or uncertainties in the apparatus. To address forward model mismatch, we introduce an untrained forward model residual block within the model-based architecture to match the data consistency in the measurement domain for each instance. We propose two variants in well-known model-based architectures (LU and DEQ) and prove convergence under mild conditions. The experiments show significant quality improvement in removing artifacts and preserving details across three distinct applications, encompassing both linear and nonlinear inverse problems. Moreover, we highlight reconstruction effectiveness in intermediate steps and showcase robustness to random initialization of the residual block and a higher number of iterations during evaluation.

1 Introduction

Consider an inverse problem of the following form:

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{\epsilon}. \tag{1}$$

The goal is to reconstruct the latent signal $\boldsymbol{x}$ from the measurements $\boldsymbol{y}$ in the presence of noise $\boldsymbol{\epsilon}$, where typically the forward model $\mathcal{A}$ is assumed to be known. Inverse problems are generally challenging because they are often ill-posed, *i.e.,* the solution is not unique, or the reconstruction is highly sensitive to noise and/or model mismatch. The traditional approach to recovering $\boldsymbol{x}$ from the measurements $\boldsymbol{y}$ is by solving a regularized optimization problem of the form:

$$\min_{\boldsymbol{x}} \ \frac{1}{2} \ \|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2 + \gamma r(\boldsymbol{x}), \tag{2}$$

where $\gamma \geq 0$ is an appropriately-chosen parameter. The regularizer r is usually predetermined based on some known or desired structure, e.g., ℓ_1-, ℓ_2-, or total variation (TV) norm to promote sparsity, smoothness, or edges in image reconstructions, respectively. Solving (2) requires careful consideration of the underlying physics or the forward model $\mathcal{A}$ to obtain a stable and accurate reconstruction. However, knowing $\mathcal{A}$ can be challenging in practice. The reasons for this include inaccurate measurements and challenging calibration, highly nonlinear and/or computationally expensive models replaced by simplified versions, as well as access to only approximations of certain features. Usually, some knowledge of the true model, designated $\mathcal{A}_0$ in this work, is available. This occurs in many applications, including blind deconvolution/deblurring problems, recovering seismic layer models using the simplified acoustic wave equation as the forward model (Mousa & Al-Shuhail, 2011), determining fault locations in media with unknown structures (Mahmoud & Khalid, 2013), etc.

Figure 1: A proximal LU network is trained for a deblurring task using a *single* forward model. The top row shows the intermediate reconstructions over 8 iterations using the true model, while the bottom row shows the evaluation results when a small perturbation is added to the forward model (the Peak Signal-to-Noise Ratio of the true kernel to the noisy kernel is 40.9 dB). This quality degradation is due to the accumulation of errors in the forward model.

When the forward model is precisely known, the inverse problem in (2) can be solved via classical optimization methods, where r is predefined. Machine-learning approaches, as summarized in Arridge et al. (2019); Iqbal et al. (2023); Ongie et al. (2020), have demonstrated superior reconstruction performance by effectively learning a regularizer from data. For example, the Plug-and-Play method (Venkatakrishnan et al., 2013) trains a general denoiser independently of the forward model and iteratively minimizes (2) with the learned denoiser as the regularization updates. Another class of approaches, the *loop unrolling* (LU) method (Gregor & LeCun, 2010; Hershey et al., 2014; Adler & Öktem, 2018) and its extension using deep equilibrium models (Gilton et al., 2021a), build on the observation that many iterative algorithms for solving (2) can be re-expressed as a sequence of neural networks (Gregor & LeCun, 2010), which can then be trained to remove artifacts and noise patterns associated with a known $\mathcal{A}$, resulting in higher quality reconstructions.

While these model-based machine learning solvers have demonstrated impressive performance, they encounter challenges while dealing with inaccurate forward models. Plug-and-Play, LU, and its deep equilibrium extensions all entail a gradient update of the data-fidelity term, which can be in error when the forward model is inaccurate. This causes the learned denoiser or regularizer to be ineffective. In particular, while LU often exhibits state-of-the-art performance, has faster runtime, and has improved stability in training (Guan et al., 2022) compared to other approaches, it requires precise knowledge of the forward model. Fig. 1 demonstrates the sensitivity of LU to mismatch in $\mathcal{A}$.

Some works passively address errors in the $\boldsymbol{y}$-space (equivalent to model mismatch under some conditions) by enhancing the robustness of the neural network. For example, Krainovic et al. (2023) suggests that introducing jittering in the $\boldsymbol{y}$-space can achieve robustness in worst-case ℓ_2 perturbations. In Hu et al. (2023), a deep equilibrium solver is trained using various incorrect $\mathcal{A}$'s, demonstrating greater robustness to variations in $\mathcal{A}$ compared to the Plug-and-Play method. In contrast, our proposed approach involves an active strategy for mitigating model mismatch. The experimental results in Section 5 illustrate the effectiveness of our method compared to the same LU network that merely learns a robust mapping passively.

Model mismatches in linear inverse problems can be more directly resolved by alternatively updating the model parameters and the underlying signal. For instance, Fergus et al. (2006); Cai et al. (2009); Levin et al. (2009); Cho & Lee (2009) reconstructed the latent images with updates in the forward model solved directly from least squares. However, these methods are typically limited to linear cases with predefined regularizers. To improve performance in this setting, Nan & Ji (2020); Gilton et al. (2021b) learn a regularizer from data, resulting in improved performance on linear inverse problems. Our work further introduces an untrained residual network that can learn the forward model more flexibly to handle nonlinear problems and hence has broader scientific applications. Existing LU approaches learn a black-box solver that treats the provided $\mathcal{A}_0$ as if it is the true forward model, while the proposed method updates the forward model and the reconstruction jointly, providing both a solution to the IP and an estimate of the true forward model.

