Improving Diffusion Inverse Problem Solving with Decoupled Noise Annealing

Supplementary Material

A. DAPS with Pixel Diffusion Models

Pixel diffusion models learn to approximate the time-dependent score function $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) \approx \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t; \sigma_t)$, where σ_t is a predefined noise schedule with $\sigma_0 = 0$ and $\sigma_T = \sigma_{\max}$. Langevin dynamics is used in DAPS due to its simplicity and flexibility. However, other advanced MCMC sampler such as Hamiltonian Monte Carlo (HMC) can be used to enhance the efficiency and quality. We will give detailed derivation in Appendix A.1. We also include a discussion of using the Metropolis Hasting algorithm, a gradient-free MCMC sampling method in Appendix A.2.

According to the Langevin dynamics updating rule Eq. (4), HMC updating rule Eq. (9) and Metropolis Hasting updating rule Eq. (14), we summarize the algorithm of DAPS with pixel diffusion models in Algorithm 1 with Langevin dynamics, HMC or Metropolis Hasting. We also include an ablation study of different methods in Appendix I.

A.1. Sampling with Hamiltonian Monte Carlo

Hamiltonian Monte Carlo [4] is designed to efficiently sample from complex, high-dimensional probability distributions by leveraging concepts from Hamiltonian dynamics to reduce random walk behavior and improve exploration of the target distribution. Recall our goal is to sample from the proposal distribution $p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$. We define the Hamiltonian $H(\mathbf{x}_0, \mathbf{p}) = U(\mathbf{x}_0) + K(\mathbf{p})$, where the potential energy $U(\mathbf{x}_0) = -\log p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$ and the kinetic energy $K(\mathbf{p}) = \frac{1}{2}\mathbf{p}^\top \mathbf{p}$. We then simulate Hamiltonian dynamics to propose new samples.

1. Start with the estimator from probability flow ODE $\hat{\mathbf{x}}_0^{(0)} = \hat{\mathbf{x}}_0(\mathbf{x}_{x_t})$ and sample a initial momentum $\mathbf{p}^{(0)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ at diffusion time step t.
2. Simulate Hamiltonian Dynamics with step size η_t, momentum damping factor γ_t, and number of steps N, where the updating rule for $j = 1, \cdots, N$,

$$(\hat{\mathbf{x}}_0^{(j+1)}, \mathbf{p}^{(j+1)}) = \text{Hamiltonian-Dynamics}(\hat{\mathbf{x}}_0^{(j)}, \mathbf{p}^{(j)}), \tag{9}$$

is given by:

$$\mathbf{p}^{(j+1)} = (1 - \gamma_t \eta_t) \cdot \mathbf{p}^{(j)} - \eta_t \nabla_{x_k} U(x_k) + \sqrt{2\gamma_t \eta_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{10}$$

$$\hat{\mathbf{x}}_0^{(j+1)} = \hat{\mathbf{x}}_0^{(j)} + \eta_t \mathbf{p}^{(j+\frac{1}{2})}. \tag{11}$$

Empirically, Hamiltonian Monte Carlo efficiently explores the target distribution $p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$ and requires less number of steps N to achieve similar performance compared to Langevin dynamics, allowing for more efficient sampling. We find HMC can speed up LatentDAPS with large-scale text-conditioned LDMs quite a bit. We will discuss this more in Appendix B.1.

A.2. Sampling with Metropolis Hasting algorithm

The Metropolis-Hastings algorithm [37] is a MCMC method used to sample from a target distribution $p(\mathbf{x})$, especially when direct sampling is infeasible. It works by constructing a Markov chain whose stationary distribution corresponds to $p(\mathbf{x})$. Here we discuss how to sample from $p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$ using Metropolis Hasting under *Gaussian approximation*. We adopt the Gaussian kernel as the proposal distribution which is symmetry,

$$q(\mathbf{x} \to \mathbf{x}') = \mathcal{N}(\mathbf{x}'; \mathbf{x}, \eta_t^2 \boldsymbol{I}) \tag{12}$$

where η_t is a hyperparameter to control the strength of the perturbation at diffusion time step t. Then we give the process of the Metropolis Hasting algorithm.

1. Start with the estimator from probability flow ODE $\hat{\mathbf{x}}_0^{(0)} = \hat{\mathbf{x}}_0(\mathbf{x}_{x_t})$.
2. Perturb the current position by zero-centered Gaussian noise with standard deviation η_t, and number of steps N, for $j = 1, \cdots, N$,

$$\mathbf{x}_{prop}^{(j)} = \hat{\mathbf{x}}_0^{(j)} + \eta_t \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{13}$$

Algorithm 1 Decoupled Annealing Posterior Sampling (DAPS)

Require: Score model $\boldsymbol{s_\theta}$, measurement $\mathbf{y}$, noise schedule σ_t, $(t_i)_{i\in\{0,\dots,N_A\}}$.
  Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$.
  for $i = N_A, N_A - 1, \dots, 1$ **do**
    Initial $\mathbf{p}^{(0)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for HMC only
    Compute $\hat{\mathbf{x}}_0^{(0)} = \hat{\mathbf{x}}_0(\mathbf{x}_{t_i})$ by solving the probability flow ODE in Eq. (48) with $\boldsymbol{s_\theta}$
    for $j = 0, \dots, N-1$ **do**
      Langevin dynamics:

$$\hat{\mathbf{x}}_0^{(j+1)} \leftarrow \hat{\mathbf{x}}_0^{(j)} + \eta_t \Big(\nabla_{\hat{\mathbf{x}}_0} \log p(\hat{\mathbf{x}}_0^{(j)} | \mathbf{x}_{t_i}) + \nabla_{\hat{\mathbf{x}}_0} \log p(\mathbf{y} | \hat{\mathbf{x}}_0^{(j)}) \Big) + \sqrt{2\eta_t} \boldsymbol{\epsilon}_j, \ \boldsymbol{\epsilon}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}).$$

      or HMC:

$$(\hat{\mathbf{x}}_0^{(j+1)}, \mathbf{p}^{(j+1)}) \leftarrow \text{Hamiltonian-Dynamics}(\hat{\mathbf{x}}_0^{(j)}, \mathbf{p}^{(j)}),$$

      or Metropolis Hasting:

$$\hat{\mathbf{x}}_0^{(j+1)} \leftarrow \text{Metropolis-Hasting}(\hat{\mathbf{x}}_0^{(j)})$$

    end for
    Sample $\mathbf{x}_{t_{i-1}} \sim \mathcal{N}(\hat{\mathbf{x}}_0^{(N)}, \sigma_{t_{i-1}}^2 \boldsymbol{I})$.
  end for
  Return $\mathbf{x}_0$

