A Numerical DEs Perspective on Unfolded Linearized ADMM Networks for Inverse Problems

Weixin An
Xidian University
weixinanut@163.com

Yingjie Yue
Xidian University
yueyingjie001@163.com

Yuanyuan Liu*
Xidian University
yyliu@xidian.edu.cn

Fanhua Shang*
Tianjin University
fhshang@xidian.edu.cn

Hongying Liu*
Xidian University
Peng Cheng Laboratory
hyliu@xidian.edu.cn

ABSTRACT

Many research works show that the continuous-time Differential Equations (DEs) allow for a better understanding of traditional Alternating Direction Multiplier Methods (ADMMs). And many unfolded algorithms directly inherit the traditional iterations to build deep networks. Although they obtain a faster convergence rate and superior practical performance, there is a lack of an appropriate explanation of the unfolded network architectures. Thus, we attempt to explore the connection between the existing unfolded Linearized ADMM (LADMM) and numerical DEs, and propose efficient unfolded network design schemes. First, we present an unfolded Euler LADMM scheme as a by-product, which originates from the Euler method for solving first-order DEs. Then inspired by the trapezoid method in numerical DEs, we design a new more effective network scheme, called unfolded Trapezoid LADMM scheme. Moreover, we analyze that the Trapezoid LADMM scheme has higher precision than the Euler LADMM scheme. To the best of our knowledge, this is the first work to explore the connection between unfolded ADMMs and numerical DEs with theoretical guarantees. Finally, we instantiate our Euler LADMM and Trapezoid LADMM schemes into ELADMM and TLADMM with the proximal operators, and ELADMM-Net and TLADMM-Net with convolutional neural networks. And extensive experiments show that our algorithms are competitive with state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies** → **Regularization**; **Neural networks**; **Reconstruction**; *Information extraction.*

KEYWORDS

differential equations, unfolded LADMMs, network architecture design, inverse problems

*Corresponding author

MM '22, October 10–14, 2022, Lisboa, Portugal

ACM ISBN 978-1-4503-9203-7/22/10…$15.00
https://doi.org/10.1145/3503161.3547887

ACM Reference Format:
Weixin An, Yingjie Yue, Yuanyuan Liu, Fanhua Shang, and Hongying Liu. 2022. A Numerical DEs Perspective on Unfolded Linearized ADMM Networks for Inverse Problems. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22), October 10–14, 2022, Lisboa, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3503161.3547887

1 INTRODUCTION

In machine learning, we often encounter some constrained optimization problems, which are formulated as the following problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d,\mathbf{y}\in\mathbb{R}^m} f(\mathbf{x}) + g(\mathbf{y}),\ \text{s.t. } \mathbf{A}\mathbf{x}+\mathbf{y}=\mathbf{b}, \tag{1}$$

where $\mathbf{A}\in\mathbb{R}^{m\times d}$, $\mathbf{b}\in\mathbb{R}^m$, $f:\mathbb{R}^d\to\mathbb{R}$ and $g:\mathbb{R}^m\to\mathbb{R}$ are convex functions but maybe non-smooth. For the more general case with the constraint $\mathbf{A}\mathbf{x}+\mathbf{C}\mathbf{y}=\mathbf{b}$, there are some similar algorithms for it, where $\mathbf{C}\in\mathbb{R}^{m\times m}$, and this paper mainly focuses on the case of an identity matrix $\mathbf{C}$. The problem (1) naturally appears in many multimedia fields [10, 21]. A common application is ℓ_1-norm regularized ill-posed inverse problems, including image denoising, image inpainting, and compressive sensing (CS) [9, 10, 23, 39]. For example, many algorithms have achieved good performance in natural image CS. And in this paper, we propose novel algorithms for CS problems, which achieve better performance in terms of quality, speed, and storage cost, as shown in Fig. 1.

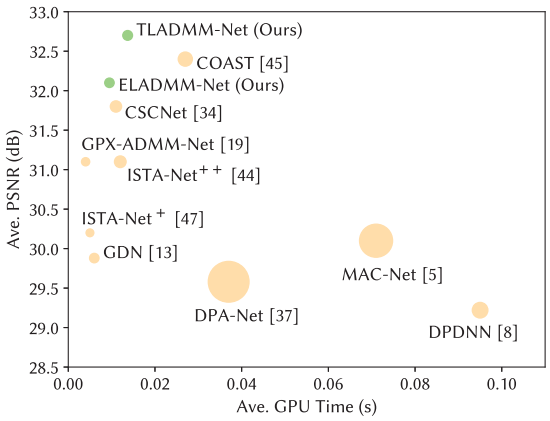

Figure 1: Comparison of the results for natural images CS tasks at CS ratio $\gamma = 30\%$ on BSD68 dataset, and the circle size is proportional to the number of parameters. More results are shown in Table 3.

To solve Problem (1), ADMM is a common choice [3], which alternately updates the variables in the Gauss-Seidel manner. However, it suffers from a heavy computational complexity due to performing matrix inversions. To alleviate this issue, [25] proposed an efficient LADMM by linearizing the quadratic penalty term. [27] and [14] introduced a Nesterov momentum term to accelerate ADMM. [7] proposed Jacobi ADMM to facilitate parallel computing. Besides, there are other stochastic versions [30, 48] to reduce complexity of ADMMs. The connection between these traditional ADMMs is shown in Fig. 2. Nonetheless, it is not trivial to set the hyperparameters (e.g., update rates) of traditional ADMMs, and these algorithms can not meet the high solution speed requirements.

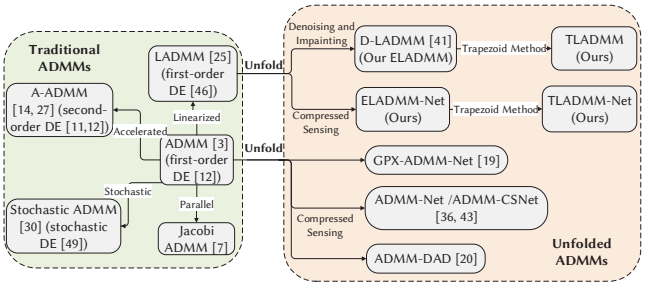

Figure 2: Traditional ADMMs and unfolded ADMMs as DEs. More detailed description of this figure is shown in Section 2.

Inspired by the idea of end-to-end learning, many researchers unfolded the traditional optimization algorithms into deep neural networks (DNNs) [22, 28, 38], called unfolded algorithms, which obtained superior performance and interpretability in solving various ill-posed inverse problems. For example, [6] proposed the asymptotic coupling between the weights in LISTA [15], which reduced the trainable parameters and improved the performance of LISTA. Moreover, [6] first provided the rigorous analysis of a linear convergence. [24, 26, 40] further improved the network structure and theoretical insights of LISTA. Besides, compared with DNN-based methods, [44] shows that unfolded algorithms usually obtain stronger generalization power on small-scale datasets. As for ADMMs, [31, 36, 41, 43] proposed to train some learnable weights, or introduce certain special network, e.g., convolutional neural network (CNN), for image restoration problems, which can achieve extraordinary experimental results.

On the other hand, the convergence analysis of unfolded algorithms becomes more difficult than traditional algorithms due to trainable parameters. Many existing works mainly simplify the proof by adding constraints to the learned parameters [6, 41], but their principles are not yet fully understood, which motivates researchers to find a more concise framework to prove convergence. Recently, by connecting the gradient-based methods and DEs, significant progress has been made in bridging this theoretical gap. For example, [35] formulated a second-order ordinary DE as the continuous limit of Nesterov's accelerated gradient method. [32] regarded the LISTA as a residual network and explained it as the Euler method in numerical DE. Nevertheless, for the more widely used unfolded ADMMs, their DEs theoretical behavior remains a mystery. To solve this problem, we need a new algorithm design and theoretical analysis to answer the following questions:

1. Is there a close connection between unfolded ADMMs and numerical discretizations of DEs with theoretical guarantees?
2. Can more accurate numerical discretizations of DEs derive superior unfolded ADMMs?

Moreover, we find that some unfolded ADMMs can not significantly improve the data reconstruction performance by increasing the number of network layers and even worsen. Thus, it is necessary to design better unfolded ADMMs on small-scale datasets or under the condition of limited parameter quantities and layers.