Our contributions in this paper can be summarized as follows:

- **General model-based architecture in handling model-mismatch:** We present a general model-based architecture for solving IPs when an approximation of the forward model is available. Unlike classical model-based architectures which require precise knowledge of $\mathcal{A}$, we propose two variants of the model-based algorithms that can iteratively update the forward model along with the reconstruction.
- **Introducing untrained neural network for model-mismatch:** We present a novel approach by incorporating an untrained neural network that can match the data consistency in the measurement domain within a model-based architecture to accommodate model-mismatch in both linear and nonlinear IPs. We demonstrate that a random initialization of this network during evaluation maintains the same level of reconstruction.
- **Proof of convergence and empirical validation:** We establish the convergence of the proposed algorithm under mild conditions and empirically verify its convergence using a Deep Equilibrium Model (DEQ) structure.
- **Improved performance in linear and nonlinear IP tasks:** In contrast to model-based architectures passively trained to be robust to model mismatch, our methods exhibit substantial improvement in image blind deblurring (linear), seismic blind deconvolution (linear), and landscape defogging (nonlinear) tasks.
- **Effectiveness in iterative reconstructions and robustness:** We show that proposed algorithms lead to more effective intermediate reconstructions. They also exhibit robustness to different random initializations of the residual block and are more stable in scenarios involving a higher number of iterations during evaluation.

2 Background

2.1 Loop Unrolling Methods

The traditional approach to solving inverse problems is by formulating them as an optimization problem of the form in (2). A natural approach to solving (2) is via a *proximal gradient descent* algorithm, which can be applied even when the regularizer r is not differentiable, as is often the case (Parikh & Boyd, 2014). The resulting update involves first taking a gradient step (with a fixed step size η) that aims to minimize the data-fidelity term in (2). This is then followed by a proximal step, resulting in the iteration (Parikh & Boyd, 2014):

$$\boldsymbol{x}_{k+1} = \text{prox}_{\gamma,r}(\boldsymbol{x}_k + \eta\mathcal{A}^*(\boldsymbol{y} - \mathcal{A}(\boldsymbol{x}_k))), \tag{3}$$

where $\mathcal{A}^*$ denotes the adjoint operator of $\mathcal{A}$, $\gamma > 0$ is again the parameter that controls the weight of the regularizer, and the proximal operator of a function g is defined as:

$$\text{prox}_g(\boldsymbol{x}) = \arg\min_{\boldsymbol{z}} \ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2 + g(\boldsymbol{z}). \tag{4}$$

As we can see, the choice of regularizer manifests itself entirely through the proximal operator. The LU algorithm essentially keeps the update in (3) but replaces the proximal operator with a neural network, and limits the algorithm to a finite number of iterations K. The final output $\boldsymbol{x}_K$ is compared with the ground truth $\boldsymbol{x}$ and the network parameters are updated accordingly through *end-to-end* training.

By structuring the network in a way that mirrors proximal gradient descent, and taking advantage of an accurate descent direction derived from the knowledge of $\mathcal{A}$, the learned portion of the network can be interpreted as the proximal operator of a learned regularizer that enforces desired signal structures. However, when the forward model is inexact or only approximately known, the gradient update in (3) can introduce errors. Since LU is trained end-to-end, the error will manifest itself in the learned proximal operators. As a result, the neural network used in LU will no longer act as a pure proximal operator since it must both enforce signal structure as well as compensate for the errors in $\mathcal{A}$, potentially becoming less effective and interpretable.

2.2 Deep Equilibrium Models

Deep equilibrium models (DEQ) form a class of implicit neural networks, where the output is the fixed point solution of a neural network block Bai et al. (2019). For an initial input $\boldsymbol{x}^{(0)}$, and a neural network ϕ_θ, a DEQ repeatedly applies ϕ_θ,

$$\boldsymbol{x}^{(k+1)} = \phi_\theta(\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(k)}),$$

until convergence to an equilibrium solution. The forward pass can be performed using any efficient root-finding algorithm. The backward pass, instead of saving all computational graphs to do backpropagation in one pass (also known as end-to-end training), can be performed by either direct computation of the vector-Jacobian product (Bai et al., 2019) or using the "Jacobian-free" strategy proposed in Fung et al..

Later work in Gilton et al. (2021a) extends LU using a DEQ architecture, so instead of having a pre-determined fixed number of iterations, this work allows a potentially high number of iterations until a fixed-point solution is reached. Using DEQ shows a higher reconstruction quality, lower memory usage, and observe consistent improvement in reconstruction.

2.3 Half-Quadratic Splitting (HQS)

Another key tool that we will rely on in our approach is *variable splitting*. Variable splitting is an iterative optimization method that solves problems where the objective function is a sum of multiple components (Geman & Yang, 1995; Nikolova & Ng, 2005; Bergmann et al., 2015; Hurault et al., 2022; Yang & Wang, 2017). It works by introducing an auxiliary variable and iteratively optimizing the objective function with respect to each variable while fixing the others. For example, by introducing an auxiliary variable $\boldsymbol{z}$, we can re-express the optimization problem in (2) as

$$\min_{\boldsymbol{x},\boldsymbol{z}} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2 + \gamma r(\boldsymbol{z}), \text{ s.t. } \boldsymbol{x} = \boldsymbol{z}.$$

The solution can be approximated using the following unconstrained problem,

$$\min_{\boldsymbol{x},\boldsymbol{z}} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2 + \gamma r(\boldsymbol{z}) + \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{z}\|_2^2,$$

where $\mu \geq 0$ is a tuning parameter. This optimization can be solved by iteratively updating $\boldsymbol{x}_k$ and $\boldsymbol{z}_k$ until convergence:

$$\begin{aligned}
\boldsymbol{x}_{k+1} &= \arg\min_{\boldsymbol{x}} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2 + \frac{\mu}{2}\|\boldsymbol{x} - \boldsymbol{z}_k\|_2^2, \\
\boldsymbol{z}_{k+1} &= \arg\min_{\boldsymbol{z}} \ \gamma r(\boldsymbol{z}) + \frac{\mu}{2}\|\boldsymbol{z} - \boldsymbol{x}_{k+1}\|_2^2.
\end{aligned}$$

3 Proposed Method I: $\mathcal{A}$-adaptive Loop Unrolling Architecture

The discussion above has assumed exact knowledge of $\mathcal{A}$. Here we now suppose that we have an initial guess of the forward model $\mathcal{A}_0$ and consider a neural network f_θ that learns the measurement residual due to model misfit given the signal $\boldsymbol{x}$ and $\mathcal{A}_0$, i.e.,

