Deep unfolding as iterative regularization for imaging inverse problems

Zhuo-Xu Cui\(^1,6\), Qingyong Zhu\(^1,6\), Jing Cheng\(^2\), Bo Zhang\(^3,4\)\(\ast\) and Dong Liang\(^{1,2,5,\ast}\)

\(^1\) Research Center for Medical AI, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, People’s Republic of China
\(^2\) Paul C. Lauterbur Research Center for Biomedical Imaging, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, People’s Republic of China
\(^3\) LSEC and Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, People’s Republic of China
\(^4\) School of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, People’s Republic of China
\(^5\) Key Laboratory of Biomedical Imaging Science and System, Chinese Academy of Sciences, Shenzhen, People’s Republic of China

E-mail: dong.liang@siat.ac.cn

Received 4 May 2023; revised 10 December 2023
Accepted for publication 3 January 2024
Published 19 January 2024

Abstract
Deep unfolding methods have gained significant popularity in the field of inverse problems as they have driven the design of deep neural networks (DNNs) using iterative algorithms. In contrast to general DNNs, unfolding methods offer improved interpretability and performance. However, their theoretical stability or regularity in solving inverse problems remains subject to certain limitations. To address this, we reevaluate unfolded DNNs and observe that their algorithmically-driven cascading structure exhibits a closer resemblance to iterative regularization. Recognizing this, we propose a modified training approach and configure termination criteria for unfolded DNNs, thereby establishing the unfolding method as an iterative regularization technique. Specifically, our method involves the joint learning of a convex penalty function using an input-convex neural network to quantify distance to a real data manifold. Then, we train a DNN unfolded from the proximal gradient descent algorithm, incorporating this learned penalty. Additionally, we introduce a new termination criterion for the unfolded DNN. Under the assumption that the real data manifold intersects the solutions of the inverse problem with a unique real
solution, even when measurements contain perturbations, we provide a theoretical proof of the stable convergence of the unfolded DNN to this solution. Furthermore, we demonstrate with an example of magnetic resonance imaging reconstruction that the proposed method outperforms original unfolding methods and traditional regularization methods in terms of reconstruction quality, stability, and convergence speed.

Keywords: unfolding, deep learning, iterative regularization, accelerated MRI

1. Introduction

In the field of medical imaging, solving the inverse problem is a common and challenging task. For example, in magnetic resonance imaging (MRI), the objective is to reconstruct high-quality images from undersampled k-space data, reducing scan times and minimizing artifacts due to patient motion [31]. Similarly, in computed tomography (CT) reconstruction, the goal is to reconstruct CT images from undersampled angular projection data, thereby minimizing a patient’s exposure to x-ray radiation [6].

In recent years, deep learning (DL) has achieved remarkable advancements across various domains, sparking interest in leveraging DL to address inverse problems in a data-driven manner. This trend is clearly observable in DL-based CT, MRI, and positron emission tomography (PET) reconstruction [7, 9, 44, 46, 48, 52]. Early research primarily focused on the direct utilization of deep neural networks (DNNs) to establish the mapping between measurements and solutions for inverse problems. However, these methods exhibit limited interpretability and are susceptible to issues of instability [4]. The root cause of this instability can be attributed to the inherent complexity of solving inverse problems, marked by the presence of multiple solutions or discontinuities to the measurements. Previous studies have demonstrated the critical role of regularization in achieving stable solutions for inverse problems [13, 45]. Consequently, the incorporation of DL methods within the framework of regularization gained prominence, with unfolding methods emerging as particularly noteworthy [14, 49].

Unfolding methods were initially inspired by variational regularization models. In their early stages, these methods allowed for the adaptability of regularization parameters and parameters related to the regularizer, such as the order of differential operators [3]. This adaptability was achieved through a bilevel optimization process, enabling parameter backpropagation. This process employed loss functions to compare the output of iterative algorithms for the variational model with labeled data [14]. Subsequent research aimed to enhance the versatility of these models by treating the entire regularization term, including its gradient and proximal mapping, as learnable network modules. Notable examples of this approach include architectural designs like Unet [43] and multi-layer convolutional neural network (CNN) [39]. This innovative approach effectively unfolded the regularization model into a DNN, enabling end-to-end training [1, 8, 18, 27, 28, 50]. However, it is crucial to note that end-to-end training does not guarantee that the output of an unfolded DNN corresponds to a solution derived from a specific variational regularization model. Therefore, from a theoretical perspective, current unfolded DNNs cannot be classified as regularized methods, especially within the framework of variational regularization. Revisiting variational regularization, it is based on a variational model, and its solution becomes stable for the inverse problem when the regularization parameter is chosen appropriately. It is evident that the reason unfolded DNNs exhibit this theoretical gap compared to variational regularization models is that the latter relies solely on the variational model without direct consideration of corresponding algorithms, whereas unfolded DNNs are algorithmically driven.
Fortunately, in addition to variational regularization, another regularization method based on iterative algorithms exists, known as iterative regularization. Unlike variational regularization, iterative regularization does not rely on the variational model and regularization parameter, but instead establishes appropriate termination criteria for iterative algorithms. Under certain conditions, the terminated iteration becomes a stable solution to the inverse problem \cite{5, 16, 24, 25}. Given the structural similarity between iterative regularization and the cascading structure of unfolded DNNs, it is theoretically feasible to model unfolded DNNs within the framework of iterative regularization. Therefore, this paper aims to bridge the theoretical gap associated with unfolding methods in solving inverse problems. Specifically, our method involves the joint learning of a convex penalty function using an input-convex neural network (ICNN) to quantify the distance to a real data manifold. We then train a DNN unfolded from the proximal gradient descent (PGD) algorithm, incorporating this learned penalty. Additionally, we introduce a new termination criterion for the unfolded DNN. Through these improvements, we provide a theoretical proof of its equivalence to an iterative regularization method. Our primary contributions can be summarized as follows:

- Firstly, we introduce a novel procedure for the simultaneous training of a convex penalty function and a DNN unfolded from the PGD algorithm with this learned penalty function. When the real data manifold intersects the solutions of the inverse problem with a unique real solution, we provide a proof that the unfolded DNN converges to the real solution under perturbation-free measurements.
- Secondly, when the measurements contain perturbations, we present a new termination criterion for the unfolded DNN and establish its finite-step stopping property. Furthermore, the output of the unfolded DNN is guaranteed to converge to the real solution of the inverse problem as perturbations decrease. This implies that the unfolded DNN’s output remains continuous and stable with respect to the measurements.
- Thirdly, in our proposed training procedure, the loss function assesses the distance from each layer of the unfolded DNN to the real data manifold, rather than solely relying on the $L_2$-norm between the last layer of the unfolded DNN and the labeled data. This modification results in faster convergence, as supported by our numerical experiments.
- Finally, we validate the effectiveness of our proposed method through an example of MRI reconstruction. Our experimental results clearly demonstrate that our method outperforms conventional unfolding and traditional regularization methods in terms of reconstruction quality and stability.

The remainder of the paper is organized as follows. Section 2 provides a review of the background. Section 3 introduces modifications to the training procedure and demonstrates that the unfolding method is an iterative regularization technique. Implementation details are outlined in section 4. In section 5, we present the results of experiments conducted on several datasets. A discussion is presented in section 6. The final section, section 7, offers some concluding remarks. All proofs are presented in the appendix.

2. Background

2.1. Inverse problems in imaging

The inverse problem involves the reconstruction of an unknown image denoted as $x$ from measurement:
\[ Ax = y \quad (2.1) \]

where \( A : X \rightarrow Y \) is a bounded linear operator operating between two Hilbert spaces, \( X \) and \( Y \). The measurement is denoted as \( y \in Y \). The inner products and norms in spaces \( X \) and \( Y \) are denoted as \( \langle \cdot, \cdot \rangle \) and \( \| \cdot \| \), respectively, which should be clear from the context. For example, in MRI, \( A \) corresponds to a subsampled Fourier transform that comprises the Fourier transform and a binary sampling operator [31]. In CT, \( A \) represents a subsampled Radon transform, which is a partial collection of line integrations [6].

### 2.2. Handcraft regularization methods

Variational regularization is a successful and well-established method for solving inverse problems. Given a measurement that may contain perturbations, expressed as \( y^\delta = y + n \), where \( n \) represents the perturbation with \( \|n\| = \delta \), a stable solution to the inverse problem (2.1) is achieved by solving the following variational problem:

\[
x^* (y^\delta, \lambda) = \arg\min_{x \in X} \frac{1}{2} \| Ax - y^\delta \|^2 + \lambda r(x) \quad (2.2)
\]

where \( \| Ax - y^\delta \|^2 \) is the data consistency, \( r(x) \) represents the regularizer, and \( \lambda \) is the regularization parameter, which critically influences the method’s performance. From a statistical perspective, the regularizer \( r \) serves as a tool for characterizing the priors of the solution. Early research primarily focused on designing handcrafted regularizers, such as total variation (TV) [42], wavelets [34], and wavelet frames [11, 40], among others. Theoretically, when \( \lambda \) is dynamically adjusted based on \( y^\delta \) to satisfy

\[
\lim_{\delta \to 0} \lambda(\delta) = 0 \quad \text{and} \quad \lim_{\delta \to 0} \frac{\delta^2}{\lambda(\delta)} = 0,
\]

the solution \( x^* (y^\delta, \lambda) \rightarrow x^\dagger \) as \( \delta \rightarrow 0 \) under certain conditions [21], where \( x^\dagger \) is the \( \tau \)-minimizing solution (see definition 3.4 for detail) to inverse problem (2.1). This regularization property indicates that the solution to the variational model (2.2) is continuous and stable with respect to perturbations in the measurements, meaning that perturbations in the measurements do not significantly degrade the solution.

It’s essential to note that the variational regularization theory relies solely on the optimal solution \( x^* (y^\delta, \lambda) \) of (2.2), without direct consideration of practical methods for solving it. In practice, the variational problem is usually addressed using iterative algorithms. Thus, iterative regularization has emerged as an attractive alternative regularization method. It achieves regularization without depending on the regularizer \( r \) or the regularization parameter \( \lambda \). Instead, it utilizes iterative algorithms with termination criteria, resulting in stable approximations of the real solution. For example, the classical Landweber iteration is defined as:

\[
x_{k+1}^\delta = x_k^\delta - \eta A^* (Ax_k^\delta - y^\delta).
\]

The iteration continues until the criterion \( \| Ax_k^\delta - y^\delta \| \leq \tau \delta \) (\( \tau > 0 \)) is satisfied, at which point the iteration terminates after a finite number of steps denoted as \( K(\delta) \), yielding the output \( x_{K(\delta)}^\delta \).
Theoretically, due to this finite step termination, we have $x_{K(\delta)}^\delta \to x^\dagger$ as $\delta \to 0$ under certain conditions [26], where $x^\dagger$ is the $\| \cdot \|$-minimizing solution to inverse problem (2.1).

