Deep Equilibrium Models for Snapshot Compressive Imaging

Yaping Zhao[1,2], **Siming Zheng**[3,4], **Xin Yuan**[1,*]

[1] Westlake University, Hangzhou, China
[2] The University of Hong Kong, Pokfulam, Hong Kong SAR, China
[3] Computer Network Information Center, Chinese Academy of Science, Beijing, China
[4] University of Chinese Academy of Sciences, Beijing, China
zhaoyp@connect.hku.hk, zhengsiming@cnic.cn, xyuan@westlake.edu.cn

Abstract

The ability of snapshot compressive imaging (SCI) systems to efficiently capture high-dimensional (HD) data has led to an inverse problem, which consists of recovering the HD signal from the compressed and noisy measurement. While reconstruction algorithms grow fast to solve it with the recent advances of deep learning, the fundamental issue of accurate and **stable** recovery remains. To this end, we propose deep equilibrium models (DEQ) for video SCI, fusing data-driven regularization and stable convergence in a theoretically sound manner. Each equilibrium model implicitly learns a nonexpansive operator and analytically computes the fixed point, thus enabling unlimited iterative steps and infinite network depth with only a **constant memory** requirement in training and testing. Specifically, we demonstrate how DEQ can be applied to two existing models for video SCI reconstruction: recurrent neural networks (RNN) and Plug-and-Play (PnP) algorithms. On a variety of datasets and real data, both quantitative and qualitative evaluations of our results demonstrate the **effectiveness and stability** of our proposed method. The code and models are available at: https://github.com/IndigoPurple/DEQSCI

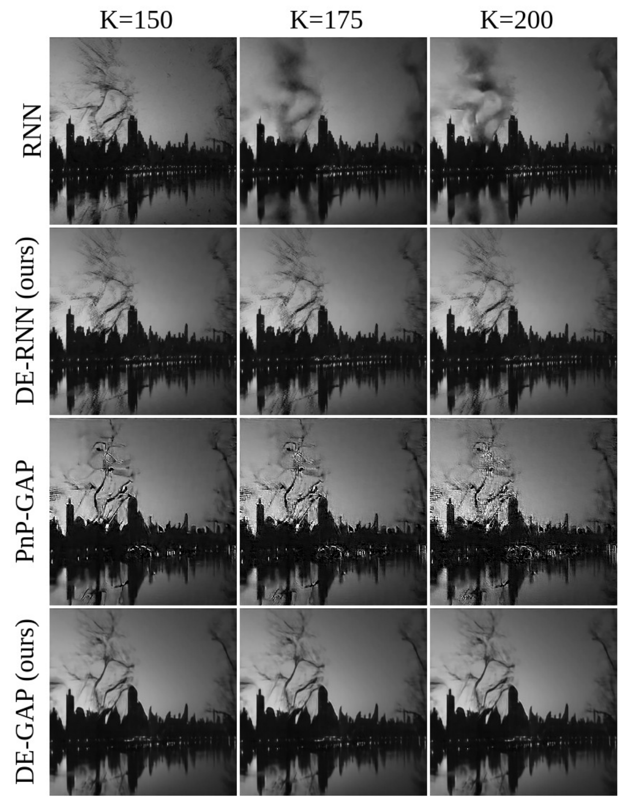

Figure 1: Our proposed deep equilibrium models (DEQ) for SCI can lead to stable recovery as K increases, where K denotes the iteration number during the corresponding optimization progress. We test our model under two different frameworks, *i.e.*, RNN (Cheng et al. 2020) and PnP-GAP (Yuan et al. 2020), the fidelity and stability of our model can be obviously observed.

Introduction

Aiming at the efficient and effective acquisition of high-dimensional (HD) visual signal, snapshot compressive imaging (SCI) systems have benefited from the advent of novel optical designs to sample the HD data as two-dimensional (2D) measurements. Considering the video SCI system, the 2D measurement of a video, *i.e.*, a three-dimensional (3D) data-cube leads to an inverse problem. The goal of such an inverse problem is to recover a video from a collection of noisy snapshots, which could be modeled as (Yuan, Brady, and Katsaggelos 2021):

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

where $\boldsymbol{y} \in \mathbb{R}^n$ is the 2D measurement with n equaling the number of each video frame's pixels, $\boldsymbol{\Phi} \in \mathbb{R}^{n \times nB}$ is the sensing matrix, $\boldsymbol{x} \in \mathbb{R}^{nB}$ is the 3D data (by vectorizing each frame and stacking them), and $\boldsymbol{e}$ is the measurement noise; here B denotes that every B video frames are collapsed into a single 2D measurement. Though algorithms have been fully developed to reconstruct the video from its snapshot measurement in recent years, the fundamental issue remains: this inverse problem is inherently ill-posed, which makes the recovery of the signal x inaccurate and **unstable** for noise-affected data $\boldsymbol{y}$ (Jalali and Yuan 2019).

The rapid advancement of deep learning and artificial in-

[*]Corresponding author.

telligence have empowered a new wave of revolutionary solutions towards these previously intractable problems. For instance, BIRNAT (Cheng et al. 2020) employed recurrent neural networks (RNNs) to reconstruct the video frames in a sequential manner and explore the temporal correlation within the video SCI signal. Inspired by particular optimization algorithms, GAP-net (Meng, Jalali, and Yuan 2020), DUN-3DUnet (Wu, Zhang, and Mou 2021) designed deep unfolding structures, which consist of a fixed number of architecturally identical blocks. The heart of RNN and deep unfolding are deep neural networks, which have posed new challenges due to their ever-growing depth and huge training memory occupation. To overcome these difficulties, inspired by (Gomez et al. 2017), a recent work (RevSCI) (Cheng et al. 2021) utilized reversible convolutional neural networks to develop a memory-efficient structure. However, all of these aforementioned algorithms inevitably suffer growing memory occupation with increasing layer depth, and thus models need to be painstakingly designed.

Inspired by the plug-and-play (PnP) framework (Venkatakrishnan, Bouman, and Wohlberg 2013; Sreehari et al. 2016) which has been proposed for inverse problems with provable convergence (Chan, Wang, and Elgendy 2017; Ryu et al. 2019), PnP-FFDNet (Yuan et al. 2020) and PnP-FastDVDNet (Yuan et al. 2021) bridged the gap between deep learning and conventional optimization algorithms with the plug-and-play (PnP) framework, utilizing a pre-trained denoiser as the proximal operator. While enjoying the advantages of both data-driven regularization and flexible iterative optimization steps, those algorithms still have hyperparameters to be tuned. Nevertheless, an accurate result must be guaranteed with a proper parameter setting. Due to the intrinsic unstable characteristic of the iterative recovery, even some complicated strategy needs to be employed (Wei et al. 2020). As we illustrate in Fig. 1 and Fig. 2, the hyperparameters are unavoidable to be handcrafted to achieve satisfactory performance in traditional algorithms.

An important and interesting research topic in deep learning is to train arbitrary deep networks, in which the deep equilibrium models (DEQ) (Bai, Koltun, and Kolter 2020) stands up as the leading method. A recent work (Gilton, Ongie, and Willett 2021) leverages DEQ to solve the inverse problems in imaging, which corresponds to the potentially infinite number of iteration steps in the PnP scheme.