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{\epsilon} = \mathcal{A}_0(\boldsymbol{x}) + f_\theta(\boldsymbol{x}, \mathcal{A}_0) + \boldsymbol{\epsilon}.$$

We assume that $\mathcal{A}_0$ is a useful estimate of the true forward model, in other words, $\mathcal{A}(\boldsymbol{x})$ and $\mathcal{A}_0(\boldsymbol{x})$ are close. We can then express the optimization problem in (2) as

$$\min_{\boldsymbol{x},\theta} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}_0(\boldsymbol{x}) - f_\theta(\boldsymbol{x}, \mathcal{A}_0)\|_2^2 + \gamma r(\boldsymbol{x}) + \tau\|f_\theta(\boldsymbol{x}, \mathcal{A}_0)\|_2^2. \tag{5}$$

Introducing an auxiliary variable $\boldsymbol{z}$, the solution of (5) is equivalent to solving

$$\min_{\boldsymbol{x},\boldsymbol{z},\theta} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}_0(\boldsymbol{z}) - f_\theta(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \gamma r(\boldsymbol{x}) + \tau\|f_\theta(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \lambda\|\boldsymbol{x} - \boldsymbol{z}\|_2^2. \tag{6}$$

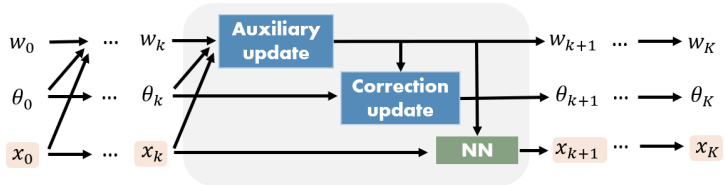

Figure 2: Illustration of the kth iteration of an $\mathcal{A}$-adaptive LU network. $\boldsymbol{x}_0$ is fed into the network, the auxiliary update and the correction update corresponding to updates in θ and $\boldsymbol{z}$ respectively, and the proximal network in green is updated using end-to-end training. The final output contains the parameters for estimating the function mismatch and the reconstruction estimate.

Similar to the HQS updates, we first initialize $\boldsymbol{x}_0$, $\boldsymbol{z}_0$, and θ_0. We then update each variable in the objective function by keeping other variables fixed. For $k = 1, 2, \ldots, K$, we have

$$\begin{aligned}
\boldsymbol{z}_{k+1} &= \arg\min_{z} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}_0(\boldsymbol{z}) - f_{\theta_k}(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \tau\|f_{\theta_k}(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \lambda\|\boldsymbol{x}_k - \boldsymbol{z}\|_2^2, \\
\theta_{k+1} &= \arg\min_{\theta} \ \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}_0(\boldsymbol{z}_{k+1}) - f_{\theta}(\boldsymbol{z}_{k+1}, \mathcal{A}_0)\|_2^2 + \tau\|f_{\theta}(\boldsymbol{z}_{k+1}, \mathcal{A}_0)\|_2^2, \\
\boldsymbol{x}_{k+1} &= \mathrm{prox}_{\frac{\gamma}{2\lambda}, r}(\boldsymbol{x}_k - \eta(\boldsymbol{x}_k - \boldsymbol{z}_{k+1})).
\end{aligned} \tag{7}$$

It should be noted that the update on $\boldsymbol{z}$ no longer has a closed-form solution due to the nonlinearity of f_θ. However, this can be efficiently computed using Autograd in Pytorch (Paszke et al., 2017) or other differentiation computing algorithms. Meanwhile, the update on θ follows the regular backpropagation for network parameters. We can then replace the proximal operator with a neural network, and the update on $\boldsymbol{x}$ connects all the components to form an LU network as illustrated in Fig. 2. The parameters in the proximal network, are learned through end-to-end training, where η is a trainable step size.

While we refer to f_θ as a network in the context of learning forward model mismatch, it differs from classical training approaches where weights remain fixed during evaluation. In our framework, the parameters θ in f are updated both during training and evaluation for each instance. The objective is to iteratively align the data fidelity term for a specific measurement $\boldsymbol{y}$ through a nonlinear estimation f with the minimal ℓ_2-norm. Our proposed technique involves using untrained (random) weights θ_0 at the initial iteration and adjusting θ_k based on the loss outlined in the second line of (7). The concept of employing an untrained neural network is reminiscent of Deep Image Prior (Ulyanov et al., 2018), which initializes a reconstruction network randomly, treating the weights as an implicit prior for the reconstruction. In our approach, however, we suggest using an untrained neural network as an integral part of the reconstruction process to address model mismatch while still being capable of learning regularization updates from all training data. Subsequent experiments demonstrate that during evaluation, the residual network f can be initialized with various untrained weights, resulting in different performance levels.

Note that the proposed iterative updates look reminiscent of the alternating direction method of multipliers (ADMM), like Yang et al. (2016). When the forward model is precisely known, ADMM aims to minimize the Lagrangian of the objective function with an additional auxiliary variable. The proposed algorithm uses HQS as a simpler method to split variables in solving an optimization problem, but the algorithm could easily be generalized to other solvers such as ADMM.

4 Proposed Method II: $\mathcal{A}$-adaptive Deep Equilibrium Architecture

Loop unrolling algorithms typically restrict the number of iterations to a small value during end-to-end training, primarily to alleviate the high memory costs associated with processing a large number of steps. In such cases, the focus often leans more toward managing computational resources rather than achieving convergence. However, when it comes to another well-known model-based architecture the deep equilibrium model (DEQ) extension, the situation differs. The DEQ extension seeks a fixed-point solution, which requires

careful consideration of convergence to ensure the model's effectiveness in finding the desired equilibrium. In this section, we first prove the convergence of the proposed iterative updates in (7) under mild conditions and thus propose an extension of $\mathcal{A}$-adaptive LU through the integration of a deep equilibrium model, namely $\mathcal{A}$-adaptive DEQ.