However, in medical imaging, relying solely on $\| \cdot \|$ is inadequate for characterizing image priors. To address this limitation, it is necessary to introduce a penalty function $f$ and utilize the PGD method:

$$
\xi_{k+1} = x_k - \eta A^* (Ax_k - y) \\
x_{k+1} = \text{arg min}_{x \in X} \frac{1}{2} \| x - \xi_{k+1} \|^2 + f(x).
$$

(2.3)

In general, the function $f$ can also be selected from a set of handcrafted functions, such as TV, wavelets, and wavelet frames. Please note that while the penalty function $f$ serves a role similar to the regularizer used in variational regularization for characterizing priors about the solution $x$, the regularization property of iterative regularization relies on the termination criterion rather than depending on some variational model defined by $f$. Therefore, the roles of $f$ in iterative regularization and the regularizer in variational regularization are not entirely consistent.

### 2.3. Unfolding methods

In traditional variational regularization, regularizers are manually crafted, posing challenges in accurately and adaptively representing data priors. Unfolding methods replace the regularizer, including its gradient and proximal mapping, with learnable network modules (such as Unet, multi-layer CNN, etc). This process unfolds the corresponding algorithm into a DNN.

By optimizing the loss function between the output of the unfolded DNN and labeled data, parameter backpropagation for the network modules is achieved, enabling end-to-end training. Taking the PGD (ISTA)-Net [51] as an example, the proximal mapping of the regularizer $r$ is replaced by a multi-layer CNN denoted as $S_{\theta_k}$ with parameter $\theta_k$ at iteration $k$. This transformation unfolds the PGD algorithm into the following DNN:

$$
\xi_{k+1}^n = \xi_k^n - \eta A^* (Ax_k^n - y^n) \\
x_{k+1}^n = S_{\theta_{k+1}} (\xi_{k+1}^n).
$$

(2.4)

Once the number of unfolding layers, denoted as $K$, is determined, the network’s output $x_K^n$ can be considered a vector function of the parameter $\Theta := [\theta_1, \ldots, \theta_K]$. With a given set of training data $\{(x^n, y^n)\}_{n=1}^N$, the network can be end-to-end trained by optimizing the loss function between $x_K^n(\Theta)$ and the ground truth $x^n$:

$$
\min_{\Theta} \frac{1}{N} \sum_{n=1}^N L(x_K^n(\Theta), x^n)
$$

(2.5)

where $L(\cdot, \cdot)$ represents the loss function, which is typically chosen as $L(\cdot, \cdot) = \| \cdot - \cdot \|^2$, $L(\cdot, \cdot) = \| \cdot - \cdot \|_1$, and so on [30].

Conversely, for the learned network module $S_{\theta_k}$, ensuring the existence of a regularization function $r$ for which the following equation holds can be challenging:

$$
\text{Prox}_r(\xi_k^\delta) = S_{\theta_k}(\xi_k^\delta)
$$

(2.6)

for $k = 1, \ldots, K$, where $\text{Prox}_r$ denotes the proximal mapping of $r$. This complexity arises because finding an functional $r$ that aligns with the behavior (2.6) is a non-trivial task.
implies that the output of the unfolded DNN, i.e. \( x_n^k(\Theta) \), cannot be fully covered by the solution of a variational regularization model. Consequently, unfolded DNN (2.4) cannot entirely inherit the theoretical properties of variational regularization. While it is true that a zeroth-order unfolding method has been proposed in [10], which allows \( S_{\theta} \) in equation (2.4) to capture the zeroth-order information about the \( k \)-space self-consistency regularizer in MRI, ensuring that the output of (2.4) can be covered by a \( k \)-space interpolated variational regularization model. However, this type of unfolding method is specific to certain \( k \)-space interpolation models and lacks generality. Upon reevaluating unfolded DNN (2.4), we have observed that its algorithmically-driven cascading structure exhibits a closer resemblance to iterative regularization (2.3). This realization has prompted us to modify the training procedure of the unfolded DNN (2.4) and configure appropriate termination criteria. Ultimately, this establishes the unfolding method as an iterative regularization technique.

3. Unfolded DNN for inverse problems

In the previous section, we discussed that end-to-end training of the unfolded DNN (2.4) using loss (2.5) cannot effectively regulate the network module \( S_{\theta} \) to meet (2.6). This limitation prevents (2.4) from inheriting the theoretical properties of variational regularization models. To address this issue, in this section, we will reevaluate (2.4) from the perspective of iterative regularization. Our approach is as follows: If we can learn a penalty function with interpretability and then enforce that the module \( S_{\theta} \) in (2.4) learns an approximation of the proximal mapping of this learned convex penalty function, then (2.4) can be covered by the iterative regularization framework (2.3).

Recognizing this, we will redesign the training procedure of the unfolded DNN (2.4) and demonstrate that it conforms to the iterative regularization framework. Specifically, we will jointly train a convex penalty function \( f(\cdot) \) and the unfolded DNN (2.4). The training aims to ensure that the convex penalty function measures the distance between each iteration \( \xi_n^k \) and a real data manifold, and that the module \( S_{\theta} \) in the unfolded DNN approximates the proximal mapping of the learned convex penalty function.

3.1. Training procedure

Now, let us delve into the detailed procedure of joint training the penalty function and the unfolded DNN. We assume that the training data, \( \{x^a\}_{a=1}^N \in X \) are independent samples from the distribution of real data (truth images) \( P_x \), and \( \{y^a\}_{a=1}^N \in Y \) are independent samples from the distribution of measurements \( P_y \). Based on the measurements, \( \{y^a\}_{a=1}^N \), the initial values, \( \{x_0^a\}_{a=1}^N \), of the unfolded DNN can be obtained using a pseudo-inverse mapping \( A^\dagger \) of \( A \), i.e. \( \{x_0^a\}_{a=1}^N = A^\dagger y^a \sim P_0 = A^\dagger P_y \). With these training data, the joint training procedure can be implemented by alternately optimizing the following two subproblems:

**Subproblem 1:** At the \( n \)th iteration, with the penalty function \( f_{\phi} \) fixed, for each iteration \( x_n^a \), we make the network module \( S_{\theta} \) approximate the proximity mapping of \( f_{\phi} \). Consequently, we define the loss function:

\[
J_{1,n}([\theta_1, \ldots, \theta_K]) := \sum_{n=1}^N \sum_{k=1}^K \frac{1}{2} \|S_{\theta_k}(\xi_n^a) - \xi_n^a\|^2 + f_{\phi}(S_{\theta_k}(\xi_n^a))
\]

(3.7)
where $\xi_k^n = x_k^n - \eta A^r (Ax_k^n - y^n)$ and $x_k^{n+1} = S_{\theta_k}(\xi_k^n)$ for $k = 0, \ldots, K - 1$. This loss is minimized using the Adam algorithm [29], terminated by $T_\theta$ iterations, step size $\gamma$, and decay rates $\beta_1$ and $\beta_2$. If $f_{\phi'}$ is a convex function, the optimal network modules $S_{\theta_k}$ minimize (3.7) to a unique minimum, which implies that the optimal $S_{\theta_k}$ serves as the proximal mapping of $f_{\phi'}$ at $\xi_k^n$.

**Subproblem 2:** With the already updated $\{S_{\theta_k^n}\}_{k=1}^K$ fixed, we proceed to train the penalty function. Looking back at the loss function (3.7), if we want $S_{\theta_k}(\xi_k^n)$ to approach the true solution, $f$ should have the capability to discriminate between $S_{\theta_k}(\xi_k^n)$ and the true solution. Therefore, we define the loss function as follows:

$$J_{2,t+1}(\phi) := \mathbb{E}_{x_t \sim P_t} [f_{\phi}(x^t)] - \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{x_t \sim P_t} \left[ f_{\phi} \left( S_{\theta_k^{t+1}}(\xi_k^n) \right) \right].$$

Minimizing the loss function $J_{2,t}(\phi)$ is an adversarial learning process that endows the penalty function with the role of a discriminator. We will discuss its specific mathematical implications in subsequent sections. To ensure the theoretical properties of this model, we use ICNN to represent $f_{\phi}$ to ensure its convexity, and the architecture of ICNN will be presented in the following section.

It’s worth mentioning that, although our proposed model and the original PGD-Net network structure both rely on (2.4), the trained network module $S_{\theta_k}$ serves different purposes. The original PGD-Net trains the network module $S_{\theta_k}$ by optimizing the loss function (2.5), which only enforces the final layer’s output $x_k^n(\theta)$ to match the true solution. $S_{\theta_k}$ obtained through this training mode currently lacks complete interpretability. In contrast, by introducing a learnable convex penalty function, our designed loss function (3.7) compels $S_{\theta_k}$ to approximate the proximal mapping of the penalty function. In the following sections, we will elaborate on the specific mathematical significance of the learned convex penalty function. Therefore, in our proposed method, $S_{\theta_k}$ is interpretable and can be considered a “white-box” solution. Additionally, it is worth noting that our proposed loss function (3.7) drives every layer $\{S_{\theta_k}\}_{k=1}^K$ to approach the true solution, rather than solely penalizing the final layer’s output of the unfolded DNN. Consequently, this training strategy has the potential to significantly accelerate the numerical convergence of the unfolded DNN.

The specific training steps are outlined in algorithm 1, which requires a series of input parameters. $\eta, \gamma, \beta_1,$ and $\beta_2$ are parameters within the optimizer, and their specific values will be provided in section 4. $T$ represents the total number of iterations for alternating optimization in the algorithm. When training the DNN, it is common to shuffle the training set’s order and iterate through the training set multiple times, such that $T = MN$, where $M$ is the number of times the training set is traversed, and $N$ is the size of the training set. $T_{\theta}$ and $T_{\phi}$ denote the total number of iterations in the inner loop for optimizing the loss functions $J_{1,t}$ and $J_{2,t}$, respectively. The choice of these values can draw from empirical guidance, as mentioned in [37]. $K$ represents the number of layers in the unfolded network (2.4). The choice of $K$ involves a trade-off between model complexity and performance, which will be discussed in more detail in section 6. For a more vivid presentation of this joint training process, we illustrate it in figure 1.
Figure 1. Illustration of the joint training procedure of the unfolded DNN and penalty function.

