# SITCOM: Step-wise Triple-Consistent Diffusion Sampling for Inverse Problems

Ismail R. Alkhouri[1,2*], Shijun Liang[1], Cheng-Han Huang[1], Jimmy Dai[2],
Qing Qu[2], Saiprasad Ravishankar[1], Rongrong Wang[1]
[1]Michigan State University
[2]University of Michigan

Diffusion models (DMs) are a class of generative models that allow sampling from a distribution learned over a training set. When applied to solving inverse imaging problems (IPs), the reverse sampling steps of DMs are typically modified to approximately sample from a measurement-conditioned distribution in the image space. However, these modifications may be unsuitable for certain settings (such as in the presence of measurement noise) and non-linear tasks, as they often struggle to correct errors from earlier sampling steps and generally require a large number of optimization and/or sampling steps. To address these challenges, we state three conditions for achieving measurement-consistent diffusion trajectories. Building on these conditions, we propose a new optimization-based sampling method that not only enforces the standard data manifold measurement consistency and forward diffusion consistency, as seen in previous studies, but also incorporates backward diffusion consistency that maintains a diffusion trajectory by optimizing over the input of the pre-trained model at every sampling step. By enforcing these conditions, either implicitly or explicitly, our sampler requires significantly fewer reverse steps. Therefore, we refer to our accelerated method as **S**tep-w**i**se **T**riple-**Co**nsistent Sa**m**pling (**SITCOM**). Compared to existing state-of-the-art baseline methods, under different levels of measurement noise, our extensive experiments across five linear and three non-linear image restoration tasks demonstrate that SITCOM achieves competitive or superior results in terms of standard image similarity metrics while requiring a significantly reduced run-time across all considered tasks.

## 1. Introduction

Inverse problems (IPs) arise in a wide range of science and engineering applications, including computer vision [1], signal processing [2], medical imaging [3], remote sensing [4], and geophysics [5]. In these applications, the primary goal is to recover an unknown image or signal $\mathbf{x} \in \mathbb{R}^n$ from measurements or degraded image $\mathbf{y} \in \mathbb{R}^m$, which are often corrupted by noise. Mathematically, the unknown signal and the measurements are related as

$$\mathbf{y} = \mathcal{A}(\mathbf{x}) + \mathbf{n}\,, \tag{1}$$

where $\mathcal{A}(\cdot) : \mathbb{R}^n \to \mathbb{R}^m$ (with $m \leq n$) represents the linear or non-linear forward operator that models the measurement process, and $\mathbf{n} \in \mathbb{R}^m$ denotes the noise in the measurement domain, e.g., assumed sampled from a Gaussian distribution $\mathcal{N}(\mathbf{0}, \sigma_{\mathbf{y}}^2\mathbf{I})$, where $\sigma_{\mathbf{y}} > 0$ denotes the noise level. Exactly solving these inverse problems is challenging due to their ill-posedness in many settings, requiring advanced techniques to achieve accurate solutions.

Deep learning techniques have recently been utilized as a prior to aid in solving these problems [6, 7]. One framework that has shown significant potential is the use of generative models, particularly diffusion models (DMs) [8]. Given a training dataset, DMs are trained to learn the underlying distribution $p(\mathbf{x})$. During inference, DMs enable sampling from this learned distribution through an iterative procedure [9]. When employed to solving inverse problems, DM-based IP solvers often

*The first two authors contributed equally.

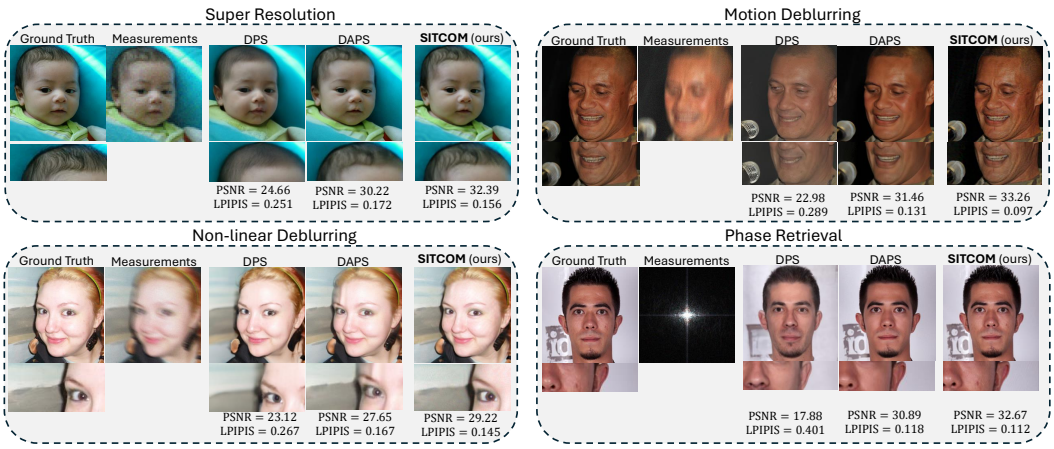


Figure 1: **Qualitative results on the FFHQ dataset** on two linear tasks (*top*) and two non-linear tasks (*bottom*) under measurement noise of $\sigma_{\mathbf{y}} = 0.05$. The PSNR and LPIPS values are given below each restored image. Zoomed-in regions show how SITCOM captures greater image details when compared to two general (non)linear DM-based methods (DPS [10] and DAPS [12]).

modify the reverse sampling steps to allow sampling from the measurements-conditioned distribution $p(\mathbf{x}|\mathbf{y})$ [10, 11]. These modifications typically rely on approximations that may not be suitable for all tasks and settings, and in addition to generally requiring many sampling iterations, often suffer from errors accumulated during early diffusion sampling steps [12]. In most DM-based IP solvers, these approximations are designed to enforce standard measurement consistency on the estimated image (or posterior mean) at every reverse sampling iteration, as in [10], and may also include resampling using the forward diffusion process (which we refer to as forward diffusion consistency), such as in [13, 14].

A key bottleneck in DMs is their computational speed, as they are slower than other generative models due to the large number of sampling steps. Although various methods have been proposed to reduce sampling frequency (e.g., [15]), these improvements have yet to be fully realized for DMs applied to IPs. Most existing methods still require dense sampling, which continues to pose speed challenges.

**Contributions:** In this paper, we: (*i*) identify key issues in accelerating DMs for IPs, (*ii*) propose three conditions that could fully leverage the information from the measurements and the pre-trained diffusion model to effectively address these issues, and (*iii*) present a new optimization-based method in the pixel space that satisfies these conditions. We refer to our accelerated sampling method as **S**tep-wi**s**e **T**riple-**Co**nsistent Sa**m**pling (SITCOM). We evaluate our method on several image restoration tasks: Super Resolution, Box In-painting, Random In-painting, Motion Deblurring, Gaussian Deblurring, Non-linear Deblurring, High Dynamic Range, and Phase Retrieval. Compared to leading baselines, our approach consistently achieves either state-of-the-art or highly competitive quantitative results, while also reducing the number of sampling steps and, consequently, the computational time. See Figure 1 for examples.

## 2. Background: Diffusion Models & Their Usage in Solving IPs

Pre-trained Diffusion Models (DMs) generate images by applying a pre-defined iterative denoising process [8]. In the Variance-Preserving Stochastic Differentiable Equations (SDEs) setting [9, 16], DMs are formulated using the forward and reverse processes

$$d\mathbf{x}_t = -\frac{\beta_t}{2}\mathbf{x}_t dt + \sqrt{\beta_t} d\mathbf{w}\,, \qquad d\mathbf{x}_t = -\beta_t\Big[\frac{1}{2}\mathbf{x}_t + \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\Big]dt + \sqrt{\beta_t}d\bar{\mathbf{w}}\,, \tag{2}$$

where $\beta : \{0, \dots, T\} \to (0, 1)$ is a pre-defined function that controls the amount of additive perturbations at time $t$, $\mathbf{w}$ (resp. $\bar{\mathbf{w}}$) is the forward (resp. reverse) Weiner process [17], $p_t(\mathbf{x}_t)$ is the

distribution of $\mathbf{x}_t$ at $t$, and $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ is the score function that is replaced by a neural network (typically a time-encoded U-Net [18]) $\boldsymbol{s} : \mathbb{R}^n \times \{0, \dots, T\} \to \mathbb{R}^n$, parameterized by $\theta$. In practice, given the score function $\boldsymbol{s}_\theta$, the SDEs in (2) can be discretized as in (3) where $\boldsymbol{\eta}_t, \boldsymbol{\eta}_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

$$\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\eta}_{t-1}\,, \quad \mathbf{x}_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\Big[\mathbf{x}_t + \beta_t \boldsymbol{s}_\theta(\mathbf{x}_t, t)\Big] + \sqrt{\beta_t}\boldsymbol{\eta}_t\,. \tag{3}$$

When employed to solve inverse problems, the score function in (2) is replaced by a conditional score function which, by Bayes' rule, is $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y}) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$. Solving the SDE in (2) with the conditional score is referred to as *posterior sampling* [10]. As there doesn't exist a closed-form expression for the term $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ (which is termed as the measurements matching term in [19]), previous works have explored different approaches, which we will briefly discuss below. We refer the reader to the recent survey in [19] for an overview on DM-based methods for solving IPs.

A well-known method is Diffusion Posterior Sampling (DPS) [10], which uses the approximation $p(\mathbf{y}|\mathbf{x}_t) \approx p(\mathbf{y}|\hat{\mathbf{x}}_0)$ where $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ (or simply $\hat{\mathbf{x}}_0$) is the estimated image at time $t$ as a function of the pre-trained model and $\mathbf{x}_t$ (Tweedie's formula [20]), given as

$$\hat{\mathbf{x}}_0(\mathbf{x}_t) = \frac{1}{\sqrt{\bar{\alpha}_t}}\Big[\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\Big] =: f(\mathbf{x}_t; t, \boldsymbol{\epsilon}_\theta)\,, \tag{4}$$

where $\bar{\alpha}_t = \prod_{j=1}^{t} \alpha_j$ and $\alpha_t = 1 - \beta_t$. We call the function $f$, defined in (4), as **'Tweedie-network denoiser'** (also termed as 'posterior mean predictor' in [21]). Here, $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) = -\sqrt{1-\bar{\alpha}_t}\boldsymbol{s}_\theta(\mathbf{x}_t, t)$ [22] outputs the noise in $\mathbf{x}_t$. Tweedie's formula, like in our method, is also adopted in other DM-based IP solvers such as [23–25]. The drawback of these methods is that they require a large number of sampling steps.

The work in ReSample [14], solves an optimization problem on the estimated posterior mean in the latent space for many steps to enforce measurement consistency, requiring many sampling and optimization steps.

The work in [26] introduced RED-Diff, a variational Bayesian method that fits a Gaussian distribution to the posterior distribution of the clean image conditional on the measurements. This approach involves solving an optimization problem using stochastic gradient descent (SGD) to minimize a data-fitting term while maximizing the likelihood of the reconstructed image under the denoising diffusion prior (as a regularizer). However, the SGD process requires multiple iterations, each involving evaluations of the pre-trained DM on a different noisy image at some randomly selected time, making it quite computationally expensive.

Recently, Decoupling Consistency with Diffusion Purification (DCDP) [1] proposed separating diffusion sampling steps from measurement consistency by using DMs as diffusion purifiers [3, 27], with the goal of reducing the run-time. However, DCDP requires tuning the number of forward diffusion steps for purification. Shortly after, Decoupled Annealing Posterior Sampling (DAPS) [12] introduced another decoupled approach, incorporating gradient descent noise annealing via Langevin dynamics. DAPS, similar to DPS and RED-Diff, also requires a large number of sampling and optimization steps. Under measurement noise, DCDP achieves SOTA run-time across various linear restoration tasks, while DAPS sets the SOTA in restoration quality. Both will serve as primary baselines in our experiments.

# 3. SITCOM: Step-wise Triple-Consistent Sampling

## 3.1. Motivation: Addressing the Challenges in Applying DMs to IPs

Most inverse problems are ill-conditioned and undersampled. DMs, when trained on a dataset that closely resembles the target image, can provide critical information to alleviate ill-conditioning and improve recovery. Despite various previous efforts, a key challenge remains: How to *efficiently* integrate DMs into the framework of inverse problems? We will now elaborate on this challenge in detail.

