Denoising Diffusion Restoration Models

Bahjat Kawar
Department of Computer Science
Technion, Haifa, Israel
bahjat.kawar@cs.technion.ac.il

Michael Elad
Department of Computer Science
Technion, Haifa, Israel
elad@cs.technion.ac.il

Stefano Ermon
Department of Computer Science
Stanford, California, USA
ermon@cs.stanford.edu

Jiaming Song
NVIDIA
Santa Clara, California, USA
jiamings@nvidia.com

Abstract

Many interesting tasks in image restoration can be cast as linear inverse problems. A recent family of approaches for solving these problems uses stochastic algorithms that sample from the posterior distribution of natural images given the measurements. However, efficient solutions often require problem-specific supervised training to model the posterior, whereas unsupervised methods that are not problem-specific typically rely on inefficient iterative methods. This work addresses these issues by introducing Denoising Diffusion Restoration Models (DDRM), an efficient, unsupervised posterior sampling method. Motivated by variational inference, DDRM takes advantage of a pre-trained denoising diffusion generative model for solving any linear inverse problem. We demonstrate DDRM's versatility on several image datasets for super-resolution, deblurring, inpainting, and colorization under various amounts of measurement noise. DDRM outperforms the current leading unsupervised methods on the diverse ImageNet dataset in reconstruction quality, perceptual quality, and runtime, being $5\times$ faster than the nearest competitor. DDRM also generalizes well for natural images out of the distribution of the observed ImageNet training set.[1]

1 Introduction

Many problems in image processing, including super-resolution [31, 17], deblurring [28, 48], inpainting [55], colorization [29, 58], and compressive sensing [1], are instances of linear inverse problems, where the goal is to recover an image from potentially noisy measurements given through a known linear degradation model. For a specific degradation model, image restoration can be addressed through end-to-end *supervised* training of neural networks, using pairs of original and degraded images [14, 58, 41]. However, real-world applications such as medical imaging often require flexibility to cope with multiple, possibly infinite, degradation models [46]. Here, *unsupervised* approaches based on learned priors [36], where the degradation model is only known and used during inference, may be more desirable since they can adapt to the given problem without re-training [51]. By learning sound assumptions over the underlying structure of images (*e.g.*, priors, proximal operators or denoisers), unsupervised approaches can achieve effective restoration without training on specific degradation models [51, 40].

Under this unsupervised setting, priors based on deep neural networks have demonstrated impressive empirical results in various image restoration tasks [40, 50, 43, 38, 15]. To recover the signal,

[1] Project website: https://ddrm-ml.github.io/

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

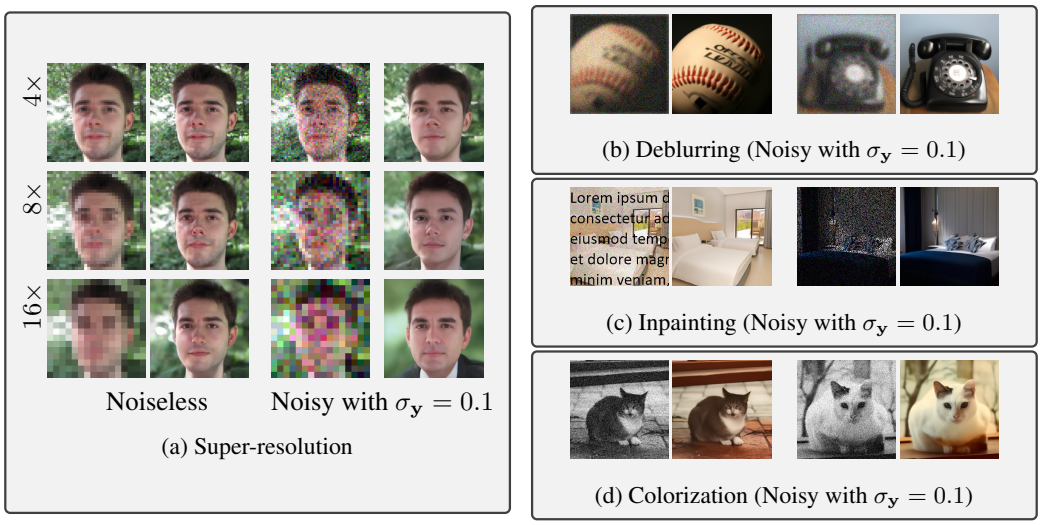


Figure 1: Pairs of measurements and recovered images with a 20-step DDRM on super-resolution, deblurring, inpainting, and colorization, with or without noise, and with unconditional generative models. The images are not accessed during training.

most existing methods obtain a prior-related term over the signal from a neural network (*e.g.*, the distribution of natural images), and a likelihood term from the degradation model. They combine the two terms to form a posterior over the signal, and the inverse problem can be posed as solving an optimization problem (*e.g.*, maximum a posteriori [8, 40]) or solving a sampling problem (*e.g.*, posterior sampling [2, 3, 25]). Then, these problems are often solved with iterative methods, such as gradient descent or Langevin dynamics, which may be demanding in computation and sensitive to hyperparameter tuning. An extreme example is found in [30] where a "fast" version of the algorithm uses $15,000$ neural function evaluations (NFEs).

Inspired by this unsupervised line of work, we introduce an efficient approach named Denoising Diffusion Restoration Models (DDRM), that can achieve competitive results in as low as 20 NFEs. DDRM is a denoising diffusion generative model [44, 19, 45] that gradually and stochastically denoises a sample to the desired output, conditioned on the measurements and the inverse problem. This way we introduce a variational inference objective for learning the posterior distribution of the inverse problem at hand. We then show its equivalence to the objective of an unconditional denoising diffusion generative model [19], which enables us to deploy such models in DDRM for various linear inverse problems (see Figure 2). To our best knowledge, DDRM is the first general sampling-based inverse problem solver that can efficiently produce a range of high-quality, diverse, yet valid solutions for general content images.

We demonstrate the empirical effectiveness of DDRM by comparing with various competitive methods based on learned priors, such as Deep Generative Prior (DGP) [38], SNIPS [25], and Regularization by Denoising (RED) [40]. On ImageNet examples, DDRM mostly outperforms the neural network baselines under noiseless super-resolution and deblurring measured in PSNR and KID [5], and is at least $50\times$ more efficient in terms of NFEs when it is second-best. Our advantage becomes even larger when measurement noise is involved, as noisy artifacts produced by iterative methods do not appear in our case. Over various real-world images, we further show DDRM results on super-resolution, deblurring, inpainting and colorization (see Figure 1). A DDRM trained on ImageNet also works on images that are out of its training set distribution (see Figure 6).

2 Background

Linear Inverse Problems. A general linear inverse problem is posed as

$$\mathbf{y} = \boldsymbol{H}\mathbf{x} + \mathbf{z}, \tag{1}$$

where we aim to recover the signal $\mathbf{x} \in \mathbb{R}^n$ from measurements $\mathbf{y} \in \mathbb{R}^m$, where $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ is a known linear degradation matrix, and $\mathbf{z} \sim \mathcal{N}(0, \sigma_{\mathbf{y}}^2 \boldsymbol{I})$ is an *i.i.d.* additive Gaussian noise with

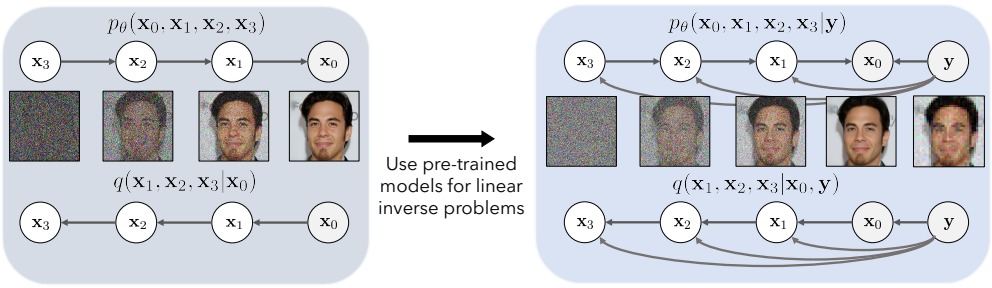


Figure 2: Illustration of our DDRM method for a specific inverse problem (super-resolution + denoising). We can use unsupervised DDPM models as a good solution to the DDRM objective.

known variance. The underlying structure of $\mathbf{x}$ can be represented via a generative model, denoted as $p_\theta(\mathbf{x})$. Given $\mathbf{y}$ and $\boldsymbol{H}$, a posterior over the signal can be posed as: $p_\theta(\mathbf{x}|\mathbf{y}) \propto p_\theta(\mathbf{x})p(\mathbf{y}|\mathbf{x})$, where the "likelihood" term $p(\mathbf{y}|\mathbf{x})$ is defined via Equation (1); such an approach leverages a learned prior $p_\theta(\mathbf{x})$, and we call it an "unsupervised" approach based on the terminology in [36], as the prior does not necessarily depend on the inverse problem. Recovering $\mathbf{x}$ can be done by sampling from this posterior [2], which may require many iterations to produce a good sample. Alternatively, one can also approximate this posterior by learning a model via amortized inference (*i.e.*, supervised learning); the model learns to predict $\mathbf{x}$ given $\mathbf{y}$, generated from $\mathbf{x}$ and a specific $\boldsymbol{H}$. While this can be more efficient than sampling-based methods, it may generalize poorly to inverse problems that have not been trained on.

Denoising Diffusion Probabilistic Models. Structures learned by generative models have been applied to various inverse problems and often outperform data-independent structural constraints such as sparsity [7]. These generative models learn a model distribution $p_\theta(\mathbf{x})$ that approximates a data distribution $q(\mathbf{x})$ from samples. In particular, diffusion models have demonstrated impressive unconditional generative modeling performance on images [13]. Diffusion models are generative models with a Markov chain structure $\mathbf{x}_T \to \mathbf{x}_{T-1} \to \ldots \to \mathbf{x}_1 \to \mathbf{x}_0$ (where $\mathbf{x}_t \in \mathbb{R}^n$), which has the following joint distribution:

$$p_\theta(\mathbf{x}_{0:T}) = p_\theta^{(T)}(\mathbf{x}_T) \prod_{t=0}^{T-1} p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}).$$

After drawing $\mathbf{x}_{0:T}$, only $\mathbf{x}_0$ is kept as the sample of the generative model. To train a diffusion model, a fixed, factorized variational inference distribution is introduced:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = q^{(T)}(\mathbf{x}_T|\mathbf{x}_0) \prod_{t=0}^{T-1} q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0),$$

which leads to an evidence lower bound (ELBO) on the maximum likelihood objective [44]. A special property of some diffusion models is that both $p_\theta^{(t)}$ and $q^{(t)}$ are chosen as conditional Gaussian distributions for all $t < T$, and that $q(\mathbf{x}_t|\mathbf{x}_0)$ is also a Gaussian with known mean and covariance, *i.e.*, $\mathbf{x}_t$ can be treated as $\mathbf{x}_0$ directly corrupted with Gaussian noise. Thus, the ELBO objective can be reduced into the following denoising autoencoder objective (please refer to [45] for derivations):

$$\sum_{t=1}^{T} \gamma_t \mathbb{E}_{(\mathbf{x}_0, \mathbf{x}_t) \sim q(\mathbf{x}_0) q(\mathbf{x}_t|\mathbf{x}_0)} \left[\left\| \mathbf{x}_0 - f_\theta^{(t)}(\mathbf{x}_t) \right\|_2^2 \right] \tag{2}$$

where $f_\theta^{(t)}$ is a θ-parameterized neural network that aims to recover a noiseless observation from a noisy $\mathbf{x}_t$, and $\gamma_{1:T}$ are a set of positive coefficients that depend on $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$.

3 Denoising Diffusion Restoration Models

Inverse problem solvers based on posterior sampling often face a dilemma: unsupervised approaches apply to general problems but are inefficient, whereas supervised ones are efficient but can only address specific problems.

To solve this dilemma, we introduce Denoising Diffusion Restoration Models (DDRM), an unsupervised solver for general linear inverse problems, capable of handling such tasks with or without noise in the measurements. DDRM is efficient and exhibits competitive performance compared to popular unsupervised solvers [40, 38, 25].

The key idea behind DDRM is to find an unsupervised solution that also suits supervised learning objectives. First, we describe the variational objective for DDRM over a specific inverse problem (Section 3.1). Next, we introduce specific forms of DDRM that are suitable for linear inverse problems and allow pre-trained unconditional and class-conditional diffusion models to be used directly (Sections 3.2, 3.3). Finally, we discuss practical algorithms that are compute and memory efficient (Sections 3.4, 3.5).

3.1 Variational Objective for DDRM

For any linear inverse problem, we define DDRM as a Markov chain $\mathbf{x}_T \to \mathbf{x}_{T-1} \to \ldots \to \mathbf{x}_1 \to \mathbf{x}_0$ conditioned on $\mathbf{y}$, where

$$p_\theta(\mathbf{x}_{0:T}|\mathbf{y}) = p_\theta^{(T)}(\mathbf{x}_T|\mathbf{y}) \prod_{t=0}^{T-1} p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y})$$

and $\mathbf{x}_0$ is the final diffusion output. In order to perform inference, we consider the following factorized variational distribution conditioned on $\mathbf{y}$:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0, \mathbf{y}) = q^{(T)}(\mathbf{x}_T|\mathbf{x}_0, \mathbf{y}) \prod_{t=0}^{T-1} q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}),$$

leading to an ELBO objective for diffusion models conditioned on $\mathbf{y}$ (details in Appendix A).

In the remainder of the section, we construct suitable variational problems given $\boldsymbol{H}$ and $\sigma_\mathbf{y}$ and connect them to unconditional diffusion generative models. To simplify notations, we will construct the variational distribution q such that $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2 \boldsymbol{I})$ for noise levels $0 = \sigma_0 < \sigma_1 < \sigma_2 < \ldots < \sigma_T$.[2] In Appendix B, we will show that this is equivalent to the distribution introduced in DDPM [19] and DDIM [45],[3] up to fixed linear transformations over $\mathbf{x}_t$.

This is a text...