Algorithm 1. Training procedure.

Input: input \{x^n\}_{n=1}^N \sim P_x, \{y^n\}_{n=1}^N \sim P_y, Adam optimizer, positive constants \( \eta, \gamma, \beta_1, \beta_2, K, T, T_0 \) and \( T_\phi \).

Initialize: \{x^n_0\}_{n=1}^N, unfolded DNN \( G_\Theta : \mathbb{R}^n \to \mathbb{R}^n \) with initial parameter \( \Theta_0 \), convex and 1-Lipschitz continuous non-negative network \( f_\phi \) with initial parameter \( \phi_0 \).

1: for \( t = 0, \ldots, T-1 \) do
2: Define loss function \( J_{1,t} \) for \( G_\Theta \).
3: Set \( \Theta_{t+1} \leftarrow \Theta_t \).
4: for \( s = 0, \ldots, T_{\Theta} - 1 \) do
5: \( \Theta_{s+1} = \text{Adam} (\Theta_s, J_{1,t}(\Theta_s), \gamma, \beta_1, \beta_2) \).
6: end for
7: \( \Theta_{t+1} \leftarrow \Theta_{s+1} \).
8: Define loss function \( J_{2,t+1} \) for \( f_\phi \).
9: for \( s = 0, \ldots, T_{\phi} - 1 \) do
10: \( \phi_{s+1} = \text{Adam} (\phi_s, J_{2,t+1}(\phi_s), \gamma, \beta_1, \beta_2) \).
11: end for
12: \( \phi_{t+1} \leftarrow \phi_{s+1} \).
13: end for

Output: \( G_{\Theta} \) (or \( \{S_\theta\}_{k=1}^K \)) and \( f_\phi \).

Now, let us normalize the assumptions and state the result regarding \( f_\phi \). For algorithm 1, we can define a sequence of algorithmic operators \( T_k : \mathbb{R}^n \to \mathbb{R}^n \) to represent the iteration of unfolded DNN (2.4). In particular, we define

\[
T_{k+1}(u) = S_{\theta_{k+1}}(T_k(u) - \eta A^*(AT_k(u) - y)).
\]

Then the \( k \)th iteration can be reformulated as

\[
x_k = T_k(x_0).
\]

Initialized with distribution \( P_0 \), we also define the distribution \( P_k \) as the push forward of the algorithmic operator \( T_k \) applied on \( P_0 \), i.e.

\[
P_k = (T_k)_# P_0.
\]

The next assumption will relate \( P_k \) to the real data distribution \( P_x \). Before giving this assumption, we make some definitions and assumptions about the real data manifold \( \mathcal{M} \). Drawing on
litteratures [17, 32], we first formalize the idea that real data are contained in a low dimensional manifold $\mathcal{M}$.

**Assumption 3.1.** Assume the distribution $P_x$ is supported on a convex compact set $\mathcal{M}$ of $X$, i.e. $P_x(\mathcal{M}^c) = 0$.

Based on the above assumption, the projection onto data manifold $\mathcal{M}$ can be defined as

$$P_{\mathcal{M}} : X \to \mathcal{M}, \quad P_{\mathcal{M}}(x) := \arg\min_{z \in \mathcal{M}} \|z - x\|.$$  

This projection can be used to express the pointwise distance function

$$d_{\mathcal{M}}(x) := \inf_{z \in \mathcal{M}} \|z - x\| = \|P_{\mathcal{M}}(x) - x\|.$$  

The strong assumption on each real data can be reconstructed from its measurement data is discarded. This paper follows [17] and makes the following weaker assumption:

**Assumption 3.2.** For any $k \leq K$, the push forward of the projection operator $P_{\mathcal{M}}$ on distribution $P_k$ recovers the real data distribution $P_x$ up to a set of measure zero, i.e. $P_k$ pushed forward by $P_{\mathcal{M}}$ satisfies

$$(P_{\mathcal{M}})_\# P_k = P_x.$$  

Drawing on the proof technique from [32], we show that if the loss function (3.8) is obtained minimally, the trained convex penalty function can measure the distance from the iterates of unfolded DNN to the real data manifold.

**Theorem 3.3.** Suppose assumptions 3.1 and 3.2 hold. For any $k \leq K$, the minimizer to the functional

$$E_{X \sim P_k}[f(X)] - \frac{1}{K} \sum_{k=1}^{K} E_{Z \sim P_k}[f(Z)]$$  

(3.9)

is given by the distance of $Z \sim P_k$ to the real data manifold $\mathcal{M}$, where $f$ is constrained to a convex and 1-Lipschitz continuous non-negative function.

To minimize the gap between algorithm 1 and theorem 3.3, we employ ICNN to ensure the convexity of the penalty function $f_\phi$ and introduce spectral normalization [35] or gradient penalty [15] to account for continuity.

Following theorem 3.3, it becomes clear how the proposed method (algorithm 1) works. In essence, the learned penalty function quantifies the distance to a real data manifold, and network modules $S_{\theta_k}$ approximate the proximity mapping of this learned penalty. Hence, the proposed method is an interpretable DL approach.

### 3.2. Testing procedure

Based on the learned proximal mapping $\{S_{\theta_k}^\perp\}_{k=1}^{K}$ (the output of algorithm 1), we can apply the proposed unfolded PGD to solve the inverse problem (2.1). However, in practical applications, measurements often include perturbations. Past experience indicates that iterative algorithms typically exhibit semi-convergence, meaning they start the iteration process by approaching the true solution but gradually deviate from it as the iteration progresses [22, 23]. Therefore, it is crucial to terminate them at the right moment to ensure the stability of the proposed unfolded PGD against measurement perturbations.
In algorithm 3.1, we say that if \( f_{\phi^*} \) minimizes \( \{ T_n \}_n \), then we equip the unfolded PGD algorithm with a termination criterion (3.10), which is adapted from Morozov’s criterion [36]. It’s worth noting that in practice, the use of \( f_{\phi^*} \) in the termination criterion may pose certain challenges. Theorem 3.3 demonstrates that if \( \phi^* \) minimizes (3.8), then we have \( f_{\phi^*} = d_{Ax}(x^*) = 0 \), where \( x^* \) is the true image. If \( \phi^* \) fails to minimize (3.8), then \( f_{\phi^*}(x^*) \) will also tend to have small values during network training. Therefore, in practice, we can approximate \( f_{\phi^*} \approx 0 \).

Another key difference between the proposed model (algorithm 2) and the original unfolded PGD lies in the termination criterion (3.10). Subsequent sections will provide theoretical evidence (theorem 3.14) that equipping the termination criterion (3.10) will make the proposed method stable with respect to measurement perturbations. It’s essential to note that the stability primarily relies on the training strategy in algorithm 1, as indicated in the theoretical analysis later in this section. Therefore, even if the original unfolded PGD is equipped with the termination criterion (3.10), its stability cannot be guaranteed. We will validate this claim experimentally in section 5.

3.3. Convergence and regularity

Before analyzing the convergence and stability, we normalize the assumption on the learned network modules \( S_{\theta^*} \).

Definition 3.4. We say \( x \) solves minimization problem \( \min_{z \in X} g(z) \) in exactly with error \( \epsilon \) if \( x \) satisfies

\[
g(x) \leq \min_{z \in X} g(z) + \epsilon
\]

Algorithm 1 has difficulty ensuring that the loss function (3.7) converges to zero absolutely, resulting in difficulties for \( S_{\theta^*}(\xi_k) \) to accurately approximate the proximal mapping of \( f_{\phi^*} \). Thus, we make the following more relaxed assumption.

Assumption 3.5. In algorithm 2, the learned network modules \( S_{\theta^*}(\xi_k) \) solves \( \min_{\frac{1}{2}||z - \xi_k||^2 + f_{\phi^*}(z)} \) in exactly with error \( \epsilon_k \) and \( \sum_{k=0}^{+\infty} \epsilon_k < +\infty \), where \( \xi_k = x_k^\delta - \eta A^* (Ax_k^\delta - y^\delta) \) and \( x_{k+1}^\delta = S_{\theta^*}(\xi_k) \).
**Remark 3.6.** In practice, if assumption 3.5 does not hold, since $f_{\phi^T}$ is convex, we can obtain a solution satisfying assumption 3.5 by performing additional few (sub)gradient descent steps with $S_{\phi^T}(\xi_k)$ as the initial value.

Consider first that the measurement is mixed with perturbations. In this case, we will show that algorithm 2 will be terminated after a finite number of iteration steps.

**Proposition 3.7.** Suppose assumption 3.5 hold. Let $0 < \eta < \frac{1}{4}$ and $\tau > \max\{\frac{3}{2\eta}, \frac{\eta}{2}\}$. When the measurement is mixed with noisy perturbation, i.e. $\delta > 0$, algorithm 2 will terminate after finite $k_*$ steps of iteration.

When the measurements are perturbation-free, i.e. $\delta = 0$, we will normalize some assumptions and state the result on the convergence of algorithm 2.

**Definition 3.8.** An solution $x^* \in X$ is called a $f$-minimizing solution to inverse problem (2.1) if it satisfies

$$f(x^*) \leq f(x), \forall x \in \{u \in X|Au = y\}.$$ 

**Assumption 3.9.** The real data manifold $M$ intersects the solutions of inverse problem (2.1) with the unique real solution, i.e. $M \cap \{x \in X|Ax = y\} = \{x^*\}$

**Remark 3.10.** If $f$ measures the distance to the real data manifold $M$, i.e. $f(\cdot) = d_M(\cdot)$, and assumption 3.9 holds, then the $f$-minimizing solution is the unique real solution to the inverse problem (2.1).

Assumption 3.9 is quite stringent. In our analysis, we will delve into the convergence of algorithm 2 under scenarios where it holds as well as when it does not.

**Theorem 3.11.**

(a) Consider $\phi^T$ obtaining the minimum value of (3.8). Suppose assumption 3.1, 3.2, 3.5 and 3.9 hold, and the measurement is perturbation-free. The sequence generated by algorithm 2 converges to the real solution $x^*$ of inverse problem (2.1).

(b) Consider that $\phi^T$ cannot obtain the minimum value of (3.8). Suppose assumption 3.1, 3.2 and 3.5 hold, the measurement is perturbation-free, and $f_{\phi^T}(\cdot)$ is strongly convex. The sequence generated by algorithm 2 converges to a $f_{\phi^T}$-minimizing solution of inverse problem (2.1).

**Remark 3.12.** The above theorem proves the convergence of the proposed algorithm as the iterations converge to infinity. However, in practice, the number of iterations (number of layers of network unfolding) $K$ is limited. We will show the fast convergence of the algorithm 2 later by numerical experiments, thus demonstrating that the algorithm output approximates the limit point well, even when $K$ is limited.