Proposition 4.1. *Assume r is convex, and use the fact that $||\boldsymbol{x}-\boldsymbol{z}||_2^2$ is $L-$smooth with respect to $\boldsymbol{x}$, in another word let $f(\boldsymbol{x}) = ||\boldsymbol{x}-\boldsymbol{z}||_2^2$, $||\nabla f(\boldsymbol{x}_1) - \nabla f(\boldsymbol{x}_2)||_2 \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2||_2$ for some $L > 0$. The algorithm in (7) converges when the step size $\eta_k = 1/L$ for all $k \geq 1$.*

Proof. Let $J(\boldsymbol{z}, \theta, \boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{y} - \mathcal{A}_0(\boldsymbol{z}) - f_\theta(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \gamma r(\boldsymbol{x}) + \tau\|f_\theta(\boldsymbol{z}, \mathcal{A}_0)\|_2^2 + \lambda\|\boldsymbol{x} - \boldsymbol{z}\|_2^2$. For a fixed $\boldsymbol{x}_k$ and θ_k, $J(\boldsymbol{z}_{k+1}, \theta_k, \boldsymbol{x}_k) \leq J(\boldsymbol{z}_k, \theta_k, \boldsymbol{x}_k)$ because we are minimizing over $\boldsymbol{z}$. Similarly, given fixed $\boldsymbol{z}_{k+1}$ and $\boldsymbol{x}_k$, $J(\boldsymbol{z}_{k+1}, \theta_{k+1}, \boldsymbol{x}_k) \leq J(\boldsymbol{z}_{k+1}, \theta_k, \boldsymbol{x}_k)$ when minimizing over θ. When fixing θ_{k+1} and $\boldsymbol{z}_{k+1}$, from the convergence result of the proximal gradient method (Parikh & Boyd, 2014), $J(\boldsymbol{z}_{k+1}, \theta_{k+1}, \boldsymbol{x}_{k+1}) \leq J(\boldsymbol{z}_{k+1}, \theta_{k+1}, \boldsymbol{x}_k) - \frac{1}{2L}||L(\boldsymbol{x}_k - \boldsymbol{x}_{k+1})||_2^2 \leq J(\boldsymbol{z}_{k+1}, \theta_{k+1}, \boldsymbol{x}_k)$. Therefore, the value of the objective function decreases for updates of $\boldsymbol{z}_k, \theta_k, \boldsymbol{x}_k$ at all k. In addition, because this value is also bounded below, the algorithm converges. □

In contrast to employing a fixed number of iterations K in $\mathcal{A}$-adaptive LU, as outlined in equation (7), $\mathcal{A}$-adaptive DEQ allows $\boldsymbol{x}_k$ to converge to a fixed point solution. This convergence is achieved over a potentially large number of iterations, motivated by the convergence property established in Proposition 4.1.

We denote $F_{\rho,\eta}$ as the function representing one iterative update of variables $\boldsymbol{z}_k$, θ_k and $\boldsymbol{x}_k$ in (7), where ρ parameterizes the proximal neural network and η denotes the trainable step size. According to Proposition 4.1, for a sufficiently large number of iterations k_0, a fixed point solution is reached for all $k > k_0$,

$$\boldsymbol{z}_k, \theta_k, \boldsymbol{x}_k = F_{\rho,\eta}(\boldsymbol{z}_k, \theta_k, \boldsymbol{x}_k).$$

$\mathcal{A}$-adaptive DEQ shares the same architecture as $\mathcal{A}$-adaptive LU, where f is still an untrained neural network and the proximal operator is replaced using a neural network, but it is trained differently. Following the training scheme introduced in Gilton et al. (2021a), the forward pass of $\mathcal{A}$-adaptive DEQ solves for a fixed point solution, which can be efficiently achieved using Anderson acceleration (Walker & Ni, 2011). Whereas in the backward pass, we adopted the "Jacobian-free" backpropagation strategy stated in Fung et al.. We show in the experiments that the $\mathcal{A}$-adaptive DEQ performs even better than $\mathcal{A}$-adaptive LU due to a higher number of effective iterations. In addition, we verify its convergence during evaluation.

5 Experiments and Discussion

In this section, we demonstrate the efficacy of the proposed $\mathcal{A}$-adaptive LU algorithm and show an improvement in reconstruction by learning a more accurate forward model compared to the following baseline methods: 1) a neural network that maps from the initial reconstruction $\boldsymbol{x}_0$ to the ground-truth $\boldsymbol{x}$, referred to as a direct inverse mapping, which is independent of the forward operator, and 2) a robust LU that is trained with inexact $\mathcal{A}_0$'s from the initial $\boldsymbol{x}_0$. Notice that $\boldsymbol{x}_0$ is initialized with $\mathcal{A}_0^*\boldsymbol{y}$ when $\boldsymbol{x}$ and $\boldsymbol{y}$ are from different spaces or $\boldsymbol{x}_0 = \boldsymbol{y}$ if they are from the same space. We evaluate the algorithms on linear IP tasks (image blind deblurring and seismic blind deconvolution), and a nonlinear IP task (landscape defogging). For each task, the direct inverse network and the proximal network in LU and the proposed methods share the same architecture.

Notice that in linear IPs, the forward model is defined by a matrix in a specific dimension, so the true forward model can be determined by either solving for the least squares solution or by using a gradient descent update inside the loop unrolling method. Our experiments first show the proposed methods work in linear cases, then more interestingly, when no precise formulation of the true forward model is known in nonlinear IPs (infeasible to solve directly), our proposed method shows its effectiveness in reconstruction.

Image Blind Deblurring In this problem, we aim to remove the blur from images $\boldsymbol{y}$ when a small amount of noise is present. The forward model is defined by a Gaussian blur kernel where we have inaccurate

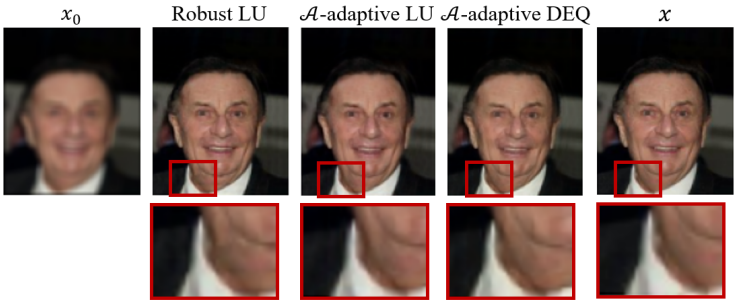

Figure 3: Comparing the deblurring results using robust LU, $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ to the ground truth, where $\boldsymbol{x}_0$ is the initial blurry images and $\boldsymbol{x}$ is the ground truth. The bottom row shows the zoomed regions in red boxes. The proposed methods generate sharper edges.