3.2 A Diffusion Process for Image Restoration

Similar to SNIPS [25], we consider the singular value decomposition (SVD) of $\boldsymbol{H}$, and perform the diffusion in its spectral space. The idea behind this is to tie the noise present in the measurements $\mathbf{y}$ with the diffusion noise in $\mathbf{x}_{1:T}$, ensuring that the diffusion result $\mathbf{x}_0$ is faithful to the measurements. By using the SVD, we identify the data from $\mathbf{x}$ that is missing in $\mathbf{y}$, and synthesize it using a diffusion process. In conjunction, the noisy data in $\mathbf{y}$ undergoes a denoising process. For example, in inpainting with noise (*e.g.*, $\boldsymbol{H} = \text{diag}([1, \ldots, 1, 0, \ldots, 0])$, $\sigma_\mathbf{y} \geq 0$), the spectral space is simply the pixel space, so the model should generate the missing pixels and denoise the observed ones in $\mathbf{y}$. For a general linear $\boldsymbol{H}$, its SVD is given as

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top \tag{3}$$

where $\boldsymbol{U} \in \mathbb{R}^{m \times m}$, $\boldsymbol{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is a rectangular diagonal matrix containing the singular values of $\boldsymbol{H}$, ordered descendingly. As this is the case in most useful degradation models, we assume $m \leq n$, but our method would work for $m > n$ as well. We denote the singular values as $s_1 \geq s_2 \geq \ldots \geq s_m$, and define $s_i = 0$ for $i \in [m+1, n]$.

We use the shorthand notations for values in the spectral space: $\bar{\mathbf{x}}_t^{(i)}$ is the i-th index of the vector $\bar{\mathbf{x}}_t = \boldsymbol{V}^\top \mathbf{x}_t$, and $\bar{\mathbf{y}}^{(i)}$ is the i-th index of the vector $\bar{\mathbf{y}} = \boldsymbol{\Sigma}^\dagger \boldsymbol{U}^\top \mathbf{y}$ (where $\dagger$ denotes the Moore–Penrose pseudo-inverse). Because $\boldsymbol{V}$ is an orthogonal matrix, we can recover $\mathbf{x}_t$ from $\bar{\mathbf{x}}_t$ exactly by left

[2] This is called "Variance Exploding" in [47].

[3] This is called "Variance Preserving" in [47].

multiplying $\boldsymbol{V}$. For each index i in $\bar{\mathbf{x}}_t$, we define the variational distribution as:

$$q^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{x}_0,\mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}) & \text{if } s_i > 0 \\ \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_T^2) & \text{if } s_i = 0 \end{cases} \tag{4}$$

$$q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{t+1}}, \eta^2\sigma_t^2) & \text{if } s_i = 0 \\ \mathcal{N}(\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{\mathbf{y}}/s_i}, \eta^2\sigma_t^2) & \text{if } \sigma_t < \frac{\sigma_{\mathbf{y}}}{s_i} \\ \mathcal{N}((1-\eta_b)\bar{\mathbf{x}}_0^{(i)} + \eta_b\bar{\mathbf{y}}^{(i)}, \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2) & \text{if } \sigma_t \geq \frac{\sigma_{\mathbf{y}}}{s_i} \end{cases} \tag{5}$$

where $\eta \in (0, 1]$ is a hyperparameter controlling the variance of the transitions, and η and η_b may depend on $\sigma_t, s_i, \sigma_{\mathbf{y}}$. We further assume that $\sigma_T \geq \sigma_{\mathbf{y}}/s_i$ for all positive s_i.[4]

In the following statement, we show that this construction has the "Gaussian marginals" property similar to the inference distribution used in unconditional diffusion models [19].

Proposition 3.1. *The conditional distributions $q^{(t)}$ defined in Equations 4 and 5 satisfy the following:*

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2\boldsymbol{I}), \tag{6}$$

defined by marginalizing over $\mathbf{x}_{t'}$ (for all $t' > t$) and $\mathbf{y}$, where $q(\mathbf{y}|\mathbf{x}_0)$ is defined as in Equation (1) with $\mathbf{x} = \mathbf{x}_0$.

We place the proof in Appendix C. Intuitively, our construction considers different cases for each index of the spectral space. (*i*) If the corresponding singular value is zero, then $\mathbf{y}$ does not directly provide any information to that index, and the update is similar to regular unconditional generation. (*ii*) If the singular value is non-zero, then the updates consider the information provided by $\mathbf{y}$, which further depends on whether the measurements' noise level in the spectral space ($\sigma_{\mathbf{y}}/s_i$) is larger than the noise level in the diffusion model (σ_t) or not; the measurements in the spectral space $\bar{\mathbf{y}}^{(i)}$ are then scaled differently for these two cases in order to ensure Proposition 3.1 holds.

Now that we have defined $q^{(t)}$ as a series of Gaussian conditionals, we define our model distribution p_θ as a series of Gaussian conditionals as well. Similar to DDPM, we aim to obtain predictions of $\mathbf{x}_0$ at every step t; and to simplify notations, we use the symbol $\mathbf{x}_{\theta,t}$ to represent this prediction made by a model[5] $f_\theta(\mathbf{x}_{t+1}, t+1) : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ that takes in the sample $\mathbf{x}_{t+1}$ and the conditioned time step $(t+1)$. We also define $\bar{\mathbf{x}}_{\theta,t}^{(i)}$ as the i-th index of $\bar{\mathbf{x}}_{\theta,t} = \boldsymbol{V}^\top \mathbf{x}_{\theta,t}$.

We define DDRM with trainable parameters θ as follows:

$$p_\theta^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}) & \text{if } s_i > 0 \\ \mathcal{N}(0, \sigma_T^2) & \text{if } s_i = 0 \end{cases} \tag{7}$$

$$p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1},\mathbf{y}) = \begin{cases} \mathcal{N}(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_{t+1}}, \eta^2\sigma_t^2) & \text{if } s_i = 0 \\ \mathcal{N}(\bar{\mathbf{x}}_{\theta,t}^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_{\theta,t}^{(i)}}{\sigma_{\mathbf{y}}/s_i}, \eta^2\sigma_t^2) & \text{if } \sigma_t < \frac{\sigma_{\mathbf{y}}}{s_i} \\ \mathcal{N}((1-\eta_b)\bar{\mathbf{x}}_{\theta,t}^{(i)} + \eta_b\bar{\mathbf{y}}^{(i)}, \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2) & \text{if } \sigma_t \geq \frac{\sigma_{\mathbf{y}}}{s_i}. \end{cases} \tag{8}$$

Compared to $q^{(t)}$ in Equations (4) and (5), our definition of $p_\theta^{(t)}$ merely replaces $\bar{\mathbf{x}}_0^{(i)}$ (which we do not know at sampling) with $\bar{\mathbf{x}}_{\theta,t}^{(i)}$ (which depends on our predicted $\mathbf{x}_{\theta,t}$) when $t < T$, and replaces $\bar{\mathbf{x}}_0^{(i)}$ with 0 when $t = T$. It is possible to learn the variances [35] or consider alternative constructions where Proposition 3.1 holds; we leave these options as future work.

3.3 "Learning" Image Restoration Models

Once we have defined $p_\theta^{(t)}$ and $q^{(t)}$ by choosing $\sigma_{1:T}$, η and η_b, we can learn model parameters θ by maximizing the resulting ELBO objective (in Appendix A). However, this approach is not desirable

[4] This assumption is fair, as we can set a sufficiently large σ_T.

[5] Equivalently, the authors of [19] predict the noise values to subtract in order to recover $\mathbf{x}_{\theta,t}$.

(a) Inpainting results on cat images.

(b) Deblurring results ($\sigma_{\mathbf{y}} = 0.05$) on bedroom images.

Figure 3: DDRM results on bedroom and cat images, for inpainting and deblurring.

since we have to learn a different model for each inverse problem (given $\boldsymbol{H}$ and $\sigma_{\mathbf{y}}$), which is not flexible enough for arbitrary inverse problems. Fortunately, this does not have to be the case. In the following statement, we show that an optimal solution to DDPM / DDIM can also be an optimal solution to a DDRM problem, under reasonable assumptions used in prior work [19, 45].

Theorem 3.2. *Assume that the models $f_\theta^{(t)}$ and $f_\theta^{(t')}$ do not have weight sharing whenever $t \neq t'$, then when $\eta = 1$ and $\eta_b = \frac{2\sigma_t^2}{\sigma_t^2 + \sigma_{\mathbf{y}}^2 / s_i^2}$, the ELBO objective of DDRM (details in Appendix A) can be rewritten in the form of the DDPM / DDIM objective in Equation (2).*

We place the proof in Appendix C.

Even for different choices of η and η_b, the proof shows that the DDRM objective is a weighted sum-of-squares error in the spectral space, and thus pre-trained DDPM models are good approximations to the optimal solution. Therefore, we can apply the same diffusion model (*unconditioned* on the inverse problem) using the updates in Equation (7) and Equation (8) and only modify $\boldsymbol{H}$ and its SVD ($\boldsymbol{U}$, $\boldsymbol{\Sigma}$, $\boldsymbol{V}$) for various linear inverse problems.

3.4 Accelerated Algorithms for DDRM

Typical diffusion models are trained with many timesteps (*e.g.*, 1000) to achieve optimal unconditional image synthesis quality, but sampling speed is slow as many NFEs are required. Previous works [45, 13] have accelerated this process by "skipping" steps with appropriate update rules. This is also true for DDRM, since we can obtain the denoising autoencoder objective in Equation (2) for any choice of increasing $\sigma_{1:T}$. For a pre-trained diffusion model with T' timesteps, we can choose $\sigma_{1:T}$ to be a subset of the T' steps used in training.

3.5 Memory Efficient SVD

Our method, similar to SNIPS [25], utilizes the SVD of the degradation operator $\boldsymbol{H}$. This constitutes a memory consumption bottleneck in both algorithms as well as other methods such as Plug and Play (PnP) [51], as storing the matrix $\boldsymbol{V}$ has a space complexity of $\Theta(n^2)$ for signals of size n. By leveraging special properties of the matrices $\boldsymbol{H}$ used, we can reduce this complexity to $\Theta(n)$ for denoising, inpainting, super resolution, deblurring, and colorization (details in Appendix D).

4 Related Work

Various deep learning solutions have been suggested for solving inverse problems under different settings (see a detailed survey in [37]). We focus on the *unsupervised* setting, where we have access to a dataset of clean images at training time, but the degradation model is known only at inference time. This setup is inherently general to all linear inverse problems, a property desired in many real-world applications such as medical imaging [46, 20].

Table 1: Noiseless $4\times$ super-resolution and deblurring results on ImageNet 1K (256×256).

Method	$4\times$ super-resolution				Deblurring			
	PSNR↑	SSIM↑	KID↓	NFEs↓	PSNR↑	SSIM↑	KID↓	NFEs↓
Baseline	25.65	0.71	44.90	**0**	19.26	0.48	38.00	**0**
DGP	23.06	0.56	21.22	1500	22.70	0.52	27.60	1500
RED	26.08	0.73	53.55	100	26.16	0.76	21.21	500
SNIPS	17.58	0.22	35.17	1000	34.32	0.87	**0.49**	1000
DDRM	**26.55**	0.72	7.22	20	35.64	0.95	0.71	20
DDRM-CC	**26.55**	**0.74**	**6.56**	20	**35.65**	**0.96**	0.70	20

Almost all unsupervised inverse problem solvers utilize a trained neural network in an iterative scheme. PnP, RED, and their successors [51, 40, 32, 49] apply a denoiser as part of an iterative optimization algorithm such as steepest descent, fixed-point, or alternating direction method of multipliers (ADMM). OneNet [39] trained a network to directly learn the proximal operator of ADMM. A similar use of denoisers in different iterative algorithms is proposed in [34, 16, 30]. The authors of [43] leverages robust classifiers learned with additional class labels.

Another approach is to search the latent space of a generative model for a generated image that, when degraded, is as close as possible to the given measurements. Multiple such methods were suggested, mainly focusing on generative adversarial networks (GANs) [7, 11, 33]. While they exhibit impressive results on images of a specific class, most notably face images, these methods are not shown to be largely successful under a more diverse dataset such as ImageNet [12]. Deep Generative Prior (DGP) mitigates this issue by optimizing the latent input as well as the weights of the GAN's generator [38].

More recently, denoising diffusion models were used to solve inverse problems in both supervised (*i.e.*, degradation model is known during training) [42, 41, 13, 10, 54] and unsupervised settings [22, 26, 25, 21, 46, 47, 9]. Unlike previous approaches, most diffusion-based methods can successfully recover images from measurements with significant noise. However, these methods are very slow, often requiring hundreds or thousands of iterations, and are yet to be proven on diverse datasets. Our method, motivated by variational inference, obtains problem-specific, non-equilibrium update rules that lead to high-quality solutions in much fewer iterations.

ILVR [9] suggests a diffusion-based method that handles noiseless super-resolution, and can run in 250 steps. In Appendix H, we prove that when applied on the same underlying generative diffusion model, ILVR is a special case of DDRM. Therefore, ILVR can be further accelerated to run in 20 steps, but unlike DDRM, it provides no clear way of handling noise in the measurements. Similarly, the authors of [22] suggest a score-based solver for inverse problems that can converge in a small number of iterations, but does not handle noise in the measurements.

5 Experiments

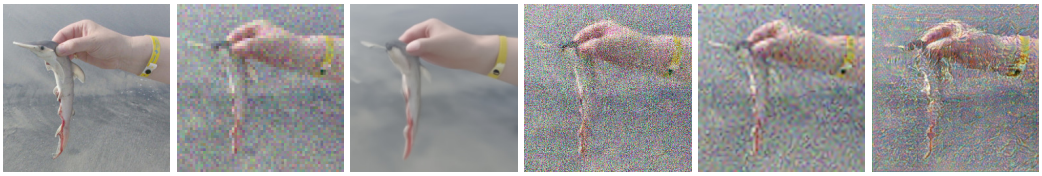
5.1 Experimental Setup

We demonstrate our algorithm's capabilities using the diffusion models from [19], which are trained on CelebA-HQ [23], LSUN bedrooms, and LSUN cats [56] (all 256×256 pixels). We test these models on images from FFHQ [24], and pictures from the internet of the considered LSUN category, respectively. In addition, we use the models from [13], trained on the training set of ImageNet 256×256 and 512×512, and tested on the corresponding validation set. Some of the ImageNet models require class information. For these models, we use the ground truth labels as input, and denote our algorithm as DDRM class conditional (DDRM-CC). In all experiments, we use $\eta = 0.85$, $\eta_b = 1$, and a uniformly-spaced timestep schedule based on the 1000-step pre-trained models (more details in Appendix E). The number of NFEs (timesteps) is reported in each experiment.

