Nest-DGIL: Nesterov-optimized Deep Geometric Incremental Learning for CS Image Reconstruction

Xiaohong Fan, Yin Yang, Ke Chen, Yujie Feng, and Jianping Zhang

Abstract*—Proximal gradient-based optimization is one of the most common strategies for solving image inverse problems as well as easy to implement. However, these techniques often generate heavy artifacts in image reconstruction. One of the most popular refinement methods is to fine-tune the regularization parameter to alleviate such artifacts, but it may not always be sufficient or applicable due to increased computational costs. In this work, we propose a deep geometric incremental learning framework based on second Nesterov proximal gradient optimization. The proposed end-to-end network not only has the powerful learning ability for high/low frequency image features, but also can theoretically guarantee that geometric texture details will be reconstructed from preliminary linear reconstruction. Furthermore, it can avoid the risk of intermediate reconstruction results falling outside the geometric decomposition domains and achieve fast convergence. Our reconstruction framework is decomposed into four modules including general linear reconstruction, cascade geometric incremental restoration, Nesterov acceleration and post-processing. In the image restoration step, a cascade geometric incremental learning module is designed to compensate for the missing texture information from different geometric spectral decomposition domains. Inspired by overlap-tile strategy, we also develop a post-processing module to remove the block-effect in patch-wise-based natural image reconstruction. All parameters in the proposed model are learnable, an adaptive initialization technique of physical-parameters is also employed to make model flexibility and ensure converging smoothly. We compare the reconstruction performance of the proposed method with existing state-of-the-art methods to demonstrate its superiority. Our source codes are available at *https://github.com/fanxiaohong/Nest-DGIL.

***Index Terms*—CS image reconstruction, Unfolding explainable network, Geometric incremental learning, Sparse-view CT, Operator spectral decomposition, Nesterov acceleration.**

This work was supported by the National Natural Science Foundation of China Project (11771369, 12071402, 12261131501), the Education Bureau of Hunan Province, P. R. China (22A0119), the National Key Research and Development Program of China (2020YFA0713503), the Natural Science Foundation of Hunan Province (2020JJ2027, 2020ZYT003), the Project of Scientific Research Fund of the Hunan Provincial Science and Technology Department (2022RC3022), the Postgraduate Scientific Research Innovation Project of Hunan Province (CX20210598) and Postgraduate Scientific Research Innovation Project of Xiangtan University (XDCX2021B097), P. R. China. (Corresponding author: Jianping Zhang).

X. Fan and Y. Feng are with the School of Mathematics and Computational Science, Xiangtan University, and Hunan Key Laboratory for Computation and Simulation in Science and Engineering, Xiangtan 411105, China (fanxiaohong@smail.xtu.edu.cn, fengyujie@smail.xtu.edu.cn).

Y. Yang is with the School of Mathematics and Computational Science, Xiangtan University, and Hunan National Applied Mathematics Center, Xiangtan 411105, China (yangyinxtu@xtu.edu.cn).

K. Chen is with the Centre for Mathematical Imaging Techniques and Department of Mathematical Sciences, The University of Liverpool, Liverpool, Merseyside L6972L, United Kingdom (ke.chen@liverpool.ac.uk)

J. Zhang is with the School of Mathematics and Computational Science, Xiangtan University, and Key Laboratory of Intelligent Computing & Information Processing of Ministry of Education, Xiangtan 411105, China. (jpzhang@xtu.edu.cn).

I. Introduction

IMAGE reconstruction is one of the most basic problems in computer vision and also is a challenging problem in medical imaging community. One of the most popular approaches is Compressed Sensing (CS) reconstruction that refers to the process of reconstructing an image $\boldsymbol{x} \in \mathbb{R}^N$ from the imaging system defined by

$$\boldsymbol{\Phi x} = \boldsymbol{y}, \tag{1}$$

where an observation $\boldsymbol{y} \in \mathbb{R}^M$ $(M = m_1 m_2)$ has been sampled significantly below Shannon-Nyquist rate [1]–[4], $\boldsymbol{x}$ and $\boldsymbol{y}$ are resized from $n_1 \times n_2$ and $m_1 \times m_2$ images respectively. It has been widely used in single-pixel cameras [5], accelerating magnetic resonance imaging (MRI) [6], sparse-view computational tomography (CT) [7], high-speed videos [8] and other fields. A new class of reconstruction methods for robust structured compressible signal recovery is proposed [9]. Therefore, CS reconstruction has become a powerful image compressing and reconstructing tool.

The task (1) of reconstructing an unknown $\boldsymbol{x}$ from under-sampled observation $\boldsymbol{y}$, is ill-posed due to the under-determination of linear measurement matrix $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N}$ $(M \ll N)$. Therefore, the classical mathematical model of image reconstruction can be given by

$$\min_{\boldsymbol{x} \in \mathcal{X}} \left\{ \mathcal{S}(\boldsymbol{x}) + \lambda \mathcal{R}(\boldsymbol{x}) \right\}, \tag{2}$$

where $\mathcal{R}(\boldsymbol{x})$ is a regularizer with image geometric prior (*e.g.* sparsity and smoothness etc.), λ is a regularization parameter that balances data fidelity term $\mathcal{S}(\boldsymbol{x}) = \frac{1}{2} \left\| \boldsymbol{\Phi x} - \boldsymbol{y} \right\|_2^2$ and image prior $\mathcal{R}(\boldsymbol{x})$. The CS ratio is defined as $\frac{M}{N}$, $\mathcal{X}$ is a kind of image space. $\mathcal{R}(\boldsymbol{x})$ can be carried out in image domain or traditional transform domain (such as gradient domain, wavelet domain, etc.) [10]–[14], however it cannot adequately represent the complex textures of image.

To solve (2), many iterative algorithms have been developed, such as iterative shrinkage-thresholding algorithm (ISTA) [10], [15], fast iterative shrinkage-thresholding algorithm (FISTA) [16], alternating direction method of multipliers (ADMM) [17], and so on. These methods are based on interpretable and predefined image geometric prior rather than learning directly; most of them have advantages of theoretical analysis and strong convergence. However, they not only take hundreds of iterations to converge but also have to face with the difficulty of choosing optimal geometric prior $\mathcal{R}(\boldsymbol{x})$ and physical parameters, which may lead to non-optimal image reconstruction results.

In fact, the optimal solution of (2) can be given by

$$g(\boldsymbol{x}) := \boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{x} + \lambda\mathcal{R}'(\boldsymbol{x}) = \boldsymbol{\Phi}^T\boldsymbol{y}. \tag{3}$$

Especially if the operator $g(\cdot)$ is an one-to-one smooth mapping, thus the optimization algorithm is seeking a suitable non-linear reconstruction mapping $f(\cdot)$ to achieve image reconstruction, which is denoted by

$$\boldsymbol{x} = f(\boldsymbol{y}) = g^{-1}(\boldsymbol{\Phi}^T\boldsymbol{y}).$$

It is well known that high-quality reconstruction consists of three key factors, i.e. the geometric prior of $\boldsymbol{x}$ in image space $\mathcal{X}$, non-linear mapping f and measurement matrix Φ. In this work, Φ is assumed to be a known prior, thus a neural network can be learnt to solve the optimization problem defined by

$$\mathcal{E}(\mathcal{X}) = \inf_{f}\sup_{\boldsymbol{x}\in\mathcal{X}} \|\boldsymbol{x} - f(\boldsymbol{\Phi}\boldsymbol{x})\|. \tag{4}$$

The popular convolutional neural network (CNN) is suitable for learning the non-linear reconstruction mapping f directly in training space $\mathcal{X}$ [18]–[23]. Recently, regularization model architecture is embedded into deep convolutional neural network (DnCNN) for image denoising [24]. With the observation that unrolled iterative methods have the form of a CNN (filtering followed by pointwise nonlinearity), an indirect inversion approximated by a CNN is proposed to solve normal-convolutional inverse problems [25], such network combines multi-resolution decomposition and residual learning to remove undesirable artifacts while preserving image geometries [26]. An untrained image reconstruction framework called the Deep Decoder is proposed to generate natural images by using very few weight parameters [27]. CSformer is a hybrid end-to-end CS framework, which is composed of adaptive sampling and recovery, to explore the representation capacity of local and global features [28]. In contrast to many classical model-based methods with sparsity theory, learning methods can dramatically reduce computational complexity and achieve impressive reconstruction performance. However, these existing deep learning-based methods are trained as a black box and are driven by massive training data, with limited theoretical insights from the geometric characteristic domains.

There are many approaches in the literature for formulating and designing a CNN architecture in terms of interpretable components. Learned Iterative Shrinkage Thresholding Algorithm (LISTA) is first proposed to learn optimal sparse codes [29]. ISTA algorithm is embedded into deep network (ISTA-Net and ISTA-Net+), and the aim is to learn proximal mapping using nonlinear transforms [30]. FISTA-Net that consists of gradient descent, proximal mapping and two-step update, is designed by mapping the FISTA algorithm into a deep network [7]. AMP-Net is established by unfolding the iterative denoising process of the well-known approximate message passing (AMP) algorithm rather than learning regularization terms [31]. These networks designing deep architectures have theoretical support and can alleviate the instability of the pure data-driven methods, we also refer readers to [32]–[49] for more details.

However, the existing deep unfolding methods still have the drawback of insufficiently theoretical relationship between optimization theory and network architecture design. ISTA-Net+ and FISTA-Net have neglected other regularization priories (just ℓ_1 prior information), the nonlinear transform and its inverse transform are directly replaced by several convolution layers, which leads to not straight reasonable explanation. Although AMP-Net iterates the denoising process rather than learning regularization terms, the CNN denoiser is not well analyzed and explained. Unfortunately, the two-step update, which is a well-designed linear combination of previous reconstructions in FISTA-Net, runs the risk of being outside the definition domain and losing meaning [50], [51].

To overcome above drawbacks of existing deep unfolding methods, inspired by the operator spectral decomposition, we reformulate a new geometric characteristic series dealing with the proximal-point update of the classical reconstruction, to develop a deep Geometric Incremental Learning (GIL) framework based on the second Nesterov proximal gradient optimization in this work. The derivation from mathematical theory to our network design is natural and explainable. Our contributions can be summarized as follows

- We propose a Nesterov-informed Deep Geometric Incremental Learning (Nest-DGIL) framework, which has the powerful learning ability for high/low frequency image features and can theoretically guarantee that more geometric texture details will be reconstructed from preliminary linear reconstruction. Such network gives us a new perspective to design explainable architecture.
- A cascade GIL module which is inspired by a geometric spectral decomposition of the nonlinear inverse operator and combines with a multi-regularisers truncation restoration, is designed to obtain more texture compensation from different geometric decomposition domains.
- Inspired by the second Nesterov acceleration with fast convergence rate, we adopt a smartly chosen additional estimation rather than gradient evaluation to avoid complicated calculations as well as the risk of intermediate reconstruction results falling outside the solution domains, and ensure that the approximation results of intermediate stages are meaningful.
- All the parameters in the proposed architecture are learnable end to end, and adaptive initializations are used to make model flexibility and ensure converging smoothly. Extensive experiments show that the proposed Nest-DGIL architecture outperforms other state-of-the-art methods.

The rest of paper is organized as follows. In Section II, after introducing the setup of image inverse problems and the algorithm of proximal gradient optimization, we present the second-Nesterov DGIL scheme to exactly impose update accelerations by directly modifying the neural network architecture. We then propose an operator spectral geometric decomposition approach to learn proximal-point solution in our framework. The experimental results are shown in Section III. Finally, we conclude the paper in Section IV.

II. Methodology

In this section, we propose an optimization-based image reconstruction, which is called as an explainable Nest-DGIL framework (*see* Fig. 1), and describe its specific formulation

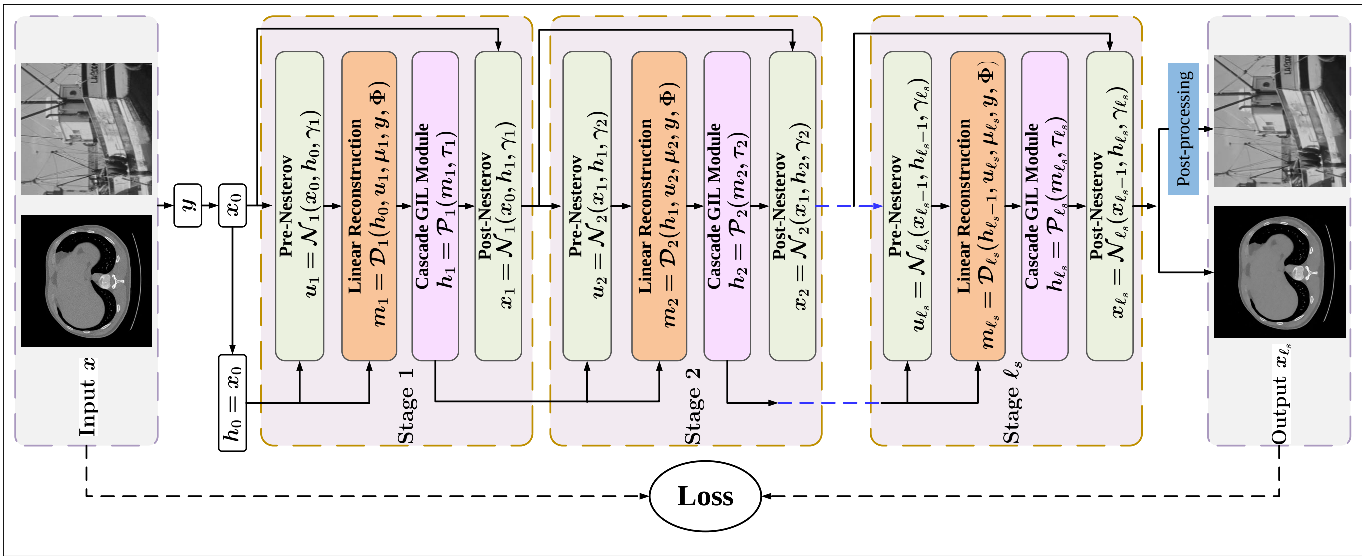

Input x
y
x_0
$h_0 = x_0$
Pre-Nesterov $u_1 = \mathcal{N}_1(x_0, h_0, \gamma_1)$
Linear Reconstruction $m_1 = \mathcal{D}_1(h_0, u_1, \mu_1, y, \Phi)$
Cascade GIL Module $h_1 = \mathcal{P}_1(m_1, \tau_1)$
Post-Nesterov $x_1 = \mathcal{N}_1(x_0, h_1, \gamma_1)$
Stage 1
Pre-Nesterov $u_2 = \mathcal{N}_2(x_1, h_1, \gamma_2)$
Linear Reconstruction $m_2 = \mathcal{D}_2(h_1, u_2, \mu_2, y, \Phi)$
Cascade GIL Module $h_2 = \mathcal{P}_2(m_2, \tau_2)$
Post-Nesterov $x_2 = \mathcal{N}_2(x_1, h_2, \gamma_2)$
Stage 2
Pre-Nesterov $u_{\ell_s} = \mathcal{N}_{\ell_s}(x_{\ell_s-1}, h_{\ell_s-1}, \gamma_{\ell_s})$
Linear Reconstruction $m_{\ell_s} = \mathcal{D}_{\ell_s}(h_{\ell_s-1}, u_{\ell_s}, \mu_{\ell_s}, y, \Phi)$
Cascade GIL Module $h_{\ell_s} = \mathcal{P}_{\ell}(m_{\ell_s}, \tau_{\ell_s})$
Post-Nesterov $x_{\ell_s} = \mathcal{N}_{\ell_s}(x_{\ell_s-1}, h_{\ell_s}, \gamma_{\ell_s})$
Stage ℓ_s
Post-processing
Output x_{ℓ_s}
Loss

Fig. 1. The overall architecture of our Nest-DGIL network. It consists of four main modules, i.e. linear reconstruction module $\mathcal{D}_k$, cascade geometric incremental learning module $\mathcal{P}_k$, Nesterov acceleration module $\mathcal{N}_k$ and post-processing module.

in detail. The proposed framework not only inherits the main advantages of the classical optimization approaches, but also explicitly incorporates the process of embedding learnable non-linear spectral geometric decomposition into deep network.