**Remark 3.13.** In theorem 3.11 (b), the strong convexity can be achieved by adding a term $\nu\|\cdot\|^2$ to $f_{\phi^T}(\cdot)$, where $\nu$ is a positive constant.

In the presence of perturbation in the measurements, denoted as $\delta > 0$, proposition 3.7 elucidates that algorithm 2 will undergo $k_*$ iterations and subsequently halt. In scenarios with perturbation-free measurements, as established by theorem 3.11, algorithm 2 asymptotically converges to a real solution or an $f_{\phi^T}$-minimizing solution for inverse problem (2.1). Our subsequent discussion will focus on demonstrating that the solution obtained after $k_*$ iterations provides a stable approximation of the real solution or the $f_{\phi^T}$-minimizing solution.
Theorem 3.14. Let the assumptions of theorem 3.11 hold and let \( k_\ast = k_\ast(\delta, y^\delta) \) be chosen according to the termination criterion (3.10). Then, the output \( x^\delta_\ast \) of algorithm 2 converges to the real solution or \( f_{\phi_T} \)-minimizing solution of inverse problem (2.1) as \( \delta \to 0 \).

The above theorem demonstrates that the early terminated solution of algorithm 2, denoted as \( x^\delta_\ast \), remains continuous with respect to the measurements. This implies that perturbations in the measurements do not lead to drastic variations in \( x^\delta_\ast \). Therefore, algorithm 2 can stably solve the inverse problem (2.1).

4. Implementation

The evaluation was performed on two multichannel MRI data with various trajectories. The details of the MRI data are as follows:

4.1. Data acquisition

4.1.1. Knee data. The knee raw data\(^7\) is acquired from a 3T Siemens scanner (Siemens Magnetom Skyra, Prisma and Biograph mMR)\(^8\). Data acquisition uses a 15 channel knee coil array and conventional Cartesian 2D TSE protocol employed clinically at NYU School of Medicine. The following sequence parameters are used: Echo train length 4, matrix size \( 320 \times 320 \), in-plane resolution \( 0.5 \, \text{mm} \times 0.5 \, \text{mm} \), slice thickness 3 mm, no gap between slices. Timing varied between systems, with repetition time (TR) ranging between 2200 and 3000 milliseconds, and echo time (TE) between 27 and 34 milliseconds. From them, we randomly select T1-weighted data of 34 individuals (1002 slices in total) as the training set and data of 3 individuals (95 slices in total) as the test set.

4.1.2. Brain data. To verify the generalization of the proposed method, we test it on human brain MRI data\(^8\), which was collected by [2]. These MRI data were acquired using a 3D T2 fast spin echo with an extended echo train acquisition (CUBE) sequence with Cartesian readouts using a 12-channel head coil. The matrix dimensions are \( 256 \times 232 \times 208 \) with 1 mm isotropic resolution. The training data contains 360 slices \( k \)-space data from four subjects and the testing data contains 164 slices \( k \)-space data from two subjects. Each slice has a spatial dimension of \( 256 \times 232 \).

4.1.3. Sampling patterns. These datasets consist of fully sampled \( k \)-space raw data. Performing a Fourier inverse transformation on them yields multi-channel images. The measurements \( y \) are obtained through undersampling the \( k \)-space directly. Three types of undersampling patterns are considered, each with different acceleration factors and undersampling techniques. A visualization of these patterns is depicted in figure 2.

4.2. Network architecture and training

This paper considers the reconstruction of multichannel MR images to avoid additional calculations to estimate the coil sensitivities. The forward process of measurement acquisition can be modeled as the inverse problem:

---

\(^7\) https://fastmri.org/.

\(^8\) https://drive.google.com/file/d/1qp-l9kJbRfQU1W5wCjOQZi7IT6jwA37/view?usp=sharing.
Figure 2. Various sampling patterns: (a) 1D uniform undersampling at $R = 4$, (b) 1D random undersampling at $R = 8$, and (c) 2D random undersampling at $R = 12$, where $R$ represents the acceleration factor, implying that the measurement $y$ is obtained by using only $1/R$ of the full $k$-space data. In the sampling patterns, white regions indicate the locations of acquired data, while black regions represent the locations of unacquired data.

$PFx = y$

where $x = [x_1, \ldots, x_C]$ denotes the $C$-channel MR image, $F$ denotes Fourier transform, and $P$ denotes the undersampling operator with above patterns. The unfolded PGD for above inverse problem reads:

$$\xi^n_{k+1} = x^n_k - \eta F^{-1} P^* (PFx^n_k - y^n)$$

$$x^n_{k+1} = S_{\theta_{k+1}}(\xi^n_{k+1}).$$

Without additional explanation, the above network is unfolded with ten layers (i.e. $K = 10$). The impact of different numbers of unfolding layers, $K$, on the reconstruction results will be discussed in the following sections.

Recalling the traditional wavelet domain sparse penalty function, $\|Wx\|_1$, where $W$ is the wavelet transform, its corresponding proximal mapping is $W^* \text{Prox}_{\| \cdot \|_1}(Wx)$. This encoding-activation-decoding structure motivates us to choose a similar network representation for $S_{\theta_i}$. Therefore, the $S_{\theta_i}$ module is represented by the U-Net [41]. Its encoder comprises $3 \times 3$ convolution, batch normalization [19], and ReLU, while the decoder uses the nearest neighbor upsampling and $3 \times 3$ convolution. Max-pooling layers and skip connections through channel concatenation are also included. Real and imaginary values are concatenated along the channel dimension to handle complex values. Consequently, each input and output has $2C$ channels.

On the other hand, the purpose of the penalty function is to extract features from the image. Previous research [20] has demonstrated the superiority of the pyramid-shaped discriminator, known as the PatchGAN, in terms of feature extraction from images. However, in this paper, we require the penalty function to satisfy convexity, which may limit the feature extraction capabilities of a pyramid-shaped discriminator. Consequently, we introduce a residual structure to compensate for the features extracted at each layer. We will validate this design’s ability to extract features as effectively as its non-convex version in our experiments. The network includes both general and non-negative convolution layers, along with batch normalization, ReLU, and average pooling.
In training algorithm 1, we follow the procedure detailed in [37], where we set $T_\Theta = 1$ and $T_\phi = 5$. For the loss function $J_{1,t}$, we introduce a strongly convex term $\mu_1\|\mathbb{S}_{\hat{h}}(\zeta_k^n) - x^n\|^2$ to enhance training stability. As for the loss function $J_{2,t+1}$, we incorporate a gradient penalty term to enforce 1-Lipschitz continuity of $f_\phi$. Specifically, we have:

$$J_{1,t}(\Theta) := \sum_{n=1}^N \sum_{k=1}^K \left[ \frac{1}{2} \| \mathbb{S}_{\hat{h}}(\zeta_k^n) - \zeta_k^n \|^2 + f_\phi(\mathbb{S}_{\hat{h}}(\zeta_k^n)) + \mu_1\|\mathbb{S}_{\hat{h}}(\zeta_k^n) - x^n\|^2 \right]$$

$$J_{2,t+1}(\phi) := \mathbb{E}_{x' \sim \mathbb{P}_\gamma}[f_0(x')] - \frac{1}{K} \sum_{k=1}^K \mathbb{E}_{z \sim \mathbb{P}_\gamma}[f_0(\mathbb{S}_{h}(\zeta_k^n))] + \mu_2\mathbb{E}_{z \sim \mathbb{P}_\gamma}[\left(\|\nabla f_0(\check{x})\| - 1\right)^2]$$

where $\mathbb{P}_\gamma$ samples points uniformly along straight lines between pairs of points sampled from the data distribution $\mathbb{P}_x$ and the distribution $\mathbb{P}_\gamma$.

Algorithm 1 uses the ADAM optimizer [29] in relation to the loss functions $J_{1,t}$ and $J_{2,t+1}$. The mini-batch size is set to 1, and the training spans 100 epochs. The learning rate $\gamma$ is configured at $10^{-4}$, and $\beta_1$ and $\beta_2$ take on the values of 0.9 and 0.999, respectively. For the labels, we utilize the fully sampled MR image, while the input data for the network consists of the regridded downsampled $k$-space measurement from 1D and 2D random trajectories. In the absence of specific instructions, we train the network separately for different trajectories. The models were implemented on an Ubuntu 20.04 operating system, powered by an NVIDIA A800 Tensor Core GPU with 80 GB of memory, within the open-source PyTorch 1.10 framework [38] with CUDA 11.3 and CUDNN support.

4.3. Performance evaluation

In this study, the quantitative evaluations are all calculated on the image domain. The image is derived using an inverse Fourier transform followed by an elementwise square-root of sum-of-the squares (SSoS) operation, i.e. $z[n] = (\sum_{i=1}^C |x_i[n]|^2)^{1/2}$, where $z[n]$ denotes the nth element of image $z$, and $x_i[n]$ denotes the nth element of the i-th coil image $x_i$. For quantitative evaluation, the peak signal-to-noise ratio (PSNR), normalized mean square error (NMSE) value and structural similarity (SSIM) index [47] are adopted.

5. Experimentation results

In this section, we evaluate our newly proposed PGD unfolded network with a convex adversarial penalty function, denoted as CAPGD-Net, on knee and brain datasets. We conduct a series of extensive comparative experiments to demonstrate the effectiveness of our methods. Specifically, we compare it against the traditional iterative method SPIRiT [33], which can be viewed as an iterative sequence (2.3) equipped with a structural low-rank penalty function to characterize $k$-space self-consistency. Although the original SPIRiT does not use the discrepancy principle to turn it into an iterative regularization method when dealing with perturbed measurements. We also compare it to the state-of-the-art (SOTA) unfolded method for multi-channel $k$-space reconstruction, HDSLR [39]. Furthermore, given the similarity of our proposed method’s training strategy (algorithm 1) to GANs, where the unfolded network in algorithm 1 resembles a generator and the penalty function resembles a discriminator, we also compare it to the SOTA GAN model.
in MRI reconstruction, optimal transport (OT)-GAN [37]. To validate the advantages of the proposed training strategy (algorithm 1), we design a comparative method, PGD-Net, which has the same network structure ($S_{\theta_k}$) as CAPGD-Net but uses an $L_2$-norm loss function to measure the distance between its final output and the label. Finally, for theoretical completeness, CAPGD-Net requires the penalty function $f_{\phi}$ to ensure convexity. As shown in figure 3, ensuring convexity necessitates some network weights to be positive, potentially affecting $f_{\phi}$’s ability to characterize image features. To assess the performance implications of convexity, we conduct a comparative experiment in which we remove the constraint that some weights in the $f_{\phi}$ network of CAPGD-Net must be positive, resulting in a PGD unfolded network with a nonconvex adversarial penalty function, denoted as NCAPGD-Net.
5.1. Experiments without perturbations

In this section, we evaluate the performance of the proposed method and the comparative methods under the scenario where no additional perturbations in the measurement. Figure 4 presents the reconstruction results of the knee data using various methods under a 1D uniform trajectory with an acceleration factor of 6, corresponding to a sampling rate of 1/6. It is evident that the traditional iterative regularization method, SPIRiT, is limited by the accuracy of handcrafted priors, resulting in reconstruction results that still exhibit significant aliasing patterns and noise. In the case of HDSL, noise is effectively suppressed, but some aliasing patterns persist. OT-GAN successfully mitigates both aliasing patterns and noise; however, it lacks data consistency correction, leading to notable discrepancies between the imaging results and the reference data.