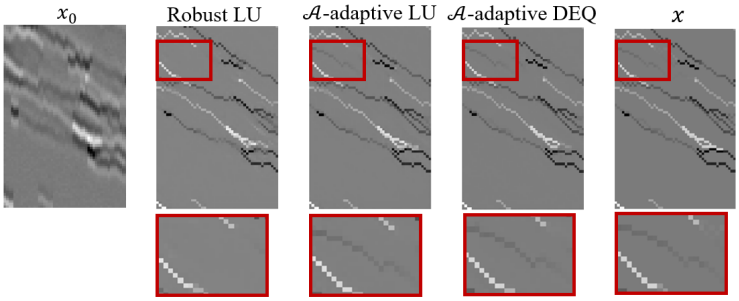

Figure 4: Comparing the deconvolution results using robust LU, $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ to the ground truth. The bottom row shows the zoomed regions in red boxes. A seismic layer in the red box is missing in reconstruction using the baseline robust LU method.

knowledge of the variance and size of the kernel. To validate this approach, experiments are conducted on the CelebA dataset (Liu et al., 2015), which was resized to $3 \times 120 \times 100$. The blurry images are generated with different Gaussian kernels for each pair of data $(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{A}_{0,i})$.

Seismic Blind Deconvolution Here an acoustic wave generated by a vibroseis truck on the surface of the earth propagates through the earth's layers. The reflected signals are collected by the geophones and stored as raw measurements. The measurements $\boldsymbol{y}$ are obtained in the form of (1). However, the frequencies of the acoustic wave are often inaccurately recorded due to the limited resolution and noise in the measurement process (Zabihi Naeini & Sams, 2017), which leads to an inaccurate estimate of the wavelet in the forward model. The forward process can be viewed as a convolution between the acoustic wave and the layer reflectivity $\boldsymbol{x}$. Therefore, the goal of seismic deconvolution is to reconstruct the layer reflectivities $\boldsymbol{x}$, with the inaccurate estimate of the wavelet in the forward model taken into account. The true model for each data pair $(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{A}_{0,i})$ can be expressed as follows,

$$\boldsymbol{y}_i = \boldsymbol{A}_{0,i}\boldsymbol{x}_i + f_\theta(\boldsymbol{x}_i, \boldsymbol{w}_i) + \boldsymbol{\epsilon}_i,$$

where $\boldsymbol{A}_{0,i}$ is a Toeplitz matrix derived from the inexact source wavelet $\boldsymbol{w}_i$. The measurement is simulated by applying an inaccurate wavelet with small additive Gaussian noise to the forward model. Notice that noise added to the true model may result in artifacts due to an extra magnification factor applied by $\boldsymbol{x}_i$, which is distinct from the measurement noise $\boldsymbol{\epsilon}_i$. The data is generated following the procedure in Iqbal et al. (2019).

Landscape Image Defogging We also demonstrate the proposed algorithms with a nonlinear IP. Consider the problem of removing the effects of fog, haze or mist from an image $\boldsymbol{y}$ to restore its true appearance

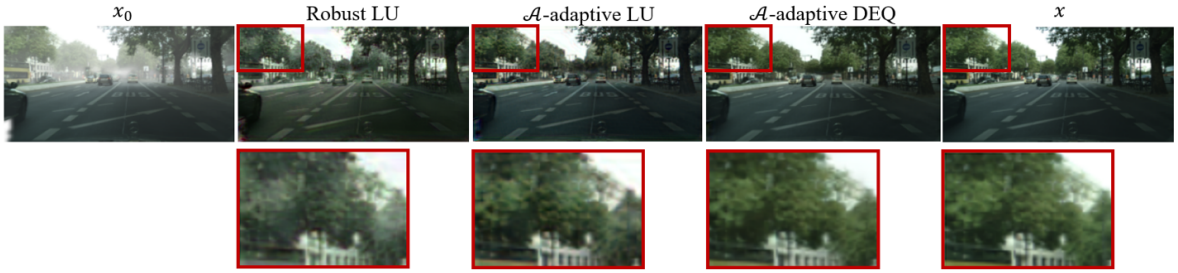

Figure 5: Comparing the defogging results using robust LU, $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ to the ground truth. The bottom row shows the zoomed regions in red boxes. The proposed methods reproduce cleaner images.

$\boldsymbol{x}$. Examples of $\boldsymbol{y}$ and $\boldsymbol{x}$ are illustrated in Fig. 5. The degradation process can be modeled as follows,

$$\boldsymbol{y} = \boldsymbol{x} \odot \boldsymbol{T}(\boldsymbol{x}) + \boldsymbol{L}(\boldsymbol{x}) \odot (\boldsymbol{I} - \boldsymbol{T}(\boldsymbol{x})) + \boldsymbol{\epsilon}, \tag{8}$$

where $\boldsymbol{T} \in [0,1]^{3\times m\times n}$ is an unknown transmission map as a function of the fog-free image $\boldsymbol{x}$, m and n denote the width and height of the image. $\boldsymbol{T}(\boldsymbol{x})$ is then applied pointwise ($\odot$) to $\boldsymbol{x}$. $\boldsymbol{L}(\boldsymbol{x})$ is the unknown atmospheric light profile (Tufail et al., 2019). The goal is to estimate the model mismatch and recover the clean image. Since $\boldsymbol{T}$ and $\boldsymbol{L}$ are unknown functions with respect to $\boldsymbol{x}$, we use two separate networks to learn transmission mapping and the light profile respectively. This example shows the flexibility of our proposed methods. Instead of having one single network f_θ that learns the residual, if we know the explicit formulation of the true model, we can customize the residual network in a more structured way. To evaluate the proposed approachs, the CityScapes - Depth and Segmentation dataset given in Cordts et al. (2016) is used. This dataset provides depth maps, as well as foggy and clean views of urban street scenes.