A. The second-Nesterov acceleration algorithm

Before designing an explainable learning network for image reconstruction, we first analyze a proximal gradient-based reconstruction algorithm that satisfies the linear rate of convergence $(\mathcal{O}(\frac{1}{k}))$ [52]. We recall the image reconstruction problem (2) as follow

$$\boldsymbol{x} = \arg\min_{\boldsymbol{x}\in\mathcal{X}} \left\{\mathcal{F}(\boldsymbol{x}) := \mathcal{S}(\boldsymbol{x}) + \lambda\mathcal{R}(\boldsymbol{x})\right\}. \tag{5}$$

Based on the first-order gradient descent theory, we can obtain an approximation solution of (5) via the following alternative iterations

$$\boldsymbol{m}_k = \boldsymbol{x}_{k-1} - \mu_k \boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{x}_{k-1} - \boldsymbol{y}), \tag{6}$$

$$\boldsymbol{x}_k := \arg\min_{\boldsymbol{x}} \left\{\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{m}_k\|_2^2 + \tau_k\mathcal{R}(\boldsymbol{x})\right\}, \tag{7}$$

where k denotes the iteration stage, $\mu_k = \frac{1}{\mathcal{L}_k}$ is the step-length, $\mathcal{L}_k$ is the estimated Lipschitz constant at stage k and $\tau_k = \frac{\lambda}{\mathcal{L}_k}$ is the regularization parameter. The proximal gradient-based reconstruction algorithm described above is generally considered to be a time consuming approach [17]. We can also see that it satisfies the linear rate of convergence $(\mathcal{O}(\frac{1}{k}))$ from Lemma 1 in Appendix A.

Let us now interpret the convergence rate of FISTA-type algorithms in a particular case which is representative in image inverse problems. FISTA which is a fast version of ISTA and adopts a well-designed linear combination of the previous reconstructions $\boldsymbol{x}_k$ and $\boldsymbol{x}_{k-1}$ as a refined acceleration step, has been proposed to solve (5) in the improved rate $(\mathcal{O}(\frac{1}{k^2}))$ [16]. However, the result of two-step update in FISTA has the risk of falling outside the geometric domains [51]. Inspired by the second-Nesterov acceleration [50], [51], we can approximate the solution of (5) via the following alternative iterations (denoted as Nesterov-II scheme)

$$\boldsymbol{u}_k = (1-\gamma_k)\,\boldsymbol{x}_{k-1} + \gamma_k\boldsymbol{h}_{k-1}, \tag{8}$$

$$\boldsymbol{m}_k = \boldsymbol{h}_{k-1} - \mu_k\boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{u}_k - \boldsymbol{y}), \tag{9}$$

$$\boldsymbol{h}_k = \arg\min_{\boldsymbol{x}} \left\{\frac{1}{2}\|\boldsymbol{x} - \boldsymbol{m}_k\|_2^2 + \tau_k\mathcal{R}(\boldsymbol{x})\right\}, \tag{10}$$

$$\boldsymbol{x}_k = (1-\gamma_k)\,\boldsymbol{x}_{k-1} + \gamma_k\boldsymbol{h}_k, \tag{11}$$

where γ_k is a relaxation parameter.

The Nesterov-II scheme (8)-(11) adopts a smartly chosen additional estimation rather than gradient evaluation to avoid complicated calculations and achieves accelerated convergence $(\mathcal{O}(\frac{1}{k^2}))$. In addition, the Nesterov-II scheme can ensure that the intermediate reconstruction results during iterations are all in the geometric domains [50], [51].

B. Operator spectral geometric decomposition

One of the main ingredients for constructing the explainable neural network is to determine learnable parameters by manually embedding the convolution and activation function into optimization-informed reconstruction model. The following technical decomposition is very useful for designing the learning module solving texture restoration problem (10). To extract features at more geometric domains, we consider an edge-preserving regularizer defined by

$$\mathcal{R}(\boldsymbol{x}) = \sum_{\ell=1}^{s} \phi_\ell\left(\mathcal{K}(\boldsymbol{x})\right), \text{ where } \phi_\ell(\boldsymbol{z}) = g_\ell(\|\boldsymbol{z}\|)$$

for some potential function $\phi_\ell : \mathbb{R}^d \mapsto \mathbb{R}$ defined in term of some function $g_\ell : \mathbb{R} \mapsto \mathbb{R}$, and $\mathcal{K}$ is the feature extractor. However, the TV regularizer is a special case of our proposed edge-preserving regularizer $\mathcal{R}(\boldsymbol{x})$. We employ the edge-preserving regularizer $\mathcal{R}(\boldsymbol{x})$ to extract features at more geometric domains rather than in the gradient domain only. Taking the derivative of (10) with respect to $\boldsymbol{x}$, we can obtain

$$\begin{aligned}\boldsymbol{x} =&\boldsymbol{m}_k + \tau_k \frac{\partial \mathcal{R}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{m}_k + \tau_k \sum_{\ell=1}^{s} \mathcal{K}' \phi_\ell' \left(\mathcal{K}(\boldsymbol{x})\right) \\ =&\boldsymbol{m}_k + \mathcal{K}' \psi \left(\mathcal{K}(\boldsymbol{x})\right) = \boldsymbol{m}_k + \mathcal{M}(\boldsymbol{x}),\end{aligned} \tag{12}$$

where $\mathcal{M}(\boldsymbol{x})$ denotes the nonlinear high-frequency geometric characteristic prior of image $\boldsymbol{x}$.

We can also understand the regularization operation (10) as a compensating process of restoring the missing texture $\boldsymbol{\omega}_k^* = \boldsymbol{x}^* - \boldsymbol{m}_k$ from preliminary linear reconstruction $\boldsymbol{m}_k$. Since $\boldsymbol{x}$ is unknown, it is difficult to directly solve $\boldsymbol{x}$ from (12) with non-linear feature extractor $\mathcal{M}$. One of our goals is to devise numerical techniques that naturally account for this nonlinearity by constructing accurate and efficient scheme, which can admit physically feature correction series. The main technique we will use is operator spectral decomposition [53], which is analyzed in Lemma 2 in Appendix A.

It was mentioned that the spectral radius of $\mathcal{M}(\boldsymbol{x})$ is dependent of the operator $\mathcal{K}$. If τ_k is small enough, the result in Lemma 2 easily shows that the nonlinear operator $\mathcal{M}$ satisfies the spectral constraint $\|\mathcal{M}\| < 1$. Naturally, we can employ the Taylor expansion to split the nonlinear inverse operator $(I - \mathcal{M})^{-1}$ into different geometric domains for restoring $\boldsymbol{h}_k$, an approximation of solution $\boldsymbol{x}$ in (12) is given as follow

$$\begin{aligned}\boldsymbol{h}_k &= (I - \mathcal{M})^{-1}(\boldsymbol{m}_k) = \left(I + \sum_{i=1}^{\infty} \mathcal{M}^i\right)(\boldsymbol{m}_k) \\ &= \left(\sum_{i=0}^{n} \mathcal{M}^i + \mathcal{E}(\mathcal{M}^n)\right)(\boldsymbol{m}_k) = \boldsymbol{m}_k + \sum_{i=1}^{n+1} \boldsymbol{\omega}_{k,i},\end{aligned} \tag{13}$$

where $\boldsymbol{\omega}_{k,i} = \mathcal{M}^i(\boldsymbol{m}_k)$ $(i = 1, \ldots, n)$, $I = \mathcal{M}^0$, and $\mathcal{E}(\mathcal{M}^n)(\cdot) = \theta_{\tau_k} \circ \mathcal{M}^n(\cdot)$ calculates the truncation remainder $\boldsymbol{\omega}_{k,n+1} = \mathcal{E}(\mathcal{M}^n(\boldsymbol{m}_k))$ of operator decomposition.

To proceed, let us define the left-hand and right-hand operators for any odd number i $(i \leq n)$ by (more explaination in Appendix B)

$$\begin{aligned}\boldsymbol{m}_{k,0} = \boldsymbol{m}_k,\ \cdots,\ \boldsymbol{m}_{k,i} = \psi\left(\mathcal{K}(\boldsymbol{m}_{k,i-1})\right), \\ \boldsymbol{h}_k = \boldsymbol{r}_{k,0} + \boldsymbol{m}_{k,0},\ \cdots,\ \boldsymbol{r}_{k,i-1} = \mathcal{K}'(\boldsymbol{h}_{k,i}),\end{aligned} \tag{14}$$

otherwise for any even number i $(i \leq n)$, one has

$$\begin{aligned}\boldsymbol{m}_{k,0} = \boldsymbol{m}_k,\ \cdots,\ \boldsymbol{m}_{k,i} = \mathcal{K}'(\boldsymbol{m}_{k,i-1}), \\ \boldsymbol{h}_k = \boldsymbol{r}_{k,0} + \boldsymbol{m}_{k,0},\ \cdots,\ \boldsymbol{r}_{k,i-1} = \psi\left(\mathcal{K}(\boldsymbol{h}_{k,i})\right),\end{aligned} \tag{15}$$

where $\boldsymbol{h}_{k,i} = \boldsymbol{m}_{k,i} + \boldsymbol{r}_{k,i}$. Thus we rewrite $\boldsymbol{\omega}_{k,i} \in \mathcal{X}_i$ $(i \leq n)$ and $\boldsymbol{\omega}_{k,n+1}$ as follows

$$\boldsymbol{\omega}_{k,i} = \mathcal{M}^i(\boldsymbol{m}_k) = \underbrace{\underline{\mathcal{K}'\psi(\mathcal{K}}[\cdots \underline{\mathcal{K}'\psi(\mathcal{K}}[\underline{\mathcal{K}'\psi(\mathcal{K}}}_{i}(\boldsymbol{m}_k))]])],$$

$$\boldsymbol{\omega}_{k,n+1} = \begin{cases} \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\cdots \underline{\psi(\mathcal{K}[}}_{n}\underline{\mathcal{K}'\boldsymbol{\theta}_{\tau_k}\left(\psi(\mathcal{K}\right.}(\boldsymbol{m}_{k,n}))]]\left.\right)])]), \\ \qquad\qquad \text{if } n \text{ is even number;} \\ \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\cdots \underline{[\mathcal{K}'}}_{n}\ \underline{\psi(\mathcal{K}\boldsymbol{\theta}_{\tau_k}\left(\mathcal{K}'\right.}(\boldsymbol{m}_{k,n}))]]\left.\right)]]), \\ \qquad\qquad \text{if } n \text{ is odd number.}\end{cases}$$

where $\mathcal{X}_i$ is a geometric characteristic subspace, and image space $\mathcal{X}$ is a linear combination of subspace $\{\mathcal{X}_i\}_{i=1}^{n+1}$. Finally we can define the geometric incremental component $\boldsymbol{\omega}_k$ by

$$\boldsymbol{\omega}_k = \boldsymbol{h}_k - \boldsymbol{m}_k = \sum_{i=1}^{n} \mathcal{M}^i(\boldsymbol{m}_k) + \mathcal{E}(\mathcal{M}^n(\boldsymbol{m}_k)) = \sum_{i=1}^{n+1} \boldsymbol{\omega}_{k,i}, \tag{16}$$

where $\mathcal{M}^i(\boldsymbol{m}_k)$ represents the texture compensation from feature decomposition $\boldsymbol{\omega}_{k,i}$ in subspace $\mathcal{X}_i$. We also list the spectral geometric decomposition schemes of two examples $\phi_\ell(r) = \frac{1}{2}\|r\|^2$ and $\phi_\ell(r) = \|r\|$ in Appendix C.

C. Truncation optimization problem

In this part, we aim to present an optimization minimization for modeling high-frequency feature restoration, which can help to obtain the truncation remainder $\mathcal{E}(\mathcal{M}^n(\boldsymbol{m}_k))$.