3. Update position by,

$$\hat{\mathbf{x}}_0^{(j+1)} = \text{Metropolis-Hasting}(\hat{\mathbf{x}}_0^{(j)}) = \begin{cases} \mathbf{x}_{prop}^{(j)} & \text{w.p. } \alpha(\hat{\mathbf{x}}_0^{(j)} \to \mathbf{x}_{prop}^{(j)}) \\ \hat{\mathbf{x}}_0^{(j)} & \text{otherwise} \end{cases}, \tag{14}$$

where the acceptance probability $\alpha(\hat{\mathbf{x}}_0^{(j)} \to \mathbf{x}_{prop}^{(j)})$ is given by,

$$\alpha(\hat{\mathbf{x}}_0^{(j)} \to \mathbf{x}_{prop}^{(j)}) = \min\left(1, \frac{p(\mathbf{x}_{prop}^{(j)} \mid \mathbf{x}_t, \mathbf{y})}{p(\hat{\mathbf{x}}_0^{(j)} \mid \mathbf{x}_t, \mathbf{y})}\right) \approx \min\left(1, \frac{\mathcal{N}(\mathbf{x}_{prop}^{(j)}; \mathbf{x}_t, \sigma_t^2 \boldsymbol{I}) \cdot \mathcal{N}(\mathbf{y}; \mathcal{A}(\mathbf{x}_{prop}^{(j)}), \beta_{\mathbf{y}}^2 \boldsymbol{I})}{\mathcal{N}(\hat{\mathbf{x}}_0^{(j)}; \mathbf{x}_t, \sigma_t^2 \boldsymbol{I}) \cdot \mathcal{N}(\mathbf{y}; \mathcal{A}(\hat{\mathbf{x}}_0^{(j)}), \beta_{\mathbf{y}}^2 \boldsymbol{I})}\right) \tag{15}$$

Empirically, Metropolis Hasting usually performs worse than Langevin dynamics and HMC and less efficient but more flexible to tasks that don't have access to the gradient of the data likelihood, $\nabla_{\mathbf{x}_0} \log p(\mathbf{y} \mid \mathbf{x}_0)$, *i.e.* the forward function $\mathcal{A}$ is non-differentiable or its gradient is inaccessible. We will discuss more about such tasks in Appendix C.

B. DAPS with Latent Diffusion Models

Latent diffusion models (LDMs) [40] operate the denoising process not directly on the pixel space, but in a low-dimensional latent space. LDMs have been known for their superior performance and computational efficiency in high-dimensional data synthesis. In this section, we show that our method can be naturally extended to sampling with latent diffusion models.

Let $\mathcal{E} : \mathbb{R}^n \to \mathbb{R}^k$ and $\mathcal{D} : \mathbb{R}^k \to \mathbb{R}^n$ be a pair of encoder and decoder. Let $\mathbf{z}_0 = \mathcal{E}(\mathbf{x}_0)$ where $\mathbf{x}_0 \sim p(\mathbf{x}_0)$, and $p(\mathbf{z}; \sigma)$ be the noisy distribution of latent vector $\mathbf{z}$ by adding Gaussian noises of variance σ^2 to the latent code of clean data. We have the following Proposition according to the factor graph in Fig. 3b.

Proposition 2. *Suppose $\mathbf{z}_{t_1}$ is sampled from the measurement conditioned time-marginal $p(\mathbf{z}_{t_1} \mid \mathbf{y})$, then*

$$\mathbf{z}_{t_2} \sim \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})}[\mathcal{N}(\mathcal{E}(\mathbf{x}_0), \sigma_{t_2}^2 \boldsymbol{I})] \tag{16}$$

satisfies the measurement conditioned time-marginal $p(\mathbf{z}_{t_2} \mid \mathbf{y})$. Moreover,

$$\mathbf{z}_{t_2} \sim \mathbb{E}_{\mathbf{z}_0 \sim p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})}[\mathcal{N}(\mathbf{z}_0, \sigma_{t_2}^2 \boldsymbol{I})]. \tag{17}$$

also satisfies the measurement conditioned time-marginal $p(\mathbf{z}_{t_2} \mid \mathbf{y})$.

Remark. We can efficiently sample from $p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})$ using similar strategies as in Sec. 3, *i.e.*,

$$\mathbf{x}_0^{(j+1)} = \mathbf{x}_0^{(j)} + \eta \cdot \left(\nabla_{\mathbf{x}_0^{(j)}} \log p(\mathbf{x}_0^{(j)} \mid \mathbf{z}_{t_1}) + \nabla_{\mathbf{x}_0^{(j)}} \log p(\mathbf{y} \mid \mathbf{x}_0^{(j)}) \right) + \sqrt{2\eta} \boldsymbol{\epsilon}_j. \tag{18}$$

We further approximate $p(\mathbf{x}_0^{(j)} \mid \mathbf{z}_{t_1})$ by $\mathcal{N}(\mathbf{x}_0^{(j)}; \mathcal{D}(\hat{\mathbf{z}}_0(\mathbf{z}_{t_1})), r_{t_1}^2 \boldsymbol{I})$, where $\hat{\mathbf{z}}_0(\mathbf{z}_{t_1})$ is computed by solving the (unconditional) probability flow ODE with a latent diffusion model $\boldsymbol{s}_{\boldsymbol{\theta}}$ starting at $\mathbf{z}_{t_1}$. The Langevin dynamics can then be rewritten as

$$\mathbf{x}_0^{(j+1)} = \mathbf{x}_0^{(j)} - \eta \cdot \nabla_{\mathbf{x}_0^{(j)}} \left(\frac{\|\mathbf{x}_0^{(j)} - \mathcal{D}(\hat{\mathbf{z}}_0(\mathbf{z}_{t_1}))\|^2}{2r_{t_1}^2} + \frac{\|\mathcal{A}(\mathbf{x}) - \mathbf{y}\|^2}{2\beta_{\mathbf{y}}^2} \right) + \sqrt{2\eta} \boldsymbol{\epsilon}_j. \tag{19}$$

On the other hand, we can also decompose $p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y}) \approx p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}) p(\mathbf{y} \mid \mathbf{z}_0)$ and run Langevin dynamics directly on the latent space,

$$\mathbf{z}_0^{(j+1)} = \mathbf{z}_0^{(j)} + \eta \cdot \left(\nabla_{\mathbf{z}_0^{(j)}} \log p(\mathbf{z}_0^{(j)} \mid \mathbf{z}_{t_1}) + \nabla_{\mathbf{z}_0^{(j)}} \log p(\mathbf{y} \mid \mathbf{z}_0^{(j)}) \right) + \sqrt{2\eta} \boldsymbol{\epsilon}_j. \tag{20}$$

