Dual Alternating Direction Method of Multipliers for Inverse Imaging

Li Song, Zhou Ge, and Edmund Y. Lam, *Fellow, IEEE*

***Abstract*— Inverse imaging covers a wide range of imaging applications, including super-resolution, deblurring, and compressive sensing. We propose a novel scheme to solve such problems by combining duality and the alternating direction method of multipliers (ADMM). In addition to a conventional ADMM process, we introduce a second one that solves the dual problem to find the estimated nontrivial lower bound of the objective function, and the related iteration results are used in turn to guide the primal iterations. We call this D-ADMM, and show that it converges to the global minimum when the regularization function is convex and the optimization problem has at least one optimizer. Furthermore, we show how the scheme can give rise to two specific algorithms, called D-ADMM-L2 and D-ADMM-TV, by having different regularization functions. We compare D-ADMM-TV with other methods on image super-resolution and demonstrate comparable or occasionally slightly better quality results. This paves the way of incorporating advanced operators and strategies designed for basic ADMM into the D-ADMM method as well to further improve the performances of those methods.**

***Index Terms*— Inverse imaging, dual problem, D-ADMM, image super-resolution.**

I. Introduction

INVERSE imaging [1]–[3] is a core topic in computational imaging and includes a wide array of problems, such as deblurring, super-resolution, magnetic resonance imaging (MRI) and compressive imaging, among many others. Generally, a linear imaging model can be represented as

$$\boldsymbol{y} = A\boldsymbol{x} + \boldsymbol{e}, \tag{1}$$

where A is the system matrix [4], $\boldsymbol{x}$ is the object, $\boldsymbol{y}$ is the image, and $\boldsymbol{e}$ denotes additive noise. Inverse imaging aims at estimating $\hat{\boldsymbol{x}}$ from $\boldsymbol{y}$, where we often assume that both are in the discrete domain, with $\hat{\boldsymbol{x}} \in \mathbb{R}^N$ and $\boldsymbol{y} \in \mathbb{R}^M$. In this paper, we focus on the synthesis pursuit problem, where the system matrix A and the measurement $\boldsymbol{y}$ are known.

A broad range of methods have been proposed for these imaging problems. Block matching and 3D filtering (BM3D) [5] and non-local means (NLM) [6] are widely-used for image denoising. Neighbor embedding [7] and sparse representation [8], [9] have good performances on image super-resolution. Gradient-based methods [10] and edge analysis [11] have been designed for image deblurring. With the advent of deep learning, many different kinds of neural networks [12], [13] are also becoming popular approaches for inverse imaging. While these end-to-end neural networks are designed and trained to tackle the required tasks automatically, it is often not clear what contributes to the superior performance. An important technique related to deep learning is the unrolled network [14]–[16], which seeks to incorporate classical knowledge into deep learning approaches. However, data-driven methods need image sets for training and testing, and are not always practical in some situations.

Generally, inverse imaging involves solving the unconstrained optimization problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} h(\boldsymbol{x}) + \beta\mathcal{R}(\boldsymbol{x}), \tag{2}$$

where β is a constant, $h(\boldsymbol{x})$ is the data consistency term evaluating how $\boldsymbol{x}$ matches with the observation $\boldsymbol{y}$, and $\mathcal{R}(\boldsymbol{x})$ is a regularization term [17] that incorporates prior knowledge (such as sparsity) of the object $\boldsymbol{x}$. Iterative methods, such as alternating direction method of multipliers (ADMM) [18], show excellent performances for this kind of optimization problems. ADMM makes use of auxiliary variables to separate the smooth (usually $h(\boldsymbol{x})$) and non-smooth (usually $\mathcal{R}(\boldsymbol{x})$) parts in the optimization process. Several variants of the method are developed recently, such as consensus ADMM [19], proximal ADMM [20], stochastic ADMM [21], etc.

This paper describes another development of ADMM, which makes use of the dual problem to guide the primal ADMM iteration process. The main contributions are:

- We propose a methodology for tackling inverse imaging, which is called the dual alternating direction method of multipliers (D-ADMM). We employ ADMM to solve the dual problem, and the solutions of its iterations are then used to guide the ADMM iterations of the primal problem to arrive at the global minimum given that the regularization function is convex, and the optimization problem has at least one optimizer (which we assume throughout this paper). Compared with ADMM, the dual iterations can bring the estimated nontrivial lower bound of the objective function to the primal iterations, leading to

Manuscript received February 10, 2021; revised May 4, 2021, June 8, 2021, August 12, 2021, and December 9, 2021; accepted April 6, 2022. Date of publication April 21, 2022; date of current version April 27, 2022. This work was supported in part by the Research Grants Council of Hong Kong [General Research Fund (GRF)] under Grant 17200019, Grant 17201620, and Grant 17200321; and in part by the University of Hong Kong under Grant 104005864. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Alessandro Foi. *(Corresponding author: Edmund Y. Lam.)*

The authors are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong, SAR, China (e-mail: songli@eee.hku.hk; gezhou@eee.hku.hk; elam@eee.hku.hk).

Digital Object Identifier 10.1109/TIP.2022.3167915

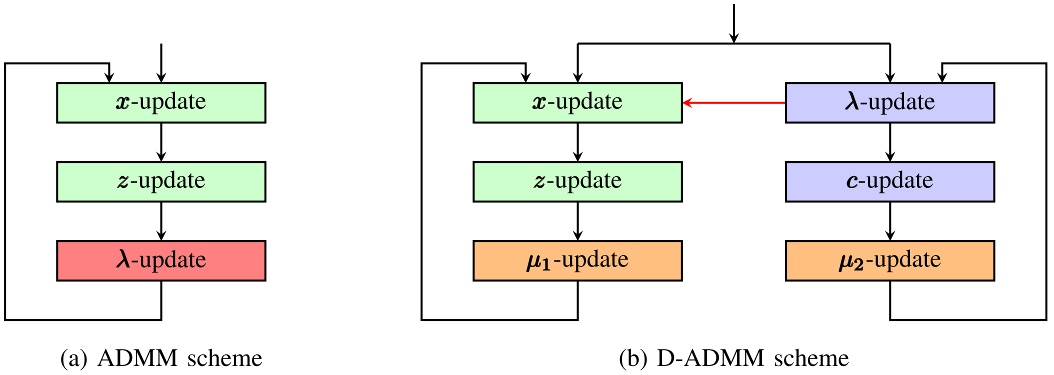


Fig. 1. Conventional ADMM and the proposed D-ADMM schemes. (a) Conventional ADMM determines the value of $\boldsymbol{\lambda}$ from primal variables $\boldsymbol{x}$ and $\boldsymbol{z}$. (b) Our proposed D-ADMM involves a second ADMM to solve the dual problem and shares the iteration results of $\boldsymbol{\lambda}$ with the primal iterations.

comparable or, at times, somewhat better quality images with fewer number of iterations required than ADMM in the non-strongly convex case, given the same error tolerance.

- We give a global convergence proof and a fixed point convergence proof of the D-ADMM scheme.
- As D-ADMM is applicable for various kinds of regularization functions, we show in particular two algorithms, using ℓ_2- and TV-regularization functions. Their iteration steps are derived and presented accordingly. Derivations of D-ADMM-TV with a general proximal operator also show the potential of incorporating learned denoisers into D-ADMM.

The rest of this paper is organized as follows. Section II gives an overview of the conventional ADMM, and some related recent improvements. Section III then discusses the mathematical derivations of the proposed D-ADMM scheme. Two specific algorithms of the D-ADMM scheme are also included. Section IV shows image super-resolution experiments and one numerical example, where computational results are presented and analyzed. Finally, Section V gives some conclusions, and a complete convergence proof is given in the Appendix.

II. Related Work

A. Foundation of ADMM

In many practical applications, the linear inverse imaging problem is ill-posed. To estimate $\boldsymbol{x}$ from the observation $\boldsymbol{y}$ and a known system matrix A, a typical method is to solve

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \mathcal{R}(\boldsymbol{x}) + \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{y}\|_2^2, \tag{3}$$

where $\|A\boldsymbol{x} - \boldsymbol{y}\|_2^2$ aims to enforce data fidelity, and $\mathcal{R}(\boldsymbol{x})$ is the regularization function. To solve this problem, the variable splitting method [22] can be used to form the constrained optimization problem

$$\begin{aligned} &\underset{\boldsymbol{x},\boldsymbol{z}}{\text{minimize}} \;\; \mathcal{R}(\boldsymbol{z}) + \frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{y}\|_2^2 \\ &\text{subject to } \boldsymbol{x} - \boldsymbol{z} = 0, \end{aligned} \tag{4}$$

where the augmented Lagrangian function is

$$L_\rho(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\lambda}) = \frac{1}{2}\|A\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \mathcal{R}(\boldsymbol{z}) + \boldsymbol{\lambda}^T(\boldsymbol{x}-\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2, \tag{5}$$

and ρ is the penalty parameter. We can also form an equivalent scaled form

$$\tilde{L}_\rho(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\lambda}) = \frac{1}{2}\|A\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \mathcal{R}(\boldsymbol{z}) + \frac{\rho}{2}\|\boldsymbol{x}-\boldsymbol{z}+\boldsymbol{\mu}\|_2^2, \tag{6}$$

where $\boldsymbol{\mu} = \boldsymbol{\lambda}/\rho$ is the scaled Lagrange multiplier, also called the dual variable. The standard ADMM iterations related to Eq. 5 are

$$\boldsymbol{x}^{(k+1)} = (A^TA + \rho I)^{-1}(A^T\boldsymbol{y} - \boldsymbol{\lambda}^{(k)} + \rho\boldsymbol{z}^{(k)}) \tag{7}$$

$$\boldsymbol{z}^{(k+1)} = \text{prox}_{\frac{\mathcal{R}}{\rho}}\left(\boldsymbol{x}^{(k+1)} + \frac{\boldsymbol{\lambda}^{(k)}}{\rho}\right) \tag{8}$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \rho(\boldsymbol{x}^{(k+1)} - \boldsymbol{z}^{(k+1)}), \tag{9}$$

where prox is the proximal operator [23]. This operator is defined to obtain the solution of a small optimization problem

$$\text{prox}_{\tau f}(\boldsymbol{z}) = \arg\min_{\boldsymbol{x}}\left(f(\boldsymbol{x}) + \frac{1}{2\tau}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2\right), \tag{10}$$

which often has a closed form or can be solved very quickly with simple specialized methods.

As shown in Eq. 9, in the conventional ADMM, the value of the dual variable $\boldsymbol{\lambda}$ is derived by a gradient ascent update step. This means that the role of the dual variable $\boldsymbol{\lambda}$ is more like a "follower". This is illustrated in Fig. 1(a), which shows that $\boldsymbol{\lambda}$ is totally determined by the values of other primal variables $\boldsymbol{x}$ and $\boldsymbol{z}$. If the primal variables converge, $\boldsymbol{\lambda}$ is also regarded to converge. However, it is shown in [18] that when $\mathcal{R}$ is only subdifferentiable, the convergence of $\boldsymbol{\lambda}$ may not be monotone. Furthermore, according to the duality theory, the converged value of the dual variable should be the solution of the dual problem. The dual iterations, related to the estimated nontrivial lower bound of the objective function [24], may be different from the iterations of $\boldsymbol{\lambda}$ in Eq. 9. Filling in this gap during the iterations can lead to better iterative results for ADMM.

B. Improvements to ADMM

The plug-and-play approach [25] has become popular for the iterative optimization methods. A commonly-used strategy is based on the observation that $\mathcal{R}(\cdot)$ is identical to the regularization function in image denoising, and therefore it is possible to use an external denoiser $\mathcal{D}_\sigma(\cdot)$ to deal with this step. The plug-and-play ADMM (PnP-ADMM) [26]–[28]

is a combination of this idea and ADMM, which can take advantage of the external bounded denoisers. In this method, Eq. 7 is regarded as a direct inversion step since the system matrix A is included. Eq. 8 is a correction step of $\boldsymbol{x}^{(k+1)}$ considering the regularization term $\mathcal{R}(\cdot)$, in order to make the inverse imaging result more realistic. For example, when $\mathcal{R}(\cdot)$ is specified as total variation (TV) regularization [29], the $\boldsymbol{z}$-update step is a standard TV denoising problem based on the "noisy" image $\boldsymbol{x}^{(k+1)}$. Hence, proper denoising algorithms or networks, denoted as $\mathcal{D}_\sigma(\cdot)$, can be used in lieu of Eq. 8 as

$$\boldsymbol{z}^{(k+1)} = \mathcal{D}_\sigma\left(\boldsymbol{x}^{(k+1)} + \frac{\boldsymbol{\lambda}^{(k)}}{\rho}\right). \tag{11}$$

Specifically, this leads to a new trend for combining iterative methods such as ADMM with learning-based methods. Comparing Eq. 8 with Eq. 11, we know that the denoiser is related to the prior knowledge embedded in the regularization function. Nowadays, learning-based image denoisers are becoming popular due to their excellent performances [30]. These learned denoisers, also regarded as deep denoising priors, can be plugged into the iterations directly. Zhang *et al.* [31] trained a set of efficient image denoisers and then integrated them into iterative methods directly based on the variable splitting strategy. Romano *et al.* [32] proposed an adaptive Laplacian-based regularization function mathematically, which can make the objective function better defined. There are also some specifically-designed networks for plug-and-play, such as the multiple self-similarity net (MSSN) [33].