We have extracted the decomposed principal high-frequency feature component $\sum_{i=1}^{n} \boldsymbol{\omega}_{k,i}$, then we will approximate the truncation information $\boldsymbol{\omega}_{k,n+1}$ by solving the optimization with $\mathcal{R}(\boldsymbol{x}) = \|\boldsymbol{x}\|_p$ in (10), and the solution is rewritten as

$$\boldsymbol{x} = \mathrm{Prox}_{\tau_k\|\cdot\|_p}(\boldsymbol{z}_k^0) = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}_k^0\|_2^2 + \tau_k\|\boldsymbol{x}\|_p, \tag{17}$$

where $\boldsymbol{z}_k^0 = \psi(\mathcal{K}(\boldsymbol{m}_{k,n}))$ for an even number n and $\boldsymbol{z}_k^0 = \mathcal{K}'(\boldsymbol{m}_{k,n})$ for an odd number n. Relying only on ℓ_1 regularization prior is not enough to reconstruct geometric features from different abundant texture and smooth regions, thus we put forward a better truncation remainder by different regularisers $\|\boldsymbol{z}\|_p$ with different p values to extract more useful information from the remainder term. In fact, the problem (17) has the close-form solution for the given p-values [54]–[56], i.e.

$$\begin{aligned}\boldsymbol{z}_{k,1} &= \mathrm{Prox}\ {}_{\tau_k\|\cdot\|_0}\left(\boldsymbol{z}_k^0\right) = \begin{cases} \boldsymbol{z}_k^0, & |\boldsymbol{z}_k^0| \geq (2\tau_k)^{1/2} \\ 0, & \text{otherwise} \end{cases}; \\ \boldsymbol{z}_{k,2} &= \mathrm{Prox}\ {}_{\tau_k\|\cdot\|_1}\left(\boldsymbol{z}_k^0\right) = \mathrm{sign}(\boldsymbol{z}_k^0)\max\{|\boldsymbol{z}_k^0| - \tau_k, 0\}; \\ \boldsymbol{z}_{k,3} &= \mathrm{Prox}\ {}_{\tau_k\|\cdot\|_{\frac{3}{2}}}\left(\boldsymbol{z}_k^0\right) \\ &= \boldsymbol{z}_k^0 + \frac{9}{8}\tau_k^2\,\mathrm{sign}(\boldsymbol{z}_k^0)\left(1 - \sqrt{1 + \frac{16|\boldsymbol{z}_k^0|}{9\tau_k^2}}\right); \\ \boldsymbol{z}_{k,4} &= \mathrm{Prox}\ {}_{\tau_k\|\cdot\|_2}\left(\boldsymbol{z}_k^0\right) = \frac{\boldsymbol{z}_k^0}{1 + 2\tau_k}.\end{aligned}$$

By a weighted sum of different shrinkage-thresholding results, we can obtain the update scheme of the geometric incremental compensation (10) as

$$\begin{aligned}\boldsymbol{h}_k =&\boldsymbol{m}_k + \sum_{i=1}^{n} \underbrace{\underline{\mathcal{K}'\psi(\mathcal{K}}[\cdots \underline{\mathcal{K}'\psi(\mathcal{K}}[\underline{\mathcal{K}'\psi(\mathcal{K}}}_{i}(\boldsymbol{m}_k))]])] \\ &+ \begin{cases} \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\cdots \underline{\psi(\mathcal{K}[}}_{n}\underline{\mathcal{K}'\boldsymbol{\theta}_{\tau_k}\left(\psi(\mathcal{K}\right.}(\boldsymbol{m}_{k,n}))]]\left.\right)])]), \\ \qquad\qquad \text{if } n \text{ is even number;} \\ \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\cdots \underline{[\mathcal{K}'}}_{n}\ \underline{\psi(\mathcal{K}\boldsymbol{\theta}_{\tau_k}\left(\mathcal{K}'\right.}(\boldsymbol{m}_{k,n}))]]\left.\right)]]), \\ \qquad\qquad \text{if } n \text{ is odd number,}\end{cases}\end{aligned}$$

where $\theta_{\tau_k}(\boldsymbol{z}_k^0) = \sum_{i=1}^{4} \eta_{k,i}\boldsymbol{z}_{k,i}$, $\eta_{k,i} = \beta_{k,i}/\sum_{i=1}^{4} \beta_{k,i}$ is the normalization for learnable parameters $\beta_{k,i}$ $(i = 1, \ldots, 4)$ that adapts different importance of every regulariser.

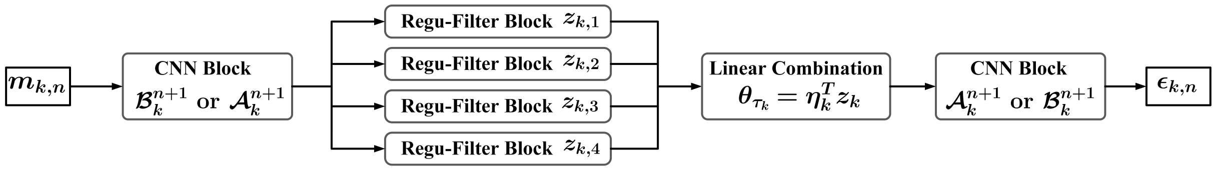

$m_{k,n}$
CNN Block
$\mathcal{B}_k^{n+1}$ or $\mathcal{A}_k^{n+1}$
Regu-Filter Block $z_{k,1}$
Regu-Filter Block $z_{k,2}$
Regu-Filter Block $z_{k,3}$
Regu-Filter Block $z_{k,4}$
Linear Combination
$\theta_{\tau_k} = \eta_k^T z_k$
CNN Block
$\mathcal{A}_k^{n+1}$ or $\mathcal{B}_k^{n+1}$
$\epsilon_{k,n}$

Fig. 2. Spectral decomposition truncated error $\epsilon_{k,n}$.

D. Nesterov-informed DGIL framework

As suggested in the previous section, a new designed framework will not only have similar geometric decomposition properties to the above classical Nesterov-based optimization, but also the new approach will be better in explainable analysis for the proposed network. In this part, we should analyze four main components of the proposed Nest-DGIL framework in details, including a linear reconstruction module $\mathcal{D}_k$, a cascade GIL module $\mathcal{P}_k$, Nesterov acceleration module $\mathcal{N}_k$ and loss function.

Linear reconstruction module $\mathcal{D}_k$: The module $\mathcal{D}_k$ corresponding to (9) is directly used to generate preliminary approximation solution $\boldsymbol{m}_k$, which can be denoted by

$$\mathcal{D}_k(\boldsymbol{h}_{k-1}, \boldsymbol{u}_k, \mu_k, \boldsymbol{y}, \boldsymbol{\Phi}) = \boldsymbol{h}_{k-1} - \mu_k \boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{u}_k - \boldsymbol{y}).$$

It is well known that the step-length μ_k should be positive and decreases with the increase of iterations smoothly. To increase network flexibility, we set the step-length μ_k to be learnable during iterations, while it is fixed in traditional model-based methods. There are a variety of ways to use training data to adaptively learn step-length μ_k. To facilitate the back propagation, we employ the **softplus** function $sp(x) = \ln(1+\exp(x))$ to implement the initialization of the learnable step-length μ_k [7], and the initial guess for the stage k is given by

$$\mu_k = sp\left(\alpha_1 k + c_1\right), \alpha_1 < 0, k = 1, 2, \ldots, \ell_s. \tag{18}$$

Cascade GIL module $\mathcal{P}_k$: We also notice that the module $\mathcal{D}_k$ often results in heavy artifacts. We further design a cascade GIL module $\boldsymbol{h}_k = \mathcal{P}_k(\boldsymbol{m}_k, \tau_k)$ to extract the high-frequency feature defined in different geometric characteristic domain $\mathcal{X}_i$.

The geometric texture information in each domain $\mathcal{X}_i$ can be represented by the partial derivatives of image $\boldsymbol{x}$, *e.g.*, feature extractor $\mathcal{K} = \nabla$ or $\mathcal{K} = \Delta$. As is well known that the convolution in neural network can be seen as the combination of several derivative operations. So we start by treating the operator $\mathcal{K}$ and function $\psi(\cdot)$ as an embedded convolution $Conv(\cdot)$ and a composited Rectified Linear Unit (ReLU) activation. Therefore, the operation $\mathcal{A}_k^i(\cdot)$ in each cascade of GIL module $\mathcal{P}_k$ is given by new network layer ReLU $(Conv(\cdot))$ for replacing a non-linear operator $\psi\left(\mathcal{K}(\boldsymbol{x})\right)$. Inversely, since the intensities of approximation image $\boldsymbol{h}_{k,i}$ processed by $\mathcal{K}'(\cdot)$ are non-negative, we filter out the negative values in $Conv(\boldsymbol{h}_{k,i})$ with ReLU without changing the non-negative values. Similar to $\mathcal{A}_k^i(\cdot)$, we design layer $\mathcal{B}_k^i(\cdot) = \text{ReLU}\left(Conv(\cdot)\right)$ to represent $\mathcal{K}'(\cdot)$. We believe that this is reasonable, which means that clustering these cascades directly corresponds to finding all geometric feature compensations that are dominated by the same stage. In summary, the proposed GIL module $\mathcal{P}_k$ strictly corresponds to the operator spectral geometric decomposition.

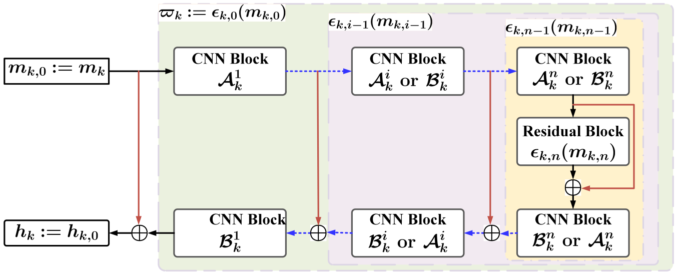

$\varpi_k := \epsilon_{k,0}(m_{k,0})$
$\epsilon_{k,i-1}(m_{k,i-1})$
$\epsilon_{k,n-1}(m_{k,n-1})$
$m_{k,0} := m_k$
CNN Block
$\mathcal{A}_k^1$
CNN Block
$\mathcal{A}_k^i$ or $\mathcal{B}_k^i$
CNN Block
$\mathcal{A}_k^n$ or $\mathcal{B}_k^n$
Residual Block
$\epsilon_{k,n}(m_{k,n})$
$h_k := h_{k,0}$
CNN Block
$\mathcal{B}_k^1$
CNN Block
$\mathcal{B}_k^i$ or $\mathcal{A}_k^i$
CNN Block
$\mathcal{B}_k^n$ or $\mathcal{A}_k^n$

Fig. 3. Cascade GIL module $h_k = \mathcal{P}_k(m_k, \theta_k) = m_k + \varpi_k$ and geometric incremental component ϖ_k, where the module shares same parameter $\mathcal{A}_k^i$ and $\mathcal{B}_k^i$ in each $\varpi_{k,i}$.

To proceed, we first construct the final incremental error by (*see* Fig. 2)

$$\boldsymbol{\epsilon}_{k,n}(\boldsymbol{m}_{k,n}) = \begin{cases} \mathcal{B}_k^{n+1}(\theta_{\tau_k}(\mathcal{A}_k^{n+1}(\boldsymbol{m}_{k,n}))), \\ \qquad \text{if } n \text{ is even number;} \\ \mathcal{A}_k^{n+1}(\theta_{\tau_k}(\mathcal{B}_k^{n+1}(\boldsymbol{m}_{k,n}))), \\ \qquad \text{if } n \text{ is odd number.} \end{cases}$$

Consequently, we are now going to show an incremental learning extrapolation relationship from (14) to expand the cascade architecture by

$$\begin{aligned} &\boldsymbol{m}_{k,i} + \boldsymbol{\epsilon}_{k,i}(\boldsymbol{m}_{k,i}) \\ &= \begin{cases} \boldsymbol{m}_{k,i} + \mathcal{B}_k^{i+1}\left(\boldsymbol{m}_{k,i+1} + \boldsymbol{\epsilon}_{k,i+1}(\boldsymbol{m}_{k,i+1})\right), \\ \qquad \text{if } i \text{ is even number;} \\ \boldsymbol{m}_{k,i} + \mathcal{A}_k^{i+1}\left(\boldsymbol{m}_{k,i+1} + \boldsymbol{\epsilon}_{k,i+1}(\boldsymbol{m}_{k,i+1})\right), \\ \qquad \text{if } i \text{ is odd number,} \end{cases} \end{aligned}$$

where $i = n-1, \cdots, 1, 0$.

The total incremental learning estimation in the stage k is performed in Fig.3 by

$$\begin{aligned} \boldsymbol{h}_k =& \mathcal{P}_k(\boldsymbol{m}_k, \tau_k) = \boldsymbol{m}_k + \varpi_k \\ =& \boldsymbol{m}_k + (\sum_{i=1}^{n} \varpi_{k,i} + \varpi_{k,n+1}) \\ =& \boldsymbol{m}_k + \sum_{i=1}^{n} \underbrace{\underline{\mathcal{B}_k^1 \mathcal{A}_k^2}[\underline{\mathcal{B}_k^3 \cdots\cdots \mathcal{A}_k^3}[\underline{\mathcal{B}_k^2 \mathcal{A}_k^1}}_{i}(\boldsymbol{m}_k)]] \\ &+ \begin{cases} \underbrace{\underline{\mathcal{B}_k^1}\ \underline{\mathcal{A}_k^2}[\cdots \underline{\mathcal{A}_k^n}[}_{n} \underline{\mathcal{B}_k^{n+1} \boldsymbol{\theta}_{\tau_k}\left(\mathcal{A}_k^{n+1}\right.}(\boldsymbol{m}_{k,n})]))], \\ \qquad \text{if } n \text{ is even number;} \\ \underbrace{\underline{\mathcal{B}_k^1}\ \underline{\mathcal{A}_k^2}[\cdots \underline{\mathcal{B}_k^n}[}_{n} \underline{\mathcal{A}_k^{n+1} \boldsymbol{\theta}_{\tau_k}\left(\mathcal{B}_k^{n+1}\right.}(\boldsymbol{m}_{k,n})]))], \\ \qquad \text{if } n \text{ is odd number.} \end{cases} \end{aligned} \tag{19}$$

where $\varpi_{k,i}$-block and $\varpi_{k,n+1}$-block strictly corresponding to the operator spectral geometric decomposition (II-C) are denoted in Fig. 4 and Fig. 5.

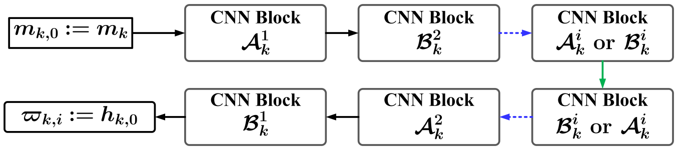

$m_{k,0} := m_k$
CNN Block $\mathcal{A}_k^1$
CNN Block $\mathcal{B}_k^2$
CNN Block $\mathcal{A}_k^i$ or $\mathcal{B}_k^i$
$\varpi_{k,i} := h_{k,0}$
CNN Block $\mathcal{B}_k^1$
CNN Block $\mathcal{A}_k^2$
CNN Block $\mathcal{B}_k^i$ or $\mathcal{A}_k^i$

Fig. 4. Spectral decomposition component $\varpi_{k,i}$ in the geometric subspace $\mathcal{X}_i$ $(1 \le i \le n)$.