To accommodate the state-of-the-art SCI architectures and to enable **low-memory stable** reconstruction, this paper sets about utilizing DEQ for solving the inverse problem of video SCI. Specifically, we applied DEQ to two existing models for video SCI reconstruction: RNN and PnP. Therefore, the former one is equivalent to an infinite-depth network using only constant memory; the latter one is tuning-free, and directly solves for the fixed point during the iterative optimization process. On a variety of simulation and real datasets, quantitative and qualitative evaluations demonstrate the effectiveness of our proposed method. As shown in Fig. 2, our reconstruction converges to stable results along with the increasing iterations during optimization.

In a nutshell, we aim to address the following two challenges which the SCI reconstruction are facing while using deep neural network and iterative optimization algorithms:

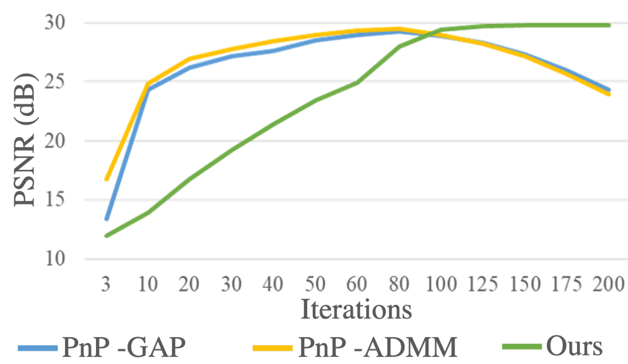

Figure 2: The quantitative comparison of different frameworks with or without our proposed DEQ for SCI. The convergence trends of different algorithms demonstrate that our model's results can converge to a higher level.

- How deep should the model be? Can it be infinite?
- Is there a tuning-free framework to be used? If yes, how to use it for SCI reconstruction?

By employing the most recent development of DEQ, we demonstrate that the answers to all the above questions are positive. Our specific contributions are as follows:

1) We firstly propose deep equilibrium models for video SCI, which fuses data-driven regularization and **stable** convergence in a theoretically sound manner.
2) Each equilibrium model analytically computes the fixed point, thus enabling unlimited iterative steps and infinite network depth with only a **constant memory** requirement in training and testing.
3) We analyze convergence for each equilibrium model, to ensure the implicit operators in our models are nonexpansive.
4) On a variety of simulations and real datasets, both quantitative and qualitative evaluations of our results demonstrate the **effectiveness and stability** of our proposed method.

Related Work

Snapshot Compressive Imaging

The underlying principle of SCI is to compress the 3D data cube into a 2D measurement by hardware, and then reconstruct the desired signal by algorithms. Considering video SCI, it compresses the spatio-temporal data-cube across the temporal dimension, and thus enables a low-speed camera to capture high-speed scenes. For instance, Llull *et al.* (Llull et al. 2013) proposed the coded aperture compressive temporal imaging (CACTI) system, which decomposes the 3D cube into its constituent 2D frames and imposes 2D masks for modulation.

Given the masks and measurements, plenty of algorithms including conventional optimization(Liu et al. 2018; Yang et al. 2015, 2014; Yuan 2016), end-to-end deep learning (Qiao et al. 2020; Zheng et al. 2021; Wang et al. 2022; Cheng et al. 2022; Meng and Yuan 2021), deep unfolding (Meng, Jalali, and Yuan 2020; Wu, Zhang, and Mou 2021) and plug-and-play (Yuan et al. 2020; Yuan et al. 2021; Wu et al. 2022; Yang and Zhao 2022) are proposed for reconstruction. To solve the ill-posed problem in Eq. (1), additional regularization is usually needed to ensure accurate and stable recovery with respect to noise perturbation. To this end, these algorithms obtain the estimated value $\hat{\boldsymbol{x}}$ of $\boldsymbol{x}$ by solving the following problem:

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{1}{2}||\boldsymbol{y} - \boldsymbol{\Phi x}||_2^2 + R(\boldsymbol{x}), \tag{2}$$

where $||\boldsymbol{y} - \boldsymbol{\Phi x}||_2^2$ is the fidelity term and $R(\boldsymbol{x})$ is the regularization term.

By introducing an auxiliary parameter $\boldsymbol{v}$, the unconstrained optimization in Eq. (2) can be converted into:

$$(\boldsymbol{x}, \boldsymbol{v}) = \arg\min_{\boldsymbol{x},\boldsymbol{v}} \frac{1}{2}||\boldsymbol{y} - \Phi\boldsymbol{x}||_2^2 + R(\boldsymbol{v}),\ s.t.\ \boldsymbol{x} = \boldsymbol{v}. \tag{3}$$

Using the alternating direction method of multipliers (ADMM) (Boyd et al. 2011) and introducing another parameter $\boldsymbol{u}$, Eq. (3) could be divided into the following sequence of sub-problems:

$$\boldsymbol{x}^{(k+1)} = \arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{\Phi x}\|_2^2 + \frac{\rho}{2}\|\boldsymbol{x} - (\boldsymbol{v}^{(k)} - \frac{1}{\rho}\boldsymbol{u}^{(k)})\|_2^2, \tag{4}$$

$$\boldsymbol{v}^{(k+1)} = \arg\min_{\boldsymbol{v}} \mu R(\boldsymbol{v}) + \tfrac{\rho}{2}||\boldsymbol{v} - (\boldsymbol{x}^{(k)} + \tfrac{1}{\rho}\boldsymbol{u}^{(k)})||_2^2, \tag{5}$$

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \rho(\boldsymbol{x}^{(k+1)} - \boldsymbol{v}^{(k+1)}), \tag{6}$$

where the superscript ^{k} denotes the iteration number; ρ is the penalty parameter and μ is the regularization weight. Since Eq. (5) can be regarded as a denoising process of $\boldsymbol{v}$, implicitly we have:

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}^{(k+1)}(\boldsymbol{x}^{(k+1)} + \tfrac{1}{\rho}\boldsymbol{u}^{(k)}), \tag{7}$$

where $\mathcal{D}$ is a denoiser.

On the other hand, generalized alternating projection (GAP) (Liao, Li, and Carin 2014) can be used as a (little bit) lower computational workload algorithm with the following two steps:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{v}^{(k)} + \boldsymbol{\Phi}^\top(\boldsymbol{\Phi\Phi}^\top)^{-1}(\boldsymbol{y} - \boldsymbol{\Phi v}^{(k)}), \tag{8}$$

$$\boldsymbol{v}^{(k+1)} = \mathcal{D}^{(k+1)}(\boldsymbol{x}^{(k+1)}). \tag{9}$$

Eq. (8) can be solved efficiently due to the special structure of $\boldsymbol{\Phi}$ in SCI (Jalali and Yuan 2019).

