Adapt and Diffuse: Sample-adaptive Reconstruction via Latent Diffusion Models

Zalan Fabian [*1] **Berk Tinaz** [*1] **Mahdi Soltanolkotabi** [1]

Abstract

Inverse problems arise in a multitude of applications, where the goal is to recover a clean signal from noisy and possibly (non)linear observations. The difficulty of a reconstruction problem depends on multiple factors, such as the ground truth signal structure, the severity of the degradation and the complex interactions between the above. This results in natural sample-by-sample variation in the difficulty of a reconstruction problem. Our key observation is that most existing inverse problem solvers lack the ability to adapt their compute power to the difficulty of the reconstruction task, resulting in subpar performance and wasteful resource allocation. We propose a novel method, *severity encoding*, to estimate the degradation severity of corrupted signals in the latent space of an autoencoder. We show that the estimated severity has strong correlation with the true corruption level and can provide useful hints on the difficulty of reconstruction problems on a sample-by-sample basis. Furthermore, we propose a reconstruction method based on latent diffusion models that leverages the predicted degradation severities to fine-tune the reverse diffusion sampling trajectory and thus achieve sample-adaptive inference times. Our framework, Flash-Diffusion, acts as a wrapper that can be combined with any latent diffusion-based baseline solver, imbuing it with sample-adaptivity and acceleration. We perform experiments on both linear and nonlinear inverse problems and demonstrate that our technique greatly improves the performance of the baseline solver and achieves up to $10\times$ acceleration in mean sampling speed[1].

[*]Equal contribution [1]Dept. of Electrical and Comp. Eng., Univ. of Southern California, Los Angeles, CA. Correspondence to: Zalan Fabian <zfabian@usc.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

[1]Code is available at https://github.com/z-fabian/flash-diffusion

1. Introduction

Inverse problems arise in a multitude of computer vision (Ledig et al., 2017; Wang et al., 2018), biomedical imaging (Sriram et al., 2020) and scientific (Hand et al., 2018) applications, where the goal is to recover a clean signal from noisy and degraded observations. As information is fundamentally lost in the process, structural assumptions on the clean signal are needed to make recovery possible. Traditional compressed sensing (Candes et al., 2006) approaches utilize explicit regularizers that encourage sparsity of the reconstruction in transformation domains. More recently, data-driven deep learning methods have established new state-of-the-art in tackling most inverse problems (see an overview in (Ongie et al., 2020)).

A key shortcoming of available techniques is their inherent inability to adapt their compute power allocation to the difficulty of reconstructing a given corrupted sample. There is a natural sample-by-sample variation in the difficulty of recovery due to multiple factors. First, variations in the measurement process (e. g. more or less additive noise, different blur kernels) greatly impact the difficulty of reconstruction. Second, a sample can be inherently difficult to reconstruct for the particular model, if it is different from examples seen in the training set (out-of-distribution samples). Third, the amount of information loss due to the interaction between the specific sample and the applied degradation can vary vastly. For instance, applying a blur kernel to an image consisting of high-frequency textures destroys significantly more information than applying the same kernel to a smooth image. Finally, the implicit bias of the model architecture towards certain signal classes (e.g. piece-wise constant or smooth for convolutional architectures) can be a key factor in determining the difficulty of a recovery task. Therefore, expending the same amount of compute to reconstruct all examples is potentially wasteful, especially on datasets with varied corruption parameters.

Sample-adaptive methods that incorporate the difficulty of a reconstruction problem, or *the severity of degradation*, and allocate compute effort accordingly are thus highly desired. To the best of our knowledge, such methods have not been studied extensively in the literature. *Unrolled networks* (Zhang & Ghanem, 2018; Sun et al., 2016) have

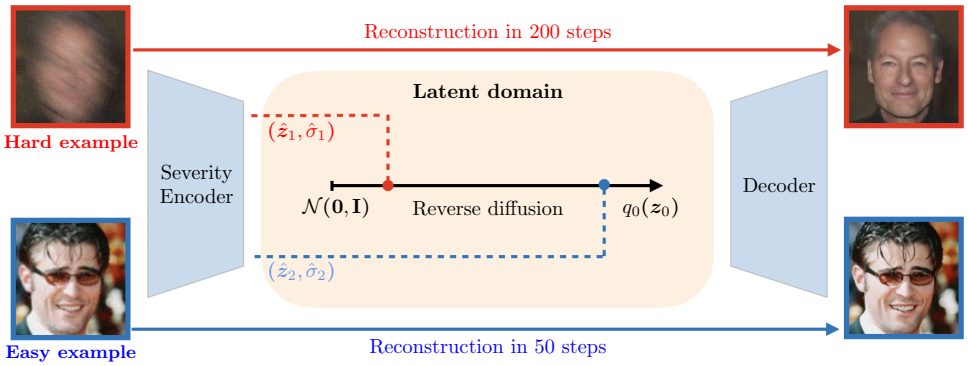


Figure 1: **Overview of our method**: we estimate the degradation severity of corrupted images in the latent space of an autoencoder (Severity Encoder). We leverage the severity predictions ($\hat{\sigma}$) to find the optimal start time in a latent reverse diffusion process on a sample-by-sample basis. As a result, inference cost is automatically scaled by the difficulty of the reconstruction task at test time.

been proposed for reconstruction, that map the iterations of popular optimizers to learnable submodules, where deeper networks can be used to tackle more challenging reconstruction tasks. However, network size is determined in training time and therefore these methods are unable to adapt on a sample-by-sample basis. Deep Equilibrium Models have been proposed to solve inverse problems (Gilton et al., 2021) by training networks of arbitrary depth through the construction of fixed-point iterations. These methods can adapt their compute in test time by scaling the number of iterations to convergence, however it is unclear how the optimal number of iterations correlates with degradation severity.

Diffusion models have established new state-of-the-art performance in synthesizing data of various modalities (Dhariwal & Nichol, 2021; Nichol et al., 2021; Ramesh et al., 2022; Rombach et al., 2022; Saharia et al., 2022b; Ho et al., 2022a; Saharia et al., 2022a; Ho et al., 2022b; Kong et al., 2020), inverse problem solving and image restoration (Kadkhodaie & Simoncelli, 2021; Saharia et al., 2021; Song et al., 2021b; Chung & Ye, 2022; Chung et al., 2022c;a;b; Kawar et al., 2021; 2022a;b; Fabian et al., 2023). Diffusion-based sampling techniques generate the missing information destroyed by the corruption step-by-step through a diffusion process that transforms pure noise into a target distribution. Recent work (Chung et al., 2022c) has shown that the sampling trajectory can be significantly shortened by starting the reverse diffusion process from a good initial reconstruction, instead of pure noise. However, this approach treats the noise level of the starting manifold as a hyperparameter independent of degradation severity. Therefore, even though sampling is accelerated, the same number of function evaluations are required to reconstruct any sample.

More recently, latent domain diffusion, that is a diffusion process defined in the low-dimensional latent space of a pre-trained autoencoder, has demonstrated great success in image synthesis (Rombach et al., 2022) and has been successfully applied to solving inverse problems (Rout et al., 2024; Song et al., 2023; Chung et al., 2023) and in high-resolution (Luo et al., 2023) and real-world (Jiang et al., 2023) image restoration. Latent diffusion has the clear benefit of improved efficiency due to the reduced dimensionality of the problem leading to faster sampling. In addition to this, the latent space consists of compressed representations of relevant information in data and thus provides a natural space to quantify the loss of information due to image corruptions, which strongly correlates with the difficulty of the reconstruction task.

In this paper, we propose a novel reconstruction framework (Figure 1), where the cost of inference is automatically scaled based on the difficulty of the reconstruction task on a sample-by-sample basis. Our contributions are as follows:

- We propose a novel method that we call *severity encoding*, to estimate the degradation severity of noisy, degraded images in the latent space of an autoencoder. We show that the estimated severity has strong correlation with the true corruption level and can give useful hints at the difficulty of reconstruction problems on a sample-by-sample basis. Training the severity encoder is efficient, as it can be done by fine-tuning a pre-trained encoder.
- We propose a technique for solving general noisy inverse problems based on latent diffusion models that leverages the predicted degradation severities to fine-tune the reverse diffusion sampling trajectory and thus achieve sample-adaptive inference times. Our framework is extremely flexible: it acts as a wrapper that can be combined with any latent diffusion-based inverse problem solver, imbuing it with sample-adaptivity and acceleration. We call our method Flash-Diffusion: Fast

Latent Sample-Adaptive Reconstruction ScHeme.

- We perform in-depth numerical experiments on multiple datasets and on both linear and nonlinear noisy inverse problems. We demonstrate that (1) Flash-Diffusion greatly improves the reconstruction performance of the baseline solver, especially in case of degradations with highly varying severity, and (2) Flash-Diffusion accelerates the baseline solver by up to $10\times$ without any loss in reconstruction quality.

2. Background

Diffusion models – Diffusion in the context of generative modeling consists of transforming a clean data distribution $\boldsymbol{x}_0 \sim q_0(\boldsymbol{x})$ through a forward noising process, defined over $0 \leq t \leq T$, $t \in \mathbb{R}$, into some tractable distribution q_T. Typically, q_t is chosen such that $\boldsymbol{x}_t$ is obtained from $\boldsymbol{x}_0$ via adding *i.i.d.* Gaussian noise, that is $q_t(\boldsymbol{x}_t|\boldsymbol{x}_0) \sim \mathcal{N}(\boldsymbol{x}_0, \sigma_t^2\mathbf{I})$, where σ_t^2 is from a known variance schedule. Diffusion models (DMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song & Ermon, 2020a;b) learn to reverse the forward corruption process in order to generate data samples starting from a simple Gaussian distribution. The forward process can be described as an Itô stochastic differential equation (SDE) (Song et al., 2020) $\mathrm{d}\boldsymbol{x} = f(\boldsymbol{x}, t)\mathrm{d}t + g(t)\mathrm{d}\boldsymbol{w}$, where $\boldsymbol{w} \in \mathbb{R}^n$ is the standard Wiener process. The forward SDE is reversible (Anderson, 1982) and can be simulated given $\nabla_{\boldsymbol{x}} \log q_t(\boldsymbol{x})$, which is referred to as the *score* of the data distribution. The score is typically approximated by a neural network $s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ and learned from data. After discretizing the reverse SDE, a variety of discrete time sampling algorithms can be derived.

Denoising Diffusion Probabilistic Models (DDPMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020) are obtained from the discretization of the variance preserving SDE with $f(\boldsymbol{x}, t) = -\frac{1}{2}\beta_t\boldsymbol{x}$ and $g(t) = \sqrt{\beta_t}$, where β_t is a predefined variance schedule that is a strictly increasing function of t. One can sample from the corresponding forward diffusion process at any time step i as $\boldsymbol{x}_i = \sqrt{\bar{\alpha}_i}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_i}\boldsymbol{\varepsilon}$, with $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $\bar{\alpha}_i = \prod_{j=1}^{i} \alpha_i$, $\alpha_i = 1 - \beta_i$. By minimizing the denoising score-matching objective

$$L_{DM} = \mathbb{E}_{\varepsilon, i, \boldsymbol{x}_i \sim q_0(\boldsymbol{x}_0) q_i(\boldsymbol{x}_i|\boldsymbol{x}_0)} \left[\left\| \varepsilon_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i) - \varepsilon \right\|^2 \right]$$

the score model $\varepsilon_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i)$ learns to predict the noise on the input corrupted signal.

Latent Diffusion Models (LDMs) – LDMs (Rombach et al., 2022) aim to mitigate the computational burden of traditional DMs by running diffusion in a low-dimensional latent space of an autoencoder. In particular, an encoder $\mathcal{E}_0$ is trained to extract a compressed representation $\boldsymbol{z} \in \mathbb{R}^d$, $d << n$ of the input signal $\boldsymbol{x}$ in the form $\boldsymbol{z} = \mathcal{E}_0(\boldsymbol{x})$. To recover the clean signal from the latent representation $\boldsymbol{z}$, a decoder $\mathcal{D}_0$ is trained such that $\mathcal{D}_0(\mathcal{E}_0(\boldsymbol{x})) \approx \boldsymbol{x}$. A score model that progressively denoises $\boldsymbol{z}$ can be trained in the latent space of the autoencoder via the objective following the DDPM framework. The final generated image can be obtained by passing the denoised latent through $\mathcal{D}_0$.

Diffusion models for solving inverse problems – Solving a general noisy inverse problem amounts to finding the clean signal $\boldsymbol{x} \in \mathbb{R}^n$ from a noisy and degraded observation $\boldsymbol{y} \in \mathbb{R}^m$ in the form

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

where $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^m$ denotes a deterministic degradation and $\boldsymbol{n} \sim \mathcal{N}(\mathbf{0}, \sigma_y^2\mathbf{I})$ is Gaussian noise. As information is fundamentally lost in the measurement process, structural assumptions on clean signals are necessary to recover $\boldsymbol{x}$. Deep learning approaches can learn a generative model $p_\theta(\boldsymbol{x})$ that represents the underlying structure of clean data and can be leveraged as a prior to solve (1). In particular, the posterior over clean data can be written as $p_\theta(\boldsymbol{x}|\boldsymbol{y}) \propto p_\theta(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x})$, where the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$ is represented by (1). Thus, one can sample from the posterior by querying the generative model. The score of the posterior can be written as

$$\nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}|\boldsymbol{y}) = \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) + \nabla_{\boldsymbol{x}} \log p(\boldsymbol{y}|\boldsymbol{x}),$$

where the first term corresponds to an unconditional score model trained to predict noise on the signal without any information about the forward model $\mathcal{A}$. The score of the likelihood term however is challenging to estimate in general. Various approaches have emerged to incorporate the data acquisition model into a diffusion process, including projection-based approaches (Song et al., 2021b; Chung & Ye, 2022; Chung et al., 2022c), restricting updates to stay on a given manifold (Chung et al., 2022b), spectral approaches (Kawar et al., 2022a), or methods that tailor the diffusion process to the degradation (Welker et al., 2022; Fabian et al., 2023; Delbracio & Milanfar, 2023). Chung et al. (2022a) proposes Diffusion Posterior Sampling (DPS) that approximates the gradient of the likelihood as $\nabla_{\boldsymbol{x}_i} \log p(\boldsymbol{y}|\boldsymbol{x}_i) \approx \nabla_{\boldsymbol{x}_i} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i))$, where $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i)$ is the posterior mean estimate obtained from the score model.

More recently, a flurry of activity has emerged to deploy LDMs for inverse problem solving. The most straightforward approach is to perform posterior sampling in latent space via the DPS approximation

$$\nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\boldsymbol{z}_i) \approx \nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))),$$

which can be viewed as a "vanilla" extension of DPS to the latent domain (Rout et al., 2024) . We referred to this technique as Latent-DPS following Song et al. (2023). However, Latent-DPS achieves poor performance out-of-the-box due

to the inaccuracy of the approximation in early stages of diffusion and due to additional complications introduced by the decoder (Rout et al., 2024; Song et al., 2023; Chung et al., 2023). To address the latter concern, Rout et al. (2024) proposes two extensions of Latent-DPS: GML-DPS, that guides the diffusion towards fixed points of the encoder-decoder composition, and PSLD which adds a "gluing" term in order to avoid inconsistencies at mask boundaries. Authors in Song et al. (2023) take a different route and propose Re-Sample, a sampling technique that enforces data consistency directly on the posterior means by solving the optimization problem

$$\hat{\boldsymbol{z}}_0(\boldsymbol{y}) = \arg\min_{\boldsymbol{z}} \|\boldsymbol{y} - \mathcal{A}(\mathcal{D}_0(\boldsymbol{z}))\|_2^2 + \lambda\|\boldsymbol{z} - \hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)\|_2^2,$$

or its pixel domain equivalent, both referred to as hard data consistency. Chung et al. (2023) leverages text-to-image LDMs and incorporates prompt-tuning into their algorithm while combining Latent-DPS and hard data consistency.

A key challenge of diffusion-based solvers is their heavy compute demand, as reconstructing a single sample requires typically $500 - 1000$ evaluations of a large score model. Come-Closer-Diffuse-Faster (CCDF) (Chung et al., 2022c) shortens the sampling trajectory by leveraging a good initial posterior mean estimate $\hat{\boldsymbol{x}}_0$ from a reconstruction network. They initialize the reverse process by jumping to a fixed time step in the forward process via $\boldsymbol{x}_k = \sqrt{\bar{\alpha}_k}\hat{\boldsymbol{x}}_0 + \sqrt{1-\bar{\alpha}_k}\boldsymbol{\varepsilon}$, and only perform $k << N$ reverse diffusion steps, where k is a fixed hyperparameter. More recently, (Rout et al., 2023) proposes leveraging an efficient second-order approximation for estimating the posterior mean, notably accelerating sampling.