The standard reverse sampling procedure in DMs consists of applying the backward discrete steps in (3) for $t \in \{T, T-1, \ldots, 1\}$, forming the standard diffusion trajectory for which $\mathbf{x}_0$ is the generated image. To incorporate the measurement $\mathbf{y}$ into these steps, a common approach adopted in previous works that demonstrate superior performance (e.g., [1, 12, 14]) is to the $\hat{\mathbf{x}}_0$ computed via (4) as follows:

$$\hat{\mathbf{x}}_0'(\mathbf{x}_t) = \arg\min_{\mathbf{x}} \|\mathcal{A}(\mathbf{x}) - \mathbf{y}\|^2 + \lambda \|\mathbf{x} - \hat{\mathbf{x}}_0(\mathbf{x}_t)\|^2 \,, \tag{5}$$

where $\lambda \in \mathbb{R}_+$ is a regularization parameter. The $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$ obtained from (5) is close to $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ while also remaining consistent with the measurements. When using $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$ to sample $\mathbf{x}_{t-1}$, the second formula in (3) can be rewritten as in (6), where the derivation is provided in Appendix A.

$$\mathbf{x}_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t \,. \tag{6}$$

By substituting $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ into (6) with the measurement-consistent $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$, the modified sampling formula becomes:

$$\mathbf{x}_{t-1} = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0'(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t \,. \tag{7}$$

While this approach effectively ensures data consistency at each step, it inevitably causes $\hat{\mathbf{x}}_0'$ to deviate from the diffusion trajectory[2], leading to two major issues:

- **(I1)** The image $\hat{\mathbf{x}}_0(\mathbf{x}_t)$, initially constructed through Tweedie's formula, usually appears quite natural (e.g., columns 3 to 5 of Figure 2 ); however, the modified version, $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$, is likely to exhibit severe artifacts (e.g., columns 6 to 8 of Figure 2).
- **(I2)** Since the DM network, $\boldsymbol{\epsilon}_\theta$, is trained via minimizing the objective function $\mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\|\epsilon - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2$ (denoising score matching [20]) on a finite dataset, it performs best on noisy images lying in the high-density regions of the *training distribution* $\mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$, $\mathbf{x}_0 \sim p(\mathbf{x}_0)$. We define an algorithm as **forward-consistent** if it likely applies $\boldsymbol{\epsilon}_\theta$ only to in-distribution inputs (i.e., those from the same distribution used for training). For example, if the forward diffusion used to train $\boldsymbol{\epsilon}_\theta$ adds Gaussian noise, the in-distribution input to $\boldsymbol{\epsilon}_\theta$ should ideally be sampled from a Gaussian with specific parameters. If Poisson noise is used in the forward process, inputs drawn from suitable Poisson distributions are more likely to fall within the well-trained region of the network. In summary, forward consistency requires that inputs to $\boldsymbol{\epsilon}_\theta$ during sampling align with the forward process. While the $\mathbf{x}_{t-1}$ generated from (6) is forward-consistent by design, the one generated from the modified formula (7) is not. Therefore, in the latter case, the DM network, $\boldsymbol{\epsilon}_\theta$, may be applied to many out-of-distribution inputs, leading to degraded performance.

We pause to verify our claimed Issue (**I1**) through a box-inpainting experiment. Columns 3 to 5 of Figure 2 show $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$ at various $t$. The results clearly demonstrate successful enforcement of data consistency, as the region outside the box aligns with the original image. However, this enforcement compromises the natural appearance of the image, introducing significant artifacts in the reconstructed area inside the box. Details about the setting of the results in Figure 2 are given in Section C.

Issue (**I2**) was previously observed in [13], which proposed a remedy known as '*resampling*'. In this approach, the sampling formula in (7) is replaced by

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta}_t \,. \tag{8}$$

Provided $\hat{\mathbf{x}}_0$ is close to the ground truth $\mathbf{x}_0$, $\mathbf{x}_{t-1}$ generated this way will stay in-distribution with high probability. For a more detailed explanation of the rationale behind this remedy, we refer the reader to [13]. This method has since been adopted by subsequent works, such as [12, 14], and we will also employ it to address (**I2**).

[2]Diffusion trajectory refers to the path that leads to an in-distribution image, where the distribution is the one learned by the DM from the training set.

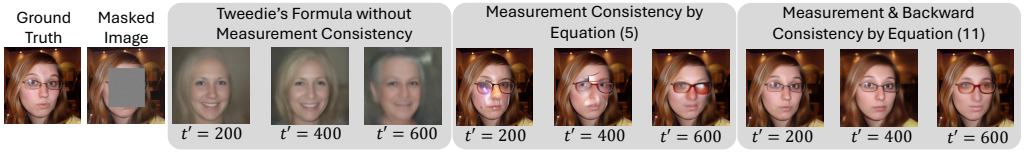


Figure 2: Effects of enforcing backward-consistency in box-inpainting: Results of using Tweedie's formula without measurement consistency (columns 3 to 5), enforcing measurement-consistency via (5) (columns 6 to 9), and enforcing both measurement-consistency and backward-consistency via (12) (columns 10 to 12) at different time steps $t'$. Experimental details are given in Appendix C.

## 3.2. Network Regularization & Backward Diffusion Consistency

Previous studies, such as [12, 14], mitigate issue **(I1)** by using a large number of sampling steps, which inevitably increases the computational burden. In contrast, this paper proposes employing a *network regularization* to resolve issue **(I1)**. This approach not only accelerates convergence but also enhances reconstruction quality. Let's first clarify the underlying intuition.

It is widely observed that the U-Net architecture or trained transformers exhibit an effective image bias [28–31]. From columns 3 to 5 of Figure 2, we observe that without enforcing data consistency, the reconstructed $\hat{\mathbf{x}}_0$, derived directly from Tweedie-network denoiser $f(\mathbf{x}_t; t, \boldsymbol{\epsilon}_\theta)$ for each time $t$, exhibits natural textures. This indicates that the reconstruction using the combination of Tweedie's formula and the DM network has a natural regularizing effect on the image.

By definition, the output of $f(\mathbf{x}_t; t, \boldsymbol{\epsilon}_\theta)$ in (4) represents the *denoised* version of $\mathbf{x}_t$ at time $t$ using the Tweedie's formula and the DM denoiser $\boldsymbol{\epsilon}_\theta$. Due to the implicit bias of $\boldsymbol{\epsilon}_\theta$, this denoised image tends to align with the clean image manifold, even if $\mathbf{x}_t$ does not correspond to a training image, as shown in columns 3 to 5 of Figure 2. We refer to this regularization effect of $f(\mathbf{x}_t; t, \boldsymbol{\epsilon}_\theta)$, which arises from network bias, as "network regularization".

By employing network regularization, we can address **(I1)** by ensuring that the data-consistent $\hat{\mathbf{x}}_0'$ is also network-consistent. We refer the latter condition as **Backward Consistency** and define it formally as follows.

**Definition 1** (Backward Consistency)**.** *We say an $\hat{\mathbf{x}}_0'$ is backward-consistent with Tweedie's formula and the DM neural network $\boldsymbol{\epsilon}_\theta$ at time $t$ if there exists some $\mathbf{v}_t$ such that $\hat{\mathbf{x}}_0' = f(\mathbf{v}_t; t, \boldsymbol{\epsilon}_\theta)$. In other words, backward consistency requires $\hat{\mathbf{x}}_0'$ to be a 'denoised version' of some noisy image $\mathbf{v}_t$ via the Tweedie-network denoiser $f$ at time $t$.*

The subset of images that are in the range of the function $f$ (i.e., backward-consistent) is denoted by $\mathcal{C}_t$ and defined as

$$\mathcal{C}_t := \{f(\mathbf{v}_t; t, \boldsymbol{\epsilon}_\theta) : \mathbf{v}_t \in \mathbb{R}^n\}\,. \tag{9}$$

Enforcing $\hat{\mathbf{x}}_0'$ to be both measurement- and backward-consistent involves solving the following optimization problem

$$\hat{\mathbf{x}}_0', \hat{\mathbf{v}}_t := \underset{\mathbf{v}_t', \mathbf{x}_0'}{\operatorname{argmin}} \left\{ \|\mathcal{A}(\mathbf{x}_0') - \mathbf{y}\|_2^2 \;\; \text{subject to} \;\; \mathbf{x}_0' = f(\mathbf{v}_t'; t, \boldsymbol{\epsilon}_\theta) \right\}. \tag{10}$$

However, (10) may violate forward consistency, as $\hat{\mathbf{v}}_t$ could possibly be far from $\mathbf{x}_t$. Therefore, we propose adding a regularization term, for which (10) becomes

$$\hat{\mathbf{x}}_0', \hat{\mathbf{v}}_t := \underset{\mathbf{v}_t', \mathbf{x}_0'}{\operatorname{argmin}} \left\{ \|\mathcal{A}(\mathbf{x}_0') - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}_t - \mathbf{v}_t'\|_2^2 \;\; \text{subject to} \;\; \mathbf{x}_0' = f(\mathbf{v}_t'; t, \boldsymbol{\epsilon}_\theta) \right\}. \tag{11}$$

During the reverse sampling process, at each time $t$, with the given $\mathbf{x}_t$, we seek a $\mathbf{v}_t'$ in the nearby region (i.e., $\|\mathbf{x}_t - \mathbf{v}_t'\|$ is small), such that $\mathbf{v}_t'$ can be denoised by $f$ to produce a clean image $\mathbf{x}_0'$ (i.e., $\mathbf{x}_0' = f(\mathbf{v}_t'; t, \boldsymbol{\epsilon}_\theta)$), which is also consistent with the measurements $\mathbf{y}$ (i.e., $\|\mathcal{A}(\mathbf{x}_0') - \mathbf{y}\|_2^2$ is small). We need to identify such a $\mathbf{v}_t'$ because $\mathbf{x}_t$ itself cannot be directly denoised by $f$ to yield an image consistent with the measurements. By substituting the constraint into the objective function, the optimization problem in (11) is reduced to

$$\hat{\mathbf{v}}_t := \underset{\mathbf{v}_t'}{\operatorname{argmin}} \left\{ \|\mathcal{A}\big(f(\mathbf{v}_t'; t, \boldsymbol{\epsilon}_\theta)\big) - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}_t - \mathbf{v}_t'\|_2^2 \right\}, \quad \hat{\mathbf{x}}_0' = f(\hat{\mathbf{v}}_t; t, \boldsymbol{\epsilon}_\theta). \tag{12}$$

The benefit of the considered backward consistency constraint is shown in columns 6 to 8 of Figure 2. After obtaining $\hat{\mathbf{x}}_0'$, the resampling formula in (8) is used to obtain $\mathbf{x}_{t-1}$.

### 3.3. Triple Consistency Conditions

We now summarize the three key conditions that apply at each sampling step.

(C1) **Measurement Consistency**: The reconstruction $\hat{\mathbf{x}}_0'$ is consistent with the measurements This means that $\mathcal{A}(\hat{\mathbf{x}}_0') \approx \mathbf{y}$.

(C2) **Backward Consistency**: The reconstruction $\hat{\mathbf{x}}_0'$ is a denoised image produced by the Tweedie-network denoiser $f$. More generally, we define the backward consistency to include any form of DM network regularization (e.g., using the DM probability-flow (PF) ODE [32]) applied to $\hat{\mathbf{x}}_0'$.

(C3) **Forward Consistency**: The pre-trained DM network $\boldsymbol{\epsilon}_\theta$ is provided with in-distribution inputs with high probability. To ensure this, we apply the resampling formula in (8) and enforce that $\hat{\mathbf{v}}_t$ remains close to $\mathbf{x}_t$.

