DMPlug: A Plug-in Method for Solving Inverse Problems with Diffusion Models

Hengkang Wang[1] **Xu Zhang**[2*] **Taihui Li**[1] **Yuxiang Wan**[1] **Tiancong Chen**[1] **Ju Sun**[1]
[1]Department of Computer Science and Engineering, University of Minnesota
{wang9881,lixx5027,wan01530,chen6271,jusun}@umn.edu
[2]Amazon.com Inc., spongezhang@gmail.com

Abstract

Pretrained diffusion models (DMs) have recently been popularly used in solving inverse problems (IPs). The existing methods mostly interleave iterative steps in the reverse diffusion process and iterative steps to bring the iterates closer to satisfying the measurement constraint. However, such interleaving methods struggle to produce final results that look like natural objects of interest (i.e., manifold feasibility) and fit the measurement (i.e., measurement feasibility), especially for nonlinear IPs. Moreover, their capabilities to deal with noisy IPs with unknown types and levels of measurement noise are unknown. In this paper, we advocate viewing the reverse process in DMs as a function and propose a novel plug-in method for solving IPs using pretrained DMs, dubbed DMPlug. DMPlug addresses the issues of manifold feasibility and measurement feasibility in a principled manner, and also shows great potential for being robust to unknown types and levels of noise. Through extensive experiments across various IP tasks, including two linear and three nonlinear IPs, we demonstrate that DMPlug consistently outperforms state-of-the-art methods, often by large margins especially for nonlinear IPs. The code is available at https://github.com/sun-umn/DMPlug.

1 Introduction

Inverse problems (IPs) are prevalent in numerous fields, such as computer vision, medical imaging, remote sensing, and autonomous driving [1–4]. The goal of IPs is to recover an unknown object $\boldsymbol{x}$ from noisy measurements $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{n}$, where $\mathcal{A}$ is a (possibly nonlinear) forward model and $\boldsymbol{n}$ denotes the measurement noise. IPs are often ill-posed: typically, even if $\boldsymbol{y}$ is noiseless, $\boldsymbol{x}$ cannot be uniquely determined from $\boldsymbol{y}$ and $\mathcal{A}$. Hence, incorporating prior knowledge on $\boldsymbol{x}$ is necessary to obtain a reliable estimate of the underlying $\boldsymbol{x}$.

Traditionally, IPs are solved in the regularized data-fitting framework, often motivated as performing the Maximum a Posterior (MAP) inference:

$$\min_{\boldsymbol{x}} \ \ell(\boldsymbol{y}, \mathcal{A}(\boldsymbol{x})) + \Omega(\boldsymbol{x}) \qquad \ell(\boldsymbol{y}, \mathcal{A}(\boldsymbol{x})) : \text{data-fitting loss}, \ \Omega(\boldsymbol{x}) : \text{regularizer}. \tag{1}$$

Here, minimizing the data-fitting loss promotes $\boldsymbol{y} \approx \mathcal{A}(\boldsymbol{x})$, and the regularizer encodes prior knowledge on $\boldsymbol{x}$. Recently, the advent of deep learning (DL) has brought about a few new algorithmic ideas to solve IPs. For example, given a training set of measurement-object pairs, i.e., $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}_{i=1,\dots,N}$, one can hope to train a DL model that directly predicts $\boldsymbol{x}$ for a given $\boldsymbol{y}$ [5–16]. However, such hopes can be shattered by practical challenges in collecting **massive and realistic** paired training sets, especially for complex IPs [17, 18]. Even if such challenges can be tackled, one may need to collect a new training set and train a new DL model for every new IP [15], overlooking potential shared priors on $\boldsymbol{x}$ across IPs. An attractive alternative family of ideas combine pretrained priors on $\boldsymbol{x}$ and

*This work is not related to Dr. Xu Zhang's position at Amazon.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-3744

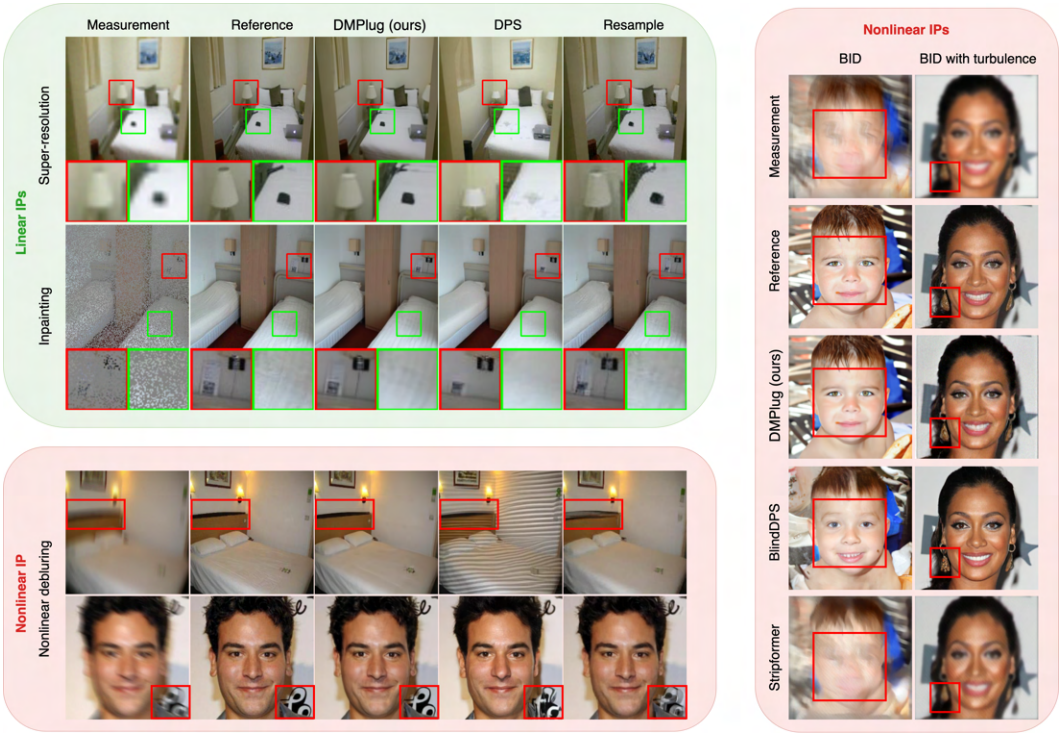

Figure 1: Visualization of sample results from our DMPlug method (**Ours**) and main competing methods (**DPS** [19] and **Resample** [20] for super-resolution, inpainting, and nonlinear deblurring; **BlindDPS** [21] and **Stripformer** [6] for blind image deblurring (BID) and BID with turbulence) on IPs we focus on in this paper. All measurements contain Gaussian noise with $\sigma = 0.01$.

regularized data-fitting in Eq. (1). For example, they first model the distribution of $\boldsymbol{x}$ using deep generative models, such as generative adversarial networks (GANs) and diffusion models (DMs), based on training sets of the form $\{\boldsymbol{x}_i\}_{i=1,\ldots,N}$, and then encode these pretrained generative priors when solving Eq. (1). In this way, pretrained priors on $\boldsymbol{x}$ can be reused in an off-the-shelf manner in different IPs about the same family of structured objects.

In this paper, we focus on solving IPs with pretrained DMs. DMs have recently emerged as a dominant family of deep generative models due to their relative stability during training (e.g., vs. GANs) and their strong capabilities to generate photorealistic images (and, depending on applications, other structured objects) once trained [22–27]. These strengths of DMs have motivated ideas to use pretrained DMs to solve IPs, such as denoising, super-resolution, inpainting, deblurring, and phase retrieval [28, 29, 20, 30–33, 21, 26, 27, 34]. Most of these ideas interleave iterative reverse diffusion steps and iterative steps (sometimes projection steps, especially for linear IPs) to move closer to the feasibility set $\{\boldsymbol{x}|\boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$ (see Fig. 3 and Fig. 4 (1)). However, they typically cannot guarantee the final convergence of the iteration sequence to either the feasible set (i.e., **measurement feasibility**), or the object manifold $\mathcal{M}$ captured by pretrained DMs (i.e., **manifold feasibility**), as they have modified both iterative processes (see detailed arguments in Section 2).

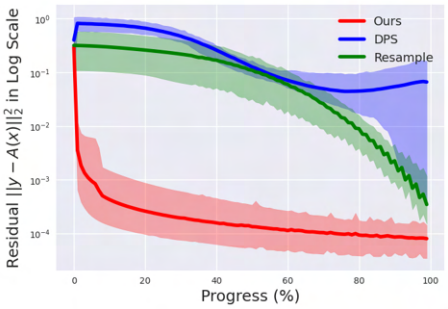

Figure 2: **Evolution of the data-fitting loss $\|\boldsymbol{y} - \mathcal{A}(\boldsymbol{x})\|_2^2$ of our DMPlug method vs. SOTA methods over percentage progress**, for noiseless nonlinear deblurring on the CelebA dataset. Here, the percentage progress is calculated with respect to the total number of iterations taken by each method. The shadow regions indicate the ranges of the loss over 50 instances.

Fig. 1 shows visible artifacts produced by these ideas on several IPs, highlighting **(Issue 1) insufficient manifold feasibility**. To quickly confirm **(Issue 2) insufficient measurement feasibility**,

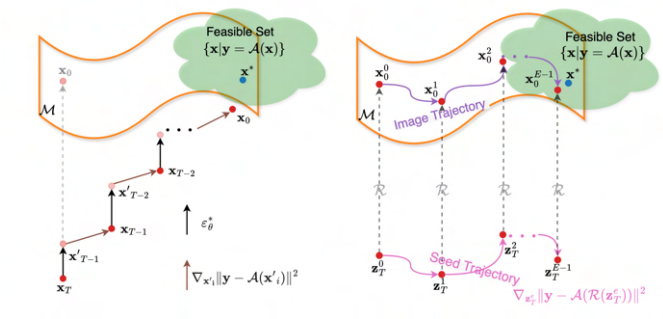

Figure 3: Interleaving methods (left) vs. our DMPlug method (right) for solving IPs using pretrained DMs. While interleaving methods cannot ensure the feasibility of the final estimate for either the object manifold $\mathcal{M}$ or the feasible set $\{\boldsymbol{x}|\boldsymbol{y}=\mathcal{A}(\boldsymbol{x})\}$, our DMPlug method ensures the manifold feasibility while promoting $\boldsymbol{y}\approx\mathcal{A}(\boldsymbol{x})$ through global optimization.

we experiment on **noiseless** instances of the nonlinear deblurring problem (a nonlinear IP) following [20, 19], and find that state-of-the-art (SOTA) methods fail to find an $\boldsymbol{x}$ that satisfies $\boldsymbol{y}=\mathcal{A}(\boldsymbol{x})$, as shown in Fig. 2. Furthermore, most SOTA DM-based methods for IPs assume known noise types (e.g., often Gaussian) and known, often very low, noise levels, casting doubts on their performance when faced with unknown noise types and levels, i.e., **(Issue 3) robustness to unknown noise types and levels**, as we confirm in Table 4.

Our contributions In this paper, **we propose a novel plug-in method, dubbed DMPlug, to solve IPs with pretrained DMs to mitigate all of the above three issues**. DMPlug departs from the popular and dominant interleaving line of ideas by viewing the whole reverse diffusion process as a function $\mathcal{R}(\cdot)$, consisting of multiblock stacked DL models, mapping from the seed space to the object manifold $\mathcal{M}$. This novel perspective allows us to naturally parameterize the object to be recovered as $\boldsymbol{x}=\mathcal{R}(\boldsymbol{z})$ and then plug this reparametrization into Eq. (1), leading to a unified optimization formulation **with respect to the seed** $\boldsymbol{z}$. Conceptually, since $\mathcal{R}(\boldsymbol{z})$ probably produces a feasible point on the object manifold $\mathcal{M}$ and global optimization of the unified formulation encourages $\boldsymbol{y}\approx\mathcal{A}(\mathcal{R}(\boldsymbol{z}))$, the optimized $\boldsymbol{z}$ could lead to an $\boldsymbol{x}=\mathcal{R}(\boldsymbol{z})$ that **enjoys both manifold and measurement feasibility**, i.e., **(tackling Issues 1 & 2)**. Fig. 3 and Fig. 4 schematically illustrate the dramatic difference between interleaving methods and our plug-in method (DMPlug), and Figs. 1 and 2 confirms DMPlug's **strong capability** in finding feasible solutions. For noisy IPs with unknown noise types and levels, we observe that our DmPlug enjoys a benign "early-learning-then-overfitting" (ELTO) property: the quality of the estimated object climbs first to a peak and then degrades once the noise is picked up, as shown in Fig. 6 (2). This benign property, in combination with appropriate early-stopping (ES) methods that stop the estimate sequence near the peak, allows our method to solve IPs without the exact noise information, i.e., **(tackling Issue 3)**, as shown in Table 4). Fortunately, we find that an ES method, ES-WMV [35] (co-developed by a subset of the current authors), works well for this purpose (see Table 4).

Our contributions can be summarized as follows. **(1)** In Section 3.1, we **pioneer a novel plug-in method, DMPlug**, which is significantly different from the prevailing interleaving methods, to solve IPs with pretrained DMs. Then we **make the proposed method practical** in terms of computational and memory expenses by leveraging one key observation; **(2)** In Section 4 and Appendix E, we perform extensive experiments on various linear and nonlinear IPs and show that **our method outperforms SOTA methods**, both qualitatively and quantitatively—often by large margins, especially on nonlinear IPs. For example, measured in PSNR, our method can lead SOTA methods by about 2dB and $3\sim6$dB for linear and nonlinear IPs, respectively. Moreover, our method demonstrates flexibility in employing different priors and optimizers, as explored in Section 4.3; **(3)** In Section 3.3, we **observe an early-learning-then-overfitting (ELTO) property**, i.e., that our DMPlug tends to recover the desirable object first and then overfit to the potential noise. By taking advantage of this benign property and integrating the ES method ES-WMV [35], our method is **the first to achieve robustness to unknown noise types and levels**, leading SOTA methods by about 1dB and 3.5dB in terms of PSNR for linear and nonlinear IPs, respectively.

2 Background and related work

Diffusion models (DMs) The denoising diffusion probabilistic model (**DDPM**) [23] is a seminal DM for unconditional image generation. It gradually transforms $\boldsymbol{x}_0\sim p_{\text{data}}$ into total noise $\boldsymbol{x}_T\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I})$ (i.e., forward diffusion process) and then learns to gradually recover $\boldsymbol{x}_0$ from $\boldsymbol{x}_T$ through

https://doi.org/10.52202/079017-3744

incremental denoising (i.e., reverse diffusion process). The forward process can be described by a stochastic differential equation (SDE), $d\boldsymbol{x} = -\beta_t/2 \cdot \boldsymbol{x}dt + \sqrt{\beta_t}d\boldsymbol{w}$, where β_t is the noise schedule and $\boldsymbol{w}$ is the standard Wiener process. The corresponding reverse process is described by

$$\text{(Reverse SDE for DDPM) } d\boldsymbol{x} = -\beta_t \left[\boldsymbol{x}/2 + \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] dt + \sqrt{\beta_t}d\overline{\boldsymbol{w}}. \tag{2}$$

Here, $\overline{\boldsymbol{w}}$ is the time-reversed standard Wiener process, $p_t(\boldsymbol{x})$ is the probability density at time t, and $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ is the (Stein) score function, which is approximated by a DL model $\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x})$ via score matching methods during DM training [36, 37]. For discrete settings, given time steps $t \in \{1, \ldots, T\}$, a variance schedule $\beta_1, \ldots, \beta_T$, $\alpha_t \doteq 1 - \beta_t$ with $\alpha_T \to 0$, and $\bar{\alpha}_t \doteq \prod_{s=1}^{t} \alpha_s$, the DDPM has the forward process $\boldsymbol{x}_t = \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{z}$ and the reverse process

$$\text{(DDPM) } \boldsymbol{x}_{t-1} = 1/\sqrt{\alpha_t} \cdot \left(\boldsymbol{x}_t - \beta_t/\sqrt{1-\bar{\alpha}_t} \cdot \boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x}_t)\right) + \sqrt{\beta_t}\boldsymbol{z}, \tag{3}$$

where $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. As the DDPM has a slow reverse/sampling process, [24] proposes the denoising diffusion implicit model (**DDIM**) to mitigate this issue. With the same notation as that of the DDPM, the DDIM makes a crucial change to the DDPM: relaxing the forward process to be non-Markovian by making $\boldsymbol{x}_t$ depend on both $\boldsymbol{x}_0$ and $\boldsymbol{x}_{t-1}$. This simple change allows skipping iterative steps in the reverse process, without retraining the DDPM. This leads to much smaller numbers of reverse steps, and hence substantial speedup in sampling. The reverse process is now defined as

$$\text{(DDIM) } \boldsymbol{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\widehat{\boldsymbol{x}}_0(\boldsymbol{x}_t) + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x}_t), \tag{4}$$

where $\widehat{\boldsymbol{x}}_0(\boldsymbol{x}_t) \doteq [\boldsymbol{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x}_t)]/\sqrt{\bar{\alpha}_t}$ is the predicted $\boldsymbol{x}_0$ with $\boldsymbol{x}_t$.