3. Method

3.1. Severity encoding

The goal of inverse problem solving is to recover the clean signal $\boldsymbol{x}$ from a corrupted observation $\boldsymbol{y}$ (see (1)). The degradation $\mathcal{A}$ and additive noise $\boldsymbol{n}$ fundamentally destroy information in $\boldsymbol{x}$. The amount of information loss, or the *severity* of the degradation, strongly depends on the interaction between the signal structure and the specific degradation. For instance, blurring removes high-frequency information, which implies that applying a blur kernel to an image with abundant high-frequency detail (textures, hair, background clutter etc.) results in a *more severe degradation* compared to applying the same kernel to a smooth image with few details. In other words, the difficulty of recovering the clean signal does not solely depend on the degradation process itself, but also on the specific signal the degradation is applied to. Thus, tuning the reconstruction method's capacity purely based on the forward model misses a key component of the problem: the data itself.

Quantifying the severity of a degradation is a challenging task in image domain. As an example, consider the forward model $\boldsymbol{y} = c\boldsymbol{x},\ c \in \mathbb{R}^+$ that simply rescales the clean signal. Recovery of $\boldsymbol{x}$ from $\boldsymbol{y}$ is trivial, however image similarity metrics such as PSNR or NMSE that are based on the Euclidean distance in image domain may indicate arbitrarily large discrepancy between the degraded and clean signals. On the other hand, consider $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n},\ \boldsymbol{n} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ where the clean signal is simply perturbed by some additive random noise. Even though the image domain perturbation is (potentially) small, information is fundamentally lost and perfect reconstruction is no longer possible.

What is often referred to as the *manifold hypothesis* (Bengio et al., 2013) states that natural images live in a lower dimensional manifold embedded in n-dimensional pixel-space. This in turn implies that the information contained in an image can be represented by a low-dimensional latent vector that encapsulates the relevant features of the image. Autoencoders (Kingma & Welling, 2013; Razavi et al., 2019) learn a latent representation from data by first summarizing the input image into a compressed latent vector $\boldsymbol{z} = \mathcal{E}_0(\boldsymbol{x})$ through an encoder. Then, the original image can be recovered from the latent via the decoder $\hat{\boldsymbol{x}} = \mathcal{D}_0(\boldsymbol{z})$ such that $\boldsymbol{x} \approx \hat{\boldsymbol{x}}$. As the latent space of autoencoders contains only the relevant information of data, it is a more natural space to quantify the loss of information due to the degradation than the image domain.

In particular, assume that we have access to the latent representation of clean images $\boldsymbol{z}_0 = \mathcal{E}_0(\boldsymbol{x}_0),\ \boldsymbol{z}_0 \in \mathbb{R}^d$, for instance from a pre-trained autoencoder. We propose a *severity encoder* $\hat{\mathcal{E}}_{\boldsymbol{\theta}}$ that achieves two objectives simultaneously: (1) it can predict the latent representation of a clean image, given a noisy and degraded observation and (2) it can quantify the error in its own latent estimation. We denote

$$\hat{\mathcal{E}}_{\boldsymbol{\theta}}(\boldsymbol{y}) = (\hat{\boldsymbol{z}},\ \hat{\sigma}) := (\hat{\mathcal{E}}_{\boldsymbol{z}}(\boldsymbol{y}), \hat{\mathcal{E}}_{\sigma}(\boldsymbol{y})),$$

with $\hat{\boldsymbol{z}} \in \mathbb{R}^d$ the estimate of $\boldsymbol{z}_0$ and $\hat{\sigma} \in \mathbb{R}$ the *estimated degradation severity* to be specified shortly. We use the notation $\hat{\mathcal{E}}_{\boldsymbol{z}}(\boldsymbol{y}) = \hat{\boldsymbol{z}}$ and $\hat{\mathcal{E}}_{\sigma}(\boldsymbol{y}) = \hat{\sigma}$ for the two conceptual components of our model, however in practice a single architecture is used to represent $\hat{\mathcal{E}}_{\boldsymbol{\theta}}$. The first objective can be interpreted as image reconstruction in the latent space of the autoencoder: for $\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{n}$ and $\boldsymbol{z}_0 = \mathcal{E}_0(\boldsymbol{x})$, we have $\hat{\mathcal{E}}_{\boldsymbol{z}}(\boldsymbol{y}) = \hat{\boldsymbol{z}} \approx \boldsymbol{z}_0$. The second objective captures the intuition that recovering $\boldsymbol{z}_0$ from $\boldsymbol{y}$ exactly may not be possible, and the prediction error is proportional to the loss of information about $\boldsymbol{x}$ due to the corruption. Thus, even though the predicted latent $\hat{\boldsymbol{z}}$ might be away from the true $\boldsymbol{z}_0$, the encoder quantifies the uncertainty in its own prediction. More specifically, we make the assumption that the prediction error in latent space can be modeled as zero-mean *i.i.d.* Gaussian. That is, we assume that

$$\hat{\boldsymbol{z}}(\boldsymbol{y}) - \boldsymbol{z}_0 := \boldsymbol{e}(\boldsymbol{y}) \sim \mathcal{N}(\mathbf{0}, \sigma_*^2(\boldsymbol{y})\mathbf{I}),$$

and we interpret the variance in prediction error σ_*^2 as the measure of degradation severity. We optimize the joint objective

$$\mathbb{E}_{\boldsymbol{x}_0\sim q_0(\boldsymbol{x}_0),\boldsymbol{y}\sim\mathcal{N}(\mathcal{A}(\boldsymbol{x}_0),\sigma_y^2\mathbf{I})}\Big[\Big\|\boldsymbol{z}_0-\hat{\mathcal{E}}_{\boldsymbol{z}}(\boldsymbol{y})\Big\|^2+$$
$$\lambda_\sigma\Big\|\bar{\sigma}^2(\boldsymbol{y},\boldsymbol{z}_0)-\hat{\mathcal{E}}_\sigma(\boldsymbol{y})\Big\|^2\Big]:=L_{lat.rec.}+\lambda_\sigma L_{err.},\quad(2)$$

with $\boldsymbol{z}_0=\mathcal{E}_0(\boldsymbol{x}_0)$ for a fixed, pre-trained encoder $\mathcal{E}_0$ and

$$\bar{\sigma}^2(\boldsymbol{y},\boldsymbol{z}_0)=\frac{1}{d-1}\sum_{i=1}^{d}(\boldsymbol{e}^{(i)}-\frac{1}{d}\sum_{j=1}^{d}\boldsymbol{e}^{(j)})^2$$

is the sample variance of the prediction error estimating σ_*^2. Here $\lambda_\sigma>0$ is a hyperparameter that balances between reconstruction accuracy ($L_{lat.rec.}$) and error prediction performance ($L_{err.}$). We empirically observe that even small loss values of $L_{lat.rec.}$ (that is fairly good latent reconstruction) may correspond to visible reconstruction error in image domain as semantically less meaningful features in image domain are often not captured in the latent representation. Furthermore, small errors in latent space can be severely amplified in image space after decoding, depending on the smoothness of the pretrained decoder. Therefore, we utilize an extra loss term that imposes image domain consistency with the ground truth image in the form

$$L_{im.rec.}=\mathbb{E}_{\boldsymbol{x}_0,\boldsymbol{y}}\Big[\Big\|\boldsymbol{x}_0-\mathcal{D}_0(\hat{\mathcal{E}}_{\boldsymbol{z}}(\boldsymbol{y}))\Big\|^2\Big],$$

resulting in the final combined loss $L_{sev}=L_{lat.rec.}+\lambda_\sigma L_{err.}+\lambda_{im.}L_{im.rec.}$, with $\lambda_{im.}\geq 0$ hyperparameter. Training the severity encoder is fast, as one can fine-tune the pre-trained encoder $\mathcal{E}_0$.

3.2. Sample-adaptive inference

Diffusion-based inverse problem solvers synthesize missing data that has been destroyed by the degradation process through diffusion. As shown in Figure 2, depending on the amount of missing information (easy vs. hard samples), the optimal number of diffusion steps may greatly vary. Too few steps may not allow the diffusion process to generate realistic details on the image, leaving the reconstruction overly smooth. On the other hand, diffusion-based solvers are known to hallucinate details that may be inconsistent with the ground truth signal, or even become unstable when too many reverse diffusion steps are applied. Authors in Chung et al. (2022c) observe that there always exists an optimal spot between 0 and N diffusion steps that achieves the best reconstruction performance. We aim to automatically find this "sweet spot" on a sample-by-sample basis.

Our proposed severity encoder learns to map degraded signals to a noisy latent representation, where the noise level is proportional to the degradation severity. This provides us the unique opportunity to leverage a latent diffusion process to progressively denoise the latent estimate obtained from our encoder. Even more importantly, we can automatically scale the number of reverse diffusion steps required to reach the clean latent manifold based on the predicted severity.

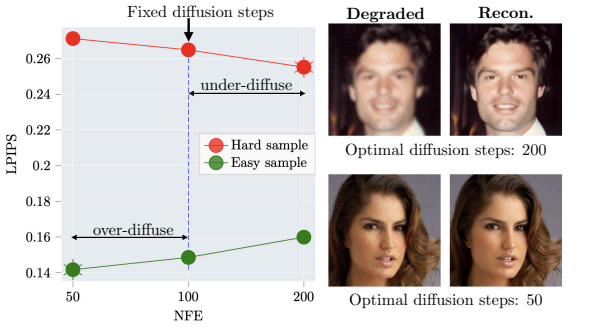


Figure 2: The optimal number of reverse diffusion steps varies depending on the severity of degradations. Fixing the number of steps results in over-diffusing some samples, whereas others could benefit from more iterations.

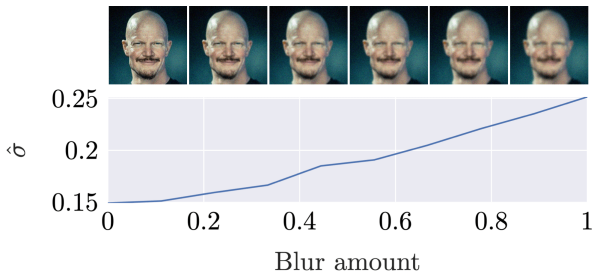


Figure 3: Effect of degradation on predicted severities: given a ground truth image corrupted by varying amount of blur, $\hat{\sigma}$ is a non-decreasing function of the blur amount.

Finding the optimal starting time – We find the time index i_{start} in the latent diffusion process at which the signal-to-noise ratio (SNR) matches the SNR predicted by the severity encoder. Assume that the latent diffusion process is specified by the conditional distribution $q_i(\boldsymbol{z}_i|\boldsymbol{z}_0)\sim\mathcal{N}(a_i\boldsymbol{z}_0,b_i^2\mathbf{I})$, where a_i and b_i are determined by the specific sampling method (e. g. $a_i=\sqrt{\bar{\alpha}_i}$ and $b_i^2=1-\bar{\alpha}_i$ for DDPM). On the other hand, we have the noisy latent estimate $\hat{\boldsymbol{z}}\sim\mathcal{N}(\boldsymbol{z}_0,\sigma_*^2(\boldsymbol{y})\mathbf{I})$, where we estimate σ_*^2 by $\hat{\mathcal{E}}_\sigma(\boldsymbol{y})$. Then, SNR matching gives us the starting time index

$$i_{start}(\boldsymbol{y})=\arg\min_{i\in[1,2,..,N]}\Big|\frac{a_i^2}{b_i^2}-\frac{1}{\hat{\mathcal{E}}_\sigma(\boldsymbol{y})}\Big|.\quad(3)$$

Thus, we start reverse diffusion from the initial reconstruction $\hat{\boldsymbol{z}}$ provided by the severity encoder and progressively denoise it using a pre-trained unconditional score model,

where the length of the sampling trajectory is directly determined by the predicted severity of the degraded example.

Noise correction – Even though we assume that the prediction error in latent space is *i.i.d.* Gaussian in order to quantify the estimation error by a single scalar, in practice the error often has some structure. This can pose a challenge for the score model, as it has been trained to remove isotropic Gaussian noise. We observe that it is beneficial to mix $\hat{\boldsymbol{z}}$ with some *i.i.d. correction noise* in order to suppress structure in the prediction error. In particular, we initialize the reverse process by

$$\boldsymbol{z}_{start} = \sqrt{1 - c\hat{\sigma}^2}\hat{\boldsymbol{z}} + \sqrt{c\hat{\sigma}^2}\boldsymbol{\varepsilon}, \;\; \boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$

where $c \geq 0$ is a tuning parameter. We discuss alternatives to this initialization scheme based on adaptive DDIM encoding in Appendix E.

4. Experiments

Dataset – We perform experiments on CelebA-HQ (256×256) (Karras et al., 2018) and LSUN Bedrooms (Yu et al., 2015). We match the training and validation splits used in (Rombach et al., 2022), and set aside 200 images from the validation split for testing. For comparisons involving the CelebA dataset and image domain score models we test on FFHQ (Karras et al., 2019), as pre-trained image-domain score models have been trained on the entire CelebA-HQ.

Degradations – We consider three degradations of diverging characteristics. Gaussian blur: we apply Gaussian blur with kernel size of 61. In order to investigate the effect of strongly varying degradation severity on sampling techniques, we analyze two settings: (1) varying Gaussian blur, where we sample the kernel standard deviation uniformly on $[0, 3]$ (0 corresponds to no blurring) and (2) fixed Gaussian blur where the kernel standard deviation is set to 3. Nonlinear blur: we deploy GOPRO motion blur simulated by a neural network model from Tran et al. (2021). This is a nonlinear forward model due to the camera response function. We randomly sample nonlinear blur kernels for each image. Varying random inpainting: we randomly mask $70 - 80\%$ of the pixels, where the masking ratio is drawn uniformly. In all experiments, we add Gaussian noise to images in the $[0, 1]$ range with noise std of 0.05.

Baseline solvers and comparison methods – Flash-Diffusion acts as a wrapper around any latent diffusion solver adding sample-adaptivity and acceleration to the baseline inverse problem solver. We experiment with the following baseline solvers: Latent-DPS (Rout et al., 2024), a straightforward adaptation of DPS (Chung et al., 2022a) to the latent domain; GML-DPS and PSLD (linear-only) (Rout et al., 2024), algorithms that are extensions of Latent-DPS with improved performance; and ReSample (Song et al., 2023), a state-of-the-art latent diffusion solver leveraging hard data consistency. We compare the baseline solvers to their adaptive version obtained from our framework, which we denote by `Flash`(baseline solver). Furthermore, we compare the results with DPS (Chung et al., 2022a), a well-established diffusion solver that operates in the pixel space. As our framework requires fine-tuning to the specific degradation, we additionally compare with SwinIR (Liang et al., 2021), a strong Transformer-based supervised image restoration model trained on corrupted-clean data pairs. Finally, we show results of decoding the severity encoder's $\hat{\boldsymbol{z}}$ estimate directly without diffusion, denoted by AE (autoencoded). Further details on comparison methods are in Appendix A.

Models – We use pre-trained DMs from Dhariwal & Nichol (2021) for DPS and from Rombach et al. (2022) for LDMs. We fine-tune severity encoders from pre-trained LDM encoders and utilize a single convolution layer on top of $\hat{\boldsymbol{z}}$ to predict $\hat{\sigma}$. For more details on the experimental setup and hyperparameters, see Appendix A.

Metrics – We evaluate image distortion based on PSNR and SSIM and perceptual image quality based on LPIPS and FID. To evaluate speedup of adaptive solvers compared to their baseline, we compare the average number of reverse diffusion steps across the test dataset, denoted by $\overline{\text{NFE}}$.

4.1. Severity encoding

In this section, we investigate properties of the predicted degradation severity $\hat{\sigma}$. We perform experiments on a $1k$-image subset of the validation split.

Effect of degradation level – First, we isolate the effect of degradation on $\hat{\sigma}$ (Fig. 3). We fix the clean image and apply increasing amount of Gaussian blur. We observe that $\hat{\sigma}$ is an increasing function of the blur amount applied to the image: heavier degradations on a given image result in higher predicted degradation severity. This implies that the severity encoder learns to capture the amount of information loss caused by the degradation.