Assuming $p(\mathbf{z}_0^{(j)} \mid \mathbf{z}_{t_1})$ by $\mathcal{N}(\mathbf{z}_0^{(j)}; \hat{\mathbf{z}}_0(\mathbf{z}_t), r_{t_1}^2 \boldsymbol{I})$, we derive another Langevin MCMC updating rule in the latent space,

$$\mathbf{z}_0^{(j+1)} = \mathbf{z}_0^{(j)} - \eta \cdot \nabla_{\mathbf{z}_0^{(j)}} \left(\frac{\|\mathbf{z}_0^{(j)} - \hat{\mathbf{z}}_0(\mathbf{z}_{t_1})\|^2}{2r_{t_1}^2} + \frac{\|\mathcal{A}(\mathcal{D}(\mathbf{z}_0^{(j)})) - \mathbf{y}\|^2}{2\beta_{\mathbf{y}}^2} \right) + \sqrt{2\eta} \boldsymbol{\epsilon}_j. \tag{21}$$

Both approaches are applicable to our posterior sampling algorithm. We summarize DAPS with latent diffusion models in Algorithm 2. It is worth mentioning that when employing the *Gaussian approximation* for $p(\mathbf{x}_0 \mid \mathbf{x}_t)$, pixel-space Langevin dynamics typically results in higher approximation errors compared to latent-space Langevin dynamics but offers significantly faster computation. A practical approach is to utilize pixel-space Langevin dynamics during the early stages (*i.e.*, for $N_A \geqslant i > M$) to balance approximation accuracy with efficiency, transitioning to latent-space Langevin dynamics in the later stages (*i.e.*, for $M \geqslant i \geqslant 1$) to achieve improved sample quality, where M is a hyperparameter that trade-off efficiency and approximation accuracy. For simplicity, we set $M = N_A$ throughout our experiments, and leave the exploration of this hyperparameter as a future direction.

Algorithm 2 Latent Diffusion Decoupled Annealing Posterior Sampling (LatentDAPS) with Langevin Dynamics

Require: Latent space score model s_{θ}, measurement $\mathbf{y}$, noise schedule σ_t, $t_{i\in\{0,\dots,N_A\}}$, encoder $\mathcal{E}$ and decoder $\mathcal{D}$.
Sample $\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \sigma_T^2 \boldsymbol{I})$.
for $i = N_A, N_A - 1, \dots, 1$ **do**
Compute $\hat{\mathbf{z}}_0^{(0)} = \hat{\mathbf{z}}_0(\mathbf{z}_{t_i})$ by solving the probability flow ODE in Eq. (48) with $\mathbf{s}_{\boldsymbol{\theta}}$.
Pixel space Langevin dynamics:
$\hat{\mathbf{x}}_0^{(0)} = \mathcal{D}(\hat{\mathbf{z}}_0^{(0)})$
for $j = 0, \dots, N-1$ **do**

$$\hat{\mathbf{x}}_0^{(j+1)} \leftarrow \hat{\mathbf{x}}_0^{(j)} + \eta_t \Big(\nabla_{\hat{\mathbf{x}}_0} \log p(\hat{\mathbf{x}}_0^{(j)} \mid \mathbf{z}_{t_i}) + \nabla_{\hat{\mathbf{x}}_0} \log p(\mathbf{y} \mid \hat{\mathbf{x}}_0^{(j)}) \Big) + \sqrt{2\eta_t} \boldsymbol{\epsilon}_j, \ \boldsymbol{\epsilon}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{22}$$

end for
$\hat{\mathbf{z}}_0^{(N)} = \mathcal{E}(\hat{\mathbf{x}}_0^{(N)})$
Or, latent space Langevin dynamics:
for $j = 0, \dots, N-1$ **do**

$$\hat{\mathbf{z}}_0^{(j+1)} \leftarrow \hat{\mathbf{z}}_0^{(j)} + \eta_t \Big(\nabla_{\hat{\mathbf{z}}_0^{(j)}} \log p(\hat{\mathbf{z}}_0^{(j)} \mid \mathbf{z}_{t_i}) + \nabla_{\hat{\mathbf{z}}_0^{(j)}} \log p(\mathbf{y} \mid \hat{\mathbf{z}}_0^{(j)}) \Big) + \sqrt{2\eta_t} \boldsymbol{\epsilon}_j, \ \boldsymbol{\epsilon}_j \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{23}$$

end for
Sample $\mathbf{z}_{t_{i-1}} \sim \mathcal{N}(\hat{\mathbf{z}}_0^{(N)}, \sigma_{t_{i-1}}^2 \boldsymbol{I})$.
end for
Return $\mathcal{D}(\mathbf{z}_0)$.

B.1. Sampling with Large-scaled Text-conditioned Latent Diffusion Models

Large-scale text-conditioned latent diffusion models (LDMs) [39, 40] provide a diverse and powerful prior for solving inverse problems. Given a text prompt $\mathbf{c}$, our goal is to sample from $p(\mathbf{x}_0 \mid \mathbf{c}, \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{x}_0) p(\mathbf{x}_0 \mid \mathbf{c})$, where $p(\mathbf{x}_0 \mid \mathbf{c})$ is modeled by the large-scale text-conditioned LDMs. To achieve this, we can utilize Algorithm 2 for posterior sampling.

To evaluate the performance of DAPS with text-conditioned LDMs, we use Stable Diffusion v1.5 [40] and assess it on the same eight tasks from the FFHQ 256 dataset, as described in [42, 43]. The sampling process is enhanced using classifier-free guidance for text conditioning with a guidance scale of 7.5. The quantitative results are shown in Tab. 4, and the qualitative results are presented in Fig. 8. A significant advantage of text-conditioned LDMs is their ability to flexibly control posterior sampling using text prompts. As illustrated in Fig. 9, text prompts enable effective exploration of different modes in the posterior distribution. Finally, we summarize the text prompts used for each task in Tab. 5.

Table 4. **Quantitative evaluation on FFHQ 256×256 of LatentDAPS with Stable Diffusion v1.5.**. The value shows the mean over 100 images, and all tasks assume the measurement noise level $\beta_{\mathbf{y}} = 0.01$. Numbers for PSLD are copied from the original paper.