Pretrained DMs for solving IPs Ideas for solving IPs with DMs can be classified into two categories: supervised and zero-shot [31, 38]. The former trains DM-based IP solvers based on paired training sets of the form $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}$ and is hence not our focus here (see our arguments in Section 1). The latter makes use of pretrained DMs as data-driven priors: **(I)** Most of the work in this category considers modeling $p_t(\boldsymbol{x}|\boldsymbol{y})$ directly and replaces the (unconditional) score function $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ in Eq. (2) by the conditional score function $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}|\boldsymbol{y}) = \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{y}|\boldsymbol{x})$, leading to the conditional reverse SDE

$$d\boldsymbol{x} = [-\beta_t/2 \cdot \boldsymbol{x} - \beta_t(\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{y}|\boldsymbol{x}))] \, dt + \sqrt{\beta_t}d\overline{\boldsymbol{w}}. \tag{5}$$

Here, while $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ can be naturally approximated by the pretrained score function $\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(t)}(\boldsymbol{x})$, $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{y}|\boldsymbol{x})$ is intractable as $\boldsymbol{y}$ does not directly depend on $\boldsymbol{x}(t)$[2]. Ideas to circumvent this difficulty include approximating $p_t(\boldsymbol{y}|\boldsymbol{x}(t))$ by $p_t(\boldsymbol{y}|\widehat{\boldsymbol{x}}(0)[\boldsymbol{x}(t)])$, where $\widehat{\boldsymbol{x}}(0)[\boldsymbol{x}(t)]$ is implemented as $\widehat{\boldsymbol{x}}_0(\boldsymbol{x}_t)$ of Eq. (4) in discretization [19, 21], and interleaving unconditional reverse steps of Eq. (3) or Eq. (4) and (approximate) projections onto the feasible set $\{\boldsymbol{x}|\boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$ to bypass the likelihood $p_t(\boldsymbol{y}|\boldsymbol{x})$ [39, 30, 33, 32, 40, 41, 20]; **(II)** An interesting alternative is to recall that the MAP framework (see also Eq. (1)) involves $\max_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x}) + \log p(\boldsymbol{x})$, and first-order methods, especially proximal-gradient style methods, to optimize the MAP formulation typically only need to access $p(\boldsymbol{x})$ through $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$, i.e., the score function—the central object in DMs! So, one can derive IP solvers by wrapping pretrained DMs around first-order methods for (approximately) optimizing the MAP formulation, see, e.g., [27, 28].

Despite the disparate conceptual ways of utilizing pretrained DM priors, most of the methods under (I) and (II) proposed in the literature so far follow a single algorithmic template, i.e., Algorithm 1. There, Lines 3-5 are simply a reverse iterative step in the DDIM (Eq. (4); obviously, one could replace this by a reverse iterative step in other appropriate pretrained DMs. Line 6 helps move the iterate closer or onto the feasible set $\{\boldsymbol{x}|\boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$. In other words, these methods interleave iterative steps to move toward the data manifold $\mathcal{M}$ defined by the pretrained DM and iterative steps to move toward the feasible set $\{\boldsymbol{x}|\boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$, i.e., as illustrated in Fig. 3 (left). However, it is unclear, a priori, whether such interleaving iterative sequence will converge to either, leading to concerns about **(Issue 1) insufficient manifold feasibility** and **(Issue 2) insufficient measurement feasibility**. In fact, our Figs. 1 and 2 confirm these issues empirically, echoing observations made in several prior papers [20, 42, 30]. In principle, Issue 2 can be mitigated by ensuring that Line 6 finds a feasible

[2]Recall that in the continuous formulation of diffusion processes, $\boldsymbol{x}$ is a function of t, i.e., $\boldsymbol{x}(t)$ where the continuous variable $t \in [0, T]$ is often omitted.

Algorithm 1 Template for interleaving methods

Input: # Diffusion steps T, measurement $\boldsymbol{y}$

$\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
for $i = T-1$ to 0 **do**
  $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(i)}(\boldsymbol{x}_i)$
  $\hat{\boldsymbol{x}}_0 \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}\big(\boldsymbol{x}_i - \sqrt{1-\bar{\alpha}_i}\hat{\boldsymbol{s}}\big)$
  $\boldsymbol{x}'_{i-1} \leftarrow$ DDIM reverse with $\hat{\boldsymbol{x}}_0$ and $\hat{\boldsymbol{s}}$
  $\boldsymbol{x}_{i-1} \leftarrow$ (Approximately) Projection [39, 30, 33, 32, 40, 41, 34] or gradient update [20, 28, 19, 21, 29, 27, 26] with $\hat{\boldsymbol{x}}_0$ and $\boldsymbol{x}'_{i-1}$ to get closer to $\{\boldsymbol{x} | \boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$
end for

Output: Recovered object $\boldsymbol{x}_0$

Algorithm 2 Proposed plug-in method, DMPlug

Input: # Epochs E, # diffusion steps T, $\boldsymbol{y}$

Initialize seed $\boldsymbol{z}_T^0 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
for $e = 0$ to $E-1$ **do**
  for $i = T-1$ to 0 **do**
    $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(i)}(\boldsymbol{z}_i^e)$
    $\hat{\boldsymbol{z}}_0^e \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}\big(\boldsymbol{z}_i^e - \sqrt{1-\bar{\alpha}_i}\hat{\boldsymbol{s}}\big)$
    $\boldsymbol{z}_{i-1}^e \leftarrow$ DDIM reverse with $\hat{\boldsymbol{z}}_0^e, \hat{\boldsymbol{s}}$
  end for } $\mathcal{R}$
  Update $\boldsymbol{z}_T^{e+1}$ from $\boldsymbol{z}_T^e$ via a gradient update for Eq. (7)
end for

Output: Recovered object $\mathcal{R}\big(\boldsymbol{z}_T^{E-1}\big)$

Figure 4: Comparison of prevailing interleaving methods and our plug-in method

$\boldsymbol{x}_{i-1}$ for $\{\boldsymbol{x}|\boldsymbol{y} = \mathcal{A}(\boldsymbol{x})\}$ in each iteration [39, 30, 33, 32, 40, 41, 20]. However, this is possible only for easy IPs (e.g., linear IPs where we can perform a closed-form projection) and is difficult for typical nonlinear IPs as hard nonconvex problems are entailed. Moreover, although most of these methods have considered noisy IPs alongside noiseless ones [39, 30, 33, 32, 20, 28, 19, 21, 29], their common assumption about known noise types (often Gaussian) and levels (often low) is unrealistic: there are many types of measurement noise in practice—often hybrid from multiple sources [43], and the noise levels are usually unknown and hard to estimate [44], raising concerns about **(Issue 3) the robustness of these methods to unknown noise types and levels**; see Section 4.2.

3 Method

In this section, we propose a simple plug-in method, DMPlug, to solve IPs with pretrained DMs to address Issues 1 & 2 in Section 3.1, discuss its connection to and difference from several familiar ideas in Section 3.2, and finally explain how to integrate an early-stopping strategy, ES-WMV [35], into our plug-in method to address Issue 3 in Section 3.3, leading to an algorithm, DMPlug+ES-WMV, summarized in Algorithm 3, that solves IPs with pretrained DMs under the regularized data-fitting framework Eq. (1), even in the presence of unknown noise.

3.1 Our plug-in method: DMPlug

Reverse process as a function and our plug-in method Interleaving methods discussed above can have trouble satisfying both manifold feasibility and measurement feasibility because they interleave and hence modify both processes. Although projection-style modifications can improve measurement feasibility [39, 30, 33, 32, 40, 41, 20], their application to nonlinear IPs seems tricky, and a single "projection" step could be as difficult and expensive as solving the original problem due to the typical nonconvexity induced by the nonlinearity in $\mathcal{A}$ [28, 20]. In contrast, we propose **viewing the whole reverse process as a function** $\mathcal{R}(\cdot)$ that maps from the seed space to the object space (or the object manifold $\mathcal{M}$). Mathematically, if we write a single inverse step as a function g that depends on $\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(i)}$, i.e., $g_{\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(i)}}$ for the i-th reverse step that maps $\boldsymbol{x}_{i+1}$ to $\boldsymbol{x}_i$, then

$$\mathcal{R} = g_{\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(0)}} \circ g_{\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(1)}} \circ \cdots \circ g_{\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(T-2)}} \circ g_{\boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(T-1)}}. \qquad (\circ \text{ means function composition}) \tag{6}$$

Note that for the DDPM, and those DMs based on SDEs in general, $\mathcal{R}$ is a stochastic function due to noise injection in each step. To reduce technicality, we focus on DMs based on ordinary differential equations (ODEs), in particular, the DDIM, due to their increasing popularity [45, 46], resulting in deterministic $\mathcal{R}$'s. This conceptual leap allows us to reparametrize our object of interest as $\boldsymbol{x} = \mathcal{R}(\boldsymbol{z})$ and plug this reparametrization into the traditional regularized data-fitting framework in Eq. (1), yielding the following unified optimization formulation:

$$(\textbf{DMPlug})\ \boldsymbol{z}^* \in \arg\min_{\boldsymbol{z}}\ \ell(\boldsymbol{y}, \mathcal{A}(\mathcal{R}(\boldsymbol{z}))) + \Omega(\mathcal{R}(\boldsymbol{z})), \qquad \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*). \tag{7}$$

We stress that here the optimization is with respect to the seed variable $\boldsymbol{z}$, given the pretrained reverse process $\mathcal{R}$. Since we never modify the reverse diffusion process $\mathcal{R}$, we expect $\mathcal{R}(\boldsymbol{z})$ to

https://doi.org/10.52202/079017-3744

produce an object on the object manifold $\mathcal{M}$, i.e., enforcing manifold feasibility to **address Issue 1**. Moreover, optimizing the unified formulation Eq. (7) is expected to promote $\boldsymbol{y} \approx \mathcal{A}(\mathcal{R}(\boldsymbol{z}^*))$, inherent in the regularized data-fitting framework, i.e., promoting data feasibility to **address Issue 2**.

When there are multiple objects of interest in the IP under consideration, i.e., $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}_1, \cdots, \boldsymbol{x}_k) + \boldsymbol{n}$ with k objects (e.g., in blind image deblurring $\boldsymbol{y} = \mathcal{A}(\boldsymbol{k}, \boldsymbol{x}) + \boldsymbol{n}$, both the blur kernel $\boldsymbol{k}$ and sharp image $\boldsymbol{x}$ are objects of interest, where $\mathcal{A}(\boldsymbol{k}, \boldsymbol{x}) = \boldsymbol{k} * \boldsymbol{x}$; see also Appendix C.4), different objects may have different priors that should be encoded and treated differently. Our unified optimization formulation Eq. (7) facilitates the natural integration of multiple priors. Another bonus feature of our unified formulation lies in the flexibility in choosing numerical optimization solvers. We briefly explore both aspects in Section 4.3.

Fast samplers for memory and computational efficiency When implementing DMPlug with typical gradient-based solvers, e.g., *ADAM* [47], the gradient calculation in each iterative step requires a forward and a backward pass through the entire $\mathcal{R}$, i.e., T blocks of the basic DL model in the given DM. For high-quality image generation [23–25], T is typically tens or hundreds, resulting in prohibitive memory and computational burdens. To resolve this, we use the DDIM, which allows skipping reverse sampling steps thanks to its non-Markovian property, as the sampler $\mathcal{R}$. We observe, *to our surprise and also in our favor*, that a very small number of reverse steps, such as 3, is sufficient for our method to beat SOTA methods on all IPs we evaluate (see Fig. 5 and Section 4), and further increasing the number does not substantially improve the performance (actually even slightly degrades it perhaps due to the numerical difficulty caused by vanishing gradients as more steps are included). So, **we default the number of reverse steps to** 3 **unless otherwise stated**. This number is not sufficient for generating high-quality photorealistic images with existing DMs, but is good enough for our method. The discrepancy suggests a fundamental difference between image generation and image "regression" involved in solving IPs—we leave this for future work.

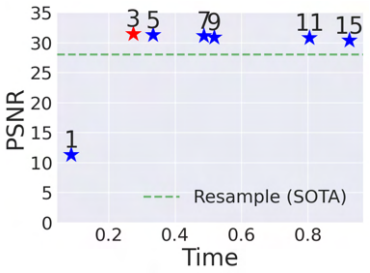

Figure 5: **PSNR (dB) vs. per-iteration wall-clock time (s)** running on an *NVIDIA A100*, for various reverse steps in $\mathcal{R}$. Experiments on CelebA for $4\times$ super-resolution; solver: *ADAM*; maximum iterations: $6,000$

3.2 Seemingly similar ideas

(A) GAN inversion for IPs Our plug-in method is reminiscent of GAN inversion to solve IPs [48, 49]: for a pretrained GAN generator $G_{\boldsymbol{\theta}}$, GAN inversion performs a similar reparametrization $\boldsymbol{x} = G_{\boldsymbol{\theta}}(\boldsymbol{z})$ and plugs it into Eq. (1). The remaining task is also to minimize the resulting formulation with respect to the trainable seed $\boldsymbol{z}$, and produce an $\boldsymbol{x} = G_{\boldsymbol{\theta}}(\boldsymbol{z}^*)$ through a solution $\boldsymbol{z}^*$. However, there are numerous signs that GAN inversion does not work well for IPs. For example, [50] also finetunes the generator $G_{\boldsymbol{\theta}}$ alongside the trainable seed to boost performance, and [33, 32] report superior performance of DM-based methods compared to GAN-inversion-based ones for solving IPs; **(B) Diffusion inversion (DI)** Given an object $\boldsymbol{x}$, DI aims to find a seed $\boldsymbol{z}$ so that $\mathcal{R}(\boldsymbol{z})$ reproduces $\boldsymbol{x}$, an important algorithmic component for DM-based image and text editing [51–53, 24, 54, 55]. A popular choice for DI is to modify DDIM [53, 24, 54, 55]. Although we cannot use DI to solve general IPs, DI can be considered as an IP and solved through $\min_{\boldsymbol{z}} \ell(\boldsymbol{x}, \mathcal{R}(\boldsymbol{z}))$ using our plug-in method. **(C) Algorithm unrolling (AU)** The multiblock structure in $\mathcal{R}$ also resembles those DL models used in AU, a popular family of supervised methods for solving IPs [56, 16]. In AU, the trainable DL model also consists of multiple blocks of basic DL models, induced by unfolded iterative steps to solve Eq. (1). While in AU the weights in these DL blocks are trainable, the weights in our DL blocks are fixed. More importantly, as a supervised approach, AU requires paired training sets of the form $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}_{i=1,\ldots,N}$, in contrast to the zero-shot nature of our method here.