We emphasize that **C1-C3** aim to ensure that all intermediate reconstructions $\hat{\mathbf{x}}_0'(\mathbf{x}_t)$ (with $t > 0$) are as accurate as possible, allowing us to effectively reduce the number of sampling steps. If reducing sampling steps is not necessary, these conditions become less critical, as the final reconstruction at $t = 0$ can still be accurate with a large number of sampling steps, even if the intermediate reconstructions are less precise.

Previous works, such as [12, 14], enforce measurement consistency by applying $\mathcal{A}(\hat{\mathbf{x}}_0) = \mathbf{y}$ exactly, whereas DPS [10] does not ensure consistency along the diffusion trajectory.

### 3.4. The Proposed Sampler

Given $\mathbf{x}_t$, $\boldsymbol{\epsilon}_\theta$, and towards satisfying the above conditions, our method, at sampling time $t$, consists of the following three steps:

$$\hat{\mathbf{v}}_t := \operatorname{argmin}_{\mathbf{v}_t'} \quad \|\mathcal{A}\Big(\tfrac{1}{\sqrt{\bar{\alpha}_t}}\left[\mathbf{v}_t' - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta(\mathbf{v}_t', t)\right]\Big) - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}_t - \mathbf{v}_t'\|_2^2 \tag{S1}$$

$$\hat{\mathbf{x}}_0' = f(\hat{\mathbf{v}}_t; t, \boldsymbol{\epsilon}_\theta) \equiv \tfrac{1}{\sqrt{\bar{\alpha}_t}}\left[\hat{\mathbf{v}}_t - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta(\hat{\mathbf{v}}_t, t)\right] \tag{S2}$$

$$\mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{\mathbf{x}}_0' + \sqrt{1-\bar{\alpha}_{t-1}}\boldsymbol{\eta}_t\,, \quad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})\,. \tag{S3}$$

The minimization in the first step optimizes over the input $\mathbf{v}_t'$ of the pre-trained diffusion model at time $t$, where the first term of the objective enforces measurement consistency for the posterior mean estimated image, satisfying condition **C1**. The second term serves as a regularization term, implicitly promoting closeness between $\hat{\mathbf{v}}_t$ and $\mathbf{x}_t$ (i.e., condition **C3**), with $\lambda > 0$ acting as the regularization parameter. The argument of the forward operator in (S1) and the second step in (S2) enforce that $\hat{\mathbf{v}}_t$ and $\hat{\mathbf{x}}_0'$, respectively, maintain the diffusion trajectory through obeying Tweedie's formula, thereby satisfying the backward consistency condition, **C2**. After obtaining the measurement-consistent estimate, $\hat{\mathbf{x}}_0'$, as given in (S2), it must be mapped back to time $t-1$ to generate $\mathbf{x}_{t-1}$. This is achieved through the forward diffusion step in (S3) as outlined in the forward consistency condition, **C3**. A diagram of SITCOM procedure is provided in Figure 3 (*left*).

**Remark 1.** *Obtaining the estimated image at time* 0 *given some* $\mathbf{x}_t$ *using the standard DM PF-ODE* [32] *is more accurate compared to the one-step Tweedie's formula. However, since PF-ODE is an iterative procedure, it requires more computational time. In SITCOM, PF-ODE could replace Tweedie's formula in* (S2). *Nevertheless, we chose not to use it, as this would increase the run time, and our empirical results are already highly competitive using Tweedie's formula.*

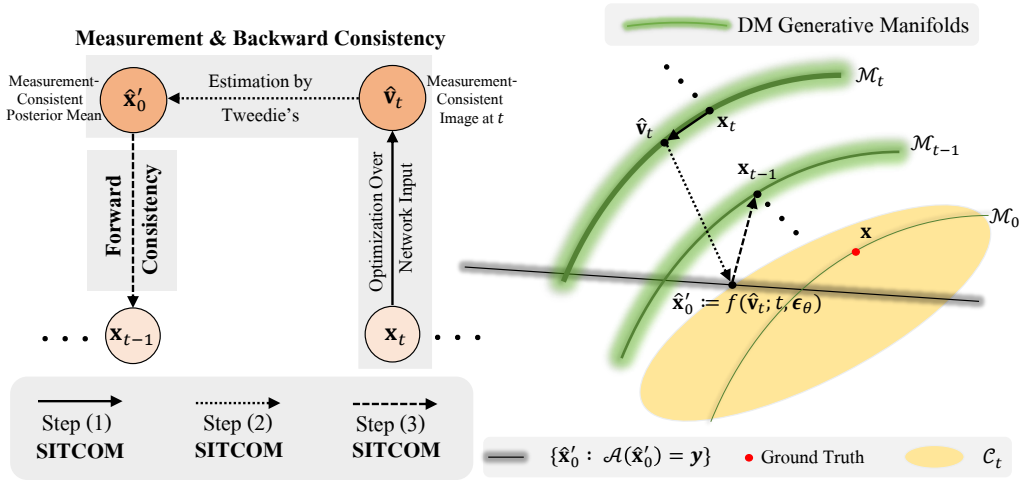


Figure 3: Illustrative diagram of the proposed procedure in SITCOM (*left*). Conceptual illustration of SITCOM, where $\mathcal{M}_t$ is the DM generative manifold at time $t$ and $\mathcal{C}_t$ is the subset of images that are backward-consistent, defined in (9) (*right*). Step (1) (solid arrow), Step (2) (dotted arrow), and Step (3) (dashed arrow) correspond to ($\mathtt{S}_1$), ($\mathtt{S}_2$), and ($\mathtt{S}_3$), respectively.

A conceptual illustration of SITCOM is shown in Figure 3 (*right*). The DM generative manifold, $\mathcal{M}_t$, is defined as the set of all $\mathbf{x}_t$ sampled from $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$, and $\mathbf{x}_0 \sim p_0(\mathbf{x})$. This set coincides with the entire space $\mathbb{R}^n$ equipped with the probability measure induced by the distribution of $\mathbf{x}_t$, which we denote as $\mathcal{P}_t$. In Figure 3 (*right*), the variation of color around each $\mathcal{M}_t$ indicates the concentration of the measure $\mathcal{P}_t$, with darker colors representing higher concentration. SITCOM's Step (1) and Step (2) enforce measurement consistency and backward consistency, thus map $\mathbf{x}_t$ to $\hat{\mathbf{x}}_0' = f(\hat{\mathbf{v}}_t; t, \boldsymbol{\epsilon}_\theta)$ which lies within the intersection of $(i)$ measurement-consistent set $\{\hat{\mathbf{x}}_0' : \mathcal{A}(\hat{\mathbf{x}}_0') \approx \mathbf{y}\}$ (the shaded black line) and $(ii)$ the backward-consistent set $\mathcal{C}_t$ (the yellow ellipsoid) defined in (9). Subsequently, $\mathbf{x}_{t-1}$ is generated by inserting $\hat{\mathbf{x}}_0'$ into the resampling formula, which enforces the forward consistency.

**Handling Measurement Noise:** To avoid the case where the first term of the objective in ($\mathtt{S}_1$) reaches small values yielding noise overfitting (i.e., when additive Gaussian noise in (1) is considered, $\sigma_{\mathbf{y}} > 0$), we propose refraining from enforcing strict measurement fitting $\mathcal{A}(\mathbf{x}) = \mathbf{y}$. Instead, we use the stopping criterion $\left\|\mathcal{A}\left(\frac{1}{\sqrt{\bar{\alpha}_t}}\left[\mathbf{v}_t' - \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}_\theta(\mathbf{v}_t', t)\right]\right) - \mathbf{y}\right\|_2^2 < \delta^2$ , where $\delta \in \mathbb{R}_+$ is a hyper-parameter that indicates the level of tolerance for noise and helps prevent overfitting. This is equivalent to enforcing an $\ell_2$ constraint, and is in spirit similar to [33]. Since the noise level cannot be accurately estimated, in our experiments, we use $\delta$ that is slightly larger than the actual level of noise in the measurements, i.e., $\delta > \sigma_{\mathbf{y}}\sqrt{m}$.

## 3.5. SITCOM with Arbitrary Stepsizes

In this subsection, we explain how to apply SITCOM with a large stepsize and present the final algorithm. The pre-trained DM is trained with $T$ diffusion steps. Given that our method is designed to satisfy measurement and diffusion consistency, SITCOM requires $N \ll T$ sampling iterations, using a step size of $\Delta t := \lfloor \frac{T}{N} \rfloor$. Thus, we introduce the index $i$ instead of $t$ with a relation $t = i\Delta t$.

The procedure of SITCOM is outlined in Algorithm 1. As inputs, SITCOM takes $\mathbf{y}$, $\mathcal{A}(\cdot)$, $\boldsymbol{\epsilon}_\theta$, the number of sampling steps $N$, $\bar{\alpha}_i$ for all $i \in \{1, \ldots, N\}$, the number of optimization steps $K$ per sampling step, stopping criteria $\delta$, and the learning rate $\gamma$.

Starting with initializing $\mathbf{v}_i^{(0)}$ as $\mathbf{x}_i$ (satisfying condition **C3**), lines 3 through 6 correspond to the first step of SITCOM, where ($\mathtt{S}_1$) is solved via either gradient descent (as shown in the algorithm), or the ADAM optimizer [34]. In lines 5 and 6, the stopping criterion is applied to prevent strict data fidelity (avoiding noise overfitting). Following the gradient updates in the inner loop, $\hat{\mathbf{v}}_i$ is obtained

**Algorithm 1 S**tep-w**i**se **T**riple-**Co**nsistent Sa**m**pling (**SITCOM**).

**Input**: Measurements $\mathbf{y}$, forward operator $\mathcal{A}(\cdot)$, pre-trained DM $\boldsymbol{\epsilon}_\theta(\cdot\,,\cdot)$, number of diffusion steps $N$, DM noise schedule $\bar{\alpha}_i$ for $i \in \{1,\dots,N\}$, number of gradient updates $K$, stopping criterion $\delta$, learning rate $\gamma$, and regularization parameter $\lambda$.
**Output**: Restored image $\hat{\mathbf{x}}$.

**Initialization**: $\mathbf{x}_N \sim \mathcal{N}(\mathbf{0},\mathbf{I})$, $\Delta t = \lfloor \frac{T}{N} \rfloor$

1: **For each** $i \in \{N, N-1,\dots,1\}$. (Reducing diffusion sampling steps)

2: **Initialize** $\mathbf{v}_i^{(0)} \leftarrow \mathbf{x}_i$. (Initialization to ensure Closeness: **C3** )

3: **For each** $k \in \{1,\dots,K\}$. (Gradient updates for measurement & backward consistency: **C1**, **C2**)

4: $\mathbf{v}_i^{(k)} = \mathbf{v}_i^{(k-1)} - \gamma \nabla_{\mathbf{v}_i} \Big[ \big\| \mathcal{A}\Big( \frac{1}{\sqrt{\bar{\alpha}_i}} \big[ \mathbf{v}_i - \sqrt{1-\bar{\alpha}_i}\, \boldsymbol{\epsilon}_\theta(\mathbf{v}_i, i\Delta t) \big] \Big) - \mathbf{y} \big\|_2^2 + \lambda \|\mathbf{x}_i - \mathbf{v}_i\|_2^2 \Big] \Big|_{\mathbf{v}_i = \mathbf{v}_i^{(k-1)}}$.