Our Contributions: In this paper, we establish a connection between unfolded ADMMs and numerical discretizations of DEs with theoretical guarantees. Moreover, we propose novel DEs guided unfolded ADMM design schemes to obtain superior practice performance and better theoretical results for solving Problem (1). Our main contributions are listed as follows:

• We first propose a unified framework, named unfolded Euler LADMM scheme. Then we derive that the Euler LADMM scheme is closely related to the Euler method for solving first-order approximating DEs in Lemma 1 below.

• The trapezoid method is introduced into unfolded LADMM to obtain an implicit Trapezoid LADMM scheme. Moreover, our theoretical result in Theorem 1 shows that the proposed unfolded Trapezoid scheme has better precision than unfolded Euler scheme. To reduce the computational burden, we also propose a novel explicit Trapezoid LADMM scheme, which turns the implicit Trapezoid LADMM scheme into an iterative algorithm through the idea of forecast-correction. Then, we analyze its convergence properties. **To the best of our knowledge, this is the first work to analyze the unfolded LADMM from the perspective of numerical DEs with theoretical guarantees.** Fig. 2 shows the connections of our algorithms and related methods.

• For different applications, we also implement the proposed Euler and explicit Trapezoid LADMM schemes by replacing the nonlinear operators with proximal operators or convolutional networks. As a result, we design four special algorithms, named ELADMM, ELADMM-Net, TLADMM, and TLADMM-Net, respectively.

• We first conduct an image denoising task on small-scale datasets, which confirms that our ELADMM and TLADMM outperform compared algorithms under the condition of limited parameter quantity. Secondly, we extensively evaluate the advantages of our ELADMM and TLADMM against ISTA-based networks for image inpainting tasks. Lastly, we also perform CS tasks to verify the promising performance of our ELADMM-Net and TLADMM-Net.

2 RELATED WORKS

The connection between numerical DEs and optimization algorithms was observed by [2, 4, 18, 33], which pointed out that optimization algorithms can be regarded as discretizations of DEs. The basic idea is to make the step size very small so that the solution path converges to the curve modeled by the DEs. Probably the simplest optimization algorithm related to numerical DEs is gradient descent (GD). Considering the objective function $f(\mathbf{x})$, if we set the step size to be infinitesimal, then GD can be viewed as:

$$\dot{\mathbf{X}} = -\nabla f(\mathbf{X}), \tag{2}$$

where $\mathbf{X} = \mathbf{X}(t)$ denotes the continuous limit of $\mathbf{x}_k$ and $\dot{\mathbf{X}} \equiv \frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t}$. These findings help researchers analyze optimization algorithms from the perspective of DEs.

Unfolded ISTA and the Euler Method. Consider LISTA: $\mathbf{x}_{k+1} = g(\mathbf{x}_k, \mathbf{b}, \Theta_k) \triangleq ST(\mathbf{W}_1^k\mathbf{b} + \mathbf{W}_2^k\mathbf{x}_k, \theta_k)$, where $ST(\cdot)$ represents the soft-thresholding operator, and $\Theta_k = \left\{\mathbf{W}_1^k, \mathbf{W}_2^k, \theta_k\right\}$ are trainable parameters. [32] proposed that the LISTA can be viewed as a residual network with a residual function $r(\mathbf{x}, \mathbf{b}, \Theta) = g(\mathbf{x}, \mathbf{b}, \Theta) - \mathbf{x}$. Then, the iterations of LISTA is formulated as follows:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + r(\mathbf{x}_k, \mathbf{b}, \Theta_k), \tag{3}$$

where $r(\mathbf{x}_k, \mathbf{b}, \Theta_k)$ generalizes the expression of $-\nabla f(\mathbf{X})$ in (2). Thus (3) can be regarded as a discretized Euler method for solving (2) with initial condition $\mathbf{x}_0 = \mathbf{X}(0)$. The ISTA-based networks have made progress with the help of numerical DEs, but they can not solve problems with equality constraints. Thus, more general ADMMs need to be discussed.

Traditional ADMMs and DEs. Along with the idea of DEs, [12] proved that the continuity limits of ADMM and A-ADMM are consistent with the first-order and second-order dynamical systems, respectively. They clarify that the trajectory of the dynamical system weakly converges to a minimizer of objective functions, even in the presence of small perturbations. As for LADMM, whose iterations for solving Problem (1) are

$$\begin{cases} \mathbf{x}_{k+1} = \mathrm{Prox}_{\frac{f}{L_1}}\left\{\mathbf{x}_k - \dfrac{1}{L_1}\mathbf{A}^\top\left(\boldsymbol{\lambda}_k + \beta(\mathbf{A}\mathbf{x}_k + \mathbf{y}_k - \mathbf{b})\right)\right\}, \\ \mathbf{y}_{k+1} = \mathrm{Prox}_{\frac{g}{L_2}}\left\{\mathbf{y}_k - \dfrac{1}{L_2}\left(\boldsymbol{\lambda}_k + \beta(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_k - \mathbf{b})\right)\right\}, \\ \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + \beta(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b}), \end{cases} \tag{4}$$

where Prox is the proximal operator[1], $\boldsymbol{\lambda}$ is Lagrange multiplier, β is a penalty parameter, and $L_1, L_2 > 0$ are Lipsitz constants, [46] used the differential inclusion tool to analyze it. In addition, [11] interpreted accelerated ADMM as non-smooth dynamical systems. [49] applied the stochastic modified equation and asymptotic expansion to study the dynamics of stochastic ADMM. Furthermore, it provided a unified framework for different variants of stochastic ADMM. These ideas are all shown in Fig. 2.

Unfolded ADMMs. As mentioned above, many works show that the unfolded ADMMs outperform traditional ADMMs. [36, 43] proposed ADMM-CSNet to improve the performance of CS tasks by rewriting the ADMM procedure into a learnable network. [41] interpreted the LADMM as an end-to-end deep network and proposed an unfolded algorithm, named Differentiable Linearized ADMM (D-LADMM), for solving Problem (1). Moreover, they provided a rigorous analysis of the linear convergence. But, D-LADMM can not significantly improve the data reconstruction performance by increasing the number of network layers and even worsen. [19] and [20] proposed GPX-ADMM-Net and Deep Analysis Decoding (ADMM-DAD) network respectively, which further improved the performance on visual and speech CS tasks. The connection between these methods and traditional ADMMs is also shown in Fig. 2. However, none of the work analyzed the connection between DEs and the unfolded ADMMs. This motivates us to first draw a relatively comprehensive connection between the architectures of unfolded LADMM and the discretization methods of DEs. More importantly, we demonstrate that such a connection enables us to design new more effective unfolded networks.

Trapezoid Method in Numerical DEs. The trapezoid method approximates the integral of the function $f(\mathbf{x})$ by the trapezoid integral formula with higher precision compared with the Euler method. In unfolded algorithms, we concretize it into the following paradigm with a DE-stepsize h:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left[f(\mathbf{x}_k, \Theta_k) + f(\mathbf{x}_{k+1}, \Theta_k)\right]. \tag{5}$$

Obviously, this is an implicit method, and it needs to be estimated by a forecast-correction scheme before each update of $\mathbf{x}_{k+1}$.

3 OUR SCHEMES FOR UNFOLDED LADMMS

In this section, we find that D-LADMM [41] can be interpreted as the Euler method in numerical DEs. According to this observation, we propose a new scheme, called Trapezoid LADMM scheme, which is inspired by the trapezoid method in numerical DEs.

3.1 Euler Scheme for Unfolded LADMM

Prior works established a connection between unfolded algorithms for solving unconstrained problems and numerical DEs. And existing DE's analysis only focuses on traditional ADMMs. In this part, we take a step toward how to interpret unfolded ADMMs for solving constrained problems as numerical discretizations of DEs.