In each of the inverse problems we show, pixel values are in the range $[0, 1]$, and the degraded measurements are obtained as follows: (*i*) for super-resolution, we use a block averaging filter to downscale the images by a factor of 2, 4, or 8 in each axis; (*ii*) for deblurring, the images are blurred

Table 2: $4\times$ super resolution and deblurring results on ImageNet 1K (256×256). Input images have an additive noise of $\sigma_{\mathbf{y}} = 0.05$.

Method	$4\times$ super-resolution				Deblurring			
	PSNR↑	SSIM↑	KID↓	NFEs↓	PSNR↑	SSIM↑	KID↓	NFEs↓
Baseline	22.55	0.46	67.86	**0**	18.35	0.20	75.50	**0**
DGP	20.69	0.43	42.17	1500	21.20	0.45	34.02	1500
RED	22.90	0.49	43.45	100	14.69	0.08	121.82	500
SNIPS	16.30	0.14	67.77	1000	16.37	0.14	77.96	1000
DDRM	25.21	0.66	12.43	20	25.45	0.66	15.24	20
DDRM-CC	**25.22**	**0.67**	**10.82**	20	**25.46**	**0.67**	**13.49**	20



Original Low-res DDRM (20) SNIPS RED DGP

Figure 4: $4\times$ noisy super resolution comparison with $\sigma_{\mathbf{y}} = 0.05$.

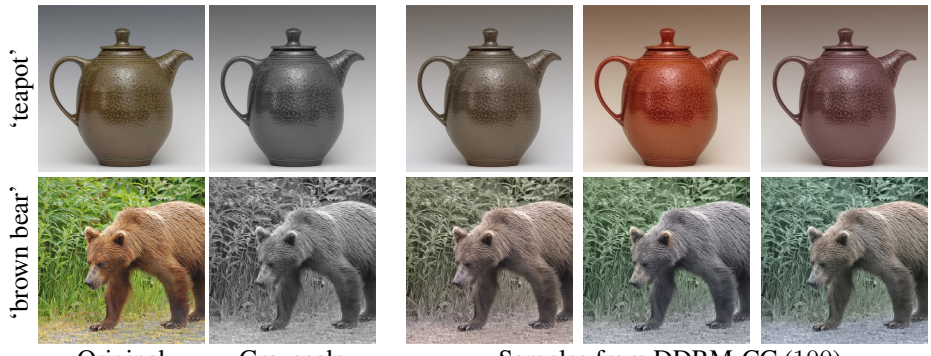
by a 9×9 uniform kernel, and singular values below a certain threshold are zeroed, making the problem more ill-posed. (*iii*) for colorization, the grayscale image is an average of the red, green, and blue channels of the original image; (*iv*) and for inpainting, we mask parts of the original image with text overlay or randomly drop 50% of the pixels. Additive white Gaussian noise can optionally be added to the measurements in all inverse problems. We additionally conduct experiments on bicubic super-resolution and deblurring with an anisotropic Gaussian kernel in Appendix I.

Our code is available at `https://github.com/bahjat-kawar/ddrm`.

5.2 Quantitative Experiments

In order to quantify DDRM's performance, we focus on the ImageNet dataset (256×256) for its diversity. For each experiment, we report the average peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [52] to measure faithfulness to the original image, and the kernel Inception distance (KID) [5], multiplied by 10^3, to measure the resulting image quality.

We compare DDRM (with 20 and 100 steps) with other unsupervised methods that work in reasonable time (requiring 1500 NFEs or less) and can operate on ImageNet. Namely, we compare with RED



Original Grayscale Samples from DDRM-CC (100)

Figure 5: 512×512 ImageNet colorization. DDRM-CC produces various samples for multiple runs on the same input.

Figure 6: Results on 256×256 USC-SIPI images using an ImageNet model. Blurred images have a noise of $\sigma_{\mathbf{y}} = 0.01$.

[40], DGP [38], and SNIPS [25]. The exact setup of each method is detailed in Appendix F. We used the same hyperparameters for noisy and noiseless versions of the same problem for DGP, RED, and SNIPS, as tuning them for each version would compromise their unsupervised nature. Nevertheless, the performance of baselines like RED with such a tuning does not surpass that of DDRM, as we show in Appendix F. In addition, we show upscaling by bicubic interpolation as a baseline for super-resolution, and the blurry image itself as a baseline for deblurring. OneNet [39] is not included in the comparisons as it is limited to images of size 64×64, and generalization to higher dimensions requires an improved network architecture.

We evaluate all methods on the problems of $4\times$ super-resolution and deblurring, on one validation set image from each of the 1000 ImageNet classes, following [38]. Table 1 shows that DDRM outperforms all baseline methods, in all metrics, and on both problems with only 20 steps. The only exception to this is that SNIPS achieves better KID than DDRM in noiseless deblurring, but it requires $50\times$ more NFEs to do so. Note that the runtime of all the tested methods is perfectly linear with NFEs, with negligible differences in time per iteration. DGP and DDRM-CC use ground-truth class labels for the test images to aid in the restoration process, and thus have an unfair advantage.

DDRM's appeal compared to previous methods becomes more substantial when significant noise is added to the measurements. Under this setting, DGP, RED, and SNIPS all fail to produce viable results, as evident in Table 2 and Figure 4. Since DDRM is fast, we also evaluate it on the entire ImageNet validation set in Appendix F.

5.3 Qualitative Experiments

DDRM produces high quality reconstructions across all the tested datasets and problems, as can be seen in Figures 1 and 3, and in Appendix I. As it is a posterior sampling algorithm, DDRM can produce multiple outputs for the same input, as demonstrated in Figure 5. Moreover, the unconditional ImageNet diffusion models can be used to solve inverse problems on out-of-distribution images with general content. In Figure 6, we show DDRM successfully restoring 256×256 images from USC-SIPI [53] that do not necessarily belong to any ImageNet class (more results in Appendix I).

6 Conclusions

We have introduced DDRM, a general sampling-based linear inverse problem solver based on unconditional/class-conditional diffusion generative models as learned priors. Motivated by variational inference, DDRM only requires a few number of NFEs (*e.g.*, 20) compared to other sampling-based baselines (*e.g.*, 1000 for SNIPS) and achieves scalability in multiple useful scenarios, including denoising, super-resolution, deblurring, inpainting, and colorization. We demonstrate the empirical

successes of DDRM on various problems and datasets, including general natural images outside the distribution of the observed training set. To our best knowledge, DDRM is the first unsupervised method that effectively and efficiently samples from the posterior distribution of inverse problems with significant noise, and can work on natural images with general content.

In terms of future work, apart from further optimizing the timestep and variance schedules, it would be interesting to investigate the following: (*i*) applying DDRM to non-linear inverse problems, (*ii*) addressing scenarios where the degradation operator is unknown, and (*iii*) self-supervised training techniques inspired by DDRM as well as ones used in supervised techniques [41] that further improve performance of unsupervised models for image restoration.

Acknowledgements

We thank Kristy Choi, Charlie Marx, and Avital Shafran for insightful discussions and feedback. This research was supported by NSF (#1651565, #1522054, #1733686), ONR (N00014-19-1-2145), AFOSR (FA9550-19-1-0024), ARO (W911NF-21-1-0125), Sloan Fellowship, Amazon AWS, Stanford Institute for Human-Centered Artificial Intelligence (HAI), Google Cloud, the Israel Science Foundation (ISF) under Grant 335/18, the Israeli Council For Higher Education - Planning & Budgeting Committee, and the Stephen A. Kreynes Fellowship.

References

[1] Richard G Baraniuk. Compressive sensing [lecture notes]. *IEEE signal processing magazine*, 24(4):118–121, 2007.

[2] Johnathan M Bardsley. Mcmc-based image reconstruction with uncertainty quantification. *SIAM Journal on Scientific Computing*, 34(3):A1316–A1332, 2012.

[3] Johnathan M Bardsley, Antti Solonen, Heikki Haario, and Marko Laine. Randomize-then-optimize: A method for sampling from posterior distributions in nonlinear inverse problems. *SIAM Journal on Scientific Computing*, 36(4):A1895–A1910, 2014.

[4] Christopher M Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[5] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.

[6] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6228–6237, 2018.

[7] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed sensing using generative models. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 537–546, 2017.

[8] Daniela Calvetti and Erkki Somersalo. Hypermodels in the bayesian imaging framework. *Inverse Problems*, 24(3):034013, 2008.

[9] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021.

[10] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. *arXiv preprint arXiv:2112.05146*, 2021.

[11] Giannis Daras, Joseph Dean, Ajil Jalal, and Alex Dimakis. Intermediate layer optimization for inverse problems using deep generative models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 2421–2432, 2021.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[13] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[14] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[15] Jinjin Gu, Yujun Shen, and Bolei Zhou. Image processing using multi-code gan prior. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3012–3021, 2020.

[16] Bichuan Guo, Yuxing Han, and Jiangtao Wen. Agem: Solving linear inverse problems via deep priors and sampling. *Advances in Neural Information Processing Systems*, 32, 2019.

[17] Muhammad Haris, Gregory Shakhnarovich, and Norimichi Ukita. Deep back-projection networks for super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1664–1673, 2018.

[18] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[20] Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alex Dimakis, and Jonathan Tamir. Robust compressed sensing mri with deep generative priors. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[21] Ajil Jalal, Sushrut Karmalkar, Alex Dimakis, and Eric Price. Instance-optimal compressed sensing via posterior sampling. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pages 4709–4720, 2021.

[22] Zahra Kadkhodaie and Eero Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. *Advances in Neural Information Processing Systems*, 34, 2021.

[23] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018.

[24] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[25] Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[26] Bahjat Kawar, Gregory Vaksman, and Michael Elad. Stochastic image denoising by sampling from the posterior distribution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1866–1875, October 2021.

[27] Diederik P Kingma and Max Welling. Auto-Encoding variational bayes. *arXiv preprint arXiv:1312.6114v10*, December 2013.

[28] Orest Kupyn, Tetiana Martyniuk, Junru Wu, and Zhangyang Wang. Deblurgan-v2: Deblurring (orders-of-magnitude) faster and better. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8878–8887, 2019.

[29] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Learning representations for automatic colorization. In *European conference on computer vision*, pages 577–593. Springer, 2016.

[30] Rémi Laumont, Valentin De Bortoli, Andrés Almansa, Julie Delon, Alain Durmus, and Marcelo Pereyra. Bayesian imaging using plug & play priors: When Langevin meets Tweedie. *arXiv preprint arXiv:2103.04715*, 2021.

[31] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4681–4690, 2017.

[32] Gary Mataev, Peyman Milanfar, and Michael Elad. DeepRED: deep image prior powered by RED. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[33] Sachit Menon, Alex Damian, McCourt Hu, Nikhil Ravi, and Cynthia Rudin. Pulse: Self-supervised photo upsampling via latent space exploration of generative models. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[34] Chris Metzler, Ali Mousavi, and Richard Baraniuk. Learned d-amp: Principled neural network based compressive image recovery. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[35] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

[36] Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

[37] Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

[38] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. In *European Conference on Computer Vision (ECCV)*, 2020.

[39] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.

[40] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

[41] Chitwan Saharia, William Chan, Huiwen Chang, Chris A Lee, Jonathan Ho, Tim Salimans, David J Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. *arXiv preprint arXiv:2111.05826*, 2021.

[42] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *arXiv preprint arXiv:2104.07636*, 2021.

[43] Shibani Santurkar, Dimitris Tsipras, Brandon Tran, Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Image synthesis with a single (robust) classifier. *arXiv preprint arXiv:1906.09453*, 2019.

[44] Jascha Sohl-Dickstein, Eric A Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. *arXiv preprint arXiv:1503.03585*, March 2015.

[45] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, April 2021.

[46] Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. *arXiv preprint arXiv:2111.08005*, 2021.

[47] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[48] Maitreya Suin, Kuldeep Purohit, and AN Rajagopalan. Spatially-attentive patch-hierarchical network for adaptive motion deblurring. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3606–3615, 2020.

[49] Yu Sun, Brendt Wohlberg, and Ulugbek S Kamilov. An online plug-and-play algorithm for regularized image reconstruction. *IEEE Transactions on Computational Imaging*, 5(3):395–408, 2019.

[50] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

[51] Singanallur V Venkatakrishnan, Charles A Bouman, and Brendt Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013.

[52] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[53] Allan G Weber. The USC-SIPI image database version 5. *USC-SIPI Report*, 315(1), 1997.

[54] Jay Whang, Mauricio Delbracio, Hossein Talebi, Chitwan Saharia, Alexandros G Dimakis, and Peyman Milanfar. Deblurring via stochastic refinement. *arXiv preprint arXiv:2112.02475*, 2021.

[55] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5485–5493, 2017.

[56] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[57] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[58] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666. Springer, 2016.

A Details of the DDRM ELBO objective

DDRM is a Markov chain conditioned on $\mathbf{y}$, which would lead to the following ELBO objective [45]:

$$\mathbb{E}_{\mathbf{x}_0\sim q(\mathbf{x}_0),\mathbf{y}\sim q(\mathbf{y}|\mathbf{x}_0)}[\log p_\theta(\mathbf{x}_0|\mathbf{y})] \tag{9}$$

$$\geq -\mathbb{E}\left[\sum_{t=1}^{T-1} D_{\mathrm{KL}}(q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})\|p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{y}))\right] + \mathbb{E}\left[\log p_\theta^{(0)}(\mathbf{x}_0|\mathbf{x}_1,\mathbf{y})\right]$$

$$-\mathbb{E}[D_{\mathrm{KL}}(q^{(T)}(\mathbf{x}_T|\mathbf{x}_0,\mathbf{y})\|p_\theta^{(T)}(\mathbf{x}_T|\mathbf{y}))] \tag{10}$$

where $q(\mathbf{x}_0)$ is the data distribution, $q(\mathbf{y}|\mathbf{x}_0)$ follows Equation 1 in the main paper, the expectation on the right hand side is given by sampling $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, $\mathbf{y} \sim q(\mathbf{y}|\mathbf{x}_0)$, $\mathbf{x}_T \sim q^{(T)}(\mathbf{x}_T|\mathbf{x}_0,\mathbf{y})$, and $\mathbf{x}_t \sim q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})$ for $t \in [1, T-1]$.