We remark here that some researchers also tackle non-convex problems using ADMM since such problems are more general. Yet, it is not easy to find the minimizer in each step. Latorre *et al.* proposed a gradient descent-based ADMM [34]. This method updates the primal variables with the gradient instead of tackling the more challenging non-convex minimization. The $\boldsymbol{x}$-update step is a combination of gradient descent and projection on the generative priors

$$\boldsymbol{x}^{(k+1)} = \mathcal{P}\left(\boldsymbol{x}^{(k)} - \alpha^{(k)}\frac{\partial L_\rho}{\partial \boldsymbol{x}}\bigg|_{(\boldsymbol{x}^{(k)},\boldsymbol{z}^{(k)},\boldsymbol{\lambda}^{(k)})}\right), \tag{12}$$

where $\mathcal{P}(\cdot)$ denotes projection and α is a constant with $\alpha^{(k)} = 2^{-k}\alpha$. For non-convex functions, gradient descent may converge to local minima. After several iterations, the update step may be significantly shorter, and the convergence can be slow.

Meanwhile, Sun *et al.* presented an inertial ADMM (IADMM) for non-convex image deblurring [35]. Before the ADMM process, the inertial update steps for all the variables are computed as

$$\tilde{\boldsymbol{x}}^{(k)} = \boldsymbol{x}^{(k)} + \alpha\left(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}\right) \tag{13}$$

$$\tilde{\boldsymbol{z}}^{(k)} = \boldsymbol{z}^{(k)} + \alpha\left(\boldsymbol{z}^{(k)} - \boldsymbol{z}^{(k-1)}\right) \tag{14}$$

$$\tilde{\boldsymbol{\lambda}}^{(k)} = \boldsymbol{\lambda}^{(k)} + \alpha\left(\boldsymbol{\lambda}^{(k)} - \boldsymbol{\lambda}^{(k-1)}\right), \tag{15}$$

where $\alpha \geq 0$ is a user-defined parameter. When $\alpha = 0$, IADMM will revert to the conventional ADMM. Nevertheless, the update directions in the inertial scheme, such as $\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)}$, are not always proper directions for updating parameters. When $\boldsymbol{x}^{(k)}$ is close to its optimal value, inertial update may lead to oscillation near the minimum.

There are also some improvements for online ADMM parameter adaptation. For example, Peng proposed the adaptive ADMM (AADMM) scheme to give online estimation of the penalty parameter ρ according to the local Lipschitz constant [36]. Such methods can be embedded into many ADMM-related algorithms for parameter updates.

III. Foundation of Dual ADMM

In this paper, we regard the dual variable $\boldsymbol{\lambda}$ as a "colleague" rather than a "follower" of the primal variables, as shown in Fig. 1(b). The basic idea is to first solve the dual problem and the primal problem separately, then the dual variable $\boldsymbol{\lambda}$ can share its iterative results with the primal variables by passing the values to the primal process. The primal process has its own iteration steps, but the difference between this and the conventional ADMM is that the global dual variable $\boldsymbol{\lambda}$ is not an intermediate variable in the primal iterations but the direct solution of the dual problem.

Theoretically, a convex minimization problem can have non-unique minimizers. In real applications, as described in [18], it is often the case that ADMM converges to some points with modest accuracy within a few tens of iterations, and these results are regarded as the outputs of ADMM when the residual is smaller than the error tolerance. Even if the regularization function $\mathcal{R}(\cdot)$ is convex, due to the ill-posed nature of the system matrix A, the minimization problem is convex but not strongly convex, which means the minimizer may not be unique.

From the perspective of primal-dual theory [37], the dual solution is always related to a nontrivial lower bound of the objective function. Since our goal is to solve the minimization problem in Eq. 3, the solution of the dual problem can help to boost the convergence of the primal iterations, leading to comparable or sometimes slightly better quality results than the standard ADMM iterations in real applications with fewer number of iterations given the same stopping criteria. The worst case exists when the estimated lower bound is identical to that of the primal solution, i.e. the iterations of $\boldsymbol{\lambda}$ in the conventional ADMM are the relevant iterative solutions of the dual problem. In this case when the problem is strongly convex, D-ADMM will revert to the conventional ADMM, where there is only a unique minimizer. A conventional gradient ascent step in ADMM can find this minimizer easily since the problem is well-posed.

In this section, we focus on the mathematical foundation of our idea. We need to find the dual problem of Eq. 3 or other equivalent forms. We first consider the commonly-used scaled form in Eq. 6. In this expression, due to the existence of the quadratic term $\|\boldsymbol{\mu}\|_2^2$ related to the scaled dual variable, this function only has a minimum point instead of a saddle point. Hence, it is not straightforward to obtain its dual function. Moreover, there is only one linear term related to $\boldsymbol{\lambda}$ in Eq. 5, and it may seem possible to find the dual function. However, according to the definition of conjugate function [38], if we want to find a closed form for the dual function of Eq. 5, the system matrix A should be invertible, but actually A may not even be a square matrix. Hence, we will need to find another form related to the original optimization problem in Eq. 3 using the Lagrangian method.

A. Primal ADMM Process

We incorporate Eq. 1 into Eq. 3 to obtain the constrained optimization problem

$$\begin{aligned}&\underset{\boldsymbol{x},\boldsymbol{e}}{\text{minimize}}\ \mathcal{R}(\boldsymbol{x})+\frac{1}{2}\|\boldsymbol{e}\|_2^2\\&\text{subject to } A\boldsymbol{x}+\boldsymbol{e}=\boldsymbol{y}.\end{aligned}\tag{16}$$

Lagrange multiplier can be used such that [39]

$$\min_{\boldsymbol{x},\boldsymbol{e}} L(\boldsymbol{x},\boldsymbol{e},\boldsymbol{\lambda})=\min_{\boldsymbol{x}\in\mathbb{D},\boldsymbol{e}\in\mathbb{E}}\left\{\mathcal{R}(\boldsymbol{x})+\frac{1}{2}\|\boldsymbol{e}\|_2^2+\boldsymbol{\lambda}^T(A\boldsymbol{x}-\boldsymbol{y}+\boldsymbol{e})\right\},\tag{17}$$

where $L(\boldsymbol{x},\boldsymbol{e},\boldsymbol{\lambda})$ is the Lagrangian function, $\boldsymbol{\lambda}\in\mathbb{R}^M$ is the Lagrange multiplier, and $\mathbb{D}$ and $\mathbb{E}$ are the domains of the latent image and noise, respectively. In this form, the quadratic term related to the system matrix A and the dual variable $\boldsymbol{\lambda}$ does not exist, then we can easily find the closed form of its dual function (analyzed in the next subsection). In our approach to be explained next, the value of $\boldsymbol{\lambda}$ is updated according to the dual problem instead of the primal problem. Therefore, when analyzing the primal problem, the variable $\boldsymbol{\lambda}$ is regarded as a known constant. Compared with Eq. 3, Eq. 17 is now an optimization problem depending on a parameter $\boldsymbol{\lambda}$. The reason for extending Eq. 3 to Eq. 17 is based on the fundamental idea of the theory of dual optimization [40]. This theory is concerned with representing a given minimization problem as "half" of a minimax problem whose saddle point exists. In this paper, the primal problem and the dual problem are two halves of the whole minimax problem. In this way, we introduce the dual to help with the solution of the original optimization problem in Eq. 3. As is summarized in [40], the dual approach has many advantages, both theoretically and computationally. For example, the dual function is a "lower representative" of Eq. 17, which makes it easier to find the minimum. Strong duality can also prove the optimality of the problem.

We first consider the primal problem. In general, the regularization function $\mathcal{R}(\boldsymbol{x})$ may not be smooth. Using the variable splitting method, we can separate the non-smooth part from others. Then, it is possible to deal with these parts separately in different sub-problems with simpler optimization methods. The standard form of Eq. 17 with splitting variable $\boldsymbol{z}$ is

$$\begin{aligned}&\underset{\boldsymbol{x},\boldsymbol{z},\boldsymbol{e}}{\text{minimize}}\ \mathcal{R}(\boldsymbol{z})+\frac{1}{2}\|\boldsymbol{e}\|_2^2+\boldsymbol{\lambda}^T(A\boldsymbol{x}-\boldsymbol{y}+\boldsymbol{e})\\&\text{subject to } \boldsymbol{z}-\boldsymbol{x}=0.\end{aligned}\tag{18}$$

The corresponding augmented Lagrangian is

$$\begin{aligned}\tilde{L}_1(\boldsymbol{x},\boldsymbol{z},\boldsymbol{e},\boldsymbol{\mu_1})=\mathcal{R}(\boldsymbol{z})&+\frac{1}{2}\|\boldsymbol{e}\|_2^2+\boldsymbol{\lambda}^T(A\boldsymbol{x}-\boldsymbol{y}+\boldsymbol{e})\\&+\boldsymbol{\mu_1}^T(\boldsymbol{z}-\boldsymbol{x})+\frac{\rho_1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2,\end{aligned}\tag{19}$$

where $\boldsymbol{\mu}_1$ is the Lagrange multiplier and ρ_1 is the penalty parameter. The variable $\boldsymbol{e}$ is first considered since it is separable from other variables. Setting $\partial\tilde{L}_1/\partial\boldsymbol{e}=0$, the $\boldsymbol{e}$-update step can be obtained as [39]

$$\boldsymbol{e}^{(k+1)}=-\boldsymbol{\lambda}.\tag{20}$$

The augmented Lagrangian without $\boldsymbol{e}$ is

$$L_1(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\mu_1})=\mathcal{R}(\boldsymbol{z})+\boldsymbol{\lambda}^T(A\boldsymbol{x}-\boldsymbol{y})+\boldsymbol{\mu_1}^T(\boldsymbol{z}-\boldsymbol{x})+\frac{\rho_1}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2.\tag{21}$$

This can be solved by iterating the following steps [18]:

$$\boldsymbol{x}^{(k+1)}=\frac{1}{\rho_1}\left(\rho_1\boldsymbol{z}^{(k)}+\boldsymbol{\mu_1}^{(k)}-A^T\boldsymbol{\lambda}\right)\tag{22}$$

$$\boldsymbol{z}^{(k+1)}=\arg\min_{\boldsymbol{z}} L_1\left(\boldsymbol{x}^{(k+1)},\boldsymbol{z},\boldsymbol{\mu_1}^{(k)}\right)\tag{23}$$

$$\boldsymbol{\mu_1}^{(k+1)}=\boldsymbol{\mu_1}^{(k)}+\rho_1\left(\boldsymbol{z}^{(k+1)}-\boldsymbol{x}^{(k+1)}\right).\tag{24}$$

In these steps, no variable is directly related to the value of $\boldsymbol{e}$, hence the $\boldsymbol{e}$-update step can be omitted for simplicity.