We first unfold LADMM into a unified DEs-inspired DNN, called unfolded Euler LADMM scheme:

$$\begin{cases} \mathbf{x}_{k+1} = \mathcal{F}_f\left(\mathbf{x}_k + \dfrac{h\beta_k}{\theta_k}F_k(\mathbf{x}_k)\right), \\ \mathbf{y}_{k+1} = \mathcal{G}_g\left(\mathbf{y}_k + \dfrac{h}{\eta_k}G_k(\mathbf{y}_k)\right), \\ \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + h\beta_k(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b}), \end{cases} \tag{6}$$

where $F_k(\mathbf{x}) = -\mathbf{W}^\top(\frac{\boldsymbol{\lambda}_k}{\beta_k} + \mathbf{A}\mathbf{x} + \mathbf{y}_k - \mathbf{b})$, $G_k(\mathbf{y}) = -(\frac{\boldsymbol{\lambda}_k}{\beta_k} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{y} - \mathbf{b})$, $\{\mathbf{W}, \theta_k, \eta_k, h, \beta_k\}$ are learnable parameters, and $\mathcal{F}_f$ and $\mathcal{G}_g$ are nonlinear operators. For example, if $\mathcal{F}_f$ and $\mathcal{G}_g$ are proximal operators, (6) degenerates to ELADMM and further D-LADMM [41] with $h \equiv 1$; if $\mathcal{F}_f$ and $\mathcal{G}_g$ are general non-linear operators, e.g., CNNs, (6) can be used for solving CS problems, and we call it ELADMM-Net in such case. The clear connection can be seen in Fig. 2. For theoretical explanation, Lemma 1 offers a new perspective of explaining the unfolded Euler LADMM scheme.

Lemma 1 (Unfolded Euler LADMM scheme and DEs). *Suppose that f and g are closed convex functions but maybe non-smooth, $\mathbf{A}$ has full column rank, and $\mathcal{F}_f$ and $\mathcal{G}_g$ are proximal operators. Then, we consider Problem (1) and optimal trajectory function $\{\mathbf{X}(t), \mathbf{Y}(t), \boldsymbol{\Lambda}(t)\}$. The continuous limit associated with the updates in (6) with time scale $t = kh$, corresponds to the first-order approximating DEs:*

$$\begin{cases} \nabla f_{\mu_1}(\mathbf{X}(t)) + \theta\dot{\mathbf{X}}(t) - F(\mathbf{X}(t)) = 0, \mathbf{X}(0) = \mathbf{x}_0, \\ \nabla g_{\mu_2}(\mathbf{Y}(t)) + \eta\dot{\mathbf{Y}}(t) - G(\mathbf{Y}(t)) = 0, \mathbf{Y}(0) = \mathbf{y}_0, \\ \dot{\boldsymbol{\Lambda}}(t) - \beta(\mathbf{A}\mathbf{X}(t) + \mathbf{Y}(t) - \mathbf{b}) = 0, \boldsymbol{\Lambda}(0) = \boldsymbol{\lambda}_0, \end{cases} \tag{7}$$

[1] The proximal operator of function f is $\mathrm{Prox}_{fa}(x) = \arg\min_z\{\frac{a}{2}\|z - x\|^2 + f(z)\}$.

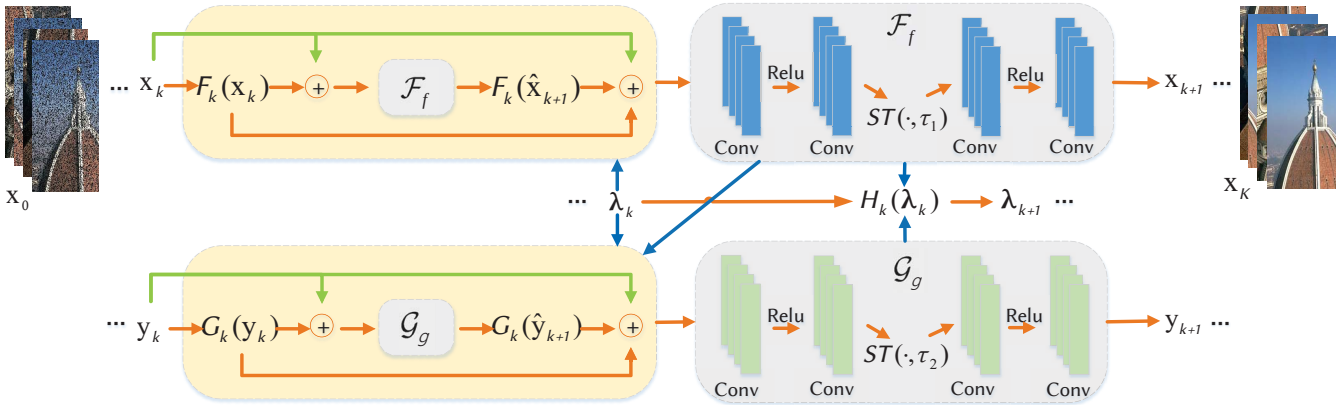

Figure 3: The k-th stage of Algorithm 1. Note that $F_k(\mathbf{x}) = -\mathbf{W}^\top(\frac{\boldsymbol{\lambda}_k}{\beta_k} + \mathbf{A}\mathbf{x} + \mathbf{y}_k - \mathbf{b})$, $G_k(\mathbf{y}) = -(\frac{\boldsymbol{\lambda}_k}{\beta_k} + \mathbf{A}\mathbf{x}_{k+1} + \mathbf{y} - \mathbf{b})$, and $H_k(\boldsymbol{\lambda}_k) = \boldsymbol{\lambda}_k + h\beta_k(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b})$. The orange line represents the independent update of each variable, the blue line represents the interaction between the variables, and most notably, the green line represents the skip connection, just like the identity mapping of the residual network, ResNet [17].

where $\theta_k \to \theta$, $\eta_k \to \eta$ in the limit $h \to 0$, $\nabla f_{\mu_1}(\cdot)$ and $\nabla g_{\mu_2}(\cdot)$ are the Moreau-Yosida approximation[2] of $\partial f(\cdot)$ and $\partial g(\cdot)$ repectively, $F(\mathbf{X}(t)) = -\mathbf{W}^\top(\boldsymbol{\Lambda}(t) + \beta(\mathbf{A}\mathbf{X}(t) + \mathbf{Y}(t) - \mathbf{b}))$, and $G(\mathbf{Y}(t)) = -(\frac{\boldsymbol{\Lambda}(t)}{\beta} + \mathbf{A}\mathbf{X}(t) + \mathbf{Y}(t) - \mathbf{b})$.

Lemma 1 indicates that the trajectory of (7) closely resembles the sequence $\{\mathbf{x}_k, \mathbf{y}_k, \boldsymbol{\lambda}_k\}$ generated by Euler LADMM (6). Specifically, one can think of the Euler LADMM scheme to solve Problem (1) as applying the Euler method to solve DEs (7) with the initial conditions $\mathbf{x}_0 = \mathbf{X}(0), \mathbf{y}_0 = \mathbf{Y}(0), \boldsymbol{\lambda}_0 = \boldsymbol{\Lambda}(0)$. For example, D-LADMM can be regarded as solving such DEs with DE-stepsize $h \equiv 1$, while our Euler LADMM scheme is capable of choosing h more flexibly. Question 1 in Section 1 has been answered.

3.2 Our Unfolded Trapezoid LADMM Scheme

We have shown above that some unfolded ADMMs can be interpreted as the Euler method in DEs, which broadens our horizons to design unfolded LADMM networks. To further explore the structural diversity of unfolded networks and improve the accuracy of the Euler LADMM scheme, we propose a new scheme, dubbed the unfolded Trapezoid LADMM scheme, and interpret its connection with DEs.

3.2.1 Implicit Trapezoid LADMM scheme and DEs. As discussed above, we can regard the Euler LADMM scheme as the Euler method to solve the first-order DEs. In this subsection, we introduce the trapezoid method into the update of $\mathbf{x}$ and $\mathbf{y}$, and propose an implicit Trapezoid LADMM scheme for Problem (1) as follows:

$$\begin{cases} \mathbf{x}_{k+1} = \mathcal{F}_f\Big(\mathbf{x}_k + \frac{h\beta_k}{2\theta_k}\big(F_k(\mathbf{x}_k) + F_k(\mathbf{x}_{k+1})\big)\Big), \\ \mathbf{y}_{k+1} = \mathcal{G}_g\Big(\mathbf{y}_k + \frac{h}{2\eta_k}\big(G_k(\mathbf{y}_k) + G_k(\mathbf{y}_{k+1})\big)\Big), \\ \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + h\beta_k(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b}). \end{cases} \tag{8}$$

[2]The Moreau-Yosida approximation of a convex function f with parameter $\mu > 0$ is defined as $f_\mu(x) := \inf_z \left\{ f(z) + \frac{1}{2\mu}\|z - x\|^2 \right\}$. For any $\mu > 0$, f_μ is a convex, continuously differentiable function [46].

Similarly, we find that the implicit Trapezoid LADMM scheme (8) also recasts as the approximating DEs (7) by Lemma 2.