Interaction between degradation and image – Next, we investigate the relation between $\hat{\sigma}$ and the underlying degradation severity (Fig. 4). We parameterize the corruption level by t, where $t = 0$ corresponds to no blur and $t = 1$ corresponds to the highest blur level where in both cases we add Gaussian noise ($\sigma = 0.05$). We vary the blur kernel width linearly for $t \in (0, 1)$. We observe that the predicted $\hat{\sigma}$ severities strongly correlate with the corruption level. However, the existence of outliers suggest that factors other than the corruption level may also contribute to the predicted severities. The bottom image is predicted to be *surprisingly easy*, as other images of the same corruption level are typically assigned higher predicted severities. This sample is overwhelmingly smooth, with a lack of fine details and textures,

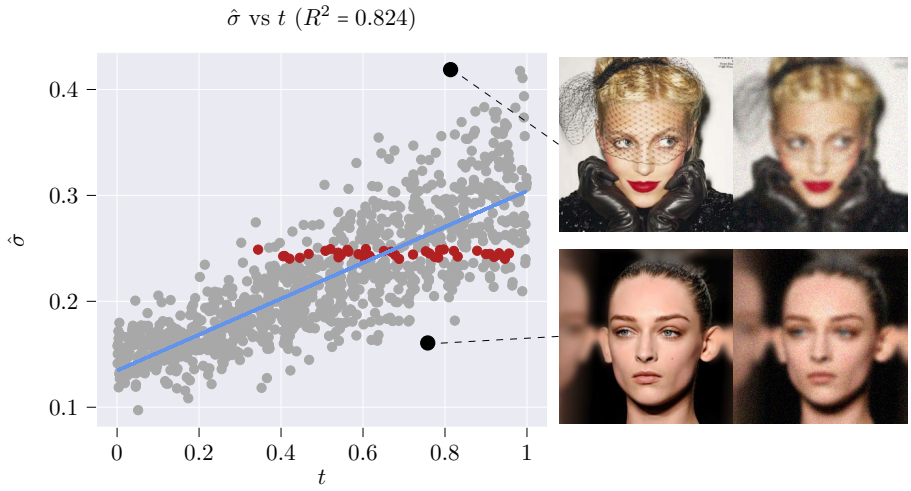


Figure 4: Blur amount (t) vs. predicted degradation severity ($\hat{\sigma}$). Outliers indicate that the predicted degradation severity is not solely determined by the amount of blur. The bottom image is *surprisingly easy* to reconstruct, as it is overwhelmingly smooth with features close to those seen in the training set. The top image is *surprisingly hard*, due to more high-frequency details and unusual features not seen during training. Points in red suggest that a given degradation severity may result from a wide range of blur levels (see Fig. 5 and discussion under *Identifying contributors to severity in Section 4.1*).

such as hair, present in other images. Moreover, the image shares common features with others in the training set. On the other hand, the top image is considered *surprisingly difficult*, as it contains unexpected features and high-frequency details that are uncommon in the dataset. This example highlights the potential application of our technique to hard example mining and dataset distillation.

Identifying contributors to severity – Finally, we analyze the contribution of different factors to the predicted degradation severities (Fig. 5). We apply severity encoding to both the clean image with noise (no blur) and the noisy and degraded image, resulting in predicted severities $\hat{\sigma}_{noisy}$ and $\hat{\sigma}_{degr.+noisy}$. We quantify the difficulty of samples relative to each other via percentiles of the above two quantities, where we use $\hat{\sigma}_{noisy}$ as a proxy for the difficulty originating from the image structure. The reason for not using the clean image without additive noise is due to the fact that the severity encoder is trained on varying degradation severities, but with a fixed additive noise. Therefore, the model is not intended to accept noiseless measurements as inputs. We observe that for a fixed $\hat{\sigma}_{degr.+noisy}$, the composition of degradation severity may greatly vary. The two presented images have been assigned approximately the same $\hat{\sigma}_{degr.+noisy}$, however the top image is inherently easy to encode (low $\hat{\sigma}_{noisy}$ percentile) compared to other images in the dataset, therefore the severity is mostly attributed to the image degradation. On the other hand, the bottom image with the same $\hat{\sigma}_{degr.+noisy}$ is less corrupted by blur, but with high $\hat{\sigma}_{noisy}$ indicating a difficult image. This example further corroborates the interaction between ground truth signal structure and the applied corruption in determining the difficulty of a reconstruction task. For a detailed discussion on the limitations of severity encoding, we refer the reader to Appendix C.

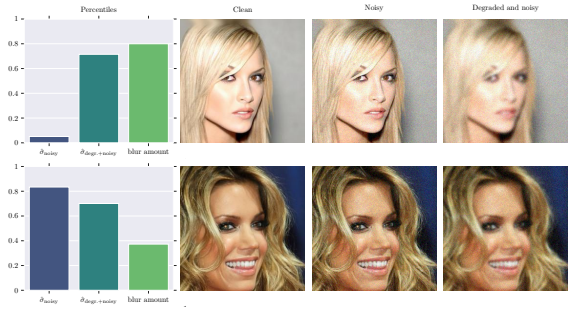


Figure 5: Contributors to severity. Degraded images with approx. the same $\hat{\sigma}$ may have different factors contributing to the predicted severity. The main contributor to $\hat{\sigma}$ in the top image is the image degradation (blur), whereas the bottom image is inherently more difficult to reconstruct.

4.2. Sample-adaptive reconstruction

Comparison of latent diffusion solvers– Table 1 summarizes our experimental results comparing various latent diffusion solvers with and without our Flash-Diffusion wrapper. Reconstructed samples are depicted in Figure 6. First, we observe that adaptive solvers consistently outperform their baselines across all degradations, datasets and image quality metrics. The performance improvement is especially significant in experiments with highly varying degradation severity,

FFHQ	Gaussian Deblurring (Varying)					Gaussian Deblurring (Fixed)					Nonlinear Deblurring					Random Inpainting				
Method	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE
Latent-DPS	23.69	0.6418	0.3579	87.26	1000	22.88	0.6136	0.3690	89.38	1000	22.07	0.5974	0.3814	90.89	1000	23.96	0.6566	0.3666	93.65	1000
`Flash`(Latent-DPS)	29.17	0.8182	0.2240	55.57	100.3	27.44	0.7691	0.2823	80.44	127.7	27.17	0.7659	0.2695	69.78	136.1	29.21	0.8414	**0.1945**	**53.95**	104.7
PSLD (Rout et al., 2024)	25.06	0.6769	0.3194	79.79	1000	23.72	0.6183	0.3324	88.45	1000	-	-	-	-	-	24.94	0.6617	0.3672	85.64	1000
`Flash`(PSLD)	29.26	0.8205	**0.2203**	**53.27**	100.3	27.44	0.7657	**0.2797**	**65.35**	127.7	-	-	-	-	-	27.06	0.8018	0.2185	55.12	104.7
GML-DPS (Rout et al., 2024)	24.98	0.6884	0.3471	100.27	1000	24.01	0.6574	0.3621	102.80	1000	23.00	0.6426	0.3812	108.79	1000	25.20	0.7044	0.3527	103.3	1000
`Flash`(GML-DPS)	29.21	0.8276	0.2274	69.16	100.3	27.47	0.7699	0.2816	69.81	127.7	27.11	0.7640	0.2756	81.93	136.1	28.95	0.8437	0.1957	59.39	104.7
ReSample (Song et al., 2023)	28.77	0.8219	0.2587	81.96	500	27.62	0.7789	0.3148	102.47	500	26.61	0.7318	0.2838	68.57	500	27.51	0.7892	0.2460	63.39	500
`Flash`(ReSample)	29.07	0.8330	0.2383	74.76	49.9	27.77	0.7845	0.3092	100.84	63.6	26.88	0.7660	**0.2667**	**64.57**	67.8	28.13	0.8260	0.2045	56.67	52.1
AE	29.46	0.8358	0.2671	89.29	-	27.69	0.7820	0.3396	110.56	-	27.17	0.7786	0.3364	111.24	-	29.23	0.8432	0.2515	85.87	-
SwinIR (Liang et al., 2021)	**30.69**	**0.8583**	0.2409	87.61	-	**28.41**	**0.8021**	0.3091	108.49	-	**27.60**	**0.7928**	0.3093	99.56	-	**30.08**	**0.8654**	0.2223	78.32	-
DPS (Chung et al., 2022a)	28.34	0.7791	0.2465	81.70	1000	25.49	0.6829	0.3035	97.89	1000	22.77	0.6191	0.3601	109.58	1000	28.30	0.8049	0.2451	82.78	1000

LSUN Bedrooms	Gaussian Deblurring (Varying)					Gaussian Deblurring (Fixed)					Nonlinear Deblurring					Random Inpainting				
Method	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE
Latent-DPS	22.42	0.5984	0.4227	58.72	1000	21.53	0.5602	0.4391	58.34	1000	19.93	0.5430	0.4724	80.54	1000	22.29	0.6056	0.4355	61.23	1000
`Flash`(Latent-DPS)	28.05	0.8099	0.2185	38.19	101.6	25.92	0.7410	0.2892	54.29	125.3	**26.60**	0.7709	0.2672	50.99	121.2	27.73	0.8472	**0.1876**	32.62	99.0
PSLD (Rout et al., 2024)	23.92	0.6584	0.3427	52.40	1000	22.71	0.5942	0.3941	62.05	1000	-	-	-	-	-	23.95	0.6578	0.3808	55.25	1000
`Flash`(PSLD)	28.08	0.8133	**0.2146**	**38.12**	101.6	25.88	0.7389	**0.2872**	**52.47**	125.3	-	-	-	-	-	27.62	0.8462	0.1879	**32.44**	99.0
GML-DPS (Rout et al., 2024)	23.55	0.6508	0.4253	84.81	1000	22.65	0.6173	0.4494	91.17	1000	21.75	0.6038	0.4591	96.15	1000	23.30	0.6522	0.4389	88.49	1000
`Flash`(GML-DPS)	28.05	0.8096	0.2200	38.16	101.6	**25.94**	0.7412	0.2902	54.27	125.3	26.57	0.7698	0.2690	50.97	121.2	**27.82**	**0.8498**	0.1880	34.25	99.0
ReSample (Song et al., 2023)	28.19	0.8193	0.2499	52.83	500	25.84	0.7394	0.3401	83.30	500	26.20	0.7658	0.2709	51.37	500	27.17	0.8121	0.2235	38.77	500
`Flash`(ReSample)	**28.27**	**0.8278**	0.2284	46.42	50.5	25.65	**0.7426**	0.3103	65.59	62.4	26.55	**0.7837**	**0.2586**	**49.62**	60.3	27.45	0.8334	0.1985	32.71	49.2

Table 1: Experimental results on FFHQ (top) and LSUN Bedrooms (bottom). We underline the better result between the baseline solver and its adapted variant, and use **bold** to highlight the best overall technique for the specific metric. Note that PSLD is only defined for linear inverse problems. Adaptive techniques consistently outperform their corresponding baseline across all metrics and achieve $8\times - 10\times$ speedup.

such as varying Gaussian blur, nonlinear blur with randomly sampled kernel and random inpainting, further highlighting the importance of sample-adaptivity. Interestingly, we observe that even though ReSample achieves the best performance among baseline solvers by a large margin, the performance gap diminishes between adaptive solvers with `Flash`(Latent-DPS) often outperforming more advanced techniques. We hypothesize that the differences between baseline solvers stem from differences in the diffusion dynamics in the early and middle stages (*chaotic* and *semantic* stages in Song et al. (2023)) of sampling, a regime typically bypassed by Flash-Diffusion. In terms of sampling speedup, we obtain $8\times - 10\times$ acceleration in average number of reverse diffusion steps across the test dataset compared to the non-adaptive baseline, without any drop in reconstruction quality. Furthermore, we observe that simply leveraging DDIM acceleration in the baseline solvers, given the same compute budget in inference time, cannot compete with the reconstruction quality obtained from Flash-Diffusion (details in Appendix D).

Comparison with other methods – In Table 1 (top), we observe that SwinIR, a state-of-the-art supervised restoration model achieves higher PSNR and SSIM than diffusion methods, however it falls short in terms of perceptual image quality. This phenomenon is due to the perception-distortion trade-off (Blau & Michaeli, 2018): improving perceptual image quality is fundamentally at odds with distortion metrics. Furthermore, we highlight that our initial reconstructions obtained from the severity encoder are poorer quality than SwinIR reconstructions (compare AE vs. SwinIR), however diffusion samplers greatly improve over the perceptual quality of the initial reconstruction, eventually surpassing both SwinIR and all non-adaptive diffusion solver baselines.

Importance of sample-adaptivity – In order to highlight the importance of sample-adaptivity, we compare our adaptive technique to a simple non-adaptive modification. Here, instead of predicting the starting time via severity encoding on a sample-by-sample basis, we set the starting time for all samples to N'. For the sake of simplicity, we choose `Flash`(Latent-DPS) as our adaptive method, and vary the fixed start time between $10 - 500$ for the non-adaptive variant. We leverage SwinIR reconstructions as initialization for the non-adaptive variant. We perform experiments on CelebA-HQ under varying Gaussian blur. The results are depicted in Fig. 7. Our adaptive method achieves the best FID across any fixed number of steps by a large margin, despite the lower quality initialization from the severity encoder. Moreover, our technique with sample-adaptivity achieves near-optimal LPIPS with $2\times$ less average number of diffusion steps compared to the non-adaptive variant. We observe that the predicted start times from our technique are spread around the mean and not closely concentrated, further highlighting the adaptivity of Flash-Diffusion.

Robustness and limitations– The performance of Flash-Diffusion relies on the accuracy of the severity encoder trained in a supervised fashion on the degraded image distribution. Therefore, it is crucial to understand the robustness of severity encoding to noise level and forward model shifts in test time. In terms of noise, we observe that severity encoding maintains its performance on noise levels lower than in the training setting, however it breaks down at noise levels

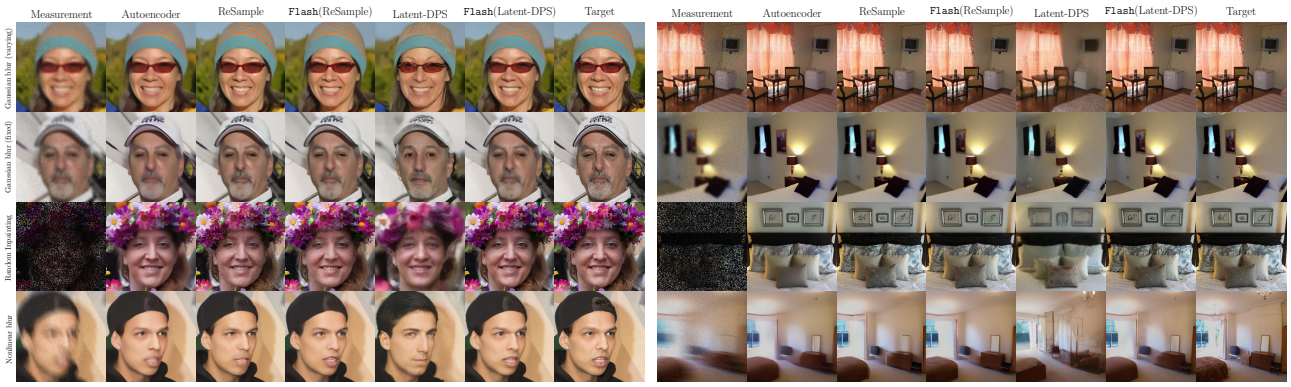


Figure 6: Visual comparison of reconstructions on the FFHQ (left) and LSUN Bedrooms (right) datasets.

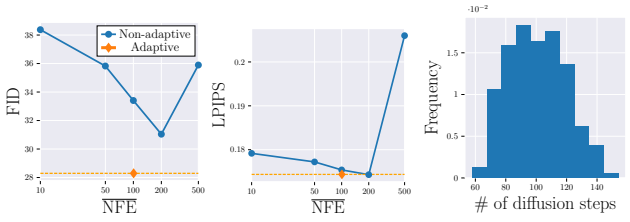


Figure 7: Comparison of Flash(Latent-DPS) with a non-adaptive variant with fixed starting time (varying Gaussian blur, CelebA-HQ). Top: Reconstruction quality as a function of the number of diffusion steps for the non-adaptive variant with various fixed starting times. The data point corresponding to our adaptive method depicts the average number of diffusion steps due to adaptive starting time. Our adaptive method achieves the best FID and near-optimal LPIPS compared to any choice of non-adaptive starting time. Bottom: Histogram of predicted starting times for Flash(Latent-DPS).

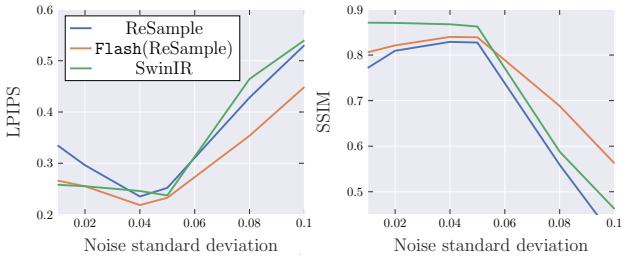


Figure 8: Robustness experiment: we perturb the forward model noise level and compare the performance of ReSample and its adaptive variant. We observe improved robustness after imbuing the solver with sample-adaptivity.

well above the training regime. In terms of forward model mismatch, we find that as long as the train and test operators are similar (heavy Gaussian blur and nonlinear blur), severity encoding can maintain acceptable performance. However in case of significant mismatch, the accuracy of severity encoding breaks down. An in-depth characterization of severity encoder robustness is detailed in Appendix C.

Moreover, we analyze the robustness of Flash-Diffusion solvers compared to their non-adaptive counterparts with respect to a mismatch in forward model noise level. In particular, we base our comparison on ReSample, the best performing baseline solver in our experiments. Fig. 8 demonstrates that the performance of ReSample degrades more rapidly than its adaptive counterpart when the noise level is higher than expected, following a similar trend to SwinIR, a supervised non-diffusion reconstructor. In the low-noise regime, Flash(ReSample) shows improved robustness compared to the baseline solver. More experiments on forward model mismatch can be found in Appendix C.2.