Referring to the usage of the proximal operator [23], the $\boldsymbol{z}$-update step in Eq. 23 can be further clarified. After removing the unrelated terms with respect to $\boldsymbol{z}$, the minimization problem is actually

$$\boldsymbol{z}^{(k+1)}=\arg\min_{\boldsymbol{z}}\left\{\mathcal{R}(\boldsymbol{z})+\boldsymbol{\mu_1}^{(k)T}\boldsymbol{z}+\frac{\rho_1}{2}\|\boldsymbol{z}-\boldsymbol{x}^{(k+1)}\|_2^2\right\}\tag{25}$$

$$=\arg\min_{\boldsymbol{z}}\left\{\frac{1}{\rho_1}\mathcal{R}(\boldsymbol{z})+\frac{1}{2}\left\|\boldsymbol{z}-\boldsymbol{x}^{(k+1)}+\frac{\boldsymbol{\mu_1}^{(k)}}{\rho_1}\right\|_2^2\right\},\tag{26}$$

which leads to the expression with a proximal operator

$$\boldsymbol{z}^{(k+1)}=\text{prox}_{\frac{\mathcal{R}}{\rho_1}}\left(\boldsymbol{x}^{(k+1)}-\frac{\boldsymbol{\mu_1}^{(k)}}{\rho_1}\right).\tag{27}$$

B. Dual ADMM Process

The dual problem is analyzed in this part. We first find the dual function (also called the optimal value function in [40]) of Eq. 17 as

$$\begin{aligned}g(\boldsymbol{\lambda})&=\inf_{\boldsymbol{x}\in\mathbb{D},\boldsymbol{e}\in\mathbb{E}} L(\boldsymbol{x},\boldsymbol{e},\boldsymbol{\lambda})\\&=-\boldsymbol{\lambda}^T\boldsymbol{y}-\sup_{\boldsymbol{x}\in\mathbb{D}}\left\{-\boldsymbol{\lambda}^TA\boldsymbol{x}-\mathcal{R}(\boldsymbol{x})\right\}\\&\quad-\sup_{\boldsymbol{e}\in\mathbb{E}}\left\{-\boldsymbol{\lambda}^T\boldsymbol{e}-\frac{1}{2}\|\boldsymbol{e}\|_2^2\right\}\\&=-\boldsymbol{\lambda}^T\boldsymbol{y}-\mathcal{R}^*(-A^T\boldsymbol{\lambda})-\frac{1}{2}\|-\boldsymbol{\lambda}\|_2^2,\end{aligned}\tag{28}$$

where $\mathcal{R}^*(\cdot)$ is the conjugate function of $\mathcal{R}(\cdot)$ [38]. The advantage of introducing Eq. 17 over Eq. 5 is that it will be easier to find the dual function. Then, the dual problem becomes

$$\arg\max_{\boldsymbol{\lambda}} g(\boldsymbol{\lambda})=\arg\min_{\boldsymbol{\lambda}}\left\{\boldsymbol{\lambda}^T\boldsymbol{y}+\mathcal{R}^*(-A^T\boldsymbol{\lambda})+\frac{1}{2}\|\boldsymbol{\lambda}\|_2^2\right\},\tag{29}$$

which is still a convex optimization problem since $\mathcal{R}^*(\cdot)$ is always convex [38]. We now show how the global dual variable $\boldsymbol{\lambda}$ can be updated accordingly.

Eq. 29 can also be solved by ADMM as

$$\begin{aligned}&\underset{\boldsymbol{\lambda},\boldsymbol{c}}{\text{minimize}}\ \ \boldsymbol{y}^T\boldsymbol{\lambda} + \mathcal{R}^*(\boldsymbol{c}) + \frac{1}{2}\|\boldsymbol{\lambda}\|_2^2\\ &\text{subject to } A^T\boldsymbol{\lambda} + \boldsymbol{c} = 0,\end{aligned} \tag{30}$$

where $\boldsymbol{c}$ is an auxiliary variable. The augmented Lagrangian is

$$\begin{aligned}L_2(\boldsymbol{\lambda}, \boldsymbol{c}, \boldsymbol{\mu_2}) = \boldsymbol{y}^T\boldsymbol{\lambda} + \mathcal{R}^*(\boldsymbol{c}) + \frac{1}{2}\|\boldsymbol{\lambda}\|_2^2\\ + \boldsymbol{\mu_2}^T(A^T\boldsymbol{\lambda} + \boldsymbol{c}) + \frac{\rho_2}{2}\|A^T\boldsymbol{\lambda} + \boldsymbol{c}\|_2^2.\end{aligned} \tag{31}$$

Consequently, the related ADMM iterations are [18]

$$\boldsymbol{\lambda}^{(k+1)} = \left(\rho_2 AA^T + I\right)^{-1}\left(-\boldsymbol{y} - A\boldsymbol{\mu_2}^{(k)} - \rho_2 A\boldsymbol{c}^{(k)}\right) \tag{32}$$

$$\boldsymbol{c}^{(k+1)} = \arg\min_{\boldsymbol{c}} L_2(\boldsymbol{\lambda}^{(k+1)}, \boldsymbol{c}, \boldsymbol{\mu_2}^{(k)}) \tag{33}$$

$$\boldsymbol{\mu_2}^{(k+1)} = \boldsymbol{\mu_2}^{(k)} + \rho_2\left(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)}\right). \tag{34}$$

Similarly, Eq. 33 can also be written as a proximal update step

$$\boldsymbol{c}^{(k+1)} = \text{prox}_{\frac{\mathcal{R}^*}{\rho_2}}\left(-A^T\boldsymbol{\lambda}^{(k+1)} - \frac{\boldsymbol{\mu_2}^{(k)}}{\rho_2}\right). \tag{35}$$

Furthermore, with the relationship between $\mathcal{R}$ and $\mathcal{R}^*$ explained in Lemma 5.1 in Appendix A, the $\boldsymbol{c}$-update step can be written as a proximal step with $\mathcal{R}$ directly, which can help to avoid finding the explicit form of the conjugate function $\mathcal{R}^*$. When solving the minimization problem in Eq. 33, a necessary condition is

$$\nabla\mathcal{R}^*(\boldsymbol{c}) + \rho_2(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}) + \boldsymbol{\mu_2}^{(k)} = 0. \tag{36}$$

Using Lemma 5.1, we have

$$\rho_2\boldsymbol{c} + \rho_2\nabla\mathcal{R}(\rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \rho_2\boldsymbol{c} + \boldsymbol{\mu_2}^{(k)}) = 0. \tag{37}$$

If we define $\boldsymbol{c_r}^{(k+1)} = \rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{\mu_2}^{(k)}$ and $\tilde{\boldsymbol{c}} = \boldsymbol{c}_r{}^{(k+1)} + \rho_2\boldsymbol{c}$, then Eq. 37 can be expressed as

$$\tilde{\boldsymbol{c}} + \rho_2\nabla\mathcal{R}(\tilde{\boldsymbol{c}}) = \boldsymbol{c}_r{}^{(k+1)}. \tag{38}$$

This can also be solved by a proximal operator

$$\tilde{\boldsymbol{c}}^{(k+1)} = \text{prox}_{\rho_2\mathcal{R}}(\boldsymbol{c}_r{}^{(k+1)}). \tag{39}$$

and finally we have the $\boldsymbol{c}$-update step

$$\boldsymbol{c}^{(k+1)} = \frac{1}{\rho_2}[\text{prox}_{\rho_2\mathcal{R}}(\boldsymbol{c}_r{}^{(k+1)}) - \boldsymbol{c}_r{}^{(k+1)}]. \tag{40}$$

Referring to the relationship between Eq. 29 and Eq. 17, we combine the dual ADMM iterations (Eq. 32–34 and 40) with the primal ADMM iterations (Eq. 22–24 and 27) to merge the information embedded in them, particularly providing $\boldsymbol{\lambda}^{(k+1)}$ for the update in $\boldsymbol{x}^{(k+1)}$ [41]. We name this method the dual ADMM (D-ADMM) scheme. This is summarized in Algorithm 1. The quantities $\boldsymbol{x}^{(0)}$ and $\boldsymbol{z}^{(0)}$ are initialized as $A^T\boldsymbol{y}$, and $\boldsymbol{\mu}_2{}^{(0)}$ is set to be $-A^T\boldsymbol{y}$. Other initial values are set as 0.

Algorithm 1 General D-ADMM Scheme

```
Input: $\boldsymbol{y}$: measurement signal; $A$: system matrix; $\rho_1, \rho_2$:
    penalty parameters; $\mathcal{R}(\cdot)$: regularization function.
Output: $\boldsymbol{x}$: reconstructed signal.
while $\boldsymbol{x}$ has not converged do
    Global dual ADMM update:
    $\boldsymbol{\lambda}^{(k+1)} = (\rho_2 AA^T + I)^{-1}(-\boldsymbol{y} - A\boldsymbol{\mu_2}^{(k)} - \rho_2 A\boldsymbol{c}^{(k)})$.
    $\boldsymbol{c}_r{}^{(k+1)} = \rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{\mu_2}^{(k)}$.
    $\boldsymbol{c}^{(k+1)} = \frac{1}{\rho_2}[\text{prox}_{\rho_2\mathcal{R}}(\boldsymbol{c}_r{}^{(k+1)}) - \boldsymbol{c}_r{}^{(k+1)}]$.
    ADMM update:
    $\boldsymbol{x}^{(k+1)} = \frac{1}{\rho_1}(\rho_1\boldsymbol{z}^{(k)} + \boldsymbol{\mu_1}^{(k)} - A^T\boldsymbol{\lambda}^{(k+1)})$.
    $\boldsymbol{z}^{(k+1)} = \text{prox}_{\frac{\mathcal{R}}{\rho_1}}\left(\boldsymbol{x}^{(k+1)} - \frac{\boldsymbol{\mu_1}^{(k)}}{\rho_1}\right)$.
    Local dual variables update:
    $\boldsymbol{\mu_1}^{(k+1)} = \boldsymbol{\mu_1}^{(k)} + \rho_1(\boldsymbol{z}^{(k+1)} - \boldsymbol{x}^{(k+1)})$.
    $\boldsymbol{\mu_2}^{(k+1)} = \boldsymbol{\mu_2}^{(k)} + \rho_2(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)})$.
end while
```

An advantage of D-ADMM is that $\boldsymbol{\lambda}^{(k)}$ is related to the lower bound for all possible $\boldsymbol{x}$ instead of the specific value $\boldsymbol{x}^{(k)}$, which helps the primal iterations to find a minimizer faster. In D-ADMM, $\boldsymbol{\lambda}$ is not an intermediate variable of $\boldsymbol{x}$, and it helps to bring new information embedded in the dual problem to the $\boldsymbol{x}$-update step. Besides, it is evident in Fig. 1 that parallel computing can be adopted, and the processing time will be much reduced compared with simply using two ADMM iterations. The primal iterations and the dual iterations can be computed at the same time, and the processing time is determined by the slower one of these two iterations, rather than the direct sum of their processing time.

A complete convergence proof based on variational inequality is given in Appendix B.

C. Two Examples

1) ℓ_2-Regularization: In the last section, we present the D-ADMM scheme that is suitable for convex regularization functions. In this section, we further specify the regularization function $\mathcal{R}(\cdot)$ as a commonly-used ℓ_2 regularization function

$$\mathcal{R}(\boldsymbol{x}) = \frac{\gamma}{2}\|\boldsymbol{x}\|_2^2. \tag{41}$$

According to the definition of conjugate function, $\mathcal{R}^*(\cdot)$ can be represented as

$$\mathcal{R}^*(\boldsymbol{y}) = \sup_{\boldsymbol{x}}\left\{\boldsymbol{y}^T\boldsymbol{x} - \frac{\gamma}{2}\|\boldsymbol{x}\|_2^2\right\} = \frac{1}{2\gamma}\|\boldsymbol{y}\|_2^2. \tag{42}$$

Combining Eq. 41 and Eq. 23, we have the explicit formula

$$\boldsymbol{z}^{(k+1)} = \frac{1}{\gamma + \rho_1}\left(\rho_1\boldsymbol{x}^{(k+1)} - \boldsymbol{\mu_1}^{(k)}\right). \tag{43}$$

Similarly, we can put Eq. 42 into Eq. 33 and obtain

$$\boldsymbol{c}^{(k+1)} = -\frac{\gamma}{1 + \gamma\rho_2}\left(\rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{\mu_2}^{(k)}\right). \tag{44}$$

These two results can also be obtained using the proximal operator directly. The D-ADMM algorithm with ℓ_2 regularization, denoted as D-ADMM-L2, is summarized in Algorithm 2.

Algorithm 2 D-ADMM Algorithm With ℓ_2 Regularization (D-ADMM-L2)

```
Input: $\boldsymbol{y}$: measurement signal; $A$: system matrix; $\rho_1, \rho_2$: penalty parameters; $\gamma$: scale parameter.
Output: $\boldsymbol{x}$: reconstructed signal.
while $\boldsymbol{x}$ has not converged do
    Global dual ADMM update:
    $\boldsymbol{\lambda}^{(k+1)} = (\rho_2 AA^T + I)^{-1}(-\boldsymbol{y} - A\boldsymbol{\mu_2}^{(k)} - \rho_2 A\boldsymbol{c}^{(k)})$.
    $\boldsymbol{c}^{(k+1)} = -\frac{\gamma}{1+\gamma\rho_2}(\rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{\mu_2}^{(k)})$.
    ADMM update:
    $\boldsymbol{x}^{(k+1)} = \frac{1}{\rho_1}(\rho_1\boldsymbol{z}^{(k)} + \boldsymbol{\mu_1}^{(k)} - A^T\boldsymbol{\lambda}^{(k+1)})$.
    $\boldsymbol{z}^{(k+1)} = \frac{1}{\gamma+\rho_1}(\rho_1\boldsymbol{x}^{(k+1)} - \boldsymbol{\mu_1}^{(k)})$.
    Local dual variables update:
    $\boldsymbol{\mu_1}^{(k+1)} = \boldsymbol{\mu_1}^{(k)} + \rho_1(\boldsymbol{z}^{(k+1)} - \boldsymbol{x}^{(k+1)})$.
    $\boldsymbol{\mu_2}^{(k+1)} = \boldsymbol{\mu_2}^{(k)} + \rho_2(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)})$.
end while
```

2) TV-Regularization: Another commonly-used function is the ℓ_1-regularization, and the related TV regularization is the ℓ_1-regularization of the gradient. The regularization function $\mathcal{R}(\cdot)$ is now specified as

$$\mathcal{R}(\boldsymbol{x}) = \gamma \, \|D\boldsymbol{x}\|_1, \tag{45}$$

where D is the system matrix of the differential operation.