Lemma 2 (Unfolded Implicit Trapezoid LADMM scheme and DEs). *The same notations and assumptions as in Lemma 1 are used. The continuous limit associated with the implicit Trapezoid LADMM scheme (8), with time scale $t = hk$, also corresponds to the first-order approximating DEs (7).*

Lemma 2 shows that the implicit Trapezoid LADMM scheme (8) for Problem (1) solves the same first-order approximating DEs (7). And from Theorems 15.1 - 15.5 in [1], we know that the existence and uniqueness of solutions of DEs (7) can be ensured under the condition of Lipschitz continuity. Therefore, we can further fairly compare the precision of our two schemes.

3.2.2 Advantages of Trapezoid LADMM scheme over Euler LADMM scheme. Theorem 1 explains that the Trapezoid LADMM scheme is more accurate than the Euler LADMM scheme under certain circumstances.

Theorem 1. *Suppose that 1) f and g are L_f-smooth and L_g-smooth respectively; 2) $\mathcal{F}_f$ and $\mathcal{G}_g$ are non-expansive mappings. Then the local and global error bound of the implicit Trapezoid LADMM scheme (8) are $O(h^3)$ and $O(h^2)$, while the local and global error bound of the Euler LADMM scheme (6) are $O(h^2)$ and $O(h)$, respectively.*

Theorem 1 shows that the lower error bound of the Trapezoid LADMM scheme can be obtained under the mild assumptions. In fact, Theorem 1 almost holds when f and g are special non-smooth functions, such as ℓ_1-norm, which will be analyzed in the Appendix. Moreover, we can know that each updated point of the Trapezoid LADMM scheme will be closer to the optimal trajectory, thus reducing the deviation from the optimal trajectory. Therefore, it can be understood to improve the convergence speed to a certain extent.

3.2.3 A Practical Scheme: Explicit Trapezoid LADMM. The implicit scheme (8) requires a solution to solve the equations w.r.t. $\mathbf{x}$ and $\mathbf{y}$, respectively, which are computationally intractable. To solve this problem, we propose an explicit Trapezoid LADMM scheme

Algorithm 1 Our Explicit Trapezoid LADMM Scheme

Input: $\mathbf{A}$, hyper-parameter α, the layer number K, the training dataset $\mathcal{D} = \{(\mathbf{b}^i, \mathbf{x}^i)\}_{i=1}^N$.

Initialize: $\mathbf{x}_0$, $\mathbf{y}_0$, $\boldsymbol{\lambda}_0$, learnable parameters $\boldsymbol{\Theta} = \{\mathbf{W}, h, \theta_k, \eta_k, \beta_k\}_{k=1}^K$, note that $a_k = \frac{\theta_k}{h}$, $c_k = \frac{\eta_k}{h}$.

Inference:

1: Choose mini-batch observations $\mathbf{b}$ of size N_b from $\mathcal{D}$, N_p is the size of each $\mathbf{x}$, and K is the number of layers;
2: **for** $k = 0, 1, \ldots, K-1$ **do**
3: $\hat{\mathbf{x}}_{k+1} = \mathcal{F}_f\big(\mathbf{x}_k + \frac{\beta_k}{a_k} F_k(\mathbf{x}_k)\big)$; //forecast step
4: $\mathbf{x}_{k+1} = \mathcal{F}_f\big(\mathbf{x}_k + \frac{\beta_k}{2a_k}[F_k(\mathbf{x}_k) + F_k(\hat{\mathbf{x}}_{k+1})]\big)$; //correction step
5: $\hat{\mathbf{y}}_{k+1} = \mathcal{G}_g\big(\mathbf{y}_k + \frac{1}{c_k} G_k(\mathbf{y}_k)\big)$; //forecast step
6: $\mathbf{y}_{k+1} = \mathcal{G}_g\big(\mathbf{y}_k + \frac{1}{2c_k}[G_k(\mathbf{y}_k) + G_k(\hat{\mathbf{y}}_{k+1})]\big)$; //correction step
7: $\boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + h\beta_k\big(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b}\big)$;
8: **end for**

Training:

9: **if** the ground truth $\mathbf{x}^*$ of observation $\mathbf{b}$ exists **then**
10: $Loss_1 = \min_{\boldsymbol{\Theta}} \frac{1}{N_b N_p} \sum_{k=1}^K \frac{k}{\sum k} \big(\|\mathbf{x}_k - \mathbf{x}^*\|^2 + \|\mathbf{y}_k - \mathbf{y}^*\|^2\big)$;
11: **else**
12: $Loss_2 = \min_{\boldsymbol{\Theta}} \frac{1}{N_b N_p} \sum_{k=1}^K \frac{k}{\sum k} \big(f(\mathbf{x}_k) + g(\mathbf{b} - \mathbf{A}\mathbf{x}_k)\big)$;
13: **end if**

Output: $\mathcal{M}(\mathcal{D}; \boldsymbol{\Theta}) = \mathbf{x}_K$.

as shown in Algorithm 1. Specifically, we design the forecast step w.r.t $\hat{\mathbf{x}}$, and then use $F_k(\hat{\mathbf{x}}_{k+1})$ in step 4 of Algorithm 1 instead of $F_k(\mathbf{x}_{k+1})$ in (8) as well as $\mathbf{y}$. Similarly, $\mathcal{F}_f$ and $\mathcal{G}_g$ are non-linear operators and vary in different applications. For example, 1) for ℓ_1-norm, if $\mathcal{F}_f$ and $\mathcal{G}_g$ are proximal operators, Algorithm 1 degenerates to TLADMM, and if $\mathcal{F}_f$ and $\mathcal{G}_g$ are general non-linear operators, e.g., CNNs, Algorithm 1 can be used for CS problems, and we call it TLADMM-Net in such case; 2) for ℓ_2-norm, the non-linear operator degenerates into an identity one or matrix inversion. Compared with the Euler LADMM scheme, although there are two more auxiliary variables $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$, the experimental performance is much better at the cost of increasing the limited time. Here we keep each learnable variable for the convenience of our proofs. It is worth noting that the learnable parameters $\{h, \theta_k, \eta_k, \beta_k\}$ are all scalars rather than vector or matrix, which reduces parameter redundancy to adapt to small-scale datasets. We design the training loss function as the weighted multi-layer loss, which alleviates the vanishing gradient problem for deep networks. When there does not exist ground truth, we utilize its model objective as a loss function.

Fig. 3 shows the k-th block architecture of the explicit Trapezoid LADMM scheme, which can be viewed as the k-th stage of inference. Compared with the methods plugging in a pre-trained DNN as denoiser [16], our networks are end-to-end trained, without relying on any pre-trained network. It is worth noting that $\mathcal{F}_f$ and $\mathcal{G}_g$ are simulated by simple CNNs in Fig. 3 for generality. Due to the introduction of the trapezoid method, each parameter update is first a forward trial procedure, followed by a further correction, so we can know that a well-trained Trapezoid LADMM scheme $\mathcal{M}(\mathcal{D}; \boldsymbol{\Theta})$ can be seen as a traditional iterative with better parameters for the distribution of the training data.

4 CONVERGENCE ANALYSIS

The errors of our schemes have been analyzed from the perspective of DEs. This section provides the convergence analysis of our explicit Trapezoid LADMM scheme from the perspective of unfolded algorithms. We will focus on the cases where $\mathcal{F}_f$ and $\mathcal{G}_g$ are the proximal operators about the f and g, respectively. We sketch the main proofs as follows.

Proof sketch: We first analyze that our explicit Trapezoid LADMM scheme converges to the implicit Trapezoid LADMM scheme. Then, we refer to D-LADMM and prove the convergence of our implicit Trapezoid LADMM scheme by Theorem 2. Finally, the linear convergence rate of our schemes can be also obtained.