3.3 Achieving robustness to unknown noise

The early-learning-then-overfitting (ELTO) phenomenon When $\boldsymbol{y}$ contains noise, solving Eq. (7) can promote measurement feasibility, i.e., $\boldsymbol{y} \approx \mathcal{A}(\mathcal{R}(\boldsymbol{z}^*))$, but $\mathcal{A}(\mathcal{R}(\boldsymbol{z}^*))$ may also learn the noise, i.e., **overfitting to noise**. Interestingly, **our method seems to favor desirable content and resist noise**: **(A)** our method converges much faster when used to regress clean natural images than random

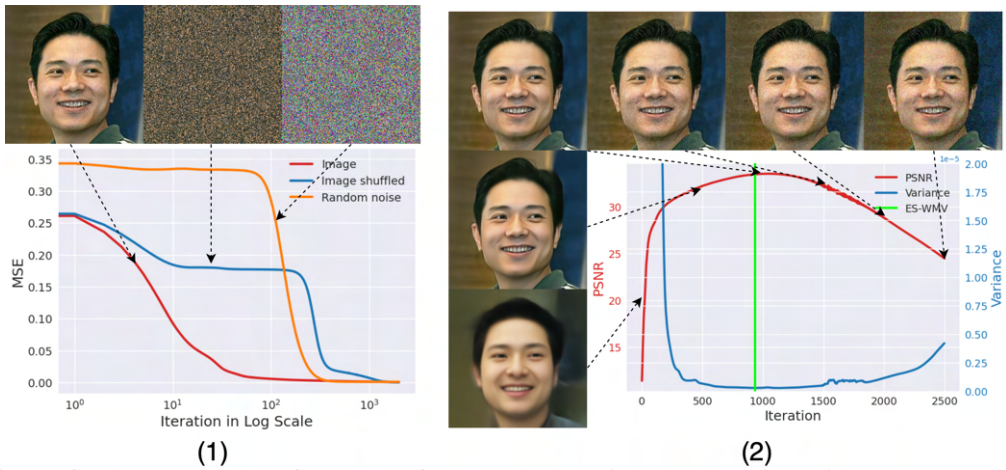

Figure 6: The recovery performance of our method on image regression with (1): (a) a clean natural image, (b) the same image with pixels randomly shuffled, (c) random noise iid sampled from $\mathrm{Uniform}(0, 1)$, and (2) a noisy natural image with Gaussian noise at $\sigma = 0.08$. Here, image regression means, given any image $\boldsymbol{x}$, performing $\min_{\boldsymbol{z}} \ \ell(\boldsymbol{x}, \mathcal{R}(\boldsymbol{z}))$.

noise (Fig. 6 (1), suggesting that our method shows high resistance to noise and low resistance to structured content, and **(B)** when performing regression against a noisy image $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{n}$, our method, although powerful enough to overfit the noisy image $\boldsymbol{x}$ ultimately, picks up the desired image content first and then learns the noise, leading to a hallmark "early-learning-then-overfitting" (**ELTO**) phenomenon so that **the recovery quality climbs to a peak before the potential degradation due to noise** (Fig. 6 (2)). We stress that similar ELTO phenomena have been widely reported in the literature on using deep image priors (DIPs) to solve IPs [57–61], although **this is the first time this phenomenon has been reported for DM-based methods**. Inspired by related studies in DIP, we also perform a spectral analysis of intermediate recovery and reveal that our method has **a spectral bias toward low-frequency components during learning**, similar to DIP methods; see Appendix B.

Achieving robustness via early stopping (ES) One may wonder how widely this ELTO phenomenon occurs when using our DMPlug to solve IPs. Besides extensive additional visual confirmation (see Appendix F), in Table 4 we show that our method, without using the noise information, leads the SOTA methods in terms of peak performance during iteration—particularly, by large margins on non-linear deblurring. The quantitative results suggest that the ELTO phenomenon is likely widespread, especially across noise types and levels. Hence, if we can perform proper early stopping (ES) to locate the peak performance, we **tackle Issue 3**. Deriving an effective ES strategy here is nontrivial, as in practice we do not have groundtruth images to compute any reference-based performance metrics such as PSNR. Fortunately, in the DIP literature, [35] discovers that for DIP-based methods for IPs, the valleys of the running-variance curves of the intermediate reconstructions are well aligned with the performance peaks. Based on this crucial observation, they propose an ES strategy, ES-WMV, that can accurately detect performance peaks with small performance loss on various IP tasks. Inspired by their success, we integrate the ES-WMV strategy into our plug-in method and find that ES-WMV is highly synergetic with our method and performs reliable ES with negligible performance loss (see Table 4). Details of the entire algorithm can be found in Algorithm 3.

4 Experiments

In this section, we evaluate our plug-in method, DMPlug, and compare it with other SOTA methods on two linear IPs, including **super-resolution** and **inpainting**, and three nonlinear IPs, including **nonlinear deblurring**, **blind image deblurring (BID)**, and **BID with turbulence**. Following [20], we construct the evaluation sets by sampling 100 images from CelebA [62], FFHQ [63] and LSUN-bedroom [64], respectively, and resizing all images to 256×256; we measure recovery quality using three standard metrics for image restoration, including peak signal-to-noise-ratio (PSNR), structural similarity index (SSIM) and Learned Perceptual Image Patch Similarity (LPIPS) [65] with the default backbone. We describe in detail the five IPs tested and the formulations we use for them in Appendix C; we provide implementation details of our method and the competing methods in

https://doi.org/10.52202/079017-3744

Appendix D; we include comparisons of computational costs, along with more quantitative and qualitative results in Appendix E.

Table 1: (Nonlinear IP) **Nonlinear deblurring** with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	CelebA [62] (256×256)			**FFHQ [63]** (256×256)			**LSUN [64]** (256×256)		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
BKS-styleGAN [66]	1.047	22.82	0.653	1.051	22.07	0.620	0.987	20.90	0.538
BKS-generic [66]	1.051	21.04	0.591	1.056	20.76	0.583	0.994	18.55	0.481
MCG [30]	0.705	13.18	0.135	0.675	13.71	0.167	0.698	14.28	0.188
ILVR [41]	0.335	21.08	0.586	0.374	20.40	0.556	0.482	18.76	0.444
DPS [19]	0.149	24.57	0.723	0.130	25.00	0.759	0.244	23.46	0.684
ReSample [20]	0.104	28.52	0.839	0.104	27.02	0.834	0.143	26.03	0.803
DMPlug (ours)	**0.073**	**31.61**	**0.882**	**0.057**	**32.83**	**0.907**	**0.083**	**30.74**	**0.882**
Ours vs. Best compe.	−0.031	+3.09	+0.043	−0.047	+5.79	+0.073	−0.060	+4.71	+0.079

4.1 IP tasks and experimental results

Linear IPs Due to space constraints, our experimental results on **Super-resolution & inpainting** are included in Appendix E.2. Our DMPlug can lead the best SOTA methods by about 2dB in PSNR and 0.02 in SSIM on average, respectively.

Nonlinear IPs **Nonlinear deblurring** We use the learned blurring operators from [66] with a known Gaussian-shaped kernel and Gaussian additive noise with $\sigma = 0.01$, following [20, 19]. We compare our DMPlug against several strong baselines: Blur Kernel Space (BKS)-styleGAN2 [66] based on GAN priors, BKS-generic [66] based on Hyper-Laplacian priors [68], and DM-based methods that can handle nonlinear IPs, including MCG, ILVR, DPS and ReSample. Despite the advancements made by the recent ReSample [20] to enhance the DPS method, from Table 1, it is evident that our DMPlug can still significantly outperform the SOTA ReSample by substantial margins. Specifically, our method improves LPIPS, PSNR, and SSIM by 0.05, 4.5dB, and 0.07, respectively, across the three datasets on average. In addition, our DMPlug delivers a more faithful and precise restoration of details, as shown in Fig. 1.

Table 2: (Nonlinear IP) **BID with turbulence** with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	CelebA [62] (256×256)			**FFHQ [63]** (256×256)		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
DeBlurGANv2 [5]	0.356	24.17	0.739	0.379	23.52	0.723
Stripformer [6]	0.318	24.97	0.745	0.343	24.30	0.725
MPRNet [7]	0.379	22.64	0.696	0.399	22.28	0.682
TSR-WGAN [67]	0.304	23.09	0.732	0.333	22.56	0.715
ILVR [41]	0.337	21.25	0.589	0.375	20.24	0.554
BlindDPS [21]	**0.137**	25.45	0.730	0.165	24.40	0.712
DMPlug (ours)	0.146	**28.34**	**0.790**	**0.164**	**27.91**	**0.812**
Ours vs. Best compe.	−0.009	+2.89	+0.045	−0.001	+3.51	+0.087

BID & BID with turbulence BID is about recovering a sharp image $\boldsymbol{x}$ (and kernel $\boldsymbol{k}$) from $\boldsymbol{y} = \boldsymbol{k} * \boldsymbol{x} + \boldsymbol{n}$ where $*$ denotes the linear convolution and the spatially invariant blur kernel $\boldsymbol{k}$ is unknown; for BID with turbulence which often arises in long-range imaging through atmospheric turbulence, we model the forward process as a simplified "tilt-then-blur" process, following [69]: $\boldsymbol{y} = \boldsymbol{k} * \mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{x}) + \boldsymbol{n}$, where $\mathcal{T}_{\boldsymbol{\phi}}(\cdot)$ is the tilt operation parameterized by unknown $\boldsymbol{\phi}$ (see more details in Appendix C). We mainly compare our DMPlug to two DM-based models, including ILVR and BlindDPS [21], which is an extension of DPS. In addition, we choose two classical MAP-based methods, i.e., Pan-Dark Channel Prior (Pan-DCP) [70] and Pan-ℓ_0 [71], DIP-based SelfDeblur [72], and four methods that are based on supervised training on paired datasets, including DeBlurGANv2 [5], Stripformer [6], MPRNet [7] and TSR-WGAN [67]. It is important to mention that **BlindDPS [21] use pretrained DMs not only for images but also for blur kernels (and tilt maps), lending it an unfair advantage over other methods**. Although our method is flexible in working with multiple DM priors, as shown in Section 4.3, we opt to only use the pretrained DMs for images to ensure a fair comparison. Tables 2 and 3 show that our method, despite using fewer priors than BlindDPS, can surpass the best competing methods by approximately 0.03, 4.5dB, and 0.1 in terms of LPIPS, PSNR, and SSIM, respectively, on average. In Fig. 1, the reconstructions of our method look sharper and more precise than those of the main competitors.

Table 3: (Nonlinear IP) **BID** with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	CelebA [62] (256×256)						**FFHQ [63]** (256×256)					
	Motion blur			**Gaussian blur**			**Motion blur**			**Gaussian blur**		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
SelfDeblur [72]	0.568	16.59	0.417	0.579	16.55	0.423	0.628	16.33	0.408	0.604	16.22	0.410
DeBlurGANv2 [5]	0.313	20.56	0.613	0.350	24.29	0.743	0.353	19.67	0.581	0.374	23.58	0.726
Stripformer [6]	0.287	22.06	0.644	0.316	25.03	0.747	0.324	21.31	0.613	0.339	24.34	0.728
MPRNet [7]	0.332	20.53	0.620	0.375	22.72	0.698	0.373	19.70	0.590	0.394	22.33	0.685
Pan-DCP [70]	0.606	15.83	0.483	0.653	20.57	0.701	0.616	15.59	0.464	0.667	20.69	0.698
Pan-ℓ_0 [71]	0.631	15.16	0.470	0.654	20.49	0.675	0.642	14.43	0.443	0.669	20.34	0.671
ILVR [41]	0.398	19.23	0.520	0.338	21.20	0.588	0.445	18.33	0.484	0.375	20.45	0.555
BlindDPS [21]	0.164	23.60	0.682	0.173	25.15	0.721	0.185	21.77	0.630	0.193	23.83	0.693
DMPlug (ours)	**0.104**	**29.61**	**0.825**	**0.140**	**28.84**	**0.795**	**0.135**	**27.99**	**0.794**	**0.169**	**28.26**	**0.811**
Ours vs. Best compe.	−0.060	+6.01	+0.143	−0.033	+3.69	+0.048	−0.050	+6.22	+0.164	−0.024	+3.92	+0.083

4.2 Robustness to unknown noise

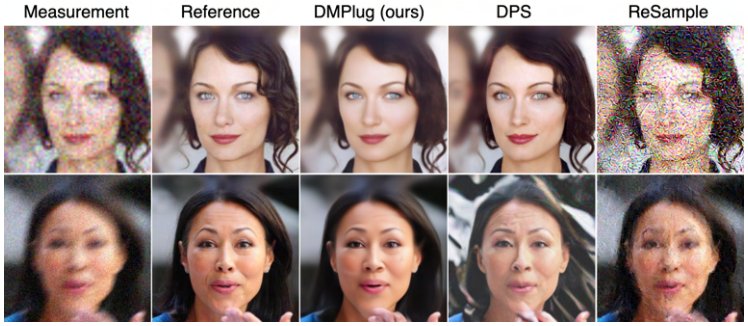

Figure 7: (**Robustness**) Visualization of sample results from our DMPlug and main competing methods for $4\times$ super-resolution (**top**) and nonlinear deblurring (**bottom**) with low-level Gaussian noise.

For robustness experiments, we choose super-resolution and nonlinear deblurring to represent linear and nonlinear IPs, respectively. To simulate scenarios involving unknown noise, we generate measurements with four types of noise: Gaussian, impulse, shot, and speckle noise, and across two different noise levels: low (level-1) and high (level-2), following [43] (see details in Appendix D.1), but we use the same formulation and code for each IP designed for mild Gaussian noise, regardless of the actual noise type and level. Table 4 and Fig. 7 clearly show that (1) **most current IP solvers, except for DPS, suffer from the robustness issue**, corroborating the hypotheses made in Section 2, and (2) the peak performance of our DMPlug can lead SOTA methods by around 1dB and 3.5dB in PSNR for the two tasks, respectively. To check the compatibility of our method with ES-WMV [35], we measure the detection performance via PSNR gaps, i.e., the absolute PSNR difference between the peak and the detected ES point following [35, 59]. Table 4 indicates that the detection gaps in the two exemplary tasks are nearly negligible, with PSNR gaps smaller than 0.5dB and 0.2dB, respectively. This suggests that **the proposed method is highly synergetic with ES-WMV**.

4.3 Ablation studies

We conduct two ablation studies to demonstrate the flexibility of our method in terms of using multiple and non-DDIM DM priors and using alternative optimizers. **(Flexibility of using DM priors)** First, we explore the possibility of using different types of DMs. For super-resolution, we show in Table 5 that latent diffusion models (LDMs) [74, 25] are also synergetic with our method. Next, we study the potential of using multiple DMs together, taking BID with turbulence as an example. Using extra pretrained DMs for blur kernels and tilt maps from [21], our method can achieve even better reconstruction results. **(Flexibility of using alternative optimizers)** Here, we test the built-in *ADAM* [47] and *L-BFGS* [75] optimizers in *PyTorch*, with several different learning rates to solve Eq. (7). As shown in Table 6, the best *ADAM* and *L-BFGS* combinations can lead to comparable performance for our DMPlug. We choose *ADAM* as the default optimizer because in *PyTorch*, optimizing multiple groups of variables with different learning rates—as for the case of BID (with turbulence)—is easy to program with *ADAM* but tricky for *L-BFGS*.

https://doi.org/10.52202/079017-3744

Table 4: (**Robustness and ES**) **Super-resolution** and **nonlinear deblurring** on CelebA [62] with different types and levels of noise. We only show PSNR↑ and PSNR Gap↓ to save space. (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	(Linear) Super-resolution (4×)				**(Nonlinear) Non-uniform image deblurring**			
	Gaussian	**Impulse**	**Shot**	**Speckle**	**Gaussian**	**Impulse**	**Shot**	**Speckle**
	Low/High	Low/High	Low/High	Low/High	Low/High	Low/High	Low/High	Low/High
ADMM-PnP [73]	20.17/17.97	14.28/14.52	19.97/17.82	19.42/18.41	N/A	N/A	N/A	N/A
DMPS [29]	20.62/17.54	18.78/16.05	19.96/16.74	20.77/18.73	N/A	N/A	N/A	N/A
DDRM [32]	15.45/14.79	14.82/14.14	15.31/14.59	15.46/15.03	N/A	N/A	N/A	N/A
MCG [30]	17.43/15.83	16.39/15.07	17.19/15.49	17.44/16.43	12.88/12.85	13.16/13.04	13.21/13.13	13.24/13.07
ILVR [41]	21.08/21.03	20.93/20.00	21.19/21.12	20.96/20.89	21.70/21.43	21.43/21.00	21.56/21.24	21.53/21.36
DPS [19]	25.51/24.58	24.89/23.92	25.47/24.27	25.69/24.97	23.97/23.35	23.74/23.18	24.32/23.58	23.45/23.61
ReSample [20]	14.30/13.04	15.56/13.48	14.38/12.87	15.64/14.23	23.17/20.45	20.69/18.91	22.94/20.11	23.59/21.66
BKS-styleGAN [66]	N/A	N/A	N/A	N/A	22.61/22.53	22.64/22.34	22.96/22.79	22.70/22.56
BKS-generic [66]	N/A	N/A	N/A	N/A	16.85/15.09	14.86/13.44	16.69/14.74	17.04/15.99
DMPlug (ours)	**26.49/25.29**	**26.01/24.76**	**26.34/26.34**	**26.81/25.81**	**27.58/26.60**	**27.22/26.13**	**27.71/26.55**	**27.68/26.96**
Ours vs. Best compe.	0.98/0.71	1.12/0.84	0.87/2.07	1.12/0.84	3.61/3.25	3.48/2.95	3.39/2.97	4.23/3.35
PSNR Gap↓	0.36/0.46	0.38/0.60	0.25/0.49	0.20/0.21	0.15/0.12	0.14/0.13	0.10/0.19	0.12/0.09

Table 5: **(Flexibility)** Ablation on different **priors for x, k and ϕ** on 50 cases from CelebA [62] for super-resolution (SR), nonlinear deblurring (ND), and BID with turbulence.