Method	Super Resolution 4×		Inpaint (Box)		Inpaint (Random)		Gaussian deblurring		Motion deblurring		Phase retrieval		Nonlinear deblurring		High dynamic range	
	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓	PSNR↑
LatentDAPS (SD-v1.5, ours)	**0.109**	**31.43**	**0.133**	23.33	**0.064**	**34.60**	**0.160**	**30.73**	0.101	33.84	0.256	27.47	0.116	31.67	0.186	24.52
PSLD (SD-v1.5)	0.201	30.73	0.167	-	0.096	30.31	0.221	30.10	-	-	-	-	-	-	-	-

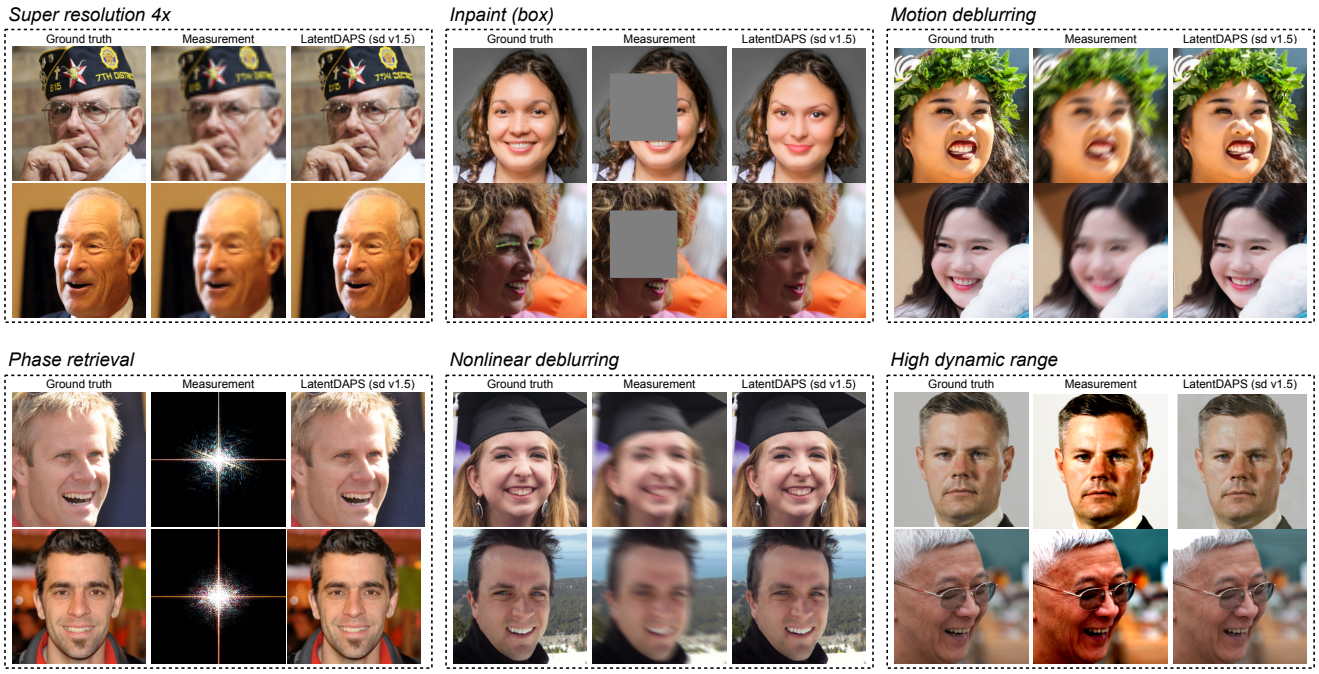


Figure 8. **Sampling results of LatentDAPS (SD v1.5) on FFHQ 256×256 images.** The sampling is enhanced with classifier-free guidance for text with guidance scale 7.5. The used text prompt is shown in Tab. 5

Figure 9. **Exploring different modes in the posterior distribution by text prompts.** Multiple sampling results using LatentDAPS (SD v1.5) were obtained for the super resolution 8× problem on a 512×512 natural image. Text prompts provided additional control over the sampled results..

Table 5. **The text prompts** used in LatentDAPS (SD v1.5).

	Text Prompts
FFHQ Evaluation	*A natural looking human face.*
Teaser Fig. 1	*Blue butterfly on white flower, green blurred background.* *Juicy burger with toppings, fresh fries, blurred restaurant background.* *Sunlit mountain reflected in a serene lake, surrounded by trees.*

Algorithm 3 Decoupled Annealing Posterior Sampling (DAPS) with Discrete Diffusion Models

Require: Score model $\boldsymbol{s_\theta}$, measurement $\mathbf{y}$, $(t_i)_{i\in\{0,\dots,N_A\}}$.
  Sample $\mathbf{x}_T \sim p_T$.
  for $i = N_A, N_A - 1, \dots, 1$ **do**
    Compute $\hat{\mathbf{x}}_0^{(0)} = \hat{\mathbf{x}}_0(\mathbf{x}_{t_i})$ by solving the reverse continuous-time Markov chain in Eq. (25) with $\boldsymbol{s_\theta}$
    for $j = 0, \dots, N-1$ **do**
      Metropolis Hasting

$$\hat{\mathbf{x}}_0^{(j+1)} \sim p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y}) \approx p(\mathbf{y} \mid \mathbf{x}_0) \exp(\|\mathbf{x}_0 - \hat{\mathbf{x}}_0(\mathbf{x}_{t_i})\|_0 / r_{t_i}).$$

    end for
    Sample $\mathbf{x}_{t_{i-1}} \sim p(\mathbf{x}_{t_{i-1}} \mid \hat{\mathbf{x}}_0^{(N)})$ following Eq. (24).
  end for
  Return $\mathbf{x}_0$

C. DAPS with Discrete Diffusion Models

Discrete diffusion models [3, 6, 33] are designed to generate categorical data over a finite support. Similar to diffusion models in the continuous space, discrete diffusion models evolve a family of distributions $p_0(\mathbf{x}), \dots, p_T(\mathbf{x})$ according to a continuous-time Markov chain. Specifically, the forward diffusion process is defined as

$$\frac{\mathrm{d}p_t}{\mathrm{d}t} = \boldsymbol{Q}_t p_t, \tag{24}$$

where $\boldsymbol{Q}_t$ are predefined transition matrices, such that p_t converges to a simple distribution like uniform distribution or a special "mask" state, as $t \to \infty$. To reverse the forward process, it suffices to learn the "concrete score", $s(\mathbf{x}, t) = \left[\frac{p_y(\mathbf{y})}{p_t(\mathbf{x})}\right]_{\mathbf{y}\neq\mathbf{x}}$. The reverse diffusion process is given by

$$\frac{\mathrm{d}p_{T-t}}{\mathrm{d}t} = \overline{\boldsymbol{Q}}_{T-t} p_{T-t}, \tag{25}$$

where $\overline{\boldsymbol{Q}}_t(\mathbf{y}, \mathbf{x}) = s(\mathbf{x}, t)\boldsymbol{Q}_t(\mathbf{x}, \mathbf{y})$ and $\overline{\boldsymbol{Q}}_t(\mathbf{x}, \mathbf{x}) = -\sum_{y\neq x} \overline{\boldsymbol{Q}}_t(\mathbf{y}, \mathbf{x})$.