Assumption 1. *Suppose that* $\frac{h}{2\theta_k} L_x < 1$, $\frac{h}{2\eta_k} L_y < 1$, *where* L_x *and* L_y *are the maximal Lipschitz constants of the sequences* $\{F_k(\cdot)\}$ *and* $\{G_k(\cdot)\}$ *with respect to* $\mathbf{x}$ *and* $\mathbf{y}$, *respectively, and* $\mathcal{F}_f$ *and* $\mathcal{G}_g$ *are non-expansive.*

Firstly, in (8), $\mathbf{x}_{k+1}$ and $\mathbf{y}_{k+1}$ can be regarded as fixed points, and we can adopt the iterative technique. We set:

$$\begin{cases} \mathbf{x}_{k+1}^{(i+1)} = \mathcal{F}_f\Big(\mathbf{x}_k + \dfrac{h\beta_k}{2\theta_k}\big(F_k(\mathbf{x}_k) + F_k(\mathbf{x}_{k+1}^{(i)})\big)\Big), \\ \mathbf{y}_{k+1}^{(i+1)} = \mathcal{G}_g\Big(\mathbf{y}_k + \dfrac{h}{2\eta_k}\big(G_k(\mathbf{y}_k) + G_k(\mathbf{y}_{k+1}^{(i)})\big)\Big), \\ \boldsymbol{\lambda}_{k+1} = \boldsymbol{\lambda}_k + h\beta_k(\mathbf{A}\mathbf{x}_{k+1} + \mathbf{y}_{k+1} - \mathbf{b}), \end{cases} \tag{9}$$

where $\mathbf{x}_{k+1}^{(0)}$ is an initial estimation. If Assumption 1 holds, (9) converges to (8) for i large enough. For ease of illustration, we set $i = 0$ and introduce the $\mathbf{x}$- and $\mathbf{y}$-step of the Euler LADMM scheme to estimate $\mathbf{x}_{k+1}^0$ and $\mathbf{y}_{k+1}^0$ in Algorithm 1. Secondly, we give the proof of Theorem 2 in the Appendix with reference to D-LADMM [41].

Theorem 2 (Convergence of implicit Trapezoid LADMM scheme). *Let the sequence* $\{\boldsymbol{\omega}_k = (\mathbf{x}_k, \mathbf{y}_k, -\boldsymbol{\lambda}_k)^\top\}$ *be generated by the implicit Trapezoid LADMM scheme (8), then there exists* $\boldsymbol{\Theta} \in \mathcal{S}(\epsilon)$ *such that* $\{\boldsymbol{\omega}_k\}$ *converges to a solution* $\boldsymbol{\omega}^*$ *of Problem (1).*

Theorem 2 plays a key role in further proving a linear convergence rate of Algorithm 1. Theorem 2 shows the proposed implicit Trapezoid LADMM scheme can asymptotically converge to the solution of Problem (1). And if h is small enough to satisfy Assumption 1, then the explicit Trapezoid LADMM scheme also converges to the same solution. Finally, following D-LADMM [41] and Theorem 2 in this paper, we still gain the linear convergence rate of the proposed algorithms. In particular, the proposed Algorithm 1 can achieve a faster convergence speed than D-LADMM in practice, which can be verified by the results in the next section.

5 EXPERIMENTS

In this section, we perform various multimedia applications to verify the effectiveness of our methods. We first test $\mathcal{F}_f$ and $\mathcal{G}_g$ as simple non-linear operators, i.e., soft-thresholding operator, in Subsections 5.1-5.3, and we call our scheme (6) and Algorithm 1 as ELADMM and TLADMM, respectively. Then we use CNN to simulate general $\mathcal{F}_f$ and $\mathcal{G}_g$ to solve the CS problems in Subsections 5.4-5.6, and we call them ELADMM-Net and TLADMM-Net, respectively. We initialize β_k and h as small values to find the next point on a larger scale and meet Assumption 1, respectively. For fair comparison, the

Table 1: Comparison of the denoising results in terms of PSNR (dB) with salt-and-pepper noise rate 10%. The best and second best results are highlighted in red and blue colors, respectively.

Algorithms	Barb	Boat	Bridge	Couple	Finger	Goldhill	Lena	Man	Mandrill	Peppers	Washsat	Zelda	Ave.	Time (s)
D-LADMM ($K=15$, [41])	32.12	31.16	26.36	31.63	31.44	32.53	35.23	31.06	24.75	34.66	34.82	37.82	31.97	0.2756
D-LADMM ($K=30$, [41])	30.55	30.23	25.67	30.78	30.11	31.46	34.50	30.12	23.24	32.12	34.13	35.62	30.71	0.5685
ELADMM ($K=15$, Ours)	32.07	31.38	26.45	31.49	31.68	32.37	35.67	30.88	23.96	34.20	34.32	37.92	31.87	0.2748
TLADMM ($K=8$, Ours)	33.36	33.29	27.94	32.98	33.65	34.39	37.75	32.94	24.58	34.13	36.43	39.30	33.39	0.2701
TLADMM ($K=15$, Ours)	34.46	33.40	28.26	33.65	34.30	34.58	39.33	33.24	25.07	34.92	37.06	40.27	34.04	0.5031

number of network layers is set to match the compared algorithms rather than a default value.

5.1 Synthetic Data

We first evaluate the effectiveness of our methods on a small-scale synthetic dataset similar to [41]. We consider the following constrained model:

$$\min_{\mathbf{x},\mathbf{y}} \alpha \|\mathbf{x}\|_1 + \|\mathbf{y}\|_1, \quad \text{s.t. } \mathbf{A}\mathbf{x} + \mathbf{y} = \mathbf{b}, \tag{10}$$

where $\mathbf{b} \in \mathbb{R}^m$ is an observation, $\mathbf{x}$ is what to recover, $\mathbf{y}$ denotes the noise to be removed and α is a hyper-parameter that balances the recovery result and denoising performance. In this experiment, we set $m=250$, $d=500$ and apply the Bernoulli sampling operator (with probability $p=0.08$ and 0.1) on both $\mathbf{x}$ and $\mathbf{y}$.

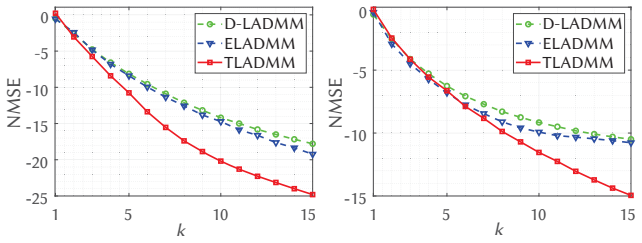

Figure 4: Comparison of the NMSE performance on synthetic datasets. Left: $p = 0.08$; Right: $p = 0.1$.

For Problem (10), our $\mathcal{F}_f$ and $\mathcal{G}_g$ both degenerate into the soft-thresholding operator, and we adopt the stochastic gradient descend (SGD) algorithm to train D-LADMM [41], our ELADMM, and our TLADMM. We choose Normalized Mean Square Error (NMSE) to measure the performance of all methods. From Fig. 4, we observe that in the case of linear convergence rate, ELADMM performs slightly better than D-LADMM due to ELADMM choosing h more flexibly. And TLADMM performs significantly better than D-LADMM and ELADMM, which verifies the conclusion in Theorem 1. Note that ISTA-based networks can not solve this problem.

5.2 Natural Image Denoising

We further utilize natural images to evaluate the denoising performance and verify the stability of our TLADMM on small-scale datasets. The training and testing datasets are the same as in [41] and the denoising performance is evaluated with Peak Signal-to-Noise Ratio (PSNR). For dictionary $\mathbf{A}$ in (10), the patch-dictionary method [42] is used to initialize it. We consider the situation where the ground truth is unknown and thus train our methods by using the following model loss function:

$$Loss_2 = \min_{\Theta} \frac{1}{N_b N_p} \sum_{k=1}^{K} \frac{k}{\sum k} \left(\alpha \|\mathbf{x}_k\|_1 + \|\mathbf{A}\mathbf{x}_k - \mathbf{b}\|_1\right). \tag{11}$$

Table 1 shows that our TLADMM with $K=15$ can improve the denoising performance by about 2.1/2.2 dB on average compared with D-LADMM/ELADMM, respectively. The underlying reason is that TLADMM is more accurate, and the points closer to the optimal trajectory of $\mathbf{x}$- and $\mathbf{y}$- subproblem can be obtained in one iteration, while D-LADMM or ELADMM only corresponds to applying the Euler method with lower precision. Moreover, taking the model loss as the training loss function imposes strict constraints on the solution, which can be seen as an alternative to the absence of ground truth. From Algorithm 1 and Table 1, although our TLADMM has two auxiliary steps, it performs better at almost the same time cost. Thus, our TLADMM can improve the denoising performance under fewer network layers or parameters. In short, these all verify that a more accurate method for solving the first-order DEs can guide the design of better unfolded LADMM networks.