	Prior for $\boldsymbol{x}$	Prior for $\boldsymbol{k}$	Prior for $\boldsymbol{\phi}$	PSNR↑
SR	DM	N/A	N/A	31.51
	LDM	N/A	N/A	31.10
ND	DM	N/A	N/A	31.61
	LDM	N/A	N/A	30.64
Turbulence	DM	Simplex	None	28.21
	DM	Simplex+DM	None	29.33
	DM	Simplex	DM	28.35
	DM	Simplex+DM	DM	28.91

Table 6: **(Flexibility)** Ablation on different **optimizers and learning rates** on 50 cases from CelebA [62] for super-resolution.

	Learning rate	LPIPS↓	PSNR↑	SSIM↑
ADAM	0.1	0.375	20.22	0.479
	0.01	0.066	31.51	0.879
	0.001	0.093	31.45	0.879
L-BFGS	1.0	0.254	28.53	0.786
	0.1	0.173	31.89	0.888
	0.01	0.225	30.20	0.839

5 Discussion

In this paper, we focus on solving IPs with pretrained DMs. To deal with **(Issue 1)** insufficient manifold feasibility and **(Issue 2)** insufficient measurement feasibility of the prevailing interleaving methods, we **pioneer a novel plug-in method**, DMPlug, and make it practical in terms of computation and memory requirements. Taking advantage of a benign ELTO property and integrating an ES method ES-WMV [35], our method is **the first to achieve robustness to unknown noise (Issue 3)**. Extensive experiment results demonstrate that our method can lead SOTA methods, both qualitatively and quantitatively—often by large margins, particularly for nonlinear IPs. As for limitations, our empirical results in Section 3.1 suggest a fundamental gap between image generation and regression using pretrained DMs, that we have not managed to nail down. Also, our work is mostly empirical, and we leave a solid theoretical understanding for future work.

Acknowledgements

Wang H. is partially supported by a UMN CSE DSI PhD Fellowship. Wan Y. is partially supported by a UMN CSE InterS&Ections Seed Grant. This research is part of AI-CLIMATE:"AI Institute for Climate-Land Interactions, Mitigation, Adaptation, Tradeoffs and Economy," and is supported by USDA National Institute of Food and Agriculture (NIFA) and the National Science Foundation (NSF) National AI Research Institutes Competitive Award no. 2023-67021-39829. The authors acknowledge the Minnesota Supercomputing Institute (MSI) at the University of Minnesota for providing resources that contributed to the research results reported in this article.

References

[1] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art," Mar. 2021, arXiv:1704.05519 [cs]. [Online]. Available: http://arxiv.org/abs/1704.05519

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*, ser. Texts in Computer Science. Cham: Springer International Publishing, 2022. [Online]. Available: https://link.springer.com/10.1007/978-3-030-34372-9

[3] R. Olsen and R. Olsen, "Introduction to Remote Sensing," Jan. 2007, book Title: Remote Sensing from Air and Space ISBN: 9780819462350 Publisher: SPIE. [Online]. Available: https://spiedigitallibrary.org/eBooks/PM/Remote-Sensing-from-Air-and-Space/Chapter1/Introduction-to-Remote-Sensing/10.1117/3.673407.ch1

[4] J. Sylvester and C. L. Epstein, "Introduction to the Mathematics of Medical Imaging," in *The American Mathematical Monthly*, vol. 112, May 2005, p. 479, iSSN: 00029890 Issue: 5. [Online]. Available: https://www.jstor.org/stable/10.2307/30037514?origin=crossref

[5] O. Kupyn, T. Martyniuk, J. Wu, and Z. Wang, "DeblurGAN-v2: Deblurring (Orders-of-Magnitude) Faster and Better," Aug. 2019, arXiv:1908.03826 [cs]. [Online]. Available: http://arxiv.org/abs/1908.03826

[6] F.-J. Tsai, Y.-T. Peng, Y.-Y. Lin, C.-C. Tsai, and C.-W. Lin, "Stripformer: Strip Transformer for Fast Image Deblurring," Jul. 2022, arXiv:2204.04627 [cs]. [Online]. Available: http://arxiv.org/abs/2204.04627

[7] S. W. Zamir, A. Arora, S. Khan, M. Hayat, F. S. Khan, M.-H. Yang, and L. Shao, "Multi-Stage Progressive Image Restoration," Mar. 2021, arXiv:2102.02808 [cs]. [Online]. Available: http://arxiv.org/abs/2102.02808

[8] M. Delbracio and P. Milanfar, "Inversion by direct iteration: An alternative to denoising diffusion for image restoration," *Transactions on Machine Learning Research*, 2023, featured Certification. [Online]. Available: https://openreview.net/forum?id=VmyFF5lL3F

[9] D. Gilton, G. Ongie, and R. Willett, "Deep Equilibrium Architectures for Inverse Problems in Imaging," Jun. 2021, arXiv:2102.07944 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2102.07944

[10] Z. Luo, F. K. Gustafsson, Z. Zhao, J. Sjölund, and T. B. Schön, "Image Restoration with Mean-Reverting Stochastic Differential Equations," May 2023, arXiv:2301.11699 [cs]. [Online]. Available: http://arxiv.org/abs/2301.11699

[11] L. Guo, C. Wang, W. Yang, S. Huang, Y. Wang, H. Pfister, and B. Wen, "ShadowDiffusion: When Degradation Prior Meets Diffusion Model for Shadow Removal," Dec. 2022, arXiv:2212.04711 [cs]. [Online]. Available: http://arxiv.org/abs/2212.04711

[12] J. Whang, M. Delbracio, H. Talebi, C. Saharia, A. G. Dimakis, and P. Milanfar, "Deblurring via Stochastic Refinement," Dec. 2021, arXiv:2112.02475 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2112.02475

[13] S. Gao, X. Liu, B. Zeng, S. Xu, Y. Li, X. Luo, J. Liu, X. Zhen, and B. Zhang, "Implicit Diffusion Models for Continuous Super-Resolution," Sep. 2023, arXiv:2303.16491 [cs]. [Online]. Available: http://arxiv.org/abs/2303.16491

[14] O. Kupyn, V. Budzan, M. Mykhailych, D. Mishkin, and J. Matas, "DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks," Apr. 2018, arXiv:1711.07064 [cs]. [Online]. Available: http://arxiv.org/abs/1711.07064

[15] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett, "Deep Learning Techniques for Inverse Problems in Imaging," May 2020, arXiv:2005.06001 [cs, eess, stat]. [Online]. Available: http://arxiv.org/abs/2005.06001

[16] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing," Aug. 2020, arXiv:1912.10557 [cs, eess]. [Online]. Available: http://arxiv.org/abs/1912.10557

[17] K. Zhang, W. Ren, W. Luo, W.-S. Lai, B. Stenger, M.-H. Yang, and H. Li, "Deep Image Deblurring: A Survey," May 2022, arXiv:2201.10700 [cs]. [Online]. Available: http://arxiv.org/abs/2201.10700

[18] J. Koh, J. Lee, and S. Yoon, "Single-image deblurring with neural networks: A comparative survey," *Computer Vision and Image Understanding*, vol. 203, p. 103134, Feb. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1077314220301533

[19] H. Chung, J. Kim, M. T. Mccann, M. L. Klasky, and J. C. Ye, "Diffusion Posterior Sampling for General Noisy Inverse Problems," Feb. 2023, arXiv:2209.14687 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2209.14687

[20] B. Song, S. M. Kwon, Z. Zhang, X. Hu, Q. Qu, and L. Shen, "Solving Inverse Problems with Latent Diffusion Models via Hard Data Consistency," Oct. 2023, arXiv:2307.08123 [cs]. [Online]. Available: http://arxiv.org/abs/2307.08123

[21] H. Chung, J. Kim, S. Kim, and J. C. Ye, "Parallel Diffusion Models of Operator and Image for Blind Inverse Problems," Nov. 2022, arXiv:2211.10656 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2211.10656

[22] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis," Jun. 2021, arXiv:2105.05233 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2105.05233

[23] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," Dec. 2020, arXiv:2006.11239 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2006.11239

 https://doi.org/10.52202/079017-3744

[24] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," Oct. 2022, arXiv:2010.02502 [cs]. [Online]. Available: http://arxiv.org/abs/2010.02502
[25] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," Apr. 2022, arXiv:2112.10752 [cs]. [Online]. Available: http://arxiv.org/abs/2112.10752
[26] Y. He, N. Murata, C.-H. Lai, Y. Takida, T. Uesaka, D. Kim, W.-H. Liao, Y. Mitsufuji, J. Z. Kolter, R. Salakhutdinov, and S. Ermon, "Manifold Preserving Guided Diffusion," Oct. 2023. [Online]. Available: https://openreview.net/forum?id=o3BxOLoxm1
[27] X. Xu and Y. Chi, "Provably Robust Score-Based Diffusion Posterior Sampling for Plug-and-Play Image Reconstruction." [Online]. Available: https://www.semanticscholar.org/paper/Provably-Robust-Score-Based-Di%EF%AC%80usion-Posterior-for-Xu-Chi/a713ae4d638ef679a7165fbf7301a6c045e2588d?utm_source=alert_email&utm_content=LibraryFolder&utm_campaign=AlertEmails_DAILY&utm_term=LibraryFolder&email_index=0-0-0&utm_medium=33840575&citedSort=relevance&citedQueryString=Denoising%20Diffusion%20Models%20for%20Plug-and-Play%20Image%20Restoration
[28] Y. Zhu, K. Zhang, J. Liang, J. Cao, B. Wen, R. Timofte, and L. Van Gool, "Denoising Diffusion Models for Plug-and-Play Image Restoration," May 2023, arXiv:2305.08995 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2305.08995
[29] X. Meng and Y. Kabashima, "Diffusion Model Based Posterior Sampling for Noisy Linear Inverse Problems," Jan. 2024, arXiv:2211.12343 [cs, math, stat]. [Online]. Available: http://arxiv.org/abs/2211.12343
[30] H. Chung, B. Sim, D. Ryu, and J. C. Ye, "Improving Diffusion Models for Inverse Problems using Manifold Constraints," Oct. 2022, arXiv:2206.00941 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2206.00941
[31] X. Li, Y. Ren, X. Jin, C. Lan, X. Wang, W. Zeng, X. Wang, and Z. Chen, "Diffusion Models for Image Restoration and Enhancement – A Comprehensive Survey," Aug. 2023, arXiv:2308.09388 [cs]. [Online]. Available: http://arxiv.org/abs/2308.09388
[32] B. Kawar, M. Elad, S. Ermon, and J. Song, "Denoising Diffusion Restoration Models," Oct. 2022, arXiv:2201.11793 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2201.11793
[33] Y. Wang, J. Yu, and J. Zhang, "Zero-Shot Image Restoration Using Denoising Diffusion Null-Space Model," Dec. 2022, arXiv:2212.00490 [cs]. [Online]. Available: http://arxiv.org/abs/2212.00490
[34] G. Liu, H. Sun, J. Li, F. Yin, and Y. Yang, "Accelerating Diffusion Models for Inverse Problems through Shortcut Sampling," May 2024, arXiv:2305.16965 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2305.16965
[35] H. Wang, T. Li, Z. Zhuang, T. Chen, H. Liang, and J. Sun, "Early Stopping for Deep Image Prior," Dec. 2023, arXiv:2112.06074 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2112.06074
[36] A. Hyvärinen, "Estimation of Non-Normalized Statistical Models by Score Matching," *Journal of Machine Learning Research*, vol. 6, no. 24, pp. 695–709, 2005. [Online]. Available: http://jmlr.org/papers/v6/hyvarinen05a.html
[37] Y. Song and S. Ermon, "Generative Modeling by Estimating Gradients of the Data Distribution," Oct. 2020, arXiv:1907.05600 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1907.05600
[38] J. Song, C. Meng, and A. Vahdat, "Denoising diffusion models: A generative learning big bang," Nov. 2023, CVPR 2023 Tutorial. [Online]. Available: https://cvpr2023-tutorial-diffusion-models.github.io/
[39] Z. Kadkhodaie and E. Simoncelli, "Stochastic Solutions for Linear Inverse Problems using the Prior Implicit in a Denoiser," in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 13 242–13 254. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/6e28943943dbed3c7f82fc05f269947a-Abstract.html
[40] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-Based Generative Modeling through Stochastic Differential Equations," Feb. 2021, arXiv:2011.13456 [cs, stat]. [Online]. Available: http://arxiv.org/abs/2011.13456
[41] J. Choi, S. Kim, Y. Jeong, Y. Gwon, and S. Yoon, "ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models," Sep. 2021, arXiv:2108.02938 [cs]. [Online]. Available: http://arxiv.org/abs/2108.02938
[42] Y. Sanghvi, Y. Chi, and S. H. Chan, "Kernel diffusion: An alternate approach to blind deconvolution," *arXiv preprint arXiv:2312.02319*, 2023.
[43] D. Hendrycks and T. Dietterich, "Benchmarking Neural Network Robustness to Common Corruptions and Perturbations," Mar. 2019, arXiv:1903.12261 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1903.12261
[44] F. Li, F. Fang, Z. Li, and T. Zeng, "Single image noise level estimation by artificial noise," *Signal Processing*, vol. 213, p. 109215, Dec. 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016516842300289X
[45] C. Lu, Y. Zhou, F. Bao, J. Chen, C. Li, and J. Zhu, "DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5775–5787, Dec. 2022. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/hash/260a14acce2a89dad36adc8eefe7c59e-Abstract-Conference.html
[46] K. Zheng, C. Lu, J. Chen, and J. Zhu, "DPM-Solver-v3: Improved Diffusion ODE Solver with Empirical Model Statistics," *Advances in Neural Information Processing Systems*, vol. 36, pp.