5. Conclusions

In this work, we make the key observation that the difficulty of solving an inverse problem may vary greatly on a sample-by-sample basis, depending on the ground truth signal structure, the applied corruption, the model, the training set and the complex interactions between these factors. Despite this natural variation in the difficulty of a reconstruction task, most techniques apply a rigid model that expends the same compute irrespective of the amount of information destroyed in the noisy measurements, resulting in suboptimal performance and wasteful resource allocation. We propose Flash-Diffusion, a sample-adaptive method that predicts the degradation severity of corrupted signals, and utilizes this estimate to automatically tune the compute allocated to reconstruct the sample. In particular, we use the prediction error of an encoder in latent space as a proxy for reconstruction difficulty, which we call severity encoding. Then, we leverage a latent diffusion process to reconstruct the sample, where the length of the sampling trajectory is directly scaled by the predicted severity. Our method can be coupled with any latent diffusion solver, greatly enhancing its performance and accelerating sampling by up to $10\times$. We demonstrate the gains in terms of reconstruction quality and speedup on a variety of degradations and datasets.

Impact Statement

The presented paper leverages deep learning for reconstructing corrupted images. Deep learning-based reconstruction approaches are known to hallucinate realistic-looking features that may be inconsistent with the ground truth. Therefore, deploying our technique in safety-critical applications requires additional caution. Moreover, the technique leverages diffusion models capable of generating photorealistic synthetic images, which in turn can be abused by bad actors to spread misinformation.

Acknowledgments

We thank the anonymous reviewers who provided valuable suggestions and constructive feedback on our work. M. Soltanolkotabi is supported by the Packard Fellowship in Science and Engineering, a Sloan Research Fellowship in Mathematics, an NSF-CAREER under award #1846369, DARPA FastNICS program, and NSF-CIF awards #1813877 and #2008443. and NIH DP2LM014564-01. This research is also in part supported by AWS credits through an Amazon Faculty research award.

References

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828, 2013.

Blau, Y. and Michaeli, T. The perception-distortion tradeoff. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6228–6237, 2018.

Candes, E. J., Romberg, J. K., and Tao, T. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59(8):1207–1223, 2006.

Chung, H. and Ye, J. C. Score-based diffusion models for accelerated mri. *Medical Image Analysis*, 80:102479, 2022.

Chung, H., Kim, J., Mccann, M. T., Klasky, M. L., and Ye, J. C. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022a.

Chung, H., Sim, B., Ryu, D., and Ye, J. C. Improving diffusion models for inverse problems using manifold constraints. *arXiv preprint arXiv:2206.00941*, 2022b.

Chung, H., Sim, B., and Ye, J. C. 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*, pp. 12413–12422, 2022c.

Chung, H., Ye, J. C., Milanfar, P., and Delbracio, M. Prompt-tuning latent diffusion models for inverse problems. *arXiv preprint arXiv:2310.01110*, 2023.

Delbracio, M. and Milanfar, P. Inversion by direct iteration: An alternative to denoising diffusion for image restoration. *arXiv preprint arXiv:2303.11435*, 2023.

Dhariwal, P. and Nichol, A. Diffusion Models Beat GANs on Image Synthesis. *arXiv preprint arXiv:2105.05233*, 2021.

Fabian, Z., Tinaz, B., and Soltanolkotabi, M. Diracdiffusion: Denoising and incremental reconstruction with assured data-consistency. *arXiv preprint arXiv:2303.14353*, 2023.

Gilton, D., Ongie, G., and Willett, R. Deep equilibrium architectures for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:1123–1133, 2021.

Hand, P., Leong, O., and Voroninski, V. Phase retrieval under a generative prior. *Advances in Neural Information Processing Systems*, 31, 2018.

Ho, J., Jain, A., and Abbeel, P. Denoising Diffusion Probabilistic Models. *arXiv preprint arXiv:2006.11239*, 2020.

Ho, J., Saharia, C., Chan, W., Fleet, D. J., Norouzi, M., and Salimans, T. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23(47):1–33, 2022a.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022b.

Jiang, Y., Zhang, Z., Xue, T., and Gu, J. Autodir: Automatic all-in-one image restoration with latent diffusion. *arXiv preprint arXiv:2310.10123*, 2023.

Kadkhodaie, Z. and Simoncelli, E. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. *Advances in Neural Information Processing Systems*, 34:13242–13254, 2021.

Karras, T., Aila, T., Laine, S., and Lehtinen, J. Progressive Growing of GANs for Improved Quality, Stability, and Variation. *arXiv:1710.10196 [cs, stat]*, 2018.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Kawar, B., Vaksman, G., and Elad, M. Snips: Solving noisy inverse problems stochastically. *Advances in Neural Information Processing Systems*, 34:21757–21769, 2021.

Kawar, B., Elad, M., Ermon, S., and Song, J. Denoising diffusion restoration models. *arXiv preprint arXiv:2201.11793*, 2022a.

Kawar, B., Song, J., Ermon, S., and Elad, M. Jpeg artifact correction using denoising diffusion restoration models. *arXiv preprint arXiv:2209.11888*, 2022b.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., 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*, pp. 4681–4690, 2017.

Liang, J., Cao, J., Sun, G., Zhang, K., Van Gool, L., and Timofte, R. SwinIR: Image restoration using Swin Transformer. *arXiv:2108.10257*, 2021.

Luo, Z., Gustafsson, F. K., Zhao, Z., Sjölund, J., and Schön, T. B. Refusion: Enabling large-size realistic image restoration with latent-space diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1680–1691, 2023.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Ongie, G., Jalal, A., Baraniuk, C. A. M. R. G., Dimakis, A. G., and Willett, R. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 2020.

Preechakul, K., Chatthee, N., Wizadwongsa, S., and Suwajanakorn, S. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10619–10629, 2022.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Razavi, A., Van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Rout, L., Chen, Y., Kumar, A., Caramanis, C., Shakkottai, S., and Chu, W.-S. Beyond first-order tweedie: Solving inverse problems using latent diffusion. *arXiv preprint arXiv:2312.00852*, 2023.

Rout, L., Raoof, N., Daras, G., Caramanis, C., Dimakis, A., and Shakkottai, S. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Saharia, C., Ho, J., Chan, W., Salimans, T., Fleet, D. J., and Norouzi, M. Image Super-Resolution via Iterative Refinement. *arXiv:2104.07636 [cs, eess]*, 2021.

Saharia, C., Chan, W., Chang, H., Lee, C., Ho, J., Salimans, T., Fleet, D., and Norouzi, M. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pp. 1–10, 2022a.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022b.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Song, B., Kwon, S. M., Zhang, Z., Hu, X., Qu, Q., and Shen, L. Solving inverse problems with latent diffusion models via hard data consistency. *arXiv preprint arXiv:2307.08123*, 2023.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Song, Y. and Ermon, S. Generative Modeling by Estimating Gradients of the Data Distribution. *arXiv:1907.05600 [cs, stat]*, 2020a.

Song, Y. and Ermon, S. Improved Techniques for Training Score-Based Generative Models. *arXiv:2006.09011 [cs, stat]*, 2020b.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Song, Y., Shen, L., Xing, L., and Ermon, S. Solving inverse problems in medical imaging with score-based generative models. *arXiv preprint arXiv:2111.08005*, 2021b.

Sriram, A., Zbontar, J., Murrell, T., Defazio, A., Zitnick, C. L., Yakubova, N., Knoll, F., and Johnson, P. End-to-end variational networks for accelerated MRI reconstruction. In *Medical Image Computing and Computer Assisted Intervention*, pp. 64–73, 2020.

Sun, J., Li, H., Xu, Z., et al. Deep ADMM-Net for compressive sensing MRI. *Advances in Neural Information Processing Systems*, 29, 2016.

Tran, P., Tran, A. T., Phung, Q., and Hoai, M. Explore image deblurring via encoded blur kernel space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11956–11965, 2021.

Wang, Y., Tao, X., Qi, X., Shen, X., and Jia, J. Image inpainting via generative multi-column convolutional neural networks. *Advances in neural information processing systems*, 31, 2018.

Welker, S., Chapman, H. N., and Gerkmann, T. Driftrec: Adapting diffusion models to blind image restoration tasks. *arXiv preprint arXiv:2211.06757*, 2022.

Yu, F., Seff, A., Zhang, Y., Song, S., Funkhouser, T., and Xiao, J. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Zhang, J. and Ghanem, B. ISTA-Net: Interpretable optimization-inspired deep network for image compressive sensing. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1828–1837, 2018.

A. Experimental Details

Here we provide additional details on the experimental setup and hyperparameters.

Datasets – For experiments on CelebA-HQ and FFHQ datasets, we use the train and validation splits provided in the GitHub repo of "Taming Transformers"[2]. For LSUN Bedrooms experiments, we use the custom split provided in GitHub repo of "Latent Diffusion"[3]. For the test split, we subsample 200 ids from the corresponding validation ids files. Specific ids we used will be available when the codebase is released. We provide the specifics for each comparison method next.

Model architecture – In all experiments, we use LDM models pre-trained on the CelebA-HQ and LSUN Bedrooms datasets out of the box. We fine-tune the severity encoder from the LDM's pre-trained encoder. We obtain the degradation severity estimate $\hat{\sigma} \in \mathbb{R}^+$ from the latent reconstruction $\hat{\boldsymbol{z}} \in \mathbb{R}^d$ as

$$\hat{\sigma} = \frac{1}{d} \sum_{i=1}^{d} [Conv(\hat{\boldsymbol{z}})^2]_i,$$

where $Conv(\cdot)$ is a learned 1×1 convolution with d input and d output channels.

Training setup – We train severity encoders using Adam optimizer with batch size 28 and learning rate 0.0001 for about $200k$ steps until the loss on the validation set converges. We use Quadro RTX 5000 and Titan GPUs.

Hyperparameters – We scale the reconstruction loss terms with their corresponding dimension (d for $L_{lat.rec.}$ and n for $L_{im.rec.}$), which we find to be sufficient without tuning for $\lambda_{im.rec.}$. We tune λ_σ via grid search on $[0.1, 1, 10]$ on the varying Gaussian blur task and set to 10 for all experiments.

For `Flash` experiments, we tune the noise correction parameter c on the validation subset by grid search over $[0.8, 1.0, 1.2]$ after tuning the remaining baseline method hyperparameters.

Next, we provide details about baseline solvers, comparison methods and hyperparameter tuning.

DPS (Chung et al., 2022a): DPS approximates the gradient of the likelihood as

$$\nabla_{\boldsymbol{x}_i} \log p(\boldsymbol{y}|\boldsymbol{x}_i) \approx \nabla_{\boldsymbol{x}_i} \log p(\boldsymbol{y}|\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i)),$$

where $\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i)$ is the posterior mean estimate obtained from the score model, resulting in the update rule

$$\boldsymbol{x}_{i-1} = \boldsymbol{x}'_{i-1} - \eta_t \nabla_{\boldsymbol{x}_i} \left\| \mathcal{A}(\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i) - \boldsymbol{y} \right\|^2, \tag{4}$$

where $\boldsymbol{x}'_{i-1}$ is the result of the unconditional diffusion step and

$$\eta_i^{DPS} = \frac{\eta^{DPS}}{\|\mathcal{A}(\hat{\boldsymbol{x}}_0(\boldsymbol{x}_i)) - \boldsymbol{y}\|}, \tag{5}$$

with $\eta^{DPS} > 0$ tuning parameter.

We use the code implementation in the official GitHub repository[4]. For all other solvers, we base our implementation on the LDM repository[3]. We tune $\eta^{DPS} > 0$ by performing a grid search over $[0.1, 0.5, 1.0, 2.0, 3.0, 5.0, 10.0]$ for all experiments. We provide the optimal hyperparameters in Table 2.

Latent-DPS: This solver is the direct application of DPS in the latent space of an autoencoder that uses the approximation

$$\nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\boldsymbol{z}_i) \approx \nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))), \tag{6}$$

leading to the updates

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}'_{i-1} - \eta_i \nabla_{\boldsymbol{z}_i} \left\| \mathcal{A}(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)) - \boldsymbol{y} \right\|^2. \tag{7}$$

Here $\boldsymbol{z}'_{i-1}$ is the result of the unconditional latent diffusion update and

$$\eta_i^{LDPS} = \frac{\eta^{LDPS}}{\|\mathcal{A}(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_t))) - \boldsymbol{y}\|} \tag{8}$$

[2] https://github.com/CompVis/taming-transformers/tree/master/data

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

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

with $\eta^{LDPS} > 0$ tuning parameter. We tune $\eta^{LDPS} > 0$ by performing a grid search over $[0.1, 0.2, 0.3, 0.4, 0.5, 1.0]$ for all baseline FFHQ experiments, and over $[0.1, 0.2, 0.4, 1.0]$ for baseline LSUN Bedrooms experiments. For `Flash`(Latent-DPS) we observe larger η^{LDPS} values work better. Hence, we search over $[1.0, 2.0, 3.0, 4.0, 7.0, 10.0]$ for FFHQ experiments and $[1.0, 2.0, 4.0, 10.0]$ for LSUN Bedrooms experiments. We provide the optimal hyperparameters in Table 2.

GML-DPS (Rout et al., 2024): Latent-DPS achieves poor performance out-of-the-box due to the inaccuracy of the approximation in 6 in early stages of diffusion and due to additional complications introduced by the decoder (Rout et al., 2024; Song et al., 2023; Chung et al., 2023). GML-DPS is an extension of Latent-DPS that guides the diffusion towards fixed points of the encoder-decoder composition via the "goodness modified" approximation

$$\nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\boldsymbol{z}_i) \approx \nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))) + \lambda \nabla_{\boldsymbol{z}_i} \|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|^2, \tag{9}$$

with some $\lambda > 0$ weight. We normalize the additional guidance term similarly to DPS resulting in the update

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}'_{i-1} - \eta_i^{LDPS} \nabla_{\boldsymbol{z}_i} \|\mathcal{A}(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))) - \boldsymbol{y}\|^2 - \gamma_i^{GM} \nabla_{\boldsymbol{z}_i} \|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|^2, \tag{10}$$

where

$$\gamma_i^{GM} = \frac{\gamma^{GM}}{\|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|}, \tag{11}$$

and γ^{GM} is a hyperparameter to be tuned. We tune $\eta^{LDPS} > 0$ and $\gamma^{GM} > 0$ by performing a greedy 2D grid search (first tune η^{LDPS} with $\gamma^{GM} = 0.1$ and then tune γ^{GM} with the best η^{LDPS} value) over $[0.1, 0.2, 0.3, 0.4, 0.5, 1.0] \times [0.05, 0.1, 0.2, 0.5]$ for all baseline FFHQ experiments, and over $[0.1, 0.2, 0.4, 1.0] \times [0.05, 0.1, 0.2, 0.5]$ for baseline LSUN Bedrooms experiments. For `Flash`(GML-DPS), we search over $[1.0, 2.0, 3.0, 5.0, 10.0] \times [0.05, 0.1, 0.2, 0.5]$ for FFHQ experiments and $[1.0, 2.0, 4.0, 10.0] \times [0.05, 0.1, 0.2, 0.5]$ for LSUN Bedrooms experiments. We provide the optimal hyperparameters in Table 2.

PSLD (Rout et al., 2024): PSLD improves upon GML-DPS by adding a "gluing" term in order to avoid inconsistencies at mask boundaries. The approximation takes the form

$$\nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\boldsymbol{z}_i) \approx \nabla_{\boldsymbol{z}_i} \log p(\boldsymbol{y}|\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))) + \lambda \nabla_{\boldsymbol{z}_i} \|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{A}\boldsymbol{x}_0 + (\mathbf{I} - \mathcal{A}\mathcal{A}^T)\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|^2, \tag{12}$$

leading to updates in the form

$$\boldsymbol{z}_{i-1} = \boldsymbol{z}'_{i-1} - \eta_i^{LDPS} \nabla_{\boldsymbol{z}_i} \|\mathcal{A}(\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i))) - \boldsymbol{y}\|^2 - \gamma_i^{PSLD} \nabla_{\boldsymbol{z}_i} \|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{A}\boldsymbol{x}_0 + (\mathbf{I} - \mathcal{A}\mathcal{A}^T)\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|^2, \tag{13}$$

where $\boldsymbol{x}^*$ is the ground truth clean signal and

$$\gamma_i^{PSLD} = \frac{\gamma^{PSLD}}{\|\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i) - \mathcal{E}_0(\mathcal{A}\boldsymbol{x}_0 + (\mathbf{I} - \mathcal{A}\mathcal{A}^T)\mathcal{D}_0(\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)))\|}, \tag{14}$$

and γ^{PSLD} is a hyperparameter. Note that PSLD requires access to noiseless measurements $\mathcal{A}\boldsymbol{x}_0$ and to the transposed of the linear operator $\mathcal{A}$. In our experiments that involve measurement noise, we replace $\mathcal{A}\boldsymbol{x}_0$ with $\boldsymbol{y}$ (noisy measurement). For Gaussian blur, we implement the transposed operator $\mathcal{A}^T$ using `ConvTranspose2d` in `torch`. We omit experiments involving nonlinear blur with PSLD as the transposed operator is not known.