5: **If** $\big\| \mathcal{A}\big( \frac{1}{\sqrt{\bar{\alpha}_i}} \big[ \mathbf{v}_i^{(k)} - \sqrt{1-\bar{\alpha}_i}\, \boldsymbol{\epsilon}_\theta(\mathbf{v}_i^{(k)}, i\Delta t) \big] \big) - \mathbf{y} \big\|_2^2 < \delta^2$ . (Stopping criterion)

6: **Break** the **For** loop in step 3. (Preventing noise overfitting)

7: **Assign** $\hat{\mathbf{v}}_i \leftarrow \mathbf{v}_i^{(k)}$. (Backward diffusion consistency of $\hat{\mathbf{v}}_i$: **C2**)

8: **Obtain** $\hat{\mathbf{x}}_0' = f(\hat{\mathbf{v}}_i; t, \theta) = \frac{1}{\sqrt{\bar{\alpha}_i}} \big[ \hat{\mathbf{v}}_i - \sqrt{1-\bar{\alpha}_i}\, \boldsymbol{\epsilon}_\theta(\hat{\mathbf{v}}_i, i\Delta t) \big]$. (Backward consistency of $\hat{\mathbf{x}}_0'$: **C2**)

9: **Obtain** $\mathbf{x}_{i-1} = \sqrt{\bar{\alpha}_{i-1}}\hat{\mathbf{x}}_0' + \sqrt{1-\bar{\alpha}_{i-1}}\boldsymbol{\eta}_i,\ \boldsymbol{\eta}_i \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ . (Forward diffusion consistency: **C3**)

10: **Restored image:** $\hat{\mathbf{x}} = \mathbf{x}_0$.

in line 7, which is then used in line 8 to obtain $\hat{\mathbf{x}}_0'$ as specified in $(\mathtt{S}_2)$, satisfying condition **C2**. Note that line 8 requires no additional computation, as the $\hat{\mathbf{x}}_0'$ calculated here was already obtained while checking the stopping condition in line 6. After obtaining the double-consistent $\hat{\mathbf{x}}_0'$, the resampling is applied to map the image back to time $t-1$ while ensuring $\mathbf{x}_{t-1}$ to be in-distribution, as indicated in line 9 of the algorithm. In the next iteration, the requirement that $\hat{\mathbf{v}}_{t-1}$ is close to $\mathbf{x}_{t-1}$ ensures that the input $\hat{\mathbf{v}}_{t-1}$ to the DM network, $\boldsymbol{\epsilon}_\theta$, is also in-distribution, thus satisfying the forward-consistency (condition **C3**).

The computational requirements of SITCOM are determined by $(i)$ the number of sampling steps $N$ and $(ii)$ the number of gradient steps $K$ required for each sampling iteration. Given the proposed stopping criterion, this results in at most $NK$ Number of Function Evaluations (NFEs) of the pre-trained model (forward pass), $NK$ backward passes through the pre-trained model, and $NK$ applications each for the forward operator and its adjoint to solve the optimization problem in $(\mathtt{S}_1)$. With early stopping, the computational cost is lower. For example, for a linear operator $\mathcal{A}$ with dimensions $m \times n$, the cost of applying it (or its adjoint) to a vector is $\mathcal{O}(mn)$. For a network with width $M$ and depth $L$, the cost for making a forward pass is $\mathcal{O}(LM^2)$. The gradients are computed w.r.t. the input of the DM network, requiring an additional backward pass. This backward pass has the same computational cost as the forward pass. Consequently, this procedure is significantly more efficient than network training, where the network weights are updated instead of the input.

## 3.6. Relation with Existing Approaches

While SITCOM and DPS [10] both use Tweedie's formula, there are two major differences. First, DPS does not enforce backward consistency. Specifically, it only considers one gradient descent step of the optimization in $(\mathtt{S}_1)$, whereas our method perform multiple steps, initializing with $\mathbf{x}_t$. Second, DPS does not enforce the forward diffusion consistency, namely, it does not use resampling $(\mathtt{S}_3)$. This means that DPS does not enforce a step-wise **C1**-**C3**.

Both SITCOM and the works in [12, 14] are optimization-based methods that modify the sampling steps to enforce measurement consistency, and both involve mapping back to time $t-1$ (as in step 3 of SITCOM). However, there is a major difference between them: The optimization variable in these works is the estimated image at time $t$ (the output of the DM network), whereas in SITCOM, it is the noisy image at time $t$ (the input of the network). This means that these studies enforce **C1** and **C3**, but not **C2**.

# 4. Experimental Results

**Tasks:** Our experimental setup for IPs and noise levels used largely follows DPS [10]. For linear IPs, we evaluate five tasks: super resolution, Gaussian deblurring, motion deblurring, box inpainting, and random inpainting. For Gaussian deblurring and motion deblurring, we use $61\times61$ kernels with standard deviations of 3 and 0.5, respectively. In the super-resolution task, a bicubic resizer downscales images by a factor of 4. For box inpainting, a random $128\times128$ box is applied to mask image pixels, and for random inpainting, the mask is generated with each pixel masked with a probability of 0.7, as described in [14]. For nonlinear IP tasks, we consider three tasks: phase retrieval, high dynamic range (HDR) reconstruction, and nonlinear (non-uniform) deblurring. For phase retrieval, an oversampling rate of 2 is applied in frequency domain, and we report the best result out of four independent samples, consistent with [10, 12] (see Appendix D for more discussion on phase retrieval). In HDR reconstruction, the goal is to restore a higher dynamic range image from a lower dynamic range image (with a factor of 2). Nonlinear deblurring follows the setup in [35]. For measurement noise, we use $\sigma_{\mathbf{y}} \in \{0.01, 0.05\}$ for all tasks.

**Baselines & Datasets:** For baselines, in this section, we use DPS [10], DDNM [25], DCDP [1], and DAPS [12]. The selection criteria is based on these baselines' competitive performance on several linear and non-linear inverse problems under measurement noise. Additionally, we provide comparison results with three other baselines in Table 3 of Appendix E. We evaluate SITCOM and baselines using 100 test images from the validation set of FFHQ [36] and 100 test images from the validation set of ImageNet [37] for which the FFHQ-trained and ImageNet-trained DMs are given in [10] and [38], respectively, following the previous convention. For evaluation metrics, we use PSNR, SSIM [39], and LPIPS [40].

**SITCOM Settings:** For Algorithm 1, we set $N = 20$ and $K = 30$ for most tasks. We show the impact of $N$ and $K$ in Appendix F.1. The parameter $\lambda$ is set to 0 for all tasks other than phase retrieval where we use $\lambda = 1$, following the ablation study in Appendix F.2. The impact of the stopping criterion under the noisy setting is given in Appendix F.3. The learning rate for $(\mathtt{S}_1)$ is set to $\gamma = 0.01$ across all measurements noise levels, datasets, and tasks. Table 6 in Appendix F.4 lists all the hyper-parameters used for every task. We note that the exact set of hyper-parameters is used for the FFHQ and ImageNet datasets. Our code is available online[3].

**Main Results:** In Table 1, we present the quantitative results in terms of the average PSNR, SSIM, LPIPS, and run-time (minutes). Columns 3 to 6 correspond to the FFHQ dataset, while columns 7 to 10 reflect results for the ImageNet dataset. The table covers 8 tasks, 4 evaluation metrics, and 2 datasets, totaling 64 results. Among these, SITCOM reports the best performance in 58 out of 64 cases.

On average, SITCOM demonstrates strong reconstruction capabilities across most tasks. For the FFHQ dataset, SITCOM reports a PSNR improvement of over 1 dB in Super Resolution, random Inpainting, and Gaussian Deblurring compared to the second-best method. On ImageNet, we observe more than a 1 dB improvement in random In-painting. Other than ImageNet Gaussian Deblurring and ImageNet Phase Retrieval, for which we under-perform by 0.66 dB and 0.31 dB, respectively, our PSNR improvement when compared to the second-best results are less than 1 dB. However, in terms of run-time, SITCOM consistently requires less computational time across all tasks. For FFHQ, SITCOM is over $3\times$ faster in Box In-painting and motion Deblurring, and more than $2\times$ faster in the remaining tasks, whereas on ImageNet, the run-time improvement ranges from 36 seconds (for HDR) to 62.4 seconds (for Super Resolution), when compared to DPS, DDNM, and DAPS.

For linear tasks, SITCOM requires slightly less run-time than DCDP on both datasets. However, across the two datasets, SITCOM achieves PSNR improvements of more than 1 dB, 2 dB, and 3 dB for the tasks of super resolution, box in-painting, and random in-painting (and Gaussian Deblurring), respectively, as compared to DCDP. For non-linear tasks, SITCOM not only provides PSNR improvements over DCDP but also significantly reduces run-time.

[3] https://github.com/sjames40/SITCOM

| Task | Method | FFHQ PSNR (↑) | FFHQ SSIM (↑) | FFHQ LPIPS (↓) | FFHQ Run-time (↓) | ImageNet PSNR (↑) | ImageNet SSIM (↑) | ImageNet LPIPS (↓) | ImageNet Run-time (↓) |
|---|---|---|---|---|---|---|---|---|---|
| Super Resolution 4× | DPS | 24.44±0.56 | 0.801±0.032 | 0.26±0.022 | 1.26±0.52 | 23.86±0.34 | 0.76±0.041 | 0.357±0.069 | 2.38±1.02 |
| | DAPS | 29.24±0.42 | 0.851±0.024 | **0.135**±0.039 | 1.24 ±0.22 | 25.67±0.73 | 0.802±0.045 | 0.256±0.067 | 2.16±0.45 |
| | DDNM | 28.02±0.78 | 0.842±0.034 | 0.197±0.034 | 1.07±0.42 | 23.96±0.89 | 0.767±0.045 | 0.475±0.044 | 1.27±0.55 |
| | DCDP | 27.88±1.34 | 0.825±0.07 | 0.211±0.05 | 0.52±0.34 | 24.12±1.24 | 0.772±0.000 | 0.351±0.00 | 1.45±0.00 |
| | SITCOM (ours) | **30.68**±1.02 | **0.867**±0.045 | 0.142±0.056 | **0.45**±0.58 | **26.35**±1.21 | **0.812**±0.021 | **0.232**±0.038 | **1.12**±0.52 |
| Box In-Painting | DPS | 23.20±0.89 | 0.754±0.023 | 0.196±0.032 | 1.57±0.55 | 19.78±0.78 | 0.691±0.052 | 0.312±0.025 | 2.28 ±1.02 |
| | DAPS | 24.17±1.02 | 0.787±0.032 | 0.135±0.032 | 1.35±0.45 | 21.43±0.40 | 0.736±0.020 | 0.218±0.021 | 2.54±1.02 |
| | DDNM | 24.37±0.45 | 0.792±0.024 | 0.232±0.026 | 1.02±0.032 | 21.64±0.66 | 0.732±0.028 | 0.319±0.015 | 1.45±1.02 |
| | DCDP | 23.66±1.67 | 0.762±0.07 | 0.144±0.05 | 0.56±0.25 | 20.45±1.22 | 0.712±0.07 | 0.298±0.04 | 1.127±0.25 |
| | SITCOM (ours) | **24.68**±0.78 | **0.801**±0.042 | **0.121**±0.08 | **0.35**±0.25 | **21.88**±0.92 | **0.742**±0.032 | **0.214**±0.021 | **1.12**±0.35 |
| Random In-Painting | DPS | 28.39±0.82 | 0.844±0.042 | 0.194±0.021 | 1.52±0.30 | 24.26±0.42 | 0.772±0.02 | 0.326±0.034 | 2.27±0.25 |
| | DAPS | 31.02±0.45 | 0.902±0.015 | 0.098±0.017 | 1.56±0.40 | 28.44±0.45 | 0.872±0.024 | 0.135±0.052 | 2.14±0.45 |
| | DDNM | 29.93±0.67 | 0.889±0.032 | 0.122±0.056 | 1.45±0.35 | 29.22±0.55 | 0.912±0.034 | 0.191±0.048 | 1.54±0.52 |
| | DCDP | 28.59±0.95 | 0.852±0.06 | 0.202±0.04 | 0.55±0.25 | 26.22±1.13 | 0.791±0.06 | 0.289±0.03 | 1.44±0.34 |
| | SITCOM (ours) | **32.05**±1.02 | **0.909**±0.09 | **0.095**±0.025 | **0.45**±0.50 | **29.60**±0.78 | **0.915**±0.028 | **0.127**±0.039 | **1.14**±0.45 |
| Gaussian Deblurring | DPS | 25.52±0.78 | 0.826±0.052 | 0.211±0.017 | 1.50±0.50 | 21.86±0.45 | 0.772±0.08 | 0.362±0.034 | 2.55±0.45 |
| | DAPS | 29.22±0.50 | 0.884±0.056 | 0.164±0.032 | 1.40±0.52 | 26.12±0.78 | 0.832±0.092 | 0.245±0.022 | 2.23±0.52 |
| | DDNM | 28.22±0.52 | 0.867±0.056 | 0.216±0.042 | 1.56±0.45 | **28.06**±0.52 | **0.879**±0.072 | 0.278±0.089 | 1.75±0.63 |
| | DCDP | 26.67±0.78 | 0.835±0.08 | 0.196±0.04 | 0.56±0.23 | 23.24±1.18 | 0.781±0.06 | 0.343±0.04 | 1.34±0.43 |
| | SITCOM (ours) | **30.25**±0.89 | **0.892**±0.032 | **0.135**±0.078 | **0.46**±0.25 | 27.40±0.45 | 0.854±0.045 | **0.236**±0.039 | **1.10**±0.42 |
| Motion Deblurring | DPS | 23.40±1.42 | 0.737±0.024 | 0.270±0.025 | 2.40±0.55 | 21.86±2.05 | 0.724±0.022 | 0.357±0.032 | 2.56±0.40 |
| | DAPS | 29.66±0.50 | 0.872±0.027 | 0.157±0.012 | 1.86±0.12 | 27.86±1.20 | 0.862±0.032 | 0.196±0.021 | 2.3±0.45 |
| | SITCOM (ours) | **30.34**±0.67 | **0.902**±0.037 | **0.148**±0.041 | **0.5**±0.45 | **28.65**±0.34 | **0.876**±0.021 | **0.189**±0.036 | **1.48**±0.35 |
| Phase Retrieval | DPS | 17.34±2.67 | 0.67±0.045 | 0.41±0.08 | 1.50±0.34 | 16.82±1.22 | 0.64±0.08 | 0.447±0.032 | 2.17±0.24 |
| | DAPS | 30.67±3.12 | 0.908±0.041 | 0.122±0.084 | 1.34±0.78 | **25.76**±2.33 | 0.797±0.045 | 0.255±0.095 | 2.24±0.25 |
| | DCDP | 28.52±2.50 | 0.892±0.19 | 0.167±0.92 | 3.30±0.45 | 24.25±2.25 | 0.778±0.14 | 0.287±0.089 | 3.49±0.52 |
| | SITCOM (ours) | **30.97**±3.10 | **0.915**±0.064 | **0.112**±0.102 | **0.52**±0.34 | 25.45±2.78 | **0.808**±0.065 | **0.246**±0.088 | **1.40**±0.40 |
| Non-Uniform Deblurring | DPS | 23.42±2.15 | 0.757±0.042 | 0.279±0.067 | 1.55±0.44 | 22.57±0.67 | 0.778±0.067 | 0.310±0.102 | 2.35±0.45 |
| | DAPS | 28.23±1.55 | 0.833±0.052 | 0.155±0.041 | 1.42±0.41 | 27.65±1.2 | 0.822±0.056 | 0.169±0.044 | 2.14±0.45 |
| | DCDP | 28.78±1.44 | 0.827±0.08 | 0.162±0.04 | 3.30±0.45 | 26.56±1.09 | 0.803±0.06 | 0.182±0.05 | 3.70±0.36 |
| | SITCOM (ours) | **30.12**±0.68 | **0.902**±0.042 | **0.145**±0.037 | **0.52**±0.45 | **28.78**±0.79 | **0.832**±0.056 | **0.16**±0.048 | **1.25**±0.45 |
| High Dynamic Range | DPS | 22.88±1.25 | 0.722±0.056 | 0.264±0.089 | 1.45±0.34 | 19.33±1.45 | 0.688±0.067 | 0.503±0.132 | 2.42±0.46 |
| | DAPS | 27.12±0.89 | 0.825±0.056 | 0.166±0.078 | 1.25±0.35 | 26.30±1.02 | 0.792±0.046 | 0.177±0.089 | 2.18±0.55 |
| | SITCOM (ours) | **27.98**±1.06 | **0.832**±0.052 | **0.158**±0.032 | **0.52**±0.30 | **26.97**±0.87 | **0.821**±0.045 | **0.167**±0.052 | **1.54**±0.35 |