5.1 Training Details

The proximal networks for LU and $\mathcal{A}$-adaptive LU are kept the same for each task, with 5-layer DnCNN for image deblurring and 7-layer DnCNN for landscape defogging. In the forward mismatch network, $\mathcal{A}_0(\boldsymbol{x})$ combines with $\mathcal{A}_0$ features, processed through a 3-layer CNN. $\mathcal{A}_0$ features comprise an incorrect Gaussian kernel for deblurring and landscape depth profiles. Because the initialization of f_θ is less important, the parameters θ are not reinitialized for each training instance for better time efficiency. Both LU and $\mathcal{A}$-adaptive LU employ a maximum of $K = 5$ iterations. The hyper-parameters in $\mathcal{A}$-adaptive LU (6) are chosen via sensitivity analysis, where $\lambda = 0.01$, $\tau = 0.1$ for deblurring, and $\lambda = 0.0001$, $\tau = 0.1$ for defogging. Learning rates are set at $= 0.0001$ for updating $\boldsymbol{z}$ and network parameters. All networks are trained on NVIDIA RTX 3080 with 10GB of RAM.

5.2 Reconstruction Results

Table 1 compares the average testing peak signal-to-noise ratio (PSNR) in dB and Structural Similarity Index Measure (SSIM) of the reconstructions. The neural network learns a direct inverse regardless of $\mathcal{A}_{0,i}$, which is the worst in reconstruction. It is notable that, even with some degree of inaccuracy in $\mathcal{A}$, the robust LU still outperforms the direct inverse mapping. This is because end-to-end training enables LU to fix the gradient-step errors resulting from the model mismatch to some extent, thereby maintaining reasonably higher-quality results. Our results align with the finding in Krainovic et al. (2023) that the robust LU trains an inverse mapping that is less sensitive to perturbations in the $\boldsymbol{y}$-space, by seeing inaccuracies in the forward model at train time. However, in the next section, we will discuss the intermediate reconstruction ineffectiveness of robust LU compared to the proposed methods.

The proposed methods make an improvement by actively adjusting for the model mismatch for each data pair. Instead of having the neural network learn both the proximal step and the model correction as in a black box, the proposed methods separate the tasks and each part of the network is now learning a clearer

and easier task, resulting in a substantial performance gain over both the robust LU and the direct inverse mapping in all tasks. Furthermore, $\mathcal{A}$-adaptive DEQ allows for a higher number of iterations than $\mathcal{A}$-adaptive LU, resulting in superior performance in both PSNR and SSIM.

Table 1: Average testing PSNR and SSIM for direct inverse mapping, robust LU, $\mathcal{A}$-adaptive LU, and $\mathcal{A}$-adaptive DEQ. The best performances for each task are in bold and the second best performances are underlined.

		Deblur.	Deconv.	Defog.
Direct Inverse	PSNR	24.002	22.816	21.802
	SSIM	0.7689	0.7421	0.8738
Robust LU	PSNR	34.380	24.785	29.645
	SSIM	0.9429	0.8407	0.9563
$\mathcal{A}$-adaptive LU (proposed meth. I)	PSNR	<u>36.621</u>	<u>27.271</u>	<u>31.112</u>
	SSIM	<u>0.9731</u>	<u>0.8960</u>	<u>0.9661</u>
$\mathcal{A}$-adaptive DEQ (proposed meth. II)	PSNR	**37.809**	**28.236**	**32.149**
	SSIM	**0.9774**	**0.9082**	**0.9723**

Apart from the quantitative evaluation, we offer a visual comparison of the reconstruction results using the robust LU and the proposed methods for the presented tasks in Figs. 3, 4, and 5. In blind image deblurring, Fig. 3 shows that the proposed methods preserve a sharper jawline. In seismic blind deconvolution, the baseline method (Robust LU) fails to reconstruct a seismic layer as highlighted in red boxes in Fig. 4, whereas both proposed methods successfully capture the layer. Furthermore, in the landscape defogging example, Fig. 5 demonstrates the proposed methods remove more artifacts while preserving more detailed information.

5.3 Effectiveness of Reconstruction

We also explore the effectiveness of reconstruction by comparing the intermediate reconstruction $\boldsymbol{x}_k$'s in the robust LU and $\mathcal{A}$-adaptive LU in Fig. 6. The average testing Mean-Squared Error (MSE) between $\boldsymbol{x}_k$'s and the ground truth $\boldsymbol{x}$ are recorded for all $k = 1, ..., 5$. While LU is trained to be robust to mismatch in $\mathcal{A}$'s, the intermediate MSE of the robust LU remains high until the final iteration. In fact, at iteration 3 in the image deblurring task and iteration 2 and 4 in the landscape defogging task, we observe noticeable MSE increments in the robust LU (solid blue curve). It indicates that the robust LU lacks the capacity to learn how to effectively correct the errors in a single iteration. Because the proximal network in the robust LU aims to perform both the proximal step and the model mismatch, it learns a less interpretable mapping from the erroneous gradient step to the true $\boldsymbol{x}$. In contrast, $\mathcal{A}$-adaptive LU learns model mismatch and the proximal step separately. It thus allows effective error correction and quality improvement in every iteration, resulting in a consistent MSE decrement.

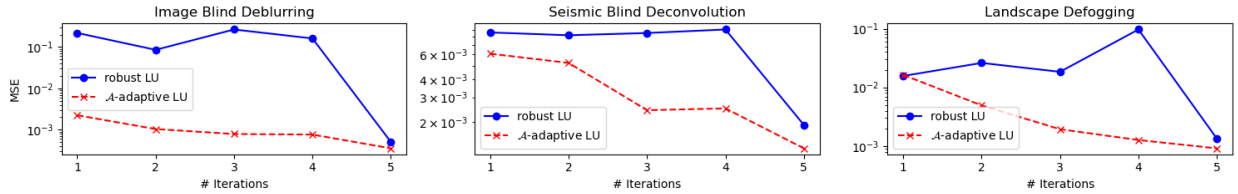

Figure 6: Average testing MSE of the intermediate reconstructions for the 5-iteration robust LU and the proposed $\mathcal{A}$-adaptive LU.

5.4 Intermediate Reconstructions of $\mathcal{A}$-adaptive DEQ

We show the intermediate reconstructions of the DEQ variant of the proposed method in Fig. 7. The proposed network is trained with a maximum of 30 iterations or when a fixed point solution is reached, whereas when evaluating $\mathcal{A}$-adaptive DEQ with more iterations (i.e., 100), the reconstruction error remains low. This result aligns with the finding in Gilton et al. (2021a).