Deep Unfolding

Inspired by optimization algorithms such as ADMM (Boyd et al. 2011) and GAP (Liao, Li, and Carin 2014), deep unfolding methods (Ma et al. 2019; Meng, Jalali, and Yuan 2020; Wu, Zhang, and Mou 2021; Yang, Zhang, and Yuan 2022) are proposed to solve inverse problems in SCI, which consist of a fixed number of architecturally identical blocks. Each of those blocks represents a single iterative step in conventional optimization algorithms. Though deep unfolding successfully assimilate the advantages of the iterative optimization algorithms and could be trained in an end-to-end manner, the fixed number of network blocks in deep unfolding is needed to be kept small for two reasons: i) these systems should be concise to keep a high inference speed for real-time reconstruction; ii) it is challenging to train deep unfolding networks for numerous stages due to memory limitations.

Plug-and-Play

The latest trend is to bridge the gap between deep learning and optimization with the PnP framework. Yuan *et al.* (Yuan et al. 2021) proposed PnP-ADMM framework and PnP-GAP framework, using a pre-trained denoiser as the proximal operator in Eq. (5) and Eq. (9), respectively. In contrast to deep unfolding, PnP relieves itself from the limited memory by integrating a flexible denoising module into the iterative optimization process. Nevertheless, it suffers manual parameter tuning in addition to the time-consuming reconstruction process. That is, its performance is highly sensitive to the internal parameter selection, including but not limited to the penalty parameter, the denoising level, and the terminal step number. Moreover, the optimal parameter setting differs image-by-image, depending on the modulation process, noise level, noise type, and the unknown image itself.

Memory-Efficient Deep Networks

Since the important factor that limits the development of deep learning and deep unfolding for SCI is limited memory on hardware devices used for training, to address this issue, RevSCI (Cheng et al. 2021) developed a memory-efficient network for large-scale video SCI. Using reversible neural networks, where each layer's input can be calculated from the layer's activation during back-propagation, which means the activation during training is not needed to be stored. Nevertheless, it still suffers growing memory occupation along with the increasing depth of the network. In contrast, DEQ reduces memory consumption to a constant (*i.e.*, independent of network depth) by directly differentiating through the equilibrium point and thus circumvents the construction and maintenance of layers. Moreover, DEQ can solve stable estimation, easily extended to larger computing in the test time, while reversible neural networks cannot.

Deep Equilibrium Models

Motivated by the surprisingly recent works (Bai, Kolter, and Koltun 2018; Dehghani et al. 2018; Dabre and Fujita 2019) that employ the same transformation in each layer and still achieve competitive results with the state-of-the-art, Bai *et al.* (Bai, Kolter, and Koltun 2019) proposed a new approach to model this process and directly computed the fixed point. To leverage ideas from DEQ, Gilton *et al.* (Gilton, Ongie, and Willett 2021) proposed DEQ for inverse problems in imaging, which corresponds to a potentially infinite number of iteration steps in the PnP reconstruction scheme.

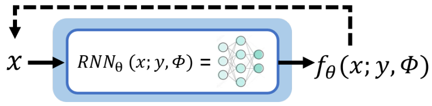

Figure 3: Illustration of our proposed DEQ for SCI using recurrent neural network (RNN), *i.e.*, DE-RNN.

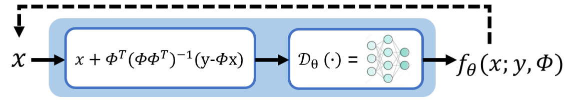

Figure 4: Illustration of our proposed DEQ for SCI using generalized alternating projection (GAP), *i.e.*, DE-GAP.

Method

Given measurement $\boldsymbol{y} \in \mathbb{R}^n$ with compression rate B and sensing matrix $\boldsymbol{\Phi} \in \mathbb{R}^{n \times nB}$ as input, we consider an optimization iteration or neural network as:

$$\boldsymbol{x}^{(k+1)} = f_\theta(\boldsymbol{x}^{(k)}; \boldsymbol{y}, \boldsymbol{\Phi}), \quad k = 0, 1, \ldots, \infty, \tag{10}$$

where θ denotes the weights of embedded neural networks; $\boldsymbol{x}^{(k)} \in \mathbb{R}^{nB}$ is the output of the k^{th} iterative step or hidden layer, and $\boldsymbol{x}^{(0)} = \boldsymbol{\Phi}^\top \boldsymbol{y}$; $f_\theta(\cdot\,; \boldsymbol{y}, \boldsymbol{\Phi})$ is an iteration map $\mathbb{R}^{nB} \rightarrow \mathbb{R}^{nB}$ towards a stable equilibrium:

$$\begin{aligned} \lim_{k \rightarrow +\infty} \boldsymbol{x}^{(k)} &= \lim_{k \rightarrow +\infty} f_\theta(\boldsymbol{x}^{(k)}; \boldsymbol{y}, \boldsymbol{\Phi}) \\ &\equiv \hat{\boldsymbol{x}} = f_\theta(\hat{\boldsymbol{x}}; \boldsymbol{y}, \boldsymbol{\Phi}), \end{aligned} \tag{11}$$

where $\hat{\boldsymbol{x}} \in \mathbb{R}^{nB}$ denotes the fixed point and reconstruction result.

In this section, we first design different f_θ for SCI, in terms of the implicit infinite-depth RNN architecture and infinitely iterative PnP framework. Following (Gilton, Ongie, and Willett 2021), for gradient calculation, we optimize the network wights θ by approximating the inverse Jacobian. Convergence for specific f_θ designs is discussed.

Forward Pass

Unlike the conventional optimization method where the terminal step number is manually chosen or a network where the output is the activation from the limited layers, the result of DEQ is the equilibrium point itself. Therefore, the forward evaluation could be any procedure that solves for this equilibrium point. Considering SCI reconstruction, we design novel iterative models that converge to equilibrium.

Recurrent Neural Networks To achieve integration of DEQ and RNN for video SCI, we have:

$$\boldsymbol{x}^{(k+1)} = \mathrm{RNN}_\theta(\boldsymbol{x}^{(k)}, \boldsymbol{y}, \boldsymbol{\Phi}), \tag{12}$$

where $\mathrm{RNN}(\cdot\ )$ is a trainable RNN network learning to iteratively reconstruct effective and stable data. As shown in Fig 3, the corresponding iteration map is:

$$f_\theta(\boldsymbol{x}; \boldsymbol{y}, \boldsymbol{\Phi}) = \mathrm{RNN}_\theta(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\Phi}). \tag{13}$$

Generalized Alternating Projection Regarding the optimization iterations in the GAP method, represented in Eq. (8)-(9), we iteratively update $\boldsymbol{x}$ by:

$$\boldsymbol{x}^{(k+1)} = \mathcal{D}_\theta^{(k+1)} \left[\boldsymbol{x}^{(k)} + \boldsymbol{\Phi}^\top (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1} (\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x}^{(k)}) \right]. \tag{14}$$

Therefore, as illustrated in Fig. 4, the iteration map is:

$$f_\theta(\boldsymbol{x}; \boldsymbol{y}, \boldsymbol{\Phi}) = \mathcal{D}_\theta(\boldsymbol{x} + \boldsymbol{\Phi}^\top (\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1} (\boldsymbol{y} - \boldsymbol{\Phi}\boldsymbol{x})). \tag{15}$$

Backward Pass

While previous work often utilizes Newton's method to achieve the equilibrium and then backpropagate through all the Newton iterations, following (Gilton, Ongie, and Willett 2021), we alternatively adopt another method with high efficiency and constant memory requirement.