Since the TV regularization function is more complex than ℓ_2 regularization, it is better to take advantage of the proximal operator. As is shown in [26], the proximal operator with TV regularization function is actually a denoising step using a TV denoiser, denoted as $\mathcal{D}_\sigma$, where σ is the parameter for this denoiser.

Both the $\boldsymbol{c}$-update and $\boldsymbol{z}$-update steps can be expressed as a TV denoising step with $\mathcal{D}_\sigma(\cdot)$. In the former, $\sigma_c = \sqrt{\rho_2\gamma}$, and in the latter, $\sigma_z = \sqrt{\frac{\gamma}{\rho_1}}$. The two update steps are

$$\boldsymbol{c}^{(k+1)} = \frac{1}{\rho_2}\left[\mathcal{D}_{\sqrt{\rho_2\gamma}}(\boldsymbol{c}_r{}^{(k+1)}) - \boldsymbol{c}_r{}^{(k+1)}\right] \tag{46}$$

$$\boldsymbol{z}^{(k+1)} = \mathcal{D}_{\sqrt{\frac{\gamma}{\rho_1}}}\left(\boldsymbol{x}^{(k+1)} - \frac{\boldsymbol{\mu_1}^{(k)}}{\rho_1}\right). \tag{47}$$

The whole process for TV regularization, denoted as D-ADMM-TV, is shown in Algorithm 3. Note that we take TV denoising as an example, while other proper denoisers can also be adopted directly after the parameters are determined with respect to the two penalty parameters ρ_1 and ρ_2.

There are also some first-order approaches proposed to solve the TV-regularization problem, such as Newton method [42], direct proximal algorithm [43] and gradient-based method [44]. They seem to have simpler forms than ADMM-based algorithms. However, as described in [45], the system matrix A varies in different imaging applications, and in general it is difficult to solve the underlying proximal step because of the presence of a complex A for some imaging applications.

3) Parameter Analysis: Generally, D-ADMM has twice as many hyper-parameters as ADMM. There are two hyper-parameters, ρ_1 and ρ_2, which are penalty parameters in the Lagrange functions. If the value of ρ_1 is larger, the primal solution is more likely to satisfy the constraint $\boldsymbol{z} = \boldsymbol{x}$ first. Similarly, when ρ_2 is larger, the dual solution is more likely to consider the constraint $A^T\boldsymbol{\lambda} + \boldsymbol{c} = 0$ first.

Algorithm 3 D-ADMM Algorithm With TV Regularization (D-ADMM-TV)

```
Input: $\boldsymbol{y}$: measurement signal; $A$: system matrix; $\rho_1, \rho_2$: penalty parameters; $\gamma$: scale parameter; $\mathcal{D}_\sigma$: TV denoiser.
Output: $\boldsymbol{x}$: reconstructed signal.
while $\boldsymbol{x}$ has not converged do
    Global dual ADMM update:
    $\boldsymbol{\lambda}^{(k+1)} = (\rho_2 AA^T + I)^{-1}(-\boldsymbol{y} - A\boldsymbol{\mu_2}^{(k)} - \rho_2 A\boldsymbol{c}^{(k)})$.
    $\boldsymbol{c}_r{}^{(k+1)} = \rho_2 A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{\mu_2}^{(k)}$.
    $\boldsymbol{c}^{(k+1)} = \frac{1}{\rho_2}[\mathcal{D}_{\sqrt{\rho_2\gamma}}(\boldsymbol{c}_r{}^{(k+1)}) - \boldsymbol{c}_r{}^{(k+1)}]$.
    ADMM update:
    $\boldsymbol{x}^{(k+1)} = \frac{1}{\rho_1}(\rho_1\boldsymbol{z}^{(k)} + \boldsymbol{\mu_1}^{(k)} - A^T\boldsymbol{\lambda}^{(k+1)})$.
    $\boldsymbol{z}^{(k+1)} = \mathcal{D}_{\sqrt{\frac{\gamma}{\rho_1}}}\left(\boldsymbol{x}^{(k+1)} - \frac{\boldsymbol{\mu_1}^{(k)}}{\rho_1}\right)$.
    Local dual variables update:
    $\boldsymbol{\mu_1}^{(k+1)} = \boldsymbol{\mu_1}^{(k)} + \rho_1(\boldsymbol{z}^{(k+1)} - \boldsymbol{x}^{(k+1)})$.
    $\boldsymbol{\mu_2}^{(k+1)} = \boldsymbol{\mu_2}^{(k)} + \rho_2(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)})$.
end while
```

In the two examples, an additional scale parameter γ is introduced, which can give a compromise between the data-fidelity term $\frac{1}{2}\|A\boldsymbol{x} - \boldsymbol{y}\|_2^2$ and the regularization term $\mathcal{R}(\boldsymbol{x})$.

4) Termination Criteria: In conventional ADMM, the stopping criteria require both the global primal residual and the global dual residual to be small. In the proposed method, the stopping criteria have a similar form as the conventional ADMM, which can be represented as

$$\|\boldsymbol{r}^{(k)}\|_2 \le \epsilon_{\text{pri}} \tag{48}$$

$$\|\boldsymbol{s}^{(k)}\|_2 \le \epsilon_{\text{dual}}, \tag{49}$$

where $\|\boldsymbol{r}^{(k)}\|_2$ is the primal residual $\|\boldsymbol{x}^{(k)} - \boldsymbol{z}^{(k)}\|_2$ and $\|\boldsymbol{s}^{(k)}\|_2$ is the dual residual $\|A^T\boldsymbol{\lambda}^{(k)} + \boldsymbol{c}^{(k)}\|_2$, subject to the constraints in the primal and the dual problems. The variables ϵ_{pri} and ϵ_{dual} are feasibility tolerances, which consist of an absolute term and a relative term

$$\epsilon_{\text{pri}} = \sqrt{p}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \max\{\|\boldsymbol{x}^{(k)}\|_2, \|\boldsymbol{z}^{(k)}\|_2\}, \tag{50}$$

$$\epsilon_{\text{dual}} = \sqrt{q}\epsilon_{\text{abs}} + \epsilon_{\text{rel}} \max\{\|A^T\boldsymbol{\lambda}^{(k)}\|_2, \|\boldsymbol{c}^{(k)}\|_2\}, \tag{51}$$

where ϵ_{abs} is the absolute tolerance, ϵ_{rel} is the relative tolerance, p is the dimension of $\boldsymbol{z}$, and q is the dimension of $\boldsymbol{c}$.

The above criteria may be further simplified. Similar to ADMM, D-ADMM also satisfies the fixed point convergence property. A direct criterion is to consider the difference of all the variables between two successive iterations. The residual can be defined as

$$\epsilon_{\text{pri}} = \frac{1}{\sqrt{p}}(\|\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k+1)}\|_2 + \|\boldsymbol{z}^{(k)} - \boldsymbol{z}^{(k+1)}\|_2 + \|\boldsymbol{\mu_1}^{(k)} - \boldsymbol{\mu_1}^{(k+1)}\|_2), \tag{52}$$

and the stopping criteria is

$$\epsilon_{\text{pri}} \le \epsilon_0, \tag{53}$$

where ϵ_0 is the tolerance. This is also adopted by the plug-and-play ADMM [26]. In our approach, we have two ADMM iterations, hence this stopping criteria should be extended to

$$\max\{\epsilon_{\text{pri}}, \epsilon_{\text{dual}}\} \leq \epsilon_0, \tag{54}$$

where ϵ_{dual} is the residual of the dual problem

$$\epsilon_{\text{dual}} = \frac{1}{\sqrt{q}}(\|\boldsymbol{\lambda}^{(k)} - \boldsymbol{\lambda}^{(k+1)}\|_2 + \|\boldsymbol{c}^{(k)} - \boldsymbol{c}^{(k+1)}\|_2 + \|\boldsymbol{\mu_2}^{(k)} - \boldsymbol{\mu_2}^{(k+1)}\|_2). \tag{55}$$

Similar to the conventional ADMM, according to the global convergence property and the fixed point convergence property described in Appendix B, if these residuals are lower than the predefined tolerances, the results are regarded as reasonable.

IV. Experiments

A. Image Super-Resolution

We first compare the performance of D-ADMM and ADMM in a real inverse imaging problem called image super-resolution (SR). This aims at recovering high-resolution images from low-resolution images [46], [47]. Generally, the forward imaging model consists of a downsampling step by a factor k_d. The system matrix S is a sampling matrix which is defined as

$$S(i,j) = \begin{cases} 1, & i\text{th pixel in } \boldsymbol{y} \text{ sampled from } j\text{th pixel in } \boldsymbol{x} \\ 0, & \text{otherwise.} \end{cases} \tag{56}$$

Generally, there is also a convolution step before sampling. System matrix for convolution is approximated as a block circulant with circulant block (BCCB) matrix C constructed by the blur kernel. The whole imaging model can therefore be described as

$$\boldsymbol{y} = SC\boldsymbol{x} + \boldsymbol{e}. \tag{57}$$

When S is an identity matrix, the SR problem becomes an image deblurring problem. Comparing SR with Eq. 1, we have $A = SC$. It is shown in [26] that when S is a standard K-fold downsampler and C represents circular convolution, AA^T has a symmetric and analytical expression. Therefore, it is possible to deal with the AA^T-related inverse matrix in the iteration process analytically.

1) Experimental Setup: We consider 10 standard grayscale images for SR experiments, as shown in Fig. 2. Their sizes are either 256 × 256 or 512 × 512. The downsampling factor is set to be 2 and 4 respectively, and the convolution involves a 9 × 9 Gaussian kernel with standard deviation equals to 1. Additive white Gaussian noise is added to the downsampled images and the noise level is 5/255.

We compare the results from our methods with some other SR techniques. The PnP-ADMM [26] method shares some similarity with our proposal, but the main difference is that D-ADMM has a second ADMM with reference to the solution of the dual problem. For fair comparison, the PnP-ADMM is selected as the original version [48] where the parameter ρ is static. If PnP-ADMM and D-ADMM adopt the same denoiser for the regularization term, then the results can clearly show the effect of the dual iterations.

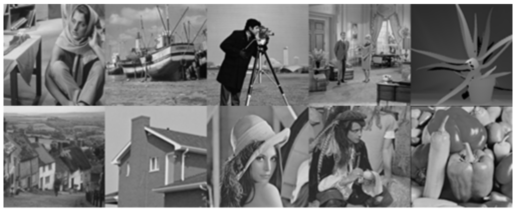


Fig. 2. Ten grayscale high-resolution images for SR experiments (From left to right, top to bottom: No. 1 - 10).

The transformed self-exemplar (TSE) [49] is a patch-searching method with an additional affine transformation for correction. We also compare our methods with deep-learning approaches. The deep convolutional network for image super-resolution (SRCNN) [50] adopts an end-to-end convolutional neural network (CNN) for image SR. The input for SRCNN is the upsampled image with bicubic interpolation, which means that SRCNN is mainly used for texture correction.