55 502–55 542, Dec. 2023. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2023/hash/ada8de994b46571bdcd7eeff2d3f9cff-Abstract-Conference.html
[47] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017, arXiv:1412.6980 [cs]. [Online]. Available: http://arxiv.org/abs/1412.6980
[48] A. Creswell and A. A. Bharath, "Inverting The Generator Of A Generative Adversarial Network," Nov. 2016, arXiv:1611.05644 [cs]. [Online]. Available: http://arxiv.org/abs/1611.05644
[49] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative Visual Manipulation on the Natural Image Manifold," Dec. 2018, arXiv:1609.03552 [cs]. [Online]. Available: http://arxiv.org/abs/1609.03552
[50] X. Pan, X. Zhan, B. Dai, D. Lin, C. C. Loy, and P. Luo, "Exploiting Deep Generative Prior for Versatile Image Restoration and Manipulation," Jul. 2020, arXiv:2003.13659 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2003.13659
[51] X. Su, J. Song, C. Meng, and S. Ermon, "Dual Diffusion Implicit Bridges for Image-to-Image Translation," Mar. 2023, arXiv:2203.08382 [cs]. [Online]. Available: http://arxiv.org/abs/2203.08382
[52] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or, "Prompt-to-Prompt Image Editing with Cross Attention Control," Aug. 2022, arXiv:2208.01626 [cs]. [Online]. Available: http://arxiv.org/abs/2208.01626
[53] G. Kim, T. Kwon, and J. C. Ye, "DiffusionCLIP: Text-Guided Diffusion Models for Robust Image Manipulation," Aug. 2022, arXiv:2110.02711 [cs]. [Online]. Available: http://arxiv.org/abs/2110.02711
[54] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, and D. Cohen-Or, "Null-text Inversion for Editing Real Images using Guided Diffusion Models," Nov. 2022, arXiv:2211.09794 [cs]. [Online]. Available: http://arxiv.org/abs/2211.09794
[55] B. Wallace, A. Gokul, and N. Naik, "EDICT: Exact Diffusion Inversion via Coupled Transformations," pp. 22 532–22 541, 2023. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2023/html/Wallace_EDICT_Exact_Diffusion_Inversion_via_Coupled_Transformations_CVPR_2023_paper.html
[56] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, Jun. 2010, pp. 399–406.
[57] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep Image Prior," *International Journal of Computer Vision*, vol. 128, no. 7, pp. 1867–1888, Jul. 2020, arXiv:1711.10925 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1711.10925
[58] T. Li, Z. Zhuang, H. Liang, L. Peng, H. Wang, and J. Sun, "Self-Validation: Early Stopping for Single-Instance Deep Generative Priors," Oct. 2021, arXiv:2110.12271 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2110.12271
[59] T. Li, H. Wang, Z. Zhuang, and J. Sun, "Deep Random Projector: Accelerated Deep Image Prior," pp. 18 176–18 185, 2023. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2023/html/Li_Deep_Random_Projector_Accelerated_Deep_Image_Prior_CVPR_2023_paper.html
[60] Z. Shi, P. Mettes, S. Maji, and C. G. M. Snoek, "On Measuring and Controlling the Spectral Bias of the Deep Image Prior," Dec. 2021, arXiv:2107.01125 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2107.01125
[61] Z. Zhuang, T. Li, H. Wang, and J. Sun, "Blind Image Deblurring with Unknown Kernel Size and Substantial Noise," *International Journal of Computer Vision*, vol. 132, no. 2, pp. 319–348, Feb. 2024, arXiv:2208.09483 [cs, eess]. [Online]. Available: http://arxiv.org/abs/2208.09483
[62] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep Learning Face Attributes in the Wild," Sep. 2015, arXiv:1411.7766 [cs]. [Online]. Available: http://arxiv.org/abs/1411.7766
[63] T. Karras, S. Laine, and T. Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," Mar. 2019, arXiv:1812.04948 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1812.04948
[64] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop," Jun. 2016, arXiv:1506.03365 [cs]. [Online]. Available: http://arxiv.org/abs/1506.03365
[65] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," Apr. 2018, arXiv:1801.03924 [cs]. [Online]. Available: http://arxiv.org/abs/1801.03924
[66] P. Tran, A. T. Tran, Q. Phung, and M. Hoai, "Explore Image Deblurring via Encoded Blur Kernel Space," pp. 11 956–11 965, 2021. [Online]. Available: https://openaccess.thecvf.com/content/CVPR2021/html/Tran_Explore_Image_Deblurring_via_Encoded_Blur_Kernel_Space_CVPR_2021_paper.html
[67] D. Jin, Y. Chen, Y. Lu, J. Chen, P. Wang, Z. Liu, S. Guo, and X. Bai, "Neutralizing the impact of atmospheric turbulence on complex scene imaging via deep learning," *Nat. Mach. Intell.*, vol. 3, no. 10, pp. 876–884, 2021. [Online]. Available: https://doi.org/10.1038/s42256-021-00392-1
[68] D. Krishnan and R. Fergus, "Fast Image Deconvolution using Hyper-Laplacian Priors," in *Advances in Neural Information Processing Systems*, vol. 22. Curran Associates, Inc., 2009. [Online]. Available: https://papers.nips.cc/paper_files/paper/2009/hash/3dd48ab31d016ffcbf3314df2b3cb9ce-Abstract.html
[69] S. H. Chan, "Tilt-then-Blur or Blur-then-Tilt? Clarifying the Atmospheric Turbulence Model," *IEEE Signal Processing Letters*, vol. 29, pp. 1833–1837, 2022, arXiv:2207.06377 [eess]. [Online]. Available: http://arxiv.org/abs/2207.06377

[70] J. Pan, D. Sun, H. Pfister, and M.-H. Yang, "Deblurring Images via Dark Channel Prior," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 10, pp. 2315–2328, Oct. 2018, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. [Online]. Available: https://ieeexplore.ieee.org/document/8048543

[71] J. Pan, Z. Hu, Z. Su, and M.-H. Yang, "L_0 -Regularized Intensity and Gradient Prior for Deblurring Text Images and Beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 2, pp. 342–355, Feb. 2017, conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7448477?casa_token=BY5p4dV5BVwAAAAA:LaPxWMHRFDbrXQYcJmp247D-oOPlvPitreoe4MXKkd53Ku96bdoLz_-JE7lug9H9Cm-CxJlpxkI

[72] D. Ren, K. Zhang, Q. Wang, Q. Hu, and W. Zuo, "Neural Blind Deconvolution Using Deep Priors," Mar. 2020, arXiv:1908.02197 [cs]. [Online]. Available: http://arxiv.org/abs/1908.02197

[73] S. H. Chan, X. Wang, and O. A. Elgendy, "Plug-and-Play ADMM for Image Restoration: Fixed Point Convergence and Applications," Nov. 2016, arXiv:1605.01710 [cs]. [Online]. Available: http://arxiv.org/abs/1605.01710

[74] A. Vahdat, K. Kreis, and J. Kautz, "Score-based generative modeling in latent space," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 287–11 302, 2021.

[75] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, Aug. 1989. [Online]. Available: https://doi.org/10.1007/BF01589116

[76] H. Chihaoui, A. Lemkhenter, and P. Favaro, "Zero-shot Image Restoration via Diffusion Inversion," Oct. 2023. [Online]. Available: https://openreview.net/forum?id=ZnmofqLWMQ

[77] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[78] P. Chakrabarty and S. Maji, "The Spectral Bias of the Deep Image Prior," Dec. 2019, arXiv:1912.08905 [cs]. [Online]. Available: http://arxiv.org/abs/1912.08905

[79] D. L. Ruderman, "The statistics of natural images," *Network: Computation in Neural Systems*, vol. 5, no. 4, pp. 517–548, Jan. 1994. [Online]. Available: https://www.tandfonline.com/doi/full/10.1088/0954-898X_5_4_006

[80] Z. Dou and Y. Song, "Diffusion Posterior Sampling for Linear Inverse Problem Solving: A Filtering Perspective," Oct. 2023. [Online]. Available: https://openreview.net/forum?id=tplXNcHZs1

[81] L. Rout, N. Raoof, G. Daras, C. Caramanis, A. G. Dimakis, and S. Shakkottai, "Solving Linear Inverse Problems Provably via Posterior Sampling with Latent Diffusion Models," Jul. 2023. [Online]. Available: https://arxiv.org/abs/2307.00619v1

A Remarks on the concurrent work [76]

We became aware of the unpublished concurrent work, SHRED [76], when we were finalizing the details of our method in mid Jan 2024 (We started the current project in Oct 2023). Although the core idea of SHRED is the same as that of our DMPlug, there are a few crucial differences between [76] and the current paper. **(1) Motivation**: [76] aims only at addressing manifold feasibility, while the current paper targets manifold feasibility, measurement feasibility, and robustness to unknown noise, simultaneously; **(2) Integral components**: Since achieving robustness to unknown noise is part of our goal, the ES-WMV ES strategy is integral to our method, besides the unified optimization formulation Eq. (7) shared with [76]; **(3) Key hyperparameters**: To address the computational and memory bottleneck induced by reverse steps, we use only 3 reverse steps for all IPs we test, vs. the 10 or more reverse steps used in [76]. So, our current setting makes the method much more practical; **(4) Flexibility**: Our framework allows for the use of more than one pre-trained DM prior when available, as shown in Section 4.3, while [76] uses only a single DM prior as proposed in their Eqs. (17) to (19); **(5) Experimental evaluation**: [76] focuses on linear IPs, including inpainting, super-resolution, compressive sensing, and one nonlinear IP—blind deconvolution, i.e., blind image deburring (BID). We focus our evaluation on nonlinear IPs, including nonlinear deblurring, BID, and BID with turbulence, besides linear IPs (inpainting and super-resolution). Moreover, [76] measures performance only by perceptual metrics, LPIPS [65] and FID [77], while we measure performance by both the perceptual metric LPIPS and the classical metrics PSNR and SSIM. In addition, [76] does not even compare their method with clear SOTA methods that use pre-trained DM, e.g., DPS [19], which they obviously were aware of (cited in the paper), whereas our comparison is more comprehensive.

B Spectral bias of our DMPlug

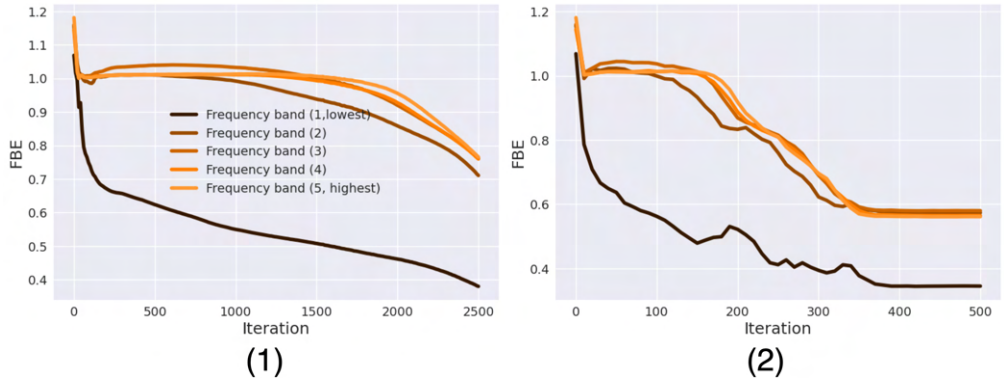

Figure 8: (1) The spectral bias of DMPlug with the *ADAM* solver [47]. (2) The spectral bias of DMPlug with the *L-BFGS* solver [75]. The IP we experiment with here is regression against a noisy image with Gaussian noise at $\sigma = 0.08$, i.e., the same denoising problem as in Fig. 6 (2).

One may wonder why the ELTO phenomenon occurs. Here, we borrow the ideas of spectral biases and spectral analysis from the DIP literature [78, 60, 59, 61] to shed some light on this. The theory of spectral biases for DIP states that low-frequency components are learned much faster than high-frequency components during DIP learning. **Spectral biases in DIP lead to the ELTO phenomenon in DIP**: because natural images are typically low-frequency dominant [79], the different learning paces imply that DIP learns mostly the desired image content (low-frequency image content plus the low-frequency part of noise) in the early stage, but gradually picks up the high-frequency part of noise in the late stage, resulting in performance degradation after a certain quality peak. Spectral analysis provides a quantitative visualization of spectral biases.

Here, we perform a similar spectral analysis of our DMPlug learning process to demonstrate its spectral biases, which causes the ELTO phenomenon. To measure spectral biases, we follow [59, 61] and use *frequency band errors (FBEs)*. For an groundtruth image $\boldsymbol{x}$ and its estimate $\widehat{\boldsymbol{x}}$, the calculation of this metric goes as follows. First, we calculate the pointwise relative error pointwise in the Fourier

https://doi.org/10.52202/079017-3744

domain, i.e., $|\mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\widehat{\boldsymbol{x}})| \,/ |\mathcal{F}(\boldsymbol{x})|$. Then, we divide the Fourier frequencies into five radial bands, compute the bandwise mean errors, i.e., the frequency-band errors (FBEs).

We visualize the evolution of FBEs of DMPlug over all five frequency bands in Fig. 8. The disparate learning paces across the frequency bands are evident: the lowest-frequency band is learned much more rapidly than the other bands, which is consistent between two different optimization solvers. With spectral biases similar to those in DIP, we can explain the ELTO phenomenon in DMPlug following the argument above for DIP.

C Setup details of the inverse problems we test

For super-resolution, inpainting, and nonlinear deblurring, we use the forward models from two of main competing methods [19, 20]; for BID and BID with turbulence, we follow the forward models from BlindDPS [21]—the SOTA DM-based method. Moreover, following [19, 20], all the measurements contain additive Gaussian noise with $\sigma = 0.01$. Our loss ℓ by default is the MSE loss.

C.1 Super-resolution

In noisy image super-resolution, the goal is to reconstruct a clean RGB image $\boldsymbol{x}$ from a noisy downsampled version $\boldsymbol{y} = \mathcal{D}(\boldsymbol{x}) + \boldsymbol{n}$, where $\mathcal{D}(\cdot) : [0,1]^{3\times tH\times tW} \to [0,1]^{3\times H\times W}$ is a downsampling operator that resizes an image with dimensions tH by tW by the factor t and $\boldsymbol{n}$ models additive noise. We set $t = 4$ in Section 4. To ensure a fair comparison, we do not include any explicit regularization terms in the formulation:

$$(\textbf{Super-resolution})\ \boldsymbol{z}^* = \arg\min_{\boldsymbol{z}}\ \ell(\boldsymbol{y}, \mathcal{D}(\mathcal{R}(\boldsymbol{z}))), \qquad \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*). \tag{8}$$

C.2 Inpainting

In noisy image inpainting, a clean RGB image $\boldsymbol{x} \in [0,1]^{3\times H\times W}$ is only partially observed and then contaminated by additive noise $\boldsymbol{n}$, described by the forward model $\boldsymbol{y} = \boldsymbol{m} \odot \boldsymbol{x} + \boldsymbol{n}$, where $\boldsymbol{m} \in \{0,1\}^{3\times H\times W}$ is a binary mask and $\odot$ denotes the Hadamard product. Given $\boldsymbol{y}$ and $\boldsymbol{m}$, the goal is to reconstruct $\boldsymbol{x}$. Following [20], the masks for the three channels of $\boldsymbol{m}$ are identical, and 70% of the mask values are randomly set to 0. To ensure a fair comparison, we do not include any explicit regularization terms in the formulation:

$$(\textbf{Inpainting})\ \boldsymbol{z}^* = \arg\min_{\boldsymbol{z}}\ \ell(\boldsymbol{y}, \boldsymbol{m} \odot \mathcal{R}(\boldsymbol{z})), \qquad \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*). \tag{9}$$

C.3 Nonlinear deblurring

We follow the setup in [19] which is inspired by [66]. Recently, [66] has proposed to learn data-driven blurring models from paired blurry-sharp training sets of the form $\{(\boldsymbol{y}_i, \boldsymbol{x}_i)\}_{i=1,\dots,N}$ through

$$\boldsymbol{\alpha}^*, \boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \sum_{i=1}^{N} ||\boldsymbol{y}_i - \mathcal{F}_{\boldsymbol{\alpha}}(\boldsymbol{x}_i, \mathcal{G}_{\boldsymbol{\beta}}(\boldsymbol{x}_i, \boldsymbol{y}_i))||, \tag{10}$$

where $\mathcal{G}_{\boldsymbol{\beta}}(\cdot,\cdot)$ predicts the latent blur kernel associated with the input blurry-sharp image pair, and $\psi_{\boldsymbol{\beta}}(\cdot,\cdot)$ models real-world nonlinear blurring process given the input image-kernel pair. To study the performance of its DPS method on nonlinear IPs, [19] proposes the following nonlinear deblurring problem with a known Gaussian-shaped kernel:

$$\boldsymbol{y} = \mathcal{F}_{\boldsymbol{\alpha}^*}(\boldsymbol{x}, \boldsymbol{g}) + \boldsymbol{n}, \quad \text{where } \boldsymbol{g} \in \mathbb{R}^{64\times 64} \text{ is Gaussian-shaped with } \sigma = 3.0. \tag{11}$$

The task here is to recover $\boldsymbol{x}$ from $\boldsymbol{y}$ and the forward model $\mathcal{F}_{\boldsymbol{\alpha}^*}(\cdot, \boldsymbol{g})$. Our formulation follows [19] and does not include extra regularizers:

$$(\textbf{Nonlinear deblurring})\ \boldsymbol{z}^* = \arg\min_{\boldsymbol{z}}\ \ell(\boldsymbol{y}, \mathcal{F}_{\boldsymbol{\alpha}^*}(\mathcal{R}(\boldsymbol{z}), \boldsymbol{g})), \qquad \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*). \tag{12}$$

C.4 Blind image deblurring (BID)

BID is about recovering a sharp image $\boldsymbol{x}$ from $\boldsymbol{y} = \boldsymbol{k} * \boldsymbol{x} + \boldsymbol{n}$ where $*$ denotes the linear convolution and the spatially invariant blur kernel $\boldsymbol{k}$ is also unknown. It is a nonlinear IP because the forward

model $\mathcal{A}(\boldsymbol{k}, \boldsymbol{x}) = \boldsymbol{k} * \boldsymbol{x}$ is nonlinear. In solving BID under the regularized data-fitting framework in Eq. (1), ℓ_2 or ℓ_1 data-fitting loss, and the sparse gradient prior to $\boldsymbol{x}$ enforced by $R_{\boldsymbol{x}}(\boldsymbol{x}) = \|\nabla \boldsymbol{x}\|_1$ or $R_{\boldsymbol{x}}(\boldsymbol{x}) = \|\nabla \boldsymbol{x}\|_1 / \|\nabla \boldsymbol{x}\|_2$ are typically used. In addition, because of the scale ambiguity in the forward model, i.e., $\boldsymbol{k} * \boldsymbol{x} = (\alpha \boldsymbol{k}) * (\frac{1}{\alpha} \boldsymbol{x})$ for any $\alpha > 0$, the scale of $\boldsymbol{k}$ is often fixed by requiring $\boldsymbol{k}$ to be on the standard **simplex** (i.e., $\boldsymbol{k} \geq \boldsymbol{0}, \boldsymbol{1}^\intercal \boldsymbol{k} = 1$) or on the sphere (i.e., $\|\boldsymbol{k}\|_2 = 1$). [18, 17, 61] provide detailed coverage of these priors and regularizers. For our experiments here, we follow the settings in BlindDPS [19]: all the blur kernels are simulated with the size of 64×64; the standard deviation of the Gaussian kernels is set to 3.0 and the intensity of the motion blur kernels is adjusted to 0.5. It is important to mention that **BlindDPS [21] uses pretrained DMs not only for images but also for blur kernels, giving it an unfair advantage over other methods**. But to ensure a fair comparison with other competing methods that only use data-driven image priors, we only use pretrained DMs for the image, plus the typical simplex constraint that is also used in BlindDPS:

$$(\textbf{BID})\ \boldsymbol{z}^*, \boldsymbol{k}^* = \arg\min_{\boldsymbol{z}, \boldsymbol{k}}\ \ell(\boldsymbol{y}, \mathrm{SoftMax}(\boldsymbol{k}) * \mathcal{R}(\boldsymbol{z})), \qquad \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*), \tag{13}$$

where $\mathrm{SoftMax}(\boldsymbol{k})$ leads to a kernel estimate that lies on the standard simplex.

