On the Convergence of Learning-Based Iterative Methods for Nonconvex Inverse Problems

Risheng Liu, *Member, IEEE*, Shichao Cheng, Yi He, Xin Fan, *Senior Member, IEEE*, Zhouchen Lin, *Fellow, IEEE*, and Zhongxuan Luo

Abstract—Numerous tasks at the core of statistics, learning and vision areas are specific cases of ill-posed inverse problems. Recently, learning-based (e.g., deep) iterative methods have been empirically shown to be useful for these problems. Nevertheless, integrating learnable structures into iterations is still a laborious process, which can only be guided by intuitions or empirical insights. Moreover, there is a lack of rigorous analysis about the convergence behaviors of these reimplemented iterations, and thus the significance of such methods is a little bit vague. This paper moves beyond these limits and proposes Flexible Iterative Modularization Algorithm (FIMA), a generic and provable paradigm for nonconvex inverse problems. Our theoretical analysis reveals that FIMA allows us to generate globally convergent trajectories for learning-based iterative methods. Meanwhile, the devised scheduling policies on flexible modules should also be beneficial for classical numerical methods in the nonconvex scenario. Extensive experiments on real applications verify the superiority of FIMA.

Index Terms—Nonconvex optimization, learning-based iteration, convergence guarantee, image deconvolution, rain streaks removal

1 Introduction

In applications throughout statistics, machine learning and computer vision, one is often faced with the challenge of solving ill-posed inverse problems. In general, the basic inverse problem leads to a discrete linear system of the form $\mathcal{T}(\mathbf{x}) = \mathbf{y} + \mathbf{n}$, where $\mathbf{x} \in \mathbb{R}^D$ is the latent variable to be estimated, $\mathcal{T}$ denotes some given linear operations on $\mathbf{x}$, and $\mathbf{y}, \mathbf{n} \in \mathbb{R}^D$ are the observation and an unknown error term, respectively. Typically, these inverse problems can be addressed by solving the composite minimization model

$$\min_{\mathbf{x}} \Psi(\mathbf{x}) := f(\mathbf{x}; \mathcal{T}, \mathbf{y}) + g(\mathbf{x}), \tag{1}$$

where f is the fidelity that captures the loss of data fitting, and g refers to the prior that promotes desired distribution on the solution. Recent studies illustrate that many problems (e.g., image deconvolution, matrix factorization and dictionary learning) naturally require to be solved in the nonconvex scenario. This trend motivates us to investigate Nonconvex Inverse Problems (NIPs) in the form of Eq. (1) and with the practical configuration that f is continuously differentiable, g is nonsmooth, and both f and g are possibly nonconvex.

Over the past decades, a broad class of first-order methods have been developed to solve special instances of Eq. (1). For example, by integrating Nesterov's acceleration [1] into the fundamental Proximal Gradient (PG) scheme, Accelerated Proximal Gradient (APG, a.k.a. FISTA [2]) method is initially proposed to solve convex models in the form of Eq. (1) for different applications, such as image restoration [2], image deblurring [3], and sparse/low-rank learning [4], etc. While these APGs generate a sequence of objectives that may oscillate [2], [5] developed a variant of APG that guarantees the monotonicity of the sequence. For nonconvex energies in Eq. (1), Li and Lin [6] investigated a monotone APG (mAPG) and proved the convergence under the Kurdyka-Łojasiewicz (KŁ) constraint [7]. The work in [8] developed another variation of APG (APGnc) for nonconvex problems, but their original analysis only characterized the fixed-point convergence. Recently, Li et al. [9] also proved the subsequence convergence of APGnc and estimated its convergence rates by further exploiting KŁ property.

Unfortunately, even with some theoretically proved convergence properties, these classical numerical solvers may still fail in real-world scenarios. This is mainly because that the abstractly designed and fixed updating schemes do not

- *R. Liu, Y. He, and X. Fan are with the DUT-RU International School of Information Science & Engineering, Dalian University of Technology, Dalian 116081, China, and also with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian 116024, China. E-mail: {rsliu, xin.fan}@dlut.edu.cn, heyiking@outlook.com.*
- *S. Cheng is with the School of Mathematical Sciences, Dalian University of Technology, Dalian 116081, China, and also with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Dalian 116024, China. E-mail: shichao.cheng@outlook.com.*
- *Z. Lin is with the Key Laboratory of Machine Perception (Ministry of Education), School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China. E-mail: zlin@pku.edu.cn.*
- *Z. Luo is with the DUT-RU International School of Information Science & Engineering, School of Mathematical Sciences, Dalian University of Technology, Dalian 116081, China, the Key Laboratory for Ubiquitous Networ and Service Software of Liaoning Province, Dalian 116024, China, and also with the Institute of Artificial Intelligence, Guilin University of Electronic Technology, Guilin 541004, China. E-mail: zxluo@dlut.edu.cn.*

Manuscript received 13 Aug. 2018; revised 4 Jan. 2019; accepted 23 May 2019. Date of publication 3 June 2019; date of current version 3 Nov. 2020. (Corresponding author: Risheng Liu.)
Recommended for acceptance by P. Ravikumar.
Digital Object Identifier no. 10.1109/TPAMI.2019.2920591

exploit the particular structure of the problem at hand nor the input data distribution [10].

In recent years, various learning-based strategies [11], [12], [13], [14], [15] have been proposed to address practical inverse problems in the form of Eq. (1). These methods first introduced hyperparameters into the classical numerical solvers and then performed discriminative learning on collected training data to obtain some data-specific (but possibly inconsistent) iteration schemes. Inspired by the success of deep learning in different application fields, some preliminary studies considered the handcrafted network architectures as the implicit priors (a.k.a. deep priors) for inverse problems. Following this perspective, various deep priors are designed and nested into numerical iterations [16], [17], [18]. Alternately, the works in [19] and [20] addressed the iteration learning issues from the perspectives of deep reinforcement and recurrent learning, respectively.

Nevertheless, existing hyperparameters learning approaches can only build iterations based on the specific energy forms (e.g., ℓ_1-penalty and MRFs), so that they are inapplicable for more generic inverse problems. Meanwhile, due to severe inconstancy of parameters during iterations, rigorous analysis on the resulted trajectories is also missing. Deep iterative methods have been executed in many learning and vision problems in practice. However, due to the complex network structure, little or even to no results have been proposed for the convergence behaviors of these methods. In summary, the lack of strict theoretical investigations is one of the most fundamental limits in prevalent learning-based iterative methods, especially in the challenging nonconvex scenario.

To break the limits of prevalent approaches, this paper explores Flexible Iterative Modularization Algorithm (FIMA), a *generic* and *convergent* algorithmic framework that combines together the learnable architecture (e.g., mainstream deep networks) with principled knowledges (formulated by mathematical models), to tackle challenging NIPs in Eq. (1). Specifically, derived from the fundamental forward-backward updating mechanism, FIMA replaces specific calculations corresponding to the fidelity and priors in Eq. (1) with two user-specified (learnable) computational modules. A series of theoretical investigations are established for FIMA. For example, we first prove the subsequence convergence of FIMA with explicit momentum policy (called eFIMA), which is as good as those mathematically designed nonconvex proximal methods with Nesterov's acceleration (e.g., various APGs in [6], [8], [9]). By introducing a carefully devised error-control policy (i.e., implicit momentum policy, called iFIMA), we further enhance the results and obtain a globally convergent Cauchy sequence for Eq. (1). We prove that this guarantee can also be preserved for FIMA with multiple blocks of unknown variables (called mFIMA). As a nontrivial byproduct, we finally show how to specify modules in FIMA for challenging inverse problems in low-level vision area (e.g., nonblind and blind image deconvolution). Our primary contributions are summarized as follows:

1) FIMA provides a generic framework that unifies almost all existing learning-based iterative methods, as well as a series of scheduling policies that make it possible to develop theoretically convergent learning-based iterations for challenging nonconvex inverse problems in the form of Eq. (1).
2) Even with highly flexible (learnable) iterations, the convergence guarantees obtained by FIMA is still as good as (eFIMA) or better (iFIMA) than prevalent mathematically designed nonconvex APGs. So it is worth noting that our devised scheduling policies together with the flexible algorithmic structures should also be beneficial for classical nonconvex algorithms.
3) FIMA also provides us a practical and effective ensemble of domain knowledge and sophisticated learned data distributions for real applications. Thus we can bring the expressive power of knowledge-based and data-driven methodologies to yield state-of-the-art performance on challenging low-level vision tasks.

2 Related Work

2.1 Classical First-Order Numerical Solvers

We first briefly review a group of classical first-order algorithms, which have been widely used to solve inverse problems. The gradient descent (GD) scheme on a differentiable function f can be reformulated as minimizing the following quadratic approximation of f at given point $\mathbf{v}$ with step size $\gamma > 0$, i.e., $Q_{\gamma f}(\mathbf{x};\mathbf{v}) := f(\mathbf{v}) + \langle \nabla f(\mathbf{v}), \mathbf{x}-\mathbf{v}\rangle + \frac{1}{2\gamma}\|\mathbf{x}-\mathbf{v}\|^2$. As for the nonsmooth function g, its proximal mapping (PM) with parameter $\gamma > 0$ can be defined as $\texttt{prox}_{\gamma g}(\mathbf{v}) = \arg\min_{\mathbf{x}} g(\mathbf{x}) + \frac{1}{2\gamma}\|\mathbf{x}-\mathbf{v}\|^2$. So it is natural to consider PG as cascade of GD (on f) and PM (on g), or equivalently optimizing the quadratic approximation of Eq. (1), i.e., $\mathbf{x}^{k+1} \in \arg\min_{\mathbf{x}} g(\mathbf{x}) + Q_{\gamma^k f}(\mathbf{x};\mathbf{v}^k)$, where $\mathbf{v}^k$ is some calculated variable at kth iteration. Thus most prevalent proximal schemes can be summarized as

$$\mathbf{v}^k = \begin{cases} \mathbf{x}^k, & \text{(A-1)} \\ \mathbf{x}^k + \beta^k(\mathbf{x}^k - \mathbf{x}^{k-1}), & \text{(A-2)} \end{cases}$$

$$\mathbf{x}^{k+1} \in \begin{cases} \texttt{prox}_{\gamma^k g}\big(\mathbf{v}^k - \gamma^k \nabla f(\mathbf{v}^k)\big), & \text{(B-1)} \\ \texttt{prox}^{\varepsilon^k}_{\gamma^k g}\big(\mathbf{v}^k - \gamma^k \nabla f\big(\mathbf{v}^k + \mathbf{e}^k\big)\big), & \text{(B-2)} \end{cases},$$

where ε^k and $\mathbf{e}^k$ in (B-2) denote the errors in PM and GD calculations, respectively [21]. Within this general scheme, we first obtain original PG by setting $\mathbf{v}^k = \mathbf{x}^k$ (i.e., (A-1)) and computing PM in (B-1) [2]. Using Nesterov's acceleration [1] (i.e., (A-2) with $\beta^k > 0$), we have the well-known APG method [2], [6], [9]. Moreover, by introducing ε^k and $\mathbf{e}^k$ to respectively capture the inexactness of PM and GD (i.e., (B-2)), we actually consider inexact PG and APG for both convex [22] and nonconvex [21] problems. Notice that in the nonconvex scenario, most classical APGs can only guarantee the subsequence convergence to the critical points [6], [9].