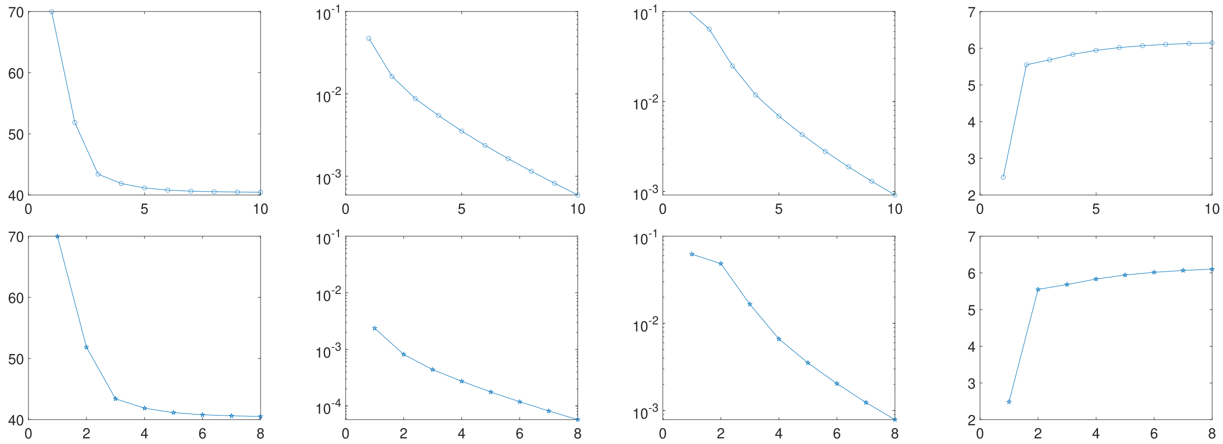
We use $\rho_1 = 0.05$ and $\rho_2 = 1/0.05 = 20$ in D-ADMM-TV. The scale parameter γ is set to be 0.01 and the error tolerance is set to be $\epsilon_0 = 10^{-3}$, which are identical to the setup in PnP-ADMM and this will guarantee that D-ADMM-TV and PnP-ADMM-TV are solving the same optimization problem in Eq. 3 with the same error tolerance. Since an optimal penalty parameter selection is not the focus of this paper, to eliminate the influence of the penalty parameter selection, we set $\rho = \rho_1 = 0.05$ and initialize with $A^T\boldsymbol{y}$ in PnP-ADMM-TV.

All these experiments are carried out on a computer with Intel(R) Core(TM) i5-4210M CPU @ 2.60GHz and 12GB RAM. MATLAB R2020a in the CPU mode is used as the platform. The SRCNN network is implemented with an additional NVIDIA GeForce GTX 850M GPU.

In our D-ADMM-TV scheme, the TV denoiser can potentially be changed to other proper denoisers, and this will lead to different results. However, our objective here is not to investigate what the best denoiser is, but rather to give a direct comparison between ADMM and D-ADMM schemes. For fair comparison, in this experiment, the TV denoiser is plugged into PnP-ADMM, which leads to the method called PnP-ADMM-TV.

2) SR Results: We first give the convergence results per iteration for D-ADMM-TV and PnP-ADMM-TV, respectively. The results for the No. 6 image are shown in Fig. 3. As shown in Fig. 3, D-ADMM-TV has faster convergence under the same error tolerance. The main reason is that $\boldsymbol{\mu_1}$ in D-ADMM has better convergence performance than $\boldsymbol{\mu}$ in ADMM, with the help of the dual iterations.

Numerical results with peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM) for the 2× SR task are shown in Table I and II, respectively. The SR images on the "Man" image (No. 9) are shown in Fig. 4. We can see that the TSE method does not give special considerations for the noisy SR situation. Similarly, SRCNN is only trained with noise-free image pairs. As shown in Fig. 4, a lot of noise still exists in (c) and (e). For a fair comparison, we append the same TV



(a) Objective function (b) Dual residual (c) Total residual (d) Data error

Fig. 3. Iterative results for 2× image super-resolution with the No. 6 image. The top row is PnP-ADMM-TV while the bottom row is D-ADMM-TV. x-axis: the number of iterations. y-axis: (a) value of the objective function in Eq. 3; (b) Dual residual $\frac{1}{\sqrt{p}}\|\boldsymbol{\mu}_1{}^{(k+1)} - \boldsymbol{\mu}_1{}^{(k)}\|_2$, which is the third term in Eq. 52; (c) Total residual expressed on the left side of Eq. 54 for D-ADMM-TV and Eq. 53 for PnP-ADMM-TV; (d) data-fidelity error $\|A\boldsymbol{x} - \boldsymbol{y}\|_2$. Note that D-ADMM-TV only needs 8 iterations using the termination criteria in Eq. 54 but PnP-ADMM-TV needs 10 iterations using the termination criteria in Eq. 53, with the same error tolerance.

TABLE I

PSNR FOR 2× SR RESULTS ON 10 IMAGES (UNIT: dB)

No.	1	2	3	4	5	6	7	8	9	10	Average
Size	512^2	512^2	256^2	512^2	512^2	512^2	256^2	512^2	512^2	256^2	
TSE	23.56	26.20	23.67	26.09	29.43	27.12	27.23	27.92	26.88	24.71	26.28
TSE*	23.80	26.70	23.91	26.53	31.23	27.78	28.02	28.84	27.48	24.94	26.92
SRCNN	23.62	26.27	23.81	26.18	29.48	27.26	27.49	28.07	26.99	24.24	26.34
SRCNN*	24.01	27.11	24.20	**26.99**	31.92	28.39	28.71	29.45	28.05	24.86	27.40
PnP-ADMM-TV	24.01	27.39	25.19	26.89	34.90	28.41	30.74	30.53	28.33	25.10	28.15
D-ADMM-TV	**24.02**	**27.41**	**25.25**	26.98	**35.90**	**28.44**	**30.78**	**30.56**	**28.36**	**25.12**	**28.28**

TABLE II

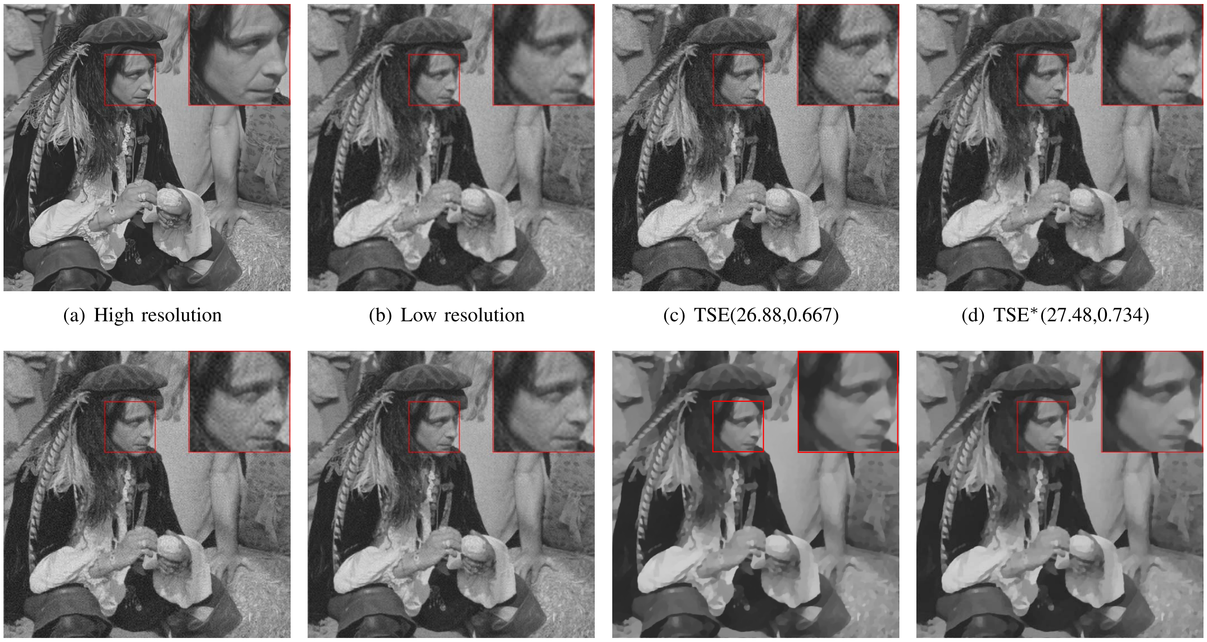
SSIM FOR 2× SR RESULTS ON 10 IMAGES

No.	1	2	3	4	5	6	7	8	9	10	Average
Size	512^2	512^2	256^2	512^2	512^2	512^2	256^2	512^2	512^2	256^2	
TSE	0.602	0.650	0.614	0.655	0.631	0.648	0.639	0.666	0.667	0.703	0.648
TSE*	0.659	0.712	0.709	0.708	0.803	0.705	0.742	0.765	0.734	0.772	0.731
SRCNN	0.621	0.668	0.648	0.672	0.643	0.667	0.671	0.692	0.683	0.715	0.668
SRCNN*	**0.685**	**0.739**	0.746	**0.736**	0.837	**0.725**	0.781	0.804	**0.768**	0.799	0.762
PnP-ADMM-TV	0.678	0.723	0.777	0.707	0.958	0.707	0.832	0.829	0.753	0.812	0.778
D-ADMM-TV	0.679	0.725	**0.780**	0.711	**0.965**	0.709	**0.833**	**0.830**	0.755	**0.813**	**0.780**

denoiser to TSE, and that leads to the results of TSE*. Since SRCNN is a learning-based method, we provide a fine tuning to the official trained model for SRCNN, leading to the results of SRCNN*. The ground truth images in the tuning dataset are 90 images randomly selected from ImageNet [51]. The low-resolution images are generated using the downsampling model in Eq. 57, which is identical to the model for D-ADMM and PnP-ADMM. We can see that TSE* and SRCNN* give better results than TSE and SRCNN. Our D-ADMM-TV gives comparable or, at times, somewhat better quality results than PnP-ADMM-TV, TSE, TSE* and SRCNN for all the images, even with fewer iterations than PnP-ADMM-TV. SRCNN* achieves the best performance on some occasions, but it should be emphasized that D-ADMM-TV does not require training data, and there is room for further improvement.

The 4× image SR is a more challenging problem. We also compare D-ADMM-TV with other methods for this problem. In this experiment, the scale parameter is also set as $\gamma = 0.01$. For the same reason to eliminate the influence of penalty parameter selection, we set $\rho = \rho_1 = 0.05$ and initialize with $A^T\boldsymbol{y}$ in ADMM-TV. The PSNR and SSIM results are shown in Table III and IV. In this experiment, D-ADMM-TV achieves competitive and sometimes a bit better results than other methods except No. 4 and No. 6. In this more difficult task, the fine tuning still provides help to SRCNN, but the effectiveness is less compared with the 2× SR results.

The convergence results per iteration for D-ADMM-TV and PnP-ADMM-TV with the No. 4 image are shown in Fig. 5. Even in a more difficult task, D-ADMM-TV has a better convergence property than PnP-ADMM-TV.


(a) High resolution (b) Low resolution (c) TSE(26.88,0.667) (d) TSE*(27.48,0.734)

(e) SRCNN(26.99,0.683) (f) SRCNN*(28.05,0.768) (g) PnP-ADMM-TV(28.33,0.753) (h) D-ADMM-TV(28.36,0.755)

Fig. 4. 2× SR results for No. 9 image (PSNR: dB, SSIM).

TABLE III

PSNR FOR 4× SR RESULTS ON 10 IMAGES (UNIT: dB)

No.	1	2	3	4	5	6	7	8	9	10	Average
Size	512^2	512^2	256^2	512^2	512^2	512^2	256^2	512^2	512^2	256^2	
TSE	20.55	21.28	18.87	21.46	25.56	23.05	22.06	22.76	22.21	19.71	21.75
TSE*	21.26	22.20	19.88	22.24	27.06	23.65	23.17	23.84	22.96	20.07	22.63
SRCNN	20.90	21.68	19.26	21.93	26.14	23.57	22.56	23.25	22.69	19.72	22.17
SRCNN*	21.94	22.67	21.21	23.06	28.83	24.81	24.25	25.46	24.57	21.63	23.84
PnP-ADMM-TV	22.72	23.99	22.15	23.63	30.29	25.31	25.99	26.76	25.06	22.25	24.81
D-ADMM-TV	**22.73**	**24.01**	**22.18**	**23.66**	**30.63**	**25.32**	**26.02**	**26.78**	**25.07**	**22.28**	**24.87**

TABLE IV

SSIM FOR 4× SR RESULTS ON 10 IMAGES

No.	1	2	3	4	5	6	7	8	9	10	Average
Size	512^2	512^2	256^2	512^2	512^2	512^2	256^2	512^2	512^2	256^2	
TSE	0.472	0.485	0.489	0.467	0.643	0.494	0.545	0.578	0.511	0.564	0.525
TSE*	0.541	0.540	0.616	0.505	0.880	0.521	0.687	0.696	0.571	0.622	0.618
SRCNN	0.540	0.562	0.586	0.545	0.718	0.570	0.628	0.663	0.588	0.599	0.600
SRCNN*	0.581	0.596	0.646	**0.568**	0.906	**0.604**	0.692	0.726	0.631	0.697	0.665
PnP-ADMM-TV	**0.602**	0.598	0.685	0.558	0.923	0.583	**0.751**	0.752	0.635	0.702	0.679
D-ADMM-TV	**0.602**	**0.599**	**0.687**	0.560	**0.926**	0.584	**0.751**	**0.753**	**0.636**	**0.703**	**0.680**

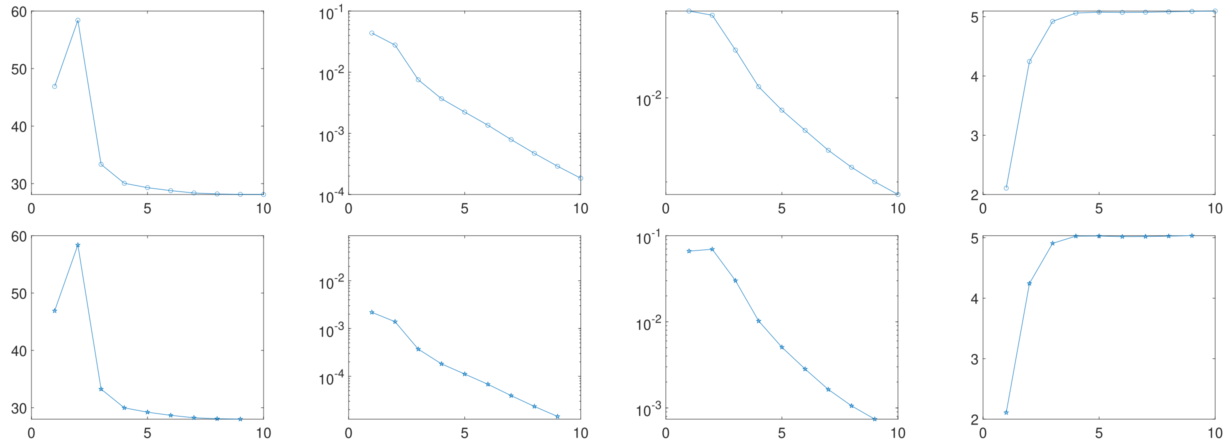
In summary, in all the image SR experiments, D-ADMM-TV can achieve comparable or occasionally slightly better quality results than PnP-ADMM-TV with fewer number of required iterations, given the same error tolerance, which demonstrates the power of combining the dual iterations with the primal iterations.