C.5 BID with turbulence

BID with turbulence often arises in long-range imaging through atmospheric turbulence. The forward imaging process can be modeled as a simplified "tilt-then-blur" process, following [69]: $\boldsymbol{y} = \boldsymbol{k} * \mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{x}) + \boldsymbol{n}$, where the tilt operator $\mathcal{T}_{\boldsymbol{\phi}}(\cdot)$ applies the spatially varying vector field $\boldsymbol{\phi}$ to the image $\boldsymbol{x}$ so that pixels of $\boldsymbol{x}$ are moved around according to the vector field $\boldsymbol{\phi}$, i.e., $\mathcal{T}_{\boldsymbol{\phi}}(\boldsymbol{x})[\boldsymbol{p}_i + \boldsymbol{\phi}_i] = \boldsymbol{x}[\boldsymbol{p}_i]$ where $\boldsymbol{p}_i$'s are the pixel coordinates. In BID with turbulence, none of the $\boldsymbol{k}, \boldsymbol{x}, \boldsymbol{\phi}$ is known and the task is to jointly estimate them from the measurement $\boldsymbol{y}$. So, it is clear that this BID variant is strictly more difficult than BID itself. We follow BlindDPS [21] for data generation: the blur kernel comes from the point spread function (PSF) and takes a Gaussian shape with standard deviation 3.0; the tilt maps are generated as iid Gaussian random vectors over the pixel grid. Similar to the BID case, **BlindDPS [21] use pretrained DMs for all three objects: image, kernel, and tilt map, unfair to other methods**. We again use pretrained DMs for the image only. For the kernel, we again only impose the simplex constraint through a $\mathrm{SoftMax}$ activation. For the tilt map, inspired by the fact that the tilt vectors are small-magnitude random vectors, we initialize the map from a zero-mean Gaussian distribution with a small standard deviation and set an extremely small learning rate. Our final formulation for this task is

$$(\textbf{BID with turbulence})\ \boldsymbol{z}^*, \boldsymbol{k}^*, \boldsymbol{\phi}^* = \arg\min_{\boldsymbol{z}, \boldsymbol{k}, \boldsymbol{\phi}}\ \ell(\boldsymbol{y}, \mathrm{SoftMax}(\boldsymbol{k}) * \mathcal{T}_{\boldsymbol{\phi}}(\mathcal{R}(\boldsymbol{z}))),\ \boldsymbol{x}^* = \mathcal{R}(\boldsymbol{z}^*). \tag{14}$$

D More implementation details

D.1 Noise generation

Following [43][3], we simulate four types of noise, with two intensity levels for each type. The detailed information is as follows. **Gaussian noise:** zero-mean additive Gaussian noise with variance 0.08 and 0.12 for low and high noise levels, respectively; **Impulse noise:** also known as salt-and-pepper noise, replacing each pixel with probability $p \in [0, 1]$ in a white or black pixel with half chance each. Low and high noise levels correspond to $p = 0.03$ and 0.06, respectively; **Shot noise:** also known as Poisson noise. For each pixel, $x \in [0, 1]$, the noisy pixel is Poisson distributed with the rate λx, where λ is 60 and 25 for low and high noise levels, respectively; **Speckle noise:** for each pixel $x \in [0, 1]$, the noisy pixel is $x(1 + \epsilon)$, where ϵ is zero-mean Gaussian with a variance level 0.15 and 0.20 for low and high noise levels, respectively.

D.2 Additional implementation details of our method

We employ the following setup for our methods across all IP tasks. For $\mathcal{R}(\cdot)$, we take the standard pretrained DMs from [22][4] and [41][5] and use the standard DDIM [24] sampler with only 3 reverse steps based on Fig. 5; we also use pretrained latent diffusion models (LDMs) [25][6] to obtain the

[3] https://github.com/hendrycks/robustness

[4] https://github.com/openai/guided-diffusion

[5] https://github.com/jychoi118/ilvr_adm?tab=readme-ov-file

[6] https://github.com/CompVis/latent-diffusion

https://doi.org/10.52202/079017-3744

results reported in Table 5. We use the pretrained DMs for blur kernels and tilt maps from [21][7] to obtain the results reported in Table 5. For $\ell(\cdot)$, we choose the standard *MSE* loss. For $\Omega(\cdot)$, we use the typical explicit regularizers for each task to make the comparisons fair. The default optimizer is *ADAM* and the learning rate (LR) for $\boldsymbol{z}$ is 1×10^{-2}; for BID (with turbulence), the LRs for blur kernel and the tilt map are 1×10^{-1} and 1×10^{-7}, respectively. For the maximum numbers of iterations, we set $5,000$ and $10,000$ for linear and nonlinear IPs, respectively, which empirically allow good convergence. We perform all experiments on *NVIDIA A100* GPUs with 40GB memory each.

Algorithm 3 DMPlug+ES–WMV for solving general IPs

Input: # diffusion steps T, $\boldsymbol{y}$, window size W, patience P, empty queue $\mathcal{Q}$, iteration counter $e = 0$, $\mathrm{VAR}_{\min} = \infty$

while not stopped **do**
  for $i = T - 1$ to 0 **do**
    $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{\varepsilon}_{\boldsymbol{\theta}}^{(i)}(\boldsymbol{z}_i^e)$
    $\hat{\boldsymbol{z}}_0^e \leftarrow \frac{1}{\sqrt{\bar{\alpha}_i}}\left(\boldsymbol{z}_i^e - \sqrt{1-\bar{\alpha}_i}\hat{\boldsymbol{s}}\right)$
    $\boldsymbol{z}_{i-1}^e \leftarrow$ DDIM reverse with $\hat{\boldsymbol{z}}_0^e, \hat{\boldsymbol{s}}$
  end for
  Update $\boldsymbol{z}_T^{e+1}$ from $\boldsymbol{z}_T^e$ via a gradient update for Eq. (7)
  push $\mathcal{R}\left(\boldsymbol{z}_T^{e+1}\right)$ to $\mathcal{Q}$, pop queue if $|\mathcal{Q}| > W$
  if $|\mathcal{Q}| = W$ **then**
    compute VAR of elements in $\mathcal{Q}$ via Eq. (15)
    if $\mathrm{VAR} < \mathrm{VAR}_{\min}$ **then**
      $\mathrm{VAR}_{\min} \leftarrow \mathrm{VAR}$, $\boldsymbol{z}^* \leftarrow \boldsymbol{z}_T^{e+1}$
    end if
    if $\mathrm{VAR}_{\min}$ stagnates for P iterations **then**
      stop and return $\boldsymbol{z}^*$
    end if
  end if
  $e = e + 1$
end while

Output: Recovered object $\mathcal{R}(\boldsymbol{z}^*)$

Early stopping (ES) using ES-WMV The "early-learning-then-overfitting" (**ELTO**) phenomenon has been widely reported in the literature on deep image prior (DIP) [57–60, 35], and ES-WMV [35] is an ES strategy that achieves the SOTA ES performance for DIP applied to various IPs. In ES-EMV, the running variance (VAR) is defined as:

$$\mathrm{VAR}(t) \doteq \frac{1}{W}\sum_{w=0}^{W-1} \|\boldsymbol{x}^{t+w} - 1/W \cdot \sum_{i=0}^{W-1} \boldsymbol{x}^{t+i}\|_F^2, \tag{15}$$

where W is the window size and $\boldsymbol{x}^i$ denotes the recovery at iteration i. [35] observes that **the first major valley** of the VAR curve is often well aligned with the peak of the PSNR curve. Based on this, [35] introduces an online algorithm to detect the first major valley of the VAR curve: if the minimal VAR does not change over P consecutive steps, i.e., the VAR does not decrease further over P consecutive steps, the iteration process is stopped. The combined algorithm, ES-WMV-integrated DMPlug, is described in Algorithm 3. For implementation, we use the official code of ES-WMV[8]. For super-resolution with unknown noise, we set its patience number as 100 and its window size as 10; for nonlinear deblurring with unknown noise, we set its patience number as 300 and its window size as 50. Table 4 indicates that the detection gaps in the two exemplary tasks are nearly negligible, with PSNR gaps smaller than 0.5dB and 0.2dB, respectively. This suggests that **our DMPlug is highly synergetic with ES-WMV**.

D.3 Implementations of competing methods

We use the default code and settings of each competitor's official implementation, listed below.

[7] `https://github.com/BlindDPS/blind-dps`

[8] `https://github.com/sun-umn/Early_Stopping_for_DIP`

- ADMM-PnP [73]: https://github.com/kanglin755/plug_and_play_admm
- DMPS [29]: https://github.com/mengxiangming/dmps
- DDRM [32]: https://github.com/bahjat-kawar/ddrm
- DPS [19] & MCG [30]: https://github.com/DPS2022/diffusion-posterior-sampling
- ILVR [41]: https://github.com/jychoi118/ilvr_adm
- ReSample [20]: https://github.com/soominkwon/resample/tree/main
- BKS [66]: https://github.com/VinAIResearch/blur-kernel-space-exploring
- SelfDeblur [72]: https://github.com/csdwren/SelfDeblur
- DeBlurGANv2 [5]: https://github.com/VITA-Group/DeblurGANv2
- Stripformer [6]: https://github.com/pp00704831/Stripformer-ECCV-2022-
- MPRNet [7]: https://github.com/swz30/MPRNet
- Pan-DCP [70]: https://jspan.github.io/projects/dark-channel-deblur/index.html
- Pan-ℓ_0 [71]: https://jspan.github.io/projects/text-deblurring/index.html
- BlindDPS [21]: https://github.com/BlindDPS/blind-dps
- TSR-WGAN [67]: https://codeocean.com/capsule/9958894/tree/v1
- FPS [80]: https://github.com/ZehaoDou-official/FPS-SMC-2023
- DiffPIR [28]: https://github.com/yuanzhi-zhu/DiffPIR

E More experiment results

E.1 Computational Efficiency

As shown in Table 7, we find that the memory usage of our method is in the same order as that of most other algorithms. When it comes to speed, our implementation using L-BFGS is significantly faster than the ADAM version. Although our method with L-BFGS remains somewhat slow, it surpasses the performance of ReSample. We will explore further acceleration of our method in future work.

Table 7: Wall-clock time (seconds) and memory usage (GB) of various algorithms for **super-resolution** on the CelebA [62] dataset, tested on a single *NVIDIA A100 GPU*. (**Ours-A**: ours with ADAM, **Ours-L**: ours with L-BFGS)

	ADMM-PnP	**DMPS**	**DDRM**	**MCG**	**DPS**	**DDNM**	**FPS**	**DiffPIR**	**ReSample**	**Ours-A**	**Ours-L**
Time	6	42	30	43	43	14	62	4	367	635	255
Memory	0.42	5.10	4.99	2.80	2.79	5.11	20.63	1.44	4.87	6.59	6.74

E.2 Quantitative results

Super-resolution & inpainting Following [20, 19], to generate measurements, we use *bicubic* downsampling for super-resolution ($4\times$) and a random mask with 70% missing pixels for inpainting; all measurements contain additive Gaussian noise with $\sigma = 0.01$. We compare our method with Plug-and-Play using ADMM (ADMM-PnP) [73] and several SOTA DM-based methods: Diffusion Model based Posterior Sampling (DMPS) [29], Denoising Diffusion Destoration Models (DDRM) [32], Manifold Constrained Gradients (MCG) [30], Iterative Latent Variable Refinement (ILVR) [41], Diffusion Posterior Sampling (DPS) [19], and ReSample [20]. The quantitative results are reported in Tables 8 to 10, with qualitative results visualized in Fig. 1. It is clear that our DMPlug consistently outperforms all competing methods on the three datasets, both quantitatively and qualitatively. Specifically, while there is no significant improvements in terms of LPIPS, the proposed method can lead the best SOTA methods by about 2dB in PSNR and 0.02 in SSIM on average, respectively.

E.3 Qualitative results

F More early stopping results

https://doi.org/10.52202/079017-3744

Table 8: (Linear IPs) **Super-resolution** and **inpainting** with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	Super-resolution ($4\times$)						**Inpainting (Random** 70%**)**					
	CelebA [62] (256×256)			**FFHQ [63]** (256×256)			**CelebA [62]** (256×256)			**FFHQ [63]** (256×256)		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
ADMM-PnP [73]	0.217	26.99	0.808	0.229	26.25	0.794	0.091	31.94	0.923	0.104	30.64	0.901
DMPS [29]	0.070	28.89	0.848	**0.076**	28.03	0.843	0.297	24.52	0.693	0.326	23.31	0.664
DDRM [32]	0.226	26.34	0.754	0.282	25.11	0.731	0.185	26.10	0.712	0.201	25.44	0.722
MCG [30]	0.725	19.88	0.323	0.786	18.20	0.271	1.283	10.16	0.049	1.276	10.37	0.050
ILVR [41]	0.322	21.63	0.603	0.360	20.73	0.570	0.447	15.82	0.484	0.483	15.10	0.450
DPS [19]	0.087	28.32	0.823	0.098	27.44	0.814	0.043	32.24	0.924	0.046	30.95	0.913
ReSample [20]	0.080	28.29	0.819	0.108	25.22	0.773	**0.039**	30.12	0.904	0.044	27.91	0.884
DMPlug (ours)	**0.067**	**31.25**	**0.878**	0.079	**30.25**	**0.871**	**0.039**	**34.03**	**0.936**	**0.038**	**33.01**	**0.931**
Ours vs. Best compe.	-0.003	$+2.36$	$+0.030$	$+0.003$	$+2.22$	$+0.028$	-0.000	$+1.79$	$+0.012$	-0.006	$+2.06$	$+0.018$

Table 9: (Linear IPs) **Super-resolution** and **inpainting** on LSUN-bedroom [64] with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	Super-resolution ($4\times$)			**Inpainting (Random** 70%**)**		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
ADMM-PnP [73]	0.358	24.09	0.728	0.223	27.75	0.842
DMPS [29]	0.172	25.54	0.772	0.320	23.09	0.712
DDRM [32]	0.415	22.73	0.644	0.273	23.66	0.673
MCG [30]	1.069	15.00	0.176	1.271	11.15	0.055
ILVR [41]	0.476	18.89	0.445	0.581	14.79	0.407
DPS [19]	0.195	25.10	0.737	0.094	28.95	0.868
ReSample [20]	0.137	24.72	0.762	0.069	26.43	0.861
DMPlug (ours)	**0.136**	**26.93**	**0.801**	**0.064**	**30.15**	**0.905**
Ours vs. Best compe.	-0.001	$+1.39$	$+0.029$	-0.005	$+1.20$	$+0.037$