Comparatively, the remaining three methods outperform the previous three. Upon closer examination, we observe that NCAPGD-Net and CAPGD-Net, as indicated by the red arrows in the top enlarged views, excel at reconstructing specific details, while PGD-Net fails to capture them. Additionally, in the bottom left corner of the enlarged view, as indicated by the red arrow, PGD-Net’s reconstruction image still exhibits a slight aliasing pattern. This highlights the superiority of the training strategy proposed in this paper (algorithm 1) over the use of an L2-norm loss between the final output and the label in PGD-Net. Finally, a detailed comparison of the reconstruction results between NCAPGD-Net and CAPGD-Net reveals that CAPGD-Net’s reconstruction quality is comparable to that of NCAPGD-Net. Therefore, this demonstrates that the convex constraint imposed on $f_\phi$ in CAPGD-Net does not adversely affect $f_\phi$’s ability to characterize image features. This effectively confirms the validity of theorem 3.3, which asserts that a convex function $f_\phi$ is sufficient to measure the distance of PGD iterations to the real data manifold.

The competitive quantitative results of the aforementioned methods are displayed in Table 1. These results align with the visual observations from Figure 4. Table 1 also presents the computation time for each method when testing on a slice. Notably, OT-GAN boasts the shortest execution time, while the remaining methods exhibit execution times that are quite similar,
Table 1. Quantitative comparison for various methods on the knee and brain data.

| Datasets & Methods  | NMSE        | PSNR (dB)    | SSIM        | Testing Time (s) (Per-Slice) |
|---------------------|-------------|--------------|-------------|------------------------------|
| Knee (uniform R = 6) |             |              |             |                              |
| SPIRiT              | 0.0134 ± 0.0065 | 30.17 ± 2.27 | 0.7641 ± 0.0648 | —                           |
| OT-GAN              | 0.0057 ± 0.0015 | 33.66 ± 1.24 | 0.9058 ± 0.0260 | 0.01                         |
| HDSLR               | 0.0055 ± 0.0020 | 33.93 ± 1.37 | 0.9096 ± 0.0292 | 0.05                         |
| PGD-Net             | 0.0051 ± 0.0015 | 34.17 ± 0.94 | 0.9218 ± 0.0177 | 0.05                         |
| NCAPGD-Net          | 0.0041 ± 0.0012 | 35.14 ± 0.91 | 0.9261 ± 0.0169 | 0.05                         |
| CAPGD-Net           | 0.0042 ± 0.0016 | 35.10 ± 0.96 | 0.9258 ± 0.0175 | 0.05                         |
| Knee (random R = 8) |             |              |             |                              |
| SPIRiT              | 0.0115 ± 0.0029 | 30.70 ± 1.92 | 0.7958 ± 0.0644 | —                           |
| OT-GAN              | 0.0120 ± 0.0048 | 30.56 ± 1.04 | 0.8522 ± 0.0383 | 0.01                         |
| HDSLR               | 0.0098 ± 0.0031 | 31.38 ± 1.42 | 0.8536 ± 0.0438 | 0.04                         |
| PGD-Net             | 0.0085 ± 0.0034 | 32.13 ± 1.18 | 0.8769 ± 0.0329 | 0.05                         |
| NCAPGD-Net          | 0.0070 ± 0.0019 | 32.80 ± 1.39 | 0.8780 ± 0.0347 | 0.05                         |
| CAPGD-Net           | 0.0073 ± 0.0020 | 32.60 ± 1.13 | 0.8802 ± 0.0324 | 0.05                         |
| Brain (random R = 12) |            |              |             |                              |
| SPIRiT              | 0.0221 ± 0.0114 | 33.86 ± 3.65 | 0.9417 ± 0.0247 | —                           |
| OT-GAN              | 0.0213 ± 0.0302 | 35.87 ± 2.62 | 0.9566 ± 0.0210 | 0.01                         |
| HDSLR               | 0.0084 ± 0.0100 | 38.50 ± 1.02 | 0.9772 ± 0.0068 | 0.03                         |
| PGD-Net             | 0.0079 ± 0.0088 | 38.94 ± 2.14 | 0.9772 ± 0.0102 | 0.03                         |
| NCAPGD-Net          | 0.0082 ± 0.0088 | 39.17 ± 3.01 | 0.9723 ± 0.0131 | 0.03                         |
| CAPGD-Net           |             |              |             |                              |

Note: The red values highlight the achieved optimal quantitative metrics.

Figure 5. Reconstruction results under 1D random undersampling at R = 8. The values in the corner are the NMSE/PSNR/SSIM values of each slice. The second and third rows illustrate the enlarged and error views, respectively. The grayscale of the reconstructed images and the color bar of the error images are at the right of the figure.

falling within an acceptable range when compared to OT-GAN. It’s important to emphasize that, even though the proposed method involves the computation of the penalty function $f_\phi$ during training, it relies exclusively on $S_\theta$ during testing. Consequently, the testing execution time for this approach remains consistent with that of PGD-Net.

Figure 5 presents the reconstruction results of knee data using various methods under a 1D random trajectory with an acceleration factor of 8. Similar to the previous experiment, SPIRiT, HDSLR, and OT-GAN exhibit aliasing patterns. PGD-Net’s reconstruction results still
contain a slight aliasing pattern, as indicated by the red arrow. Compared to NCAPGD-Net, CAPGD-Net achieves comparable, and even clearer, reconstruction performance, particularly in capturing fine details. The quantitative metrics for these methods are shown in table 1, which align with the visual impressions from figure 5.

To further validate the effectiveness of the proposed method, we tested various techniques on different organs, specifically, brain data. Figure 6 illustrates the reconstruction results of brain data using various methods under a 2D random trajectory with an acceleration factor of 12. It is evident that SPIRiT exhibits aliasing patterns. In comparison to the reference, it is clear that the reconstruction images from HDSLr and OT-GAN are overly smooth, resulting in a loss of fine details. In contrast to the first three methods, PGD-Net and CAPGD-Net perform well in reconstructing the images. However, upon closer inspection, as indicated by the red arrows, CAPGD-Net accurately captures details, whereas PGD-Net loses some of them. Furthermore, the quantitative metrics for these methods are provided in table 1. In terms of quantitative metrics, although PGD-Net performs comparably to CAPGD-Net, considering CAPGD-Net's advantages in visual impressions, we believe that CAPGD-Net still outperforms PGD-Net.

5.2. Experiments with perturbations

When measurements contain perturbations, iterative algorithms often exhibit semi-convergence when solving inverse problems. This phenomenon implies that they initially move towards the true solution but gradually deviate from it as the iteration progresses [22]. However, theorem 3.14 demonstrates that when measurements include perturbations, the proposed CAPGD-Net terminates prematurely, and the impact of this early termination solution is not dramatically affected by the perturbations, remaining continuously dependent on the measurement.
Figure 7. Reconstruction results when an additional Gaussian noise with a weight of 1 (i.e. \(\|n\|/\|y\| = 1\), where \(n\) represents Gaussian noise) is added to the measurements. These measurements were taken under 1D uniform undersampling at \(R = 6\). The values in the corner are the NMSE/PSNR/SSIM values of each slice. The second and third rows illustrate the enlarged and error views, respectively. The grayscale of the reconstructed images and the color bar of the error images are at the right of the figure.

To validate the effectiveness of theorem 3, we conduct tests to assess the performance of the proposed method and other methods using perturbed measurements. It’s important to note that, in this experiment, OT-GAN is not considered because it is not driven by an iterative algorithm and does not fall into the category of semi-convergence. Figure 7 showcases the reconstruction results of knee data when additional Gaussian noise with a weight of 1 (i.e. \(\|n\|/\|y\| = 1\), where \(n\) represents Gaussian noise) is added to the measurements. These measurements are acquired using a 1D uniform trajectory with an acceleration factor of 6.

Compared to figure 4, it is evident that HDSLR and PGD-Net, due to the lack of effective early termination criteria, are more susceptible to perturbations in the measurements, resulting in perturbation amplification in the reconstruction results. In the case of HDSLR, it even generates singular points. We applied Morozov’s criterion to configure the traditional SPIRiT, and in comparison to the results shown in figure 4, its reconstruction is less affected by perturbations in the measurements.

In algorithm 2, if \(\tau\) is set to 7.9, CAPGD-Net terminates prematurely at the 5th layer. Significantly, when compared to HDSLr and PGD-Net, the prematurely terminated CAPGD-Net, denoted as CAPGD-Net (5), is much less affected by perturbations in the measurements. Furthermore, to confirm that CAPGD-Net’s stability to measurement perturbations is indeed due to early termination, we conduct a test with CAPGD-Net without early termination, referred to as CAPGD-Net (10). It is evident from the error view that the reconstruction results of CAPGD-Net (5) are less disturbed by noise compared to CAPGD-Net (10). Thus, we have validated the effectiveness of early termination, as described in theorem 3.14 in the unfolding method.