We tune $\eta^{LDPS} > 0$ and $\gamma^{PSLD} > 0$ by performing a greedy 2D grid search over $[0.1, 0.2, 0.3, 0.4, 0.5, 1.0] \times [0.05, 0.1, 0.2, 0.5]$ for all baseline FFHQ experiments, and over $[0.1, 0.2, 0.4, 1.0] \times [0.05, 0.1, 0.2, 0.5]$ for baseline LSUN Bedrooms experiments. For `Flash`(PSLD), we search over $[1.0, 2.0, 4.0, 10.0] \times [0.05, 0.1, 0.2, 0.5]$ for all experiments. We provide the optimal hyperparameters in Table 2.

ReSample (Song et al., 2023): Instead of DPS-like approximations as in the previous solvers, ReSample applies data consistency directly on the posterior mean estimates by solving the latent-domain optimization problem

$$\hat{\boldsymbol{z}}_0(\boldsymbol{y}) = \arg\min_{\boldsymbol{z}} \|\boldsymbol{y} - \mathcal{A}(\mathcal{D}_0(\boldsymbol{z}))\|_2^2 + \lambda \|\boldsymbol{z} - \hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)\|_2^2, \tag{15}$$

or its pixel-domain variant after passing $\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)$ through the decoder. To solve the above problem, numerical methods are used where optimization is terminated when the stopping condition $\frac{1}{m}\|\boldsymbol{y} - \mathcal{A}(\mathcal{D}_0(\boldsymbol{z}))\|_2^2 < \tau$ is reached. τ is a hyperparameter, which we set to σ_y^2 as reducing the error below the noise level amounts to overfitting to the noisy measurement. Furthermore,

we set $\lambda = 1$ for all experiments following (Song et al., 2023), as we observe that the method is not sensitive to the choice of this parameter as long as the problem is solved to the desired accuracy. We use Adam optimizer with step size 0.1 and initial guess $\hat{\boldsymbol{z}}_0(\boldsymbol{z}_i)$ to minimize the objective in (15). As solving (15) is computationally heavy, it is only performed every K steps, where we set $K = 10$ for all experiments following the setting in the original paper.

To map $\hat{\boldsymbol{z}}_0(\boldsymbol{y})$ back to the noisy latent manifold, authors introduce stochastic resampling in the form

$$\boldsymbol{z}_i = \frac{\omega_i^2\sqrt{\bar{\alpha}_i}\hat{\boldsymbol{z}}_0(\boldsymbol{y}) + (1-\bar{\alpha}_i)\boldsymbol{z}_i'}{\omega_i^2 + (1-\bar{\alpha}_i)} + \sqrt{\frac{\omega_i^2(1-\bar{\alpha}_i)}{\omega_i^2 + (1-\bar{\alpha}_i)}}\epsilon, \;\; \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{16}$$

where ω_i controls the trade-off between prior consistency and data consistency and is a hyperparameter. Authors choose the adaptive schedule

$$\omega_i^2 = \omega\left(\frac{1-\bar{\alpha}_{i-1}}{\bar{\alpha}_i}\right)\left(1 - \frac{\bar{\alpha}_i}{\bar{\alpha}_{i-1}}\right), \tag{17}$$

where only ω needs to be tuned. We do not use time-travel updates as in the original algorithm as it appears to have only minor contribution to performance (see Table 8. in (Song et al., 2023)). Furthermore, we use DDIM sampling with $N = 500$ matching the setting in the original paper.

We tune $\omega > 0$ by performing a grid search over $[0, 10, 20, 40, 100, 200, 400, 1000, 2000, 4000]$ for all experiments. We provide the optimal hyperparameters in Table 2.

SwinIR: For all experiments, we train SwinIR using Adam optimizer with batch size 28 for 100 epochs. We use learning rate 0.0002 for the first 90 epochs and drop it by a factor of 10 for the remaining 10 epochs. We use Quadro RTX 5000 and Titan GPUs. We use 4 RSTB blocks, each with 6 STLs and embedding dimension 96.

Autoencoded (AE): We use the latent at severity encoders output and decode it without reverse diffusion to get the reconstruction.

We provide all the optimal hyperparameters we use in our experiments in Table 2.

Dataset	Operator	DPS	Latent-DPS	`Flash`(Latent-DPS)		GML-DPS		`Flash`(GML-DPS)			PSLD		`Flash`(PSLD)			ReSample	`Flash`(ReSample)	
		η^{DPS}	η^{LDPS}	η^{LDPS}	c	η^{LDPS}	γ^{GM}	η^{LDPS}	γ^{GM}	c	η^{LDPS}	γ^{PSLD}	η^{LDPS}	γ^{PSLD}	c	ω	ω	c
FFHQ	Gaussian blur (varying)	5.0	0.1	1.0	1.2	0.2	0.05	3.0	0.05	1.2	0.2	0.05	2.0	0.2	1.0	400	400	1.2
	Gaussian blur (fixed)	3.0	0.1	3.0	1.2	0.2	0.05	2.0	0.05	1.0	0.2	0.1	2.0	0.1	1.0	4000	2000	1.2
	Nonlinear blur	0.5	0.1	1.0	1.0	0.2	0.1	3.0	0.05	1.2	-	-	-	-	-	400	200	1.0
	Random inpainting	3.0	0.3	7.0	1.2	0.5	0.1	10.0	0.05	1.2	0.5	0.05	10.0	0.5	1.0	200	200	1.2
LSUN Bedrooms	Gaussian blur (varying)	-	0.1	2.0	1.2	0.2	0.1	2.0	0.05	1.2	0.2	0.2	2.0	0.2	1.0	2000	1000	1.2
	Gaussian blur (fixed)	-	0.1	2.0	1.0	0.2	0.1	2.0	0.05	1.0	0.2	0.1	2.0	0.1	1.0	2000	200	1.2
	Nonlinear blur	-	0.1	2.0	1.0	0.2	0.1	2.0	0.05	1.0	-	-	-	-	-	1000	1000	1.0
	Random inpainting	-	0.2	10.0	1.2	0.4	0.1	10.0	0.05	1.0	0.4	0.05	10.0	0.05	1.2	1000	400	1.2

Table 2: Optimal hyperparameters used in Flash-Diffusion experiments.

B. Ablation of Flash-Diffusion Components

Flash-Diffusion combines sample-by-sample adaptation of the sampling trajectory via severity encoding with latent-space diffusion. We ablate the effect of these two key components and demonstrate their relative contribution to the performance of Flash-Diffusion. In particular, we choose `Flash`(Latent-DPS) as our adaptive technique and we ablate the effect of adaptation by fixing the starting time for each sample to the average predicted starting time of `Flash`(Latent-DPS) on the corresponding task ($N = 100$ for Gaussian deblurring, $N = 134$ for nonlinear deblurring). We further ablate the effect of latent domain reconstruction via repeating the above experiment with a pixel-space score model. We optimized all models for best performance in terms of LPIPS via grid search over $\eta = [0.5, 1.0, 1.5, 2.0, 3.0, 5.0, 20.0]$. Our findings are summarized in Table 3. We observe, that adaptation via severity encoding provides a significant boost to the performance in terms of LPIPS. When comparing non-adaptive methods, we obtain similar results with and without latent space diffusion, however latent diffusion is preferable since (1) it has greatly reduced computational cost and (2) degradation severity is better captured in latent space.

		Gaussian Deblurring				Nonlinear Deblurring			
Adaptive?	**Latent?**	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)
✓	✓	29.16	0.8191	**0.2241**	**29.46**	27.22	0.7705	**0.2694**	**36.92**
✗	✓	**29.55**	**0.8377**	0.2346	49.06	**27.35**	**0.7844**	0.2847	54.25
✗	✗	29.23	0.8058	0.2377	54.20	26.83	0.7379	0.2843	57.82

Table 3: Ablation of different components of the Flash-Diffusion framework on the FFHQ dataset.

C. Robustness Studies

C.1. Noise robustness

Robustness of severity encoding – As demonstrated throughout the paper, our severity encoder provides useful estimates of the degradation severity of samples in the scenario where the degradation and noise level matches that in the training setup. However, it is very important to understand the limitations of severity encoding in the presence of test-time shifts in the corruption process.

Our most crucial expectation towards the severity encoder is to predict a degradation severity $\hat{\sigma}(\boldsymbol{y})$ that is a non-decreasing function of the true underlying corruption level. More specifically, assume that the forward model $\mathcal{A}$ is parameterized by a severity level $s \in [0, 1]$, with $s = 0$ being least severe, $s = 1$ being most severe. For instance, the standard deviation of the Gaussian blur kernel may serve as s in case of Gaussian blur (see a more rigorous treatment of degradation severity in (Fabian et al., 2023)). Then for a given specific clean image $\boldsymbol{x}_0$ and its corrupted versions $\boldsymbol{y}_{s'} := \mathcal{A}(\boldsymbol{x}_0; s') + \boldsymbol{n}$, $\boldsymbol{y}_{s''} := \mathcal{A}(\boldsymbol{x}_0; s'') + \boldsymbol{n}$, we expect our severity encoder to satisfy

$$\hat{\sigma}(\boldsymbol{y}_{\boldsymbol{s'}}) \leq \hat{\sigma}(\boldsymbol{y}_{\boldsymbol{s''}}), \quad , \quad \forall s', s'' \in [0, 1],\ s' < s'', \tag{18}$$

which we call the *monotonicity requirement*, i.e. the ordering of predicted severities need to match the ordering of the true underlying severity levels.

We demonstrate this property in Figure 3 for the case where the measurement noise level corresponds to the training setting ($\sigma_{\boldsymbol{y}} = 0.05$). Here, we observe that the predicted severity is a non-decreasing function of the true blur level (monotonicity requirement is satisfied) when the ground truth clean image is fixed. Next, we increase the measurement noise in test time and investigate how the monotonicity requirement breaks down (see Figure 9). At $\sigma_{\boldsymbol{y}} = 0.1$, which correspond to an increase in noise variance by a factor of $4\times$ compared to the training setting, the severity encoder fails to satisfy the monotonicity requirement (Fig. 9b), but still manages to provide useful estimates in some regimes (high blur). This is due to the fact that (1) the severity encoder has not encountered such high noise levels during training and thus the estimates are highly inaccurate, and (2) the high noise 'over-powers' the effect of blurring and therefore it becomes more challenging to distinguish between images corrupted by similar blur amounts. Finally, as we increase the test-time measurement noise to $\sigma_{\boldsymbol{y}} = 0.2$ (variance $16\times$ training setup), the severity encoder completely breaks down, and produces a similar severity estimate for all blur amounts (Fig. 9c).

To get an even more detailed understanding of the limitations of the severity encoder, we introduce the notion of *ordering accuracy* that qualitatively captures how well the monotonicity requirement is satisfied.

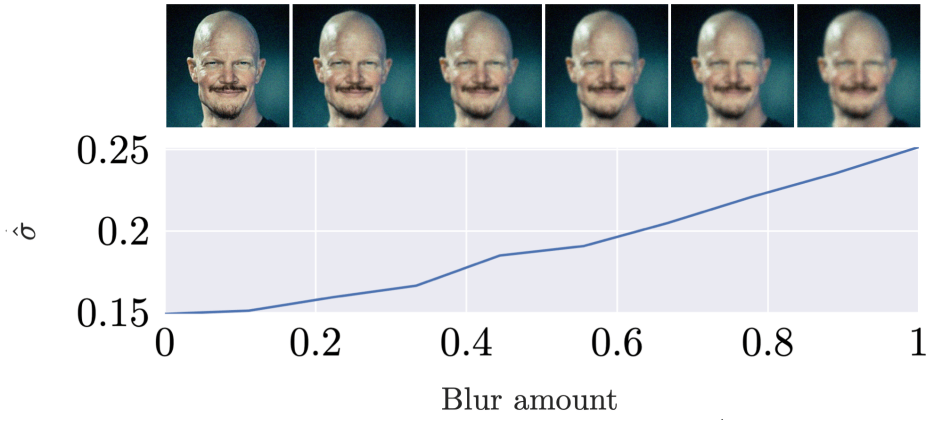
Definition 1 (Ordering accuracy)**.** *Let $\boldsymbol{x}_0$ be a clean image, and $\boldsymbol{y}_{s_1}, \boldsymbol{y}_{s_2}, ..., \boldsymbol{y}_{s_n}$ a sequence of corrupted observations of $\boldsymbol{x}_0$, such that $\boldsymbol{y}_{s_i} = \mathcal{A}(\boldsymbol{x}_0; s_i) + \boldsymbol{n}$, and $0 \leq s_1 < s_2 < ... < s_n \leq 1$, where s_i parameterizes the severity level of the degradation (higher the more severe). Let $\hat{\sigma}_i := \hat{\sigma}(\boldsymbol{y}_{s_i})$ denote the corresponding estimates of degradation severity from the severity encoder, and*

$$g(\boldsymbol{y}_{s_i}, \boldsymbol{y}_{s_j}; \boldsymbol{x}_0) = \begin{cases} 1 & \textit{if } \ \textit{sign}(s_i - s_j) = \textit{sign}(\hat{\sigma}_i - \hat{\sigma}_j) \\ 0 & \textit{otherwise} \end{cases} \tag{19}$$

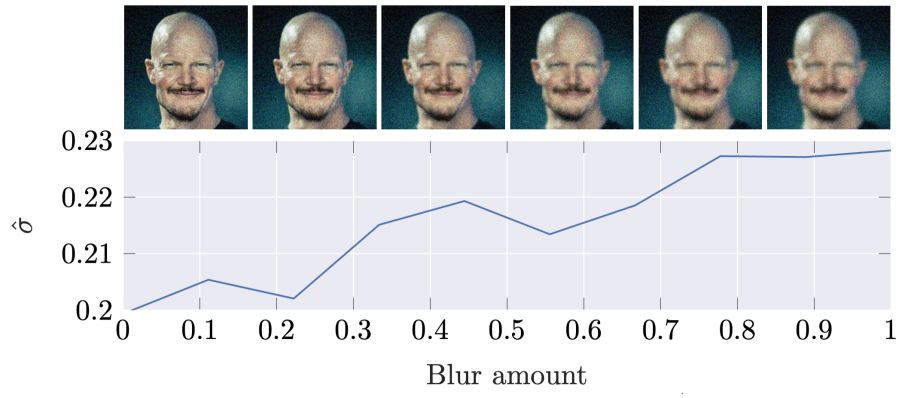
Then, the ordering accuracy (OA) of the severity encoder on sample $\boldsymbol{x}_0$ and severity levels $s_1, ..., s_n$ under the degradation model $\mathcal{A}(\cdot, s)$ is

$$\textit{OA}(\boldsymbol{x}_0; \boldsymbol{y}_{s_1}, ..., \boldsymbol{y}_{s_n}) := \frac{1}{n(n-1)} \sum_{i \neq j} g(\boldsymbol{y}_{s_i}, \boldsymbol{y}_{s_j}; \boldsymbol{x}_0) \tag{20}$$

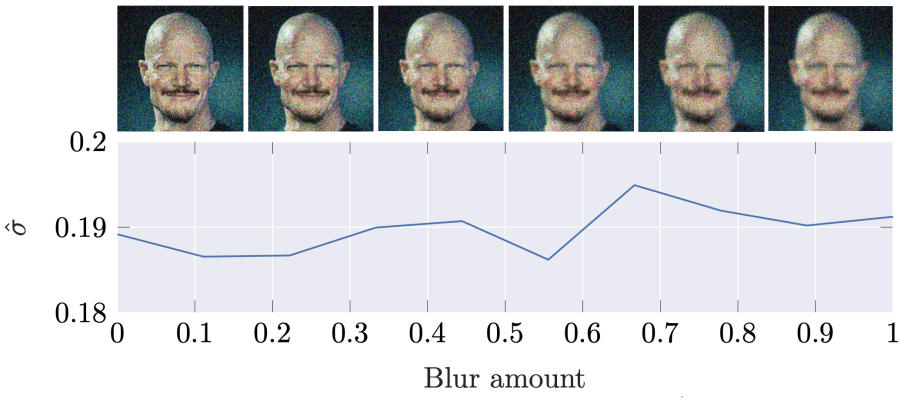
Intuitively, the ordering accuracy takes a sequence of increasingly degraded images and the corresponding severity estimates from the encoder and evaluates the portion of pairs of degraded images for which the ordering of ground truth degradation



(a) Measurement noise: $\sigma_y = 0.05$ (no test-time perturbation).