Loss Function To optimize network parameters θ, stochastic gradient descent is used to minimize a loss function as follows:

$$\theta^* = \arg\min_\theta \tfrac{1}{m} \textstyle\sum_{i=1}^m \ell(f_\theta(\hat{\boldsymbol{x}}_i; \boldsymbol{y}_i, \boldsymbol{\Phi}_i), \boldsymbol{x}_i^\star), \tag{16}$$

where m is the number of training samples; $\ell(\cdot, \cdot)$ is a given loss function, $\boldsymbol{x}_i^\star$ is the ground truth 3D data of the i-th training sample, $\boldsymbol{y}_i$ is the paired measurement, $\boldsymbol{\Phi}_i$ denotes the sensing matrix, and $f_\theta(\hat{\boldsymbol{x}}_i; \boldsymbol{y}_i, \boldsymbol{\Phi}_i)$ denotes the reconstruction result given as the fixed point $\hat{\boldsymbol{x}}$ of the iteration map $f_\theta(\cdot\,; \boldsymbol{y}, \boldsymbol{\Phi})$, as derived from Eq. (11). The mean-squared error (MSE) loss is used for our video SCI reconstruction:

$$\ell(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star) = \tfrac{1}{2} ||\hat{\boldsymbol{x}} - \boldsymbol{x}^\star||_2^2. \tag{17}$$

Since the reconstruction result is a fixed point of the iteration map $f_\theta(\cdot\,; \boldsymbol{y}, \boldsymbol{\Phi})$, gradient calculation of this loss term could be designed to avoid large memory demand. Following (Gilton, Ongie, and Willett 2021), we calculate the gradient of the loss term, which takes the network parameters θ into consideration.

Gradient Calculation Following (Gilton, Ongie, and Willett 2021), we calculate the loss gradient. Let ℓ be an abbreviation of $\ell(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star)$ in Eq. (17), then the loss gradient is:

$$\frac{\partial \ell}{\partial \theta} = \left(\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta} \right)^\top \frac{\partial \ell}{\partial \hat{\boldsymbol{x}}} = \left(\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta} \right)^\top (\hat{\boldsymbol{x}} - \boldsymbol{x}^\star), \tag{18}$$

where $\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta}$ is the Jacobian of $\hat{\boldsymbol{x}}$ evaluated at θ, and $\frac{\partial \ell}{\partial \hat{\boldsymbol{x}}}$ is the gradient of ℓ evaluated at $\boldsymbol{x}^\star$.

Then to compute the Jacobian $\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta}$, we recall the fixed point equation $\hat{\boldsymbol{x}} = f_\theta(\hat{\boldsymbol{x}}; \boldsymbol{y}, \boldsymbol{\Phi})$ in Eq. (11). By implicitly differentiating both sides of this fixed point equation, the Jacobian $\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta}$ is solved as:

$$\frac{\partial \hat{\boldsymbol{x}}}{\partial \theta} = \left[\boldsymbol{I} - \frac{\partial f_\theta(\boldsymbol{x}; \boldsymbol{y}, \boldsymbol{\Phi})}{\partial \boldsymbol{x}} \bigg|_{\boldsymbol{x} = \hat{\boldsymbol{x}}} \right]^{-1} \frac{\partial f_\theta(\hat{\boldsymbol{x}}; \boldsymbol{y}, \boldsymbol{\Phi})}{\partial \theta}, \tag{19}$$

which could be plugged into Eq. (18) and thus get:

$$\frac{\partial \ell}{\partial \theta} = \left[\frac{\partial f_\theta(\hat{\boldsymbol{x}}; \boldsymbol{y}, \boldsymbol{\Phi})}{\partial \theta} \right]^\top \left[\boldsymbol{I} - \frac{\partial f_\theta(\boldsymbol{x}; \boldsymbol{y}, \boldsymbol{\Phi})}{\partial \boldsymbol{x}} \bigg|_{\boldsymbol{x} = \hat{\boldsymbol{x}}} \right]^{-\top} (\hat{\boldsymbol{x}} - \boldsymbol{x}^\star), \tag{20}$$

where ${}^{-\top}$ denotes the inversion followed by transpose. As this method converted gradient calculation to the problem of calculating an inverse Jacobian-vector product, it avoids the backpropagation through many iterations of $f_\theta(\hat{\boldsymbol{x}};\boldsymbol{y},\boldsymbol{\Phi})$. To approximate the inverse Jacobian-vector product, we define the vector $\boldsymbol{a}^{(\infty)}$ as:

$$\boldsymbol{a}^{(\infty)} = \left[\boldsymbol{I} - \frac{\partial f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}}\right]^{-\top}(\hat{\boldsymbol{x}} - \boldsymbol{x}^\star). \tag{21}$$

Following (Gilton, Ongie, and Willett 2021), it is noted that $\boldsymbol{a}^{(\infty)}$ is a fixed point of the equation:

$$\boldsymbol{a}^{(k+1)} = \left[\frac{\partial f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}}\right]^{-\top}\boldsymbol{a}^{(k)} + (\hat{\boldsymbol{x}} - \boldsymbol{x}^\star), \quad \forall k = 0,1,\ldots,\infty. \tag{22}$$

Therefore, the same algorithm used to calculate the fixed point $\hat{\boldsymbol{x}}$ could also be used to calculate $\boldsymbol{a}^{(\infty)}$. The limit of fixed-point iterations for solving Eq. (22) with initial iterate $\boldsymbol{a}^{(0)} = \boldsymbol{0}$ is denoted equivalently to the Neumann series:

$$\boldsymbol{a}^{(\infty)} = \sum_{p=0}^{\infty}\left\{\left[\frac{\partial f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})}{\partial \boldsymbol{x}}\Big|_{\boldsymbol{x}=\hat{\boldsymbol{x}}}\right]^{\top}\right\}^{p}(\hat{\boldsymbol{x}} - \boldsymbol{x}^\star). \tag{23}$$

To quickly calculate the vector-Jacobian products in Eq. (22) and Eq. (23), a lot of auto-differentiation tools (*e.g.*, autograd packages in Pytorch(Paszke et al. 2019)) could be utilized. After the accurate approximation of $\boldsymbol{a}^{(\infty)}$ is calculated, the gradient in Eq. (18) is given by:

$$\frac{\partial \ell}{\partial \theta} = \left(\frac{\partial f_\theta(\hat{\boldsymbol{x}};\boldsymbol{y},\boldsymbol{\Phi})}{\partial \theta}\right)^{\top}\boldsymbol{a}^{(\infty)}. \tag{24}$$

Convergence Analysis

Given the iteration map $f_\theta(\cdot\,;\boldsymbol{y},\boldsymbol{\Phi}) : \mathbb{R}^{nB} \to \mathbb{R}^{nB}$, in this section, we discuss conditions that guarantee the convergence of the proposed deep equilibrium models $\boldsymbol{x}^{(k+1)} = f_\theta(\boldsymbol{x}^{(k)};\boldsymbol{y},\boldsymbol{\Phi})$ to a fixed-point $\hat{\boldsymbol{x}}$ as $k \to \infty$.