The competitive quantitative results of the previously mentioned methods are showcased in table 2. These outcomes align with the visual observations noted in figure 7. Notably, early termination not only aids in mitigating noise amplification but also shortens the computation.
Table 2. Quantitative comparison for various methods on the knee and brain data.

| Datasets & Methods               | NMSE      | PSNR (dB)  | SSIM      |
|----------------------------------|-----------|------------|-----------|
| SPIRiT                           | 0.0110 ± 0.0039 | 30.88 ± 2.28 | 0.7868 ± 0.0644 |
| HDSLR                            | 0.0109 ± 0.0023 | 30.80 ± 2.05 | 0.8108 ± 0.0712 |
| PGD-Net                          | 0.0142 ± 0.0034 | 29.67 ± 2.37 | 0.7814 ± 0.0837 |
| CAPGD-Net (No Early Termination) | 0.0137 ± 0.0048 | 29.94 ± 2.64 | 0.7894 ± 0.0828 |
| CAPGD-Net (Early Termination)    | 0.0103 ± 0.0026 | 31.07 ± 1.87 | 0.8351 ± 0.0603 |

Knee (uniform \( R = 6 \))

| Datasets & Methods               | NMSE      | PSNR (dB)  | SSIM      |
|----------------------------------|-----------|------------|-----------|
| SPIRiT                           | 0.0168 ± 0.0054 | 29.04 ± 2.21 | 0.7238 ± 0.0802 |
| HDSLR                            | 0.0172 ± 0.0040 | 28.87 ± 2.08 | 0.7543 ± 0.0856 |
| PGD-Net                          | 0.0224 ± 0.0050 | 27.69 ± 1.85 | 0.7381 ± 0.0910 |
| CAPGD-Net (No Early Termination) | 0.0137 ± 0.0048 | 29.94 ± 2.64 | 0.7894 ± 0.0828 |
| CAPGD-Net (Early Termination)    | 0.0103 ± 0.0026 | 31.07 ± 1.87 | 0.8351 ± 0.0603 |

Knee (random \( R = 8 \))

Note: The red values highlight the achieved optimal quantitative metrics.

Figure 8. Reconstruction results when an additional Gaussian noise with a weight of 1 (i.e., \( ||n||/||y|| = 1 \), where \( n \) represents Gaussian noise) is added to the measurements. These measurements were taken under 1D random undersampling at \( R = 8 \). The values in the corner are the NMSE/PSNR/SSIM values of each slice. The second and third rows illustrate the enlarged and error views, respectively. The grayscale of the reconstructed images and the color bar of the error images are at the right of the figure.

time. As indicated in table 1, thanks to early termination, the testing time for a single slice with CAPGD-Net has been reduced from 0.05 s to 0.03 s.

To further validate the stability of the proposed method against measurement perturbations, figure 8 showcases the reconstruction results of knee data when additional Gaussian noise with
Figure 9. Different iterations of reconstructions under 1D random undersampling at $R = 8$. The values in the corner are each slice’s NMSE/PSNR/SSIM values. The grayscale of the reconstructed images is at the right of the figure.

A weight of 1 (i.e. $\|n\|/\|y\| = 1$, where $n$ represents Gaussian noise) is added to the measurements. These measurements were acquired using a 1D random trajectory with an acceleration factor of 8. Compared to figure 5, it is evident that the reconstruction results of HDSL, PGD-Net, and CAPGD-Net (10) without early termination are significantly impacted by measurement perturbations. In algorithm 2, if $\tau$ is set to 8.4, CAPGD-Net terminates prematurely at the 3rd layer, denoted as CAPGD-Net (3). Prematurely terminated CAPGD-Net (3) is much less affected by measurement perturbations. The competitive quantitative results are also presented in table 2, and they align with the visual observations noted in figure 7. Thus, we provide further validation of the effectiveness of theorem 3.14.

5.3. Convergence speed

In the previous section, we have asserted that, thanks to our proposed training model (algorithm 1), CAPGD-Net exhibits faster convergence compared to PGD-Net when trained end-to-end using the $L_2$-norm. To confirm this claim, figure 9 illustrates the outputs of CAPGD-Net and PGD-Net at different layers. It is evident that, while PGD-Net ensures accurate reconstruction at the final layer, the intermediate layer outputs significantly deviate from the reference. This behavior contradicts the typical performance of iterative reconstruction algorithms. It reinforces another conclusion we previously reached, which is that end-to-end training with the $L_2$-norm does not guarantee that PGD-Net can be efficiently integrated into the solving algorithm of a specific variational regularization model. Furthermore, it suggests that even when equipped with termination criteria, PGD-Net cannot provide stable solutions when confronted with measurement perturbations. In contrast, CAPGD-Net converges rapidly and
progressively improves the reconstruction quality at each layer. This fast convergence of CAPGD-Net is the key to why early termination can also yield impressive results. Figure 10 displays the convergence performance of CAPGD-Net and PGD-Net across the entire test dataset, perfectly corroborating the visual observations presented in figure 9.

6. Discussion

In this paper, we reevaluate unfolded DNNs from the perspective of iterative regularization. Then, we propose a modified training approach and configure termination criteria for unfolded DNNs, thereby establishing the unfolding method as an iterative regularization technique. However, in practical implementation and experiments (sections 4 and 5), specific details like the choice of network unfolding layers $K$ and the selection of network representation for $f_\phi$ and $S_\theta$ have not been thoroughly investigated. Therefore, this section is dedicated to addressing these issues.
6.1. The impact of unfolded layers $K$ on model performance

In unfolding models, the number of unfolded layers, denoted as $K$, is a critical factor influencing the model’s performance. In general, a higher number of layers can lead to increased complexity, improved representational capabilities, and better performance. However, two important considerations must be taken into account. First, as model complexity increases, the improvement in network performance becomes less pronounced beyond a certain threshold. Second, an increase in the number of network layers also results in higher memory requirements. Hence, finding the right trade-off between model complexity and performance is crucial.

To address this, we assess the model’s performance and memory usage at different numbers of unfolded layers ($K$) using 1D uniform undersampling at $R = 6$. As illustrated in figure 11, memory usage shows nearly linear growth with the number of unfolded layers ($K$). Concerning quantitative metrics like NMSE, PSNR, and SSIM, we observe that $K = 10$ represents a turning point for the model’s performance. Specifically, when $K$ is less than 10, the model’s
performance benefits from increasing the number of unfolded layers ($K$). However, when $K$ exceeds 10, further increases in the number of unfolded layers have only marginal effects on the model’s performance. Therefore, selecting $K = 10$ strikes a numerically optimal balance between model complexity and performance.

6.2. The impact of network representation for $S_{\theta_k}$ and $f_\phi$ on model performance

Theoretically, as stated in theorems 3.3, 3.11, and 3.14, it only requires $f_\phi : X \to \mathbb{R}$ to be a convex function, and $S_{\theta_k} : X \to X$ approximates the proximal mapping of $f_\phi$. We do not have specific requirements for how to represent them in the neural network.

However, in practice, it is not always guaranteed that the minima of the loss functions (3.7) and (3.8) can be precisely obtained during the training of networks. Therefore, to ensure that the proximal mapping of $f_\phi$ can be approximated more accurately (i.e. making it numerically more accessible to access the minimum of the loss function (3.8)), we prefer to keep the structure of $f_\phi$ as simple as possible. Regarding $S_{\theta_k}$, as it needs to approximate the proximal mapping of the neural network $f_\phi$, we aim for it to have sufficient representational capacity (i.e. being numerically capable of optimizing the loss function (3.7) more accurately). Therefore, in section 4, we choose a simple pyramid-shaped ICNN for $f_\phi$ and a network with strong representation capability (i.e. Unet) for $S_{\theta_k}$.

To validate the above assertion, we conduct experiments to explore the influence of different network structures for $f_\phi$ and $S_{\theta_k}$ on reconstruction results. Firstly, we examine the impact of different network representations for $S_{\theta_k}$. With $f_\phi$ fixed as a pyramid-shaped ICNN, we assess the model’s performance when $S_{\theta_k}$ is represented by a network with strong representational capability, Unet, and a simpler 5-layer CNN following [39, 43] respectively. Figure 12 demonstrates that the reconstruction results based on the 5-layer CNN representation of $S_{\theta_k}$, labeled as ‘CNN+pyramid’, exhibit aliasing patterns, whereas those based on the Unet representation, labeled as ‘Unet+pyramid’, are more accurate. This implies that compared to Unet, the 5-layer CNN has a weaker ability to approximate the proximal mapping of $f_\phi$. Therefore, Unet is a more suitable choice for $S_{\theta_k}$. With $S_{\theta_k}$ fixed as a Unet, we explore how different choices of network structures for $f_\phi$ affect the reconstruction results. We compare the ResNet-based ICNN [12], known for its more complex structure and larger number of parameters, with the pyramid-shaped ICNN. While the ResNet-based ICNN has the potential to capture more image features, figure 12 clearly demonstrates that, as indicated by the red arrows, the reconstructed image with ResNet-based ICNN as a penalty function, labeled ‘Unet+ResNet’, still exhibits aliasing patterns. This suggests that the ResNet-based ICNN, despite its potential to capture more features, has a more complex proximal mapping and struggles to approximate it accurately. Therefore, the structurally simpler pyramid-shaped ICNN is a more suitable choice for $f_\phi$. Finally, we test the scenario where $f_\phi$ is ResNet and $S_{\theta_k}$ is a 5-layer CNN, referred to as ‘CNN+ResNet’. Quantitative metrics in table 3 indicate that this combination yielded the poorest performance. In summary, these experiments effectively validate our assertion regarding the choice of network representations for $f_\phi$ and $S_{\theta_k}$.

6.3. Application to alternative imaging modalities

The method proposed in this paper is designed to address general inverse problems (2.1). While we validate it by solving the MRI inverse problem as an example, its applicability to alternative imaging modalities is a worthwhile exploration. The main difference between MRI and other imaging modalities is that MRI relies on the Fourier transform, which possesses properties of
Table 3. Quantitative comparison for different combination of $S_{\theta_k}$ and $f_{\phi}$ on the knee data.

| Datasets & Methods | NMSE     | PSNR (dB) | SSIM    | Testing Time (s) |
|--------------------|----------|-----------|---------|------------------|
| Knee (uniform $R = 6$) |          |           |         |                  |
| CNN + ResNet       | 0.0052 ± 0.0016 | 34.22 ± 1.13 | 0.9114 ± 0.0183 | 0.03 |
| CNN + Pyramid      | 0.0043 ± 0.0012 | 34.87 ± 1.11 | 0.9191 ± 0.0174 | 0.03 |
| Unet + ResNet      | 0.0052 ± 0.0023 | 34.27 ± 1.27 | 0.9172 ± 0.0217 | 0.05 |
| Unet + Pyramid     | 0.0042 ± 0.0016 | 35.10 ± 0.96 | 0.9258 ± 0.0175 | 0.05 |

Note: The red values highlight the achieved optimal quantitative metrics.

Figure 12. Reconstruction results for different combination of $S_{\theta_k}$ and $f_{\phi}$ under 1D uniform undersampling at $R = 6$. The values in the corner are the NMSE/PSNR/SSIM values of each slice. The second and third rows illustrate the enlarged and error views, respectively. The grayscale of the reconstructed images and the color bar of the error images are at the right of the figure.