2.2 Learning-Based Iterative Methods

In [11], a trained version of FISTA (called LISTA) is introduced to approximate the solution of LASSO. [10], [23] extended LISTA for more generic sparse coding tasks and provided an adaptive acceleration. Unfortunately, LISTA is built on convex ℓ_1 regularization, thus may not be

applicable for other complex nonconvex inverse problems (e.g., ℓ_0 prior). By introducing hyperparameters in MRF and solving the resulted variational model with different iteration schemes, various learning-based iterative methods are proposed for inverse problems in image domain (e.g., denoising, super-resolution, and MRI imaging). For example, [13], [14], [15], [24], [25] have considered half-quadratic splitting, gradient descent, Alternating Direction Method of Multiplier (ADMM) and primal-dual method, respectively. But their parameterizations are completely based on MRF priors. Even worse, the original convergence properties are lost in these resulted iterations.

To better model complex image degradations, [16], [17], [18] considered Convolutional Neural Networks (CNNs) as implicit priors for image restoration. Since these methods discard the regularization term in Eq. (1), we may not enforce principled constraints on their solutions. It is also unclear when and where these iterative trajectories should stop. Another group of very recent works [19], [20] directly formulated the descent directions from reinforcement learning perspective or using recurrent networks. However, due to the high computational budgets, they can only be applied to relative simple tasks (e.g., linear regression). Besides, due to the complex topological network structure, it is extremely hard to provide strict theoretical analysis for these methods.

3 The Proposed Algorithms

This section develops Flexible Iterative Modularization Algorithm for nonconvex inverse problems in Eq. (1). The convergence behaviors are also investigated accordingly. Hereafter, some fairly loose assumptions are enforced on Eq. (1): f is proper and Lipschitz smooth (with modulus L) on a bounded set, g is proper, lower semi-continuous and proximable[1] and Ψ is coercive. Notice that the proofs and definitions are deferred until Supplementary Materials, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2019.2920591.

3.1 Abstract Iterative Modularization

As summarized in Section 2.1, a large amount of first-order methods can be summarized as forward-backward-type iterations. This motivates us to consider the following even more abstract updating principle:

$$\mathbf{x}^{k+1} = \mathcal{A}_g \circ \mathcal{A}_f(\mathbf{x}^k), \tag{2}$$

where $\mathcal{A}_f$ and $\mathcal{A}_g$ respectively stand for the user-specified modules for f and g, and $\circ$ denotes operation composition. Building upon this formulation, it is easy to see that designing a learning-based iterative method reduces to the problem of iteratively specifying and learning $\mathcal{A}_f$ and $\mathcal{A}_g$.

It is straightforward that most prevalent approaches [13], [14], [15], [16], [17], [18], [24] naturally fall into this general formulation. Nevertheless, currently it is still impossible to provide any strict theoretical results for practical trajectories of Eq. (2). This is mainly due to the lack of efficient mechanisms to control the propagations generated by these handcrafted operations. Fortunately, in the following, we will introduce different scheduling policies to automatically guide the iterations in Eq. (2), resulting in a series of *theoretically convergent learning-based iterative methods*.

3.2 Explicit Momentum: A Straightforward Strategy

The momentum of objective values is one of the most important properties for numerical iterations. This property is also necessary for analyzing the convergence of some classical algorithms. Inspired by these points, we present an explicit momentum FIMA (eFIMA) (i.e., Algorithm 1), in which we explicitly compare $\Psi(\mathbf{u}^k)$ and $\Psi(\mathbf{x}^k)$ and choose the variable with less objective value as our monitor (denoted as $\mathbf{v}^k$). Finally, a proximal refinement is performed to adjust the learning-based updating at each stage.

Algorithm 1. Explicit Momentum FIMA (eFIMA)

Require: $\mathbf{x}^0$, $\mathcal{A} = \{\mathcal{A}_g, \mathcal{A}_f\}$, and $\{0 < \gamma^k < 1/L\}$.
1: **while** not converged **do**
2: $\quad\mathbf{u}^k = \mathcal{A}_g \circ \mathcal{A}_f(\mathbf{x}^k)$.
3: $\quad$**if** $\Psi(\mathbf{u}^k) \leq \Psi(\mathbf{x}^k)$ **then**
4: $\quad\quad\mathbf{v}^k = \mathbf{u}^k$.
5: $\quad$**else**
6: $\quad\quad\mathbf{v}^k = \mathbf{x}^k$.
7: $\quad$**end if**
8: $\quad\mathbf{x}^{k+1} \in \mathtt{prox}_{\gamma^k g}\big(\mathbf{v}^k - \gamma^k \nabla f(\mathbf{v}^k)\big)$.
9: **end while**

The following theorem first verifies the sufficient descent of $\{\Psi(\mathbf{x}^k)\}_{k\in\mathbb{N}}$ and then proves the subsequence convergence of eFIMA. It is nice to observe that these results are not based on any specific choices of $\mathcal{A}_f$ and $\mathcal{A}_g$.

Theorem 1. *Let $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ be the sequence generated by eFIMA. Then at the kth iteration, there exists a sequence $\{\alpha^k | \alpha^k > 0\}_{k\in\mathbb{N}}$, such that*

$$\Psi\big(\mathbf{x}^{k+1}\big) \leq \Psi\big(\mathbf{v}^k\big) - \alpha^k \|\mathbf{x}^{k+1} - \mathbf{v}^k\|^2, \tag{3}$$

where $\mathbf{v}^k$ is the monitor in Algorithm 1. Furthermore, $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ is bounded and any of its accumulation points are the critical points of $\Psi(\mathbf{x})$ in Eq. (1).

Based on Theorem 1 and considering Ψ as a semi-algebraic function,[2] the convergence rate of eFIMA can be straightforwardly estimated as follows.

Corollary 1. *Let $\phi(s) = \frac{t}{\theta} s^\theta$ be a desingularizing function with a constant $t > 0$ and a parameter $\theta \in (0, 1]$ [27]. Then $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ generated by eFIMA converges after finite iterations if $\theta = 1$. The linear and sub-linear rates can be obtained if choosing $\theta \in [1/2, 1)$ and $\theta \in (0, 1/2)$, respectively.*

3.3 Implicit Momentum via Error Control

Indeed, even with the explicit momentum schedule, we may still not obtain a globally convergent iteration. This is mainly because that there is no policy to efficiently control the inexactness of the user-specified modules (i.e., $\mathcal{A}$).

1. The function g is proximable if $\min_{\mathbf{x}} g(\mathbf{x}) + \frac{\gamma}{2}\|\mathbf{x} - \mathbf{y}\|^2$ can be easily solved by the given $\mathbf{y}$ and $\gamma > 0$.

2. Indeed, a variety of functions (e.g., the indicator function of polyhedral set, ℓ_0 and rational ℓ_p penalties) satisfy the semi-algebraic property [26].

In this subsection, we show how to address this issue by controlling the first-order optimality error during iterations.

Specifically, we consider the auxiliary of Ψ at $\mathbf{x}^k$ (denoted as Ψ^k) and denote its sub-differential (denoted as $\mathbf{d}^{\mathbf{x}}_{\Psi^k}$)[3] as

$$\begin{aligned}&\Psi^k(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}) + \frac{\mu^k}{2}\|\mathbf{x}-\mathbf{x}^k\|^2,\\ &\mathbf{d}^{\mathbf{x}}_{\Psi^k} = \mathbf{d}^{\mathbf{x}}_g + \nabla f(\mathbf{x}) + \mu^k(\mathbf{x}-\mathbf{x}^k) \in \partial\Psi^k(\mathbf{x}),\end{aligned} \tag{4}$$

where $\mu^k > 0$ is the penalty parameter and $\mathbf{d}^{\mathbf{x}}_g \in \partial g(\mathbf{x})$.

As shown in Algorithm 2, at stage k, a variable $\tilde{\mathbf{u}}^k$ is obtained by proximally minimizing Ψ^k at $\mathbf{u}^k$ (i.e., Step 3 of Algorithm 2). Roughly, this new variable is just an ensemble of the last updated $\mathbf{x}^k$ and the output $\mathbf{u}^k$ of user-specified $\mathcal{A}$ following the specific proximal structure in Eq. (1). Then the monitor is obtained by checking the boundedness of $\mathbf{d}^{\tilde{\mathbf{u}}}_{\Psi^k}$. Notice that the constant C^k actually reveals our tolerance to the inexactness of $\mathcal{A}$ at kth iteration.

Algorithm 2. Implicit Momentum FIMA (iFIMA)

Require: $\mathbf{x}^0$, $\mathcal{A} = \{\mathcal{A}_g, \mathcal{A}_f\}$, $\{0 < 2C^k < \mu^k < \infty\}$, and $\{0 < \gamma^k < 1/L\}$.

1: **while** not converged **do**
2: $\mathbf{u}^k = \mathcal{A}_g \circ \mathcal{A}_f(\mathbf{x}^k)$.
3: $\tilde{\mathbf{u}}^k \in \mathtt{prox}_{\gamma^k g}\big(\mathbf{u}^k - \gamma^k\big(\nabla f(\mathbf{u}^k) + \mu^k(\mathbf{u}^k - \mathbf{x}^k)\big)\big)$.
4: **if** $\|\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k}\| \le C^k\|\tilde{\mathbf{u}}^k - \mathbf{x}^k\|$ **then**
5: $\mathbf{v}^k = \tilde{\mathbf{u}}^k$.
6: **else**
7: $\mathbf{v}^k = \mathbf{x}^k$.
8: **end if**
9: $\mathbf{x}^{k+1} \in \mathtt{prox}_{\gamma^k g}\big(\mathbf{v}^k - \gamma^k\nabla f(\mathbf{v}^k)\big)$.
10: **end while**

Proposition 1. *Let $\{\mathbf{x}^k, \tilde{\mathbf{u}}^k, \mathbf{v}^k\}_{k\in\mathbb{N}}$ be the sequences generated by Algorithm 2. Then there exist two sequences $\{\alpha^k | \alpha^k > 0\}_{k\in\mathbb{N}}$ and $\{\beta^k | \beta^k > 0\}_{k\in\mathbb{N}}$, such that the inequality (3) in Theorem 1 and $\Psi(\tilde{\mathbf{u}}^k) \le \Psi(\mathbf{x}^k) - \beta^k\|\tilde{\mathbf{u}}^k - \mathbf{x}^k\|^2$ are respectively satisfied.*

Equipped with Proposition 1, it will be straightforward to guarantee that the objective values generated by Algorithm 2 (i.e., $\{\Psi(\mathbf{x}^k)\}_{k\in\mathbb{N}}$) also has sufficient descent. So we call this version of FIMA as implicit momentum FIMA (iFIMA). Then the global convergence of iFIMA is proved as follows.