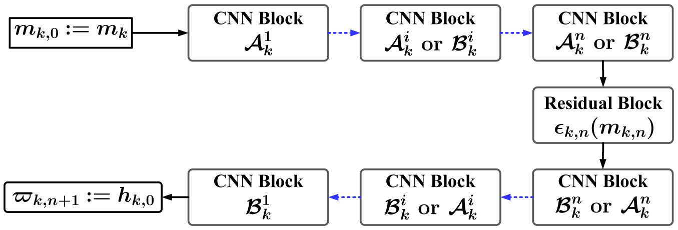

$m_{k,0} := m_k$
CNN Block $\mathcal{A}_k^1$
CNN Block $\mathcal{A}_k^i$ or $\mathcal{B}_k^i$
CNN Block $\mathcal{A}_k^n$ or $\mathcal{B}_k^n$
Residual Block $\epsilon_{k,n}(m_{k,n})$
$\varpi_{k,n+1} := h_{k,0}$
CNN Block $\mathcal{B}_k^1$
CNN Block $\mathcal{B}_k^i$ or $\mathcal{A}_k^i$
CNN Block $\mathcal{B}_k^n$ or $\mathcal{A}_k^n$

Fig. 5. Spectral decomposition truncated component $\varpi_{k,n+1}$ in the geometric subspace $\mathcal{X}_{n+1}$.

Now the task of finding an optimal Nest-DGIL framework such that its prediction $\boldsymbol{x}_{\ell_s}$ is similar to ground-truth $\boldsymbol{x}$ is to learn the network parameters and physical hyper-parameters. To proceed, we give the concrete convolution sets of $\mathcal{A} = \{\mathcal{A}_k^i\}_{k,i=1}^{\ell_s,n+1}$ and $\mathcal{B} = \{\mathcal{B}_k^i\}_{k,i=1}^{\ell_s,n+1}$ in proposed cascade GIL module. Each $\mathcal{A}_k^i$ $(2 \le i \le n+1)$ corresponds to N_f filters where each filter is of size $3 \times 3 \times N_f$, while $\mathcal{A}_k^1$ corresponds to N_f filters where each filter is of size 3×3. The operation $\mathcal{B}_k^1$ corresponds to 1 filter with size $3 \times 3 \times N_f$, but the other $\mathcal{B}_k^i$ correspond to N_f filters (each filter is of size $3 \times 3 \times N_f$). All the CNN blocks ($\mathcal{A}_k^i$ and $\mathcal{B}_k^i$) contain ReLU activation except the nearest one before and after the residual block. The last CNN block without ReLU before the residual block can make full use of the previous input information for truncation optimization with multi-regularisers. The first CNN block without ReLU after the residual block can comprehensively utilize the preliminary linear reconstruction and the geometric incremental information learned from the proposed cascade incremental learning module.

Based on the prior knowledge, the threshold value τ_k should be positive, and decrease with the increase of iterations smoothly. Similar to module $\mathcal{D}_k$, the initial guess of learnable τ_k is given as follows

$$\tau_k = sp\left(\alpha_2 k + c_2\right), \alpha_2 < 0, k = 1, 2, \ldots, \ell_s. \tag{20}$$

Unfortunately, we notice that the batch normalization (BN) is not adopted in many reconstruction network because some recent papers showed that BN layer is more likely to yield undesirable results when a network becomes deeper and more complex [57], [58].

Nesterov acceleration module $\mathcal{N}_k$: The Nesterov acceleration module

$$\mathcal{N}_k(\boldsymbol{x}_{k-1}, \boldsymbol{h}_{k-1}, \gamma_k) = (1-\gamma_k)\,\boldsymbol{x}_{k-1} + \gamma_k \boldsymbol{h}_{k-1}$$

uses previous reconstruction result to correct the current descending direction and realizes the acceleration of reconstruction convergence corresponds to (8) and (11). Based on the prior knowledge, the relaxation parameters γ_k and $1-\gamma_k$ should be positive, and γ_k increases with the increase of iterations smoothly. We design the initial guess of learnable step-length γ_k as follows

$$\begin{aligned} \gamma_k' &= \mathrm{sp}\left(\alpha_3 k + c_3\right), \alpha_3 > 0, \\ \gamma_k &= \sigma(\gamma_k') = \frac{e^{\gamma_k'}}{e^{1-\gamma_k'} + e^{\gamma_k'}}, k = 1, 2, \ldots, \ell_s, \end{aligned} \tag{21}$$

where the sigmoid function $\sigma(\cdot)$ ensures that γ_k and $1-\gamma_k$ are positive.

Algorithm 1 The Proposed Nest-DGIL Framework.

Input: The measurement matrix $\boldsymbol{\Phi}$ and its transpose $\boldsymbol{\Phi}^T$, the geometric incremental domain number $n+1$, the stage number ℓ_s, the training dataset $\mathcal{T} = \left\{\left(\boldsymbol{y}^i, \boldsymbol{x}^i\right) | i = 1, \cdots, N_b\right\}$.
Initialize: $\boldsymbol{x}_0 = \boldsymbol{\Phi}^T\boldsymbol{y}$, $\boldsymbol{h}_0 = \boldsymbol{x}_0$, the learnable parameters $\Theta = \left\{\mu_k, \tau_k, \gamma_k, \mathcal{A}_k^i(\cdot), \mathcal{B}_k^i(\cdot), \beta_{k,j}\right\}_{k,i,j=1}^{\ell_s,n+1,4}$.
Inference:

```
for $k = 1, 2, \cdots, \ell_s$ do
    $\boldsymbol{u}_k = (1-\gamma_k)\,\boldsymbol{x}_{k-1} + \gamma_k \boldsymbol{h}_{k-1}$; // (8)
    $\boldsymbol{m}_k = \boldsymbol{h}_{k-1} - \mu_k \boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{u}_k - \boldsymbol{y})$; // (9)
    $\boldsymbol{h}_k = \boldsymbol{m}_k + (\sum_{i=1}^n \varpi_{k,i} + \varpi_{k,n+1})$; // (19)
    $\boldsymbol{x}_k = (1-\gamma_k)\,\boldsymbol{x}_{k-1} + \gamma_k \boldsymbol{h}_k$ // (11)
end for
while CS reconstruction based on patch-by-patch do
    Weighted reconstruction of the overlapping patches;
end while
```

Training:

```
$\mathcal{L}_{\text{total}} = \frac{1}{N_b N}\sum_{i=1}^{N_b}\left\|\boldsymbol{x}_{\ell_s}^i - \boldsymbol{x}^i\right\|_1$; // (22)
```

Output: $\mathcal{H}(\mathcal{T};\Theta) = \boldsymbol{x}_{\ell_s}$.

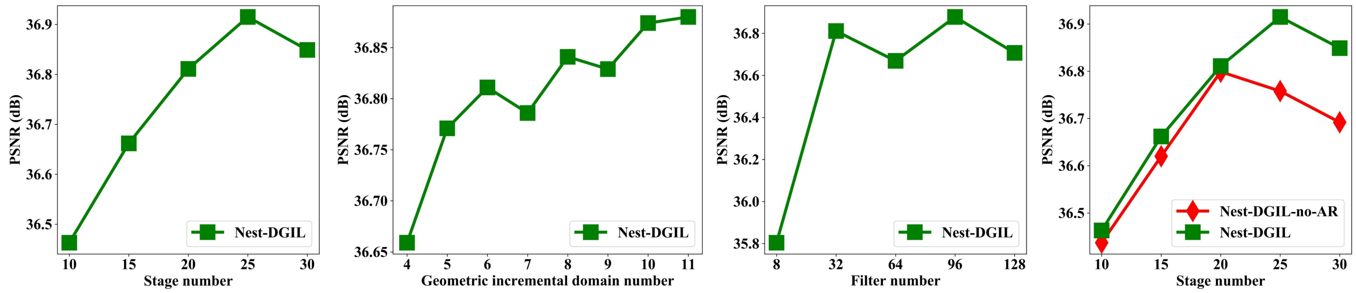
Finally, the overall Nest-DGIL algorithm is summarized in Algorithm 1, and the overall network framework is shown in Fig.1. Throughout all the experiments, we notice that Nest-DGIL and Nest-DGIL+ denote the proposed framework with a fixed random Gaussian sampling matrix and a jointly learned sampling matrix, respectively. In addition, the same weighted reconstruction post-processing technique of the overlapping patches as CSformer [28] is also performed in Nest-DGIL and Nest-DGIL+ frameworks.

Loss function: We can achieve the reconstruction prediction $\boldsymbol{x}_{\ell_s}^i$ from the proposed Nest-DGIL framework trained by using the training data-pairs $\left\{\left(\boldsymbol{y}^i, \boldsymbol{x}^i\right) | i = 1, \cdots, N_b\right\}$. The loss function is commonly employed to seek the optimal network parameters. Here we minimize the loss function between the final reconstruction output $\boldsymbol{x}_{\ell_s}$ and the ground truth $\boldsymbol{x}$, which is formulated as follows

$$\mathcal{L}_{\text{total}} = \frac{1}{N_b N}\sum_{i=1}^{N_b}\left\|\boldsymbol{x}_{\ell_s}^i - \boldsymbol{x}^i\right\|_1. \tag{22}$$

where N_b and N denote the number of data-pairs and the size of each image, respectively. A pixel-based ℓ_1-loss is used to extract perceptually high-frequency geometric details [59].

Parameters and initialization: Four modules in every stage of the proposed framework strictly correspond to the Nesterov-II scheme from (8) to (11). The learnable parameters $\Theta = \left\{\mu_k, \tau_k, \gamma_k, \mathcal{A}_k^i(\cdot), \mathcal{B}_k^i(\cdot), \beta_{k,j}\right\}_{k,i,j=1}^{\ell_s,n+1,4}$ consist of step-length μ_k, relaxation parameter γ_k, regularization parameter



(a) Different ℓ_s (b) Different $n+1$ (c) Different N_f (d) with/without AR

Fig. 6. Evaluations of the network complexity on Set11 with CS ratio 25%.

τ_k, convolutional layers $\mathcal{A}_k^i(\cdot)$ and $\mathcal{B}_k^i(\cdot)$, weight coefficient $\beta_{k,j}$. All these parameters are learned as neural network parameters by minimizing the loss (22).

Similar to conventional model-based method, the proposed method also requires an initial input $\boldsymbol{x}_0$ from a given $\boldsymbol{y}$ in Fig. 1. For natural image CS and sparse-view CT, we employ $\boldsymbol{x}_0 = \boldsymbol{\Phi}^T \boldsymbol{y}$ for initialization. We use the initialization $\boldsymbol{h}_0 = \boldsymbol{x}_0$ for both natural image CS and sparse-view CT. The convolution network is initialized with Kaiming Initialization [60]. $\{\alpha_1, \alpha_2, \alpha_3\}$ and $\{c_1, c_2, c_3\}$ are initialized with $\{-0.2, -0.5, 0.5\}$ and $\{0.1, -2, 0\}$, respectively. Throughout all the experiments, the weight coefficients $\{\beta_{k,j}\}_{k,j=1}^{\ell_s,4}$ are initialized with $\{1, 1, 1, 1\}$ at all stages. The geometric incremental domain number $n+1$ and filter number N_f are set as 6 and 32, the stage number ℓ_s is set as 20 for natural image CS and 7 for sparse-view CT, respectively. Following CSformer configurations [28], the overlap step is also set as 8 in our weighted reconstruction of the overlapping patches.

III. Experimental results and discussion

We compare the proposed method with the state-of-the-art methods on two representative CS tasks (natural image CS and sparse-view CT). Three measures including Peak Signal to Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) and Root Mean Square Error of Hounsfield Unit (RMSE(HU)) are employed to evaluate their reconstruction performances.

A. Implementation details

For natural image CS reconstruction, we utilize the 400 training images of size 180 × 180 [61] to generate the training data pairs $\left\{\left(\boldsymbol{y}^i, \boldsymbol{x}^i\right)\right\}_{i=1}^{N_b}$ with size 33 × 33 [42]. Meanwhile, we increase the training data diversity by applying the data augmentation technique [42]. We employ widely used benchmark datasets Set11[1], BSD68 [62] and Urban100 [63] for image CS reconstruction testing. The CS ratios $\{10\%, 25\%, 30\%, 40\%, 50\%\}$ are evaluated in our natural image experiments. We apply $\boldsymbol{y}^i = \boldsymbol{\Phi}\boldsymbol{x}^i$ to yield the CS measurements like [42], where $\boldsymbol{x}^i$ is the vectorized image block and $\boldsymbol{\Phi} \in \mathbb{R}^{M \times N}$ is a random Gaussian measurement matrix orthogonalizing its rows ($\boldsymbol{\Phi}\boldsymbol{\Phi}^T = I$, I is the identity matrix) in every CS ratio. When $\boldsymbol{\Phi}$ is jointly learned, we utilize a convolution layer to mimic the CS sampling process $\boldsymbol{y}^i = \boldsymbol{\Phi}\boldsymbol{x}^i$ and another convolution layer with N filters from $\boldsymbol{\Phi}^T \in \mathbb{R}^{N \times M}$ to obtain initialization $\boldsymbol{\Phi}^T\boldsymbol{y}$ like [64].

Due to not only the frequency of requests by Mayo and the AAPM for the complete 2016 Grand Challenge dataset but also the lack of such data, therefore investigators at the Mayo Clinic builds a library of CT patient projection data named ”Low Dose CT Image and Projection Data” [65]. This unique data library has facilitated the development and validation of new CT reconstruction algorithms. For sparse-view CT, we choose the first ten-patients dataset from the non-contrast chest CT scans to evaluate the reconstruction performances of compared methods, which contains 3324 full-dose CT images of 1.5 mm thickness. Among them, the first eight-patients subset is used for training, and the ninth patient’s data for validation whereas the last one for testing. In detail, there are 2621 slices of 512 × 512 images for training, and 340 slices of 512 × 512 images for validation. 363 slices of size 512 × 512 are used as LDCT-Data testing dataset. For a sparse-view CT generalizability testing, we employ a patient data from ”Fused Radiology-Pathology Lung Dataset” [66] which has 321 slices of size 512 × 512 with 1 mm thickness as FRPLung-Data testing dataset. The training images are augmented by carrying out horizontal and vertical flipping. Sinograms for this dataset are 729 pixels by 720 views, and original artifact-free images are reconstructed by iradon transform in TorchRadon [67] using all 720 views. The projection observations are downsampled to 60, 90, 120 and 180 views to simulate a few view geometries, respectively.