(b) Measurement noise: $\sigma_y = 0.1$



(c) Measurement noise: $\sigma_y = 0.2$

Figure 9: Effect of perturbation in the measurement noise level in test time on the predictions of the severity encoder. We expect the predicted severity to be a non-decreasing function of the true blur amount, as depicted in Figure 9a. At high noise level perturbations the severity encoder violates the monotonicity requirement (Figure 9b) and eventually breaks down completely at extreme measurement noise variances (Figure 9c).

level matches the ordering of predicted severities. Ideally, if the monotonicity requirement is satisfied, we have ordering accuracy of 1 for every sample. On the other hand, if the severity estimates are generated randomly we have ordering accuracy of 0.5 (the probability that any pair has the correct ordering is 0.5).

We leverage the notion of ordering accuracy to evaluate the degradation of severity encoding performance under test-time measurement noise perturbations. In particular, we consider the Gaussian deblurring task and select $n = 10$ blur kernel stds over a uniform grid in $[0.0, 3.0]$, with $s_1 = 0$ corresponding to no blurring and $s_n = 1$ corresponding to blurring with blur kernel std of 3.0. In Figure 10, we plot how the average ordering accuracy evolves as we change the measurement noise level evaluated on 100 validation samples on the CelebA dataset. We observe that for noise levels up to the training noise level, the severity encoder has perfect ordering accuracy. That is, we can expect reliable performance on noise levels that are different but lower than the training measurement noise. This property indicates that it may be beneficial to train the severity encoder at higher noise levels than the expected test-time noise in order to increase robustness. As the noise level perturbation increases, severity encoding performance drops, approaching random guessing (ordering accuracy of 0.5) at extreme noise variances.

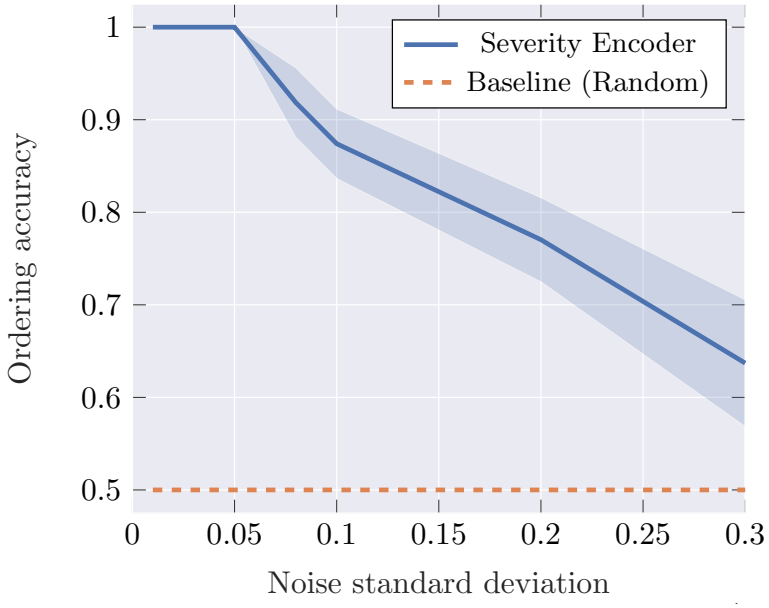


Figure 10: Evaluation of severity encoding performance under test-time noise level perturbation. The training noise level is $\sigma_{\boldsymbol{y}} = 0.05$. The severity encoder is robust to arbitrary reductions in noise level in test time (no drop in ordering accuracy). As the noise level perturbation increases, performance degrades and approaches random guessing (ordering accuracy of 0.5). Mean over 100 validation samples and 3 measurement generating random seeds is plotted, with $\pm$ standard deviation over random seeds as shaded area.

Robustness of Flash-Diffusion performance –Next, we investigate the impact of noise level mismatch in test-time on the reconstruction performance of adaptive solvers imbued with Flash-Diffusion. In particular, we perform experiments on the FFHQ dataset with varying Gaussian blur and fixed measurement noise of $\sigma_{\boldsymbol{y}} = 0.05$ in severity encoder training. We pick ReSample as our baseline solver as it yields the best performance among non-adaptive solvers in our experiments. As the stopping criteria for hard data consistency (τ in (Song et al., 2023)) depends on the measurement noise, we expect degraded performance when the true noise level differs from the one used to tune the hyperparameters. We vary the measurement noise in test time and compare the reconstruction performance of `Flash`(ReSample) to ReSample and SwinIR. We depict the results of the robustness study in Figure 8. We observe that both in terms of perceptual and distortion metrics, the performance of `Flash`(ReSample) degrades more gracefully compared to its non-adaptive counterpart and SwinIR in most cases (with the exception of lower than expected noise and distortion metrics). We hypothesize that this additional robustness is due to the sample-by-sample adaptivity of Flash-Diffusion. As the noise level increases, our method can automatically adapt by increasing the number of diffusion steps due to higher degradation severity predictions from the severity encoder.

The adaptation to higher measurement noise levels in test time is demonstrated in Figure 11. Even though the severity encoder has not been trained on noise levels other than $\sigma_{\boldsymbol{y}} = 0.05$, its predictions are still useful enough to adapt the

sampling trajectory to the changes in noise level, up to a limit. At noise levels significantly higher than in the training setting ($\sigma_{\boldsymbol{y}} = 0.1$ and above), the performance of the severity encoder breaks down and is unable to provide accurate severity estimations. This phenomenon is corroborated by our observation of degrading ordering accuracy at high noise levels (see Figure 10 and Figure 9b).

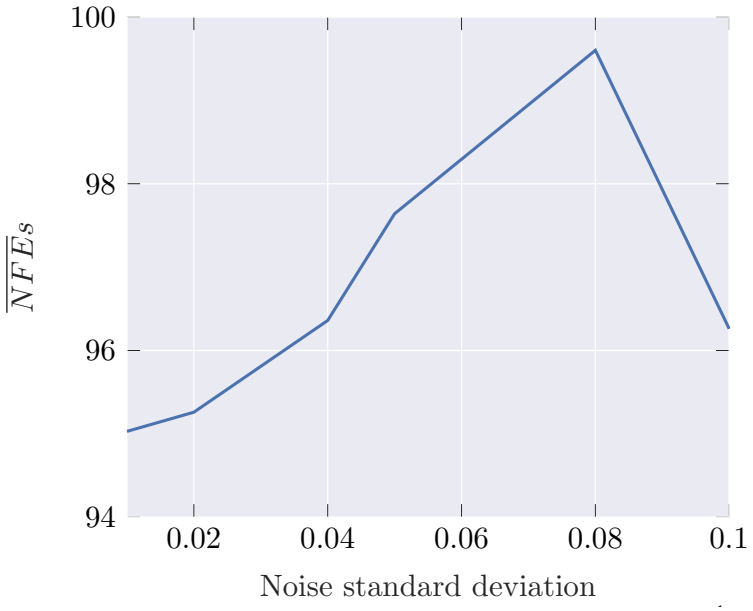


Figure 11: Due to the adaptivity of Flash-Diffusion, the length of the sampling trajectory is automatically scaled to the input noise level in test time, even in case of noise levels different from the training setup. At extreme noise discrepancies ($\sigma_{\boldsymbol{y}} = 0.1$) the severity encoder breaks down and unable to adapt to the increased degradation severity.

C.2. Robustness against forward model mismatch

Our method relies on a severity encoder that has been trained on paired data of clean and degraded images under a specific forward model. We simulate a mismatch between the severity encoder fine-tuning operator and test-time operator in order to investigate the robustness of our technique with respect to forward model perturbations. In particular, we run the following experiments to assess the test-time shift: 1) we train the encoder on varying Gaussian blur and test on nonlinear blur and 2) we train the encoder on nonlinear blur and test on varying Gaussian blur.

Robustness of severity encoding – First, following our methodology in Appendix C.1, we investigate how severity encoding performance degrades if the image degradation forward model is different from what the severity encoder has been trained on. We train the severity encoder on nonlinear blur and evaluate it on Gaussian blur of varying amounts on the CelebA dataset. We expect the predicted severity to be a non-decreasing function of the true blur amount. However, as Figure 12 demonstrates, the monotonicity requirement is violated due to the forward model mismatch. In particular, the severity encoder fails to predict the degradation severities of lightly blurred images. However, we observe satisfactory performance at high blur amounts (above 0.3 in Fig. 12). We hypothesize that at high blur levels, Gaussian blur and nonlinear blur become similar, thus the severity encoder trained on either may provide acceptable predictions for both. We further support this hypothesis by evaluating the ordering accuracy on a validation set of 100 CelebA images. The results are summarized in Table 4. We observe that the ordering accuracy is poor (mean OA $= 0.7704$) when evaluated across the full range of blur levels. However, it is near-perfect when low blur levels are excluded from the evaluation (mean OA $= 0.9778$). This experiment highlights the severity encoder's potential for generalization to unseen forward models, which opens up interesting future directions for fast adaptation to new tasks.

Robustness of Flash-Diffusion performance – We investigate the robustness of `Flash`(Latent-DPS) to forward model mismatch in test time. The results on the FFHQ test set are in Table 5. We observe minimal loss in performance when nonlinear blur encoder is used for reconstructing images corrupted by Gaussian blur. For the nonlinear deblurring task, using Gaussian blur encoder results in a more significant drop in the performance, while still providing acceptable reconstructions.

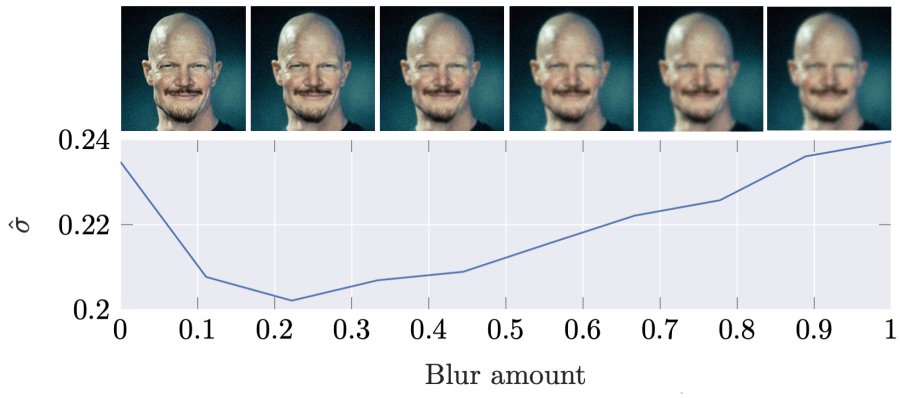


Figure 12: Effect of perturbation in the forward model in test time on the predictions of the severity encoder. We train the severity encoder on nonlinear blur and evaluate it on Gaussian blur of varying amounts. We expect the predicted severity to be a non-decreasing function of the true blur amount. However, due to the forward model mismatch the monotonicity requirement is violated for low blur levels.

Blur range	**Avg. OA**
[0, 1]	0.7704
[0.3, 1]	0.9778

Table 4: Ordering accuracy of a severity encoder trained on nonlinear blur and evaluated on Gaussian blur of varying blur amount. The severity encoder provides fair estimates even under forward model mismatch in the high blur regime.

These results are expected, as Gaussian blur can be thought of as a special case of the nonlinear blur model we consider. Therefore even when the encoder is swapped, it can provide meaningful mean and error estimation. However, the Gaussian blur encoder has never been trained on images corrupted by nonlinear blur. As such, the mean estimate is worse, resulting in a larger performance drop. Note that we did not re-tune the hyper-parameters in these experiments and doing so may potentially alleviate the loss in performance.

	Gaussian Deblurring (varying)				**Nonlinear Deblurring**			
Method	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)
`Flash`(Latent-DPS) + Gaussian blur encoder	29.16	0.8191	0.2241	29.467	25.36	0.7238	0.3416	54.90
`Flash`(Latent-DPS) + NL blur encoder	28.96	0.8129	0.2362	30.34	27.22	0.7705	0.2694	36.92

Table 5: Robustness experiments on the FFHQ test split.

D. Experiments with Accelerated Baseline Solvers

DDIM (Song et al., 2021a), an accelerated sampling scheme, has demonstrated comparable generation performance to DDPM samplers with significantly lower compute requirements, often able to reduce the number of reverse diffusion steps from $N = 1000$ to $N = 100$ in the context of image synthesis. However, to the best of our knowledge, similar acceleration has not been successfully translated to latent domain diffusion solvers without significant drop in reconstruction performance. In this section, we investigate:

1. **Non-adaptive accelerated solvers–** Can baseline solvers coupled with accelerated DDIM sampling match the reconstruction quality of Flash-Diffusion adaptive counterparts with the same number of reverse diffusion steps?
2. **Adaptive accelerated solvers–** Can Flash-Diffusion yield further speedup on top of accelerated solvers while maintaining reconstruction quality?

Non-adaptive accelerated solvers– We investigate whether the reduction in sampling steps of Flash-Diffusion can be

simply achieved using complete sampling trajectories with DDIM updates (without the shortcut initialization from the severity encoder). The DDIM updates in latent domain take the form

$$\boldsymbol{z}_{i-1} = \sqrt{\bar{\alpha}_{i-1}}\hat{\boldsymbol{z}}_0(\boldsymbol{z}_t) + \sqrt{1 - \bar{\alpha}_{i-1} - \delta^2\tilde{\beta}_i^2}\epsilon_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i) + \delta\tilde{\beta}_i^2\boldsymbol{\epsilon}, \tag{21}$$

where $\hat{\boldsymbol{z}}_0(\boldsymbol{z}_t)$ is the estimator of the posterior mean based on Tweedie's formula, $\tilde{\beta}_i = \sqrt{(1-\bar{\alpha}_{i-1})/(1-\bar{\alpha}_i)}\sqrt{1-\bar{\alpha}_i/\bar{\alpha}_{i-1}}$ and δ parameter controlling the stochasticity of the sampling process. For $\delta = 0.0$ one obtains a fully deterministic inference procedure, while $\delta = 1.0$ corresponds to the ancestral sampling of DDPM. Here, we experiment with a variant of Latent-DPS, where we replace DDPM sampling ($N = 1000$) with DDIM sampling of $N = 100$, which we denote by Latent-DPS-DDIM($N = 100$). This closely matches the average number of steps `Flash`(Latent-DPS) uses on the varying Gaussian blur task on FFHQ, but instead of jumping ahead in the reverse process at initialization, it starts from the i.i.d. Gaussian distribution similar to DPS. We tune the data consistency step size η in (5) by grid search over $[0.1, 0.5, 1.0, 1.5]$ on a validation split of 100 images of FFHQ and select the value $\eta = 1.5$. The results are shown in Table 6. We find that Latent-DPS-DDIM($N = 100$) far underperforms `Flash`(Latent-DPS) and other diffusion-based reconstruction techniques. In fact, we observe a large variation in reconstruction quality, from high fidelity reconstructions to images with heavy artifacts. We hypothesize that this effect is due to large errors in the posterior mean estimate in the early stages of diffusion resulting in inaccurate posterior sampling, which is further compounded by the large variance in sample degradations.

Finally, we perform a similar experiment on ReSample, a more advanced solver that does not rely on DPS, but instead leverages hard data consistency. Table 7 shows that ReSample performs significantly better than the naive Latent-DPS algorithm at high DDIM-acceleration ($N = 50$), maintaining similar reconstruction quality to the non-accelerated original algorithm ($N = 500$). However, the Flash-Diffusion adaptive variant of ReSample far outperforms the DDIM-accelerated non-adaptive variant while using comparable number of reverse diffusion steps. These experiments further support that straightforward application of accelerated DDIM sampling in latent diffusion solvers, leveraging the same compute budget in inference time, cannot compete with Flash-Diffusion adaptive variants in terms of reconstruction quality.

Adaptive accelerated solvers– Our proposed framework can be paired with any diffusion-based sampling technique as long as the SNR can be evaluated analytically for any time step (see (3)). Here, we investigate the performance of Flash-Diffusion, when paired with highly accelerated DDIM sampling. We run experiments on the FFHQ test set under Gaussian blur degradation of varying magnitude matching the setup in Section 4. We investigate the performance of adaptive versions of Latent-DPS-DDIM($N = 20$) and Latent-DPS-DDIM($N = 100$) for both $\delta = 0.0$ (deterministic) and $\delta = 1.0$ (DDPM-like). We tune the data consistency step size η in (5) by grid search over $[0.1, 0.5, 1.0, 1.5]$ on a validation split of 100 images of FFHQ for each combination of N and δ. We select the value $\eta = 1.5$ as it provides the best LPIPS across all experiments.

The results are shown in Table 8. First, we observe that combining Flash-Diffusion with DDIM sampling results in vastly accelerated sampling: we obtain reasonable reconstructions in 10.4 ($N = 100$) or 2.5 ($N = 20$) diffusion steps on average over the test set. Image quality in terms of distortion and perceptual metrics is comparable with other diffusion based solvers under DDIM sampling with N=100 steps, but with compute cost reduced by an order of magnitude. However, reconstruction quality degrades significantly when deploying Latent-DPS-DDIM($N = 20$). We hypothesize that the LDPS step size scheduling defined in (5) needs to be tailored to DDIM sampling in this regime, which is an interesting direction for future work. Finally, we note that we find Flash-Diffusion performance to be robust to the setting of δ in the DDIM sampler, as we obtain very similar results for $\delta = 0.0$ and $\delta = 1.0$.