Table 10: (Linear IPs) **Super-resolution** and **inpainting** on CelebA [62] with additive Gaussian noise ($\sigma = 0.01$). (**Bold**: best, under: second best, green: performance increase, red: performance decrease)

	Super-resolution ($4\times$)			**Inpainting (Random** 70%**)**		
	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑
FPS [80]	0.149	29.12	0.858	0.064	32.06	0.924
DiffPIR [28]	0.203	**31.55**	0.857	0.219	31.22	0.866
DDNM [33]	0.193	29.21	0.836	0.224	27.89	0.799
PSLD [81]	0.243	26.45	0.682	0.213	27.65	0.785
DMPlug (ours)	**0.067**	31.25	**0.878**	**0.039**	**34.03**	**0.936**
Ours vs. Best compe.	-0.082	-0.30	$+0.020$	-0.025	$+1.97$	$+0.012$

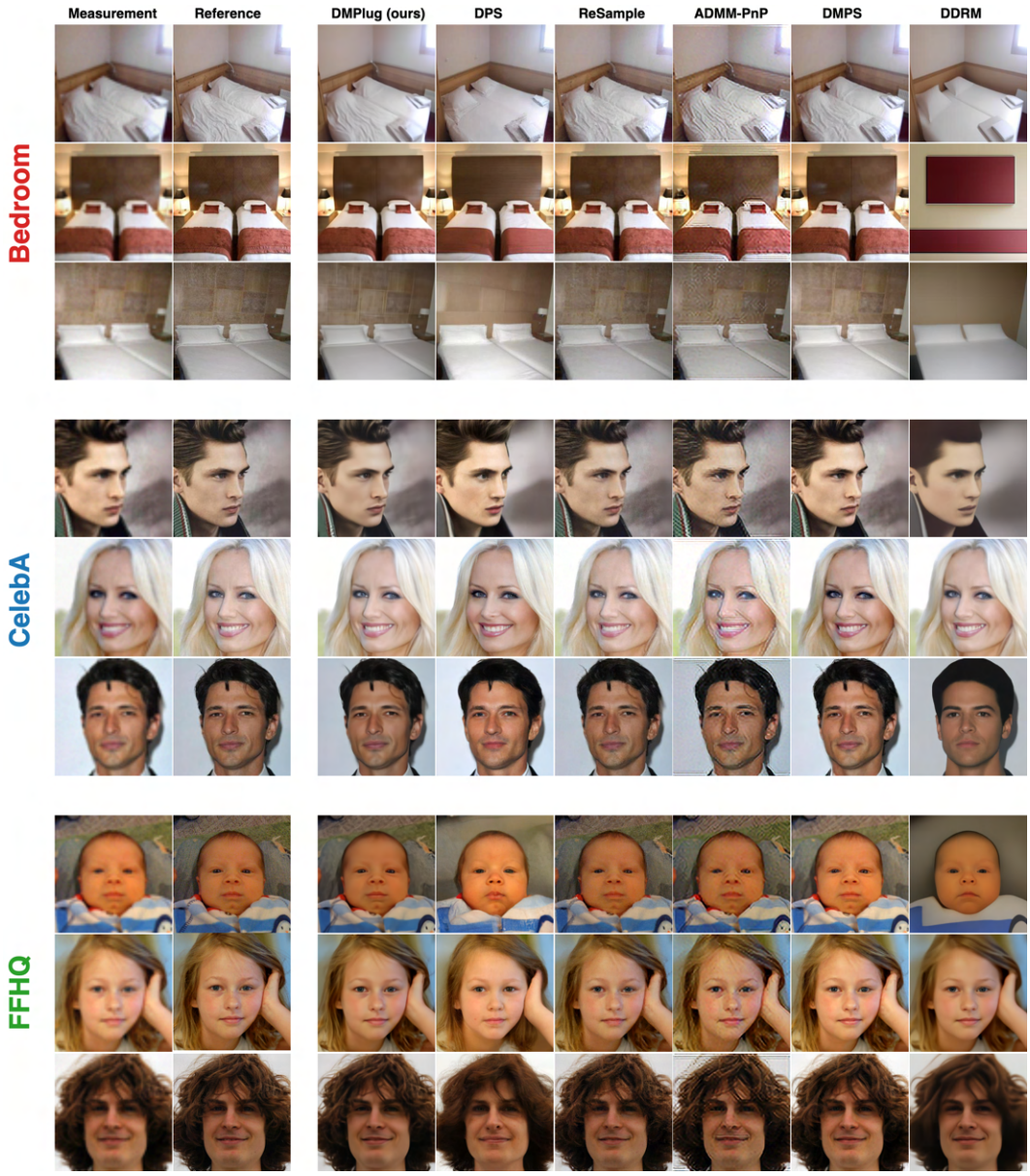

Figure 9: (Linear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for $4\times$ **super-solution**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

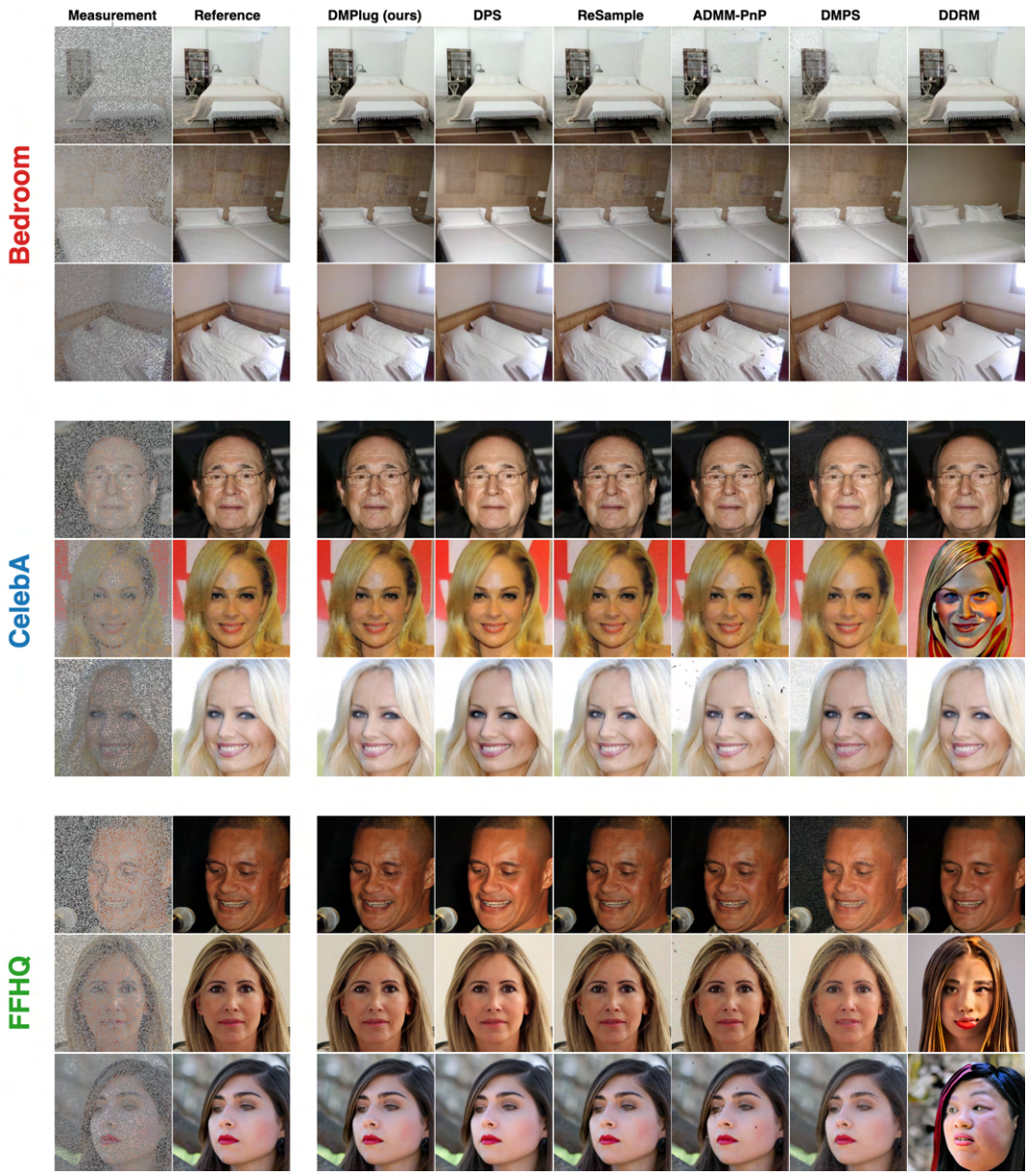

Figure 10: (Linear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **inpainting (random** 70%**)**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

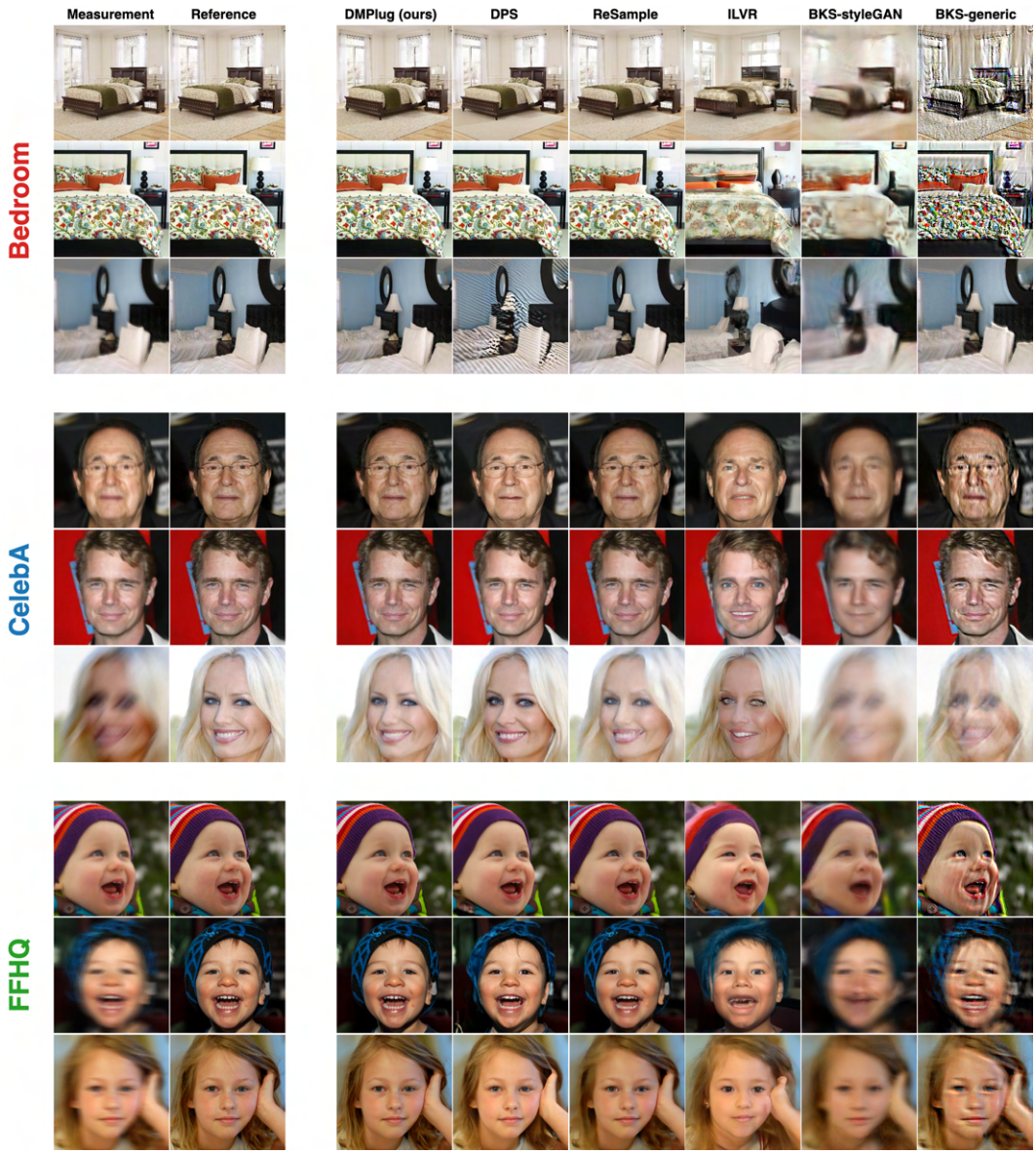

Figure 11: (Nonlinear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **nonlinear deblurring**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

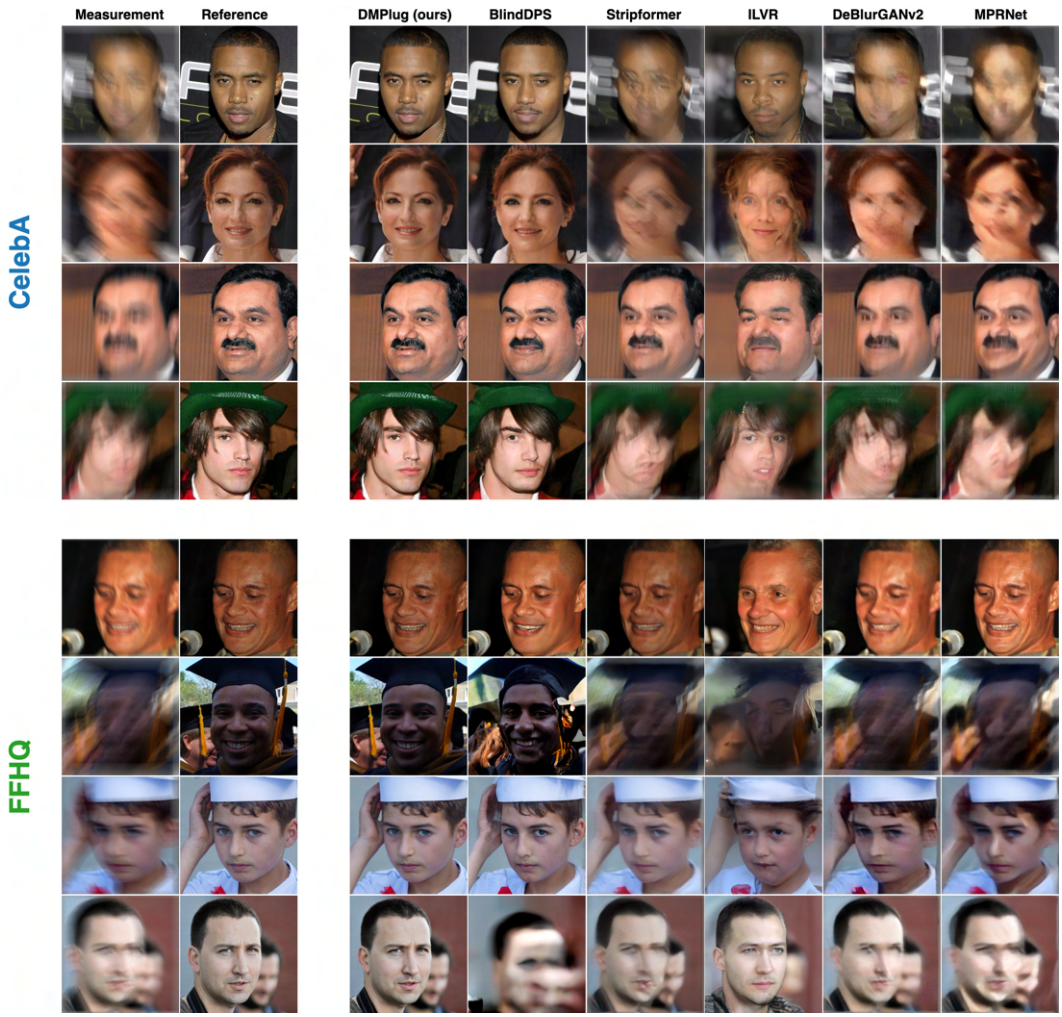

Figure 12: (Nonlinear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **BID (motion)**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