We use Pytorch to implement the proposed Nest-DGIL framework with a batch size of 32 for natural image CS and a batch size of 1 for sparse-view CT separately. We use Adam optimization [68] with a learning rate of 0.0001 to train network. In natural image reconstruction, 200 epochs are used whereas in CT reconstruction 50 epochs are used. All experiments are performed on a workstation with Intel Xeon CPU E5-2630 and Nvidia Tesla V100 GPU.

B. Intra-method evaluation

In this part, through several groups of evaluations, we aim to investigate the role of different network components in our Nest-DGIL on natural image CS reconstruction performance, including the stage number ℓ_s, geometric incremental domain number $n+1$, filter number N_f and adaptive remainder

[1] http://dsp.rice.edu/software/DAMP-toolbox.

optimization (AR) (II-C), as well as ablation study about the proposed cascade incremental learning module (CI) (19), AR and deep Nesterov-II scheme (N-II) ((8) to (11)), and different shared settings of Nest-DGIL.

Test of stage number ℓ_s: To better explore the reconstruction performance of the proposed method, we analyze reconstruction results by varying the stage number ℓ_s from 10 to 30 at 5 intervals. Using the CNN architectures with different stage number ℓ_s on Set11 with CS ratio 25%, the average PSNR curves of the reconstructed results are shown in Fig.6(a). We can observe that the average PSNR curves increase as ℓ_s increases and become stable after $\ell_s \geq 20$. Based on this observation, the 20-stage configuration is a preferable setting to balance the reconstruction performance and computational costs. We fix the stage number $\ell_s = 20$ for natural image CS throughout all the experiments.

Test of geometric incremental domain number $n+1$: To explore the relationship between different geometric incremental domains and reconstruction performance, we tune the geometric incremental domain number $n+1$ from 4 to 11. Fig.6(b) shows that the performances gradually improve when the geometric incremental spectral subspace number increases and fluctuates in a small range after 6. Considering the tradeoff between network complexity and reconstruction performance, we set geometric incremental domain number $n+1 = 6$ in all configurations.

Test of filter number N_f: To explore the degree of over and under parameterization influences the reconstruction performance, we compare them in various number of filters. Fig.6(c) shows the average PSNR performance on Set11 for different filter numbers with CS ratio 25%. When the filter number $N_f = 8$, the network works well even if it is under-parameterization and does not fully use the architecture's potential. We can find that even if the over-parameterization is severe, our method can still maintain excellent reconstruction performance without being troubled by over-fitting. Considering the tradeoff between network complexity and reconstruction performance, we set the default filter number as 32 in all configurations.

Test of with/without AR: To better explore the importance of AR, we analyze reconstruction results by varying the stage number ℓ_s from 10 to 30 at 5 intervals with/without AR. The average PSNR curves of the reconstructed results are shown in Fig.6(d). We can find that the proposed AR can indeed enhance the reconstruction performance at different stages and works better on larger stages.

TABLE I
RECONSTRUCTION PERFORMANCE COMPARISONS WITH DIFFERENT COMBINATIONS OF CI, AR AND N-II ON SET11 WITH CS RATIO 25%.

Variant	(a)	(b)	(c)	(d)	(e)	(f)	(g)
CI	-	+	-	-	+	+	+
AR	-	-	+	-	-	+	+
N-II	-	-	-	+	+	-	+
PSNR	36.38	36.74	36.58	36.44	36.80	36.78	**36.81**

TABLE II
RECONSTRUCTION PERFORMANCE COMPARISONS (AVERAGE PSNR/AVERAGE SSIM) WITH DIFFERENT SHARED SETTINGS OF PROPOSED NEST-DGIL ON SET11 WITH CS RATIO 25%.

Variant	Shared setting	Parameters	PSNR/SSIM
(a)	Shared CI, AR	93099	36.37/0.9577
(b)	Shared CI	93175	36.47/0.9585
(c)	Shared AR	1861790	36.66/0.9599
(d)	Unshared (default)	1861866	**36.81/0.9603**

Ablation study: Next, we conduct a group of ablation studies to evaluate the effectiveness of CI, AR and N-II. The reconstruction performance comparisons are shown in Table I. From variants (a) and (b), it is clear that the proposed CI can compensate for the missing texture information from different geometric domains and improve reconstruction performance effectively. Comparisons between variants (d) and (e) and between variants (c) and (f) demonstrate the effectiveness of the proposed CI further. Comparison between variants (a) and (c) shows that the proposed AR can achieve better reconstruction performance. Actually, AR is designed to approximate the remainder of operator spectral decomposition and it is not surprising that its role is smaller than CI's. Compared with variant (a), variant (d) with the N-II scheme can improve reconstruction performance and provide a theoretical guarantee of faster convergence $(\mathcal{O}(\frac{1}{k^2}))$ than ISTA $(\mathcal{O}(\frac{1}{k}))$. The proposed CI, AR and N-II in our Nest-DGIL variant (g) can promote each other, and achieves great and stable reconstruction performance improvement than baseline (a). These comparisons show that the CI is the most critical component.

Module sharing configurations: To demonstrate the flexibility of the proposed framework that does not have to be the same network parameter configurations in different stages, we conduct several variants of Nest-DGIL with different shared settings among stages. Table II lists the average PSNR and average SSIM comparisons for natural image CS with different shared settings of proposed Nest-DGIL on Set11 with CS ratio 25%. Note that the best reconstruction performance is achieved when using the default unshared variant (d), which is the most flexible with largest number of parameters. The variant (a) that shares both CI and AR in all stages is least flexible with smallest number of parameters and achieves the worst performance. If only CI or AR is shared, the variants (b) and (c) increase average 0.10 dB and 0.29 dB over the variant (a), respectively. So we adopt the default unshared variant (d) to perform the following experiments.

From the above comparisons, we can find that our method can also work well even if the network is under-parameterization so that it does not fully use the architecture's potential. In addition, while the network is over-parameterization, our method can still maintain excellent reconstruction performance without obvious over-fitting and degradation. Actually, we attribute the superiority of our method to two factors. Firstly, it has a cascade GIL module with geometric spectral decomposition and multi-regularisers truncation, which can obtain more texture compensation from different geometric decomposition domains. Secondly, the deep Nesterov-II scheme avoids the risk of intermediate reconstruction results falling outside the solution domains.

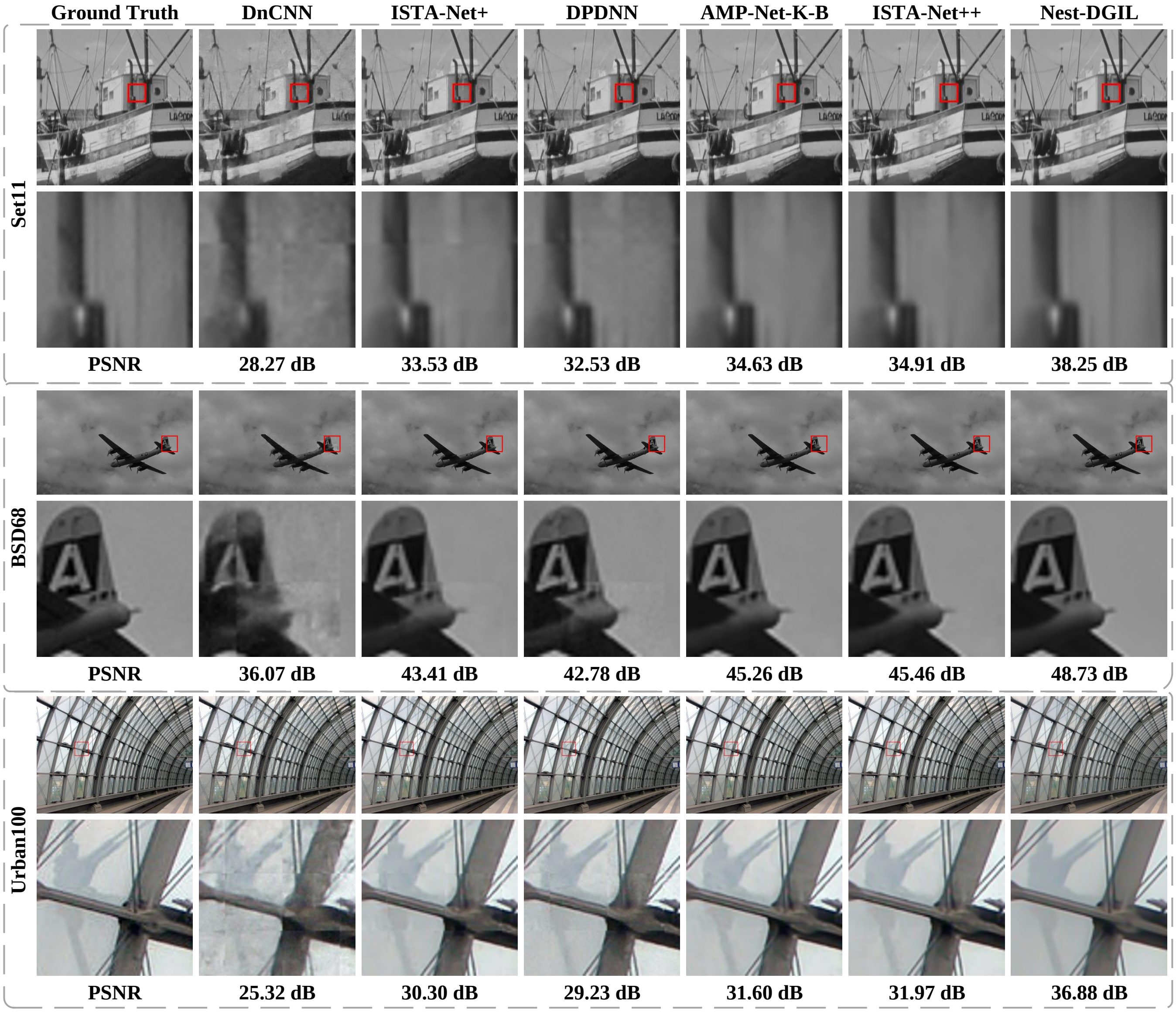


Fig. 7. Reconstruction comparisons on Set11, BSD68 and Urban100 with CS ratio 25% by different methods with fixed random Gaussian matrix.

C. Comparisons with other state-of-the-art methods

To demonstrate the superiority of our framework, we compare it with other widely used methods on two CS tasks (natural image CS and sparse-view CT).

Natural image CS: e compare our Nest-DGIL and Nest-DGIL+ with several recent representative CS reconstruction methods, including the deep learning method (DnCNN [24], CSformer [28]) and state-of-the-art deep unfolding methods (ISTA-Net [30], ISTA-Net+ [30], DPDNN [69], FISTA-Net [7], AMP-Net-K [31], AMP-Net-K-B [31], ISTA-Net++ [42], COAST [70], OPINE-Net [64], AMP-Net-K-M [31], AMP-Net-K-BM [31]) to demonstrate our method's excellent performance on natural image CS reconstruction. Following [30], the stage number of FISTA-Net is configured as 9. The results of CSformer [28] are obtained by their public pre-trained model. All of the other compared methods are trained by the same training dataset [61] and the same methods with the corresponding works.

The average PSNR/SSIM values of natural image CS reconstruction corresponding to five CS ratios with fixed random Gaussian and jointly learned sampling matrix are shown in Table III. For AMP-Net [31], the variant AMP-Net-K-B with deblocking modules can effectively improve reconstruction results than the variant AMP-Net-K which only has denoising modules. Thanks to the cross-block strategy in sampling process, ISTA-Net++ achieves state-of-the-art results. Our method almost outperforms other compared methods in all CS ratios obviously, and achieves average 1.58 dB, 1.37 dB and 2.54 dB improvement on Set11, BSD68 and Urban100. The reconstructed pixels on an inner narrow band of the patch boundary lack context information and usually have serious block-effect. This local block-effect, although a small proportion of the overall, will greatly reduce PSNR. Performing the weighted reconstruction of the overlapping patches can reduce the block-effect part of edge reconstruction and significantly improves PSNR results [28].

TABLE III
RECONSTRUCTION PERFORMANCE COMPARISONS (AVERAGE PSNR/AVERAGE SSIM) FOR NATURAL IMAGE CS WITH DIFFERENT CS RATIOS. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED IN BOLD FONT AND UNDERLINED ONES, RESPECTIVELY.