B Equivalence between "Variance Preserving" and "Variance Exploding" Diffusion Models

In our main paper, we describe our methods based on the "Variance Exploding" hyperparameters σ_t, where $\sigma_t \in [0,\infty)$ and

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2\boldsymbol{I}). \tag{11}$$

In DDIM [45], the hyperparameters are "Variance Preserving" ones α_t, where $\alpha_t \in (0,1]$ and

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\sqrt{\alpha_t}\mathbf{x}_0, (1-\alpha_t)\boldsymbol{I}). \tag{12}$$

We use the colored notation $\mathbf{x}_t$ to emphasize that this is different from $\mathbf{x}_t$ (an exception is $\mathbf{x}_0 = \mathbf{x}_0$). Using the reparametrization trick, we have that:

$$\mathbf{x}_t = \mathbf{x}_0 + \sigma_t\epsilon \tag{13}$$

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\epsilon \tag{14}$$

where $\epsilon \sim \mathcal{N}(0,\boldsymbol{I})$. We can divide by $\sqrt{1+\sigma_t^2}$ in both sides of Equation 13:

$$\frac{\mathbf{x}_t}{\sqrt{1+\sigma_t^2}} = \frac{\mathbf{x}_0}{\sqrt{1+\sigma_t^2}} + \frac{\sigma_t}{\sqrt{1+\sigma_t^2}}\epsilon. \tag{15}$$

Let $\alpha_t = 1/(1+\sigma_t^2)$, and let $\mathbf{x}_t = \mathbf{x}_t/\sqrt{1+\sigma_t^2}$; then from Equation 15 we have that

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1-\alpha_t}\epsilon, \tag{16}$$

which is equivalent to the "Variance Preserving" case. Therefore, we can use "Variance Preserving" models, such as DDPM, directly in our DDRM updates, even though the latter uses the "Variance Exploding" parametrization:

1. From $\mathbf{x}_t$, obtain predictions ϵ and $\mathbf{x}_t = \mathbf{x}_t\sqrt{1+\sigma_t^2}$.
2. From $\mathbf{x}_t$ and ϵ, apply DDRM updates to get $\mathbf{x}_{t-1}$.
3. From $\mathbf{x}_{t-1}$, get $\mathbf{x}_{t-1} = \mathbf{x}_{t-1}/\sqrt{1+\sigma_{t-1}^2}$.

Note that although the inference algorithms are shown to be equivalent, the choice between "Variance Preserving" and "Variance Exploding" may affect the training of diffusion networks.

C Proofs

Proposition 3.1. *The conditional distributions $q^{(t)}$ defined in Equations 4 and 5 satisfy the following:*

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_0, \sigma_t^2\boldsymbol{I}), \tag{6}$$

defined by marginalizing over $\mathbf{x}_{t'}$ (for all $t' > t$) and $\mathbf{y}$, where $q(\mathbf{y}|\mathbf{x}_0)$ is defined as in Equation (1) with $\mathbf{x} = \mathbf{x}_0$.

Proof. The proof uses a basic property of Gaussian marginals (see [4] for the complete version).

1. If $p(z_1|z_0) = \mathcal{N}(z_0, V_1)$, $p(z_2|z_1) = \mathcal{N}(\alpha z_1, V_2)$, then $p(z_2|z_0) = \mathcal{N}(\alpha z_0, \alpha^2 V_1 + V_2)$.
2. If $p(z_1) = \mathcal{N}(\mu_1, V_1)$ and $p(z_2) = \mathcal{N}(\mu_2, V_2)$, then $p(z_1 + z_2) = \mathcal{N}(\mu_1 + \mu_2, V_1 + V_2)$.

First, we note that $q(\mathbf{y}|\mathbf{x}_0)$ is defined from Equation 1 in the main paper, and thus for all i:

$$q(\bar{\mathbf{y}}^{(i)}|\mathbf{x}_0) = \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_{\mathbf{y}}^2/s_i^2). \tag{17}$$

Case I For $\mathbf{x}_T$, it is obvious when $s_i = 0$. When $s_i > 0$, we have Equation 17 and that:

$$q^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{x}_0, \mathbf{y}) = \mathcal{N}(\bar{\mathbf{y}}^{(i)}, \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}), \tag{18}$$

and thus

$$q^{(T)}(\bar{\mathbf{x}}_T^{(i)}|\mathbf{x}_0) = \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_{\mathbf{y}}^2/s_i^2 + \sigma_T^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}) = \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_T^2).$$

Case II For any $t < T$ and i such that $s_i > 0$ and $\sigma_t > \sigma_{\mathbf{y}}/s_i$, we have Equation 17 and that:

$$q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1}, \mathbf{x}_0, \mathbf{y}) = \mathcal{N}\left((1-\eta_b)\bar{\mathbf{x}}_0^{(i)} + \eta_b \bar{\mathbf{y}}^{(i)}, \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2\right), \tag{19}$$

and thus we can safely remove the dependence on $\mathbf{x}_{t+1}$ via marginalization. $q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_0)$ is a Gaussian with the mean being $(1-\eta_b)\bar{\mathbf{x}}_0^{(i)} + \eta_b \bar{\mathbf{x}}_0^{(i)} = \bar{\mathbf{x}}_0^{(i)}$ and variance being

$$\sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2 + \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2 = \sigma_t^2,$$

where we note that $\bar{\mathbf{y}}^{(i)}$ has a standard deviation of $\sigma_{\mathbf{y}}/s_i$.

Case III For any $t < T$ and i such that $s_i > 0$ and $\sigma_t < \sigma_{\mathbf{y}}/s_i$, we have Equation 17, so $(\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_0^{(i)})/(\sigma_{\mathbf{y}}/s_i)$ is distributed as a standard Gaussian. Moreover, similar to **Case II**, $q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_0)$ is a Gaussian with its mean being

$$\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{y}}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{\mathbf{y}}/s_i}$$

and its variance being $\eta^2\sigma_t^2$, so $q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_0)$ is a Gaussian with a mean of $\bar{\mathbf{x}}_0^{(i)}$ and a variance of

$$(1-\eta^2)\sigma_t^2 + \eta^2\sigma_t^2 = \sigma_t^2.$$

Case IV For any $t \leq T$ and i such that $s_i = 0$ (where there is no dependence on $\mathbf{y}$), we apply mathematical induction. The base case ($t = T$) is true, as we have shown earlier in **Case I**. In the step case ($t < T$), we have that $q^{(t+1)}(\bar{\mathbf{x}}_{t+1}^{(i)}|\mathbf{x}_0) = \mathcal{N}(\bar{\mathbf{x}}_0^{(i)}, \sigma_{t+1}^2)$. Similar to **Case II**, $q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_0)$ is a Gaussian with its mean being

$$\bar{\mathbf{x}}_0^{(i)} + \sqrt{1-\eta^2}\sigma_t \frac{\bar{\mathbf{x}}_{t+1}^{(i)} - \bar{\mathbf{x}}_0^{(i)}}{\sigma_{t+1}}$$

and variance being $\eta^2\sigma_t^2$, which does not depend on $\mathbf{y}$. Therefore, $q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_0)$ is also Gaussian, with a mean of $\bar{\mathbf{x}}_0^{(i)}$ and a variance of

$$(1-\eta^2)\sigma_t^2 + \eta^2\sigma_t^2 = \sigma_t^2.$$

Hence, the proof is completed via the four cases. □

Theorem 3.2. *Assume that the models $f_\theta^{(t)}$ and $f_\theta^{(t')}$ do not have weight sharing whenever $t \neq t'$, then when $\eta = 1$ and $\eta_b = \frac{2\sigma_t^2}{\sigma_t^2 + \sigma_{\mathbf{y}}^2/s_i^2}$, the ELBO objective of DDRM (details in Appendix A) can be rewritten in the form of the DDPM / DDIM objective in Equation (2).*

Proof. As there is no parameter sharing between models at different time steps t, let us focus on any particular time step t and rewrite the corresponding objective as a denoising autoencoder objective.

Case I For $t > 0$, the only term in Equation 10 that is related to $f_\theta^{(t)}$ (which is used to make the prediction $\mathbf{x}_{\theta,t}$) is:

$$\begin{aligned}
&D_{\mathrm{KL}}(q^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})\|p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1},\mathbf{y})) \\
=\ &D_{\mathrm{KL}}(q^{(t)}(\bar{\mathbf{x}}_t|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})\|p_\theta^{(t)}(\bar{\mathbf{x}}_t|\mathbf{x}_{t+1},\mathbf{y})) \\
=\ &\sum_{i=1}^{n} D_{\mathrm{KL}}(q^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})\|p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})),
\end{aligned} \tag{20}$$

where the first equality is from the orthogonality of $\boldsymbol{V}^\top$ and the second equality is from the fact that both $q^{(t)}$ and $p_\theta^{(t)}$ over the spectral space are Gaussians with identical diagonal covariance matrices (so the KL divergence can factorize).

Here, we will use a simple property of the KL divergence between univariate Gaussians [27]:

If $p = \mathcal{N}(\mu_1, V_1)$, $q = \mathcal{N}(\mu_2, V_2)$, then

$$D_{\mathrm{KL}}(p\|q) = \frac{1}{2}\log\frac{V_2}{V_1} + \frac{V_1 + (\mu_1-\mu_2)^2}{2V_2} - \frac{1}{2}.$$

Since we constructed $p_\theta^{(t)}$ and $q^{(t)}$ to have the same variance, Equation 20 is a total squared error with weights for each dimension of $\bar{\mathbf{x}}_t$ (the spectral space), so the DDPM objective (which is a total squared error objective in the original space) is still a good approximation. In order to transform it into a denoising autoencoder objective (equivalent to DDPM), the weights have to be equal. Next, we will show that our construction of $\eta = 1$ and $\eta_b = 2\sigma_t^2/(\sigma_t^2 + \sigma_{\mathbf{y}}^2/s_i^2)$ satisfies this.

All the indices i will fall into one of the three cases: $s_i = 0$, $\sigma_t < \sigma_{\mathbf{y}}/s_i$, or $\sigma_t > \sigma_{\mathbf{y}}/s_i$.

- For $s_i = 0$, the KL divergence is $\frac{(\bar{\mathbf{x}}_{\theta,t}^{(i)} - \bar{\mathbf{x}}_0^{(i)})^2}{2\sigma_t^2}$, where we recall $\bar{\mathbf{x}}_{\theta,t} = \boldsymbol{V}^\top f_\theta^{(t)}(\mathbf{x}_{t+1})$.
- For $\sigma_t < \frac{\sigma_{\mathbf{y}}}{s_i}$, the KL divergence is also $\frac{(\bar{\mathbf{x}}_{\theta,t}^{(i)} - \bar{\mathbf{x}}_0^{(i)})^2}{2\sigma_t^2}$.
- For $\sigma_t \geq \frac{\sigma_{\mathbf{y}}}{s_i}$, we have defined η_b as a solution to the following quadratic equation (the other solution is 0, which is irrelevant to our case since it does not make use of information from $\mathbf{y}$):

$$(\sigma_t^2 + \frac{\sigma_{\mathbf{y}}^2}{s_i^2})\eta_b^2 - 2\sigma_t^2\eta_b = 0; \tag{21}$$

reorganizing terms, we have that:

$$\begin{aligned}
(\sigma_t^2 + \frac{\sigma_{\mathbf{y}}^2}{s_i^2})\eta_b^2 - 2\sigma_t^2\eta_b + \sigma_t^2 &= \sigma_t^2 \\
\sigma_t^2(1-\eta_b)^2 = \sigma_t^2\eta_b^2 - 2\sigma_t^2\eta_b + \sigma_t^2 &= \sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2 \\
\frac{(1-\eta_b)^2}{\sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2} &= \frac{1}{\sigma_t^2},
\end{aligned} \tag{22}$$

So the KL divergence is

$$\frac{(1-\eta_b)^2}{2(\sigma_t^2 - \frac{\sigma_{\mathbf{y}}^2}{s_i^2}\eta_b^2)}(\bar{\mathbf{x}}_{\theta,t}^{(i)} - \bar{\mathbf{x}}_0^{(i)})^2 = \frac{(\bar{\mathbf{x}}_{\theta,t}^{(i)} - \bar{\mathbf{x}}_0^{(i)})^2}{2\sigma_t^2}.$$

Therefore, regardless of how the cases are distributed among indices, we will always have that:

$$D_{\mathrm{KL}}(q^{(t)}(\bar{\mathbf{x}}_t|\mathbf{x}_{t+1},\mathbf{x}_0,\mathbf{y})\|p_\theta^{(t)}(\bar{\mathbf{x}}_t|\mathbf{x}_{t+1},\mathbf{y})) = \sum_{i=1}^{n^2}\frac{(\bar{\mathbf{x}}_{\theta,t}^{(i)}-\bar{\mathbf{x}}_0^{(i)})^2}{2\sigma_t^2} = \frac{\|\bar{\mathbf{x}}_{\theta,t}-\bar{\mathbf{x}}_0\|_2^2}{2\sigma_t^2} = \frac{\left\|f_\theta^{(t)}(\mathbf{x}_{t+1})-\mathbf{x}_0\right\|_2^2}{2\sigma_t^2}.$$

Case II For $t=0$, we will only have two cases ($s_i=0$ or $\sigma_t<\frac{\sigma_{\mathbf{y}}}{s_i}$), and thus, similar to **Case I**,

$$\log p_\theta^{(0)}(\bar{\mathbf{x}}_0|\mathbf{x}_1,\mathbf{y}) = \sum_{i=1}^{n^2}\log p_\theta^{(0)}(\bar{\mathbf{x}}_0^{(i)}|\mathbf{x}_1,\mathbf{y}) \propto \sum_{i=1}^{n^2}(\bar{\mathbf{x}}_{\theta,0}^{(i)}-\bar{\mathbf{x}}_0^{(i)})^2 = \|\bar{\mathbf{x}}_{\theta,0}-\bar{\mathbf{x}}_0\|_2^2 = \left\|f_\theta^{(0)}(\mathbf{x}_1)-\mathbf{x}_0\right\|_2^2,$$

as long as we have a constant variance for $p_\theta^{(0)}$. Thus, every individual term in Equation 10 can be written as a denoising autoencoder objective, completing the proof. □

D Memory Efficient SVD

Here we explain how we obtained the singular value decomposition (SVD) for different degradation models efficiently.

D.1 Denoising

In denoising, the corrupted image is the original image with additive white Gaussian noise. Therefore, $\boldsymbol{H}=\boldsymbol{I}$ and all the SVD elements of $\boldsymbol{H}$ are simply the identity matrix $\boldsymbol{I}$, which in turns makes their multiplication by different vectors trivial.