Theorem 2. *Let $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ be the sequence generated by iFIMA. Then $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ is bounded and any of its accumulation points are the critical points of Ψ. If Ψ is semi-algebraic, we further have that $\{\mathbf{x}^k\}_{k\in\mathbb{N}}$ is a Cauchy sequence, thus globally converges to a critical point of $\Psi(\mathbf{x})$ in Eq. (1).*

Indeed, based on Theorem 2, it is also easy to obtain the same convergence rate as that in Corollary 1 for iFIMA.

3. Strictly speaking, $\partial\Psi^k(\mathbf{x})$ is the so-called limiting Frechét sub-differential. We state its formal definition and propose a practical computation scheme for $\mathbf{d}^{\tilde{\mathbf{u}}}_{\Psi^k}$ in Appendix, available in the online supplemental material.

3.3.1 Practical Calculation of $\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k}$ in iFIMA

Here we propose a practical calculation scheme for $\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k} \in \partial\Psi^k(\tilde{\mathbf{u}}^k)$ defined in Eq. (4) and used in Algorithm 2. In fact, it is challenging to directly calculate $\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k}$ since the sub-differential $\mathbf{d}^{\tilde{\mathbf{u}}^k}_g$ is often intractable in the non-convex scenario [28]. Fortunately, our following analysis provides an efficient practical calculation scheme for $\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k}$ within FIMA framework. Specifically, by considering $\tilde{\mathbf{u}}^k$ as the solution for Step 3 of Algorithm 2, we have

$$\begin{aligned}\tilde{\mathbf{u}}^k &\in \mathtt{prox}_{\gamma^k g}\Big(\mathbf{u}^k - \gamma^k\big(\nabla f(\mathbf{u}^k) + \mu^k(\mathbf{u}^k - \mathbf{x}^k)\big)\Big)\\ &= \arg\min_{\mathbf{u}} \frac{1}{2}\Big\|\mathbf{u} - \Big(\mathbf{u}^k - \gamma^k\big(\nabla f(\mathbf{u}^k) + \mu^k(\mathbf{u}^k - \mathbf{x}^k)\big)\Big)\Big\|^2 \\ &\qquad + g(\mathbf{u}).\end{aligned}$$

By the first-order optimality condition, we further have

$$\begin{aligned}&0 \in \partial g(\tilde{\mathbf{u}}^k) + \frac{1}{\gamma^k}\big(\tilde{\mathbf{u}}^k - (\mathbf{u}^k - \gamma^k(\nabla f(\mathbf{u}^k) + \mu^k(\mathbf{u}^k - \mathbf{x}^k)))\big)\\ &\Rightarrow -\Big(\nabla f(\mathbf{u}^k) + \mu^k(\mathbf{u}^k - \mathbf{x}^k)\Big) + \frac{1}{\gamma^k}(\mathbf{u}^k - \tilde{\mathbf{u}}^k) \in \partial g(\tilde{\mathbf{u}}^k).\end{aligned}$$

Recalling the definition of $\mathbf{d}^{\mathbf{x}}_{\Psi^k}$ in Eq. (4) and $\mathbf{d}^{\mathbf{x}}_g \in \partial g(\mathbf{x})$, we finally obtain the practical calculation scheme for $\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k}$

$$\begin{aligned}\mathbf{d}^{\tilde{\mathbf{u}}^k}_{\Psi^k} &= \mathbf{d}^{\tilde{\mathbf{u}}^k}_g + \nabla f(\tilde{\mathbf{u}}^k) + \mu^k(\tilde{\mathbf{u}}^k - \mathbf{x}^k)\\ &= \big(\mu^k - 1/\gamma^k\big)\big(\tilde{\mathbf{u}}^k - \mathbf{u}^k\big) - \big(\nabla f\big(\mathbf{u}^k\big) - \nabla f\big(\tilde{\mathbf{u}}^k\big)\big).\end{aligned}$$

3.4 Multi-Block Extension

Algorithm 3. Multi-block FIMA (mFIMA)

Require: $\mathfrak{X}^0, \mathcal{A} = \{\mathcal{A}_{g_1}, \ldots, \mathcal{A}_{g_N}, \mathcal{A}_f\}$, $\{0 < 2C^k_n < \mu^k_n < \infty\}$, and $\{0 < \gamma^k_n < 1/L_n\}$.

1: **while** not converged **do**
2: **for** $n = 1 : N$ **do**
3: $\mathbf{u}^k_n = \mathcal{A}_{g_n} \circ \mathcal{A}_f\Big(\mathfrak{X}^{k+1}_{[<n]}, \mathfrak{X}^k_{[\ge n]}\Big)$.
4: $\tilde{\mathbf{u}}^k_n \in \mathtt{prox}_{\gamma^k_n g_n}(\mathbf{u}^k_n - \gamma^k_n(\nabla_n f(\mathfrak{X}^{k+1}_{[<n]}, \mathbf{u}^k_n, \mathfrak{X}^k_{[>n]}) + \mu^k_n(\mathbf{u}^k_n - \mathbf{x}^k_n)))$.
5: **if** $\|\mathbf{d}^{\tilde{\mathbf{u}}^k_n}_{\Psi^k_n}\| \le C^k_n\|\tilde{\mathbf{u}}^k_n - \mathbf{x}^k_n\|$ **then**
6: $\mathbf{v}^k_n = \tilde{\mathbf{u}}^k_n$.
7: **else**
8: $\mathbf{v}^k_n = \mathbf{x}^k_n$.
9: **end if**
10: $\mathbf{x}^{k+1}_n \in \mathtt{prox}_{\gamma^k_n g_n}\Big(\mathbf{v}^k_n - \gamma^k_n\nabla_n f\Big(\mathfrak{X}^{k+1}_{[<n]}, \mathbf{v}^k_n, \mathfrak{X}^k_{[>n]}\Big)\Big)$.
11: **end for**
12: **end while**

In order to tackle the inverse problems with blocks of unknown variables (e.g., blind deconvolution and dictionary learning), we now discuss how to extend FIMA for multi-block NIPs, which is formulated as $\mathcal{T}(\mathfrak{X}) = \mathbf{y} + \mathbf{n}$, where $\mathfrak{X} = \{\mathbf{x}_n\}_{n=1}^N \in \mathbb{R}^{D_1} \times \cdots \times \mathbb{R}^{D_N}$ is a set of $N \ge 2$ unknown variables to be estimated. Notice that here $\mathcal{T}$ should be some given linear operations on $\mathfrak{X}$. The inference of such problem can be addressed by solving

$$\min_{\mathfrak{X}} \Psi(\mathfrak{X}) := f(\mathfrak{X}; \mathcal{T}, \mathbf{y}) + \sum_{n=1}^{N} g_n(\mathbf{x}_n), \tag{5}$$

where $f(\mathfrak{X}) : \mathbb{R}^{D_1} \times \cdots \times \mathbb{R}^{D_N} \to (-\infty, +\infty]$ is still differentiable and each $g_n(\mathbf{x}_n) : \mathbb{R}^{D_n} \to (-\infty, +\infty]$ may also nonsmooth and possibly nonconvex. Here both f and blockwise g_n $(\mathbf{x}_n)$ follow the same assumptions as that in Eq. (1) and f should also satisfy the generalized Lipschitz smooth property on bounded subsets of $\mathbb{R}^{D_1} \times \cdots \times \mathbb{R}^{D_N}$. For ease of presentation, we denote $\mathfrak{X}_{[<n]} = \{\mathbf{x}_i\}_{i=1}^{n-1}$, $\mathfrak{X}_{[\leq n]} = \{\mathbf{x}_i\}_{i=1}^{n}$ and the subscripts $[> n]$ and $[\geq n]$ are defined in the same manner. Then we summarize the main iterations of multivariable FIMA (mFIMA) as follows:[4]

$$\mathbf{u}_n^k = \mathcal{A}_{g_n} \circ \mathcal{A}_f\left(\mathfrak{X}_{[<n]}^{k+1}, \mathfrak{X}_{[\geq n]}^{k}\right),$$
$$\mathbf{x}_n^{k+1} \in \mathtt{prox}_{\gamma^k g_n}\left(\mathbf{v}_n^k - \gamma^k \nabla_n f\left(\mathfrak{X}_{[<n]}^{k+1}, \mathbf{v}_n^k, \mathfrak{X}_{[>n]}^{k}\right)\right).$$

Here $\mathbf{v}_n^k$ is the monitor of $\mathbf{x}_n^k$, obtained by the same error control strategy as that in iFIMA. Then we summarize our multi-block FIMA in Algorithm 3 and prove the convergence of mFIMA in Corollary 2.

Corollary 2. *Let $\{\mathfrak{X}^k\}_{k\in\mathbb{N}}$ be the sequence generated by mFIMA. Then we have the same convergence properties as that in Theorem 2 and Corollary 1 for $\{\mathfrak{X}^k\}_{k\in\mathbb{N}}$.*

Then we summarize our multi-block FIMA in Algorithm 3. Notice that here we adopt the error-control policy in iFIMA to guide the iterations of mFIMA.

4 Discussions

Here we would like to discuss some important aspects of our FIMA, including the difference with traditional first order methods, learning-based iterative methods, and existing meta-learning techniques. We also discuss the relation between our eFIMA and iFIMA. For convenience, we specify the components in FIMA as three categories, including the flexible modules $\mathcal{A}_g \circ \mathcal{A}_f$, the various criteria (Steps 3-7 in Algorithm 1 and Steps 3-8 in Algorithm 2) and PG operator.

4.1 FIMA versus Traditional First Order Methods

We first point out that by specifying the user-specified modules $\mathcal{A}_f$ and $\mathcal{A}_g$ as numerical calculations, our FIMA can reduce to some traditional first-order numerical algorithms (e.g., PG/APG/PALM) when ignoring the criteria and PG operator. Specifically, by respectively specifying $\mathcal{A}_f$ and $\mathcal{A}_g$ as gradient descent and proximal operator, our e/iFIMA will be the standard PG [29]. Our e/iFIMA also can reduce to the standard APG [2] when adopting Nesterov's acceleration and proximal gradient operator as $\mathcal{A}_f$ and $\mathcal{A}_g$, respectively. Considering the two-block case in mFIMA, our method becomes PALM [26] when configuring the same strategy with PG. In these degradations, we can obtain the same convergence results with existing PG/APG.