Dataset	**Method**	**Sampling matrix**	**CS Ratio**				
			10%	25%	30%	40%	50%
Set11	DnCNN [24]	Fixed random Gaussian	23.95/0.7040	26.92/0.8042	27.50/0.8270	28.75/0.8539	29.71/0.8768
	ISTA-Net [30]		23.42/0.6805	31.30/0.9066	32.86/0.9276	35.23/0.9504	37.36/0.9660
	ISTA-Net+ [30]		26.41/0.8005	32.33/0.9206	33.73/0.9376	35.96/0.9564	38.03/0.9694
	DPDNN [69]		24.75/0.7473	31.32/0.9093	32.75/0.9289	34.56/0.9457	36.82/0.9641
	FISTA-Net [7]		25.08/0.7545	30.26/0.8882	31.76/0.9124	33.80/0.9376	35.48/0.9521
	AMP-Net-K [31]		26.22/0.7924	32.08/0.9172	33.55/0.9350	35.95/0.9566	37.97/0.9693
	AMP-Net-K-B [31]		27.78/0.8392	33.39/0.9340	34.68/0.9463	36.90/0.9623	38.72/0.9724
	ISTA-Net++ [42]		28.23/0.8480	33.62/0.9361	34.80/0.9474	36.84/0.9622	38.62/0.9722
	Nest-DGIL		**30.35/0.8875**	**36.81/0.9603**	**38.26/0.9692**	**40.83/0.9806**	**43.24/0.9876**
	COAST [70]	Jointly learned	28.48/0.8584	33.88/0.9396	35.05/0.9496	37.20/0.9645	39.01/0.9742
	OPINE-Net [64]		29.64/0.8882	34.73/0.9516	36.16/0.9609	38.45/0.9732	40.47/0.9811
	AMP-Net-K-M [31]		29.27/0.8853	34.56/0.9517	36.05/0.9620	38.35/0.9741	40.56/0.9822
	AMP-Net-K-BM [31]		29.73/0.8934	34.87/0.9545	36.35/0.9641	38.62/0.9753	40.55/0.9826
	CSformer$_{bsd400}$ [28]		29.79/0.8883	34.81/0.9527	—/—	—/—	40.73/0.9824
	CSformer$_{coco}$ [28]		**30.66**/0.9027	35.46/0.9570	—/—	—/—	41.04/0.9831
	Nest-DGIL+		30.44/**0.9038**	**36.37/0.9626**	**37.77/0.9702**	**40.25/0.9805**	**42.30/0.9866**
BSD68	DnCNN [24]	Fixed random Gaussian	24.24/0.6506	26.61/0.7602	27.14/0.7876	28.21/0.8257	29.15/0.8547
	ISTA-Net [30]		23.82/0.6298	28.95/0.8408	30.12/0.8727	32.09/0.9135	34.06/0.9426
	ISTA-Net+ [30]		25.42/0.7031	29.37/0.8513	30.51/0.8810	32.38/0.9185	34.36/0.9455
	DPDNN [69]		24.70/0.6748	29.09/0.8476	30.23/0.8770	31.80/0.9099	33.84/0.9410
	FISTA-Net [7]		24.89/0.6796	28.50/0.8286	29.59/0.8609	31.26/0.8990	32.85/0.9262
	AMP-Net-K [31]		25.37/0.7028	29.34/0.8510	30.46/0.8797	32.44/0.9190	34.36/0.9454
	AMP-Net-K-B [31]		26.12/0.7310	30.02/0.8667	31.11/0.8925	33.03/0.9272	34.92/0.9508
	ISTA-Net++ [42]		26.28/0.7377	30.15/0.8690	31.18/0.8931	33.04/0.9269	34.84/0.9499
	Nest-DGIL		**28.08/0.7868**	**32.96/0.9143**	**34.34/0.9355**	**36.81/0.9615**	**39.43/0.9784**
	COAST [70]	Jointly learned	26.40/0.7441	30.30/0.8735	31.30/0.8964	33.23/0.9301	35.09/0.9526
	OPINE-Net [64]		27.86/0.8075	31.64/0.9095	32.80/0.9290	34.85/0.9540	36.83/0.9700
	AMP-Net-K-M [31]		27.71/0.8061	31.67/0.9130	32.79/0.9306	34.89/0.9556	36.97/0.9715
	AMP-Net-K-BM [31]		27.98/0.8143	31.91/0.9170	33.04/0.9345	35.09/0.9577	37.08/0.9728
	CSformer$_{bsd400}$ [28]		28.05/0.8045	31.82/0.9106	—/—	—/—	37.14/0.9766
	CSformer$_{coco}$ [28]		28.28/0.8078	31.91/0.9102	—/—	—/—	37.16/0.9714
	Nest-DGIL+		**28.64/0.8274**	**33.11/0.9298**	**34.26/0.9451**	**36.56/0.9667**	**38.77/0.9797**
Urban100	DnCNN [24]	Fixed random Gaussian	21.81/0.6285	24.45/0.7494	25.04/0.7747	26.21/0.8110	27.25/0.8453
	ISTA-Net [30]		21.28/0.6057	27.99/0.8601	29.52/0.8905	32.03/0.9301	34.39/0.9554
	ISTA-Net+ [30]		23.60/0.7190	29.12/0.8836	30.50/0.9090	32.86/0.9404	35.07/0.9600
	DPDNN [69]		22.52/0.6722	28.14/0.8669	29.67/0.8952	31.63/0.9260	33.85/0.9523
	FISTA-Net [7]		22.51/0.6714	26.81/0.8329	28.22/0.8677	30.10/0.9055	31.81/0.9296
	AMP-Net-K [31]		23.41/0.7120	28.82/0.8773	30.33/0.9040	32.77/0.9382	34.92/0.9589
	AMP-Net-K-B [31]		24.85/0.7640	30.23/0.9027	31.63/0.9242	33.94/0.9496	35.97/0.9644
	ISTA-Net++ [42]		25.29/0.7772	30.63/0.9089	31.86/0.9269	33.93/0.9502	35.79/0.9651
	Nest-DGIL		**28.97/0.8502**	**35.47/0.9520**	**37.08/0.9647**	**39.78/0.9787**	**42.36/0.9876**
	COAST [70]	Jointly learned	25.67/0.7979	31.04/0.9161	32.25/0.9319	34.38/0.9537	36.24/0.9675
	OPINE-Net [64]		26.55/0.8343	31.47/0.9289	32.79/0.9441	34.95/0.9627	36.87/0.9743
	AMP-Net-K-M [31]		25.97/0.8245	30.96/0.9266	32.21/0.9410	34.54/0.9617	36.65/0.9744
	AMP-Net-K-BM [31]		26.37/0.8377	31.22/0.9309	32.52/0.9449	34.67/0.9631	36.53/0.9744
	CSformer$_{bsd400}$ [28]		27.92/0.8458	32.43/0.9332	—/—	—/—	37.88/0.9766
	CSformer$_{coco}$ [28]		**29.61/0.8762**	34.16/0.9470	—/—	—/—	39.46/0.9811
	Nest-DGIL+		28.81/0.8664	**34.51/0.9516**	**35.75/0.9619**	**38.07/0.9762**	**40.14/0.9846**

Fig.7 shows the visualizations of the reconstructed results for natural images with CS ratio 25% in comparing the proposed Nest-DGIL framework with the state-of-the-art methods, where the visual comparisons are consist of full images and zoom-in details. We can observe that other compared methods are obviously block-effect, and details are not recovered very well or even lost. Our method can effectively reduce block-effect (*e.g.* chimney, cloud and glass in zoom-in details) and restore much more texture details, yields much better visually pleasant results.

Sparse-view CT: To further demonstrate the superiority of our approach, we conduct a group of comparisons to evaluate the reconstruction performance on sparse-view CT. We compare our method with classical methods (FBP, FISTA-TV [52]), deep learning method (RED-CNN [71], FBPConvNet [25], DU-GAN [72], Deep Decoder [27]) and state-of-the-art deep unfolding methods (PD-Net [34], ISTA-Net [30], ISTA-Net+ [30], FISTA-Net [7], AMP-Net-K [31]). The FBP with "Ram-Lak" filter which is implemented with iradon transform in TorchRadon [67], is adopted to provide initialization for the proposed method and other compared deep models. The maximum number of iterations of FISTA-TV is set as 100 and

TABLE IV
RECONSTRUCTION PERFORMANCE COMPARISONS (AVERAGE PSNR/AVERAGE RMSE(HU)) WITH DIFFERENT DOWN-SAMPLED PROJECTIONS FOR SPARSE-VIEW CT. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED IN BOLD FONT AND UNDERLINED ONES, RESPECTIVELY.

Method	**LDCT-Data** [65]				**FRPLung-Data** [66]			
	60 views	90 views	120 views	180 views	60 views	90 views	120 views	180 views
FBP	28.69/151.3	31.98/103.7	34.67/76.06	39.03/46.08	28.88/147.9	32.10/102.0	34.70/75.57	38.84/46.94
FISTA-TV [52]	35.17/71.60	38.13/50.87	40.24/39.90	43.56/27.24	35.06/72.83	38.12/51.16	40.25/39.94	43.50/27.53
RED-CNN [71]	42.01/32.65	43.45/27.69	44.44/24.78	45.80/21.18	40.88/37.11	42.89/29.45	44.21/25.31	45.97/20.68
FBPConvNet [25]	42.26/31.74	43.50/27.57	44.44/24.78	45.97/20.81	41.16/35.88	42.98/29.16	44.22/25.27	46.05/20.48
DU-GAN [72]	38.70/47.72	39.91/41.53	41.36/35.18	43.20/28.51	37.85/52.51	39.48/43.54	41.21/35.72	43.11/28.67
Deep Decoder [27]	34.85/74.67	37.90/52.39	38.79/47.35	39.24/44.93	34.94/73.65	37.87/52.47	38.95/46.45	39.40/44.15
PD-Net [34]	41.22/35.76	42.26/31.74	42.43/31.13	43.85/26.38	39.47/43.62	40.43/39.12	40.85/37.27	41.67/33.96
ISTA-Net [30]	43.38/27.98	44.40/24.94	45.17/22.86	46.52/19.58	43.16/28.51	44.63/24.13	45.59/21.63	47.07/18.27
ISTA-Net+ [30]	43.52/27.57	44.50/24.70	45.23/22.69	46.60/19.42	43.36/27.89	44.77/23.72	45.65/21.46	47.11/18.19
FISTA-Net [7]	37.31/56.16	39.28/44.77	40.68/38.13	41.65/34.08	36.80/59.35	39.04/45.88	40.53/38.63	41.60/34.12
AMP-Net-K [31]	38.64/48.09	43.55/27.44	42.72/30.19	44.83/23.72	37.94/52.06	43.77/26.58	42.77/29.82	45.06/22.94
Nest-DGIL-WP	**43.77/26.79**	**44.67/24.21**	**45.38/22.32**	**46.72/19.17**	**43.77/26.62**	**44.93/23.31**	**45.76/21.22**	**47.16/18.06**

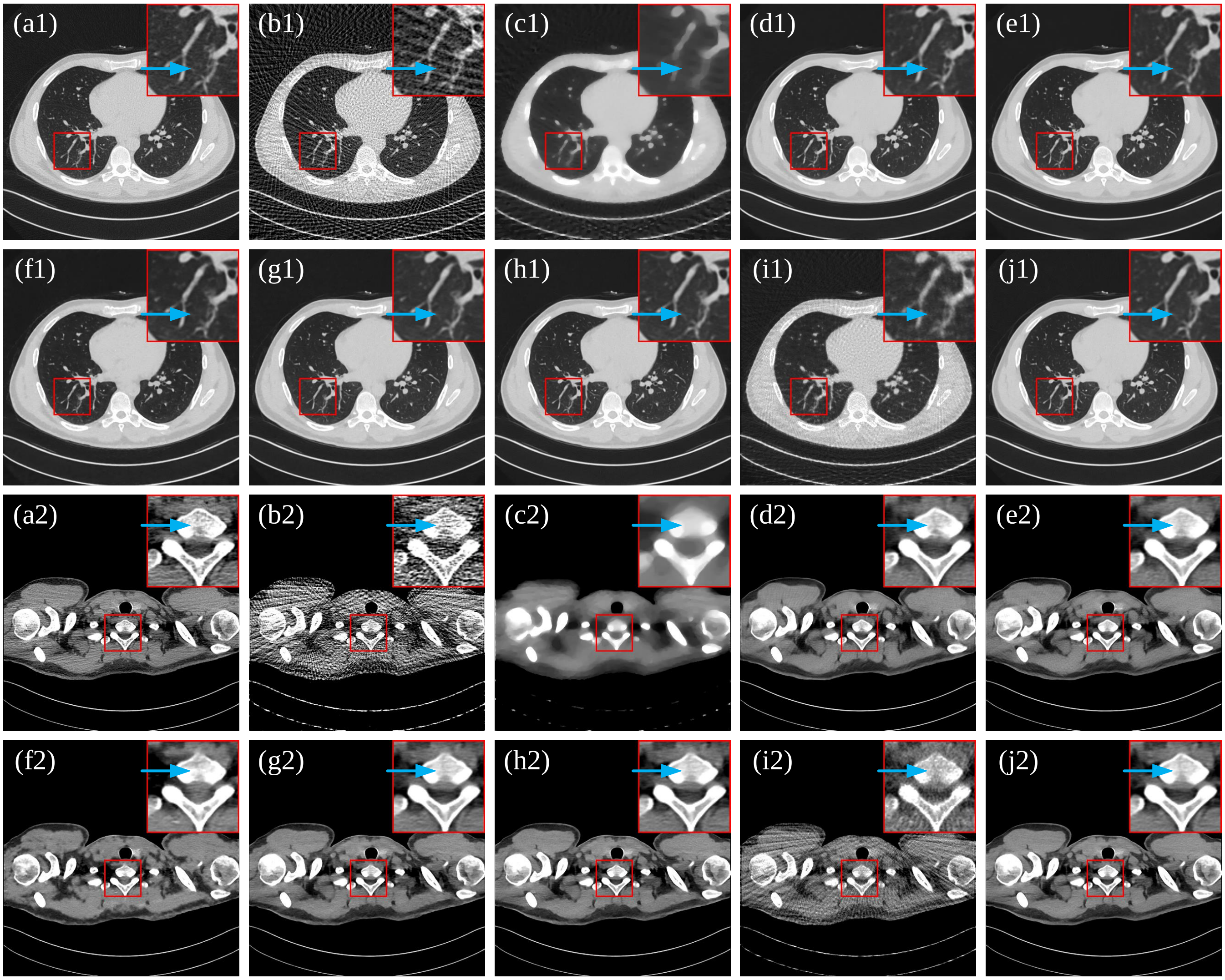


Fig. 8. The axial reconstruction results from different methods for parallel beam projection with 60 views on LDCT-Data. (a1)-(a2) Ground truth, (b1)-(b2) FBP, (c1)-(c2) FISTA-TV, (d1)-(d2) RED-CNN, (e1)-(e2) FBPConvNet, (f1)-(f2) PD-Net, (g1)-(g2) ISTA-Net, (h1)-(h2) ISTA-Net+, (i1)-(i2) FISTA-Net, (j1)-(j2) Nest-DGIL. The display windows are [-1150, 350] HU and [-160, 240] HU, respectively.

TABLE V
THE LEARNED PARAMETERS AT DIFFERENT STAGES WITH CS RATIO 25%, AND RECONSTRUCTION PERFORMANCE (AVERAGE PSNR) ON SET11.