D.2 Inpainting

In inpainting, $\boldsymbol{H}$ retains a known subset of size k of the image's pixels. This is equivalent to permuting the pixels such that the retained one are placed at the top, then keeping the first k entries. Therefore,

$$\boldsymbol{H}=\boldsymbol{I}\boldsymbol{\Sigma}\boldsymbol{P}, \tag{23}$$

where $\boldsymbol{P}$ is the appropriate permutation matrix, $\boldsymbol{\Sigma}$ is a rectangular diagonal matrix of size $k\times n$ with ones in its main diagonal, and $\boldsymbol{I}$ is the identity matrix. Since permutation matrices are orthogonal, Equation 23 is the SVD of $\boldsymbol{H}$.

We can multiply a given vector by $\boldsymbol{P}$ and $\boldsymbol{P}^T$ by storing the permutation itself rather than the matrix. $\boldsymbol{\Sigma}$ can multiply a vector by simply slicing it. Therefore, by storing the appropriate permutation and the number k, we can apply each element of the SVD with $\Theta(n)$ space complexity.

D.3 Super Resolution

For super resolution, we assume that the original image of size $d\times d$ (*i.e.* $n=3d^2$) is downscaled using a block averaging filter by r in each dimension, such that d is divisible by r. In this scenario, each pixel in the output image is the average of an $r\times r$ patch in the input image, and each such patch affects exactly one output pixel. Therefore, any output pixel is given by $(\boldsymbol{H}\mathbf{x})_i=\boldsymbol{k}^T\boldsymbol{p}_i$, where $\boldsymbol{k}$ is a vector of size r^2 with $\frac{1}{r^2}$ in each entry, and $\boldsymbol{p}_i$ is the vectorized i-th $r\times r$ patch. More formally, if $\boldsymbol{P}_1$ is a permutation matrix that reorders a vectorized image into patches, then

$$\boldsymbol{H}=\left(\boldsymbol{I}\otimes\boldsymbol{k}^T\right)\boldsymbol{P}_1,$$

where $\otimes$ is the Kronecker product, and $\boldsymbol{I}$ is the identity matrix of size $\frac{d}{r}\times\frac{d}{r}$. In order to obtain the SVD of $\boldsymbol{H}$, we calculate the SVD of $\boldsymbol{k}^T$:

$$\boldsymbol{k}^T=\boldsymbol{U}_{\boldsymbol{k}}\boldsymbol{\Sigma}_{\boldsymbol{k}}\boldsymbol{V}_{\boldsymbol{k}}^T.$$

Using properties of the Kronecker product, we observe

$$\begin{aligned}\boldsymbol{H}&=\left(\boldsymbol{I}\otimes\boldsymbol{k}^T\right)\boldsymbol{P}_1=\left((\boldsymbol{I}\boldsymbol{I}\boldsymbol{I})\otimes\left(\boldsymbol{U}_{\boldsymbol{k}}\boldsymbol{\Sigma}_{\boldsymbol{k}}\boldsymbol{V}_{\boldsymbol{k}}^T\right)\right)\boldsymbol{P}_1\\&=\left(\boldsymbol{I}\otimes\boldsymbol{U}_{\boldsymbol{k}}\right)\left(\boldsymbol{I}\otimes\boldsymbol{\Sigma}_{\boldsymbol{k}}\right)\left(\boldsymbol{I}\otimes\boldsymbol{V}_{\boldsymbol{k}}^T\right)\boldsymbol{P}_1.\end{aligned} \tag{24}$$

The Kronecker product of two orthogonal matrices is an orthogonal matrix. Therefore, $\boldsymbol{I} \otimes \boldsymbol{U_k}$ and $\boldsymbol{I} \otimes \boldsymbol{V_k}^T$ are orthogonal. Observe that the matrix $\boldsymbol{I} \otimes \boldsymbol{\Sigma_k}$ has one non-zero value ($\frac{1}{r^2}$) in each row. By applying a simple permutation on its columns, these values can be reordered to be on the main diagonal. We denote the appropriate permutation matrix by $\boldsymbol{P}_2$, and obtain

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{25}$$

where $\boldsymbol{U} = \boldsymbol{I} \otimes \boldsymbol{U_k}$ is orthogonal, $\boldsymbol{\Sigma} = (\boldsymbol{I} \otimes \boldsymbol{\Sigma_k})\, \boldsymbol{P}_2^T$ is a rectangular diagonal matrix with non-negative entries, and $\boldsymbol{V}^T = \boldsymbol{P}_2 \left(\boldsymbol{I} \otimes \boldsymbol{V_k}^T\right) \boldsymbol{P}_1$ is orthogonal. As such, Equation 25 is the SVD of $\boldsymbol{H}$. By storing the permutations and the SVD elements of $\boldsymbol{k}^T$, we can simulate each element of the SVD of $\boldsymbol{H}$ with $\Theta(n)$ space complexity, without directly calculating the Kronecker products with $\boldsymbol{I}$.

D.4 Colorization

The grayscale image is obtained by averaging the red, green, and blue channels of each pixel. This means that every output pixel is given by $(\boldsymbol{H}\mathbf{x})_i = \boldsymbol{k}^T \boldsymbol{p}_i$, where $\boldsymbol{k}^T = \begin{pmatrix}\frac{1}{3} & \frac{1}{3} & \frac{1}{3}\end{pmatrix}$ and $\boldsymbol{p}_i$ is the 3-valued i-th pixel of the original color image. The SVD of $\boldsymbol{H}$ is obtained exactly the same as in the super resolution case, with separate pixels replacing separate patches.

D.5 Deblurring

We focus on *separable blurring*, where the 2D blurring kernel is $\boldsymbol{K} = \boldsymbol{r}\boldsymbol{c}^T$, which means $\boldsymbol{c}$ is applied on the columns of the image, and $\boldsymbol{r}^T$ is applied on its rows. The blurred image can be obtained by $\boldsymbol{B} = \boldsymbol{A}_c \boldsymbol{X} \boldsymbol{A}_r^T$, where $\boldsymbol{A}_c$ and $\boldsymbol{A}_r$ apply a 1D convolution with kernels $\boldsymbol{c}$ and $\boldsymbol{r}$, respectively. Alternatively, $\boldsymbol{b} = \boldsymbol{H}\boldsymbol{x}$, where $\boldsymbol{x}$ is the vectorized image $\boldsymbol{X}$, $\boldsymbol{b}$ is the vectorized blurred image $\boldsymbol{B}$, and $\boldsymbol{H}$ is the matrix applying the 2D convolution $\boldsymbol{K}$. It can be shown that $\boldsymbol{H} = \boldsymbol{A}_r \otimes \boldsymbol{A}_c$, where $\otimes$ is the Kronecker product. In order to calculate the SVD of $\boldsymbol{H}$, we calculate the SVD of $\boldsymbol{A}_r$ and $\boldsymbol{A}_c$:

$$\boldsymbol{A}_r = \boldsymbol{U}_r \boldsymbol{\Sigma}_r \boldsymbol{V}_r^T, \quad \boldsymbol{A}_c = \boldsymbol{U}_c \boldsymbol{\Sigma}_c \boldsymbol{V}_c^T.$$

Using the properties of the Kronecker product, we observe

$$\begin{aligned}\boldsymbol{H} = \boldsymbol{A}_r \otimes \boldsymbol{A}_c &= \left(\boldsymbol{U}_r \boldsymbol{\Sigma}_r \boldsymbol{V}_r^T\right) \otimes \left(\boldsymbol{U}_c \boldsymbol{\Sigma}_c \boldsymbol{V}_c^T\right) \\ &= \left(\boldsymbol{U}_r \otimes \boldsymbol{U}_c\right) \left(\boldsymbol{\Sigma}_r \otimes \boldsymbol{\Sigma}_c\right) \left(\boldsymbol{V}_r \otimes \boldsymbol{V}_c\right)^T.\end{aligned} \tag{26}$$

The Kronecker product preserves orthogonality. Therefore, Equation 26 is a valid SVD of $\boldsymbol{H}$, with the exception of the singular values not being on the main diagonal, and not being sorted descendingly. We reorder the columns so that the singular values are on the main diagonal and denote the corresponding permutation matrix by $\boldsymbol{P}_1$. We also sort the values descendingly and denote the sorting permutation matrix by $\boldsymbol{P}_2$, and obtain the following SVD:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T, \tag{27}$$

where $\boldsymbol{U} = \left(\boldsymbol{U}_r \otimes \boldsymbol{U}_c\right) \boldsymbol{P}_2^T$, $\boldsymbol{\Sigma} = \boldsymbol{P}_2 \left(\boldsymbol{\Sigma}_r \otimes \boldsymbol{\Sigma}_c\right) \boldsymbol{P}_1^T \boldsymbol{P}_2^T$, and $\boldsymbol{V}^T = \boldsymbol{P}_2 \boldsymbol{P}_1 \left(\boldsymbol{V}_r \otimes \boldsymbol{V}_c\right)^T$.

For every matrix of the form $\boldsymbol{M} = \boldsymbol{N} \otimes \boldsymbol{L}$, it holds that $\boldsymbol{M}x$ is the vectorized version of $\boldsymbol{L}\boldsymbol{X}\boldsymbol{N}^T$. By using this property and applying the relevant permutation, we can simulate multiplying a vector by $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{U}^T$, or $\boldsymbol{V}^T$ without storing the full matrix. The space complexity of this approach is $\Theta(n)$, which is required for computing the SVD of $\boldsymbol{A}_r$ and $\boldsymbol{A}_c$, as well as storing the permutations.

The above calculations remain valid when the blurring is zero-padded, *i.e.*, images are padded with zeroes so that the convolution is not circulant around the edges. We consider a zero-padded deblurring problem in our experiments. Note that the noiseless version of this problem has a simple solution – applying the pseudo-inverse of the blurring matrix on the blurry image. This solution attains 32.41dB in PSNR on ImageNet-1K, while DDRM improves upon it and achieves 35.64dB. When noise is added to the blurry image, such a simple solution amplifies the noise and fails to provide a valid output. Therefore, we opt not to report its results.

Furthermore, the above calculations are also applicable to blurring with strided convolutions. We use this fact in our implementation of the bicubic super resolution SVD, which can be interpreted as a strided convolution with a fixed kernel.

Table 3: Ablation studies on η and η_b.

(a) PSNR (↑).

η \ η_b	0.7	0.8	0.9	1.0
0.7	25.16	25.19	25.20	25.20
0.8	25.17	25.23	25.27	25.29
0.9	25.07	25.18	25.26	25.32
1.0	24.54	25.75	24.91	25.04

(b) KID $\times 10^3$ (↓).

η \ η_b	0.7	0.8	0.9	1.0
0.7	16.27	14.30	12.76	11.65
0.8	21.07	19.07	17.37	15.98
0.9	27.85	25.64	23.81	22.40
1.0	45.10	42.50	40.10	37.84

E Ablation Studies on Hyperparameters

η and η_b. Apart from the timestep schedules, DDRM has two hyperparameters η and η_b, which control the level of noise injected at each timestep. To identify an ideal combination, we perform a hyperparameter search over $\eta, \eta_b \in \{0.7, 0.8, 0.9, 1.0\}$ for the task of deblurring with $\sigma_y = 0.05$ in 1000 ImageNet validation images, using the model trained in [13]. It is possible to also consider different η values for $s_i = 0$ and $\sigma_i < \sigma_{\mathbf{y}}/s_i$; we leave that as future work.

We report PSNR and KID results in Table 3. From the results, we observe that generally (*i*) as η_b increases, PSNR increases while KID decreases, which is reasonable given that we wish to leverage the information from $\mathbf{y}$; (*ii*) as η increases, PSNR increases (except for $\eta = 1.0$) yet KID also increases, which presents a trade-off in reconstruction error and image quality (known as the perception-distortion trade-off [6]). Therefore, we choose $\eta_b = 1$ and $\eta = 0.85$ to balance performance on PSNR and KID when we report results.

Timestep schedules. The timestep schedule has a direct impact on NFEs, as the wall-clock time is roughly linear with respect to NFEs [45]. In Tables 5 and 6, we compare the PSNR, FID, and KID of DDRM with 20 or 100 timesteps (with or without conditioning) and default $\eta = 0.85$ and $\eta_b = 1$. We observe that DDRM with 20 or 100 timesteps have similar performance when other hyperparameters are identical, with DDRM (20) having a slight edge in FID and KID.

F Experimental Setup of DGP, RED, and SNIPS

Recall that we evaluated DGP [38], RED [40], and SNIPS [25] on 256×256 ImageNet 1K images, for the problems of $4\times$ super resolution and deblurring without any noise in the measurements. Below we expand on the experimental setup of each one.

For DGP [38], we use the same hyperparameters introduced in the original paper for MSE-biased super resolution. We note that the downscaling applied in DGP is different from the block averaging filter that we used, and the numbers they reported are on the 128×128 resolution. Nevertheless, in our experiments, DGP achieved a PSNR of 23.06 on ImageNet 1K 256×256 block averaging $4\times$ super resolution, which is similar to the 23.30 reported in the original work. When applied on the deblurring problem, we retained the same DGP hyperparameters as well.

For RED [40], we apply the iterative algorithm only in the luminance channel of the image in the YCbCr space, as done in the original paper for deblurring and super resolution. As for the denoising engine enabling the algorithm, we use the same diffusion model used in DDRM to enable as fair a comparison as possible. We use the last step of the diffusion model (equivalent to denoising with $\sigma = 0.005$), as we found it to work best empirically. We also chose the steepest-descent version (RED-SD), and $\lambda = 500$ for best PSNR performance given the denoiser we used. We also set $\sigma_0 = 0.01$ when the measurements are noiseless, because σ_0 cannot be 0 as RED divides by it.

In super resolution, RED is initialized with the bicubic upsampled low-res image. In deblurring, it is initialized with the blurry image. We then run RED on the ImageNet 1K for different numbers of steps (see Table 4), and choose the best PSNR for each problem. Namely, we show in our paper RED on super resolution with 100 steps, and on deblurring with 500 steps. Interestingly, RED achieves a PSNR close to its best for super resolution in just 20 steps. However, DDRM (with 20 steps) still

Table 4: RED results on ImageNet 1K (256×256) for $4\times$ super resolution and deblurring for different numbers of steps.