Assumption 1 ***(Convergence of DE-RNN).*** *For all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^{nB}$, if there exists a constant $0 \le c < 1$ satisfies that:*

$$\|\mathrm{RNN}_\theta(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\Phi}) - \mathrm{RNN}_\theta(\boldsymbol{x}',\boldsymbol{y},\boldsymbol{\Phi})\| \le c\|\boldsymbol{x} - \boldsymbol{x}'\|, \tag{25}$$

then the DE-RNN iteration map $f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})$ is contractive.

Assumption 2 ***(Convergence of DE-GAP).*** *For all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^{nB}$, if there exists a $\varepsilon > 0$ such that the denoiser $\mathcal{D}_\theta : \mathbb{R}^{nB} \to \mathbb{R}^{nB}$ satisfies:*

$$\|(\mathcal{D}_\theta - \mathbf{I})(\boldsymbol{x}) - (\mathcal{D}_\theta - \mathbf{I})(\boldsymbol{x}')\| \le \varepsilon||\boldsymbol{x} - \boldsymbol{x}'||, \tag{26}$$

where $(\mathcal{D}_\theta - \mathbf{I})(\boldsymbol{x}) := \mathcal{D}_\theta(\boldsymbol{x}) - \boldsymbol{x}$, that is, we assume the map $\mathcal{D}_\theta - \mathbf{I}$ is ε-Lipschitz, then the DE-GAP iteration map $f_\theta(\cdot;\boldsymbol{y},\boldsymbol{\Phi})$ defined in Eq. (15) *satisfies:*

$$\|f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi}) - f_\theta(\boldsymbol{x}';\boldsymbol{y},\boldsymbol{\Phi})\| \le \eta\|\boldsymbol{x} - \boldsymbol{x}'\| \tag{27}$$

for all $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^{nB}$. The coefficient η is less than 1, in which case the DE-GAP iteration map $f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})$ is contractive.

Following (Gilton, Ongie, and Willett 2021), to prove $f_\theta(\cdot;\boldsymbol{y},\boldsymbol{\Phi})$ is contractive it suffices to show $||\partial_{\boldsymbol{x}} f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})|| < 1$ for all $\boldsymbol{x} \in \mathbb{R}^{nB}$, where $||\cdot||$ denotes the spectral norm, $\partial_{\boldsymbol{x}} f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})$ is the Jacobian of $f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})$ with respect to $\boldsymbol{x} \in \mathbb{R}^{nB}$ given by:

$$\partial_{\boldsymbol{x}} f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi}) = \partial_{\boldsymbol{x}}\mathcal{D}_\theta(\boldsymbol{x})(\mathbf{I} - \boldsymbol{\Phi}^\top(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1}\boldsymbol{\Phi}), \tag{28}$$

where $\partial_{\boldsymbol{x}}\mathcal{D}_\theta \in \mathbb{R}^{nB\times nB}$ is the Jacobian of $\mathcal{D}_\theta : \mathbb{R}^{nB} \to \mathbb{R}^{nB}$ with respect to $\boldsymbol{x} \in \mathbb{R}^{nB}$.

Finally, we derive (details can be found in (Zhao 2022) or supplementary material):

$$||\partial_{\boldsymbol{x}} f_\theta(\boldsymbol{x};\boldsymbol{y},\boldsymbol{\Phi})|| \le (1+\varepsilon)\max_i|1-\lambda_i|, \tag{29}$$

where λ_i are eigenvalues of $\boldsymbol{\Phi}^\top(\boldsymbol{\Phi}\boldsymbol{\Phi}^\top)^{-1}\boldsymbol{\Phi}$; and the inequality Eq. (29) is based on the assumption that the map $(\mathcal{D}_\theta - \mathbf{I})(\boldsymbol{x}) := \mathcal{D}_\theta(\boldsymbol{x}) - \boldsymbol{x}$ is ε-Lipschitz. Therefore the spectral norm of its Jacobian $\partial_{\boldsymbol{x}}\mathcal{D}_\theta(\boldsymbol{x}) - \mathbf{I}$ is bounded by η, which demonstrates f_θ is η-Lipschitz with $\eta = (1+\varepsilon)\max_i|1-\lambda_i|$.

It is worth noting that convergence is not yet guaranteed in our calculation above since η is larger than 1. In SCI cases, it is challenging to provide a theoretical guarantee. However, we observe our models converge well in the experiments.

Experiment

Experiment Setting

Architecture Specifics For our learned network, we have experimented with various network architectures. Specifically, for the DE-RNN model, we adopt the architecture from BIRNAT (Cheng et al. 2020). Regarding its two-stage (forward+backward) RNN as a whole, we iteratively feed the output of the backward RNN back as the input of the forward one. For the DE-GAP model, we employ different neural networks as denoisers $\mathcal{D}_\theta$ and utilize the real spectral norm (Yoshida and Miyato 2017) for convergence purposes. We found that some architectures can yield fairly good performance while combining our proposed DEQ for SCI. In a summary, these feasible network architectures are Unet (Ronneberger, Fischer, and Brox 2015) with real spectral norm (denoted as RSN-Unet), Unet with 3D convolutional kernels (denoted as Unet-3D), simple CNN networks without and with real spectral norm (denoted as CNN and RSN-CNN, respectively), and FFDnet (Zhang, Zuo, and Zhang 2018).

Training Details Following BIRNAT (Cheng et al. 2020), we choose the dataset DAVIS2017 (Pont-Tuset et al. 2017) for training. DAVIS2017 has 90 scenes and in total 6208 frames. We crop its video frames to video patch cubes with the spatial size of $256 \times 256 \times 8$, and obtain $26,000$ training samples with data augmentation. Then we train the neural network for 30 epochs. The initial learning rate is 1×10^{-3} and learning rate decayed is 10% every 10 epochs. During training, we utilize Anderson acceleration for both the forward and backward pass fixed-point iterations.

Experiment Results

Comparisons on Datasets For evaluations, we test our proposed DE-RNN and DE-GAP on six classical simulation datasets including `Kobe`, `Runner`, `Drop`, `Traffic`,

Methods	Kobe	Traffic	Runner	Drop	Vehicle	Aerial	**Average**
GAP-net-AE-S9	24.20, 0.570	21.13, 0.685	29.18, 0.886	32.21, 0.907	24.19, 0.769	24.41, 0.744	25.89, 0.760
GAP-TV	26.46, 0.885	20.89, 0.715	28.52, 0.909	34.63, 0.970	24.82 0.838	25.05, 0.828	26.73, 0.858
E2E-CNN	29.02, 0.861	23.45, 0.838	34.43, 0.958	36.77, 0.974	26.40, 0.886	27.52, 0.882	29.26, 0.900
PnP-FFDnet	**30.50**, 0.926	24.18, 0.828	32.15, 0.933	**40.70**, 0.989	25.42, 0.849	25.27, 0.829	29.70, 0.892
DE-RNN	21.46, 0.697	19.47, 0.715	27.85, 0.818	30.16, 0.909	23.65, 0.832	24.83, 0.855	24.53, 0.804
DE-GAP-Unet-3D	26.76, 0.866	21.42, 0.786	30.45, 0.894	33.82, 0.963	24.94, 0.885	24.83, 0.847	27.07, 0.878
DE-GAP-RSN-CNN	27.33, 0.887	22.58, 0.829	30.74, 0.903	35.95.0.977	25.33, 0.899	25.57, 0.881	27.92, 0.896
DE-GAP-RSN-Unet	28.92, 0.939	23.68, 0.869	32.37, 0.951	36.54, 0.972	25.50, 0.905	25.67, 0.884	28.80, 0.913
DE-GAP-CNN	28.79, 0.935	23.55, 0.864	32.35, 0.950	38.14, 0.983	25.45, 0.903	25.84, 0.890	29.02, 0.921
DE-GAP-FFDnet	29.32, **0.952**	**24.71**, **0.907**	**33.06**, **0.971**	39.89, **0.992**	**25.85**, **0.905**	**26.02**, **0.892**	**29.81**, **0.936**

Table 1: The results in terms of PSNR (dB) and SSIM by different algorithms on six datasets for video SCI reconstruction.