When remaining the criteria and PG operator in FIMA, e/iFIMA will be the inexact APGs by setting $\mathcal{A}_g \circ \mathcal{A}_f$ similar with PG. For example, eFIMA will be the monotone APG [6] when arranging $\mathcal{A}_f$ and $\mathcal{A}_g$ as Nesterov's acceleration and proximal gradient operator, respectively. Furthermore, when transforming these arrangements into our iFIMA framework, it generates a new inexact APG method. However, the convergence performance based on iFIMA is even better than that for the prevalent nonconvex APGs [6], [8]. This actually suggests that our devised error-control policy together with the flexible algorithmic structures should also be beneficial for inexact nonconvex algorithms.

Moreover, our experimental results have demonstrated that thanks to the plug-and-play architectures, FIMA can achieve much better numerical performance and final results than these traditional first order numerical methods, especially in real-world applications. Indeed, FIMA provides a generic, flexible and theoretically guaranteed way to extend standard numerical methods using plug-and-play architectures.

4.2 FIMA versus Learning-Based Iterative Methods

As discussed above, most existing learning-based iterative methods only replace their numerical computations by trained architectures, which directly lead to the missing of necessary theoretical guarantees. Fortunately, within FIMA framework, we can prove in Corollary 1 and Theorem 2 that our newly proposed learning-based scheme does not depend on the particular choices of $\mathcal{A}_f$ and $\mathcal{A}_g$ in general. It actually provides us a unified methodology to analyze and improve the convergence issues for learning-based methods. Within FIMA, we can provide an easily-implemented and strictly convergent way to extend almost all the learning-based methods reviewed in Section 2.2. For example, we can design the gradient descent operator and the encoder architecture in LISTA [11] as $\mathcal{A}_f$ and $\mathcal{A}_g$, respectively. Then we cascade the explicit or implicit momentum in our algorithms to build the convergence guaranteed iterations. Similarly, we can also regard the learnable priors in MRF [24] as $\mathcal{A}_g$ and the rest part as $\mathcal{A}_f$ to generate our eFIMA or iFIMA. When designing the solution of the subproblem about fidelity as $\mathcal{A}_f$ and exploring the data distribution by denoise CNN [17] as $\mathcal{A}_g$, we also provide the convergence guarantee for these plug-and-play learnable iterations. The criterion in our algorithms actually provides the guidance to judge whether the output of each iteration in learning based methods is reasonable. Thus, almost all the learning-based methods can strictly converge with tiny assistance under our algorithm framework.

4.3 FIMA versus Existing Meta-Learning Techniques

Meta-learning (a.k.a. "learning to learn") aims to design methods that can learn how to learn new tasks by reusing previous experience, rather than considering each new task in isolation [30], [31], [32]. It should be noticed that FIMA can also be categorized as a specific meta-learning technique from the perspective of "learning to optimize". However, compared with existing approaches [20], [33], which just learn all the hyperparameters in their optimization processes in heuristic manners and thus miss solid theoretical investigations, the main advantages of our FIMA is that we provide a *theoretically guaranteed* framework to learn *strict convergence* optimization schemes for meta-learning. But please notice that to obtain these convergence results, we

4. Due to space limit, the details of mFIMA are presented in Supplemental Material, available online.

have to set some algorithmic parameters following the theoretical guidance.

4.4 eFIMA versus iFIMA

We first clarify that the main difference between eFIMA and iFIMA is the error-control condition. It can be seen that the optimally-based condition in iFIMA is a little bit stricter than the loss-based eFIMA condition. Thus we can obtain better convergence properties but additional computations are needed at each iterations.

As for the plug-and-play computational modules in e/iFIMA, it will be shown in Section 6 that the choices of $\mathcal{A}_f$ and $\mathcal{A}_g$ do affect our speed and accuracy in practice. Fortunately, the proposed scheduling of learnable and numerical modules are automatically and adaptively adjusted by error-control conditions of both eFIMA and iFIMA. So in the specific task, FIMA actually can automatically reject improper $\mathcal{A}_f$ and $\mathcal{A}_g$ during iterations.

5 Applications

As a nontrivial byproduct, this section illustrates how to apply FIMA to tackle practical inverse problems in low-level vision area, such as image deconvolution in the standard non-blind and even more challenging blind scenarios.

Non-blind Deconvolution (Uni-block) aims to restore the latent image $\mathbf{z}$ from corrupted observation $\mathbf{y}$ with known blur kernel $\mathbf{b}$. In this part, we utilize the well-known sparse coding formulation [2]: $\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{n}$, where $\mathbf{x}$, $\mathbf{D}$ and $\mathbf{n}$ are the sparse code, given dictionary and unknown noises, respectively. Indeed, the form of $\mathbf{D}$ is given as $\mathbf{D} = \mathbf{B}\mathbf{W}^\top$, where $\mathbf{B}$ is the matrix form of $\mathbf{b}$, $\mathbf{W}^\top$ denotes the inverse of the wavelet transform $\mathbf{W}$ (i.e., $\mathbf{x} = \mathbf{W}\mathbf{z}$ and $\mathbf{z} = \mathbf{W}^\top\mathbf{x}$). So by defining $f(\mathbf{x}; \mathbf{D}, \mathbf{y}) = \|\mathbf{y} - \mathbf{D}\mathbf{x}\|^2$ and $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_p$ $(0 \leq p < 1)$, we obtain a special case of Eq. (1) as follows

$$\min_{\mathbf{x}} f(\mathbf{x}; \mathbf{D}, \mathbf{y}) + g(\mathbf{x}). \tag{6}$$

Now we are ready to design iterative modules (i.e., $\mathcal{A}_f$ and $\mathcal{A}_g$) to optimize the SC model in Eq. (6). With the well-known imaging formulation $\mathbf{y} = \mathbf{b} \otimes \mathbf{z} + \mathbf{n}$ ($\otimes$ denotes the convolution operator), we actually update $\mathbf{z}$ by solving $\mathcal{A}_f(\mathbf{z}^k) := \arg\min_{\mathbf{z}} \|\mathbf{y} - \mathbf{b} \otimes \mathbf{z}\|^2 + \tau\|\mathbf{z} - \mathbf{z}^k\|^2$ to aggregate principles of the task and information from last updated variable, where $\mathbf{z}^k = \mathbf{W}^\top\mathbf{x}^k$ and τ is a positive constant. Then $\mathcal{A}_f$ on $\mathbf{x}$ can be defined as $\mathcal{A}_f(\mathbf{x}^k) = \mathbf{W}\mathcal{A}_f(\mathbf{z}^k)$, i.e.,

$$\mathcal{A}_f(\mathbf{x}^k) = \mathbf{W}(\mathbf{B}^T\mathbf{B} + \tau\mathbf{I})^{-1}\big(\mathbf{B}^T\mathbf{y} + \tau\mathbf{W}^\top\mathbf{x}^k\big), \tag{7}$$

where $\mathbf{I}$ is the identity matrix. It is easy to check that $\mathcal{A}_f$ can be efficiently calculated by FFT [24]. As for $\mathcal{A}_g$, we consider solving $\mathcal{A}_g(\mathcal{A}_f(\mathbf{z}^k)) := \arg\min_{\mathbf{z}} g(\mathbf{z}) + \tau\|\mathbf{z} - \mathcal{A}_f(\mathbf{z}^k)\|^2$ by a network to describe the distribution of latent image.

Blind Deconvolution (Multi-block) involves the joint estimation of both the latent image $\mathbf{z}$ and blur kernel $\mathbf{b}$, given only an observed $\mathbf{y}$. Here we formulate this problem on image gradient domain and solve the following special case of Eq. (5) with two unknown variables $(\mathbf{x}, \mathbf{b})$:[5]

$$\min_{\mathbf{x},\mathbf{b}} f(\mathbf{x}, \mathbf{b}; \nabla\mathbf{y}) + g_{\mathbf{x}}(\mathbf{x}) + g_{\mathbf{b}}(\mathbf{b}), \tag{8}$$

where $f(\mathbf{x}, \mathbf{b}; \nabla\mathbf{y}) = \|\nabla\mathbf{y} - \mathbf{b} \otimes \mathbf{x}\|^2$, $g_{\mathbf{x}}(\mathbf{x}) = \lambda_{\mathbf{x}}\|\mathbf{x}\|_0$, and $g_{\mathbf{b}}(\mathbf{b}) = \chi_{\Omega_{\mathbf{b}}}(\mathbf{b})$. Here $\chi_{\Omega_{\mathbf{b}}}$ is the indicator function of the set $\Omega_{\mathbf{b}} := \{\mathbf{b} \in \mathbb{R}^{D_{\mathbf{b}}} : [\mathbf{b}]_i \geq 0, \sum_{i=1}^{D_{\mathbf{b}}}[\mathbf{b}]_i = 1\}$, where $[\cdot]_i$ denotes the ith element. So the proximal updating in mFIMA corresponding to $g_{\mathbf{x}}$ and $g_{\mathbf{b}}$ can be respectively calculated by hard-thresholding [3] and simplex projection [34]. Here we need to specify three modules (i.e., $\mathcal{A}_f$, $\mathcal{A}_{g_{\mathbf{x}}}$ and $\mathcal{A}_{g_{\mathbf{b}}}$) for miFPG. We first follow similar idea in the non-blind case to define $\mathcal{A}_f(\mathbf{x}^k, \mathbf{b}^k)$ using the aggregated deconvolution energy

$$\begin{aligned}\mathcal{A}_f(\mathbf{x}^k, \mathbf{b}^k) := \arg\min_{\mathbf{x},\mathbf{b}} &\|\nabla\mathbf{y} - \mathbf{b} \otimes \mathbf{x}\|^2 \\ &+ \tau_{\mathbf{x}}\|\mathbf{x} - \mathbf{x}^k\|^2 + \tau_{\mathbf{b}}\|\mathbf{b} - \mathbf{b}^k\|^2,\end{aligned} \tag{9}$$

where $\tau_{\mathbf{b}}$ and $\tau_{\mathbf{x}}$ are positive constants. We then train CNNs on image gradient domain and solve $\min_{\mathbf{b}}\|\nabla\mathbf{y} - \mathbf{b} \otimes \mathbf{x}\|^2 + \lambda_{\mathbf{b}}\|\mathbf{b}\|^2$ using conjugate gradient method [35] to formulate $\mathcal{A}_{g_{\mathbf{x}}}$ and $\mathcal{A}_{g_{\mathbf{b}}}$, respectively.