	SUPER-RES		DEBLURRING	
STEPS	PSNR↑	KID↓	PSNR↑	KID↓
0	25.65	44.90	19.26	38.00
20	26.05	52.51	23.49	21.99
100	26.08	53.55	25.00	26.09
500	26.00	54.19	26.16	21.21

Table 5: ImageNet 50K validation set (256×256) results on $4\times$ super resolution with additive noise of $\sigma_{\mathbf{y}} = 0.05$.

METHOD	PSNR↑	FID↓	KID↓	NFEs↓
BICUBIC	22.65	64.24	50.56	**0**
DDRM	24.70	20.16	15.25	100
DDRM-CC	**24.71**	18.22	13.57	100
DDRM	24.29	17.88	13.18	20
DDRM-CC	24.30	**15.92**	**11.47**	20

outperforms RED in PSNR, with substantially better perceptual quality (see Table 1 in the main paper).

Another interesting plug-and-play image restoration method is DPIR [57], which has recently achieved impressive results. It does so by applying the well-known Half Quadratic Splitting (HQS) plug-and-play algorithm using a newly proposed architecture. HQS requires an analytical solution of a minimization problem which is infeasible in general, due to the high memory requirements. DPIR provides efficient solutions for the specific degradation matrices $\mathbf{H}$ considered (circulant blurring, bicubic downsampling), which are different from the ones we consider (zero-padded blurring, block downsampling). In order to draw a fair comparison between the algorithms, one would have to use the same denosier architecture in both (as we have done for RED and SNIPS), and use the same degradation models. To apply DPIR on the same problems that we consider, we would need to substantially modify it and introduce efficient solutions. Therefore, we instead compare to RED, an alternative plug-and-play method.

SNIPS [25] did not originally work with ImageNet images. However, considering the method's similarity to DDRM (as both operate in the spectral space of $\boldsymbol{H}$), a comparison is necessary. We apply SNIPS with the same underlying diffusion model (with all 1000 timesteps) as DDRM for fairness. SNIPS evaluates the diffusion model τ times for each timestep. We set $\tau = 1$ so that SNIPS' runtime remains reasonable in comparison to the rest of the considered methods, and do not explore higher values of τ. It is worth mentioning that in the original work, τ was set to 3 for an LSUN bedrooms diffusion model with 1086 timesteps. We set $c = 0.67$ as it achieved the best PSNR performance.

The original work in SNIPS calculates the SVD of $\boldsymbol{H}$ directly, which hinders its ability to handle 256×256 images on typical hardware. In order to draw comparisons, we replaced the direct calculation of the SVD with our efficient implementation detailed in Appendix D.

In Figure 4 and Table 2 in the main paper, we show that DGP, RED, and SNIPS all fail to produce viable results when significant noise is added to the measurements. For these results, we use the same hyperparameters used in the noiseless case for all algorithms (except $\sigma_{\mathbf{y}}$ where applicable). While tuning the hyperparameters may boost performance, we do not explore that option as we are only interested in algorithms where given $\boldsymbol{H}$ and $\sigma_{\mathbf{y}}$, the restoration process is automatic. To further demonstrate DDRM's capabilities and speed, we evaluate it on the entire $50,000$-image ImageNet validation set in Tables 5 and 6, reporting Fréchet Inception distance (FID) [18] as well as KID, as enough samples are available.

Table 6: ImageNet 50K validation set (256×256) results on deblurring with additive noise of $\sigma_{\mathbf{y}} = 0.05$.

METHOD	PSNR↑	FID↓	KID↓	NFEs↓
BLURRY	18.05	93.36	74.13	**0**
DDRM	24.23	22.30	16.23	100
DDRM-CC	24.21	20.06	14.20	100
DDRM	24.60	21.60	15.65	20
DDRM-CC	**24.61**	**19.66**	**13.94**	20

G Runtime of Algorithms

In the main paper, we show the number of neural function evaluations (NFEs) as a proxy for the runtime of algorithms. Here, we consider the case of noisy deblurring, and measure the runtime of DDRM, RED, SNIPS, and DGP on an Nvidia RTX 3080 GPU. For each image, DDRM, RED, and SNIPS all run at around 0.09 s/it (seconds per iteration), with negligible differences of < 0.01s/it. We note that the denoiser model of DDRM, SNIPS, and RED is the same, so runtime is almost perfectly linearly correlated with NFEs. As for DGP, it uses a different model (a GAN), and it is slightly slower than our denoiser (0.11 s/it); this is partly because DGP requires additional gradient computations in order to perform an update. All in all, we observe that the runtime is indeed linear with NFEs, and since no algorithm has a significant runtime advantage over the rest, we prefer to use NFEs as a proxy for runtime, as it is a hardware-independent measure.

In this paper, we used pretrained generative models for image restoration. Since we didn't train any models, a single Nvidia RTX 3080 GPU was sufficient to run all experiments that were shown in the paper and the appendices.

H ILVR as a special case of DDRM

Given a generative diffusion model (*e.g.* DDPM [19]) that can predict $\mathbf{x}$ given $\mathbf{x}_{t+1}$ and $t+1$ for $t \in [0, T-1]$, and a noiseless measurement $\mathbf{y} = \boldsymbol{H}\mathbf{x}$, where $\boldsymbol{H}$ is a downscaling matrix, the Iterative Latent Variable Refinement (ILVR) [9] algorithm can sample from the posterior distribution $p_\theta^{(t)}(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{y})$ for $t \in [0, T-1]$.

We assume a variance exploding diffusion model, *i.e.* $\mathbf{x}_t = \mathbf{x} + \sigma_t \epsilon_t$ where $\epsilon_t \sim \mathcal{N}(0, \boldsymbol{I})$, without loss of generality (because it is equivalent to the variance preserving scheme, as we show in Appendix B). Under this setting, ILVR applies the following updates for $t = T-1, \dots, 0$:

$$\begin{aligned}
\mathbf{x}'_t &= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t, \\
\mathbf{y}_t &= \boldsymbol{H}^\dagger \mathbf{y} + \sigma_t \epsilon'_t, \\
\mathbf{x}_t &= \mathbf{x}'_t - \boldsymbol{H}^\dagger \boldsymbol{H} \mathbf{x}'_t + \boldsymbol{H}^\dagger \boldsymbol{H} \mathbf{y}_t,
\end{aligned}$$

where $\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1)$ is the prediction for $\mathbf{x}$ given by the diffusion model at timestep $t+1$, $\epsilon_t \sim \mathcal{N}(0, \boldsymbol{I})$, and $\epsilon'_t \sim \mathcal{N}(0, \boldsymbol{I})$. Substituting $\mathbf{x}'_t$, $\mathbf{y}_t$, and $\boldsymbol{H} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, the last equation becomes

$$\begin{aligned}
\mathbf{x}_t &= \mathbf{x}'_t - \boldsymbol{H}^\dagger \boldsymbol{H} \mathbf{x}'_t + \boldsymbol{H}^\dagger \boldsymbol{H} \mathbf{y}_t \\
&= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t - \boldsymbol{H}^\dagger \boldsymbol{H} \left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t \right) + \boldsymbol{H}^\dagger \boldsymbol{H} \left(\boldsymbol{H}^\dagger \mathbf{y} + \sigma_t \epsilon'_t \right) \\
&= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t - \boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t \right) + \boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T \left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \mathbf{y} + \sigma_t \epsilon'_t \right) \\
&= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t - \boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma}\boldsymbol{V}^T \left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t \right) + \boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma}\boldsymbol{V}^T \left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \mathbf{y} + \sigma_t \epsilon'_t \right) \\
&= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t - \boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma}\boldsymbol{V}\boldsymbol{V}^T \left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t \right) + \boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma}\boldsymbol{V}\boldsymbol{V}^T \left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \mathbf{y} + \sigma_t \epsilon'_t \right) \\
&= \mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t - \boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma} \left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t \epsilon_t \right) + \boldsymbol{\Sigma}^\dagger \boldsymbol{\Sigma} \left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \mathbf{y} + \sigma_t \epsilon'_t \right).
\end{aligned}$$

The second to last equality holds because $\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}$ is a square diagonal matrix, and matrix multiplication with a square diagonal matrix is commutative. Recall that $\bar{\mathbf{x}}_t = \boldsymbol{V}^T \mathbf{x}_t$, $\bar{\mathbf{y}} = \boldsymbol{\Sigma}^\dagger \boldsymbol{U}^T \mathbf{y}$, and $\bar{\mathbf{x}}_{\theta,t} =$

$\boldsymbol{V}^T\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1)$, thus

$$\begin{aligned}
\bar{\mathbf{x}}_t &= \boldsymbol{V}^T\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t\boldsymbol{V}^T\epsilon_t - \boldsymbol{V}^T\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t\epsilon_t\right) + \boldsymbol{V}^T\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T\mathbf{y} + \sigma_t\epsilon_t'\right) \\
&= \boldsymbol{V}^T\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t\boldsymbol{V}^T\epsilon_t - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\left(\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t\epsilon_t\right) + \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\left(\boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T\mathbf{y} + \sigma_t\epsilon_t'\right) \\
&= \boldsymbol{V}^T\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) + \sigma_t\boldsymbol{V}^T\epsilon_t - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\mathbf{x}_{\theta,t}(\mathbf{x}_{t+1}, t+1) - \sigma_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\epsilon_t + \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\boldsymbol{V}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T\mathbf{y} + \sigma_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\epsilon_t' \\
&= \bar{\mathbf{x}}_{\theta,t} + \sigma_t\boldsymbol{V}^T\epsilon_t - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\bar{\mathbf{x}}_{\theta,t} - \sigma_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\epsilon_t + \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T\mathbf{y} + \sigma_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\boldsymbol{V}^T\epsilon_t' \\
&= \left(\boldsymbol{I} - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\right)\bar{\mathbf{x}}_{\theta,t} + \left(\boldsymbol{I} - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\right)\sigma_t\boldsymbol{V}^T\epsilon_t + \boldsymbol{\Sigma}^\dagger\boldsymbol{U}^T\mathbf{y} + \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\sigma_t\boldsymbol{V}^T\epsilon_t' \\
&= \left(\boldsymbol{I} - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\right)\bar{\mathbf{x}}_{\theta,t} + \left(\boldsymbol{I} - \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\right)\sigma_t\boldsymbol{V}^T\epsilon_t + \bar{\mathbf{y}} + \boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\sigma_t\boldsymbol{V}^T\epsilon_t'.
\end{aligned}$$

The matrix $\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}$ is a square diagonal matrix with zeroes in its entries where the singular value is zero, and ones otherwise. In addition, $\boldsymbol{\Sigma}^\dagger$ has a row of zeroes when the singular value is zero. Therefore, it holds that

$$\bar{\mathbf{x}}_t^{(i)} = \begin{cases} \bar{\mathbf{x}}_{\theta,t}^{(i)} + \left(\sigma_t\boldsymbol{V}^T\epsilon_t\right)^{(i)} & \text{if } s_i = 0 \\ \bar{\mathbf{y}}^{(i)} + \left(\sigma_t\boldsymbol{V}^T\epsilon_t'\right)^{(i)} & \text{if } s_i \neq 0 \end{cases}, \tag{28}$$

which in turn implies

$$p_\theta^{(t)}(\bar{\mathbf{x}}_t^{(i)}|\mathbf{x}_{t+1}, y) = \begin{cases} \mathcal{N}\left(\bar{\mathbf{x}}_{\theta,t}^{(i)}, \sigma_t^2\boldsymbol{I}\right) & \text{if } s_i = 0 \\ \mathcal{N}\left(\bar{\mathbf{y}}^{(i)}, \sigma_t^2\boldsymbol{I}\right) & \text{if } s_i \neq 0 \end{cases}. \tag{29}$$

This distribution is exactly the same as Equation 8 in the main paper when $\eta = \eta_b = 1$ and $\sigma_{\mathbf{y}} = 0$.

As for $\mathbf{x}_T$, ILVR initializes it by sampling from $\mathcal{N}\left(0, \sigma_T^2\boldsymbol{I}\right)$ (or $\mathcal{N}\left(0, \boldsymbol{I}\right)$ in the variance preserving case) while DDRM samples according to Equation 7 in the main paper. The two initializations have the same variance but differ in the mean. This difference has a negligible effect on the end result since the variance is much larger than the difference in the means. Therefore, the above form of ILVR is a specific form of a DDRM (with $\eta = \eta_b = 1$), posed as a solution for linear inverse problems without noise in the measurements.

In their experiments, ILVR only tested $\boldsymbol{H}$ which is the bicubic downscaling matrix with varying scale factors. In theory, ILVR can also work for any linear degradation $\boldsymbol{H}$, as long as $\mathbf{y}$ does not contain noise.

I Additional Results

We provide additional figures below showing DDRM's versatility across different datasets, inverse problems, and noise levels (Figures 7, 8, 10, 11, and 12). We also showcase the sample diversity provided by DDRM in Figure 9; we present more uncurated samples from the ImageNet experiments in Figures 13 and 14. Moreover, we further illustrate DDRM's advantage over previous unsupervised methods by evaluating on two additional inverse problems: (i) $4\times$ super-resolution with the popular bicubic downsampling kernel; and (ii) deblurring with an anisotropic Gaussian blur kernel (with $\sigma = 20$ horizontally and $\sigma = 1$ vertically), mimicing motion blur. We show both noiseless and noisy versions in Tables 7 and 8, respectively. To maintain the unsupervised nature of the tested methods, we use the same hyperparameters as in block-averaging super-resolution and uniform deblurring.

Table 7: Noiseless $4\times$ super-resolution (using a bicubic kernel) and anisotropic Gaussian deblurring results on ImageNet 1K (256×256).

Method	$4\times$ super-resolution (Bicubic)				Deblurring (Anisotropic)			
	PSNR↑	SSIM↑	KID↓	NFEs↓	PSNR↑	SSIM↑	KID↓	NFEs↓
Baseline	26.06	0.73	72.41	**0**	19.96	0.58	25.23	**0**
DGP	20.82	0.50	29.62	1500	23.35	0.59	20.10	1500
RED	26.14	0.73	47.61	100	29.39	0.86	10.49	500
SNIPS	17.65	0.23	30.30	1000	33.34	0.86	0.58	1000
DDRM	**27.09**	**0.76**	**12.78**	20	**36.02**	**0.93**	**0.41**	20

Table 8: Noisy ($\sigma_{\mathbf{y}} = 0.05$) $4\times$ super-resolution (using a bicubic kernel) and anisotropic Gaussian deblurring results on ImageNet 1K (256×256).