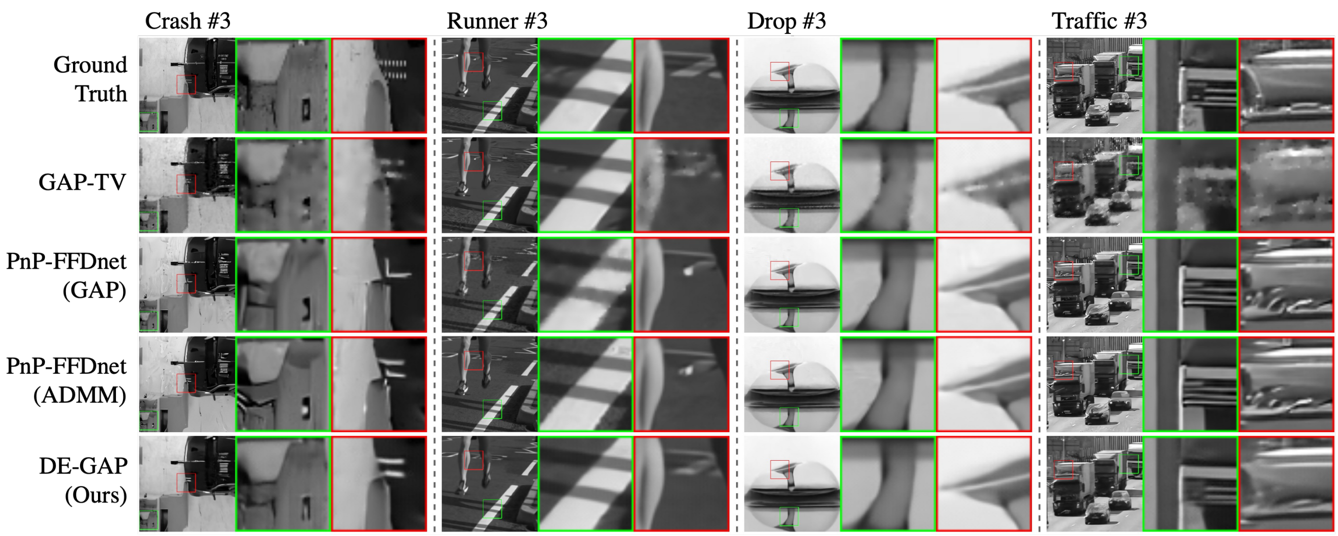

Figure 5: Comparison of selected reconstruction results with the spatial size of $256 \times 256 \times 8$.

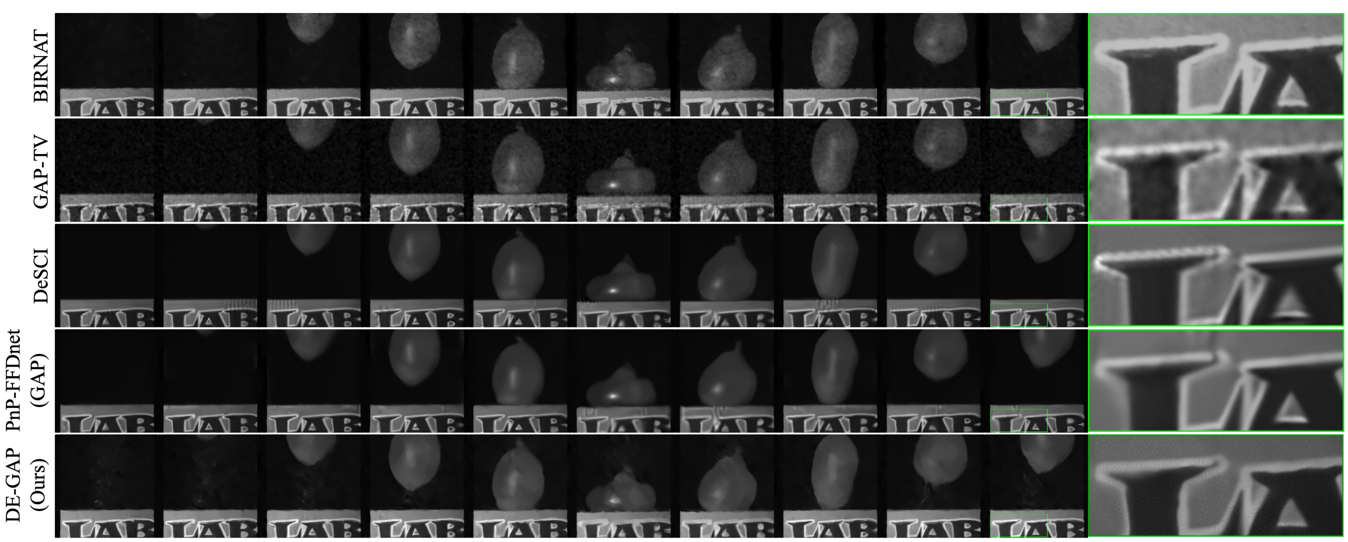

Figure 6: Comparison of selected reconstruction results of real data Water Balloon with the spatial size of $512 \times 512 \times 10$. Reconstruction of the real data is more difficult than simulations due to the inevitable measurement noise. As shown in this figure, GAP-TV, DeSCI, and PnP-FFDnet (GAP) have more artifacts and distortions around margins. Our model can maintain a clear and accurate image structure, thus leading to higher performance.

`Vehicle`, and `Aerial` (Yuan et al. 2020) with the spatial size of 256×256 and compression ratio B=8. Quantitative comparison results with other video SCI reconstruction algorithms including GAP-net (Meng, Jalali, and Yuan 2020), GAP-TV (Yuan 2016), E2E-CNN (Qiao et al. 2020) and PnP-FFDnet (Yuan et al. 2020) on Peak Signal to Noise Ratio (PSNR) and structured similarity (SSIM) (Wang et al. 2004) are provided in Table 1. What stands out in the table is that our method achieves around 0.1 dB improvement in PSNR and 0.4 in SSIM. The improvement of SSIM indicates our method could reconstruct images with relative fine structure, which is confirmed by qualitative evaluations in Fig. 5.