Note that for both of the 2× and 4× SR experiments, D-ADMM-TV has better performance than PnP-ADMM-TV for the No. 5 image. For those images with simpler structures, the gradient ascent method in ADMM is easily trapped by the larger flat areas near the true value given the same error tolerance, compared with our approach in the D-ADMM scheme.

B. Parameter Analysis

We now show how the parameters will influence the performance of D-ADMM. The first parameter is the error tolerance, which directly determines the number of iterations. The influence of the tolerance is shown in Fig. 6. We can see that the error tolerance only has a significant impact for the No. 5 image. A possible explanation is that the structure of this image is simple and lower error tolerance is required to achieve a high PSNR ($\geq$ 35 dB). Generally, when the error tolerance is lower than or equal to 10^{-3}, D-ADMM will give a stable result.

The most important parameters are the penalty parameters ρ_1 and ρ_2. We find that when ρ_2 is set to $1/\rho_1$, D-ADMM



(a) Objective function (b) Dual residual (c) Total residual (d) Data error

Fig. 5. Iterative results for 4× image super-resolution with the No. 4 image. The top row is PnP-ADMM-TV while the bottom row is D-ADMM-TV. x-axis: the number of iterations. y-axis: (a) value of the objective function in Eq. 3; (b) Dual residual $\frac{1}{\sqrt{p}}\|\boldsymbol{\mu}_1^{(k+1)} - \boldsymbol{\mu}_1^{(k)}\|_2$, which is the third term in Eq. 52; (c) Total residual expressed on the left side of Eq. 54 for D-ADMM-TV and Eq. 53 for PnP-ADMM-TV; (d) data-fidelity error $\|A\boldsymbol{x} - \boldsymbol{y}\|_2$. Note that D-ADMM-TV needs 9 iterations using the termination criteria in Eq. 54 but PnP-ADMM-TV needs 10 iterations using the termination criteria in Eq. 53, with the same error tolerance.

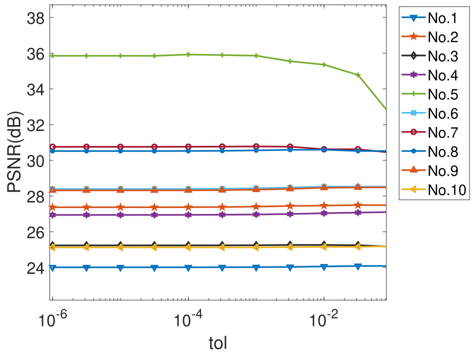


Fig. 6. Influence of the error tolerance. PSNR of the No. 5 image drops significantly when the tolerance level increases. For all the images, the PSNR becomes steady when the error tolerance is lower than or equal to 10^{-3}.

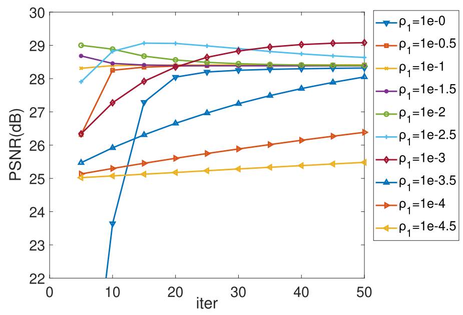


Fig. 7. Influence of ρ_1 (PSNR is calculated every 5 iterations). In this experiment, 2× SR results of the No. 6 image are used. Note that different values of the penalty parameter will lead to different iterative curves, hence a relatively precise value is recommended.

has better convergence. The influence of ρ_1 is summarized in Fig. 7. Similar to PnP-ADMM, the penalty parameters are important and require tuning based on a trial-and-error strategy or some other selection methods. It is recommended to set ρ_1 between 10^{-3} and 10^{-2}.

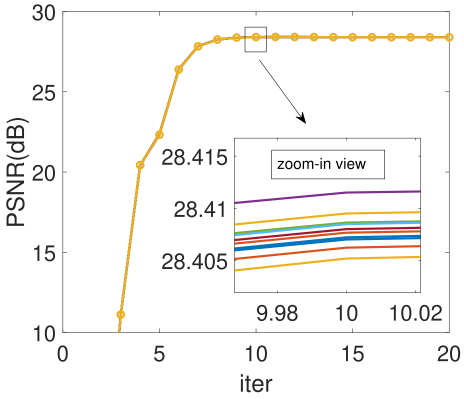


Fig. 8. Influence of the initial guesses. In this experiment, 2× SR results of the No. 6 image are used. The algorithm is initialized with 10 different random guesses between 0 and 1. A zoom-in view is provided to show the small fluctuations of the PSNR.

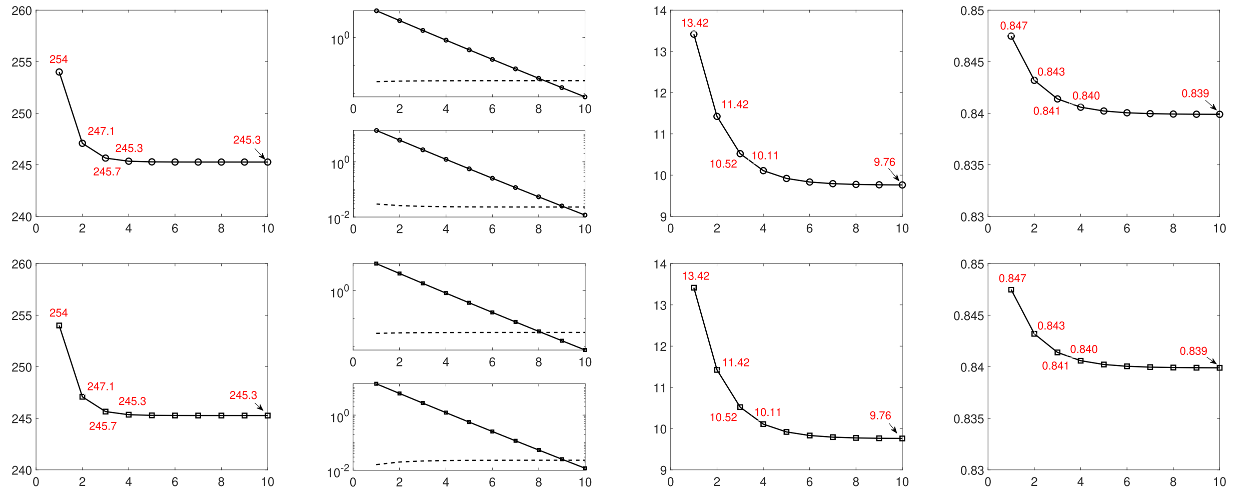
Finally, we analyze the influence of the initial values of the variables in D-ADMM. We randomly select 10 initial guesses for these variables and the results are shown in Fig. 8. As we can see, when D-ADMM-TV has converged, the difference is less than 0.01 on the PSNR, which indicates that D-ADMM is robust to the initial guesses.

C. Numerical Example: Random System

We now give a numerical example to compare between the proposed D-ADMM scheme and the conventional ADMM scheme directly in a strongly convex case. This example, called random system, comes from [18]. The code for ADMM is available online,[1] and the parameters in ADMM are set as given in the reference.

In this experiment, we consider a random matrix A with column normalization as the system matrix. The input signal

[1] https://web.stanford.edu/∼ boyd/papers/admm/



(a) Objective function (b) Residuals (c) Data error (d) Relative error

Fig. 9. Comparisons between D-ADMM-L2 and ADMM-L2 for a random system. The top row is D-ADMM-L2 and the bottom row is ADMM-L2, and x-axis represents the number of iterations. y-axis: (a) value of the objective function. (b) value of the primal residual (top) and dual residual (bottom), the dotted lines indicate the dynamic error tolerances. (c) value of data error $\|A\boldsymbol{x}-\boldsymbol{y}\|_2$; (d) value of relative error $\|\boldsymbol{x}-\boldsymbol{x_0}\|_2/\|\boldsymbol{x_0}\|_2$, where $\boldsymbol{x_0}$ represents the ground truth.

is a random vector with 5000 elements, and the measurement signal is a 1500-dimensional vector.

The convergence process of the proposed D-ADMM-L2 is shown in Fig. 9 (top row). Parameters are set as $\rho_1 = \rho_2 = 1.0$. This experiment takes the stopping criteria in Eq. 50-51. The tolerances are set as $\epsilon_{\text{abs}} = 10^{-4}$ and $\epsilon_{\text{rel}} = 10^{-3}$ in D-ADMM-L2. As a comparison, we use the conventional ADMM with ℓ_2 regularization function (ADMM-L2) to deal with the same problem with the same error tolerances, and the results are also shown in Fig. 9 (bottom row). In this example, D-ADMM-L2 and ADMM-L2 have the same performance, because the dual ascent steps and the dual problem iterations are the same during the iterations, as shown in (b).

V. Conclusion

In this paper, we present a new ADMM scheme called D-ADMM based on the duality theory. We first outline the method and show that when the regularization is convex, D-ADMM has good global convergence and fixed point convergence.

Furthermore, based on the commonly-used ℓ_2 and TV regularizations, we provide two realizations of our scheme, which are named D-ADMM-L2 and D-ADMM-TV. We consider image super-resolution as an inverse imaging application and compare the results of our methods with other techniques. D-ADMM-TV gives comparable or occasionally slightly better quality results than PnP-ADMM-TV with the same penalty parameter, and requires fewer number of iterations for the same error tolerance. This paves the way of incorporating advanced operators and strategies designed for ADMM into the basic D-ADMM method as well, to further improve the performances of those methods. Note that we should guarantee ADMM works in a good condition first with a proper penalty parameter. A numerical example is also conducted to give comparisons between the proposed method and the traditional ADMM scheme. We find that D-ADMM and ADMM have identical performance in some simple cases, such as strongly convex problem. We also show that proper denoisers can be plugged in the D-ADMM scheme directly which leads to the potential of combining D-ADMM with advanced denoisers.

Appendix A
Gradient Relationship Between a Convex Regularization Function and Its Conjugate Function

In this paper, we assume that for a regularization function $\mathcal{R}(\cdot)$, $\nabla\mathcal{R}(\cdot)$ and $\nabla\mathcal{R}^*(\cdot)$ exist. We have the lemma below.

Lemma 5.1: For a convex regularization function $\mathcal{R}(\boldsymbol{\lambda})$, its gradient satisfies the equation

$$\nabla\mathcal{R}(\nabla\mathcal{R}^*(\boldsymbol{\lambda})) = \boldsymbol{\lambda}, \tag{58}$$

where $\mathcal{R}^*(\cdot)$ is the Fenchel conjugate of $\mathcal{R}(\cdot)$.