Table 1: Average PSNR, SSIM, LPIPS, and run-time (minutes) of SITCOM and baselines using 100 test images from the **FFHQ** dataset (columns 3 to 7) and 100 test images from the **ImageNet** dataset with a **measurement noise level of** $\sigma_{\mathbf{y}} = 0.05$. The results for the $\sigma_{\mathbf{y}} = 0.01$ case are given in Table 2 of Appendix E. The first five tasks are linear, while the last three tasks are non-linear (underlined). For each task and dataset combination, the best results are bolded, and the second-best results are underlined. Values after ± represent the standard deviation. All results were obtained using a **single RTX5000 GPU** machine. For phase retrieval, the run-time is reported for the best result out of four independent runs. This is applied for SITCOM and baselines. More discussion about phase retrieval is given in Appendix D.

In summary, the results in Table 1 demonstrate that SITCOM either provides a notable improvement in restoration quality (e.g., cases where we report PSNR improvements of over 1 dB) or delivers comparable results to the baselines, all while significantly reducing computation time.

In Appendix E, we present the results with $\sigma_{\mathbf{y}} = 0.01$ case (Table 2). Additionally, Table 3 includes quantitative results for three more baselines. In addition to the FFHQ restored images in Figure 1, we also provide additional samples from both datasets in the figures found in Appendix H.

# 5. Conclusion

In this paper, we proposed three conditions to achieve measurement- and diffusion-consistent trajectories for linear and non-linear inverse imaging problems using diffusion models (DMs) as priors. These conditions form the basis of our unique optimization-based sampling method, which optimizes the input of the diffusion model at each step. This approach allows for greater control over the diffusion process and enhances data consistency with the given measurements. Through extensive experiments across eight image restoration tasks, we evaluated the effectiveness of our method. The results showed that our sampler consistently delivers improved or comparable quantitative performance against state-of-the-art baselines, even with measurement noise. Notably, our method is efficient, requiring significantly less run-time than leading baselines, making it practical for real-world applications.

# References


[1] Xiang Li, Soo Min Kwon, Ismail R Alkhouri, Saiprasad Ravishankar, and Qing Qu. Decoupled data consistency with diffusion purification for image restoration. *arXiv preprint arXiv:2403.06054*, 2024.

[2] Charles Byrne. A unified treatment of some iterative algorithms in signal processing and image reconstruction. *Inverse problems*, 20(1):103, 2003.

[3] Ismail Alkhouri, Shijun Liang, Rongrong Wang, Qing Qu, and Saiprasad Ravishankar. Diffusion-based adversarial purification for robust deep mri reconstruction. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12841–12845. IEEE, 2024.

[4] Aviad Levis, Pratul P Srinivasan, Andrew A Chael, Ren Ng, and Katherine L Bouman. Gravitationally lensed black hole emission tomography. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19841–19850, 2022.

[5] Noori BniLam and Rafid Al-Khoury. Parameter identification algorithm for ground source heat pump systems. *Applied energy*, 264:114712, 2020.

[6] Saiprasad Ravishankar, Jong Chul Ye, and Jeffrey A Fessler. Image reconstruction: From sparsity to data-adaptive methods and machine learning. *Proceedings of the IEEE*, 108(1):86–109, 2019.

[7] Victor Lempitsky, Andrea Vedaldi, and Dmitry Ulyanov. Deep image prior. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9446–9454. IEEE, 2018.

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[9] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021.

[10] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

[11] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[12] Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. *arXiv preprint arXiv:2407.01521*, 2024.

[13] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022.

[14] Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In *The Twelfth International Conference on Learning Representations*, 2023.

[15] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning*, pages 32211–32252. PMLR, 2023.

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

[17] Brian DO Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[19] Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *arXiv preprint arXiv:2410.00083*, 2024.

[20] Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[21] Siyi Chen, Huijie Zhang, Minzhe Guo, Yifu Lu, Peng Wang, and Qing Qu. Exploring low-dimensional subspaces in diffusion models for controllable image editing. *arXiv preprint arXiv:2409.02374*, 2024.

[22] Calvin Luo. Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*, 2022.

[23] Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alexandros G Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *arXiv preprint arXiv:2307.00619*, 2023.

[24] Hyungjin Chung, Jeongsol Kim, and Jong Chul Ye. Direct diffusion bridge using data consistency for inverse problems. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 7158–7169. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/165b0e600b1721bd59526131eb061092-Paper-Conference.pdf`.

[25] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*, 2022.

[26] Morteza Mardani, Jiaming Song, Jan Kautz, and Arash Vahdat. A variational perspective on solving inverse problems with diffusion models. *arXiv preprint arXiv:2305.04391*, 2023.

[27] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Animashree Anandkumar. Diffusion models for adversarial purification. In *International Conference on Machine Learning*, pages 16805–16827. PMLR, 2022.

[28] 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.

[29] Shijun Liang, Evan Bell, Qing Qu, Rongrong Wang, and Saiprasad Ravishankar. Analysis of deep image prior and exploiting self-guidance for image reconstruction. *arXiv preprint arXiv:2402.04097*, 2024.

[30] Avrajit Ghosh, Xitong Zhang, Kenneth K Sun, Qing Qu, Saiprasad Ravishankar, and Rongrong Wang. Optimal eye surgeon: Finding image priors through sparse generators at initialization. In *Forty-first International Conference on Machine Learning*, 2024.

[31] Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. 2023.

[32] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

[33] Hengkang Wang, Xu Zhang, Taihui Li, Yuxiang Wan, Tiancong Chen, and Ju Sun. Dmplug: A plug-in method for solving inverse problems with diffusion models. *arXiv preprint arXiv:2405.16749*, 2024.

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[35] Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11956–11965, 2021.

[36] 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.

[37] 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. Ieee, 2009.

[38] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[39] 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.

[40] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[41] Ce Liu, W.T. Freeman, R. Szeliski, and Sing Bing Kang. Noise estimation from a single image. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 901–908, 2006. doi: 10.1109/CVPR.2006.207.

[42] Guangyong Chen, Fengyuan Zhu, and Pheng Ann Heng. An efficient statistical method for image noise level estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 477–485, 2015.

[43] Hyungjin Chung, Jeongsol Kim, Sehui Kim, and Jong Chul Ye. Parallel diffusion models of operator and image for blind inverse problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6059–6069, 2023.

[44] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606, 2022.

[45] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play admm for image restoration: Fixed-point convergence and applications. *IEEE Transactions on Computational Imaging*, 3 (1):84–98, 2016.

# Appendix

In the Appendix, we start by showing the equivalence between the second formula in (3) and (6) (Appendix A). Then, we discuss the known limitations and future extensions of SITCOM (Appendix B). Subsequently, we present experiments to highlight the impact of the proposed backward consistency (Appendix C). This is followed by a discussion on phase retrieval (Appendix D). In Appendix E, we provide further comparison results, and in Appendix F, we perform ablation studies to examine the effects of the stopping criterion and other components/hyper-parameters in SITCOM. Appendix G covers the implementation details of tasks and baselines, followed by examples of restored images (Appendix H).

## A. Derivation of (6)

From [22], we have

$$\mathbf{s}_\theta(\mathbf{x}_t, t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\,. \tag{13}$$

Rearranging the Tweedie's formula in (4) to solve for $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$ yields

$$\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t)}{\sqrt{1-\bar{\alpha}_t}}\,. \tag{14}$$

Now, we substitute into the recursive equation for $\mathbf{x}_{t-1}$:

$$\begin{aligned}
\mathbf{x}_{t-1} &= \frac{1}{\sqrt{1-\beta_t}}\left[\mathbf{x}_t + \beta_t \mathbf{s}_\theta(\mathbf{x}_t, t)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (15)\\
&= \frac{1}{\sqrt{1-\beta_t}}\left[\mathbf{x}_t + \beta_t\left(-\frac{1}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (16)\\
&= \frac{1}{\sqrt{1-\beta_t}}\left[\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (17)\\
&= \frac{1}{\sqrt{1-\beta_t}}\left[\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\left(\frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t)}{\sqrt{1-\bar{\alpha}_t}}\right)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (18)\\
&= \frac{1}{\sqrt{1-\beta_t}}\left[\mathbf{x}_t - \frac{\beta_t}{1-\bar{\alpha}_t}\left(\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t)\right)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (19)\\
&= \frac{1}{\sqrt{1-\beta_t}}\left[\left(1 - \frac{\beta_t}{1-\bar{\alpha}_t}\right)\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_t}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t)\right] + \sqrt{\beta_t}\boldsymbol{\eta}_t && (20)\\
&= \frac{(1-\bar{\alpha}_t-\beta_t)}{\sqrt{1-\beta_t}\,(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_t}\beta_t}{\sqrt{1-\beta_t}\,(1-\bar{\alpha}_t)}\hat{\mathbf{x}}_0(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t && (21)\\
&= \frac{(\alpha_t-\bar{\alpha}_t)}{\sqrt{\alpha_t}\,(1-\bar{\alpha}_t)}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_t}\beta_t}{\sqrt{\alpha_t}\,(1-\bar{\alpha}_t)}\hat{\mathbf{x}}_0(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t && (22)\\
&= \frac{\left(\sqrt{\alpha_t} - \sqrt{\alpha_t}\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t && (23)\\
&= \frac{\sqrt{\alpha_t}\,(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0(\mathbf{x}_t) + \sqrt{\beta_t}\boldsymbol{\eta}_t\,, && (24)
\end{aligned}$$

which is equivalent to the second formula in (3).