Rain Streaks Removal (Multi-block) is another challenging task which focuses on removing the sporadic rain steak $\mathbf{r}$ and restoring rain free background scenes $\mathbf{z}$ from several types of visibility distorted observations $\mathbf{o}$. The observation $\mathbf{o}$ can be generated by $\mathbf{o} = \mathbf{z} + \mathbf{r}$. Considering the different sparsity of rain steak and background image, we formulate the sparse coding model as to minimize the following energy function:

$$\min_{\mathbf{x},\mathbf{c}} f(\mathbf{x}, \mathbf{c}; \mathbf{o}) + g_{\mathbf{x}}(\mathbf{x}) + g_{\mathbf{c}}(\mathbf{c}), \tag{10}$$

where $f(\mathbf{x}, \mathbf{c}; \mathbf{o}) = \|\mathbf{o} - \mathbf{W}^\top\mathbf{x} - \mathbf{W}^\top\mathbf{c}\|$, $g_{\mathbf{x}}(\mathbf{x}) = \lambda_{\mathbf{x}}\|\mathbf{x}\|_{0.8}$ and $g_{\mathbf{c}}(\mathbf{c}) = \lambda_{\mathbf{c}}\|\mathbf{c}\|_0$. Here, $\mathbf{W}$ is the wavelet transform which has explained in non-blind deconvolution. $\mathbf{x}$, $\mathbf{c}$ are the sparse codes of $\mathbf{z}$ and $\mathbf{r}$, respectively (i.e., $\mathbf{z} = \mathbf{W}^\top\mathbf{x}, \mathbf{r} = \mathbf{W}^\top\mathbf{c}$). By applying our multi-block FIMA to solve Eq. (10), we design $\mathcal{A}_f$ by the similar strategy in Eq. (9). As for $\mathcal{A}_{g_{\mathbf{x}}}$ and $\mathcal{A}_{g_{\mathbf{c}}}$, we achieve them by same network architecture but different training data.

6 Experimental Results

This section conducts experiments to verify our theoretical results and compares the performance of FIMA with other state-of-the-art learning-based iterative methods on real-world inverse problems. All experiments are performed on a PC with Intel Core i7 CPU at 3.4 GHz, 32 GB RAM and a NVIDIA GeForce GTX 1050 Ti GPU. More results can also be found in Supplemental Materials, available online.

6.1 Non-Blind Image Deconvolution

We first evaluate FIMA on solving Eq. (6) for image restoration. The test images are collected by [24], [36] and different levels of Gaussian noise are further added to generate our corrupted observations.

Modules Evaluation. First, the influences of different choices of $\mathcal{A}$ in FIMA is studied. Following Eq. (7), we adopt $\mathcal{A}_f^\tau$ with varying τ. As for $\mathcal{A}_g$, different choices are also considered: classical PG ($\mathcal{A}_g^{\mathtt{PG}}$), Recursive Filter [37] ($\mathcal{A}_g^{\mathtt{RF}}$), Total

5. Notice that in this section, $\mathbf{x}$ is defined with different meanings, i.e., image gradient in Eq. (8), while sparse code in Eq. (6).

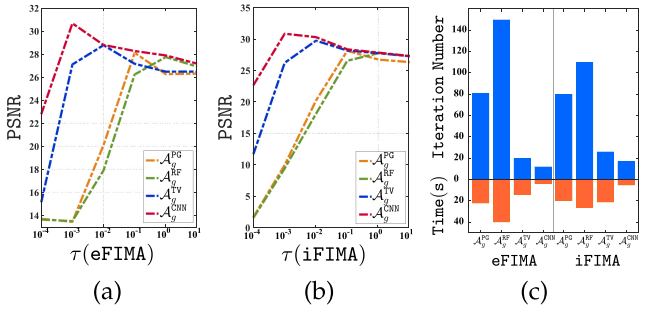


Fig. 1. Comparisons of FIMA with different $\mathcal{A}_f^\tau$ ($\tau \in [10^{-4}, 10^1]$) and $\mathcal{A}_g \in \{\mathcal{A}_g^{\text{PG}}, \mathcal{A}_g^{\text{RF}}, \mathcal{A}_g^{\text{TV}}, \mathcal{A}_g^{\text{CNN}}\}$. The bar charts in the rightmost subfigure compares the overall iteration number and running time (in seconds, "Time(s)" for short).

Variation [38] ($\mathcal{A}_g^{\text{TV}}$) and CNNs ($\mathcal{A}_g^{\text{CNN}}$). For $\mathcal{A}_g^{\text{CNN}}$, we introduce a residual structure $\mathbf{x} = \mathbf{x} + \mathcal{R}(\mathbf{x})$ [39] and define $\mathcal{R}$ as a cascade of 7 dilated convolution layers (with filter size 3×3). ReLUs are added between each two linear layers and batch normalizations are used for the 2nd to 6th linear layers. We collect 800 images, in which 400 have been used in [24] and the other 400 are randomly sampled from ImageNet [40] as training data. We set the patch size as 160×160, and feed 10 patches in each epoch. Finally, Adam is adopted and executed 60,000 epochs for learning rate ranging from 1e-4 to 1e-6. Here we just adopt similar strategies in [17] to train $\mathcal{A}_g^{\text{CNN}}$ with different noise levels. Thus, we only training $\mathcal{A}_g^{\text{CNN}}$ independently in our iterations. As for the algorithmic parameters, we set $\gamma^k = 0.1$, $\mu^k = 0.2$, and $C^k = 0.09$. Fig. 1 analyzes the contributions of $\mathcal{A}_f^\tau$ ($\tau \in [10^{-4}, 10^1]$) and $\mathcal{A}_g \in \{\mathcal{A}_g^{\text{PG}}, \mathcal{A}_g^{\text{RF}}, \mathcal{A}_g^{\text{TV}}, \mathcal{A}_g^{\text{CNN}}\}$. We observe that $\mathcal{A}_g^{\text{TV}}$ is relatively better than $\mathcal{A}_g^{\text{PG}}$ and $\mathcal{A}_g^{\text{RF}}$, while $\mathcal{A}_g^{\text{CNN}}$ performs consistently better and faster than other strategies. So hereafter we always utilize $\mathcal{A}_g^{\text{CNN}}$ in eFIMA and iFIMA. We also observe that even with different $\mathcal{A}_g$, relatively large τ in $\mathcal{A}_f^\tau$ will result in analogous quantitative results. Thus we experimentally set $\tau = 10^{-3}$ for $\mathcal{A}_f^\tau$ in eFIMA and iFIMA for all the experiments.

Convergence Behaviors. We then verify the convergence properties of FIMA. The convergence behaviors of both each module in our algorithms and other nonconvex APGs are considered. To be fair and comprehensive, we adopt specific iteration numbers ($K = 80$) and iteration errors ($\|\mathbf{x}^{k+1} - \mathbf{x}^k\|/\|\mathbf{x}^k\| \leq 10^{-4}$) as stopping criterion in Figs. 2 and 3, respectively.

In Figs. 2a, 2b, and 2c, we plot the curves of objective values ($\log(\Psi(\mathbf{x}^k))$), reconstruction errors ($\log(\|\mathbf{x}^{k+1} - \mathbf{x}^k\|^2/\|\mathbf{x}^k\|^2)$) and iteration errors for FIMA with different

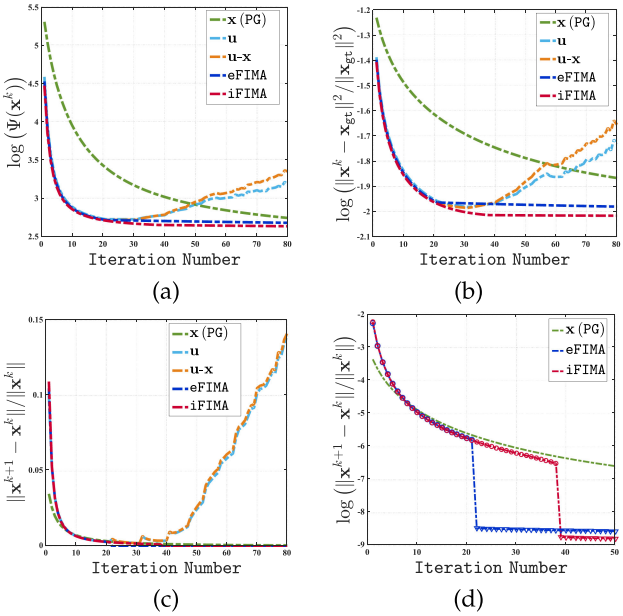


Fig. 2. The iteration curves of FIMA with different settings. The first three subfigures express the function values, constructive errors, and iteration errors, respectively. Subfigure (d) only plots the first 50 iterations for illustrate the scheduling policies of FIMA.

settings. The legends "**x**", "**u**", and "**u-x**" respectively denote that at each iteration, we only perform classical PG (i.e., only the last step in Algorithms 1 and 2), task-driven modules $\mathcal{A}$ (i.e., only perform Eq. (2)), and their naive combination (without any scheduling policies). It can be seen that the function values and reconstruction errors of PG decrease slower than our FIMA strategies, while both "**u**"-curve (i.e., naive $\mathcal{A}_g \circ \mathcal{A}_f$) and "**u-x**"-curve (i.e., $\mathcal{A}$ with PG refinement but no "explicit momentum" or "error-control" policy) have oscillations and could not converge after only 30 iterations. Moreover, we observe that adding PG to "**u**" (i.e., "**u-x**") make the curve worse rather than correct it to the descent direction. It illustrates that the pure adding strategies indeed break the convergence guarantee. In contrast, since of the choice mechanism in our algorithms, both eFIMA and iFIMA can provide a reliable variable ($\mathbf{v}^k$) in the current iteration to satisfy the convergence condition. We further explore the choice mechanism of FIMA in Fig. 2d. The "circles" in each curve represent the "explicit momentum" or "error-control" policy is satisfied, while the "triangles" denote only perform PG in the current stage. It can be seen that the eFIMA strategy is more strict than iFIMA, the judgment policy fails only 20 iterations in eFIMA while remains almost 40 iterations in

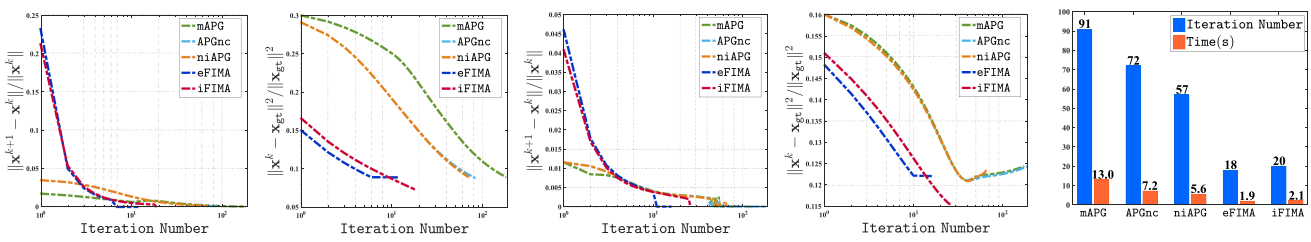


Fig. 3. Comparing iteration behaviors of FIMA to classical nonconvex APGs, including exact ones (mAPG, and APGnc) and inexact niAPG. The left four subfigures compare curves of iteration errors and reconstruction errors with different noise level (1‰ and 1 percent), respectively. The rightmost subfigure plot bar charts of the averaged iteration number and "Time(s)" on the dataset [24].