Given the similarity of continuous and discrete diffusion models, we find that DAPS can be extended to perform posterior sampling with discrete diffusion models. Instead of making the *Gaussian approximation* as in continuous diffusion models, we approximate $p(\mathbf{x}_0 \mid \mathbf{x}_t)$ with an exponential distribution over Hamming distance, i.e.,

$$p(\mathbf{x}_0 \mid \mathbf{x}_t) \approx \exp(-\|\mathbf{x}_0 - \hat{\mathbf{x}}_0(\mathbf{x}_t)\|_0 / r_t), \tag{26}$$

where r_t is determined heuristically. We put the DAPS sampling algorithm for discrete diffusion models as in Algorithm 3.

Experiments on discretized MNIST dataset. We conduct experiments on the discretized MNIST dataset to demonstrate how DAPS can be applied to inverse problems on categorical data. We first discretize and flatten MNIST data to binary strings. We consider two inverse problems: 1) inpainting and 2) XOR operator. We mask out 50% pixels for the inpainting task. For the XOR task, we draw 50% random pairs from the MNIST binary strings and compute XOR over all the pairs, which serves as a highly nonlinear test case. We compare DAPS with best-of-N samples, and a recent work on discrete diffusion posterior sampling (SVDD-PM).

As shown in Tab. 6, DAPS with discrete diffusion models is able to achieve very high classification accuracy on both inverse problem tasks, outperforming both baselines by a large margin. This suggests the effectiveness of DAPS in solving inverse problems for categorical data. We leave further exploration of DAPS in discrete-state space for future work.

Table 6. **Quantitative results on discretized MNIST on two discrete inverse problems.** Both SVDD-PM and best-of-N use 20 particles for sampling. The classification accuracy is computed using a simple ConvNet model trained using MNIST training dataset.

	Inpainting		XOR	
	PSNR ↑	Accuracy (%) ↑	PSNR ↑	Accuracy (%) ↑
DAPS	$\mathbf{18.82}_{\pm 2.03}$	**97.0**	$\mathbf{20.50}_{\pm 6.40}$	**98.0**
SVDD-PM	$11.84_{\pm 2.56}$	38.0	$13.00_{\pm 2.88}$	59.0
Best-of-N	$10.56_{\pm 1.11}$	37.0	$10.50_{\pm 1.13}$	36.0

D. Proof for Propositions

Proposition 3 (Restated)**.** *Suppose $\mathbf{x}_{t_1}$ is sampled from the measurement conditioned time-marginal $p(\mathbf{x}_{t_1} \mid \mathbf{y})$, then*

$$\mathbf{x}_{t_2} \sim \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{x}_{t_1}, \mathbf{y})} \mathcal{N}(\mathbf{x}_0, \sigma_{t_2}^2 \boldsymbol{I}) \tag{27}$$

satisfies the measurement conditioned time-marginal $p(\mathbf{x}_{t_2} \mid \mathbf{y})$.

Proof. We first factorize the measurement conditioned time-marginal $p(\mathbf{x}_{t_2} \mid \mathbf{y})$ by

$$p(\mathbf{x}_{t_2} \mid \mathbf{y}) = \iint p(\mathbf{x}_{t_2}, \mathbf{x}_0, \mathbf{x}_{t_1} \mid \mathbf{y}) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{x}_{t_1} \tag{28}$$

$$= \iint p(\mathbf{x}_{t_1} \mid \mathbf{y}) p(\mathbf{x}_0 \mid \mathbf{x}_{t_1}, \mathbf{y}) p(\mathbf{x}_{t_2} \mid \mathbf{x}_0, \mathbf{x}_{t_1}, \mathbf{y}) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{x}_{t_1}. \tag{29}$$

Recall the probabilistic graphical model in Fig. 3a. $\mathbf{x}_{t_2}$ is independent of $\mathbf{x}_{t_1}$ and $\mathbf{y}$ given $\mathbf{x}_0$. Therefore,

$$p(\mathbf{x}_{t_2} \mid \mathbf{x}_0, \mathbf{x}_{t_1}, \mathbf{y}) = p(\mathbf{x}_{t_2} \mid \mathbf{x}_0). \tag{30}$$

As a result,

$$p(\mathbf{x}_{t_2} \mid \mathbf{y}) = \iint p(\mathbf{x}_{t_1} \mid \mathbf{y}) p(\mathbf{x}_0 \mid \mathbf{x}_{t_1}, \mathbf{y}) p(\mathbf{x}_{t_2} \mid \mathbf{x}_0) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{x}_{t_1} \tag{31}$$

$$= \mathbb{E}_{\mathbf{x}_{t_1} \sim p(\mathbf{x}_{t_1} \mid \mathbf{y})} \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{x}_{t_1}, \mathbf{y})} p(\mathbf{x}_{t_2} \mid \mathbf{x}_0) \tag{32}$$

$$= \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{x}_{t_1}, \mathbf{y})} \mathcal{N}(\mathbf{x}_{t_2}; \mathbf{x}_0, \sigma_{t_2}^2 \boldsymbol{I}), \tag{33}$$

given $\mathbf{x}_{t_1}$ is drawn from the measurement conditioned time-marginal $p(\mathbf{x}_{t_1} \mid y)$. □

Proposition 4 (Restated)**.** *Suppose $\mathbf{z}_{t_1}$ is sampled from the measurement conditioned time-marginal $p(\mathbf{z}_{t_1} \mid \mathbf{y})$, then*

$$\mathbf{z}_{t_2} \sim \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})} \mathcal{N}(\mathcal{E}(\mathbf{x}_0), \sigma_{t_2}^2 \boldsymbol{I}) \tag{34}$$

satisfies the measurement conditioned time-marginal $p(\mathbf{z}_{t_2} \mid \mathbf{y})$. Moreover,

$$\mathbf{z}_{t_2} \sim \mathbb{E}_{\mathbf{z}_0 \sim p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})} \mathcal{N}(\mathbf{z}_0, \sigma_{t_2}^2 \boldsymbol{I}). \tag{35}$$

also satisfies the measurement conditioned time-marginal $p(\mathbf{z}_{t_2} \mid \mathbf{y})$.