Method	$4\times$ super-resolution (Bicubic)				Deblurring (Anisotropic)			
	PSNR↑	SSIM↑	KID↓	NFEs↓	PSNR↑	SSIM↑	KID↓	NFEs↓
Baseline	21.68	0.40	73.87	**0**	19.96	0.27	55.00	**0**
DGP	19.68	0.40	44.07	1500	22.64	0.53	25.38	1500
RED	22.65	0.46	54.90	100	11.97	0.10	130.30	500
SNIPS	16.16	0.14	69.69	1000	17.49	0.20	48.37	1000
DDRM	**25.53**	**0.68**	**14.57**	20	**26.95**	**0.73**	**10.34**	20

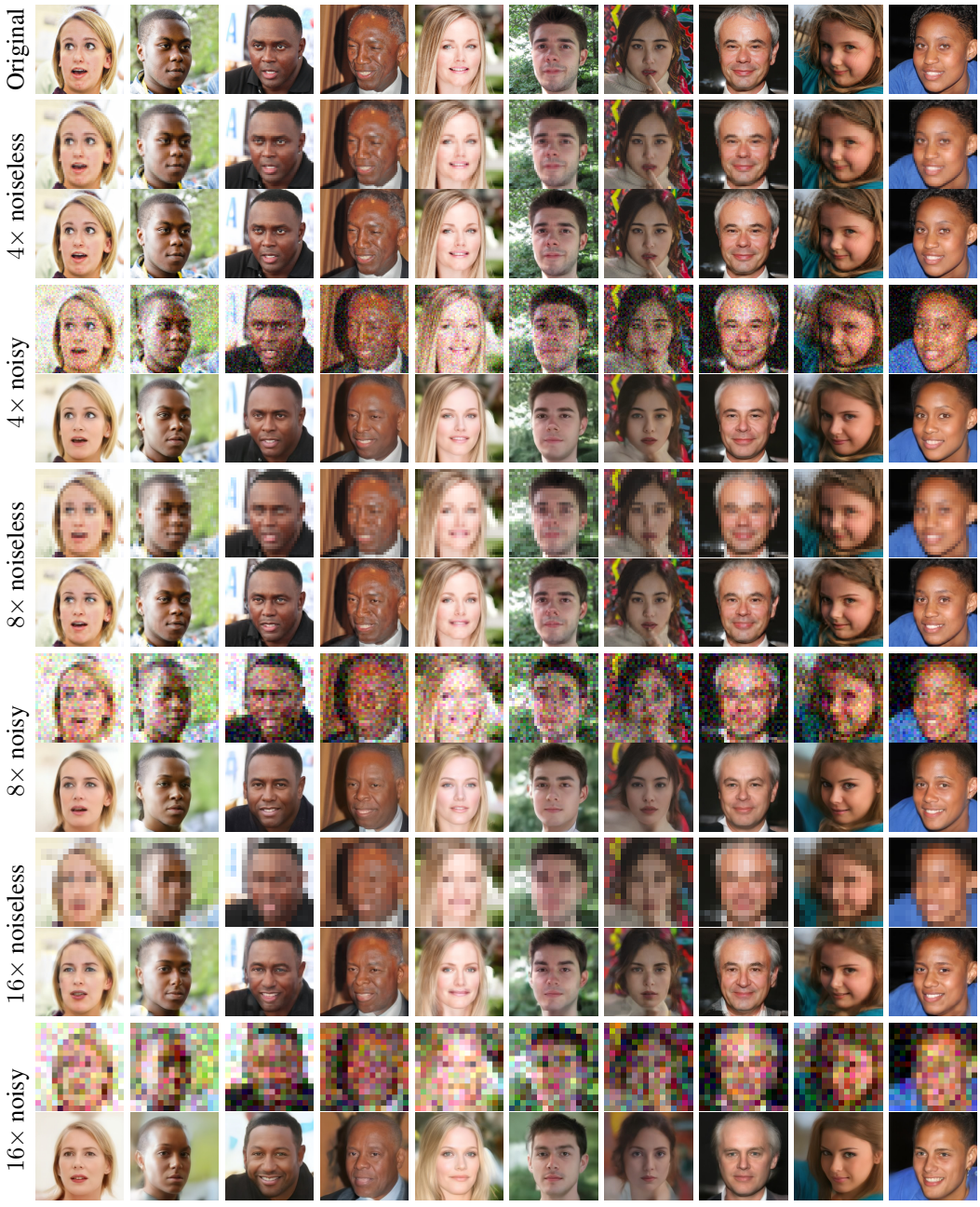


Figure 7: Pairs of low-res and recovered 256×256 face images with a 20-step DDRM. Noisy low-res images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.1$.

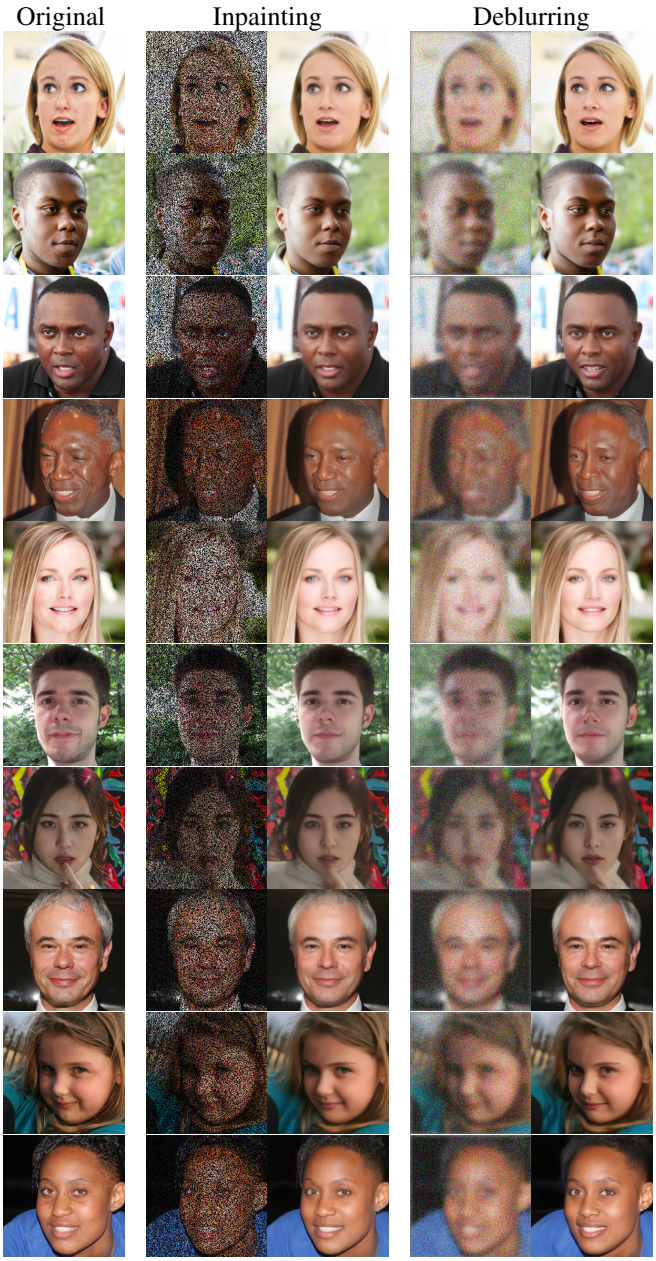


Figure 8: Pairs of degraded and recovered 256×256 face images with a 20-step DDRM. Degraded images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.1$.

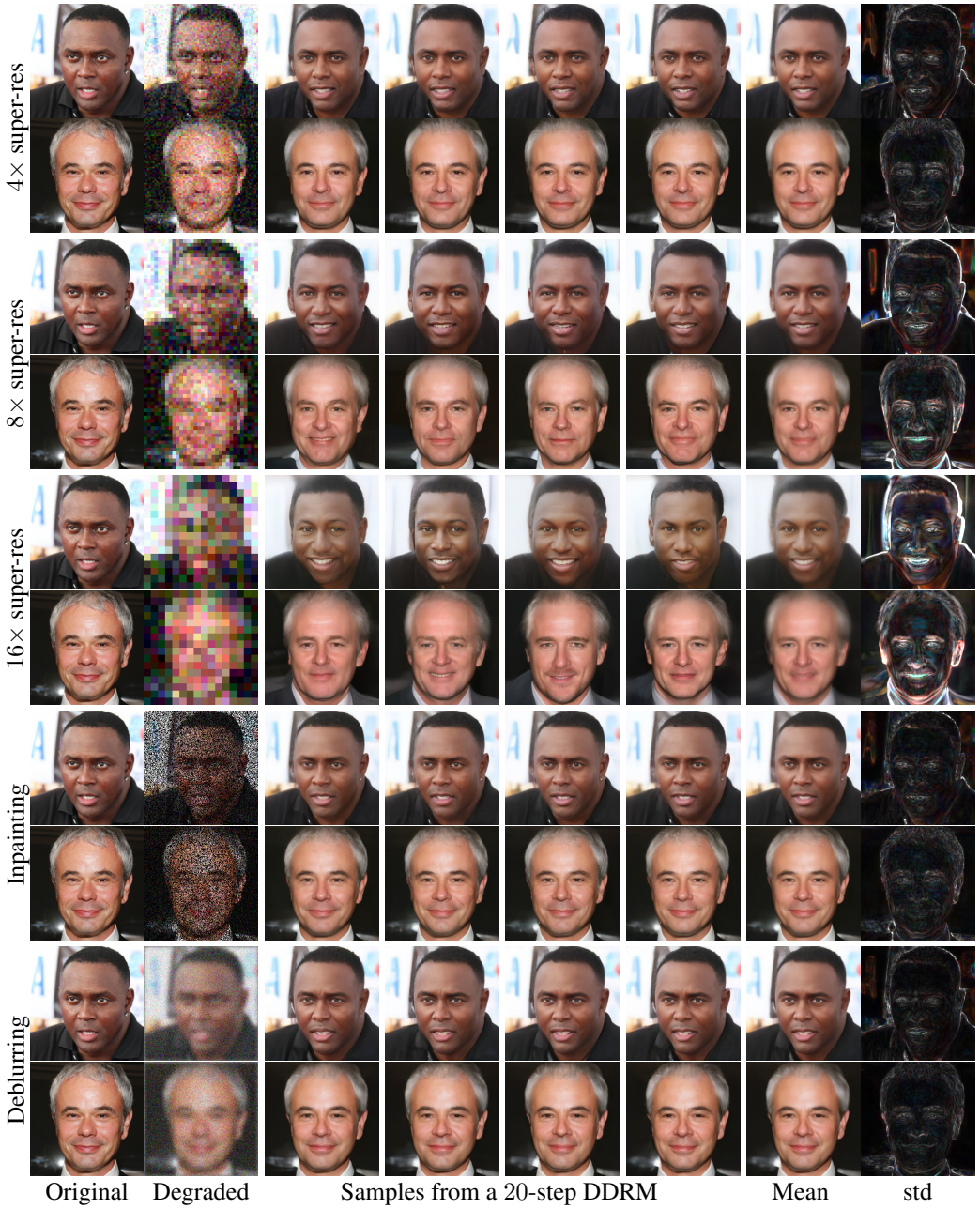


Figure 9: Original, degraded, and 6 recovered 256×256 face images with a 20-step DDRM. Degraded images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.1$. The mean and standard deviation (scaled by 4) of the sampled solution is shown.

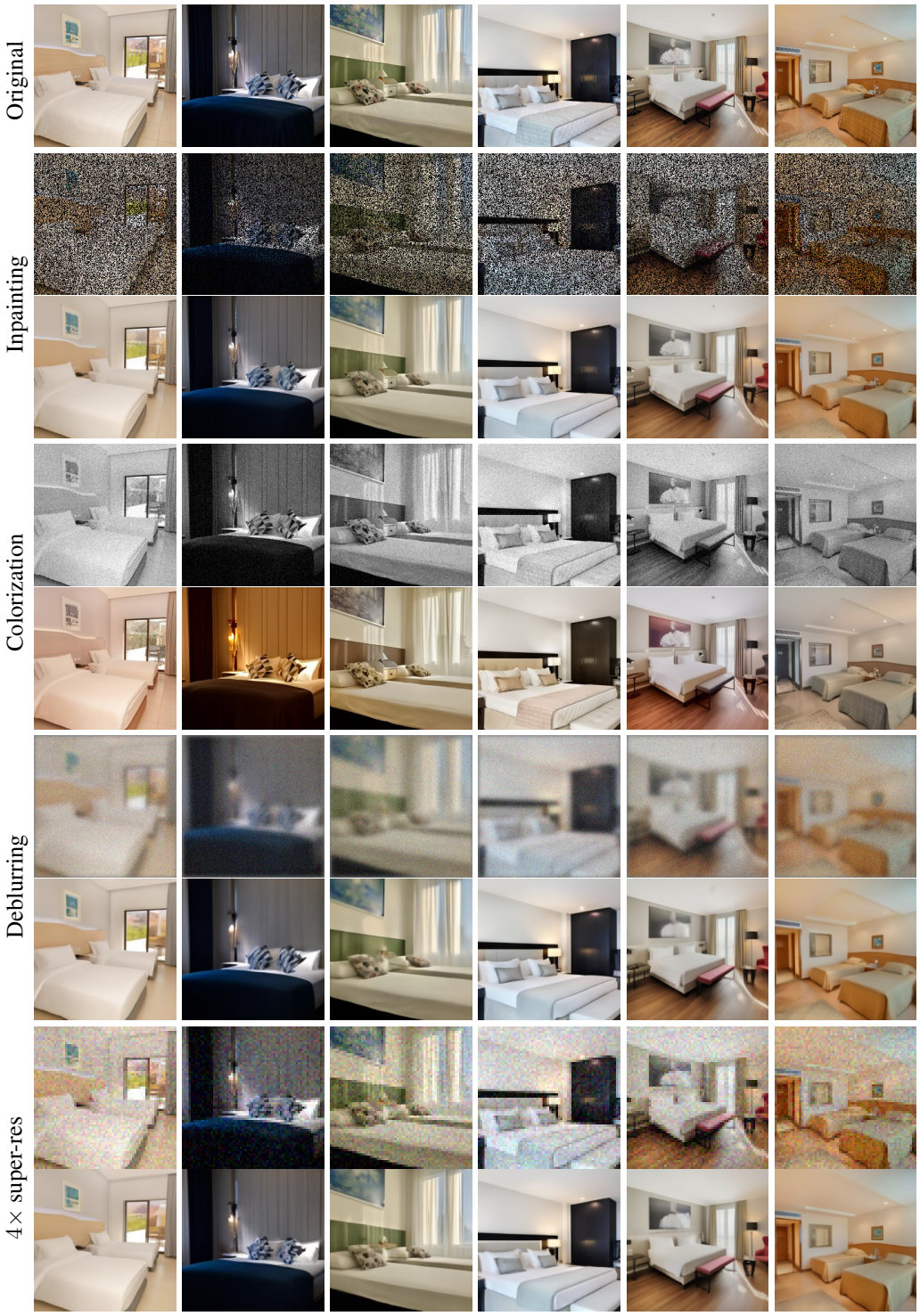


Figure 10: Pairs of degraded and recovered 256×256 bedroom images with a 20-step DDRM. Degraded images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.05$.