To sum up, both the quantitative and qualitative comparisons demonstrate that our method could achieve competitive performance in contrast to other algorithms. We do notice that there are some recent work using complicated deep networks to obtain better results than ours (Cheng et al. 2021; Wu, Zhang, and Mou 2021; Wang et al. 2021; Meng, Jalali, and Yuan 2020; Zheng, Yang, and Yuan 2022). However, these handcraft designs of different network structures are not necessarily converging to a stable point. By contrast, our paper aims to provide a stable solution for SCI.

Recalling Fig. 2, where we have run existing methods and our algorithm for iterations, while RNN and PnP fail in stable recovery, our method could converge to a fixed point and maintain at high-level results. Reconstructed frames in Fig. 1 further verified this virtue of our proposed algorithm.

Real-world Data Reconstruction We also evaluate the DE-GAP model on real-world dataset **Water Balloon** (Qiao et al. 2020) and **Chopper Wheel** (Llull et al. 2013) captured by real video SCI cameras. Note that this is more challenging due to the unavoidable noise inside the real measurements, which demands the high robustness of the algorithm.

We compare the results with other algorithms including GAP-TV (Yuan 2016), DeSCI (Liu et al. 2018) and PnP-FFDnet (Yuan et al. 2020), as shown in Figs. 6 and 7. The reconstruction results on real-world data demonstrate the effectiveness and generalization of our proposed method. Note that the reconstruction results of real data are achieved by the model trained to utilize the simulation mask, which means that our proposed model is kind of flexible and can achieve stable results by the virtue of the fact that our model can be theoretically infinitely extended.

Specifically, we observe that: i) GAP-TV and DeSCI often generate a lot of artifacts and show noisy texture. ii) PnP-FFDnet has artifacts and distortions around margins. iii) In contrast to them, our method shows high-quality results with clear content and structure.

Processing Time Though we equivalently realize infinite optimization iterations with deep neural networks plugged in to perform video SCI reconstruction, our designed methods elegantly avoid long inference time. As Table 2 shows, our method needs a relatively short processing time in comparison to other algorithms including GAP-TV (Yuan 2016), DeSCI (Liu et al. 2018), PnP-FFDnet (Yuan et al. 2020) and RevSCI (Wang et al. 2021).

GAP-TV	DeSCI	PnP-FFDnet	RevSCI	DE-RNN	DE-GAP
4.2	6180	3.0	0.19	4.68	1.90

Table 2: Average running time per measurement in seconds by different algorithms on classical six datasets.

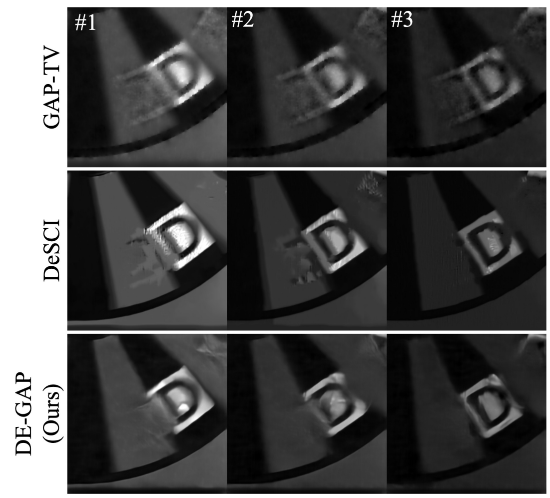

Figure 7: Comparison of reconstruction results of real data Chopper Wheel with the spatial size of $256 \times 256 \times 3$.

Future Work

Since DEQ under exact gradients may suffer from training time and stability issues, we will incorporate inexact gradients (Geng et al. 2021) and fixed point correction (Bai et al. 2022) to solve these issues and improve the performance. Our preliminary experiments found that inexact gradients could accelerate the backward passes in training our models by roughly $1.3 \sim 1.5\times$. Another direction is to integrate DEQ with semantic analysis in SCI (Zhang et al. 2022).

Conclusion

In this paper, to solve the problems of memory requirement and unstable recovery in existing methods, we propose deep equilibrium models for video SCI. Fusing data-driven regularization and stable convergence in a theoretically sound manner, we combine DEQ with existing methods and design two novel models, *i.e.*, DE-RNN and DE-GAP. Each equilibrium model implicitly learns a nonexpansive operator by training the embedded neural network and analytically computes the fixed point, thus enabling unlimited iterative steps and infinite network depth with only a constant memory requirement in the training and inference process. Furthermore, we analyze the convergence conditions for each equilibrium model to ensure the results of our models converge to equilibrium. We evaluate our proposed models using different neural networks as the implicit operator on a variety of simulations and real datasets. In comprehensive comparisons with existing algorithms, both quantitative and qualitative evaluations of our results demonstrate the effectiveness and stability of our proposed method.

Acknowledgements

This work was supported by the National Natural Science Foundation of China [62271414], Zhejiang Provincial Natural Science Foundation of China [LR23F010001] and Westlake Foundation [2021B1 501-2]. The authors would like to thank Research Center for Industries of the Future (RCIF) at Westlake University for supporting this work and the funding from Lochn Optics. The support and funding from Research Postgraduate Student Innovation Award (The University of Hong Kong) is also appreciated.

References

Bai, S.; Geng, Z.; Savani, Y.; and Kolter, J. Z. 2022. Deep Equilibrium Optical Flow Estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 620–630.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. Trellis networks for sequence modeling. *arXiv preprint arXiv:1810.06682*.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2019. Deep equilibrium models. *arXiv preprint arXiv:1909.01377*.

Bai, S.; Koltun, V.; and Kolter, J. Z. 2020. Multiscale Deep Equilibrium Models. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 5238–5250. Curran Associates, Inc.

Boyd, S.; Parikh, N.; Chu, E.; Peleato, B.; and Eckstein, J. 2011. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1): 1–122.

Chan, S. H.; Wang, X.; and Elgendy, O. A. 2017. Plug-and-Play ADMM for Image Restoration: Fixed-Point Convergence and Applications. *IEEE Transactions on Computational Imaging*, 3: 84–98.

Cheng, Z.; Chen, B.; Liu, G.; Zhang, H.; Lu, R.; Wang, Z.; and Yuan, X. 2021. Memory-efficient network for large-scale video compressive sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16246–16255.

Cheng, Z.; Chen, B.; Lu, R.; Wang, Z.; Zhang, H.; Meng, Z.; and Yuan, X. 2022. Recurrent neural networks for snapshot compressive imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Cheng, Z.; Lu, R.; Wang, Z.; Zhang, H.; Chen, B.; Meng, Z.; and Yuan, X. 2020. BIRNAT: Bidirectional recurrent neural networks with adversarial training for video snapshot compressive imaging. In *European Conference on Computer Vision*, 258–275. Springer.

Dabre, R.; and Fujita, A. 2019. Recurrent stacking of layers for compact neural machine translation models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 6292–6299.

Dehghani, M.; Gouws, S.; Vinyals, O.; Uszkoreit, J.; and Kaiser, Ł. 2018. Universal transformers. *arXiv preprint arXiv:1807.03819*.

Geng, Z.; Zhang, X.-Y.; Bai, S.; Wang, Y.; and Lin, Z. 2021. On training implicit models. *Advances in Neural Information Processing Systems*, 34: 24247–24260.