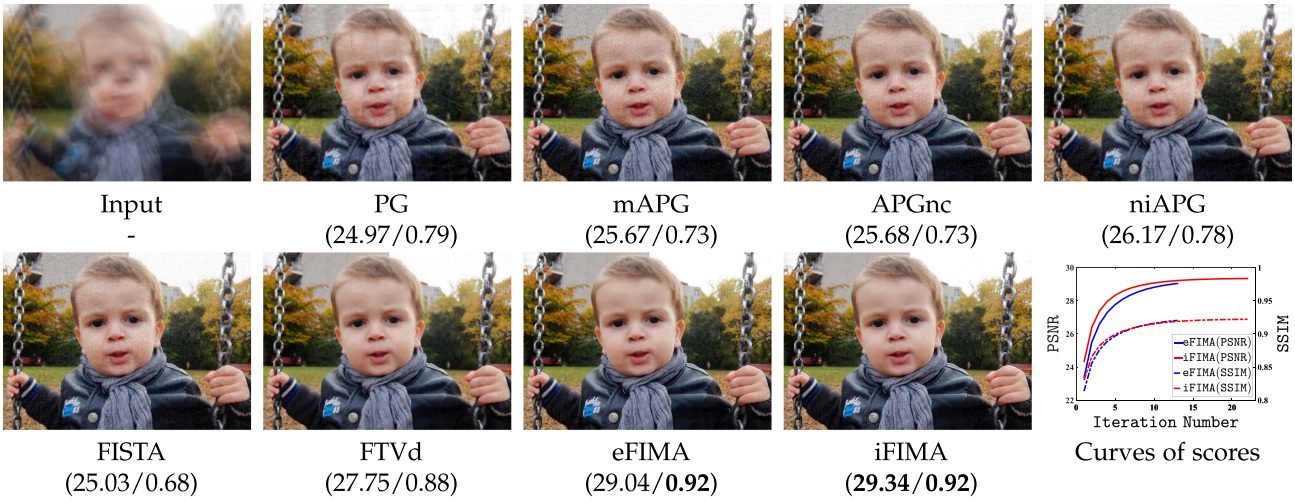


Fig. 4. The non-blind deconvolution performances (1 percent noise level) of eFIMA and iFIMA with comparisons to convex optimization based algorithms (i.e., FISTA and FTVd), and non-convex solvers (i.e., APGnc, mAPG, and niAPG). The quantitative scores (PSNR/SSIM) are reported below each image. The rightmost subfigure on the bottom row plots the curves of PSNR and SSIM of our methods.

iFIMA. Both eFIMA and iFIMA have better performance than other compared schemes, thus verifies the efficiency of our proposed scheduling policies in Section 3.

We also compare the iteration behaviors of FIMA to classical nonconvex APGs, including mAPG [6], APGnc [9]) and inexact niAPG [8] on the dataset collected by [24], which consists of 68 images corrupted by different blur kernels of the size ranging from 17×17 to 37×37. We add 1 percent and 1 percent Gaussian noise to generate our corrupted observations, respectively. In Fig. 3, the left four subfigures compare curves of iteration errors and reconstruction errors on an example image and the rightmost one illustrate the averaged iteration numbers and run time on the whole dataset. It can be seen that our eFIMA and iFIMA are faster and better than these abstractly designed classical solvers under the same iteration error ($\leq 1e-4$). Moreover, we observe that the performance of these nonconvex APGs is not satisfied when the noise level is bigger. The reconstruction errors of them (Fig. 3d) ascend after dozens of steps, while our eFIMA and iFIMA remain lower reconstruction error and fewer iterations. It illustrates that our strategy is more stable than traditional nonconvex APGs in image restoration because of the flexible modules and effective choice mechanisms.

In Fig. 4, we illustrate the visual results of eFIMA and iFIMA with comparisons to both convex image restoration approaches, including FISTA [2] (APG) and FTVd [41]) (ADMM), and nonconvex mAPG, APGnc, and niAPG on an example image with 1 percent noise level but large kernel size (i.e, 75×75) [36]. Here FISTA and FTVd solve their original convex models, while mAPG, APGnc, and niAPG are based on the nonconvex model in Eq. (6). We have that APGs outperformed the original PG. The inexact niAPG is better than exact mAPG and APGnc. Since FTVd is specifically designed for this task, it is the best among all classical solvers, but worse than our FIMA. Overall, iFIMA obtain higher PSNR than eFIMA since the error-control mechanism actually tend to perform more accurate refinements.

TABLE 1
The Number of Iterations (Including Plug-and-Play Modules) in FIMA

Image	eFIMA		iFIMA		PG
	No. Iter.	No. $\mathcal{A}$	No. Iter.	No. $\mathcal{A}$	No. Iter.
Fig. 4	13	11	22	21	542
Fig. 5	19	15	26	25	577

"No. Iter." reports the number of whole iterations and "No. $\mathcal{A}$" denotes the times the plug-and-play modules $\mathcal{A}_g \circ \mathcal{A}_f$ has been performed by FIMA during iterations. We also report the number of iterations for standard PG in the rightmost column.

We also analyze the iteration behaviors of FIMA in Table 1. We report the number of whole iterations and the times the plug-and-play modules $\mathcal{A}_g \circ \mathcal{A}_f$ has been performed by FIMA during iterations. It can be seen that $\mathcal{A}_g \circ \mathcal{A}_f$ are performed in most of the iterations. Moreover, thanks to these user-specified modules, FIMA only needs a dozen or twenty iterations to obtain our desired solutions. In contrast, there are more than 500 iterations in standard PG method. But the practical performances of PG are still worse than our FIMA (see Figs. 4 and 5 for comparisons). These results verify the efficiency and effectiveness of the mechanism of FIMA in real-world applications.

State-of-the-Art Comparisons. We compare FIMA with state-of-the-art image restoration approaches, such as IDDBM3D [42], EPLL [43], PPADMM [25], RTF [44] and IRCNN [17]. Fig. 5 first compares our FIMA with two prevalent learning-based iterative approaches (i.e., PPADMM and IRCNN) on an example image with 5 percent noise. Table 2 then reports the averaged quantitative results of all the compared methods on the image set (collected by [24]) with different levels of Gaussian noise (i.e., 1, 2, 3 and 4 percent). We have that eFIMA and iFIMA not only outperform classical numerical solvers by a large margin in terms of speed and accuracy, but also achieve better performance than other state-of-the-art approaches. Within FIMA, it can be seen that the speed of eFIMA is faster, while PSNR and SSIM of iFIMA are relatively higher. This is mainly because the "error control" strategy tends to perform more refinements than the "explicit momentum" rule during iterations.

Fig. 5. The non-blind image deconvolution performance (5 percent noise level) of FIMA with comparisons to existing plug-and-play type methods (i.e., PPADMM and IRCNN). The quantitative scores (PSNR/SSIM) are reported below each image.

TABLE 2
Averaged PSNR, SSIM and Time(s) on the Benchmark Image Set [24]

σ	Metric	State-of-the-art Image Restoration Methods					Classical Nonconvex Methods				Ours	
		IDDBM3D	EPLL	PPADMM	RTF	IRCNN	PG	mAPG	APGnc	niAPG	eFIMA	iFIMA
1%	PSNR	28.83	28.67	28.01	29.12	29.78	27.32	26.68	26.69	27.24	29.81	**29.85**
	SSIM	0.81	0.81	0.78	0.83	0.84	0.71	0.67	0.67	0.73	**0.85**	**0.85**
	Time(s)	193.13	112.03	293.99	249.83	2.67	20.36	13.02	7.16	5.29	**1.89**	2.06
2%	PSNR	27.60	26.79	26.54	25.58	27.90	25.61	25.20	25.28	25.63	28.02	**28.06**
	SSIM	0.76	0.74	0.72	0.66	0.78	0.63	0.60	0.61	0.64	**0.79**	**0.79**
	Time(s)	198.66	100.52	270.45	254.26	2.68	15.43	7.70	4.66	3.30	**1.90**	2.07
3%	PSNR	26.72	25.68	25.78	21.18	26.81	24.63	24.39	24.48	24.76	27.05	**27.07**
	SSIM	0.72	0.69	0.68	0.42	0.73	0.57	0.55	0.56	0.61	0.74	**0.75**
	Time(s)	191.25	96.32	257.94	252.47	2.68	13.89	6.44	5.37	2.63	**1.89**	2.07
4%	PSNR	26.06	24.88	25.27	17.95	26.10	24.05	23.88	23.95	24.14	26.20	**26.37**
	SSIM	0.69	0.65	0.66	0.28	0.70	0.54	0.53	0.53	0.59	0.70	**0.72**
	Time(s)	183.44	93.82	258.45	255.84	2.67	11.99	6.01	7.82	2.35	**1.89**	2.07

Here σ denotes the noise levels.

6.2 Blind Image Deconvolution

Blind deconvolution is known as one of the most challenging low-level vision tasks. Here we evaluate mFIAM on solving Eq. (8) to address this fundamentally ill-posed multi-variables inverse problem. We adopt the same CNN module $\mathcal{A}_g^{\text{CNN}}$ as that in Section 6.1 but train it on image gradient domain to enhance its ability for sharp edge detection.

In Fig. 6, we show the visual performances of mFIMA in different settings (i.e., with and without $\mathcal{A}$) on an example blurry image from [45]. We observe that mFIMA without $\mathcal{A}$ almost failed on this experiment. This is not surprising since [45], [46] have proved that standard optimization strategy is likely to lead to degenerate global solutions like the delta kernel (frequently called the no-blur solution), or many suboptimal local minima. In contrast, the CNN-based modules successful avoid trivial results and significantly improve the deconvolution performance. We also plot the curves of quantitative scores (i.e., PSNR for the latent image and Kernel Similarity (KS) for the blur kernel) on the bottom row for these two strategies on the bottom row. As these scores are stable after 20 iterations, here we only plot curves of the first 20 iterations.

We then compare mFIMA with state-of-the-art deblurring methods,[6] such as Perrone et al. [47], Levin et al. [45], Sun et al. [46], Zhang et al. [48] and Pan et al. [49] on the most widely-used Levin et al's benchmark [45], which consists of 32 blurred images generated by 4 clean images and 8 blur kernels. Table 3 reports the averaged quantitative scores, including PSNR, SSIM, and Error Rate (ER) for the latent image,

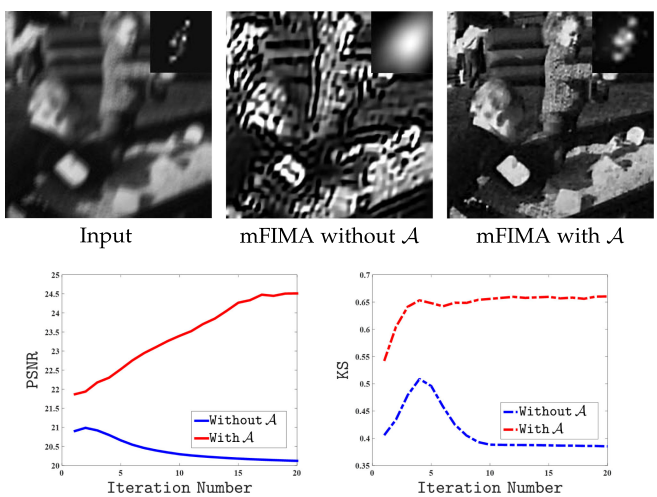


Fig. 6. The comparisons of mFIMA with and without the module $\mathcal{A}$. The top row compares the visual results of these different strategies. The bottom row plots the curves of PSNR and KS scores during iterations.