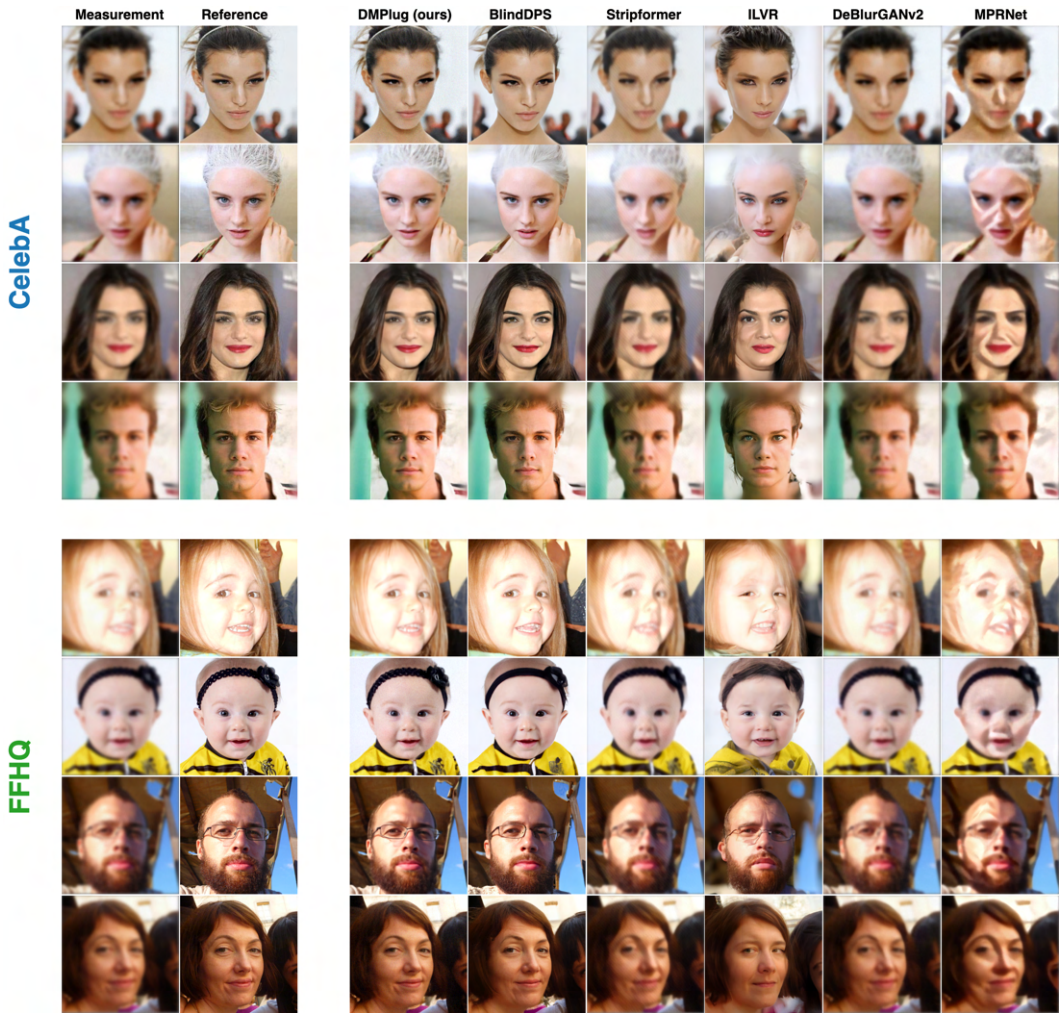

Figure 13: (Nonlinear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **BID (Gaussian)**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

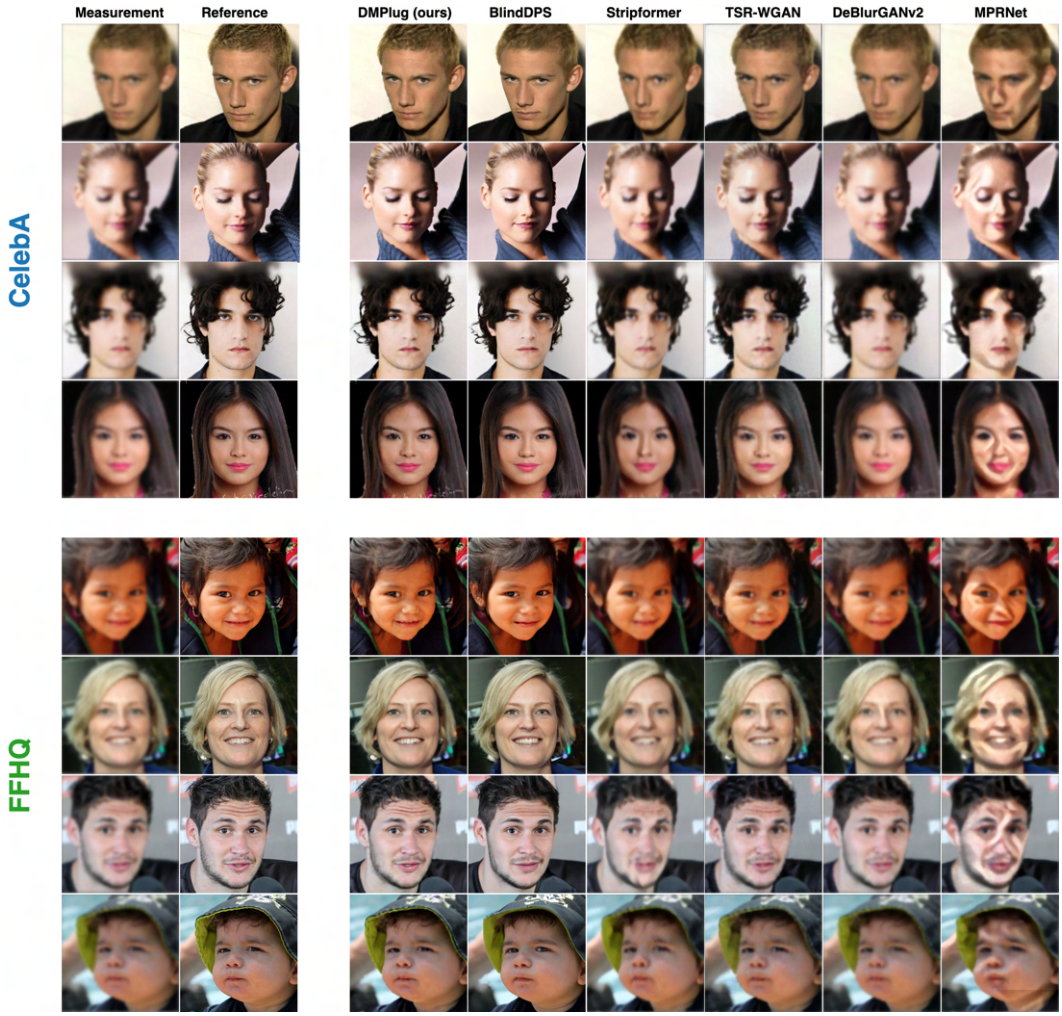

Figure 14: (Nonlinear IP) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **BID with turbulence**. All measurements contain Gaussian noise with $\sigma = 0.01$.

https://doi.org/10.52202/079017-3744

Figure 15: (Robustness) Visualization of sample results from the plug-in method (**Ours**) and competing methods for $4\times$ **super-resolution**. We generate measurements with four types of noise—Gaussian, impulse, shot, and speckle noise—across two different noise levels: low (level-1) and high (level-2), following [43]. (top: low-level noise; bottom: high-level noise)

https://doi.org/10.52202/079017-3744

Figure 16: (Robustness) Visualization of sample results from the plug-in method (**Ours**) and competing methods for **nonlinear deblurring**. We generate measurements with four types of noise—Gaussian, impulse, shot, and speckle noise—across two different noise levels: low (level-1) and high (level-2), following [43]. (top: low-level noise; bottom: high-level noise)

https://doi.org/10.52202/079017-3744

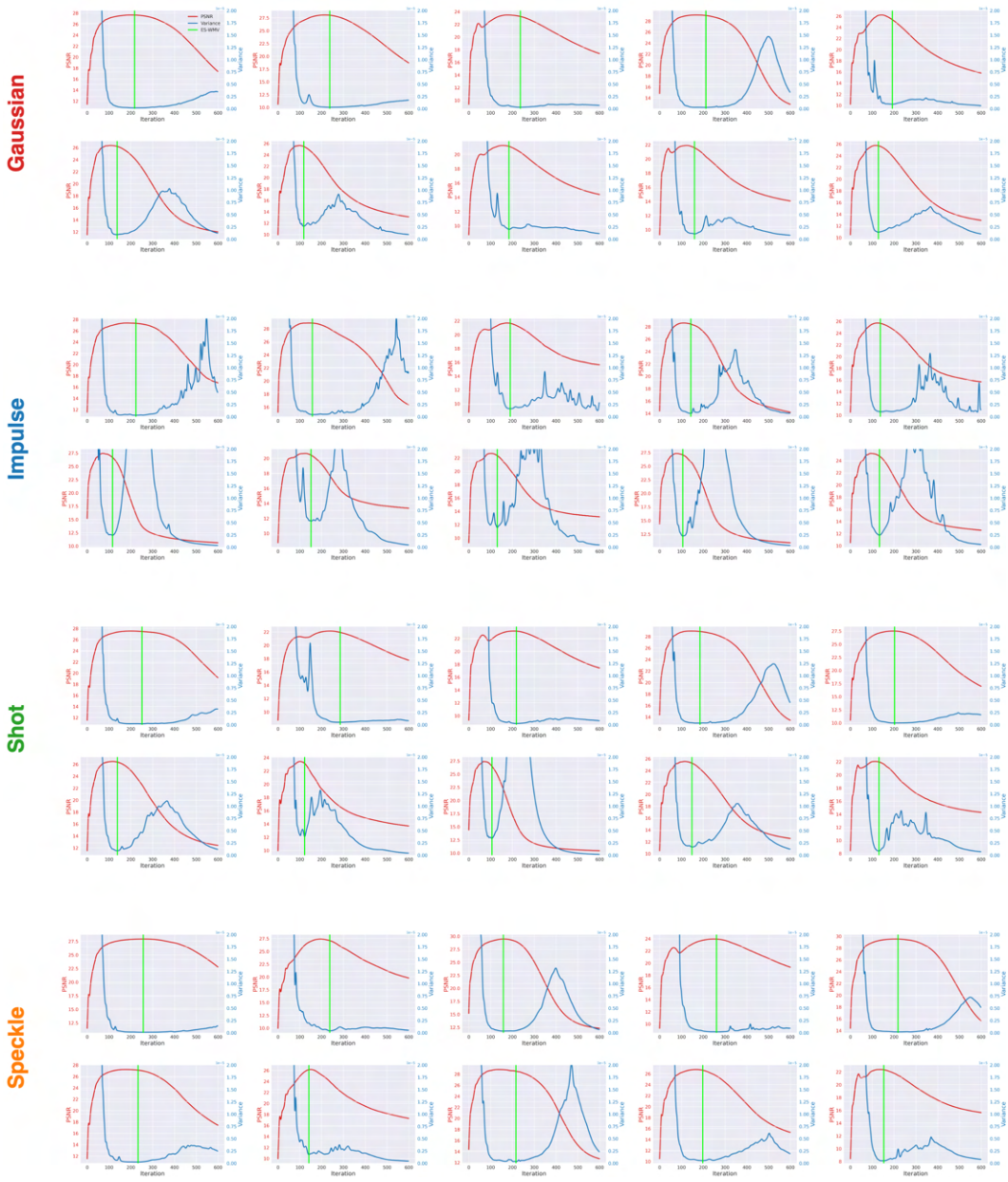

Figure 17: (Early stopping) Our DMPlgu with ES-WMV [35] for 4× **super-resolution** with different types and levels of noise. (top: low-level noise; bottom: high-level noise). Red curves are PSNR curves, and blue curves are VAR curves. The green bars indicate the detected ES point.

https://doi.org/10.52202/079017-3744

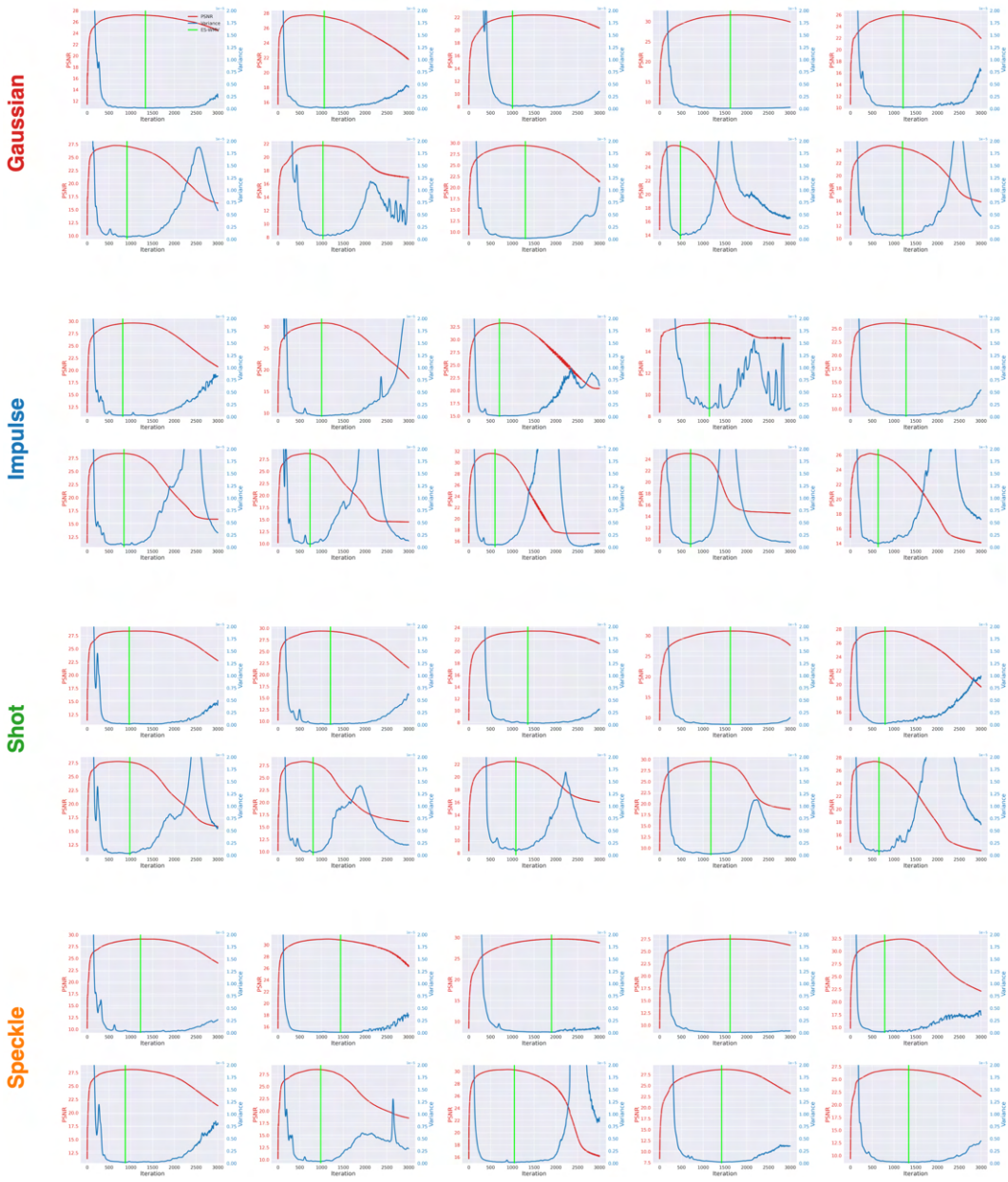

Figure 18: (Early stopping) Our DMPlug with ES-WMV [35] for **nonlinear deblurring** with different types and levels of noise. (top: low-level noise; bottom: high-level noise). Red curves are PSNR curves, and blue curves are VAR curves. The green bars indicate the detected ES point.

https://doi.org/10.52202/079017-3744

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly claim the scope and contributions of this paper in both abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of this work in Section 5.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

https://doi.org/10.52202/079017-3744

Justification: All the formulas are numbered and cross-referenced.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all the experiment details in Section 4, Appendix C and Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide all the essential code for this paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide all the experiment details in Section 4, Appendix C and Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: First, we conduct extensive experiments in this paper and report the mean results. We do not anticipate significant fluctuations in the results. Second, we are afraid that adding the statistical significance of the experiments will mess this paper up because our tables are already very dense, but we are willing to provide them if needed.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

https://doi.org/10.52202/079017-3744

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the experiment compute resources in Appendix D.2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper is strictly with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the societal impacts in Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: This paper only uses existing pretrained models for zero-shot tasks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide all the code and models we have used in Appendix D.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

 https://doi.org/10.52202/079017-3744

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new assets in this paper are well documented.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.