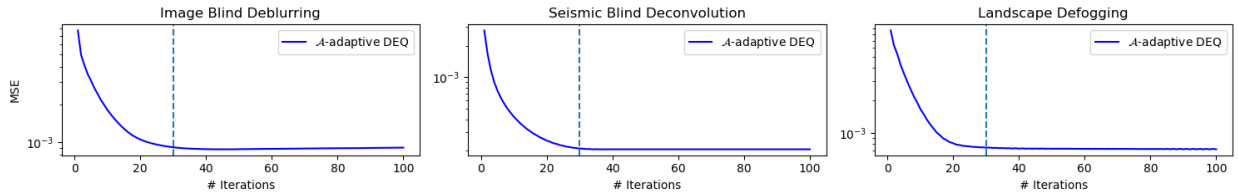

Figure 7: Average testing MSE of the intermediate reconstructions for the proposed $\mathcal{A}$-adaptive DEQ. A max of 30 iterations is used in training, this figure depicts the extended evaluation result when allowing a max of 100 iterations.

5.5 Robustness of $\mathcal{A}$-adaptive LU with More Iterations

In this section, we empirically assess the stability of our proposed $\mathcal{A}$-adaptive LU variant by evaluating it with a greater number of unrolling iterations than those employed during training. Across all three tasks, a noticeable Mean Squared Error (MSE) gap emerges between the two curves. The solid blue curve, representing robust LU, exhibits a drop in MSE at iteration 5, which corresponds to the target output during training. However, as the number of iterations exceeds the max iterations of training (in this set of experiments $k > 5$), the MSE rises significantly. Notice that although the MSE at iteration 10 is relatively low for the landscape defogging task, the reconstruction is not stable and less predictable. In contrast, the dashed red curve ($\mathcal{A}$-adaptive LU) oscillates around the lowest MSE. This observation highlights the enhanced robustness of the proposed $\mathcal{A}$-adaptive LU when subjected to a higher number of iterations during evaluation.

5.6 Random Initialization of the Mismatch Network

The residual network f_θ is updated along the reconstruction in both training and evaluation for each data instance. Because the update of θ in (7) aims to minimize the data-fidelity term with consideration of the forward model mismatch in the smallest ℓ_2-norm, the initialization of θ is less important. In this section, we show that f_θ can be initialized by an untrained neural network at even evaluation time, and still obtain the same level of performance in both proposed architectures. We evaluate the PSNR and SSIM of reconstruction results from both proposed methods using a saved model f_θ, and two commonly used neural

Table 2: Average testing PSNR and SSIM for $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ using f_θ with the saved model, uniform random initialization, and Xavier initialization.

	PSNR/SSIM	$\mathcal{A}$-adaptive LU	$\mathcal{A}$-adaptive DEQ
Deblurring	saved model	36.621/0.9731	37.809/0.9774
	uniform	36.604/0.9730	37.808/0.9774
	Xavier	36.606/0.9731	37.809/0.9773
Deconvolution	saved model	27.271/0.8960	28.236/0.9082
	uniform	27.268/0.8959	28.239/0.9084
	Xavier	27.272/0.8960	28.238/0.9084
Defogging	saved model	31.112/0.9661	32.149/0.9723
	uniform	31.277/0.9661	32.147/0.9722
	Xavier	31.276/0.9661	32.147/0.9723

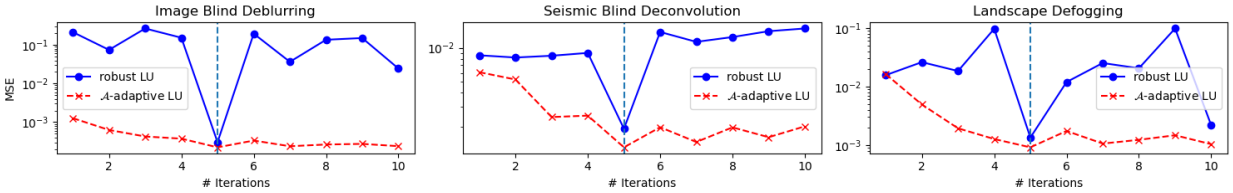

Figure 8: Average testing MSE of intermediate reconstructions using the robust LU and $\mathcal{A}$-adaptive LU with 10 unrolling iterations, while trained on 5 unrolling iterations.

network initialization methods: Kaiming uniform (He et al., 2015) and Xavier (Glorot & Bengio, 2010) initialization, as shown in Table 2.

5.7 Discussion of Evaluation Runtime

While the suggested methods demonstrate superior numerical and visual reconstruction qualities, they do incur a runtime tradeoff to be more accurate. Table 3 compares the average and standard deviation of runtime per batch during evaluation. The same batch sizes are maintained across all three methods for each task to ensure comparable computational overhead. The loop unrolling variant of the proposed method is not significantly slower than the robust LU, despite the $\mathcal{A}$-adaptive LU involving extra optimization steps within the reconstruction process compared to the one-step gradient update in the robust LU. However, the DEQ variant is noticeably slower due to its high number of iterations in solving for a fixed point solution. The $\mathcal{A}$-adaptive DEQ method also has higher variance because the number of iterations is not fixed due to the design of DEQs.

Table 3: Average and standard deviation of runtime in ms during evaluation for robust LU, $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ. The same batch size is enforced for all methods across each task.