6. In this and the following experiments, the widely used multi-scale techniques are adopted for all the compared methods.

TABLE 3
Averaged Quantitative Scores on Levin et al.'s Benchmark

Method	PSNR	SSIM	ER	KS	Time(s)
Perrone et al.	29.27	0.88	1.35	0.80	113.70
Levin et al.	29.03	0.89	1.40	0.81	41.77
Sun et al.	29.71	0.90	1.32	0.82	209.47
Zhang et al.	28.01	0.86	1.25	0.58	37.45
Pan et al.	29.78	0.89	1.33	0.80	102.60
Ours	**30.37**	**0.91**	**1.20**	**0.83**	**5.65**

Kernel Similarity for the blur kernel and the overall run time. Fig. 7 further compares the visual performance of mFIMA to Perrone et al., Sun et al. and Pan et al. (i.e., top 3 in Table 3) on a real-world challenging blurry image collected by [36]. It can be seen that mFIMA consistently outperforms all the compared methods both quantitatively and qualitatively, which verifies the efficiency of our proposed learning-based iteration methodology.

In Figs. 8 and 9, we further compare the blind image deconvolution performance of mFIMA with Perrone et al. [47], Sun et al. [46] and Pan et al. [49] (top 3 among all the compared methods in Table 2) on example images corrupted by not only unknown blur kernels, but also different levels of Gaussian noises (1 and 3 percent in Figs. 8 and 9, respectively). It can be seen that mFIMA is robust to these corruptions and outperforms all the compared state-of-the-art deblurring methods.

6.3 Rain Streaks Removal

To further verify our method can deal with various vision tasks, we provide the performance of our mFIMA on rain streaks removal. As we claimed in Section 5, we adopt the same CNN architecture to train the learnable $\mathcal{A}_{g_x}$ and $\mathcal{A}_{g_c}$. It

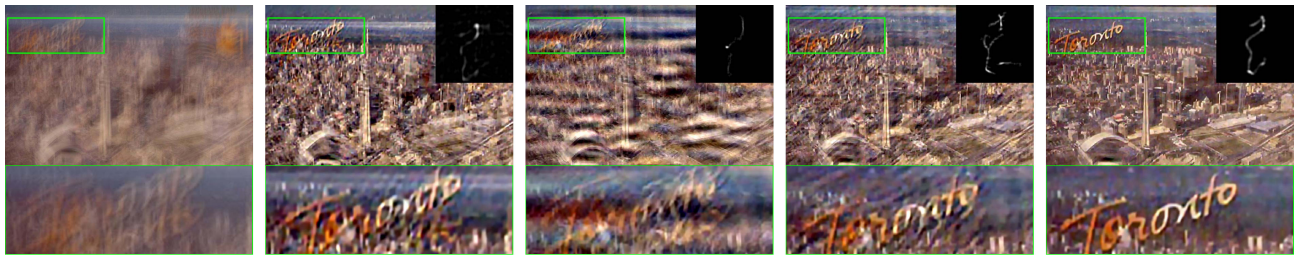


Fig. 7. Visual comparisons between mFIMA and other competitive methods (top 3 in Table 3) on a real blurry image.

Fig. 8. The blind image deconvolution results of mFIMA with comparisons to state-of-the-art approaches on blurry image with 1 percent Gaussian noise. The quantitative scores (i.e., PSNR / SSIM / KS) are reported below each image.

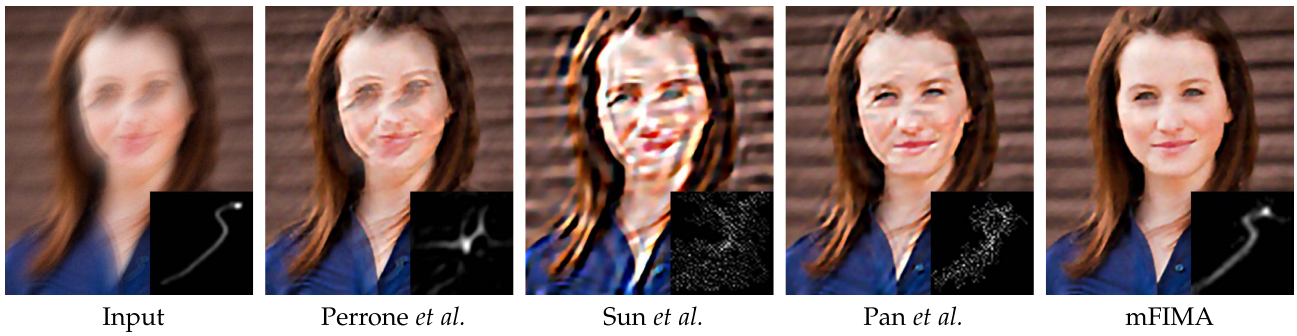


Fig. 9. The blind image deconvolution results of mFIMA with comparisons to state-of-the-art approaches on blurry facial image with 3 percent Guassian noise. The quantitative scores (i.e., PSNR / SSIM / KS) are reported below each image.

TABLE 4
Averaged PSNR and SSIM on the Benchmark Image Set [50] for Rain Streaks Removal

Methods	Metric	Image Index												Avg.
		1	2	3	4	5	6	7	8	9	10	11	12	
SR	PSNR	28.17	29.46	25.55	31.40	26.21	29.05	30.66	27.35	28.71	27.87	26.40	27.97	28.23
	SSIM	0.83	0.88	0.92	0.93	0.87	0.94	0.94	0.94	0.88	0.89	0.88	0.90	0.90
DSC	PSNR	28.07	27.81	22.72	32.59	23.26	26.32	29.85	24.95	29.16	26.21	26.94	27.70	27.13
	SSIM	0.85	0.91	0.84	0.97	0.88	0.96	0.96	0.94	0.90	0.85	0.90	0.87	0.90
LP	PSNR	32.01	32.81	28.99	33.31	28.10	30.33	32.32	30.26	31.15	30.58	29.08	31.21	30.85
	SSIM	0.90	0.94	0.95	0.97	0.93	0.98	0.97	0.97	0.94	0.93	0.93	0.95	0.95
DerainNet	PSNR	28.62	29.58	24.86	33.96	27.22	30.67	33.66	26.75	30.05	27.72	26.22	28.00	28.94
	SSIM	0.90	0.93	0.92	0.98	0.94	0.97	0.98	0.95	0.94	0.91	0.91	0.93	0.94
DetialNet	PSNR	32.55	32.97	29.00	36.27	30.03	31.67	36.13	30.82	33.13	31.56	29.20	31.85	32.10
	SSIM	0.93	0.96	0.95	**0.99**	0.96	0.98	**0.99**	0.98	0.97	0.94	0.94	0.96	0.96
UGSM	PSNR	32.42	34.03	27.85	38.03	29.40	34.92	37.94	28.40	30.28	30.33	29.31	30.80	31.98
	SSIM	0.93	0.96	0.94	**0.99**	0.96	**0.99**	**0.99**	0.97	0.92	0.93	0.94	0.94	0.95
mFIMA	PSNR	**34.10**	**36.24**	**29.46**	**40.76**	**31.94**	**35.10**	**39.51**	**35.37**	**36.66**	**33.02**	**31.17**	**33.41**	**34.73**
	SSIM	**0.95**	**0.98**	**0.96**	**0.99**	**0.97**	**0.99**	**0.99**	**0.99**	**0.98**	**0.96**	**0.95**	**0.97**	**0.97**

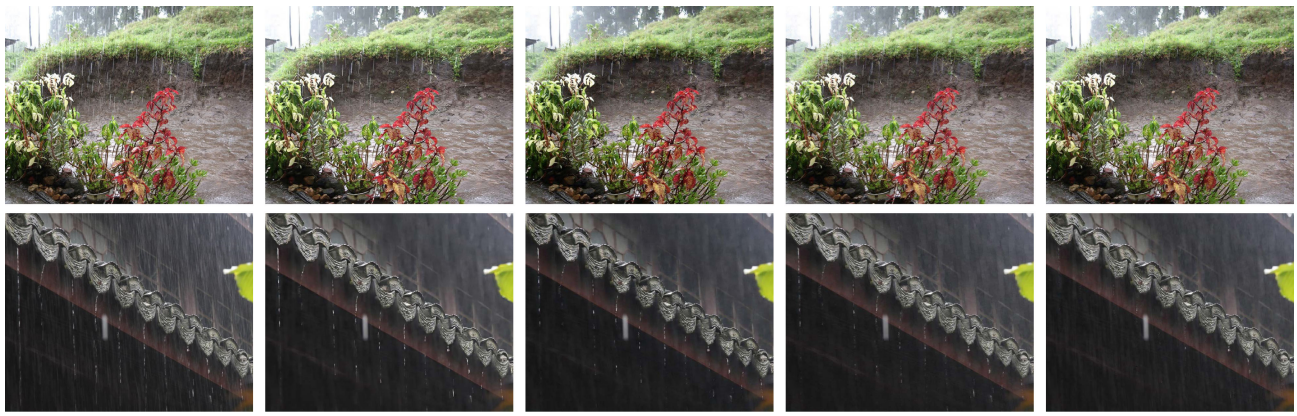


Fig. 10. Rain streaks removal results of mFIMA with comparisons to the state-of-the-art approaches on real-world rainy images.

should be noticed that we feed rainy observations into network and output the rain steak images to train the $\mathcal{A}_{g_c}$, while adopting the similar strategy with deconvolution to train $\mathcal{A}_{g_x}$.

First, we reported the quantitative scores (PSNR / SSIM) on a widely used synthetic Rain12 dataset [50] and compared with state-of-the-art deraining methods including SR [51], DSC [52], LP [50], DerainNet [53], and DetailNet [54], and UGSM [55]. As Table 4 shown, our mFIMA is easily superior to all of the competitive methods. Moreover, we also provide the visual results on the challenging real-world rainy image from [52] in Fig. 10. As can be observed, our proposed method can remove more rain streaks and preserve the more detail textures than others.

7 Conclusion

This paper provided FIMA, a framework to analyze the convergence behaviors of learning-based iterative methods for nonconvex inverse problems. We proposed two novel mechanisms to adaptively guide the trajectories of learning-based iterations and proved their strict convergence. We also showed how to apply FIMA for real-world applications, such as non-blind deconvolution, blind image deconvolution, and rain streaks removal.