Proof: The definition of the Fenchel conjugate function $\mathcal{R}^*(\boldsymbol{\lambda})$ is

$$y_r = \mathcal{R}^*(\boldsymbol{\lambda}) = \sup_{\boldsymbol{x}} \left\{\boldsymbol{\lambda}^T\boldsymbol{x} - \mathcal{R}(\boldsymbol{x})\right\}, \tag{59}$$

and a necessary condition for this optimization problem is

$$\boldsymbol{\lambda} - \nabla\mathcal{R}(\boldsymbol{x}) = 0. \tag{60}$$

If we want to find the explicit expression of $\mathcal{R}^*(\boldsymbol{x})$, then we need to solve Eq. 60 and obtain an expression of $\boldsymbol{x}$ with respect to $\boldsymbol{\lambda}$, which is often not easy. Instead, we can find the relationship of their gradients. According to Eq. 59 and the relationship between $\boldsymbol{\lambda}$ and $\boldsymbol{x}$, we have

$$\frac{dy_r}{d\boldsymbol{x}} = \frac{d\boldsymbol{\lambda}}{d\boldsymbol{x}}\boldsymbol{x} + \boldsymbol{\lambda} - \nabla\mathcal{R}(\boldsymbol{x}) = \frac{d\boldsymbol{\lambda}}{d\boldsymbol{x}}\boldsymbol{x}, \tag{61}$$

and $\nabla\mathcal{R}^*(\boldsymbol{\lambda})$ is

$$\nabla\mathcal{R}^*(\boldsymbol{\lambda}) = \frac{dy_r}{d\boldsymbol{\lambda}} = \frac{d\boldsymbol{x}}{d\boldsymbol{\lambda}}\frac{dy_r}{d\boldsymbol{x}} = \frac{d\boldsymbol{x}}{d\boldsymbol{\lambda}}\frac{d\boldsymbol{\lambda}}{d\boldsymbol{x}}\boldsymbol{x} = \boldsymbol{x}. \tag{62}$$

Finally

$$\nabla\mathcal{R}(\nabla\mathcal{R}^*(\boldsymbol{\lambda})) = \nabla\mathcal{R}(\boldsymbol{x}) = \boldsymbol{\lambda}, \tag{63}$$

which is identical to the equation in the lemma above. □

Appendix B
Complete Convergence Proof of D-ADMM

Based on the iterations in Algorithm 1, we have the lemma below.

Lemma 5.2: Variables in the $(k+1)^{th}$ iteration satisfy the following inequality: $\forall \boldsymbol{w} \in \mathbb{R}^{5N+M}$,

$$\begin{aligned}\theta(\boldsymbol{w}) - \theta(\boldsymbol{w}^{(k+1)}) + (\boldsymbol{w} - \boldsymbol{w}^{(k+1)})^T \Big[F(\boldsymbol{w}^{(k+1)}) \\ + Q(\boldsymbol{s}^{(k)} - \boldsymbol{s}^{(k+1)}) + H(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^{(k)})\Big] \geq 0,\end{aligned} \tag{64}$$

where

$$\begin{aligned}
\boldsymbol{w} &= \begin{bmatrix}\boldsymbol{c} & \boldsymbol{\lambda} & \boldsymbol{\mu_2} & \boldsymbol{z} & \boldsymbol{x} & \boldsymbol{\mu_1}\end{bmatrix}^T,\\
\boldsymbol{b} &= \begin{bmatrix}0 & \boldsymbol{y} & 0 & 0 & 0 & 0\end{bmatrix}^T,\\
F(\boldsymbol{w}) &= K\boldsymbol{w} + \boldsymbol{b}, \quad \boldsymbol{s} = \begin{bmatrix}\boldsymbol{c} & \boldsymbol{z}\end{bmatrix}^T,\\
\theta(\boldsymbol{w}) &= \mathcal{R}(\boldsymbol{z}) + \mathcal{R}^*(\boldsymbol{c}) + \frac{1}{2}\|\boldsymbol{\lambda}\|_2^2,\\
Q^T &= \begin{bmatrix}\rho_2 I_N & \rho_2 A^T & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & \rho_1 I_N & -\rho_1 I_N & 0\end{bmatrix},\\
K &= \begin{bmatrix}0 & 0 & I_N & 0 & 0 & 0\\ 0 & 0 & A & 0 & 0 & 0\\ -I_N & -A^T & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & I_N\\ 0 & A^T & 0 & 0 & 0 & -I_N\\ 0 & 0 & 0 & -I_N & I_N & 0\end{bmatrix},\\
H &= \begin{bmatrix}\rho_2 I_N & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & \frac{1}{\rho_2} I_N & 0 & 0 & 0\\ 0 & 0 & 0 & \rho_1 I_N & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & \frac{1}{\rho_1} I_N\end{bmatrix}
\end{aligned} \tag{65}$$

and I_N is the identity matrix of size $N \times N$.

Proof: Following the update strategy in Algorithm 1, the $\boldsymbol{c}$-update step is an optimization problem

$$\begin{aligned}\boldsymbol{c}^{(k+1)} = \arg\min_{\boldsymbol{c}} \Big\{\mathcal{R}^*(\boldsymbol{c}) + \boldsymbol{\mu_2}^{(k)T}\left(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}\right) \\ + \frac{\rho_2}{2}\|A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}\|_2^2\Big\},\end{aligned} \tag{66}$$

and then the solution satisfies the variational inequalities

$$\mathcal{R}^*(\boldsymbol{c}) - \mathcal{R}^*(\boldsymbol{c}^{(k+1)}) + (\boldsymbol{c} - \boldsymbol{c}^{(k+1)})^T\Big[\boldsymbol{\mu_2}^{(k)} \tag{67}$$

$$+\rho_2\left(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)}\right)\Big] \geq 0 \tag{68}$$

$$\mathcal{R}^*(\boldsymbol{c}) - \mathcal{R}^*(\boldsymbol{c}^{(k+1)}) + (\boldsymbol{c} - \boldsymbol{c}^{(k+1)})^T\boldsymbol{\mu_2}^{(k+1)} \geq 0. \tag{69}$$

Similarly, the other variables also have similar forms as

$$\begin{aligned}\frac{1}{2}\|\boldsymbol{\lambda}\|_2^2 - \frac{1}{2}\|\boldsymbol{\lambda}^{(k+1)}\|_2^2 + (\boldsymbol{\lambda} - \boldsymbol{\lambda}^{(k+1)})^T\Big[\boldsymbol{y} + A\boldsymbol{\mu_2}^{(k+1)} \\ +\rho_2 A(\boldsymbol{c}^{(k)} - \boldsymbol{c}^{(k+1)})\Big] \geq 0\end{aligned} \tag{70}$$

$$\mathcal{R}(\boldsymbol{z}) - \mathcal{R}(\boldsymbol{z}^{(k+1)}) + (\boldsymbol{z} - \boldsymbol{z}^{(k+1)})^T\boldsymbol{\mu_1}^{(k+1)} \geq 0 \tag{71}$$

$$\begin{aligned}(\boldsymbol{x} - \boldsymbol{x}^{(k+1)})^T\Big[A^T\boldsymbol{\lambda}^{(k+1)} - \boldsymbol{\mu_1}^{(k+1)} - \rho_1(\boldsymbol{z}^{(k)} \\ - \boldsymbol{z}^{(k+1)})\Big] \geq 0.\end{aligned} \tag{72}$$

Besides, the dual variables have simpler update steps, and can be reformulated as

$$\begin{aligned}(\boldsymbol{\mu_1} - \boldsymbol{\mu_1}^{(k+1)})^T\Big[\frac{1}{\rho_1}(\boldsymbol{\mu_1}^{(k+1)} - \boldsymbol{\mu_1}^{(k)}) \\ -(\boldsymbol{z}^{(k+1)} - \boldsymbol{x}^{(k+1)})\Big] \geq 0\end{aligned} \tag{73}$$

$$\begin{aligned}(\boldsymbol{\mu_2} - \boldsymbol{\mu_2}^{(k+1)})^T\Big[\frac{1}{\rho_2}(\boldsymbol{\mu_2}^{(k+1)} - \boldsymbol{\mu_2}^{(k)}) \\ -(A^T\boldsymbol{\lambda}^{(k+1)} + \boldsymbol{c}^{(k+1)})\Big] \geq 0.\end{aligned} \tag{74}$$

Combining Eq. 69-74, Lemma 5.2 can be obtained. □

The optimal solution must have the property below.

Lemma 5.3: When $\boldsymbol{w}^*$ is the optimal solution, then

$$\forall\boldsymbol{w}, \quad \theta(\boldsymbol{w}) - \theta(\boldsymbol{w}^*) + (\boldsymbol{w} - \boldsymbol{w}^*)^T F(\boldsymbol{w}^*) \geq 0, \tag{75}$$

where $F(\cdot)$ has the same form as in Lemma 5.2.

Lemma 5.3 can be easily obtained if we rewrite the whole optimization problem as a variational inequality problem [52].

To give an evaluation of the current result, a measurement between the current solution and the optimal solution should be given first. Here, we make use of the H-norm, which is defined by the following:

Definition 5.1: H-norm:

$$\|\boldsymbol{v}\|_H^2 = \boldsymbol{v}^T H\boldsymbol{v}, \tag{76}$$

where H is defined in Lemma 5.2 and ρ_1, ρ_2 should be greater than 0 to guarantee that H is positive semi-definite.

Now we consider the property of the linear function $F(\boldsymbol{w})$ and have the lemma below.

Lemma 5.4: If $\boldsymbol{w}^*$ is the optimal solution, then the current solution $\boldsymbol{w}^{(k+1)}$ after the $(k+1)^{th}$ iteration satisfies

$$(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)^T F(\boldsymbol{w}^{(k+1)}) \geq (\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)^T F(\boldsymbol{w}^*). \tag{77}$$

Proof: From Eq. 17, we can obtain that actually the $\boldsymbol{e}$-update step is

$$\boldsymbol{e}^{(k+1)} = -\boldsymbol{\lambda}^{(k+1)}, \tag{78}$$

then $\boldsymbol{e}^* = -\boldsymbol{\lambda}^*$.

We consider the difference of the two sides as

$$\begin{aligned}&(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)^T(F(\boldsymbol{w}^{(k+1)}) - F(\boldsymbol{w}^*))\\
&= (\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)^T K(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)\\
&= \frac{1}{2}\Big[(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)^T(K + K^T)(\boldsymbol{w}^{(k+1)} - \boldsymbol{w}^*)\Big]\\
&= -(\boldsymbol{\lambda}^* - \boldsymbol{\lambda}^{(k+1)})^T A(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^*)\\
&= -(\boldsymbol{\lambda}^* - \boldsymbol{\lambda}^{(k+1)})^T(A\boldsymbol{x}^{(k+1)} + \boldsymbol{e}^{(k+1)} - A\boldsymbol{x}^* - \boldsymbol{e}^*)\end{aligned}$$

$$\begin{aligned}&+(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)})^T(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)})\\ &=(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)})^T[-(A\boldsymbol{x}^{(k+1)}+\boldsymbol{e}^{(k+1)}-\boldsymbol{y})]\\ &\quad+(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)})^T(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)}). \qquad (79)\end{aligned}$$

Since $\boldsymbol{\lambda} = -\boldsymbol{e}$, then in the $(k+1)^{th}$ iteration

$$(\boldsymbol{\lambda}^*-\boldsymbol{\lambda}^{(k+1)})^T[-(A\boldsymbol{x}^{(k+1)}+\boldsymbol{e}^{(k+1)}-\boldsymbol{y})] \geq 0. \qquad (80)$$

and we can get the lemma above. □

The term related to Q in Lemma 5.2 has the property below.

Lemma 5.5: Let $\boldsymbol{s}^{(k+1)}$ and $\boldsymbol{w}^{(k+1)}$ be the generated results from $\boldsymbol{s}^{(k)}$ and $\boldsymbol{w}^{(k)}$. We have

$$(\boldsymbol{s}^{(k)}-\boldsymbol{s}^{(k+1)})^T Q^T(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*) \geq 0. \qquad (81)$$

Proof: According to the definition of Q,

$$\begin{aligned}&Q^T(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)\\ &=\begin{bmatrix}\rho_2(\boldsymbol{c}^{(k+1)}-\boldsymbol{c}^*)+\rho_2A^T(\boldsymbol{\lambda}^{(k+1)}-\boldsymbol{\lambda}^*)\\ \rho_1(\boldsymbol{z}^{(k+1)}-\boldsymbol{z}^*)-\rho_1(\boldsymbol{x}^{(k+1)}-\boldsymbol{x}^*)\end{bmatrix}\\ &=\begin{bmatrix}\boldsymbol{\mu_2}^{(k+1)}-\boldsymbol{\mu_2}^{(k)}\\ \boldsymbol{\mu_1}^{(k+1)}-\boldsymbol{\mu_1}^{(k)}\end{bmatrix}.\end{aligned}$$

In the k^{th} and $(k+1)^{th}$ $\boldsymbol{c}$-update step, we have

$$\begin{aligned}\mathcal{R}^*(\boldsymbol{c}^{(k)})-\mathcal{R}^*(\boldsymbol{c}^{(k+1)})+(\boldsymbol{c}^{(k)}-\boldsymbol{c}^{(k+1)})^T\boldsymbol{\mu_2}^{(k+1)} \geq 0.\\ \mathcal{R}^*(\boldsymbol{c}^{(k+1)})-\mathcal{R}^*(\boldsymbol{c}^{(k)})+(\boldsymbol{c}^{(k+1)}-\boldsymbol{c}^{(k)})^T\boldsymbol{\mu_2}^{(k)} \geq 0.\end{aligned} \qquad (82)$$

Adding them up, we have

$$(\boldsymbol{c}^{(k)}-\boldsymbol{c}^{(k+1)})^T(\boldsymbol{\mu_2}^{(k+1)}-\boldsymbol{\mu_2}^{(k)}) \geq 0. \qquad (83)$$

Similarly,

$$(\boldsymbol{z}^{(k)}-\boldsymbol{z}^{(k+1)})^T(\boldsymbol{\mu_1}^{(k+1)}-\boldsymbol{\mu_1}^{(k)}) \geq 0. \qquad (84)$$

Combining them together, we obtain Lemma 5.5. □

Finally, the global convergence property is given in Theorem 5.1.