5.3 Natural Image Inpainting

We also perform our ELADMM and TLADMM on image inpainting compared with ISTA-based networks and D-LADMM [41], which will show advantages over ISTA-based networks, and indicate that the introduction of the trapezoid method can improve the performance of unfolded networks in different applications.

We assume that the image is corrupted by a known mask $\mathbf{M}$ with a ratio of $r\%$ missing pixels, and the inpainting problem can be formulated as follows:

$$\min_{\mathbf{x},\mathbf{y}} \alpha\|\mathbf{x}\|_1 + \frac{1}{2}\|\mathbf{y}\|_2^2, \quad \text{s.t. } \mathbf{M}\mathbf{D}\mathbf{x} + \mathbf{y} = \mathbf{b}, \tag{12}$$

where $\mathbf{b}$ is a corrupted image block, $\mathbf{A} = \mathbf{M}\mathbf{D}$, the dictionary $\mathbf{D}$ is obtained from training clean images, and α is a hyper-parameter that balances the recovery result and sparsity. Note that $\mathcal{F}_f$ and $\mathcal{G}_g$ degenerate into a soft-thresholding operator and a constant transformation $\frac{\beta_k}{1+\beta_k}\mathcal{I}(\cdot)$, respectively. The learned solver $\mathcal{M}(\mathcal{D};\Theta)$ has a natural advantage - avoiding $\mathcal{O}(d^3)$ computational complexity due to matrix-matrix multiplications and matrix inversions in existing ADMM-based unfolded networks.

We set the number of the layers $K=20$ in all the networks, and the PSNR results are listed in Table 2. We observe that ELADMM

Table 2: Comparison of image inpainting results in terms of PSNR (dB) on the dataset Set11 with 50% missing pixels.

Algorithms	Barb	Boat	House	Lena	Peppers	C.man	Flinstones	Finger	Parrot	Foreman	Monarch	Ave. (dB)
LFISTA (ICLR2017, [29])	26.13	30.02	32.55	32.65	28.84	27.24	26.53	28.35	31.03	30.65	28.17	29.29
GLISTA (ICLR2019, [40])	25.52	28.76	31.04	31.18	27.64	25.85	24.32	26.35	29.44	29.24	26.07	27.76
D-LADMM (ICML2019, [41])	26.51	30.91	34.11	34.31	29.84	27.44	27.04	29.95	32.44	31.84	28.82	30.29
ELISTA (AAAI2021, [24])	26.75	30.55	32.84	32.94	29.64	27.68	24.40	28.52	30.95	30.85	26.45	29.23
ELADMM (Ours)	26.54	30.84	34.15	34.55	29.75	27.48	27.11	29.85	32.62	31.95	28.99	30.35
TLADMM (Ours)	27.08	31.58	34.75	35.08	30.09	27.94	27.69	30.38	33.61	32.45	29.45	30.91

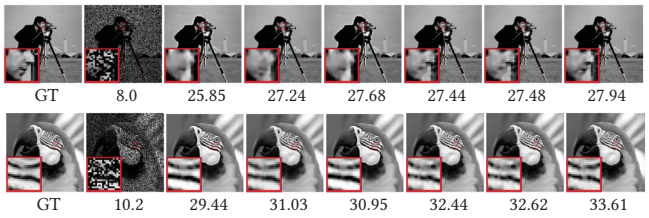

Figure 5: Comparison of visual results and PSNR (dB) for the image inpainting with 50% missing pixels. From left to right: ground truth (GT), corrupted image, and the results of GLISTA [40], LFISTA [29], ELISTA [24], D-LADMM [41], ELADMM (Ours) and TLADMM (Ours).

and TLADMM consistently outperform the ISTA-based networks, LFISTA, GLISTA, and ELISTA. In addition, it is clear that the average PSNR of our TLADMM is about 0.6 dB higher than that of D-LADMM. Finally, Fig. 5 shows the visual performance of different methods on Cameraman (called C.man) and Parrot. It can be seen that our TLADMM can restore the highest image quality.

5.4 Natural Image Compressive Sensing

Our methods can also be easily extended to solve the CS inverse problem. Here we will show the advantages of our methods when $\mathcal{F}_f$ and $\mathcal{G}_g$ are more general non-linear operators. By introducing an auxiliary variable $\mathbf{y}$, the CS model can be expressed as follows:

$$\min_{\mathbf{x},\mathbf{y}} \frac{1}{2}\|\mathbf{c}-\mathbf{\Phi x}\|^2+\alpha\|\mathbf{\Psi y}\|_1,\ \text{s.t. } \mathbf{x}=\mathbf{y}, \tag{13}$$

where $\mathbf{\Phi} \in \mathbb{R}^{n\times d}$ is an under-sampling matrix, $\mathbf{x}$ is vectorized image, the CS measurement of $\mathbf{x}$ is denoted by $\mathbf{c}$, $\mathbf{b}=\mathbf{0}$, $\mathbf{A}=\mathbf{I}$, and $\mathbf{\Psi}$ denotes a transformation matrix for a filtering operation, e.g., Discrete Cosine Transform and Discrete Wavelet Transform. Formally, we replace $\mathbf{\Psi}$ with a non-linear transformation $\mathcal{T}(\cdot)$ to sparsify natural images, where $\mathcal{T}(\cdot)$ adopts a simple CNN as in [47]. And the non-linear operator $\mathcal{F}_f(\mathbf{z})$ degenerates into $\mathbf{Q}_k(\mathbf{z}+\mathbf{c})$, where $\mathbf{Q}_k=(\mathbf{\Phi}^\top\mathbf{\Phi}+\beta_k\mathbf{I})^{-1}$, and $\mathcal{G}_g(\cdot)=\widetilde{\mathcal{T}}\big(ST(\mathcal{T}(\cdot))\big)$, where $ST(\cdot)$ denotes the soft-thresholding operator. In fact, we also avoid matrix inversions in existing ADMM-based unfolded networks by linearizing quadratic terms, so the computational complexity of our methods can be further reduced.

For fair comparison, we select the same Train400 dataset as in [44] containing large numbers of various scenarios to train our unfolded networks. We used the Adam optimizer and all the methods were trained up to 400 epochs. As for testing, we utilize two widely-used datasets: Set11 and BSD68, and reconstructed performance is evaluated by computing PSNR. We refer to the partial results in [44] for some compared algorithms.

As shown in Table 3, our TLADMM-Net with $K=10$ achieves at least 0.92/0.72 dB PSNR improvement over other algorithms on the BSD68/Set11 datasets on average, which fully verifies the effectiveness of our Trapezoid scheme and reduction of the number of layers. If we set $K=20$ to train our TLADMM-Net, recovery performance will be further improved. Compared with several recently proposed networks, the reconstruction time of our schemes is competitive, and the number of network parameters is reduced. Fig. 1 shows the details of all the algorithms for PSNR, reconstruction time, and the number of network parameters on BSD68 at CS ratio $\gamma=30\%$.

5.5 Speech Data Compressive Sensing

Moreover, we also consider other multimedia applications, speech CS, whose model is the same as (13), but $\mathbf{x}$ represents vectorized speech. We train our methods on two speech datasets, SpeechCommands and TIMIT, and we take 70% of the datasets for training and 30% for testing. We perform the same preprocessing as ADMM-DAD [20] on the raw speech data and execute the codes of ISTA-Net$^+$ [47] and ADMM-DAD as baselines.

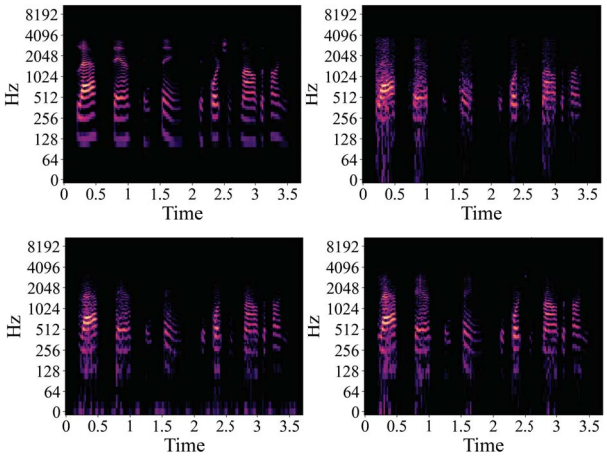

Figure 6: Comparison of the visual results for the speech CS task at $\gamma=40\%$ on TIMIT. Upper left, upper right, lower left and lower right: the spectrograms of ground truth, and the results of ADMM-DAD [20], ELADMM-Net (Ours) and TLADMM-Net (Ours).