## B. Limitations & Future Work

In SITCOM, the stopping criterion parameter is set slightly higher than the level of measurement noise, determined by $\sigma_{\mathbf{y}}$. As a result, our method requires access to (or estimation of) the measurement noise prior to the restoration process. Knowledge of noise level is also assumed in other works

such as DAPS [12]. In practice, classical approaches, such as [41, 42], can be used to estimate the noise.

Additionally, the stated conditions and proposed sampler are limited to the non-blind setting, as SITCOM assumes full access to the forward model, unlike works such as [43], which perform both image restoration and forward model estimation.

For future work, in addition to addressing the aforementioned limitations, we aim to extend SITCOM to the latent space and explore its applicability in medical image reconstruction.

# C. Impact of the proposed Backward Consistency

Here, we demonstrate the impact of the proposed backward diffusion consistency in SITCOM using two experiments.

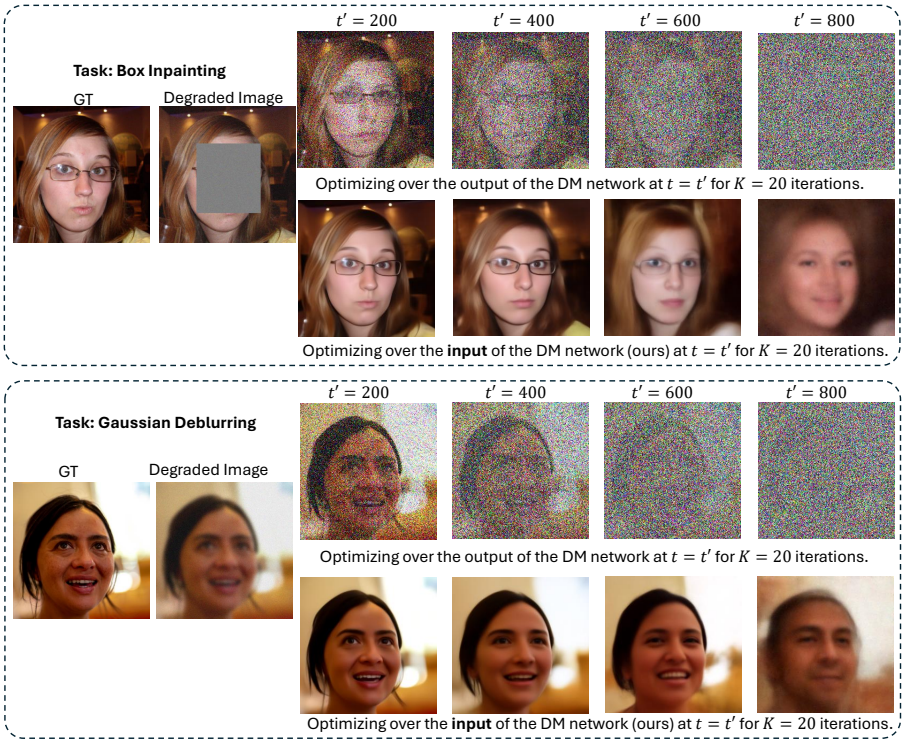


Figure 4: Results of applying optimization-based measurement consistency, for which the optimization variable is the DM output (resp. input), are shown in the first (resp. second) row for each task: Box Inpainting (*top*) and Gaussian Deblurring (*bottom*).

First, for the box-painting task, we compare optimizing over the input to the DM (as in SITCOM) with optimizing over the output of the DM network (as is done in DCDP [1] and DAPS [12]) at time steps $t' \in \{200, 400, 600\}$. For each case (selection of $t'$), we start from $t = T$ and run SITCOM with a step size of $\lfloor \frac{T}{N} \rfloor$. At $t = t'$, given $\mathbf{x}_{t'}$, we perform two separate optimizations with intializing the optimization variable as $\mathbf{x}_{t'}$: one iteratively over the DM network input (ours) and another iteratively over the DM network output (i.e., (5) but without the regularization), both running until convergence (i.e., when the loss stops decreasing). For our approach, the result of the optimization from ($\mathtt{S}_1$) is used as input to Tweedie's formula in ($\mathtt{S}_2$) to compute the posterior mean $\hat{\mathbf{x}}_0' = \hat{\mathbf{x}}_0(\mathbf{v}_t)$. For the case of optimizing over the DM output, we use (5) without regularization. Figure 2 shows the results at different time steps. The consistency between the ground truth and the unmasked regions of the estimated images suggest the convergence of the measurement consistency. As observed, SITCOM produces significantly less artifacts in the masked region when compared to optimizing over the output. This is evident both at earlier time steps ($t' = 600$) and later steps ($t' = 400$ and $t' = 200$).

For the second experiment, the goal is to show that SITCOM requires much smaller number of optimization steps to remove the noise as compared to the case where the optimization variable is the output of the DM network. The results are given in Figure 4, where we repeat the above experiment with two tasks: Box-inpainting (*top*) and Gaussian Deblurring (*bottom*), this time using a fixed number of optimization steps for both SITCOM, and optimizing over the DM output. Specifically, we run SITCOM from $t = T$ to $t = t' + 1$. Then, we apply $K = 20$ iterations (the setting in SITCOM) in $(\mathtt{S}_1)$, and $K = 20$ when optimizing (5) (without regularization) where measurement noise is $\sigma_{\mathbf{y}} = 0.05$. As shown, compared to optimizing over the DM output, SITCOM significantly reduces noise across all considered $t'$, underscoring the effect of the proposed backward diffusion consistency when optimizing over the DM input.

## D. Discussion on Phase Retrieval

As discussed in our experimental results section, for the phase retrieval task, we report the best results from 4 independent runs, following the convention in [10, 12]. For the phase retrieval results of Table 1 and Table 2 (given in Appendix E), we use this approach across all baselines where the run-time is reported for one run.


Figure 5: Results of Phase Retrieval on two images (top row) from the FFHQ dataset. Rows 2, 3, and 4 correspond to the results of DPS, DAPS, and SITCOM (ours), respectively.

The forward model for phase retrieval is adopted from DPS where the inverse problem is generally more challenging compared to other image restoration tasks. This increased difficulty arises from the presence of multiple modes that can yield the same measurements [12].

In Figure 5, we present two examples comparing SITCOM, DPS, and DAPS. For each ground truth image, we show four results from which the best one was selected. In the first column, SITCOM avoids significant artifacts, while DAPS produces one image rotated by 180 degrees. In the second column, both SITCOM and DAPS exhibit one run with severe artifacts. However, the last image from SITCOM does exhibit more artifacts compared to the second worst-case result from DAPS. Additionally, the DPS results show severe perceptual differences in both cases, with artifacts being particularly noticeable in the second column.

# E. Additional Comparison Results

In Table 2, we present the average PSNR, SSIM, LPIPS, and run-time (minutes) of DPS, DAPS, DDNM, and SITCOM using the FFHQ and ImageNet datasets for which the measurement noise level is set to $\sigma_{\mathbf{y}} = 0.01$ (different from Table 1). The goal of these results is to evaluate our method and baselines under less noisy settings.

| Task | Method | FFHQ | | | | ImageNet | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | PSNR (↑) | SSIM (↑) | LPIPS (↓) | Run-time (↓) | PSNR (↑) | SSIM (↑) | LPIPS (↓) | Run-time (↓) |
| Super Resolution 4× | DPS | 25.20±1.22 | 0.806±0.044 | 0.242±0.102 | 1.31±0.44 | 24.45±0.89 | 0.792±0.052 | 0.331±0.089 | 2.33±0.40 |
| | DAPS | 29.6±0.67 | 0.871±0.034 | **0.132**±0.088 | 1.24±0.43 | 25.98±0.74 | 0.794±0.09 | 0.234±0.089 | 2.10±1.02 |
| | DDNM | 28.82±0.67 | 0.851±0.043 | 0.188±0.13 | 1.07±0.35 | 24.67±0.78 | 0.771±0.06 | 0.432±0.34 | 1.38±0.55 |
| | Ours | **30.95**±0.89 | **0.872**±0.045 | 0.137±0.046 | **0.50**±0.34 | **26.89**±0.86 | **0.802**±0.057 | **0.224**±0.056 | **1.34**±0.45 |
| Box In-Painting | DPS | 23.56±0.78 | 0.762±0.034 | 0.191±0.087 | 1.52±0.43 | 20.22±0.67 | 0.69±0.034 | 0.297±0.077 | 1.55±0.44 |
| | DAPS | 24.41±0.67 | 0.791±0.034 | 0.129±0.067 | 1.33±0.42 | 21.79±0.34 | 0.734±0.045 | 0.214±0.034 | 2.44±0.34 |
| | DDNM | 24.67±0.067 | 0.788±0.024 | 0.229±0.055 | 1.02±0.42 | 21.99±0.54 | 0.737±0.034 | 0.315±0.022 | 1.42±0.45 |
| | Ours | **24.97**±0.55 | **0.804**±0.045 | **0.118**±0.022 | **0.37**±0.34 | **22.23**±0.44 | **0.745**±0.034 | **0.208**±0.023 | **1.23**±0.44 |
| Random In-Painting | DPS | 28.77±0.56 | 0.847±0.034 | 0.191±0.023 | 1.55±0.34 | 24.57±0.45 | 0.775±0.023 | 0.318±0.26 | 2.12±0.30 |
| | DAPS | 31.56±0.45 | 0.905±0.013 | 0.094±0.012 | 1.42±0.45 | 28.86±0.67 | 0.877±0.021 | 0.131±0.044 | 2.01±0.34 |
| | DDNM | 30.56±0.56 | 0.902±0.013 | 0.116±0.023 | 1.25±0.42 | 30.12±0.45 | 0.917±0.012 | 0.124±0.032 | 1.89±0.23 |
| | Ours | **33.02**±0.44 | **0.919**±0.012 | **0.0912**±0.013 | **0.47**±0.34 | **30.67**±0.45 | **0.918**±0.013 | **0.118**±0.012 | **1.40**±0.34 |
| Gaussian Deblurring | DPS | 25.78±0.68 | 0.831±0.034 | 0.202±0.014 | 1.33±0.44 | 22.45±0.42 | 0.778±0.067 | 0.344±0.041 | 2.12±0.44 |
| | DAPS | 29.67±0.45 | 0.889±0.045 | 0.163±0.033 | 2.15±0.37 | 26.34±0.55 | 0.836±0.034 | 0.244±0.023 | 2.22±0.43 |
| | DDNM | 28.56±0.45 | 0.872±0.024 | 0.211±0.034 | 1.24±0.34 | **28.44**±0.021 | 0.882±0.021 | 0.267±0.00 | 1.76±0.33 |
| | Ours | **32.12**±0.34 | **0.913**±0.024 | **0.139**±0.045 | **0.45**±0.25 | 28.22±0.45 | **0.891**±0.014 | **0.216**±0.021 | **1.34**±0.25 |
| Motion Deblurring | DPS | 23.78±0.78 | 0.742±0.042 | 0.265±0.024 | 1.65±0.34 | 22.33±0.727 | 0.726±0.034 | 0.352±0.00 | 2.21±0.40 |
| | DAPS | 30.78±0.56 | 0.892±0.034 | 0.146±0.023 | 1.44±0.34 | 28.24±0.62 | 0.867±0.023 | 0.191±0.017 | 2.12±0.44 |
| | Ours | **32.34**±0.44 | **0.908**±0.028 | **0.135**±0.028 | **0.52**±0.34 | **29.12**±0.38 | **0.882**±0.025 | **0.182**±0.025 | **1.45**±0.31 |
| Phase Retrieval | DPS | 17.56±2.15 | 0.681±0.056 | 0.392±0.021 | 1.52±0.42 | 16.77±1.78 | 0.651±0.076 | 0.442±0.037 | 2.18±0.38 |
| | DAPS | 31.45±2.78 | 0.909±0.035 | 0.109±0.044 | 1,85±0.32 | **26.12**±2.12 | 0.802±0.023 | 0.247±0.034 | 2.32±0.35 |
| | Ours | **31.88**±2.89 | **0.921**±0.067 | **0.102**±0.078 | **0.54**±0.45 | 25.76±1.78 | **0.813**±0.032 | **0.238**±0.067 | **1.31**±0.45 |
| Non-Uniform Deblurring | DPS | 23.78±2.23 | 0.761±0.051 | 0.269±0.064 | 1.56±0.45 | 22.97±1.57 | 0.781±0.023 | 0.302±0.089 | 2.34±0.44 |
| | DAPS | 28.89±1.67 | 0.845±0.057 | 0.150±0.056 | 1.41±0.37 | 28.02±1.15 | 0.831±0.082 | 0.162±0.034 | 2.23±0.56 |
| | Ours | **31.09**±0.89 | **0.911**±0.056 | **0.132**±0.45 | **0.56**±0.37 | **29.56**±0.78 | **0.844**±0.045 | **0.147**±0.042 | **1.34**±0.44 |
| High Dynamic Range | DPS | 23.33±1.34 | 0.734±0.049 | 0.251±0.078 | 1.34±0.42 | 19.67±0.056 | 0.693±0.034 | 0.498±0.112 | 2.34±0.41 |
| | DAPS | 27.58±0.829 | 0.828±0.00 | 0.161±0.067 | 1.26±0.44 | 26.71±0.088 | 0.802±0.032 | 0.172±0.066 | 2.12±0.32 |
| | Ours | **28.52**±0.89 | **0.844**±0.045 | **0.148**±0.035 | **0.51**±0.42 | **27.56**±0.78 | **0.825**±0.037 | **0.162**±0.046 | **1.45**±0.41 |