Tasks	Methods	Evaluation time (ms)/batch
Deblurring (batch size = 32)	robust LU	80.26 ± 0.52
	$\mathcal{A}$-adaptive LU	253.06 ± 0.82
	$\mathcal{A}$-adaptive DEQ	2219.53 ± 15.89
Deconvolution (batch size = 32)	robust LU	81.14 ± 0.67
	$\mathcal{A}$-adaptive LU	256.94 ± 0.85
	$\mathcal{A}$-adaptive DEQ	2249.60 ± 28.37
Defogging (batch size = 8)	robust LU	112.91 ± 1.03
	$\mathcal{A}$-adaptive LU	370.93 ± 1.14
	$\mathcal{A}$-adaptive DEQ	2433.66 ± 15.79

6 Conclusion

While the model-based machine learning IP solvers are powerful, they demand precise knowledge of the forward models, which can be impractical in many applications. One way to handle the model mismatch is to train a model with inaccurate forward models, which will implicitly correct the errors due to end-to-end training and demonstrate reasonable performance. However, this falls short in terms of interpretability and efficacy in reconstruction. To address this problem, we introduce a novel procedure to adapt for the forward model mismatches using untrained neural networks actively based on two well-known model-based machine learning architectures, denoted as $\mathcal{A}$-adaptive LU and $\mathcal{A}$-adaptive DEQ. Experimental results across both linear and nonlinear inverse problems showcase the effectiveness of the proposed methods in learning forward model updates. Consequently, they surpass the baseline methods significantly when trained as a robust reconstruction mapping to accommodate variations in $\mathcal{A}$. We further demonstrate the robustness of the proposed methods to random initialization of the residual block and a higher number of iterations during

evaluation. Finally, we show the accuracy-runtime tradeoff in handling model mismatch using the proposed variants of the model-based architectures.

Broader Impact Statement

This work adaptively addresses the errors in the forward model by optimizing the auxiliary variable and the untrained mismatch neural network within a model-based architecture. We show the flexibility of the proposed forward model matching module using two well-known architectures. However, the accuracy-runtime tradeoff still exists. Although the proposed $\mathcal{A}$-adaptive DEQ variant achieves the best performance for all tasks, when solving large-scale inverse problems, careful consideration of runtime is necessary.

References

Jonas Adler and Ozan Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

Simon Arridge, Peter Maass, Ozan Öktem, and Carola-Bibiane Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in Neural Information Processing Systems*, 32, 2019.

Ronny Bergmann, Raymond H Chan, Ralf Hielscher, Johannes Persch, and Gabriele Steidl. Restoration of manifold-valued images by half-quadratic minimization. *arXiv preprint arXiv:1505.07029*, 2015.

Jian-Feng Cai, Hui Ji, Chaoqiang Liu, and Zuowei Shen. Blind motion deblurring from a single image using sparse approximation. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 104–111. IEEE, 2009.

Sunghyun Cho and Seungyong Lee. Fast motion deblurring. In *ACM SIGGRAPH Asia 2009 papers*, pp. 1–8. 2009.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3213–3223, 2016.

Rob Fergus, Barun Singh, Aaron Hertzmann, Sam T Roweis, and William T Freeman. Removing camera shake from a single photograph. In *Acm Siggraph 2006 Papers*, pp. 787–794. 2006.

SW Fung, H Heaton, Q Li, D McKenzie, S Osher, and W Yin. Fixed point networks: Implicit depth models with jacobian-free backprop. *arXiv e-print. arXiv:2103.12803, 2021.*

D. Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995. doi: 10.1109/83.392335.

Davis Gilton, Gregory Ongie, and Rebecca Willett. Deep equilibrium architectures for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:1123–1133, 2021a.

Davis Gilton, Gregory Ongie, and Rebecca Willett. Model adaptation for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:661–674, 2021b.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on Machine Learning*, pp. 399–406, 2010.

Peimeng Guan, Jihui Jin, Justin Romberg, and Mark A Davenport. Loop unrolled shallow equilibrium regularizer (luser)–a memory-efficient inverse problem solver. *arXiv preprint arXiv:2210.04987*, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

John R Hershey, Jonathan Le Roux, and Felix Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *arXiv preprint arXiv:1409.2574*, 2014.

Junhao Hu, Shirin Shoushtari, Zihao Zou, Jiaming Liu, Zhixin Sun, and Ulugbek S Kamilov. Robustness of deep equilibrium architectures to changes in the measurement model. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Samuel Hurault, Arthur Leclaire, and Nicolas Papadakis. Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization. In *International Conference on Machine Learning*, pp. 9483–9505. PMLR, 2022.

Naveed Iqbal, Entao Liu, James H. Mcclellan, and Abdullatif A. Al-Shuhail. Sparse multichannel blind deconvolution of seismic data via spectral projected-gradient. *IEEE Access*, 7:23740–23751, 2019. doi: 10.1109/ACCESS.2019.2899131.

Naveed Iqbal, Mudassir Masood, Motaz Alfarraj, and Umair Bin Waheed. Deep seismic cs: A deep learning assisted compressive sensing for seismic data. *IEEE Transactions on Geoscience and Remote Sensing*, 61: 1–9, 2023. doi: 10.1109/TGRS.2023.3289917.

Anselm Krainovic, Mahdi Soltanolkotabi, and Reinhard Heckel. Learning provably robust estimators for inverse problems via jittering. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Understanding and evaluating blind deconvolution algorithms. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1964–1971. IEEE, 2009.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Magdi S Mahmoud and Haris M Khalid. Expectation maximization approach to data-based fault diagnostics. *Information Sciences*, 235:80–96, 2013.

Wail A Mousa and Abdullatif A Al-Shuhail. Processing of seismic reflection data using matlab™. *Synthesis lectures on signal processing*, 5(1):1–97, 2011.

Yuesong Nan and Hui Ji. Deep learning for handling kernel/model uncertainty in image deconvolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2388–2397, 2020.

Mila Nikolova and Michael K Ng. Analysis of half-quadratic minimization methods for signal and image recovery. *SIAM Journal on Scientific computing*, 27(3):937–966, 2005.

Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014. ISSN 2167-3888. doi: 10.1561/2400000003.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Zahid Tufail, Khawar Khurshid, Ahmad Salman, and Khurram Khurshid. Optimisation of transmission map for improved image defogging. *IET Image Processing*, 13(7):1161–1169, 2019.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, 2018.

Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pp. 945–948. IEEE, 2013.

Homer F Walker and Peng Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 49(4):1715–1735, 2011.

Xiaojuan Yang and Li Wang. Fast half-quadratic algorithm for image restoration and reconstruction. *Applied Mathematical Modelling*, 50:92–104, 2017.

Yan Yang, Jian Sun, Huibin Li, and Zongben Xu. Deep admm-net for compressive sensing mri. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

Ehsan Zabihi Naeini and Mark Sams. Accuracy of wavelets, seismic inversion, and thin-bed resolution. *Interpretation*, 5(4):T523–T530, 2017.