Table 3: Comparison of performance for image compressive sensing in terms of PSNR (dB) under different CS ratios γ = 10%, 20%, 30%, 40%, 50% on the BSD68 and Set11 datasets. As we can see, our networks achieve the best results under all CS ratios.

Algorithms	Dataset: BSD68						Dataset: Set11					
	10%	20%	30%	40%	50%	Avg.	10%	20%	30%	40%	50%	Avg.
ISTA-Net$^+$ (CVPR2018, [47])	25.24	28.00	30.20	32.10	33.93	29.89	26.57	30.85	33.74	36.05	38.05	33.05
DPDNN (TPAMI2019, [8])	24.81	27.28	29.22	30.99	32.74	29.01	26.09	29.75	32.37	34.69	36.83	31.95
GDN (TCI2019, [13])	25.19	27.95	29.88	32.07	34.09	29.84	26.03	30.16	32.95	35.25	37.60	32.40
CSCNet (CVPR2019, [34])	27.28	29.01	31.87	33.86	35.77	31.56	28.48	31.95	34.62	36.92	39.01	34.20
DPA-Net (TIP2020, [37])	25.33	-	29.58	-	-	-	27.66	-	33.60	-	-	-
MAC-Net (ECCV2020, [5])	25.70	28.23	30.10	31.89	33.37	29.86	27.92	31.54	33.87	36.18	37.76	33.45
COAST (TIP2021, [45])	26.28	29.00	32.10	32.93	34.74	31.01	28.69	32.54	35.04	37.13	38.94	34.47
ISTA-Net^{++} (ICME2021, [44])	26.25	29.00	31.10	33.00	34.85	30.84	28.34	32.33	34.86	36.94	38.73	34.24
GPX-ADMM-Net (2021, [19])	25.30	27.79	29.32	31.99	33.25	29.53	27.46	31.36	33.85	36.28	38.32	33.45
ELADMM-Net ($K=20$, Ours)	27.01	29.53	32.01	33.89	35.82	31.65	28.34	32.51	34.72	37.32	38.71	34.32
TLADMM-Net ($K=10$, Ours)	27.76	30.38	32.68	34.78	36.82	32.48	28.95	32.81	35.73	38.18	40.32	35.19
TLADMM-Net ($K=20$, Ours)	27.97	30.59	32.96	35.05	37.15	32.74	29.21	33.20	36.06	38.58	40.83	35.57

We choose a column orthogonal measurement matrix $\mathbf{\Phi}$ to downsample raw speech data. We use the Adam optimizer and train 100 epochs for all the methods with $K = 10$. We set $Loss_1$ as the loss function, and use MSE $= \frac{1}{q}\sum_{i=1}^{q} \|\mathcal{M}(\mathbf{b}^i;\Theta) - \mathbf{x}^i\|^2$ as a test criterion, where q is the number of test samples. The recovered performance is shown in Table 4. It can be seen that the MSE of our ELADMM-Net and TLADMM-Net is always lower than the baselines. Furthermore, we extract the spectrograms of an example test raw audio file in TIMIT as shown in Fig. 6. It can be clearly seen that our ELADMM-Net and TLADMM-Net distinguish more frequencies than ADMM-DAD, and TLADMM-Net further removes the noise of ELADMM-Net.

Table 4: Comparison of the test MSE results under the CS ratios γ = 25%, 40% on the speech datasets.

Algorithms	SpeechCommands		TIMIT	
	25%	40%	25%	40%
ISTA-Net$^+$ [47]	0.58×10^{-2}	0.46×10^{-2}	0.22×10^{-3}	0.20×10^{-3}
ADMM-DAD [20]	0.25×10^{-2}	0.13×10^{-2}	0.79×10^{-4}	0.42×10^{-4}
ELADMM-Net (Ours)	0.17×10^{-2}	0.78×10^{-3}	0.68×10^{-4}	0.40×10^{-4}
TLADMM-Net (Ours)	0.16×10^{-2}	0.80×10^{-3}	0.51×10^{-4}	0.22×10^{-4}

5.6 MRI Compressive Sensing

To demonstrate the generalization ability of our schemes, we also extend our schemes to the MRI reconstruction. Following previous practices, we set $\mathbf{\Phi} = \mathbf{PF}$ for Problem (13), where $\mathbf{P}$ is a mask and $\mathbf{F}$ is the discrete Fourier transform. The sampling pattern is the commonly used pseudo radial sampling. We train and test our networks on the same brain MRI dataset as ADMM-Net [36] and ISTA-Net$^+$ [47]. Our networks are separately trained for each sampling ratio with $K = 10$ and the reconstruction results are shown in Table 5. It can be seen that our networks outperform ADMM-Net in terms of both PSNR and runtime, and our TLADMM-Net outperforms our ELADMM-Net in the case of increased limited time cost. Note that our networks are comparable to the reconstruction result of ISTA-Net$^+$, but the runtime is increased, which is mainly due to the complexity of ADMM itself. Compared with ADMM-Net, the reconstruction time of TLADMM-Net is reduced by two-thirds because ADMM-Net requires matrix inversions.

Table 5: Comparison of test PSNR (dB) and runtime (s) for MRI CS with CS ratios γ = 20%, 30%, 40%, 50% on the brain dataset. The last column is average GPU time for reconstructing a 256×256 image.

Algorithms	20%	30%	40%	50%	Time
ADMM-Net [36]	37.17	39.84	41.56	43.00	0.046
ISTA-Net [47]	38.30	40.52	42.12	43.60	0.006
ISTA-Net$^+$ [47]	38.73	40.89	42.52	44.09	0.007
ELADMM-Net (Ours)	38.31	40.21	42.12	43.66	0.007
TLADMM-Net (Ours)	38.72	40.81	42.57	44.15	0.013

6 CONCLUSIONS AND FUTURE WORK

In this paper, a novel scheme of designing unfolded LADMM networks was proposed. The connection between existing unfolded algorithms and DEs was firstly analyzed, and then the trapezoid method was introduced into unfolded LADMMs to obtain a novel Trapezoid LADMM scheme. Furthermore, we analyzed the error bound and convergence of the proposed Trapezoid LADMM scheme. Extensive experimental results verified the Trapezoid LADMM scheme superior to the existing methods, which provides strong support for "higher precision numerical methods can derive better unfolded ADMMs". We anticipate that these results will provide new insights on understanding unfolded ADMM networks for solving machine learning problems. In the future, we will further explore the connections between other higher-order numerical methods (e.g., the multi-step method) and unfolded ADMMs.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China (Nos. 61876221, 61876220, 61976164 and 61836009), and Natural Science Basic Research Program of Shaanxi (Program No. 2022GY-061).

REFERENCES

[1] Ravi P Agarwal and Donal O'Regan. 2008. An Introduction to Ordinary Differential Equations. (2008).

[2] Anthony Bloch. 1994. *Hamiltonian and gradient flows, algorithms and control*. Vol. 3. American Mathematical Soc.

[3] Stephen Boyd, Neal Parikh, and Eric Chu. 2011. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc.

[4] AA Brown and Michael C Bartholomew-Biggs. 1989. Some effective methods for unconstrained optimization based on the solution of systems of ordinary differential equations. *Journal of Optimization Theory and Applications* 62, 2 (1989), 211–224.

[5] Jiwei Chen, Yubao Sun, Qingshan Liu, and Rui Huang. 2020. Learning memory augmented cascading network for compressed sensing of images. In *European Conference on Computer Vision*. Springer, 513–529.

[6] Xiaohan Chen, Jialin Liu, Zhangyang Wang, and Wotao Yin. 2018. Theoretical Linear Convergence of Unfolded ISTA and Its Practical Weights and Thresholds. *Advances in neural information processing systems* (2018).

[7] Wei Deng, Ming-Jun Lai, Zhimin Peng, and Wotao Yin. 2017. Parallel multi-block ADMM with o (1/k) convergence. *Journal of Scientific Computing* 71, 2 (2017), 712–736.

[8] Weisheng Dong, Peiyao Wang, Wotao Yin, Guangming Shi, Fangfang Wu, and Xiaotong Lu. 2018. Denoising prior driven deep neural network for image restoration. *IEEE transactions on pattern analysis and machine intelligence* 41, 10 (2018), 2305–2318.