Table 2: Average PSNR, SSIM, LPIPS, and run-time (minutes) of SITCOM and baselines using 100 test images from FFHQ and 100 test images from ImageNet with a **measurement noise level** of $\sigma_{\mathbf{y}} = 0.01$. The first five tasks are linear, while the last three tasks are non-linear (underlined). For each task and dataset combination, the best results are bolded, and the second-best results are underlined. Values after ± represent the standard deviation. All results were obtained using a **single RTX5000 GPU** machine. For phase retrieval, the run-time is reported for the best result out of four independent runs. This is applied for SITCOM and baselines.

Overall, we observe similar trends to those discussed in Section 4 for Table 1. On the FFHQ dataset, SITCOM achieves higher average PSNR values compared to the baselines across all tasks, with improvements exceeding 1 dB in 5 out of 8 tasks. For the ImageNet dataset, we observe more than 1 dB improvement on the non-linear deblurring task, while for the remaining tasks, the improvement is less than 1 dB, except for Gaussian deblurring (where SITCOM underperforms by 0.22 dB) and phase retrieval (underperforming by 0.36 dB).

In terms of run-time, generally, SITCOM significantly outperforms DDNM, DPS, and DAPS, with all methods evaluated on a single RTX5000 GPU. For the FFHQ dataset, SITCOM is at least twice as fast when compared to baselines. On ImageNet, SITCOM consistently requires much less run-time compared to DPS and DAPS. When compared to DDNM, SITCOM's run-time is similar or slightly lower. For example, on the super-resolution task, both SITCOM and DDNM average 1.34 minutes, but SITCOM achieves over a 2 dB improvement.

In Table 3, we report the average PSNR and LPIPS results using three more baselines: Denoising Diffusion Restoration Models (DDRM) [44], Plug-and-Play (PnP) ADMM [45] (a non diffusion-based solver), and Regularization by Denoising with Diffusion (RED-Diff) [26]. The results of DDRM, PnP-ADMM, and RED-Diff are sourced from [12]. DDRM and PnP-ADMM present results for linear tasks whereas RED-Diff is used for the non-linear tasks. The results of SITCOM are as reported in Table 1.

| Task | Method | FFHQ | | ImageNet | |
|---|---|---|---|---|---|
| | | PSNR (↑) | LPIPS (↓) | PSNR (↑) | LPIPS (↓) |
| Super Resolution 4× | DDRM [44] | 27.65 | 0.210 | 25.21 | 0.284 |
| | PnP-ADMM [45] | 23.48 | 0.725 | 22.18 | 0.724 |
| | SITCOM (ours) | **30.68** | **0.142** | **26.35** | **0.232** |
| Box In-Painting | DDRM [44] | 22.37 | 0.159 | 19.45 | 0.229 |
| | PnP-ADMM [45] | 13.39 | 0.775 | 12.61 | 0.702 |
| | SITCOM (ours) | **24.68** | **0.121** | **21.88** | **0.214** |
| Random In-Painting | DDRM [44] | 25.75 | 0.218 | 23.23 | 0.325 |
| | PnP-ADMM [45] | 20.94 | 0.724 | 20.03 | 0.680 |
| | SITCOM (ours) | **32.05** | **0.095** | **29.60** | **0.127** |
| Gaussian Deblurring | DDRM [44] | 23.36 | 0.236 | 23.86 | 0.341 |
| | PnP-ADMM [45] | 21.31 | 0.751 | 20.47 | 0.729 |
| | SITCOM (ours) | **30.25** | **0.235** | **27.40** | **0.236** |
| Motion Deblurring | PnP-ADMM [45] | 23.40 | 0.703 | 24.23 | 0.684 |
| | SITCOM (ours) | **30.34** | **0.148** | **28.65** | **0.189** |
| Phase Retrieval | RED-Diff [26] | 15.60 | 0.596 | 14.98 | 0.536 |
| | SITCOM (ours) | **30.97** | **0.112** | **25.45** | **0.246** |
| Non-Uniform Deblurring | RED-Diff [26] | **30.86** | 0.160 | **30.07** | 0.211 |
| | SITCOM (ours) | 30.12 | **0.145** | 28.78 | **0.160** |
| High Dynamic Range | RED-Diff [26] | 22.16 | 0.258 | 22.03 | 0.274 |
| | SITCOM (ours) | **27.98** | **0.158** | **26.97** | **0.167** |

Table 3: Average PSNR and LPIPS results of our method and other baselines over 100 FFHQ and 100 ImageNet test images. The measurement noise setting is $\sigma_{\mathbf{y}} = 0.05$. The results of DDRM and PnP-ADMM (resp. RED-Diff) are sourced from Tables 1 and 3 (resp. 2 and 4) in [12]. The remaining results are as given in Table 1 of Section 4.

When compared to DDRM and PnP-ADMM, SITCOM demonstrates notable improvements in both PSNR and LPIPS across all tasks and datasets. For instance, SITCOM achieves over a 5 dB improvement in random in-painting on both datasets. Compared to RED-Diff, SITCOM outperforms by 5 dB on FFHQ and more than 10 dB on ImageNet for phase retrieval. A similar trend is observed in the High Dynamic Range task. For non-linear non-uniform deblurring, although SITCOM performs better in terms of LPIPS, it reports approximately 1 dB (FFHQ) and 2 dB (ImageNet) less PSNR than RED-Diff, all without requiring external denoisers.

# F. Ablation Studies

## F.1. Effect of the number of Optimization steps $K$, & the number of Sampling steps $N$

In this subsection, we perform an ablation study on the number of optimization steps, $K$, and the number of sampling steps, $N$. Specifically, for the tasks of Super Resolution, Motion Deblurring, and Random In-painting, we run SITCOM using combinations from $N \in \{10, 20, 30\}$ and $K \in \{20, 30, 40\}$. The average PSNR results over 20 test images from the FFHQ dataset are presented in Table 4. As shown, for the first three tasks, SITCOM consistently achieves strong PSNR scores across all $(N, K)$ pairs, demonstrating that its performance is not very sensitive to variations in $(N, K)$ within these ranges as the results vary by nearly 1 dB. For High Dynamic Range tasks, we observe that the best results are obtained with $(N, K) = (20, 40)$. The selected $(N, K)$ values for our main results are listed in Table 6 of Appendix F.4.

## F.2. Effect of the Regularization Parameter $\lambda$

In this subsection, we perform an ablation study to assess the impact of the regularization parameter, $\lambda$, in SITCOM. Table 5 shows the results across four tasks using various $\lambda$ values. Aside from phase retrieval, the effect of $\lambda$ is minimal. We hypothesize that initializing the optimization variable in

| $(N, K)$ | $(10, 20)$ | $(10, 30)$ | $(10, 40)$ | $(20, 20)$ | $(20, 30)$ | $(20, 40)$ | $(30, 20)$ | $(30, 30)$ | $(30, 40)$ |
|---|---|---|---|---|---|---|---|---|---|
| Super Resolution 4× | 29.654 | 29.771 | 29.815 | 29.913 | 29.952 | 29.961 | 30.009 | 30.027 | 30.033 |
| Motion Deblurring | 29.976 | 30.820 | 31.264 | 31.259 | 31.380 | 30.452 | 31.282 | 30.624 | 30.438 |
| Random Inpainting | 33.428 | 34.444 | 34.699 | 34.546 | 34.558 | 34.574 | 34.619 | 34.634 | 34.639 |
| High Dynamic Range | 25.902 | 26.290 | 27.873 | 26.957 | 27.104 | 27.874 | 27.171 | 27.127 | 26.806 |

Table 4: Effect of the number of sampling steps $(N)$ and optimization steps per sampling iteration $(K)$ on the tasks listed in the first column for SITCOM. The reported PSNR values are averaged over 20 FFHQ test images.

$(\mathtt{S}_1)$ with $\mathbf{x}_t$ is sufficient to enforce forward diffusion consistency in **C3**. Therefore, we set $\lambda = 1$ for phase retrieval and $\lambda = 0$ for the other tasks.

Additionally, for all tasks other than phase retrieval, we observed that when $\lambda = 0$, the restored images exhibit enhanced high-frequency details. For visual examples, see the results of $\lambda = 0$ versus $\lambda = 1$ in Figure 6.

| $\lambda$ | 0 | 0.05 | 0.5 | 1 | 1.5 |
|---|---|---|---|---|---|
| Super Resolution 4× | 29.952 | 29.968 | 29.464 | 29.550 | 29.288 |
| Motion Deblurring | 31.380 | 31.393 | 31.429 | 31.382 | 31.150 |
| Random Inpainting | 34.559 | 34.537 | 34.523 | 34.500 | 34.301 |
| Phase Retrieval | 31.678 | 31.892 | 32.221 | 32.342 | 32.124 |

Table 5: Ablation Study on the impact of the regularization parameter $\lambda$.

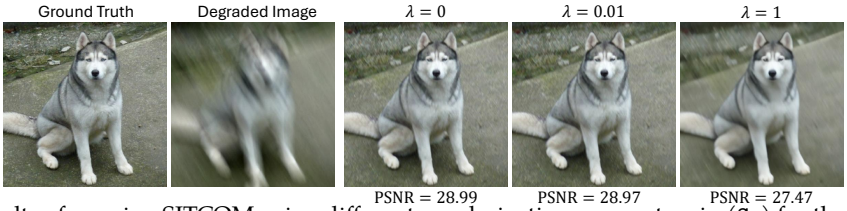


Figure 6: Results of running SITCOM using different regularization parameters in $(\mathtt{S}_1)$ for the task of Motion deblurring.