Stage(k)	μ_k	θ_{τ_k}	γ_k	$\beta_{k,1}$	$\beta_{k,2}$	$\beta_{k,3}$	$\beta_{k,4}$	PSNR
0	-	-	-	-	-	-	-	7.64
1	2.380	1.20×10^{-1}	0.595	1.086	0.999	0.931	0.887	13.66
2	2.342	6.06×10^{-2}	0.721	1.166	0.965	0.860	0.847	21.27
3	2.304	3.01×10^{-2}	0.836	1.216	0.790	0.786	0.791	26.31
4	2.266	1.49×10^{-2}	0.917	1.255	0.800	0.765	0.760	27.05
5	2.229	7.31×10^{-3}	0.963	0.984	1.023	1.015	1.012	27.16
6	2.191	3.59×10^{-3}	0.985	1.048	0.951	0.948	0.945	28.53
7	2.154	1.76×10^{-3}	0.994	0.972	1.032	1.029	1.027	28.97
8	2.117	8.62×10^{-4}	0.998	0.859	1.141	1.141	1.141	29.33
9	2.080	4.22×10^{-4}	0.999	0.949	1.052	1.052	1.052	30.67
10	2.044	2.07×10^{-4}	1.000	0.840	1.153	1.153	1.154	31.05
11	2.007	1.01×10^{-4}	1.000	0.923	1.078	1.079	1.079	31.93
12	1.971	4.97×10^{-5}	1.000	0.926	1.077	1.075	1.076	29.89
13	1.935	2.43×10^{-5}	1.000	0.824	1.159	1.159	1.159	32.98
14	1.899	1.19×10^{-5}	1.000	0.838	1.140	1.140	1.140	34.25
15	1.864	5.84×10^{-6}	1.000	1.084	0.927	0.927	0.927	34.67
16	1.829	2.86×10^{-6}	1.000	0.918	1.083	1.083	1.083	34.80
17	1.793	1.40×10^{-6}	1.000	0.576	1.305	1.305	1.305	35.32
18	1.759	6.86×10^{-7}	1.000	0.482	1.315	1.315	1.315	35.69
19	1.724	3.36×10^{-7}	1.000	0.702	1.198	1.198	1.198	36.04
20	1.690	1.65×10^{-7}	1.000	0.809	1.123	1.123	1.123	36.81

the regularization parameter is tuned to optimal. Following [7], the stage number of PD-Net, ISTA-Net, ISTA-Net+, FISTA-Net, AMP-Net-K and Nest-DGIL are all configured as 7. The number of iterations of Deep Decoder is set as 2000 [27].

Table IV lists the average PSNR and RMSE(HU) of compared methods with different down-sampled projection views. We can observe that ISTA-Net and ISTA-Net+ consistently outperform FISTA-Net for all down-sampled projections because of unshared learnable parameters. Due to abundant learnable parameters and considerable training data, RED-CNN and FBPConvNet achieve well reconstruction performance. Due to under-parameterization and without checking the data consistency with the measurement, the Deep Decoder cannot reconstruct the CT image well. Our method outperforms comparison methods at all down-sampled projections and achieves average 0.17 dB improvement on LDCT-Data and average 0.18 dB improvement on FRPLung-Data. In addition, our method can obtain more accurate HU reconstruction and provide more outstanding help for the diagnosis of clinicians.

The axial reconstruction results from different methods for parallel beam projection with 60 views are shown in Fig.8. The highlighted blue arrow in the zoom-in images shows that the FBP results suffer from severe degradation with evident noise and heavy strip artifacts. The deep learning methods RED-CNN and FBPConvNet can remove streaking artifacts effectively, but some tiny structures might be smoothed out. Although the existing deep unfolding methods (e.g., PD-Net, ISTA-Net, ISTA-Net+ and FISTA-Net) can remove some noise and streaking artifacts, but result in incomplete preservation of details and texture information. Due to cascade geometric incremental learning and adaptive remainder optimization, our method achieves the best reconstruction performance in terms of noise-artifact removal and detail preservation (e.g., tiny blood vessels and bronchi).

TABLE VI
THE LEARNED PARAMETERS AT DIFFERENT STAGES WITH 60 VIEWS, AND RECONSTRUCTION PERFORMANCE (AVERAGE PSNR) ON LDCT-DATA.

Stage(k)	μ_k	θ_{τ_k}	γ_k	$\beta_{k,1}$	$\beta_{k,2}$	$\beta_{k,3}$	$\beta_{k,4}$	PSNR
0	-	-	-	-	-	-	-	28.69
1	1.052	0.1027	0.595	0.979	0.975	1.019	1.025	20.31
2	0.964	0.0415	0.721	0.734	1.965	0.915	0.836	26.50
3	0.880	0.0165	0.836	0.462	2.363	1.176	1.068	30.45
4	0.801	0.0065	0.917	0.482	1.809	1.310	1.253	33.25
5	0.726	0.0025	0.963	0.250	2.106	1.570	1.522	39.12
6	0.657	0.0010	0.985	0.087	2.606	1.423	1.386	41.71
7	0.592	0.0004	0.994	0.192	1.923	1.642	1.628	43.77

D. Analysis for intermediate results

As introduced in previous section, the proposed framework is an iterative architecture, so it is interesting and meaningful to evaluate the intermediate results and learned parameters in different stages. The learned parameters for natural image CS with CS ratio 25% and sparse-view CT with 60 views at different stages are shown in Table V and Table VI. The reconstruction performances become better with the increasing of iterations. We can see that the parameters μ_k and θ_{τ_k} decrease monotonically while γ_k increases with respect to k. The learned parameters μ_k, θ_{τ_k} and γ_k are consistent with the parameter configuration in traditional model-based reconstruction. It implies that our approach fully inherits the characteristics and advantages of model-based methods. For more explanation, we shows the reconstructed intermediate boats images and the corresponding residual from CS ratio 25% by Nest-DGIL at different stages in Fig.9. We can observe that the block-effect removal and details recovery are performed gradually across the stages. The trained end-to-end Nest-DGIL with meaningful model-based network architecture not only facilitates the enhancement of image intermediate results but also contributes to final better performance by supervising a smoothing reconstruction flow.

Besides, we can find that the weight coefficients $\beta_{k,1}$, $\beta_{k,2}$, $\beta_{k,3}$ and $\beta_{k,4}$ work at different stages. The results demonstrate that the proposed adaptive spectral decomposition residual part $\epsilon_{k,n}$ mainly restore the missing texture information truncated by the principal part of the spectral geometric incremental decomposition. Therefore, adaptive initialization, adaptive spectral decomposition remainder and learnable parameters setting are greatly significance for model flexibility and ensuring converging smoothly.

TABLE VII
RECONSTRUCTION PERFORMANCE COMPARISONS (AVERAGE PSNR) ON DIFFERENT TRAINING DATASETS.

Dataset	**Method**	**CS Ratio**				
		10%	25%	30%	40%	50%
BSD68	Nest-DGIL+$^{\text{Set11}}$	28.10	32.12	33.23	35.34	37.30
	Nest-DGIL+	28.64	33.11	34.26	36.56	38.77
Urban100	Nest-DGIL+$^{\text{Set11}}$	27.53	32.22	33.45	35.69	37.55
	Nest-DGIL+	28.81	34.51	35.75	38.07	40.14

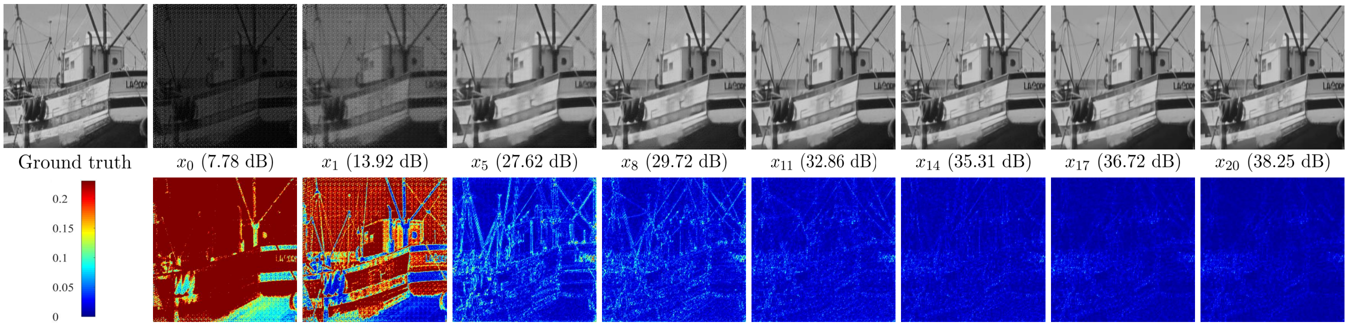


Fig. 9. Reconstructed intermediate boats images and the corresponding residual from CS ratio 25% by Nest-DGIL at different stages.

E. Generalizability evaluation on different training datasets

For natural image CS tasks, Nest-DGIL+ is trained based on 400 training images [61]. To further evaluate the generalizability of our model, we train it on Set11 with only 11 images and evaluate it on BSD68 and Urban100. The results are shown in Table VII. Due to the Set11 is really small, the potential of Nest-DGIL+$^{\text{Set11}}$ is not fully released. The reconstruction performances of Nest-DGIL+$^{\text{Set11}}$ drop 1.05 dB on BSD68 and 2.17 dB on Urban100, but still excellent. It demonstrates the excellent generalizability of our method.

IV. Conclusions

In this study, we propose a Nesterov-informed geometric incremental learning (Nest-DGIL) framework based on the second-Nesterov proximal gradient optimization. It not only has the powerful learning ability for high/low frequency image features, but also can theoretically guarantee that more geometric texture details will be reconstructed from preliminary linear reconstruction. Our Nest-DGIL network can avoid the risk of intermediate reconstruction results falling outside the geometric domain and achieve fast convergence. Such Nesterov-informed learnable architecture gives us a new perspective to design the explainable networks. Extensive experiments show that the proposed Nest-DGIL framework can greatly improve reconstruction performance upon the existing state-of-the-art methods in different applications (natural image CS and sparse-view CT). Our architecture has good generalizability for image reconstruction due to the proximal gradient-based optimization unfolding and cascade incremental learning, and can be potentially applied to other inverse problems in imaging.

Appendix A
Lemma 1 and Lemma 2

Lemma *1 ([52]):* Let $\{\boldsymbol{x}_k\}$ be the sequence generated by the proximal gradient method ((6) to (7)). Thus the proximal gradient-based reconstruction algorithm satisfies the linear rate of convergence $(\mathcal{O}(\frac{1}{k}))$, i.e.

$$\mathcal{F}(\boldsymbol{x}_k) - \mathcal{F}(\boldsymbol{x}^*) \leq \frac{\overline{\mathcal{L}}\left\|\boldsymbol{x}_0 - \boldsymbol{x}^*\right\|^2}{2k}$$

for some constant $\overline{\mathcal{L}}$ and any optimal solution $\boldsymbol{x}^*$.

Lemma *2 ([53]):* Let $\mathcal{S}$ be the Banach space of functions $\boldsymbol{x}(t)$, $a \leqslant t \leqslant b$, with the uniform norm, $\|\boldsymbol{x}\| = \max|\boldsymbol{x}(t)|$. Let $\mathcal{K}$ be an integral operator on $\mathcal{S}$, i.e.

$$(\mathcal{K}\boldsymbol{x})(s) = \int_0^1 \boldsymbol{k}(s,t)\boldsymbol{x}(t)dt, \quad a \leqslant t \leqslant b$$

with a continuous kernel $\boldsymbol{k}$. Hence one has

$$(I - \epsilon\mathcal{K})^{-1} = \sum_{n=0}^{\infty} \epsilon^n \mathcal{K}^n,$$

where $0 < \epsilon < 1/\boldsymbol{k}_{\max}$, where $\boldsymbol{k}_{\max} = \max\ \boldsymbol{k}(s,t)$, and the series converges in operator norm.

Appendix B
More explanation about the left- and right-hand operators for odd and even numbers

As we have defined $\mathcal{M}(\boldsymbol{x}) = \mathcal{K}'\left(\psi\left(\mathcal{K}(\boldsymbol{x})\right)\right)$ and $\boldsymbol{\omega}_{k,i} = \mathcal{M}^i(\boldsymbol{m}_k)$ $(i = 1, \ldots, n)$, we have

$$\boldsymbol{\omega}_{k,2} = \underbrace{\underline{\mathcal{K}'\psi(\mathcal{K}}[\underline{\mathcal{K}'\psi(\mathcal{K}}}_{2}(\boldsymbol{m}_k))]),$$

$$\boldsymbol{\omega}_{k,3} = \underbrace{\underline{\mathcal{K}'\psi(\mathcal{K}}[\underline{\mathcal{K}'\psi(\mathcal{K}}[\underline{\mathcal{K}'\psi(\mathcal{K}}}_{3}(\boldsymbol{m}_k))]]).$$

In fact, the operator $\mathcal{K}'$ is the transpose of the feature extractor $\mathcal{K}$ and can be understood as the inverse operator of $\mathcal{K}$. We can further separate operators $\mathcal{K}'(\cdot)$ and $\psi\left(\mathcal{K}(\cdot)\right)$ as follows

$$\boldsymbol{\omega}_{k,2} = \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}}_{right}\,[\underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}}_{left}(\boldsymbol{m}_k))]])],$$

$$\boldsymbol{\omega}_{k,3} = \underbrace{\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\underline{\mathcal{K}'}}_{right}\,\underbrace{\underline{\psi(\mathcal{K}}[\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}}_{left}(\boldsymbol{m}_k))]])],$$

where left- and right-hand operators are the total transform feature extractor and the inverse transform feature extractor.

When $\boldsymbol{\omega}_{k,2}$ and $\boldsymbol{\omega}_{k,3}$ are the truncation remainders, we can embed the shrinkage-thresholding operator as follows

$$\boldsymbol{\omega}_{k,2} = \underline{\mathcal{K}'}\boxed{\,\underline{\psi(\mathcal{K}}\boldsymbol{\theta}_{\tau_k}[\underline{\mathcal{K}'}\,}\underline{\psi(\mathcal{K}}(\boldsymbol{m}_k))]])],$$

$$\boldsymbol{\omega}_{k,3} = \underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}[\boxed{\,\underline{\mathcal{K}'}\boldsymbol{\theta}_{\tau_k}\underline{\psi(\mathcal{K}}\,}[\underline{\mathcal{K}'}\underline{\psi(\mathcal{K}}(\boldsymbol{m}_k))]])].$$

It is obvious that the above analysis can be easily generalized to any n. And we need to define the left- and right-hand operators for odd and even numbers to distinguish between the two designs.