‘fast computation’ and ‘linearity’. Therefore, we will discuss scenarios where these properties are absent.

In the case of an imaging system $A$ based on a non-fast computable transform (e.g. CT, PET), the PGD unfolded network we employ only involves the computation of the imaging system operators $A$ and $A^*$, without requiring higher-order or complex computations on operator $A$. It is worth noting that similar algorithms based on operators $A$ and $A^*$ have already been successfully applied to CT and PET imaging [44, 52]. Therefore, our method can be adapted to imaging modalities that are more computationally complex than MRI. Furthermore, even if imaging system $A$ is nonlinear, it is not particularly challenging to extend the methods and theories proposed in this paper to handle nonlinear inverse problems, as demonstrated in [26]. Consequently, we believe that the methods and theories presented in this paper have the potential to be applied to alternative imaging modalities.
7. Conclusion

In this paper, we proposed a modified training approach and configured termination criteria for unfolded DNNs, thereby establishing the unfolding method as an iterative regularization technique. Specifically, our method involved the joint learning of a convex penalty function using an ICNN to quantify the distance to a real data manifold. Then, we trained a DNN unfolded from the PGD algorithm, incorporating this learned penalty. We have proven that the unfolded DNN converges to a unique solution stably. Experimentally, we demonstrated, with an example of MRI reconstruction, that the proposed method outperformed conventional unfolding methods and traditional regularization methods in terms of reconstruction quality, stability, and convergence speed.

Data availability statement

The data cannot be made publicly available upon publication because they are owned by a third party and the terms of use prevent public distribution. The data that support the findings of this study are available upon reasonable request from the authors. https://github.com/ZhuoxuCui.

Acknowledgments

This work was supported in part by the National Key R & D Program of China (2022YFA1004203 and 2021YFF0501503); National Science Foundation of China (62331028, U21A6005, 62125111, 12026603, 62206273, 62201561, 62106252); Key Laboratory for Magnetic Resonance and Multimodality Imaging of Guangdong Province (2023B1212060052).

Appendix

A.1. The proof of theorem 3.3

Proof. This proof follows the technical route in [32]. First, we prove that the distance function $d_M$ is convex and 1-Lipschitz continuous. By the definition of $d_M$, for any $\lambda \in (0, 1)$, $x, y \in X$, we have

$$d_M(\lambda x + (1-\lambda)y) = \min_{z \in M} \|\lambda x + (1-\lambda)y - z\|$$

$$\leq \|\lambda x + (1-\lambda)y - \lambda x^* + (1-\lambda)y^*\|$$

$$\leq \lambda \|x - x^*\| + (1-\lambda) \|y - y^*\|$$

$$= \lambda d_M(x) + (1-\lambda) d_M(y)$$

where $x^* \in \arg \min_{z \in M} \|z - x\|$ and $y^* \in \arg \min_{z \in M} \|z - y\|$. For any $x, y \in X$, we have

$$d_M(x) - d_M(y) = \min_{z \in M} \|x - z\| - \min_{z \in M} \|y - z\|$$

$$\leq \|x - y^*\| - \|y - y^*\| \leq \|x - y\|$$

where $y^* \in \arg \min_{z \in M} \|z - y\|$. Thus, we have proved the convexity and 1-Lipschitz continuity of $d_M$. 
Next, we prove that the minimum of functional (3.9) is \( d_M \). For any convex and 1-Lipschitz continuous non-negative function \( f \), we have

\[
\frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{Z}}}[f(\mathbf{Z}^k)] - \mathbb{E}_{\mathbf{X} \sim P_{\mathbf{X}}}[f(\mathbf{X})] = \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{Z}}}[f(\mathbf{Z}^k) - f(P_M(\mathbf{Z}^k))] \\
\quad \leq \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{Z}}}[\|\mathbf{Z}^k - P_M(\mathbf{Z}^k)\|] \\
= \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{Z}}}[d_M(\mathbf{Z}^k)] \\
= \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{Z}}}[d_M(\mathbf{Z}^k)] - \mathbb{E}_{\mathbf{X} \sim P_{\mathbf{X}}}[d_M(\mathbf{X})]
\]

where the first equality is due to assumption 3.2, the first inequality is due to the 1-Lipschitz continuity of \( f \), the second equality is due to the definition of \( d_M \) and the last equality is due to the fact that \( \mathbb{E}_{\mathbf{X} \sim P_{\mathbf{X}}}[d_M(\mathbf{X})] = 0 \). The above inequality shows that \( d_M \) is the convex and 1-Lipschitz continuous non-negative minimum of functional (3.9).

\[\square\]

A.2. The proof of proposition 3.7

Proof. Let \( x^* \) be the \( f_{\partial \varphi} \)-minimizing solution of inverse problem (2.1), we have

\[
\frac{1}{2} \|x_{k+1}^0 - x^*\|^2 - \frac{1}{2} \|x_k^0 - x^*\|^2 \\
= \frac{1}{2} \|x_{k+1}^0 - x^*\|^2 - \frac{1}{2} \|x_k^0 - x_{k+1}^0\|^2 - \frac{1}{2} \|x_{k+1}^0 - x^*\|^2 - \langle x_k^0 - x_{k+1}^0, x_{k+1}^0 - x^* \rangle \\
= -\frac{1}{2} \|x_k^0 - x_{k+1}^0\|^2 - \langle \eta A^* (Ax_k^0 - y^0) + \partial_{x_k^0} f_{\partial \varphi} (x_{k+1}^0), x_{k+1}^0 - x^* \rangle \\
\leq -\frac{1}{2} \|x_k^0 - x_{k+1}^0\|^2 + f_{\partial \varphi} (x^*) - f_{\partial \varphi} (x_{k+1}^0) + \epsilon_k - \eta \langle A^* (Ax_k^0 - y^0), x_{k+1}^0 - x^* \rangle \\
\leq -\frac{1}{2} \|Ax_k^0 - Ax_{k+1}^0\|^2 + f_{\partial \varphi} (x^*) - f_{\partial \varphi} (x_{k+1}^0) + \epsilon_k - \eta \|Ax_k^0 - y^0\|^2 \\
- \eta \langle Ax_k^0 - y^0, Ax_{k+1}^0 - Ax_k^0 + y^0 - y \rangle \\
\leq f_{\partial \varphi} (x^*) - f_{\partial \varphi} (x_{k+1}^0) - \left( \eta - \frac{\eta^2}{2} \right) \|Ax_k^0 - y^0\|^2 + \eta \delta \|Ax_k^0 - y^0\| + \epsilon_k
\]

where \( \partial_{x_k^0} f_{\partial \varphi}(x_{k+1}^0) \) denotes the \( \epsilon_k \)-subdifferential of \( f_{\partial \varphi} \) at \( x_{k+1}^0 \) such that \( f_{\partial \varphi}(x^*) - f_{\partial \varphi}(x_{k+1}^0) \geq \langle \partial_{x_k} f_{\partial \varphi}(x_{k+1}^0), x^* - x_{k+1}^0 \rangle - \epsilon_k \), the first inequality is due to assumption 3.5, the second inequality is due to \( \|A\| \leq 1 \) and the last inequality is due to the fact that \( -a^2 - 2(a, b) \leq b^2 \).

Assume that \( k_* \) is the maximum number of iteration steps before algorithm 2 triggers the termination criterion (3.10), we have
When the measurement contains no perturbation, i.e., $\delta = 0$, from inequality (A.11), we have

$$f_{o^r}(x_{k+1}) - f_{o^r}(x^*) + \left( \eta - \frac{\eta^2}{2} \right) \|Ax_k - y\|^2 \leq \frac{1}{2} \|x_k - x^*\|^2 - \frac{1}{2} \|x_{k+1} - x^*\|^2 + \epsilon_k$$

Summing both sides of the above inequality from 0 to $+\infty$, we get:

$$\sum_{k=1}^{+\infty} \left[ f_{o^r}(x_{k+1}) - f_{o^r}(x^*) + \left( \eta - \frac{\eta^2}{2} \right) \|Ax_k - y\|^2 \right] \leq \frac{1}{2} \|x_0 - x^*\|^2 + \sum_{k=1}^{+\infty} \epsilon_k \quad (A.12)$$

Then, we have

$$\lim_{k \to +\infty} f_{o^r}(x_k) = f_{o^r}(x^*) \text{ and } \lim_{k \to +\infty} \|Ax_k - y\| = 0 \quad (A.13)$$

If $\phi^T$ reaches the minimum of (3.8) in algorithm 1, theorem 3.3 shows that $f_{o^r}(x_k) = d_{\mathcal{M}}(x_k)$. Define $x_k' = P_{\mathcal{M}}(x_k)$ and $x_k'' = \arg \min_{x' \in \mathcal{X}} \|x' - x\|$, we have

$$\|x_k' - x_k''\| \leq \|x_k' - x_k\| + \|x_k'' - x_k\| = d_{\mathcal{M}}(x_k) + \|x_k'' - x_k\|$$

From (A.13) we know that $\lim_{k \to +\infty} \|x_k' - x_k''\| = 0$. If further assumption 3.9 holds, we have $\lim_{k \to +\infty} x_k' = \lim_{k \to +\infty} x_k'' = x^* = x^t$. Then, we have

$$\|x_k - x^t\| \leq d_{\mathcal{M}}(x_k) + \|x_k' - x^t\|$$

which implies that $\|x_k - x^t\| \to 0$ as $k \to +\infty$.

If $\phi^T$ cannot reach the minimum of (3.8) in algorithm 1 but $f_{o^r}$ is $\sigma$-strongly convex, inequality (A.12) implies that

$$\sum_{k=1}^{+\infty} \sigma \|x_k - x^*\|^2 \leq \frac{1}{2} \|x_0 - x^*\|^2 + \sum_{k=1}^{+\infty} \epsilon_k$$

Then, we show that $x_k$ converges to a $f_{o^r}$-minimizing solution of inverse problem (2.1) as $k \to +\infty$. \qed
A.4. The proof of theorem 3.14

Proof. Let \( \delta_n \) be a sequence converging to 0 as \( n \to +\infty \) and \( k_n = k_\ast (\delta_n, y_n^\ast) \) be the maximum number of iteration steps before algorithm 2 triggers the termination criterion (3.10).