Proof. We first factorize the measurement conditioned time-marginal $p(\mathbf{z}_{t_2} \mid \mathbf{y})$ by

$$p(\mathbf{z}_{t_2} \mid \mathbf{y}) = \iint p(\mathbf{z}_{t_2}, \mathbf{x}_0, \mathbf{z}_{t_1} \mid \mathbf{y}) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{z}_{t_1} \tag{36}$$

$$= \iint p(\mathbf{z}_{t_1} \mid \mathbf{y}) p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y}) p(\mathbf{z}_{t_2} \mid \mathbf{x}_0, \mathbf{z}_{t_1}, y) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{z}_{t_1}. \tag{37}$$

Recall the probabilistic graphical model in Fig. 3b. $\mathbf{z}_{t_2}$ is independent of $\mathbf{z}_{t_1}$ and $\mathbf{y}$ given $\mathbf{z}_0$, while $\mathbf{z}_0$ is determined only by $\mathbf{x}_0$. Therefore,

$$p(\mathbf{z}_{t_2} \mid \mathbf{x}_0, \mathbf{z}_{t_1}, \mathbf{y}) = p(\mathbf{z}_{t_2} \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{z}_{t_2}; \mathcal{E}(\mathbf{x}_0), \sigma_{t_2}^2 \boldsymbol{I}). \tag{38}$$

Hence,

$$p(\mathbf{z}_{t_2} \mid \mathbf{y}) = \iint p(\mathbf{z}_{t_1} \mid \mathbf{y}) p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y}) p(\mathbf{z}_{t_2} \mid \mathbf{x}_0) \mathrm{d}\mathbf{x}_0 \mathrm{d}\mathbf{x}_{t_1} \tag{39}$$

$$= \mathbb{E}_{\mathbf{z}_{t_1} \sim p(\mathbf{z}_{t_1} \mid \mathbf{y})} \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})} p(\mathbf{z}_{t_2} \mid \mathbf{x}_0) \tag{40}$$

$$= \mathbb{E}_{\mathbf{x}_0 \sim p(\mathbf{x}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})} \mathcal{N}(\mathbf{z}_{t_2}; \mathcal{E}(\mathbf{x}_0), \sigma_{t_2}^2 \boldsymbol{I}), \tag{41}$$

assuming $\mathbf{z}_{t_1}$ is drawn from $p(\mathbf{z}_{t_1} \mid \mathbf{y})$.

Moreover, we can also factorize $p(\mathbf{z}_{t_2} \mid \mathbf{y})$ by

$$p(\mathbf{z}_{t_2} \mid \mathbf{y}) = \iint p(\mathbf{z}_{t_2}, \mathbf{z}_{t_1}, \mathbf{z}_0 \mid \mathbf{y}) \mathrm{d}\mathbf{z}_0 \mathrm{d}\mathbf{z}_{t_1} \tag{42}$$

$$= \iint p(\mathbf{z}_{t_1} \mid \mathbf{y}) p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y}) p(\mathbf{z}_{t_2} \mid \mathbf{z}_0, \mathbf{z}_{t_1}, \mathbf{y}) \mathrm{d}\mathbf{z}_0 \mathrm{d}\mathbf{z}_{t_1} \tag{43}$$

$$= \iint p(\mathbf{z}_{t_1} \mid \mathbf{y}) p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y}) p(\mathbf{z}_{t_2} \mid \mathbf{z}_{t_1}) \mathrm{d}\mathbf{z}_0 \mathrm{d}\mathbf{z}_{t_1}. \tag{44}$$

The last equation is again derived directly from Fig. 3b. Given that $\mathbf{z}_{t_1}$ is sampled from $p(\mathbf{z}_{t_1} \mid \mathbf{y})$, we have that

$$p(\mathbf{z}_{t_2} \mid \mathbf{y}) = \mathbb{E}_{\mathbf{z}_0 \sim p(\mathbf{z}_0 \mid \mathbf{z}_{t_1}, \mathbf{y})} \mathcal{N}(\mathbf{z}_{t_2}; \mathbf{z}_0, \sigma_{t_2}^2 \boldsymbol{I}). \tag{45}$$

□

E. Discussions

E.1. Sampling Efficiency

The sampling efficiency is a crucial aspect of inverse problem solvers. The time cost of diffusion model-based methods is highly dependent on the number of neural function evaluations (NFE). Here in Tab. 7 we show the NFE of the default setting of some pixel space baseline methods and DAPS with different configurations. In Fig. 13, we show the quantitative evaluation of DAPS with different NFE. As we can see, DAPS can achieve relatively much better performance than baselines with small NFE.

Table 7. **Sampling time of DAPS on phase retrieval task with FFHQ 256.** The nonparallel single image sampling time on the FFHQ 256 dataset with 1 NVIDIA A100-SXM4-80GB GPU. The time depends may differ slightly in different runs.

Configuration	ODE Steps	Annealing Steps	NFE	Seconds/Image
DPS	-	-	1000	35
DDRM	-	-	20	2
RED-diff	-	-	1000	47
DAPS-50	2	25	50	4
DAPS-100	2	50	100	7
DAPS-200	2	100	200	13
DAPS-400	4	100	400	17
DAPS-1K	5	200	1000	37
DAPS-2K	8	250	2000	61
DAPS-4K	10	400	4000	108

E.2. Limitations and Future Extension

Though DAPS achieves significantly better performance on inverse problems like phase retrieval, there are still some limitations.

First, we only adopt a very naive implementation of the latent diffusion model with DAPS, referred to as LatentDAPS. However, some recent techniques [42, 45] have been proposed to improve the performance of posterior sampling with latent diffusion models. Specifically, one main challenge is that $\mathbf{x}_{0|\mathbf{y}}$ obtained by Langevin dynamics in pixel space might not lie in the manifold of clean images. This could further lead to a sub-optimal performance for autoencoders in diffusion models since they are only trained with clean data manifold.

Furthermore, we only implement DAPS with a decreasing annealing scheduler, but the DAPS framework can support any scheduler function σ_t^A as long as $\sigma_0^A = 0$. A non-monotonic scheduler has the potential of providing DAPS with more power to explore the solution space.

Finally, we utilize fixed NFE for the ODE solver. However, one could adjust it automatically. For example, less ODE solver NFE for smaller t in later sampling steps. We would leave the discussions above as possible future extensions.

E.3. Broader Impacts

We anticipate that DAPS can offer a new paradigm for addressing challenging real-world inverse problems using diffusion models. DAPS tackles these problems by employing a diffusion model as a general denoiser, which learns to model a powerful prior data distribution. This approach could significantly enrich the array of methods available to the inverse problem-solving community. However, it is important to note that DAPS might generate biased samples if the diffusion model is trained on biased data. Therefore, caution should be exercised when using DAPS in bias-sensitive scenarios.

F. Experimental Details

F.1. Inverse Problem Setup

Most inverse problems are implemented in the same way as introduced in [13]. However, for inpainting with random pixel masks, motion deblurring, and nonlinear deblurring, we fix a certain realization for fair comparison by using the same random seeds for mask generation and blurring kernels. Moreover, for phase retrieval, we adopt a slightly different version as follows:

$$\mathbf{y} \sim \mathcal{N}(|\mathbf{FP}(0.5\mathbf{x}_0 + 0.5)|, \beta_{\mathbf{y}}^2 \boldsymbol{I}), \tag{46}$$

which normalize the data to lies in range $[0, 1]$ first. Here $\mathbf{F}$ and $\mathbf{P}$ are discrete Fourier transformation matrices and oversampling matrices with ratio k/n. Same as [13], we use an oversampling factor $k = 2$ and $n = 8$. We normalize input x_0 by shifting its data range from $[-1, 1]$ to $[0, 1]$ to better fit practical settings, where the measured signals are usually non-negative.