[9] Yuling Fan, Hongjian Wang, Hartmut Gemmeke, Torsten Hopp, and Juergen Hesser. 2022. Model-data-driven image reconstruction with neural networks for ultrasound computed tomography breast imaging. *Neurocomputing* 467 (2022), 10–21.

[10] Rita Fermanian, Mikael Le Pendu, and Christine Guillemot. 2021. Regularizing the Deep Image Prior with a Learned Denoiser for Linear Inverse Problems. In *IEEE Multimedia Siganl Processing Workshop (MMSP)*.

[11] G França, DP Robinson, and R Vidal. 1808. A nonsmooth dynamical systems perspective on accelerated extensions of ADMM (2018). *Preprint. arXiv* (1808).

[12] Guilherme Franca, Daniel Robinson, and Rene Vidal. 2018. ADMM and accelerated ADMM as continuous dynamical systems. In *International Conference on Machine Learning*. PMLR, 1559–1567.

[13] Davis Gilton, Greg Ongie, and Rebecca Willett. 2019. Neumann networks for linear inverse problems in imaging. *IEEE Transactions on Computational Imaging* 6 (2019), 328–343.

[14] Tom Goldstein, Brendan O'Donoghue, Simon Setzer, and Richard Baraniuk. 2014. Fast alternating direction optimization methods. *SIAM Journal on Imaging Sciences* 7, 3 (2014), 1588–1623.

[15] Karol Gregor and Yann LeCun. 2010. Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on international conference on machine learning*. 399–406.

[16] Harshit Gupta, Kyong Hwan Jin, Ha Q Nguyen, Michael T McCann, and Michael Unser. 2018. CNN-based projected gradient descent for consistent CT image reconstruction. *IEEE transactions on medical imaging* 37, 6 (2018), 1440–1453.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[18] Uwe Helmke and John B Moore. 2012. *Optimization and dynamical systems*. Springer Science & Business Media.

[19] Shih-Wei Hu, Gang-Xuan Lin, and Chun-Shien Lu. 2021. GPX-ADMM-Net: ADMM-based neural network with generalized proximal operator. In *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2055–2059.

[20] Vasiliki Kouni, Georgios Paraskevopoulos, Holger Rauhut, and George C Alexandropoulos. 2021. ADMM-DAD net: a deep unfolding network for analysis compressed sensing. *arXiv preprint arXiv:2110.06986* (2021).

[21] Honggui Li and Maria Trocan. 2018. Sparse solution to inverse problem of nonlinear dimensionality reduction. In *International Conference on Multimedia and Network Information System*. Springer, 322–331.

[22] Ke Li and Jitendra Malik. 2016. Learning to optimize. *arXiv preprint arXiv:1606.01885* (2016).

[23] Xiaoyong Li, Xueru Bai, Yujie Zhang, and Feng Zhou. 2022. High-Resolution ISAR Imaging Based on Plug-and-Play 2D ADMM-Net. *Remote Sensing* 14, 4 (2022), 901.

[24] Yangyang Li, Lin Kong, Fanhua Shang, Yuanyuan Liu, Hongying Liu, and Zhouchen Lin. 2021. Learned extragradient ISTA with interpretable residual structures for sparse coding. In *Proc. AAAI Conf. Artif. Intell.*

[25] Zhouchen Lin, Risheng Liu, and Zhixun Su. 2011. Linearized alternating direction method with adaptive penalty for low-rank representation. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*. 612–620.

[26] Jialin Liu and Xiaohan Chen. 2019. ALISTA: Analytic weights are as good as learned weights in LISTA. In *International Conference on Learning Representations (ICLR)*.

[27] Canyi Lu, Huan Li, Zhouchen Lin, and Shuicheng Yan. 2016. Fast proximal linearized alternating direction method of multiplier with parallel splitting. In *Thirtieth AAAI Conference on Artificial Intelligence*.

[28] Vishal Monga, Yuelong Li, and Yonina C Eldar. 2021. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine* 38, 2 (2021), 18–44.

[29] Thomas Moreau and Joan Bruna. 2019. Understanding trainable sparse coding via matrix factorization. In *5th International Conference on Learning Representations, ICLR 2017*.

[30] Hua Ouyang, Niao He, Long Tran, and Alexander Gray. 2013. Stochastic alternating direction method of multipliers. In *International Conference on Machine Learning*. PMLR, 80–88.

[31] Juan Marcos Ramírez, José Ignacio Martínez Torre, and Henry Arguello Fuentes. 2021. LADMM-Net: An Unrolled Deep Network For Spectral Image Fusion From Compressive Data. *arXiv preprint arXiv:2103.00940* (2021).

[32] Michael E Sander, Pierre Ablin, Mathieu Blondel, and Gabriel Peyré. 2021. Momentum residual neural networks. In *International Conference on Machine Learning*. PMLR, 9276–9287.

[33] Johannes Schropp and I Singer. 2000. A dynamical systems approach to constrained minimization. *Numerical functional analysis and optimization* 21, 3-4 (2000), 537–551.

[34] Wuzhen Shi, Feng Jiang, Shaohui Liu, and Debin Zhao. 2019. Scalable convolutional neural network for image compressed sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12290–12299.

[35] Weijie Su, Stephen Boyd, and Emmanuel J Candes. 2016. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *The Journal of Machine Learning Research* 17, 1 (2016), 5312–5354.

[36] Jian Sun, Huibin Li, Zongben Xu, et al. 2016. Deep ADMM-Net for compressive sensing MRI. *Advances in neural information processing systems* 29 (2016).

[37] Yubao Sun, Jiwei Chen, Qingshan Liu, Bo Liu, and Guodong Guo. 2020. Dual-path attention network for compressed sensing image reconstruction. *IEEE Transactions on Image Processing* 29 (2020), 9482–9495.

[38] Udaya SKP Thanthrige, Peter Jung, and Aydin Sezgin. 2021. Deep Unfolding of Iteratively Reweighted ADMM for Wireless RF Sensing. *arXiv preprint arXiv:2106.03686* (2021).

[39] Qi Wei, Kai Fan, Wenlin Wang, Tianhang Zheng, Chakraborty Amit, Katherine Heller, Changyou Chen, and Kui Ren. 2019. InverseNet: Solving Inverse Problems of Multimedia Data with Splitting Networks. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1324–1329.

[40] Kailun Wu, Yiwen Guo, Ziang Li, and Changshui Zhang. 2019. Sparse coding with gated learned ISTA. In *International Conference on Learning Representations*.

[41] Xingyu Xie, Jianlong Wu, Guangcan Liu, Zhisheng Zhong, and Zhouchen Lin. 2019. Differentiable linearized admm. In *International Conference on Machine Learning*. PMLR, 6902–6911.

[42] Yangyang Xu and Wotao Yin. 2016. A fast patch-dictionary method for whole image recovery. *Inverse Problems and Imaging* 10, 2 (2016), 563–583.

[43] Yan Yang, Jian Sun, Huibin Li, and Zongben Xu. 2020. ADMM-CSNet: A Deep Learning Approach for Image Compressive Sensing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42, 3 (2020), 521–538. https://doi.org/10.1109/TPAMI.2018.2883941

[44] Di You, Jingfen Xie, and Jian Zhang. 2021. ISTA-Net++: flexible deep unfolding network for compressive sensing. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

[45] Di You, Jian Zhang, Jingfen Xie, Bin Chen, and Siwei Ma. 2021. COAST: COntrollable arbitrary-sampling NeTwork for compressive sensing. *IEEE Transactions on Image Processing* 30 (2021), 6066–6080.

[46] Huizhuo Yuan, Yuren Zhou, Chris Junchi Li, and Qingyun Sun. 2019. Differential inclusions for modeling nonsmooth ADMM variants: A continuous limit theory. In *International Conference on Machine Learning*. PMLR, 7232–7241.

[47] Jian Zhang and Bernard Ghanem. 2018. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1828–1837.

[48] Shuai Zheng and James T Kwok. 2016. Fast-and-Light Stochastic ADMM.. In *IJCAI*. 2407–2613.

[49] Xiang Zhou, Huizhuo Yuan, Chris Junchi Li, and Qingyun Sun. 2020. Stochastic Modified Equations for Continuous Limit of Stochastic ADMM. *arXiv preprint arXiv:2003.03532* (2020).