For any finite accumulation point \( k \) of \( \{ k_n \} \), without loss of generality, we can assume that \( k_n = k \) for all \( n \). As \( k \) fixed, \( x_n^{k_n} \) depends continuously on \( y_n^\ast \), we have

\[
x_n^{k_n} \to x_k \quad \text{and} \quad \|Ax_n^{k_n} - y_n^\ast\|^2 + f_\Omega \left( x_n^{k_n} \right) - f_\Omega^\ast \to 0
\]

as \( n \to +\infty \), which means that \( x_k \) is a \( f_\Omega^\ast \)-minimizing solution of inverse problem (2.1).

Consider the case \( k_n \to +\infty \) as \( n \to +\infty \). For \( k < k_n \), we have

\[
\|x_n^{k_n} - x^\ast\| \leq \|x_k - x^\ast\| + \sum_{i=k+1}^{k_n} \epsilon_i \leq \|x_k - x^\ast\| + \sum_{i=k+1}^{+\infty} \epsilon_i
\]

where the first inequality is due to the inequality (A.11). Given \( \varepsilon > 0 \), we can fix some \( k_1(\varepsilon) \) such that \( \sum_{i=k+1}^{+\infty} \epsilon_i < \varepsilon / 3 \) for any \( k \geq k_1(\varepsilon) \), and we can also fix some \( k_2(\varepsilon) \) such that \( \|x_k - x^\ast\| < \varepsilon / 3 \) for any \( k \geq k_2(\varepsilon) \). Because \( x_n^{k_n} \) depends continuously on \( y_n^\ast \), we can fix some \( n(\varepsilon, k) \) such that \( \|x_n^{k_n} - x_k\| < \varepsilon / 3 \) for any \( n > n(\varepsilon, k) \). Since \( k_n \to +\infty \) as \( n \to +\infty \), for any \( \varepsilon > 0 \), we have \( \|x_n^{k_n} - x^\ast\| < \varepsilon \) as \( k_n > k := \max\{k_1(\varepsilon), k_2(\varepsilon)\}, n > n(\varepsilon, k) \). \( \square \)

ORCID iDs

Zhuo-Xu Cui 🐦 https://orcid.org/0000-0001-9283-881X
Bo Zhang 🐦 https://orcid.org/0000-0001-7355-0941
Dong Liang 🐦 https://orcid.org/0000-0003-1358-9777

References

[1] Adler J and Öktem O 2018 Learned primal-dual reconstruction IEEE Trans. Med. Imaging 37 1322–32
[2] Aggarwal H K, Mani M P and Jacob M 2019 MoDL: model-based deep learning architecture for inverse problems IEEE Trans. Med. Imaging 38 394–405
[3] Antil H, Di Z W and Khatri R 2020 Bilevel optimization, deep learning and fractional Laplacian regularization with applications in tomography Inverse Problems 36 064001
[4] Antun V, Renna F, Poon C, Adcock B and Hansen A C 2020 On instabilities of deep learning in inverse problems IEEE Trans. Med. Imaging 40 3140–53
[5] Cao C, Cui Z-X, Cheng J, Jia S, Zheng H, Liang D and Zhu Y 2023 SPIRiT-diffusion: SPIRiT-driven score-based generative modeling for vessel wall imaging Proc. Int. Society for Magnetic Resonance in Medicine
[6] Cui Z-X et al 2023 K-UNN: k-space interpolation with untrained neural network Med. Image Anal. 88 102877
[7] Dong B et al 2010 MRA-based wavelet frames and applications The Mathematics of Image Processing (IAS Lecture Notes Series vol 19) (Park City Mathematics Institute) pp 9–158
[12] Du Y and Mordatch I 2019 Implicit generation and modeling with energy based models *Advances in Neural Information Processing Systems* vol 32
[13] Engl H W, Hanke M and Neubauer A 1996 *Regularization of Inverse Problems* (Springer)
[14] Gregor K and LeCun Y 2010 Learning fast approximations of sparse coding *Int. Conf. on Int. Conf. on Machine Learning* pp 399–406
[15] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V and Courville A C 2017 Improved training of Wasserstein GANs *Advances in Neural Information Processing Systems* vol 30
[16] Hanke M, Neubauer A and Scherzer O 1995 A convergence analysis of the Landweber iteration for nonlinear ill-posed problems *Numer. Math.* 72 21–37
[17] Heaton H, Fung S W, Lin A T, Osher S and Yin W 2022 Wasserstein-based projections with applications to inverse problems *SIAM J. Math. Data Sci.* 4 581–603
[18] Huang W, Ke Z, Cui Z-X, Cheng J, Qiu Z, Jia S, Ying L, Zhu Y and Liang D 2021 Deep low-rank plus sparse network for dynamic MR imaging *Med. Image Anal.* 73 102190
[19] Ioffe S and Szegedy C 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift *Int. Conf. on Machine Learning* vol 37 pp 448–56
[20] Isola P, Zhu J-Y, Zhou T and Efros A A 2017 Image-to-image translation with conditional adversarial networks *Conf. on Computer Vision and Pattern Recognition*
[21] Ito K and Jin B 2014 *Inverse Problems: Tikhonov Theory and Algorithms* (World Scientific)
[22] Jiao Y, Jin Q, Lu X and Wang W 2016 Alternating direction method of multipliers for linear inverse problems *SIAM J. Numer. Anal.* 54 2114–37
[23] Jiao Y, Jin Q, Lu X and Wang W 2017 Preconditioned alternating direction method of multipliers for inverse problems with constraints *Inverse Problems* 33 025004
[24] Jin Q 2015 Inexact Newton–Landweber iteration in Banach spaces with nonsmooth convex penalty terms *SIAM J. Numer. Anal.* 53 2389–413
[25] Jin Q and Lu X 2014 A fast nonstationary iterative method with convex penalty for inverse problems in Hilbert spaces *Inverse Problems* 30 045012
[26] Kaltenbacher B, Neubauer A and Scherzer O 2008 *Iterative Regularization Methods for Nonlinear Ill-Posed Problems* ( Walter De Gruyter)
[27] Ke Z, Cui Z-X, Huang W, Cheng J, Jia S, Ying L, Zhu Y and Liang D 2021 Deep manifold learning for dynamic MR imaging *IEEE Trans. Comput. Imaging* 7 1314–27
[28] Ke Z et al 2021 Learned low-rank priors in dynamic MR imaging *IEEE Trans. Med. Imaging* 40 3698–710
[29] Kingma D P and Ba J 2015 Adam: a method for stochastic optimization *Int. Conf. on Learning Representations*
[30] Liang D, Cheng J, Ke Z and Ying L 2020 Deep magnetic resonance image reconstruction: inverse problems meet neural networks *IEEE Signal Process. Mag.* 37 141–51
[31] Liang Z-P, Boada F E, Constable R T, Haacke E M, Lauterbur P C and Smith M R 1992 Constrained reconstruction methods in MR imaging *Rev. Magn. Reson. Med.* 4 67–185 (available at: http://mri.beckman.illinois.edu/publications_total.html)
[32] Lunz S, Öktem O and Schönlieb C-B 2018 Adversarial regularizers in inverse problems *Advances in Neural Information Processing Systems* vol 31
[33] Lustig M and Pauly J M 2010 SPIRiT: iterative self-consistent parallel imaging reconstruction from arbitrary k-space *Magn. Reson. Med.* 64 457–71
[34] Mallat S 1999 A Wavelet Tour of Signal Processing (Academic)
[35] Miyato T, Kataoka T, Koyama M and Yoshida Y 2018 Spectral normalization for generative adversarial networks *Int. Conf. on Learning Representations*
[36] Nair M T, Tautenhahn U and Schock E 2003 Morozov’s discrepancy principle under general source conditions *Z. Anal. Anwend.* 22 199–214
[37] Oh G, Sim B, Chung H, Sunwoo L and Ye J C 2020 Unpaired deep learning for accelerated MRI using optimal transport driven cycleGAN *IEEE Trans. Comput. Imaging* 6 1285–96
[38] Paszke A et al 2019 PyTorch: an imperative style, high-performance deep learning library *Conf. on Neural Information Processing Systems* (Curran Associates Inc.)
[39] Pramanik A, Aggarwal H K and Jacob M 2020 Deep generalization of structured low-rank algorithms (Deep-SLR) *IEEE Trans. Med. Imaging* 39 4186–97
[40] Ron A and Shen Z 1997 Affine systems in $L_2 (\mathbb{R}^d)$: the analysis of the analysis operator *J. Funct. Anal.* 148 408–47
[41] Ronneberger O, Fischer P and Brox T 2015 U-Net: convolutional networks for biomedical image segmentation *Medical Image Computing and Computer-Assisted Intervention* pp 234–41
[42] Rudin L I, Osher S and Fatemi E 1992 Nonlinear total variation based noise removal algorithms Physica D 60 259–68
[43] Sriram A, Zbontar J, Murrell T, Defazio A, Zitnick C L, Yakubova N, Knoll F and Johnson P 2020 End-to-end variational networks for accelerated MRI reconstruction Medical Image Computing and Computer Assisted Intervention pp 64–73
[44] Su T et al 2022 Generalized deep iterative reconstruction for sparse-view CT imaging Phys. Med. Biol. 67 025005
[45] Tikhonov A N 1943 On the stability of inverse problems Proc. USSR Acad. Sci. 39 195–8
[46] Wang S, Su Z, Ying L, Peng X, Zhu S, Liang F, Feng D and Liang D 2016 Accelerating magnetic resonance imaging via deep learning Int. Symp. on Biomedical Imaging pp 514–7
[47] Wang Z, Bovik A C, Sheikh H R and Simoncelli E P 2004 Image quality assessment: from error visibility to structural similarity IEEE Trans. Image Process. 13 600–12
[48] Würfl T, Ghesu FC, Christlein V and Maier A 2016 Deep learning computed tomography Int. Conf. on Medical Image Computing and Computer-Assisted Intervention pp 432–40
[49] Yang Y, Sun J, Li H and Xu Z 2016 Deep ADMM-Net for compressive sensing MRI Advances in Neural Information Processing Systems vol 29
[50] Yang Y, Sun J, Li H and Xu Z 2018 ADMM-CSNet: a deep learning approach for image compressive sensing IEEE Trans. Pattern Anal. Mach. Intell. 42 521–38
[51] Zhang J and Ghanem B 2018 ISTA-Net: interpretable optimization-inspired deep network for image compressive sensing Conf. on Computer Vision and Pattern Recognition pp 1828–37
[52] Zhang Q et al 2023 Deep generalized learning model for PET image reconstruction IEEE Trans. Med. Imaging 43 1
[53] Jure Z 2018 FastMRI: an open dataset and benchmarks for accelerated MRI (arXiv:1811.08839)