	Gaussian Deblurring (varying)				**Nonlinear Deblurring**			
Method	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)
Latent-DPS-DDIM($N = 100$), $\delta = 0.0$	16.62	0.4192	0.6224	233.74	13.52	0.3694	0.6472	278.25
Latent-DPS-DDIM($N = 100$), $\delta = 1.0$	18.23	0.4771	0.5474	129.67	14.96	0.4071	0.5533	86.68

Table 6: Latent-DPS without severity encoding using DDIM sampling ($N = 100$) on the FFHQ test split.

E. Adaptive DDIM Encoding Initialization

So far we have initialized the sampling trajectory with a noise-corrected latent estimate obtained from our severity encoder, where the noise correction is aimed at suppressing structure in the estimation error. This initialization scheme is analogous

Method	Gaussian Deblurring (varying)					Nonlinear Deblurring					Random Inpainting				
	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	NFE
ReSample(DDIM, N=500)	28.77	0.8219	0.2587	81.96	500	26.62	0.7318	0.2838	68.57	500	27.51	0.7892	0.2460	63.39	500
ReSample(DDIM, N=50)	28.15	0.7910	0.2641	83.05	50	25.59	0.6810	0.3697	118.40	50	26.95	0.7525	0.3154	115.42	50
`Flash`(ReSample(DDIM, N=500))	**29.07**	**0.8330**	**0.2383**	**74.76**	49.90	**26.88**	**0.7660**	**0.2667**	**64.57**	67.81	**28.13**	**0.8260**	**0.2045**	**56.67**	52.09

Table 7: Comparison of ReSample and its DDIM-accelerated variant with Flash-Diffusion adaptation on the FFHQ test split.

Method	Avg. NFE	$\delta = 0.0$				$\delta = 1.0$			
		PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)	PSNR(↑)	SSIM(↑)	LPIPS(↓)	FID(↓)
`Flash`(Latent-DPS)	100.1	-	-	-	-	29.16	0.8191	0.2241	29.46
`Flash`(Latent-DPS-DDIM($N = 100$))	10.4	28.04	0.7956	0.2537	32.08	27.70	0.7901	0.2550	31.83
`Flash`(Latent-DPS-DDIM($N = 20$))	2.5	26.05	0.7482	0.3204	57.77	26.09	0.7504	0.3168	54.88

Table 8: Comparison of DDIM-accelerated sampling techniques in the Flash-Diffusion framework on FFHQ varying Gaussian deblurring.

to stochastic encoding (Song et al., 2023). In this section, we investigate further techniques to initialize the reverse diffusion process in our framework in a sample-adaptive fashion.

With DDIM, the generative process can be reversed deterministically by simply running

$$\boldsymbol{z}_{i+1} = \sqrt{\alpha_i}\boldsymbol{z}_i - \sqrt{1-\alpha_i}\sqrt{1-\bar{\alpha}_i}s_{\boldsymbol{\theta}}(\boldsymbol{z}_i, i), \tag{22}$$

which can be obtained from reversing (21). Here, $s_{\boldsymbol{\theta}}$ denotes the learned score (scaled version of $\epsilon_{\boldsymbol{\theta}}$). When starting from the clean latent $\boldsymbol{z}_0$ and running (22) for $t = 0, .., N-1$ one obtains a deterministic encoding of $\boldsymbol{z}_0$ and the process is referred to as DDIM encoding (Preechakul et al., 2022). STSL (Rout et al., 2023) leverages DDIM encoding of a naive reconstruction ($\mathcal{E}_0(\boldsymbol{A}^T\boldsymbol{y})$, where ε_0 denotes the original encoder pretrained with the LDM) to obtain an initialization, where the DDIM forward process is run from $t = 0$ to T. This initialization scheme can be combined with Flash adaptation, that is we can perform "partial" DDIM encoding from $t = 0$ to t_{start}, where t_{start} is the starting time index predicted by severity encoding. We refer to this technique as *adaptive DDIM encoding*. We perform experiments for the following scenarios:

- **STSL initialization with adaptation:** we use adaptive DDIM encoding of $\mathcal{E}_0(\boldsymbol{A}^T\boldsymbol{y})$ by running the DDIM forward process for t_{start} steps, where t_{start} is obtained from the severity encoder. We do not use noise correction.
- **Flash initialization with DDIM encoding:** we use adaptive DDIM encoding of $\hat{z}$ (latent estimate from severity encoder) by running the DDIM forward process for t_{start} steps, where t_{start} is obtained from the severity encoder. We do not use noise correction.

We choose Latent-DPS with DDIM sampling ($N = 500$, $\eta = 0.0$) as our baseline solver (same setting as described in Appendix D) in order to make the reverse process as close to DDIM sampling as possible within our framework. We show results on the FFHQ test set with varying Gaussian blur and random inpainting operators in Table 9. The mean t_{start} predicted by severity encoding on the dataset is 49.91 for deblurring and 52.08 for inpainting.

Initialization	Gaussian Deblurring (varying)				Random Inpainting			
	SSIM(↑)	PSNR(↑)	LPIPS(↓)	FID(↓)	SSIM(↑)	PSNR(↑)	LPIPS(↓)	FID(↓)
$\hat{\boldsymbol{z}}$ + noise correction (ours)	0.8136	28.96	0.2328	56.91	0.8447	29.11	0.2010	57.24
$\hat{\boldsymbol{z}}$ + DDIM encoding	0.8488	30.05	0.2497	82.54	0.8605	29.38	0.2132	68.23
$\mathcal{E}_0(\boldsymbol{A}^T\boldsymbol{y})$ + DDIM encoding	0.7906	28.12	0.3201	92.08	0.6337	24.87	0.4302	135.4

Table 9: Results on the FFHQ dataset using various initialization schemes.

First, we observe that STSL initialization with adaptation performs worse than our vanilla Flash initialization. This is natural, as we leverage a supervised model to obtain an initial reconstruction that is of better quality than the unsupervised

$\mathcal{E}_0(\boldsymbol{A}^T\boldsymbol{y})$ initialization. Furthermore, we suspect that since the autoencoder of LDM has been pretrained predominantly on clean images, it may compress $\boldsymbol{A}^T\boldsymbol{y}$ inefficiently.

Second, we find that adaptive DDIM encoding on top of the latent estimate $\hat{\boldsymbol{z}}$ obtained from our severity encoder performs surprisingly well. Even though this initialization falls short in terms of perceptual quality compared to vanilla Flash initialization with noise correction, it achieves superior distortion (PSNR, SSIM). This hints at blurrier, but highly reliable reconstructions faithful to the original measurement. This experiment suggests that Flash-Diffusion combined with adaptive DDIM encoding initialization may be very promising. In particular, by tuning a hyperparameter that controls the number of DDIM forward steps for encoding (similar to our noise correction parameter c) these results may be further improved. We leave this exploration for future work.

We note that a negative side-effect of using adaptive DDIM encoding in our setting is that it may limit the baseline solver to DDIM variants, as it is unclear how this initialization scheme performs when combined with arbitrary sampling techniques.

F. Additional Efficiency Experiments

We perform additional experiments on the nonlinear blurring task to demonstrate the efficiency of Flash-Diffusion compared to non-adaptive baselines. Figure 13 indicates that the adaptive sampling of Flash-Diffusion achieves better perceptual quality than any fixed-length sampling trajectory. The adaptivity is further supported by the wide spread of sampling steps utilized by Flash-Diffusion.

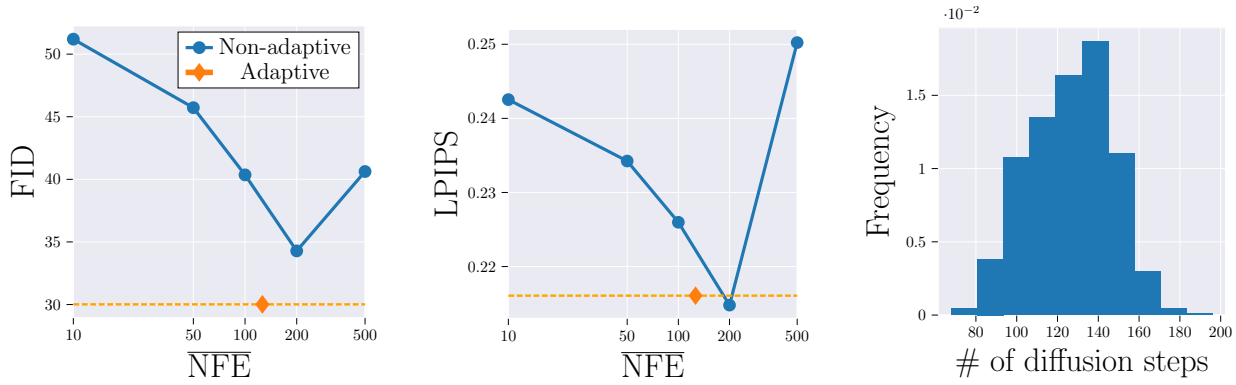


Figure 13: Comparison of `Flash`(Latent-DPS) with the non-adaptive counterpart with fixed starting time for all samples (nonlinear deblurring, CelebA-HQ). Left and center: We plot reconstruction quality as a function of the number of diffusion steps for the non-adaptive variant with various fixed starting times. The data point corresponding to our adaptive method depicts the average number of diffusion steps due to adaptive starting time. Flash-Diffusion achieves the best FID and near-optimal LPIPS compared to any choice of non-adaptive starting time. Right: Histogram of predicted starting times for `Flash`(Latent-DPS).

G. Limitations

We identify the following limitations of our framework.

1. The proposed reconstruction method requires degraded-clean image pairs for fine-tuning the severity encoder. Fine-tuning has to be performed separately for each degradation, thus the method is less flexible than DPS and similar diffusion solvers. However, we argue that the fine-tuning step has fairly low cost and greatly pays off in reconstruction performance and efficiency. Moreover, investigating the viability of a general degradation severity estimator that works for arbitrary degradations (under some reasonable assumptions) is an interesting direction for future research.
2. The proposed severity estimation method breaks down at high noise perturbations compared to the training settings and when there is a significant test-time shift in the forward model.
3. The assumption of i.i.d. Gaussian prediction error provides a simple way to estimate the severity, however does not necessarily hold in practice. We believe that more realistic error models can further improve our technique, which we leave for future work.

H. Additional Samples for Visual Comparison

Figure 14: Reconstructed samples from the FFHQ test set on varying Gaussian blur with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

Figure 15: Reconstructed samples from the FFHQ test set on fixed Gaussian blur with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

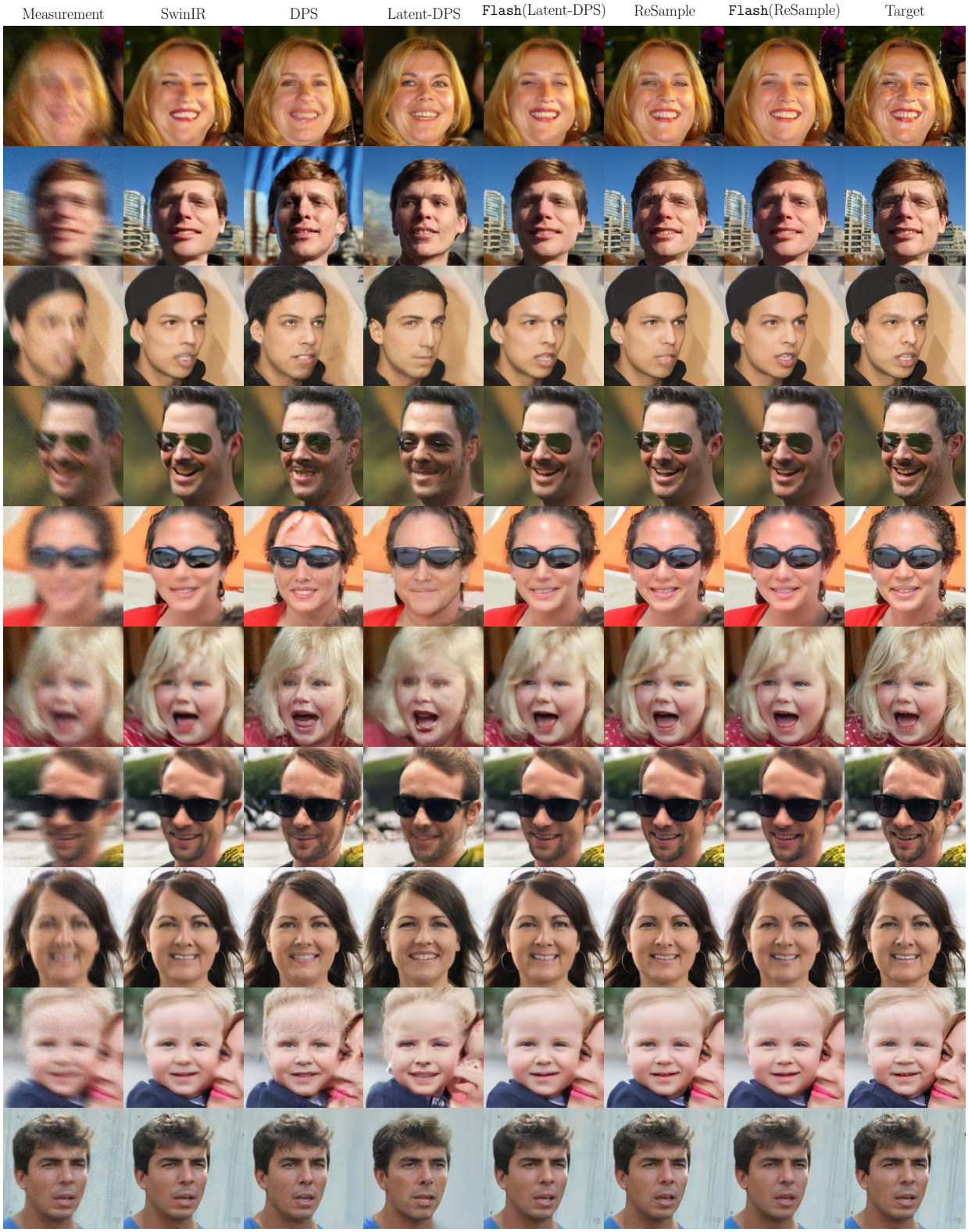


Figure 16: Reconstructed samples from the FFHQ test set on nonlinear motion blur with random kernels and additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

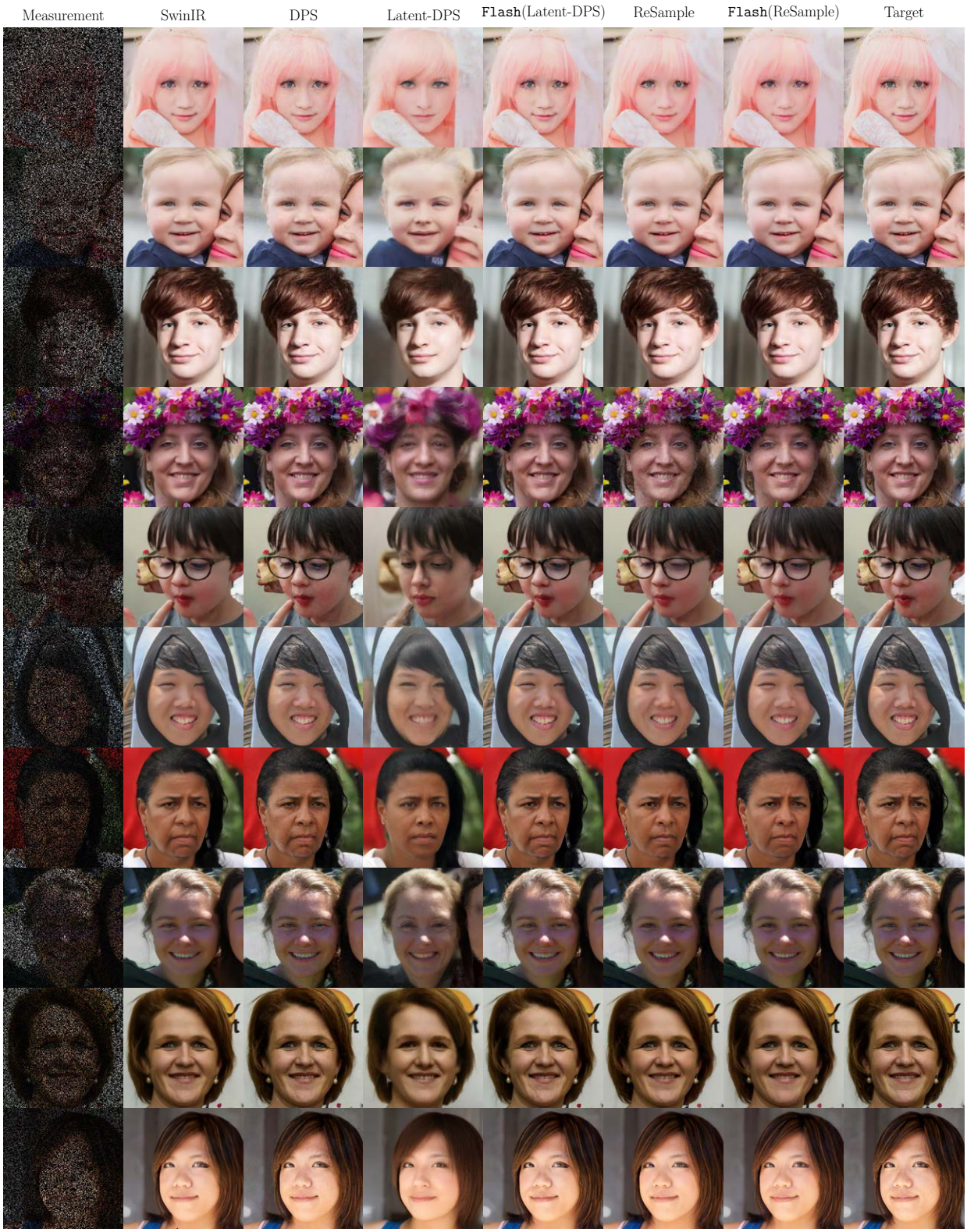


Figure 17: Reconstructed samples from the FFHQ test set on varying amounts of random inpainting with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