The measurement for high dynamic range reconstruction is defined as

$$\mathbf{y} \sim \mathcal{N}(\text{clip}(\alpha\mathbf{x}_0, -1, 1), \beta_{\mathbf{y}}^2 \boldsymbol{I}), \tag{47}$$

where the scale α controls the distortion strength. We set $\alpha = 2$ in our experiments.

F.2. DAPS Implementation Details

Euler ODE Solver For any given increasing and differentiable noisy scheduler σ_t and any initial data distribution $p(\mathbf{x}_0)$, we consider the forward diffusion SDE $\mathrm{d}\mathbf{x}_t = \sqrt{2\dot{\sigma}_t \sigma_t}\, \mathrm{d}\mathbf{w}_t$, where $\dot{\sigma}_t$ denotes the time derivative of σ_t and $\mathrm{d}\mathbf{w}_t$ represents the standard Wiener process. This SDE induces a probability path of the marginal distribution $\mathbf{x}_t$, denoted as $p(\mathbf{x}_t; \sigma_t)$. As demonstrated in [26, 49], the probability flow ODE for the above process is given by:

$$\mathrm{d}\mathbf{x}_t = -\dot{\sigma}_t \sigma_t \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t; \sigma_t)\, \mathrm{d}t. \tag{48}$$

By employing the appropriate preconditioning introduced in [25], we can transform the pre-trained diffusion model with parameter $\boldsymbol{\theta}$ to approximate the score function of the above probability path: $\boldsymbol{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, \sigma_t) \approx \nabla_{\mathbf{x}_t} \log p(\mathbf{x}_t; \sigma_t)$. In DAPS, we compute $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ by solving the ODE given $\mathbf{x}_t$ and time t as initial values.

Numerically, we use scheduler $\sigma_t = t$ and implement an Euler solver [26], which evaluates $\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t}$ at N_{ode} discretized time steps in interval $[0, t]$ and updates $\mathbf{x}_t$ by the discretized ODE. The time step t_i, $i = 1, \cdots, N_{\text{ode}}$ are selected by a polynomial interpolation between t and $t_{\min}$:

$$t_i = \left(t^{\frac{1}{\rho}} + \frac{i}{N-1} \left(t_{\min}^{\frac{1}{\rho}} - t^{\frac{1}{\rho}} \right) \right)^{\rho}. \tag{49}$$

We use $\rho = 7$ and $t_{\min} = 0.02$ throughout all experiments.

Annealing Scheduler To sample from the posterior distribution $p(\mathbf{x}_0 \mid \mathbf{y})$, DAPS adopts a noise annealing process to sample $\mathbf{x}_t$ from measurement conditioned time-marginals $p(\mathbf{x}_t \mid \mathbf{y})$, where $\mathbf{x}_t$ is defined by noisy perturbation of $\mathbf{x}_0$: $\mathbf{x}_t = \mathbf{x}_0 + \sigma_t^A \boldsymbol{\epsilon}$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, where σ_t^A is the annealing scheduler. In practice, we start from time T, assuming $p(\mathbf{x}_T \mid \mathbf{y}) \approx \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \boldsymbol{I})$, with $\sigma_{\max} = \sigma_T^A$. For simplicity, we adopt $\sigma_t^A = t$ and the same polynomial interpolation in Eq. (49) between σ_0 and σ_T for total N_A steps.

Hyperparameters Overview The hyperparameters of DAPS can be categorized into the following three categories.

(1) The ODE solver steps N_{ode} and annealing scheduler N_A. These two control the total NFE of DAPS. Need to trade-off between cost and quality. For linear tasks, $N_{\text{ode}} = 5$ and $N_A = 200$. And for nonlinear tasks $N_{\text{ode}=10}$ and $N_A = 400$. For LatentDAPS, including the one with the Stable Diffusion model, we choose $N_{\text{ode}} = 5$ and $N_{\text{A}} = 50$ for linear tasks and $N_{\text{ode}} = 10$ and $N_{\text{A}} = 100$ for nonlinear tasks.

(2) The step size η_t and total step N in Langevin dynamics (also damping factor γ_t in HMC). These two control the sample quality from $p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$. For simplicity, we adopt a linear decay scheme $\eta_t = \eta_0[\delta + t/T(1-\delta)]$, where δ is the decay ratio and T is the starting timestep. We include the final hyperparameters in Tab. 8. Moreover, instead of using the true $\beta_{\mathbf{y}} = 0.05$ in Eq. (4), we regard $\beta_{\mathbf{y}}$ as a hyperparameter and set it to 0.01 for better empirical performance. Moreover, we adopt HMC to speed up LatentDAPS with Stable Diffusion, where we report $\mu = 1 - \gamma_t \eta_t$ and $L = \eta_t^2$, which are set as fixed values for all timesteps.

(3) The $\sigma_{\max}$ and $\sigma_{\min}$ used in annealing process. We set $\sigma_{\max} = 100$ and 10 for DAPS and LatentDAPS and $\sigma_{\min} = 0.1$ to make be more robust to noise in measurement.

Table 8. **The hyperparameters** of experiments in paper for all tasks.

Algorithms	Tasks	Super Resolution 4×	Inpaint (Box)	Inpaint (Random)	Gaussian deblurring	Motion deblurring	Phase retrieval	Nonlinear deblurring	High dynamic range
DAPS	η_0	1e-4	5e-5	1e-4	1e-4	5e-5	5e-5	5e-5	2e-5
	δ	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2
	N	100	100	100	100	100	100	100	100
LatentDAPS	η_0	1e-4	2e-6	2e-6	2e-6	2e-6	4e-6	2e-6	6e-7
	δ	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2	1e-2
	N	50	50	50	50	50	50	50	50
LatentDAPS (SD v1.5)	L	1e-4	1e-5	3e-5	1e-5	2e-5	7e-5	1e-5	1e-5
	μ	0.45	0.60	0.60	0.90	0.85	0.70	0.80	0.70
	N	30	20	15	40	30	100	60	45

F.3. Baseline Details

DPS All experiments are conducted with the original code and default settings as specified in [13]. For high dynamic range reconstruction task, we use the $\xi_i = 1/\|\mathbf{y} - \mathcal{A}(\hat{\mathbf{x}}_0(\mathbf{x}_i)\|$

DDRM We adopt the default setting of $\eta_B = 1.0$ and $\eta = 0.85$ with 20 DDIM steps as specified in [30].