## F.3. Impact of the Stopping criterion For Noisy Measurements

In this subsection, we demonstrate the impact of applying the stopping criterion in SITCOM when handling measurement noise. For the tasks of super resolution and motion deblurring, we run SITCOM with and without the stopping criterion for the case of $\sigma_{\mathbf{y}} = 0.05$. The results are presented in Figure 7. As shown, for both tasks, using the stopping criterion (i.e., $\delta > 0$) not only improves PSNR values compared to the case of $\delta = 0$, but also visually reduces additive noise in the restored images. This is because, without the stopping criterion , the measurement consistency enforced by the optimization in $(\mathtt{S}_1)$ tends to fit the noise in the measurements.

## F.4. Complete List of hyper-parameters in SITCOM

Table 6 summarizes the hyper-parameters used for each task in our experiments, as determined by the ablation studies in the previous subsections. Notably, the same set of hyper-parameters is applied to both the FFHQ and ImageNet datasets.

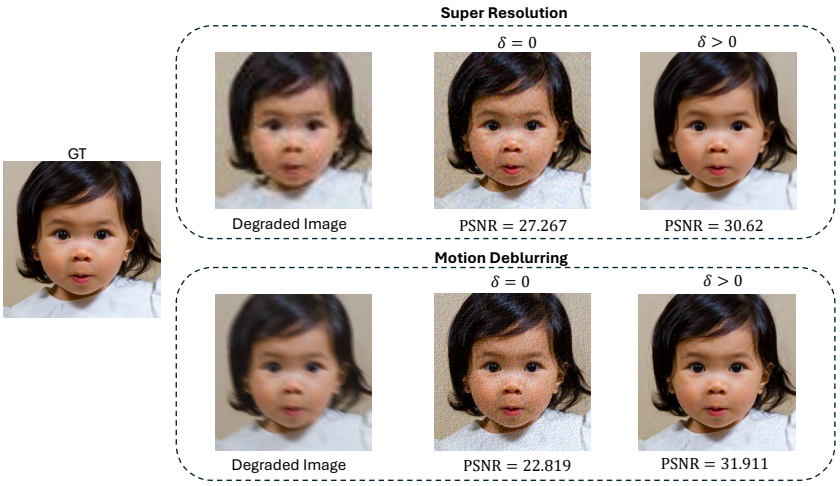


Figure 7: Impact of the stopping criterion in preventing noise overfitting. For the most right column, $\delta$ is set as in Table 6.

| **Task** | Sampling Steps $N$ | Optimization Steps $K$ | Regularization $\lambda$ | Stopping criterion $\delta$ for $\sigma_{\mathbf{y}} \in \{0.05, 0.01\}$ |
|---|---|---|---|---|
| Super Resolution 4× | 20 | 20 | 0 | $\{0.051\sqrt{m_{\text{SR}}}, 0.011\sqrt{m_{\text{SR}}}\}$ |
| Box In-Painting | 20 | 20 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |
| Random In-Painting | 20 | 30 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |
| Gaussian Deblurring | 20 | 30 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |
| Motion Deblurring | 20 | 30 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |
| Phase Retrieval | 20 | 30 | 1 | $\{0.051\sqrt{m_{\text{PR}}}, 0.011\sqrt{m_{\text{PR}}}\}$ |
| Non-Uniform Deblurring | 20 | 30 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |
| High Dynamic Range | 20 | 40 | 0 | $\{0.051\sqrt{m}, 0.011\sqrt{m}\}$ |

Table 6: Hyper-parameters of SITCOM for every task considered in this paper. The same set of hyper-parameters is used for FFHQ and ImageNet. The learning rate in Algorithm 1 is set to $\gamma = 0.01$ for all tasks, datasets, and measurement noise levels. For the stopping criterion column, $m_{\text{SR}} = 64\times64\times3$, $m = 256\times256\times3$, and $m_{\text{PR}} = 384 \times 384 \times 3$.

# G. Detailed Implementation of tasks and Baselines

The forward models of all tasks are adopted from DPS. We refer the reader to Appendix B of [10] for details. For baselines, we used the codes provided by the authors of each paper: DPS[4], DDNM[5], DAPS[6], and DCDP[7]. Default configurations are used for each task.

# H. Qualitative results

Figure 8 presents results with SITCOM, DPS, and DAPS using ImageNet. See also Figure 9, Figure 10, Figure 11, and Figure 12 for more images.

[4] https://github.com/DPS2022/diffusion-posterior-sampling

[5] https://github.com/wyhuai/DDNM

[6] https://github.com/zhangbingliang2019/DAPS

[7] https://github.com/Morefre/Decoupled-Data-Consistency-with-Diffusion-Purification-for-Image-Restoration

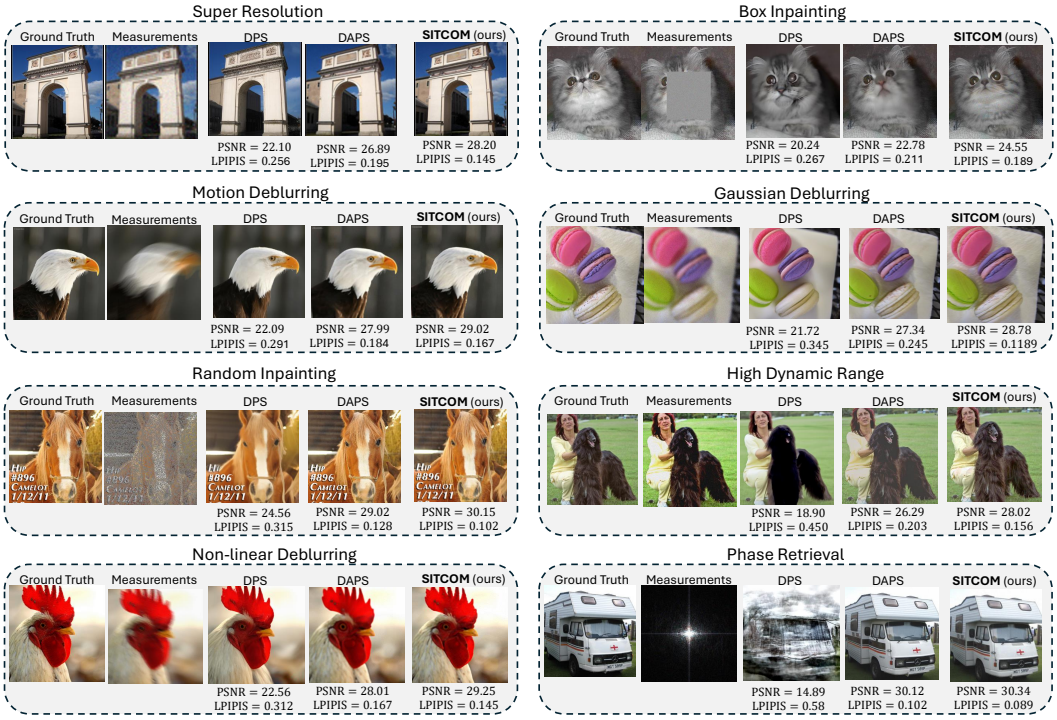


Figure 8: **Qualitative results on the ImageNet dataset** for five linear tasks and three non-linear tasks under measurement noise of $\sigma_{\mathbf{y}} = 0.05$. The PSNR and LPIPS values are given below each restored image.

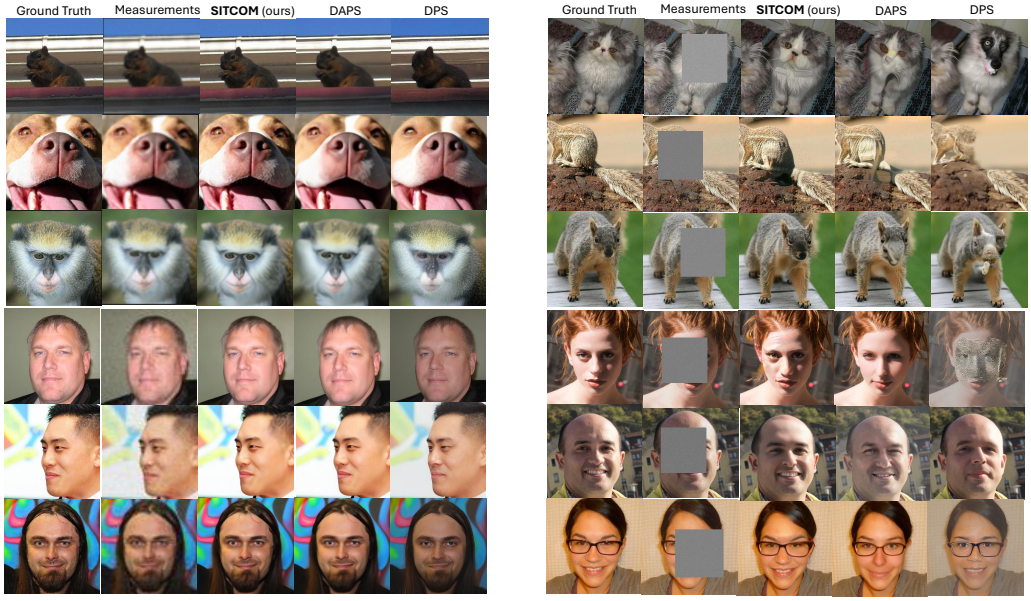


Figure 9: **Super resolution** (*left*) and **box inpainting** (*right*) results. First (resp. last) three rows are for the FFHQ (resp. ImageNet) dataset.


Figure 10: **Motion deblurring** (*left*) and **Gaussian deblurring** (*right*) results. First (resp. last) three rows are for the FFHQ (resp. ImageNet) dataset.


Figure 11: **Random inpainting** (*left*) and **non-linear (non-uniform) deblurring** (*right*) results. First (resp. last) three rows are for the FFHQ (resp. ImageNet) dataset.

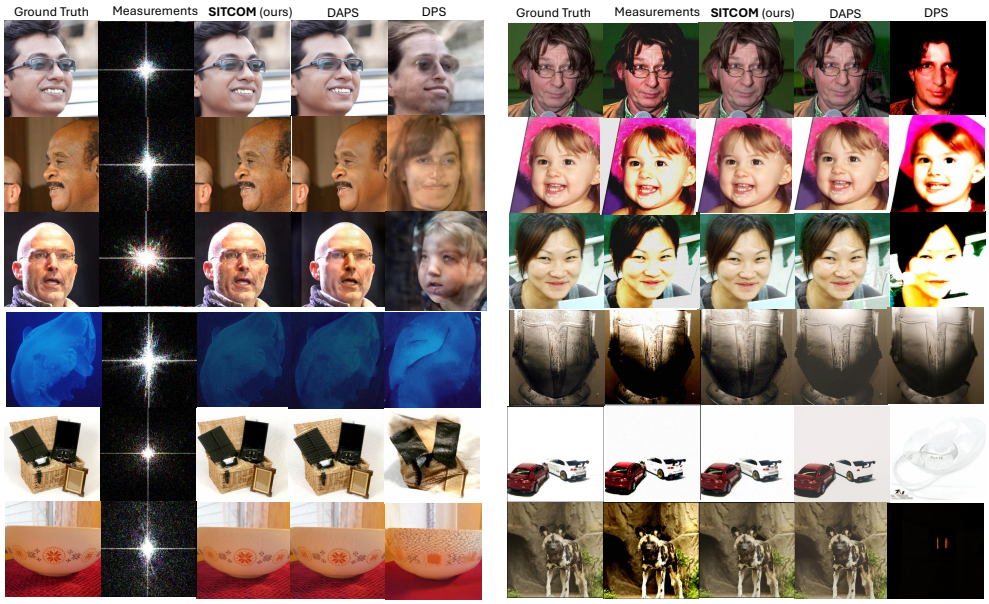


Figure 12: **Phase retrieval** (*left*) and **high dynamic range** (*right*) results. First (resp. last) three rows are for the FFHQ (resp. ImageNet) dataset.