Acknowledgments

R. Liu, S. Cheng, Y. He, X. Fan and Z. Luo are supported in part by the National Natural Science Foundation (NSF) of China (grant nos. 61672125, 61733002, 61572096 and 61632019), and the Fundamental Research Funds for the Central Universities. Z. Lin is supported by 973 Program of China (grant no. 2015CB352502), NSF of China (grant nos. 61625301 and 61731018), Qualcomm, and Microsoft Research Asia.

References

[1] Y. Nesterov, "A method of solving a convex programming problem with convergence rate o $(1/k^2)$," *Soviet Math. Doklady*, vol. 27, no. 2, pp. 372–376, 1983.

[2] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[3] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via ℓ_0 gradient minimization," *ACM Trans. Graph.*, vol. 30, no. 6, 2011, Art. no. 174.

[4] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis," *J. ACM*, vol. 58, no. 3, 2011, Art. no. 11.

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

[6] H. Li and Z. Lin, "Accelerated proximal gradient methods for nonconvex programming," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 379–387.

[7] H. Attouch and J. Bolte, "On the convergence of the proximal algorithm for nonsmooth functions involving analytic features," *Math. Program.*, vol. 116, no. 1, pp. 5–16, 2009.

[8] Q. Yao, J. T. Kwok, F. Gao, W. Chen, and T.-Y. Liu, "Efficient inexact proximal gradient algorithm for nonconvex problems," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 3308–3314.

[9] Q. Li, Y. Zhou, Y. Liang, and P. K. Varshney, "Convergence analysis of proximal gradient with momentum for nonconvex optimization," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2111–2119.

[10] T. Moreau and J. Bruna, "Understanding trainable sparse coding via matrix factorization," *arXiv preprint arXiv:1609.00285*, 2016.

[11] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 399–406.

[12] R. Liu, G. Zhong, J. Cao, Z. Lin, S. Shan, and Z. Luo, "Learning to diffuse: A new perspective to design PDEs for visual analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 12, pp. 2457–2471, Dec. 2016.

[13] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1256–1272, Jun. 2017.

[14] Y. Yang, J. Sun, H. Li, and Z. Xu, "ADMM-Net: A deep learning approach for compressive sensing MRI," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 10–18.

[15] S. Wang, S. Fidler, and R. Urtasun, "Proximal deep structured models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 865–873.

[16] S. Diamond, V. Sitzmann, F. Heide, and G. Wetzstein, "Unrolled optimization with deep priors," *arXiv preprint arXiv:1705.08041*, 2017.

[17] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep CNN denoiser prior for image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3929–3938.

[18] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9446–9454.

[19] K. Li and J. Malik, "Learning to optimize," *arXiv preprint arXiv:1606.01885*, 2016.

[20] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, and N. de Freitas, "Learning to learn by gradient descent by gradient descent," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3981–3989.

[21] B. Gu, D. Wang, Z. Huo, and H. Huang, "Inexact proximal gradient methods for non-convex and non-smooth optimization," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3093–3100.

[22] M. Schmidt, N. L. Roux, and F. R. Bach, "Convergence rates of inexact proximal-gradient methods for convex optimization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 1458–1466.

[23] A. Bronstein, P. Sprechmann, and G. Sapiro, "Learning efficient structured sparse models," in *Proc. Int. Conf. Mach. Learn.*, vol. 1, pp. 615–622, 2012.

[24] U. Schmidt and S. Roth, "Shrinkage fields for effective image restoration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2774–2781.

[25] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-play ADMM for image restoration: Fixed-point convergence and applications," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 84–98, Mar. 2017.

[26] J. Bolte, S. Sabach, and M. Teboulle, "Proximal alternating linearized minimization for nonconvex and nonsmooth problems," *Math. Program.*, vol. 146, no. 1/2, pp. 459–494, 2014.

[27] E. Chouzenoux, J.-C. Pesquet, and A. Repetti, "A block coordinate variable metric forward–backward algorithm," *J. Global Optim.*, vol. 66, no. 3, pp. 457–485, 2016.

[28] H. Attouch, J. Bolte, and B. F. Svaiter, "Convergence of descent methods for semi-algebraic and tame problems: Proximal algorithms, forwardbackward splitting, and regularized gaussseidel methods," *Math. Program.*, vol. 137, pp. 91–129, 2013.

[29] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, 2004.

[30] L.-Y. Gui, Y.-X. Wang, D. Ramanan, and J. M. Moura, "Few-shot human motion prediction via meta-learning," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 441–459.

[31] Y.-X. Wang and M. Hebert, "Learning from small sample sets by combining unsupervised meta-training with CNNs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 244–252.

[32] C. Zhang, Y. Yu, and Z.-H. Zhou, "Learning environmental calibration actions for policy self-evolution," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3061–3067.

[33] S. Thrun and L. Pratt, *Learning to Learn*. Berlin, Germany: Springer Science & Business Media, 2012.

[34] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," in *Proc. Int. Conf. Mach. Learn.*, 2008, pp. 272–279.

[35] S. Cho, J. Wang, and S. Lee, "Handling outliers in non-blind image deconvolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2011, pp. 495–502.

[36] W. S. Lai, J. B. Huang, Z. Hu, N. Ahuja, and M. H. Yang, "A comparative study for single image blind deblurring," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1701–1709.

[37] M. Unser, A. Aldroubi, and M. Eden, "Recursive regularization filters: Design, properties, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 3, pp. 272–277, Mar. 1991.

[38] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, 2008.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[40] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[41] Y. Wang, J. Yang, W. Yin, and Y. Zhang, "A new alternating minimization algorithm for total variation image reconstruction," *SIAM J. Imag. Sci.*, vol. 1, no. 3, pp. 248–272, 2008.

[42] A. Danielyan, V. Katkovnik, and K. Egiazarian, "BM3D frames and variational image deblurring," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1715–1728, Apr. 2012.

[43] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Proc. IEEE Conf. Int. Conf. Comput. Vis.*, 2011, pp. 479–486.

[44] U. Schmidt, J. Jancsary, S. Nowozin, S. Roth, and C. Rother, "Cascades of regression tree fields for image restoration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 677–689, Apr. 2016.

[45] A. Levin, Y. Weiss, F. Durand, and W. T. Freeman, "Understanding and evaluating blind deconvolution algorithms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 1964–1971.

[46] L. Sun, S. Cho, J. Wang, and J. Hays, "Edge-based blur kernel estimation using patch priors," in *Proc. IEEE Int. Conf. Comput. Photography*, 2013, pp. 1–8.

[47] D. Perrone and P. Favaro, "Total variation blind deconvolution: The devil is in the details," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 2909–2916.

[48] H. Zhang, D. Wipf, and Y. Zhang, "Multi-image blind deblurring using a coupled adaptive sparse prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1051–1058.

[49] J. Pan, D. Sun, H. Pfister, and M.-H. Yang, "Deblurring images via dark channel prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2315–2328, Oct. 2018.

[50] Y. Li, R. T. Tan, X. Guo, J. Lu, and M. S. Brown, "Rain streak removal using layer priors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2736–2744.

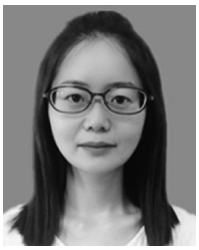
[51] L.-W. Kang, C.-W. Lin, and Y.-H. Fu, "Automatic single-image-based rain streaks removal via image decomposition," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 1742–1755, Apr. 2012.

[52] Y. Luo, Y. Xu, and H. Ji, "Removing rain from a single image via discriminative sparse coding," in *Proc. IEEE Conf. Int. Conf. Comput. Vis.*, Dec. 2015, pp. 3397–3405.

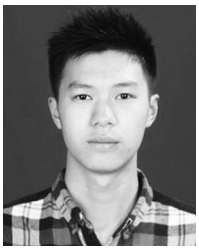
[53] X. Fu, J. Huang, X. Ding, Y. Liao, and J. Paisley, "Clearing the skies: A deep network architecture for single-image rain removal," *IEEE Trans. Image Process.*, vol. 26, no. 6, pp. 2944–2956, Apr. 2017.

[54] X. Fu, J. Huang, Y. Huang, D. Zeng, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1715–1723.
[55] L.-J. Deng, T.-Z. Huang, X.-L. Zhao, and T.-X. Jiang, "A directional global sparse model for single image rain removal," *Appl. Math. Modelling*, vol. 59, pp. 662–679, 2018.

Risheng Liu (M'12-) received the BSc and PhD degrees both in mathematics from the Dalian University of Technology, in 2007 and 2012, respectively. He was a visiting scholar with the Robotic Institute of Carnegie Mellon University from 2010 to 2012. He served as Hong Kong Scholar research fellow at the Hong Kong Polytechnic University from 2016 to 2017. He is currently an associate professor with the Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Internal School of Information and Software Technology, Dalian University of Technology. His research interests include machine learning, optimization, computer vision and multimedia. He was a co-recipient of the IEEE ICME Best Student Paper Award in both 2014 and 2015. Two papers were also selected as Finalist of the Best Paper Award in ICME 2017. He is a member of the IEEE and ACM.



Shichao Cheng received the BE degree in mathematics and applied mathematics from Henan Normal University. Xinxiang, China, in 2013. She is currently working toward the PhD degree in computational mathematics at the Dalian University of Technology, Dalian, China. Her research interests include computer vision, machine learning and optimization.



Yi He received the BE degree in computer science from Shihezi University, Shihezi, China, in 2017. He is currently working toward the master's degree in software engineering at the Dalian University of Technology, Dalian, China. His research interests include computer vision, deep learning.

Xin Fan received the BE and PhD degrees in information and communication engineering from Xian Jiaotong University, Xian, China, in 1998 and 2004, respectively. He was with Oklahoma State University, Stillwater, from 2006 to 2007, as a post-doctoral research fellow. He joined the School of Software, Dalian University of Technology, Dalian, China, in 2009. His current research interests include computational geometry and machine learning, and their applications to low-level image processing and DTI-MR image analysis. He is a senior member of the IEEE.

Zhouchen Lin (M'00-SM'08-F'18) received the PhD degree in applied mathematics from Peking University, in 2000. He is currently a professor with the Key Laboratory of Machine Perception, School of Electronics Engineering and Computer Science, Peking University. His research interests include computer vision, image processing, machine learning, pattern recognition, and numerical optimization. He is an area chair of ACCV 2009/2018, CVPR 2014/2016/2019, ICCV 2015, NIPS 2015/2018/2019 and AAAI 2019/2020, and senior program committee member of AAAI 2016/2017/2018 and IJCAI 2016/2018/2019. He is an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and the *International Journal of Computer Vision*. He is an IAPR/IEEE fellow.

Zhongxuan Luo received the BS degree in computational mathematics from Jilin University, China, in 1985, the MS degree in computational mathematics from Jilin University, in 1988, and the PhD degree in computational mathematics from the Dalian University of Technology, China, in 1991. He has been a full professor of the School of Mathematical Sciences, Dalian University of Technology since 1997. His research interests include computational geometry and computer vision.