Appendix C
Two concrete operator spectral geometric decomposition examples

Especially if $\phi_\ell(r) = \frac{1}{2}\|r\|^2$, we can obtain the first-order derivative $\phi'_\ell\left(\mathcal{K}(\boldsymbol{x})\right) = \frac{d\phi_\ell(\mathcal{K}(\boldsymbol{x}))}{d\boldsymbol{x}} = \mathcal{K}'\left(\mathcal{K}(\boldsymbol{x})\right)$ of the regularization term $\mathcal{R}(\boldsymbol{x})$ on $\mathbb{R}^N$, hence one has

$$\mathcal{M}^i(\boldsymbol{m}_k) = \underbrace{\underline{\mathcal{K}'\mathcal{K}}[\cdots\underline{\mathcal{K}'\mathcal{K}}[\underline{\mathcal{K}'\mathcal{K}}}_{i}(\boldsymbol{m}_k)]], \tag{23}$$

for $s = 1$ and $i = 1, \ldots, n$.

Alternatively if $\phi_\ell(r) = \|r\|$, thus the first-order derivative of $\phi_\ell\left(\mathcal{K}(\boldsymbol{x})\right)$ in the regularization term $\mathcal{R}(\boldsymbol{x})$ defined on $\mathbb{R}^N$ is derived as follow

$$\phi'_\ell\left(\mathcal{K}(\boldsymbol{x})\right) = \frac{d\phi_\ell\left(\mathcal{K}(\boldsymbol{x})\right)}{d\boldsymbol{x}} = \mathcal{K}'\left(\frac{\mathcal{K}(\boldsymbol{x})}{|\mathcal{K}(\boldsymbol{x})|}\right),$$

we define $\mathcal{M}^i(\boldsymbol{m}_k)$ for $s = 1$ by

$$\mathcal{M}^i(\boldsymbol{m}_k) = \underbrace{\underline{\mathcal{K}'\frac{\mathcal{K}}{\|\mathcal{K}(\boldsymbol{r}_{k,i})\|}}[\cdots\underline{\mathcal{K}'\frac{\mathcal{K}}{\|\mathcal{K}(\boldsymbol{r}_{k,1})\|}}[\underline{\mathcal{K}'\frac{\mathcal{K}}{\|\mathcal{K}(\boldsymbol{m}_k)\|}}}_{i}(\boldsymbol{m}_k)]],$$

where $i = 1, \ldots, n$, and the normalizing weight technique is adopted to prevent the scale of features from becoming too large. Inspired by it, we can restrict the feature inputs by the above architecture and realize the channel normalization like the BN layer.

References

[1] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[2] E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, Mar. 2006.

[3] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[4] D. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[5] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, Mar. 2008.

[6] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, vol. 58, no. 6, pp. 1182–1195, Dec. 2007.

[7] J. Xiang, Y. Dong, and Y. Yang, "FISTA-Net: Learning a fast iterative shrinkage thresholding network for inverse problems in imaging," *IEEE Transactions on Medical Imaging*, vol. 40, no. 5, pp. 1329–1339, May 2021.

[8] Y. Hitomi, J. Gu, M. Gupta, T. Mitsunaga, and S. K. Nayar, "Video from a single coded exposure photograph using a learned over-complete dictionary," in *International Conference on Computer Vision (ICCV)*, Nov. 2011, pp. 287–294.

[9] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, "Model-based compressive sensing," *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1982–2001, Apr. 2010.

[10] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, Aug. 2004.

[11] W. Dong, G. Shi, X. Li, Y. Ma, and F. Huang, "Compressive sensing via nonlocal low-rank regularization," *IEEE Transactions on Image Processing*, vol. 23, no. 8, pp. 3618–3632, Aug. 2014.

[12] L. He and L. Carin, "Exploiting structure in wavelet-based bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 57, no. 9, pp. 3488–3497, Sep. 2009.

[13] X. Qu, D. Guo, B. Ning, Y. Hou, Y. Lin, S. Cai, and Z. Chen, "Undersampled MRI reconstruction with patch-based directional wavelets," *Magnetic Resonance Imaging*, vol. 30, no. 7, pp. 964–977, Sep. 2012.

[14] S. Ravishankar and Y. Bresler, "MR image reconstruction from highly undersampled k-space data by dictionary learning," *IEEE Transactions on Medical Imaging*, vol. 30, no. 5, pp. 1028–1041, May 2011.

[15] M. Elad, "Why simple shrinkage is still relevant for redundant representations?" *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5559–5569, Dec. 2006.

[16] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, Jan. 2009.

[17] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2010.

[18] K. Kulkarni, S. Lohit, P. Turaga, R. Kerviche, and A. Ashok, "ReconNet: Non-iterative reconstruction of images from compressively sensed measurements," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 449–458.

[19] X. Xie, C. Wang, J. Du, and G. Shi, "Full image recover for block-based compressive sensing," in *IEEE International Conference on Multimedia and Expo (ICME)*, Jul. 2018.

[20] M. Mardani, E. Gong, J. Y. Cheng, S. S. Vasanawala, G. Zaharchuk, L. Xing, and J. M. Pauly, "Deep generative adversarial neural networks for compressive sensing MRI," *IEEE Transactions on Medical Imaging*, vol. 38, no. 1, pp. 167–179, Jan. 2019.

[21] A. Mousavi and R. G. Baraniuk, "Learning to invert: Signal recovery via deep convolutional networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017, pp. 2272–2276.

[22] G. Yang, S. Yu, H. Dong, G. Slabaugh, P. L. Dragotti, X. Ye, F. Liu, S. Arridge, J. Keegan, Y. Guo, and D. Firmin, "DAGAN: Deep de-aliasing generative adversarial networks for fast compressed sensing MRI reconstruction," *IEEE Transactions on Medical Imaging*, vol. 37, no. 6, pp. 1310–1321, Jun. 2018.

[23] Y. Zhong, C. Zhang, F. Ren, H. Kuang, and P. Tang, "Scalable image compressed sensing with generator networks," *IEEE Transactions on Computational Imaging*, vol. 8, pp. 1025–1037, 2022.

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

[25] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, Sep. 2017.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778.

[27] R. Heckel and P. Hand, "Deep decoder: Concise image representations from untrained non-convolutional networks," in *International Conference on Learning Representations*, 2019.

[28] D. Ye, Z. Ni, H. Wang, J. Zhang, S. Wang, and S. Kwong, "CSformer: Bridging convolution and transformer for compressive sensing," *IEEE Transactions on Image Processing*, vol. 32, pp. 2827–2842, 2023.

[29] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," *International Conference on Machine Learning (ICML)*, pp. 399–406, 2010.

[30] J. Zhang and B. Ghanem, "ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018, pp. 1828–1837.

[31] Z. Zhang, Y. Liu, J. Liu, F. Wen, and C. Zhu, "AMP-Net: Denoising-based deep unfolding for compressive image sensing," *IEEE Transactions on Image Processing*, vol. 30, pp. 1487–1500, 2021.

[32] Y. Yang, J. Sun, H. Li, and Z. Xu, "Deep ADMM-Net for compressive sensing mri," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[33] V. Lempitsky, A. Vedaldi, and D. Ulyanov, "Deep image prior," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018.

[34] J. Adler and O. Oktem, "Learned primal-dual reconstruction," *IEEE Transactions on Medical Imaging*, vol. 37, no. 6, pp. 1322–1332, Jun. 2018.

[35] H. K. Aggarwal, M. P. Mani, and M. Jacob, "MoDL: Model-based deep learning architecture for inverse problems," *IEEE Transactions on Medical Imaging*, vol. 38, no. 2, pp. 394–405, Feb. 2019.

[36] J. Duan, J. Schlemper, C. Qin, C. Ouyang, W. Bai, C. Biffi, G. Bello, B. Statton, D. P. O'Regan, and D. Rueckert, "VS-Net: Variable splitting network for accelerated parallel MRI reconstruction," in *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, Oct. 2019, pp. 713–722.

[37] M. Borgerding, P. Schniter, and S. Rangan, "AMP-inspired deep networks for sparse linear inverse problems," *IEEE Transactions on Signal Processing*, vol. 65, no. 16, pp. 4293–4308, Aug. 2017.

[38] M. Christopher, M. Ali, and B. Richard, "Learned D-AMP: A principled cnn-based compressive image recovery algorithm," in *Advances in Neural Information Processing Systems (NeurIPS)*, Apr. 2017.

[39] Y. Yang, J. Sun, H. Li, and Z. Xu, "ADMM-CSNet: A deep learning approach for image compressive sensing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 3, pp. 521–538, Mar. 2020.

[40] Y. Liu, Q. Liu, M. Zhang, Q. Yang, S. Wang, and D. Liang, "Ifr-net: Iterative feature refinement network for compressed sensing mri," *IEEE Transactions on Computational Imaging*, vol. 6, pp. 434–446, 2020.

[41] K. Zhang, L. V. Gool, and R. Timofte, "Deep unfolding network for image super-resolution," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2020.

[42] D. You, J. Xie, and J. Zhang, "ISTA-Net++: Flexible deep unfolding network for compressive sensing," in *IEEE International Conference on Multimedia and Expo (ICME)*, Jul. 2021.

[43] X. Chen, J. Liu, Z. Wang, and W. Yin, "Hyperparameter tuning is all you need for lista," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[44] X. Fan, Y. Yang, and J. Zhang, "Deep geometric distillation network for compressive sensing MRI," in *IEEE EMBS International Conference on Biomedical and Health Informatics (BHI)*, Jul. 2021.

[45] M. B. Alver, A. Saleem, and M. Cetin, "Plug-and-play synthetic aperture radar image formation using deep priors," *IEEE Transactions on Computational Imaging*, vol. 7, pp. 43–57, 2021.

[46] R. Hou, F. Li, and G. Zhang, "Truncated residual based plug-and-play ADMM algorithm for MRI reconstruction," *IEEE Transactions on Computational Imaging*, vol. 8, pp. 96–108, 2022.

[47] J. Yang, X. Yin, M. Zhang, H. Yue, X. Cui, and H. Yue, "Learning image formation and regularization in unrolling AMP for lensless image reconstruction," *IEEE Transactions on Computational Imaging*, vol. 8, pp. 479–489, 2022.

[48] Y. Zhang, D. Hu, T. Lyu, G. Quan, J. Xiang, G. Coatrieux, S. Luo, and Y. Chen, "SPIE-DIR: Self-prior information enhanced deep iterative reconstruction using two complementary limited-angle scans for DECT," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–12, 2023.

[49] X. Fan, Y. Yang, K. Chen, J. Zhang, and K. Dong, "An interpretable MRI reconstruction network with two-grid-cycle correction and geometric prior distillation," *Biomedical Signal Processing and Control*, vol. 84, p. 104821, Jul. 2023.

[50] Y. Nesterov, "On an approach to the construction of optimal methods of minimization of smooth convex functions," *Ekonomika i Mateaticheskie Metody*, vol. 24, no. 3, pp. 509–517, 1988.

[51] H. Liu, J. Hu, Y. Li, and Z. Wen, "Optimization: Modeling, algorithm and theory (in chinese)," *Higher Education Press*, pp. 341–354, 2021.

[52] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2419–2434, Nov. 2009.

[53] P. M. Anselone and J. W. Lee, "Spectral properties of integral operators with nonnegative kernels," *Linear Algebra and its Applications*, vol. 9, pp. 67–87, 1974.

[54] H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, "Fixed-point algorithms for inverse problems in science and engineering," *Springer New York*, Mar. 2011.

[55] C. Chaux, P. L. Combettes, J.-C. Pesquet, and V. R. Wajs, "A variational formulation for frame-based inverse problems," *Inverse Problems*, vol. 23, no. 4, pp. 1495–1518, Jun. 2007.

[56] P. L. Combettes and J. C. Pesquet, "Proximal thresholding algorithm for minimization over orthonormal bases," *SIAM Journal on Optimization*, vol. 18, no. 4, pp. 1351–1376, Jan. 2008.

[57] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *European Conference on Computer Vision (ECCV)*, Jan 2019, pp. 63–79.

[58] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, "Residual dense network for image super-resolution," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2018, pp. 2472–2481.

[59] K. Jiang, Z. Wang, P. Yi, J. Jiang, J. Xiao, and Y. Yao, "Deep distillation recursive network for remote sensing imagery super-resolution," *Remote Sensing*, vol. 10, no. 11, p. 1700, Oct. 2018.

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, Dec. 2015, pp. 1026–1034.

[61] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1256–1272, Jun. 2017.

[62] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *IEEE International Conference on Computer Vision (ICCV)*, Jul. 2001, pp. 416–423.

[63] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 5197–5206.

[64] J. Zhang, C. Zhao, and W. Gao, "Optimization-inspired compact deep compressive sensing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 765–774, May 2020.

[65] T. R. Moen, B. Chen, D. R. Holmes, X. Duan, Z. Yu, L. Yu, S. Leng, J. G. Fletcher, and C. H. McCollough, "Low-dose CT image and projection dataset," *Medical Physics*, vol. 48, no. 2, pp. 902–911, Dec. 2020.

[66] M. Rusu, P. Rajiah, R. Gilkeson, M. Yang, C. Donatelli, R. Thawani, F. J. Jacono, P. Linden, and A. Madabhushi, "Co-registration of pre-operative CT with ex vivo surgically excised ground glass nodules to define spatial extent of invasive adenocarcinoma on in vivo imaging: a proof-of-concept study," *European Radiology*, vol. 27, no. 10, pp. 4209–4217, Apr. 2017.

[67] M. Ronchetti, "Torchradon: Fast differentiable routines for computed tomography," *arXiv*, Sep. 2020.

[68] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, Dec. 2014.

[69] W. Dong, P. Wang, W. Yin, G. Shi, F. Wu, and X. Lu, "Denoising prior driven deep neural network for image restoration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2305–2318, Oct. 2019.

[70] D. You, J. Zhang, J. Xie, B. Chen, and S. Ma, "COAST: COntrollable arbitrary-sampling NeTwork for compressive sensing," *IEEE Transactions on Image Processing*, vol. 30, pp. 6066–6080, 2021.

[71] H. Chen, Y. Zhang, M. K. Kalra, F. Lin, Y. Chen, P. Liao, J. Zhou, and G. Wang, "Low-dose CT with a residual encoder-decoder convolutional neural network," *IEEE Transactions on Medical Imaging*, vol. 36, no. 12, pp. 2524–2535, Dec. 2017.

[72] Z. Huang, J. Zhang, Y. Zhang, and H. Shan, "DU-GAN: Generative adversarial networks with dual-domain u-net-based discriminators for low-dose CT denoising," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.