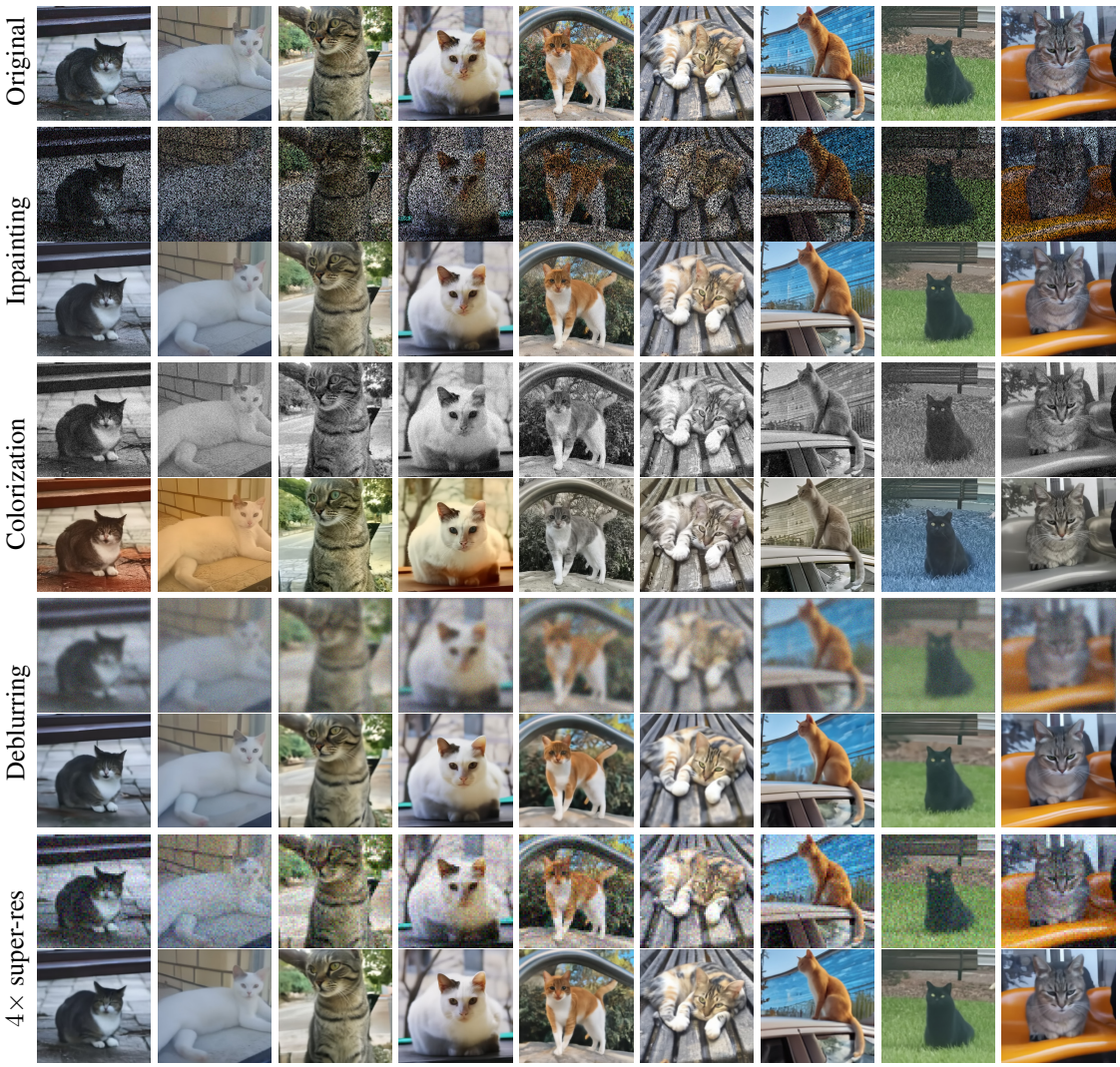


Figure 11: Pairs of degraded and recovered 256×256 cat images with a 20-step DDRM. Degraded images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.05$.

Figure 12: Pairs of degraded and recovered 256×256 USC-SIPI images with a 20-step DDRM using an ImageNet diffusion model. Degraded images contain noise with a standard deviation of $\sigma_{\mathbf{y}} = 0.05$.

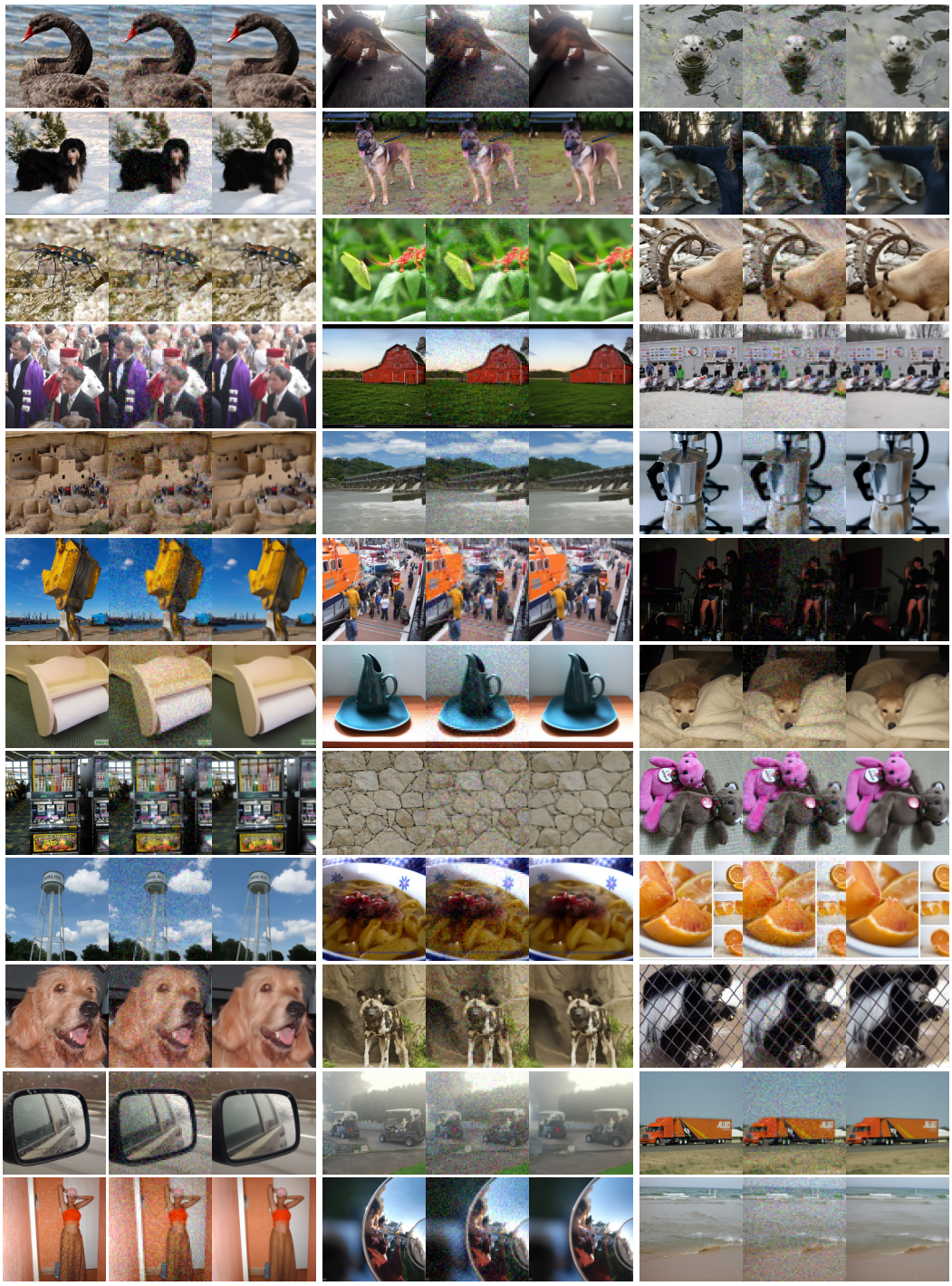


Figure 13: Uncurated samples from the noisy $4\times$ super resolution ($\sigma_{\mathbf{y}} = 0.05$) task on 256×256 ImageNet 1K. Each triplet contains (from left to right): the original image, the low-res image, and the restored image with DDRM-20.

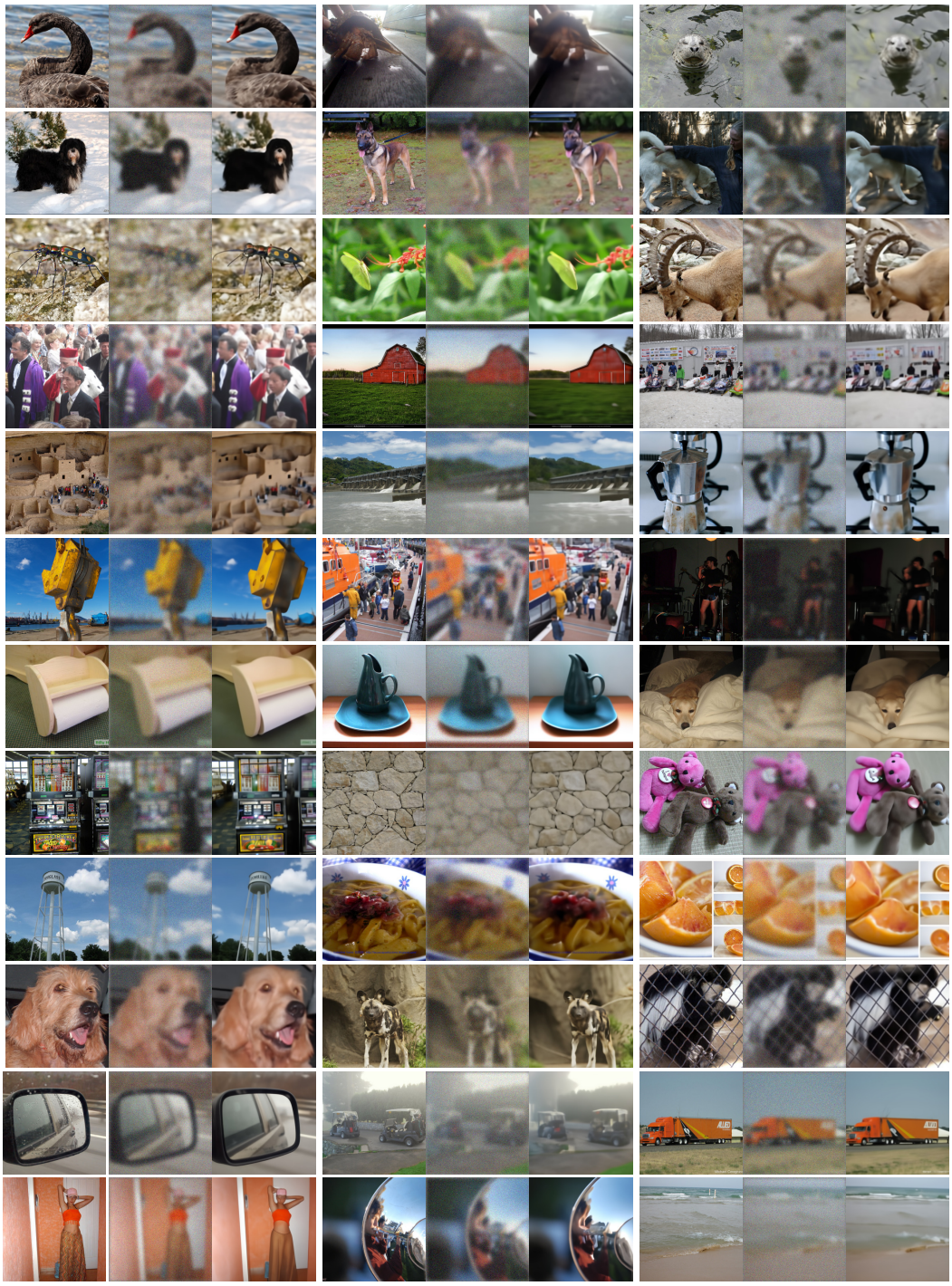


Figure 14: Uncurated samples from the noisy deblurring ($\sigma_{\mathbf{y}} = 0.05$) task on 256×256 ImageNet 1K. Each triplet contains (from left to right): the original image, the blurry image, and the restored image with DDRM-20.

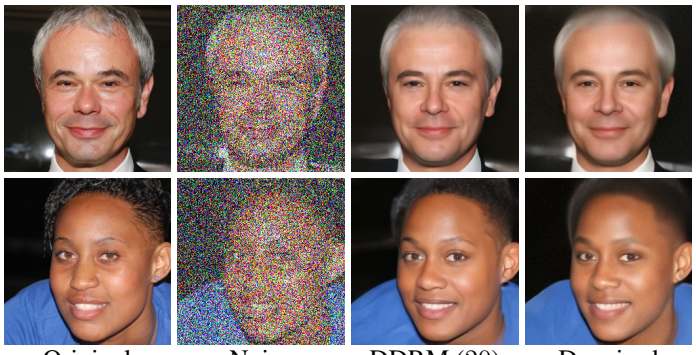


Figure 15: Denoising ($\sigma_{\mathbf{y}} = 0.75$) face images. DDRM restores more fine details (*e.g.* hair) than an MMSE denoiser. The denoiser used here is the denoising diffusion function $f_\theta(\mathbf{x}_t, t)$ used by DDRM, where t minimizes $|\sigma_t - \sigma_{\mathbf{y}}|$.