Gilton, D.; Ongie, G.; and Willett, R. 2021. Deep equilibrium architectures for inverse problems in imaging. *arXiv preprint arXiv:2102.07944*.

Gomez, A. N.; Ren, M.; Urtasun, R.; and Grosse, R. B. 2017. The reversible residual network: Backpropagation without storing activations. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2211–2221.

Jalali, S.; and Yuan, X. 2019. Snapshot compressed sensing: Performance bounds and algorithms. *IEEE Transactions on Information Theory*, 65(12): 8005–8024.

Liao, X.; Li, H.; and Carin, L. 2014. Generalized Alternating Projection for Weighted-$\ell_{2,1}$ Minimization with Applications to Model-based Compressive Sensing. *SIAM Journal on Imaging Sciences*, 7(2): 797–823.

Liu, Y.; Yuan, X.; Suo, J.; Brady, D. J.; and Dai, Q. 2018. Rank minimization for snapshot compressive imaging. *IEEE transactions on pattern analysis and machine intelligence*, 41(12): 2990–3006.

Llull, P.; Liao, X.; Yuan, X.; Yang, J.; Kittle, D.; Carin, L.; Sapiro, G.; and Brady, D. J. 2013. Coded aperture compressive temporal imaging. *Optics express*, 21(9): 10526–10545.

Ma, J.; Liu, X.-Y.; Shou, Z.; and Yuan, X. 2019. Deep tensor admm-net for snapshot compressive imaging. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10223–10232.

Meng, Z.; Jalali, S.; and Yuan, X. 2020. Gap-net for snapshot compressive imaging. *arXiv preprint arXiv:2012.08364*.

Meng, Z.; and Yuan, X. 2021. Perception inspired deep neural networks for spectral snapshot compressive imaging. In *2021 IEEE International Conference on Image Processing (ICIP)*, 2813–2817. IEEE.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32: 8026–8037.

Pont-Tuset, J.; Perazzi, F.; Caelles, S.; Arbeláez, P.; Sorkine-Hornung, A.; and Van Gool, L. 2017. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*.

Qiao, M.; Meng, Z.; Ma, J.; and Yuan, X. 2020. Deep learning for video compressive sensing. *APL Photonics*, 5(3): 030801.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer.

Ryu, E.; Liu, J.; Wang, S.; Chen, X.; Wang, Z.; and Yin, W. 2019. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, 5546–5557. PMLR.

Sreehari, S.; Venkatakrishnan, S. V.; Wohlberg, B.; Buzzard, G. T.; Drummy, L. F.; Simmons, J. P.; and Bouman, C. A.

2016. Plug-and-Play Priors for Bright Field Electron Tomography and Sparse Interpolation. *IEEE Transactions on Computational Imaging*, 2(4): 408–423.

Venkatakrishnan, S. V.; Bouman, C. A.; and Wohlberg, B. 2013. Plug-and-Play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, 945–948.

Wang, L.; Cao, M.; Zhong, Y.; and Yuan, X. 2022. Spatial-Temporal Transformer for Video Snapshot Compressive Imaging. *arXiv preprint arXiv:2209.01578*.

Wang, Z.; Bovik, A. C.; Sheikh, H. R.; and Simoncelli, E. P. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4): 600–612.

Wang, Z.; Zhang, H.; Cheng, Z.; Chen, B.; and Yuan, X. 2021. Metasci: Scalable and adaptive reconstruction for video compressive sensing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2083–2092.

Wei, K.; Aviles-Rivero, A.; Liang, J.; Fu, Y.; Schönlieb, C.-B.; and Huang, H. 2020. Tuning-free Plug-and-Play Proximal Algorithm for Inverse Imaging Problems. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 10158–10169. PMLR.

Wu, Z.; Yang, C.; Su, X.; and Yuan, X. 2022. Adaptive Deep PnP Algorithm for Video Snapshot Compressive Imaging. *arXiv preprint arXiv:2201.05483*.

Wu, Z.; Zhang, J.; and Mou, C. 2021. Dense Deep Unfolding Network with 3D-CNN Prior for Snapshot Compressive Imaging. *arXiv preprint arXiv:2109.06548*.

Yang, C.; Zhang, S.; and Yuan, X. 2022. Ensemble Learning Priors Driven Deep Unfolding for Scalable Video Snapshot Compressive Imaging. In *European Conference on Computer Vision*, 600–618. Springer.

Yang, J.; Liao, X.; Yuan, X.; Llull, P.; Brady, D. J.; Sapiro, G.; and Carin, L. 2015. Compressive Sensing by Learning a Gaussian Mixture Model from Measurements. *IEEE Transaction on Image Processing*, 24(1): 106–119.

Yang, J.; Yuan, X.; Liao, X.; Llull, P.; Sapiro, G.; Brady, D. J.; and Carin, L. 2014. Video Compressive Sensing Using Gaussian Mixture Models. *IEEE Transaction on Image Processing*, 23(11): 4863–4878.

Yang, Q.; and Zhao, Y. 2022. Revisit dictionary learning for video compressive sensing under the Plug-and-Play framework. In *Seventh Asia Pacific Conference on Optics Manufacture and 2021 International Forum of Young Scientists on Advanced Optical Manufacturing (APCOM and YSAOM 2021)*, volume 12166, 2018–2025. SPIE.

Yoshida, Y.; and Miyato, T. 2017. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*.

Yuan, X. 2016. Generalized alternating projection based total variation minimization for compressive sensing. In *2016 IEEE International Conference on Image Processing (ICIP)*, 2539–2543. IEEE.

Yuan, X.; Brady, D. J.; and Katsaggelos, A. K. 2021. Snapshot Compressive Imaging: Theory, Algorithms, and Applications. *IEEE Signal Processing Magazine*, 38(2): 65–88.

Yuan, X.; Liu, Y.; Suo, J.; and Dai, Q. 2020. Plug-and-Play Algorithms for Large-scale Snapshot Compressive Imaging. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yuan, X.; Liu, Y.; Suo, J.; Durand, F.; and Dai, Q. 2021. Plug-and-Play Algorithms for Video Snapshot Compressive Imaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–1.

Zhang, K.; Zuo, W.; and Zhang, L. 2018. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9): 4608–4622.

Zhang, Z.; Zhang, B.; Yuan, X.; Zheng, S.; Su, X.; Suo, J.; Brady, D. J.; and Dai, Q. 2022. From compressive sampling to compressive tasking: retrieving semantics in compressed domain with low bandwidth. *PhotoniX*, 3(1): 1–22.

Zhao, Y. 2022. Mathematical Cookbook for Snapshot Compressive Imaging. *arXiv preprint arXiv:2202.07437*.

Zheng, S.; Wang, C.; Yuan, X.; and Xin, H. L. 2021. Super-compression of large electron microscopy time series by deep compressive sensing learning. *Patterns*, 2(7): 100292.

Zheng, S.; Yang, X.; and Yuan, X. 2022. Two-Stage is Enough: A Concise Deep Unfolding Reconstruction Network for Flexible Video Compressive Sensing. *arXiv preprint arXiv:2201.05810*.