Theorem 5.1 (Global Convergence): During the iteration process, the distance between the current solution and the optimal solution decreases monotonically

$$\|\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*\|_H^2 \leq \|\boldsymbol{w}^{(k)}-\boldsymbol{w}^*\|_H^2-\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}\|_H^2 \qquad (85)$$

Proof: According to Lemma 5.2, we have

$$\begin{aligned}&(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)^T H(\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)})\\ &\geq \theta(\boldsymbol{w}^{(k+1)})-\theta(\boldsymbol{w}^*)+(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)^T F(\boldsymbol{w}^{(k+1)})\\ &\quad+(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)^T Q(\boldsymbol{s}^{(k)}-\boldsymbol{s}^{(k+1)}). \qquad (86)\end{aligned}$$

Lemma 5.3, 5.4 and 5.5 show that all the terms on the right hand side are nonnegative, and therefore

$$(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)^T H(\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}) \geq 0. \qquad (87)$$

Finally,

$$\begin{aligned}\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^*\|_H^2 &= \|(\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*)+(\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)})\|_H^2\\ &\geq \|\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*\|_H^2+\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}\|_H^2.\end{aligned} \qquad (88)$$

□

Note that there are two zero rows in H, which are related to the $\boldsymbol{\lambda}$ and $\boldsymbol{x}$ update steps. As shown in Algorithm 1, when the other four variables converge, these two variables will also converge, as the values of these two variables are totally determined by the others.

The fixed point convergence can be obtained easily from Theorem 5.1 as given below.

Theorem 5.2 (Fixed Point Convergence): The distance between the solutions of two successive iterations will converge to 0 as

$$\lim_{k\to\infty}\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}\|_H^2 = 0. \qquad (89)$$

Proof: According to Theorem 5.1, we have

$$\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}\|_H^2 \leq \|\boldsymbol{w}^{(k)}-\boldsymbol{w}^*\|_H^2-\|\boldsymbol{w}^{(k+1)}-\boldsymbol{w}^*\|_H^2. \qquad (90)$$

Summing up all values of k, it becomes

$$\sum_{k=0}^{\infty}\|\boldsymbol{w}^{(k)}-\boldsymbol{w}^{(k+1)}\|_H^2 \leq \|\boldsymbol{w}^{(0)}-\boldsymbol{w}^*\|_H^2. \qquad (91)$$

Since $\|\boldsymbol{w}^{(0)}-\boldsymbol{w}^*\|_H^2$ is a fixed value, the limit of the nonnegative series on the left hand side must be zero. □

REFERENCES

[1] D. J. Brady, *Optical Imaging and Spectroscopy*. Hoboken, NJ, USA: Wiley, 2009.

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

[3] A. Lucas, M. Iliadis, R. Molina, and A. K. Katsaggelos, "Using deep neural networks for inverse problems in imaging: Beyond analytical methods," *IEEE Trans. Signal Process. Mag.*, vol. 35, no. 1, pp. 20–36, Jan. 2018.

[4] Y. Censor, "Finite series-expansion reconstruction methods," *Proc. IEEE*, vol. 71, no. 3, pp. 409–419, Mar. 1983.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[6] A. Buades, B. Coll, and J.-M. Morel, "Nonlocal image and movie denoising," *Int. J. Comput. Vis.*, vol. 76, no. 2, pp. 123–139, Jul. 2007.

[7] H. Chang, D.-Y. Yeung, and Y. Xiong, "Super-resolution through neighbor embedding," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2004, pp. 1–8.

[8] J. Yang, J. Wright, T. Huang, and Y. Ma, "Image super-resolution as sparse representation of raw image patches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[9] J. Yang, J. Wright, T. S. Huang, and Y. Ma, "Image super-resolution via sparse representation," *IEEE Trans. Image Process.*, vol. 19, no. 11, pp. 2861–2873, Nov. 2010.

[10] R. Fergus, B. Singh, A. Hertzmann, S. T. Roweis, and W. T. Freeman, "Removing camera shake from a single photograph," in *Proc. ACM SIGGRAPH Papers (SIGGRAPH)*, 2006, pp. 787–794.

[11] L. Yang and H. Ji, "A variational EM framework with adaptive edge selection for blind motion deblurring," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10167–10176.

[12] Z. Ren, Z. Xu, and E. Y. Lam, "End-to-end deep learning framework for digital holographic reconstruction," *Adv. Photon.*, vol. 1, no. 1, pp. 1–12, 2019.

[13] N. Meng, H. K.-H. So, X. Sun, and E. Y. Lam, "High-dimensional dense residual convolutional neural network for light field reconstruction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 3, pp. 873–886, Mar. 2021.

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

[15] Y. Yang, J. Sun, H. Li, and Z. Xu, "ADMM-CSNet: A deep learning approach for image compressive sensing," *IEEE Trans. Pattern Anal. Mach. Intell.* vol. 42, no. 3, pp. 521–538, Mar. 2020.

[16] H. K. Aggarwal, M. P. Mani, and M. Jacob, "Modl: Model-based deep learning architecture for inverse problems," *IEEE Trans. Med. Imag.*, vol. 38, no. 2, pp. 394–405, Feb. 2019.

[17] Z. Xu and E. Y. Lam, "Maximum *a posteriori* blind image deconvolution with Huber–Markov random-field regularization," *Opt. Lett.*, vol. 34, no. 9, pp. 1453–1455, 2009.

[18] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, Jul. 2010.

[19] J. Heredia Juesas, G. Allan, A. Molaei, L. Tirado, W. Blackwell, and J. A. Martinez Lorenzo, "Consensus-based imaging using ADMM for a compressive reflector antenna," in *Proc. IEEE Int. Symp. Antennas Propag. USNC/URSI Nat. Radio Sci. Meeting*, Jul. 2015, pp. 1304–1305.

[20] A. Sawatsky, Q. Xu, C. O. Schirra, and M. A. Anastasio, "Proximal ADMM for multi-channel image reconstruction in spectral X-ray CT," *IEEE Trans. Med. Imag.*, vol. 33, no. 8, pp. 1657–1668, Aug. 2014.

[21] Y. Liu, F. Shang, and J. Cheng, "Accelerated variance reduced stochastic ADMM," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2287–2293.

[22] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "An augmented Lagrangian approach to the constrained optimization formulation of imaging inverse problems," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 681–695, Mar. 2011.

[23] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, pp. 127–239, Jan. 2014.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[25] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, "Plug-and-play priors for model based reconstruction," in *Proc. IEEE Global Conf. Signal Inf. Process.*, Dec. 2013, pp. 945–948.

[26] 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, Jan. 2017.

[27] U. S. Kamilov, H. Mansour, and B. Wohlberg, "A plug-and-play priors approach for solving nonlinear imaging inverse problems," *IEEE Signal Process. Lett.*, vol. 24, no. 12, pp. 1872–1876, Dec. 2017.

[28] J. He *et al.*, "Optimizing a parameterized plug-and-play ADMM for iterative low-dose CT reconstruction," *IEEE Trans. Med. Imag.*, vol. 38, no. 2, pp. 371–382, Feb. 2019.

[29] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D, Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, 1992.

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

[31] 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. (CVPR)*, Jul. 2017, pp. 3929–3938.

[32] Y. Romano, M. Elad, and P. Milanfar, "The little engine that could: Regularization by denoising (RED)," *SIAM J. Imag. Sci.*, vol. 10, no. 4, pp. 1804–1844, Oct. 2017.

[33] G. Song, Y. Sun, J. Liu, Z. Wang, and U. S. Kamilov, "A new recurrent plug-and-play prior based on the multiple self-similarity network," *IEEE Signal Process. Lett.*, vol. 27, pp. 451–455, 2020.

[34] F. Latorre, A. Eftekhari, and V. Cevher, "Fast and provable ADMM for learning with generative priors," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 12027–12039.

[35] T. Sun, R. Barrio, M. Rodriguez, and H. Jiang, "Inertial nonconvex alternating minimizations for the image deblurring," *IEEE Trans. Image Process.*, vol. 28, no. 12, pp. 6211–6224, Dec. 2019.

[36] G.-J. Peng, "Adaptive ADMM for dictionary learning in convolutional sparse representation," *IEEE Trans. Image Process.*, vol. 28, no. 7, pp. 3408–3422, Jul. 2019.

[37] R. I. Bot, S.-M. Grad, and G. Wanka, *Duality in Vector Optimization*. Springer, 2009.

[38] J. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Examples*. Springer, 2010.

[39] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. New York, NY, USA: Academic, 2014.

[40] R. T. Rockafellar, *Conjugate Duality and Optimization*. Philadelphia, PA, USA: SIAM, 1974.

[41] N. Komodakis and J.-C. Pesquet, "Playing with duality: An overview of recent primal?Dual approaches for solving large-scale optimization problems," *IEEE Signal Process. Mag.*, vol. 32, no. 6, pp. 31–54, Nov. 2015.

[42] A. B. Jiménez and S. Sra, "Fast Newton-type methods for total variation regularization," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 313–320.

[43] A. Barbero and S. Sra, "Modular proximal optimization for multidimensional total-variation regularization," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 2232–2313, 2018.

[44] 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.

[45] W. Zhong and J. Kwok, "Fast stochastic alternating direction method of multipliers," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 46–54.

[46] M. Irani and S. Peleg, "Improving resolution by image registration," *CVGIP, Graph. Models Image Process.*, vol. 53, no. 3, pp. 231–239, May 1991.

[47] Z. Ren, H. K.-H. So, and E. Y. Lam, "Fringe pattern improvement and super-resolution using deep learning in digital holography," *IEEE Trans. Ind. Informat.*, vol. 15, no. 11, pp. 6179–6186, Nov. 2019.

[48] S. Sreehari *et al.*, "Plug-and-play priors for bright field electron tomography and sparse interpolation," *IEEE Trans. Comput. Imag.*, vol. 2, no. 4, pp. 408–423, Dec. 2016.

[49] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5197–5206.

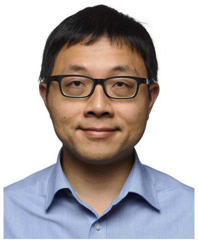
[50] C. Dong, C. C. Loy, K. He, and X. Tang, "Learning a deep convolutional network for image super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 184–199.

[51] 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.*, Jun. 2009, pp. 248–255.

[52] B. He, "A class of projection and contraction methods for monotone variational inequalities," *Appl. Math. Optim.*, vol. 35, no. 1, pp. 69–76, Jan. 1997.

Li Song received the B.S. degree in automation from the Beijing Institute of Technology in 2019. He is currently pursuing the Ph.D. degree with the Department of Electrical and Electronic Engineering, The University of Hong Kong. His research interests include computational imaging, convex optimization, and phase retrieval.

Zhou Ge received the B.S. degree from Fudan University in 2014 and the M.S. degree from Imperial College London in 2015, both in communication engineering and electrical engineering. He is currently pursuing the Ph.D. degree with the Department of Electrical and Electronic Engineering, The University of Hong Kong. His research interests include computational imaging, image processing, and event sensing.



Edmund Y. Lam (Fellow, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from Stanford University. He was a Visiting Associate Professor with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology. He is currently a Professor of electrical and electronic engineering at The University of Hong Kong. He also serves as the Computer Engineering Program Director and a Research Program Coordinator with the AI Chip Center for Emerging Smart Systems. His research interest includes computational imaging algorithms, systems, and applications. He is a fellow of Optica, SPIE, IS&T, and HKIE; and a Founding Member of the Hong Kong Young Academy of Sciences.