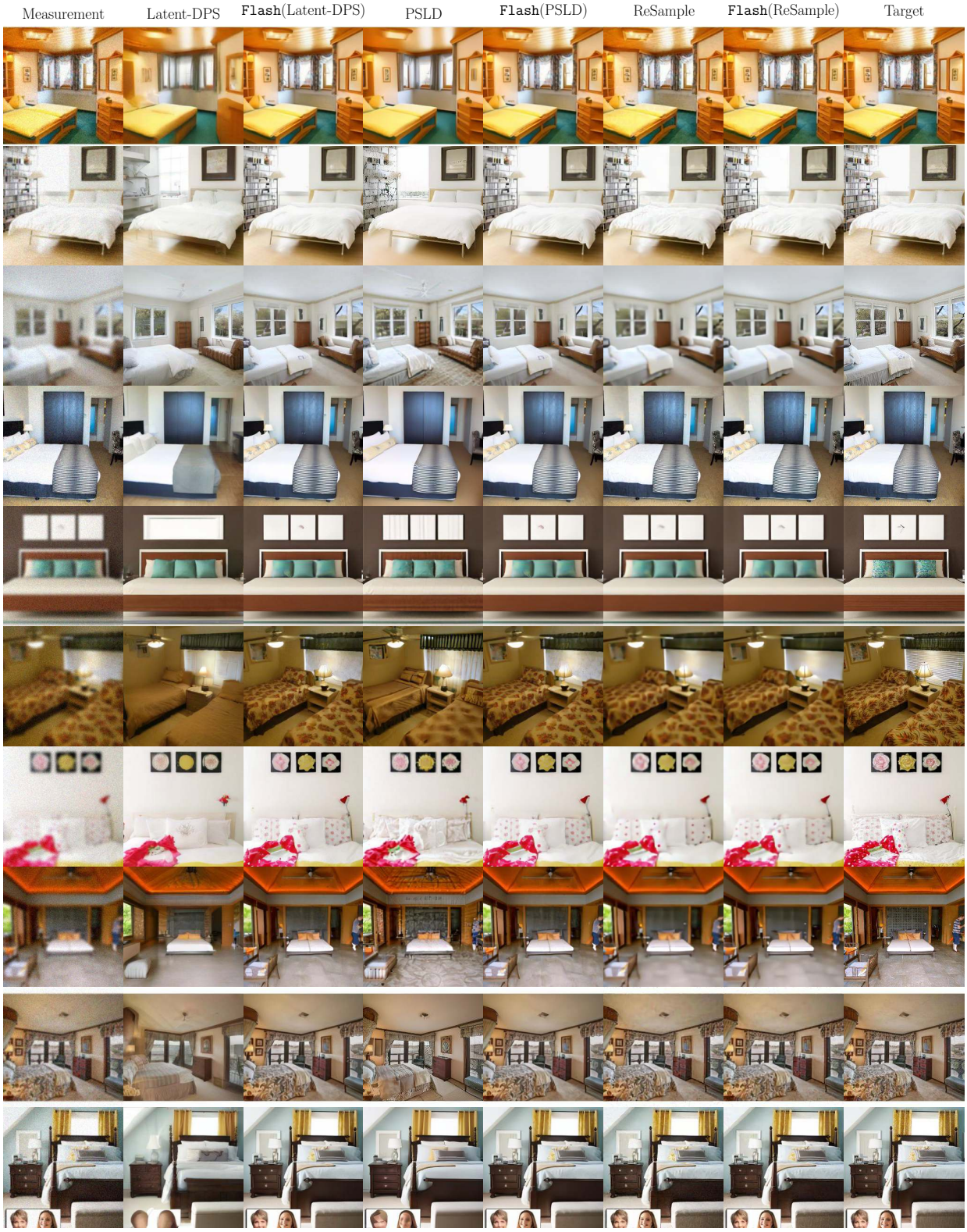


Figure 18: Reconstructed samples from the LSUN Bedrooms test set on varying Gaussian blur with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

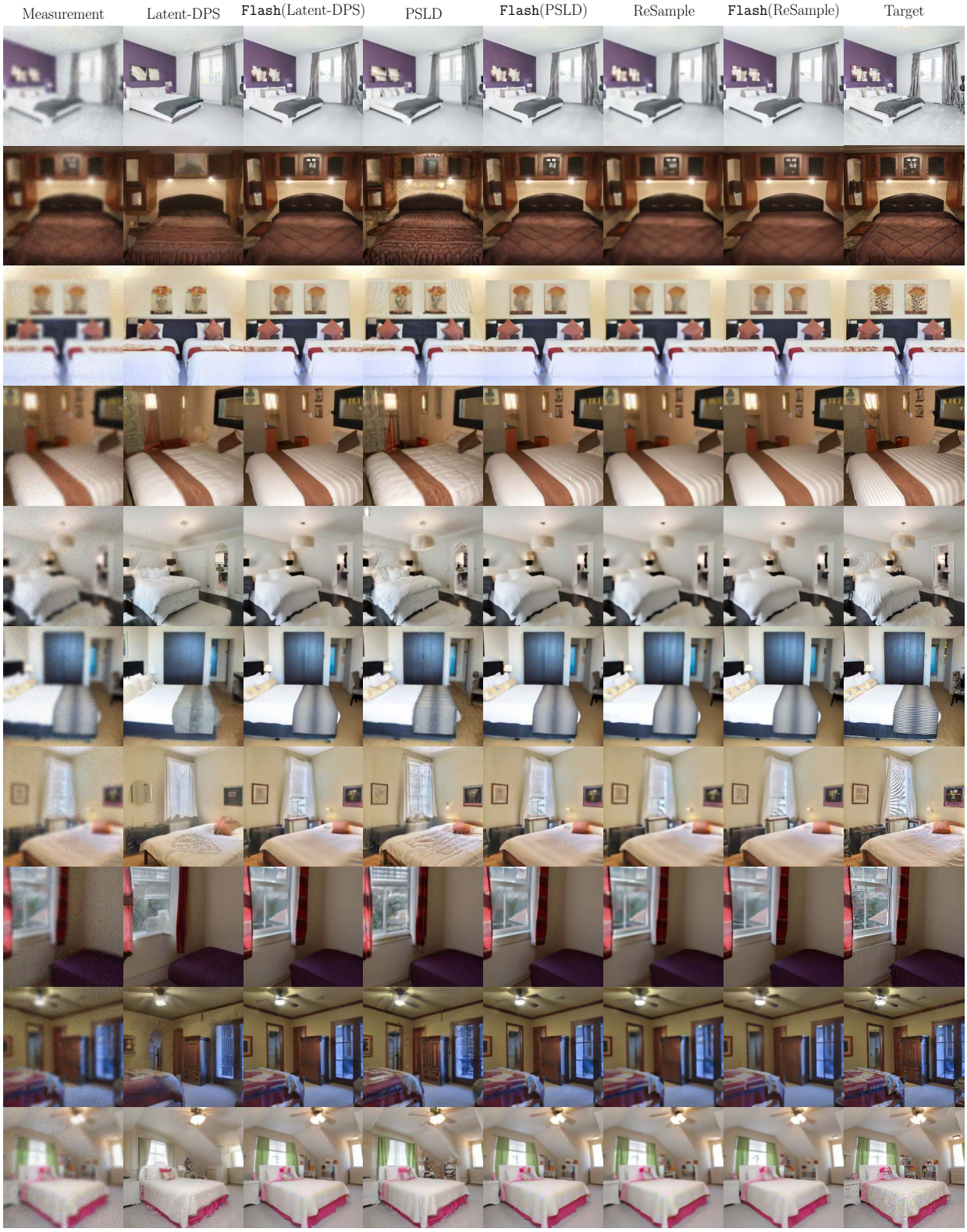


Figure 19: Reconstructed samples from the LSUN Bedrooms test set on fixed Gaussian blur with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

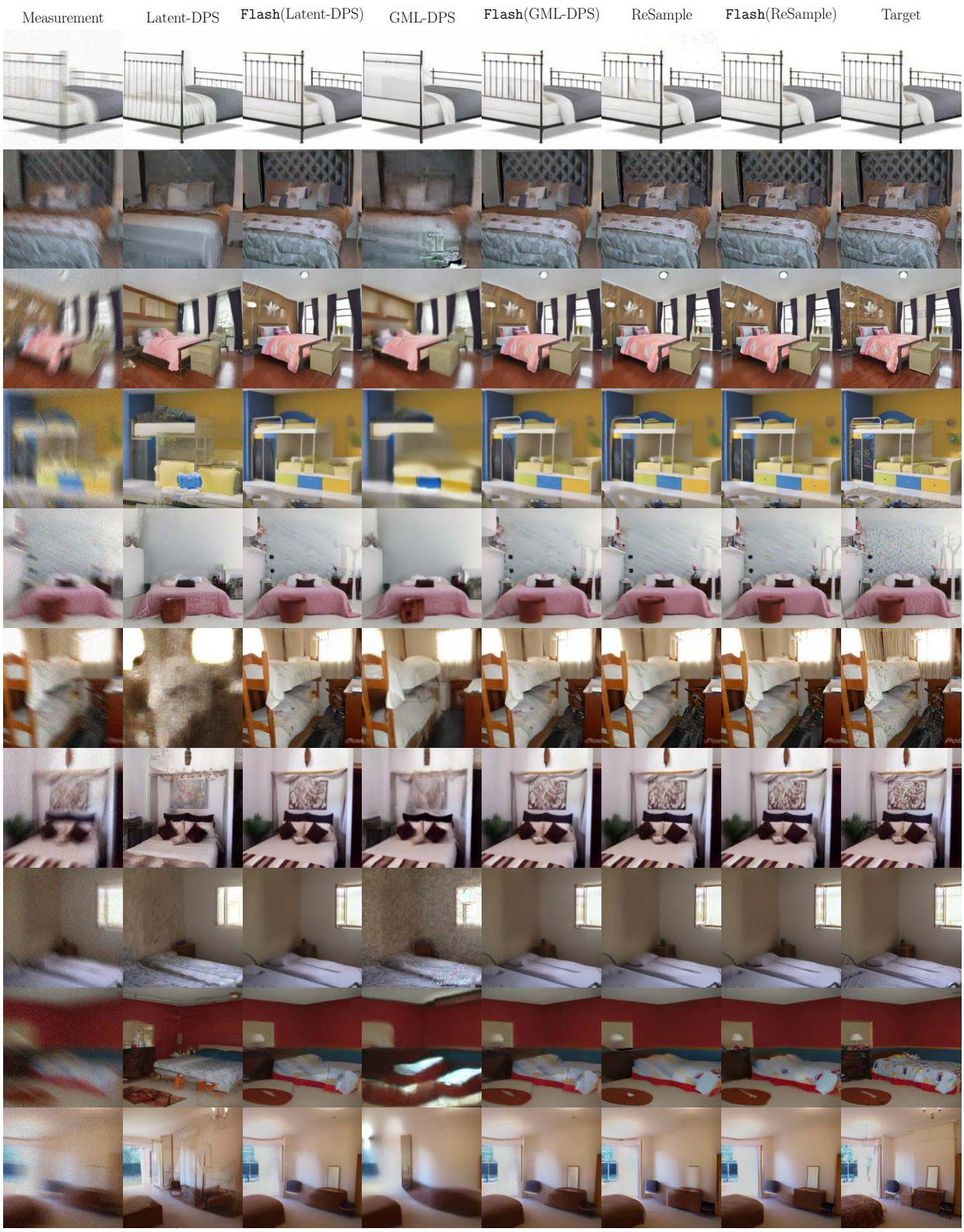


Figure 20: Reconstructed samples from the LSUN Bedrooms test set on nonlinear motion blur with random kernels and additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.

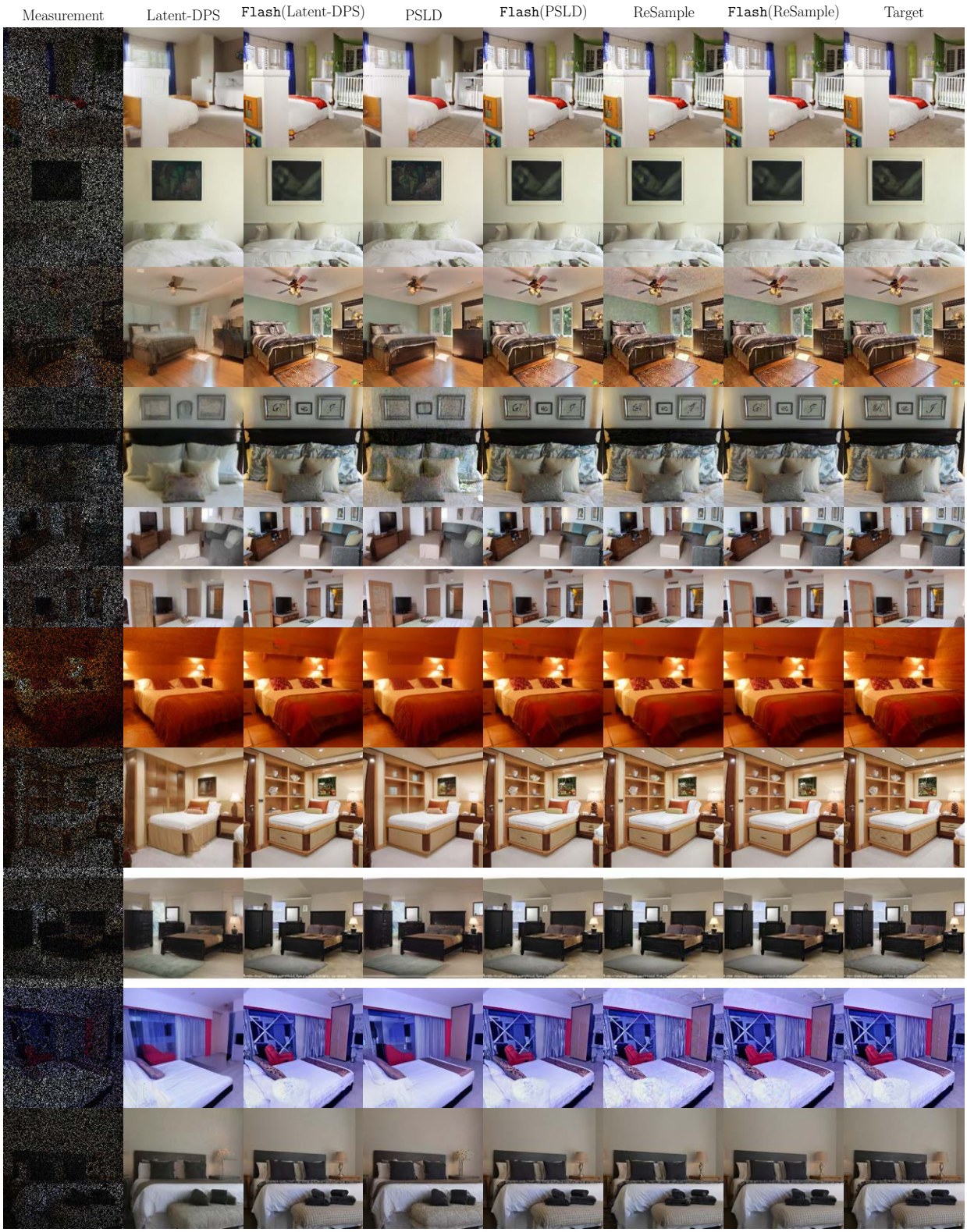


Figure 21: Reconstructed samples from the LSUN Bedrooms test set on varying amounts of random inpainting with additive noise $\sigma_{\boldsymbol{y}} = 0.05$. The samples are not cherry-picked.