DDNM We adopt the default setting of $\eta_B = 1.0$ and $\eta = 0.85$ with 100 DDIM steps as specified in [54].

DCDP We adopt the default setting in [31] and directly use the open-sourced code for all results.

FPS-SMC We adopt the default setting of $M = 20$ and $N = 1000$ as specified in [18].

DiffPIR We adopt the same evaluation settings and report the results in [59].

RED-diff For a fair comparison, we use a slightly different RED-diff[35] by initializing the algorithm with random noise instead of a solution from the pseudoinverse. This might lead to a worse performance compared with the original RED-diff algorithm. We use $\lambda = 0.25$ and $lr = 0.5$ for all experiments.

PSLD We use the official implementation of PSLD [42] with the default configurations. Specifically, we use Stable diffusion v1.5 for ImageNet experiments, which is commonly believed to be a stronger pre-trained model than LDM-VQ4 used in other experiments.

ReSample All experiments are based on the official code of ReSample[45] with 500 steps DDIM sampler.

G. Experiments on Synthetic Data Distributions

Fig. 4 shows the sampling trajectories and predicted posterior distribution of DPS and DAPS on a synthetic data distribution. Specifically, we create a 2D Gaussian mixture as the prior distribution, *i.e.*, $p(\mathbf{x}_0) = \frac{1}{2}\left(\mathcal{N}(\mathbf{x}_0; \boldsymbol{c}_1, \boldsymbol{\Sigma}_1) + \mathcal{N}(\mathbf{x}_0; \boldsymbol{c}_2, \boldsymbol{\Sigma}_2)\right)$. Let $\boldsymbol{c}_1 = (-0.3, -0.4)$ and $\boldsymbol{c}_2 = (0.6, 0.5)$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \mathrm{diag}(0.01, 0.04)$. We draw 1000 samples from this prior distribution to create a small dataset, from which we can compute a closed-form empirical Stein score function at any noise level σ.

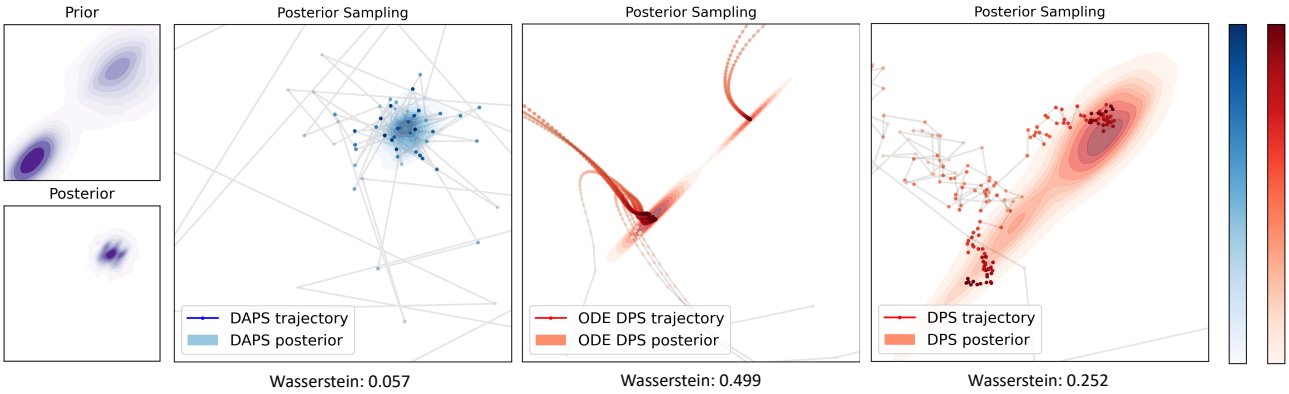


Figure 10. **DAPS and DPS (both SDE and ODE) on 2-dimensional synthetic data.** DAPS achieves much more accurate posterior sampling in terms of 2-Wasserstein distance.

Moreover, we consider the simplest measurement function that contains two modes, *i.e.*, $\mathbf{y} = \exp\left(-\frac{\|\mathbf{x}\|^2}{0.05}\right) + \exp\left(-\frac{\|\mathbf{x}-(0.5,0.5)\|^2}{0.05}\right) + \mathbf{n}$, where $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \beta_{\mathbf{y}}^2 \boldsymbol{I})$ with $\beta_{\mathbf{y}} = 0.3$. Let $\mathbf{y} = 0$, so that the likelihood $p(\mathbf{y} \mid \mathbf{x}_0)$ has two modes at $(0.5, 0.5)$ and $(0, 0)$. Since the prior distribution is large only at $(0.5, 0.5)$, the posterior distribution is single-mode, as illustrated in Fig. 10.

We run both DPS and DAPS for 200 steps and 100 independent samples on this synthetic dataset. However, as shown in Fig. 10, both SDE and ODE versions of DPS converge to two different modes. This is because DPS suffers from large errors in estimating likelihood $p(\mathbf{x}_t \mid \mathbf{y})$, especially in the early stages. These errors can hardly be corrected and are propagated along the SDE/ODE trajectory. DAPS, on the other hand, samples from a time-marginal distribution at each time step, and is able to recover the posterior distribution more accurately.

We further investigate the performance of posterior estimation by computing the Wasserstein distance between samples $\mathbf{x}_t$ and ground truth posterior $p(\mathbf{x}_t \mid \mathbf{y})$ for each step t. As shown in Fig. 11, the Wasserstein distance for DAPS decreases quickly and remains small throughout the sampling process. This conforms with our theory that the distribution of $\mathbf{x}_t$ is ensured to be $p(\mathbf{x}_t \mid \mathbf{y})$ for every noise level σ_t.

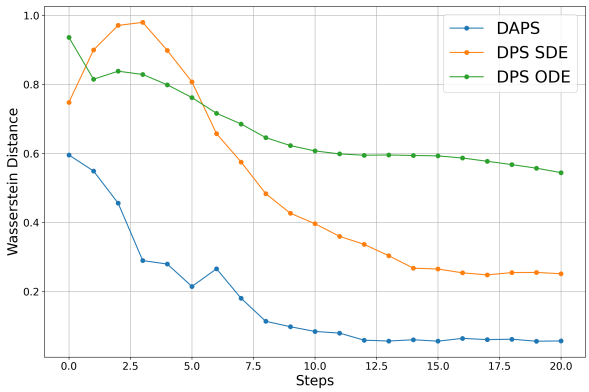


Figure 11. **Wasserstein distance** between estimated $\mathbf{x}_t \sim p(\mathbf{x}_t \mid \mathbf{y})$ and ground truth $p(\mathbf{x}_t \mid \mathbf{y})$.

H. Experimental Results on Compressed Sensing Multi-coil MRI

Compressed Sensing Multi-coil magnetic resonance imaging (CS-MRI) is a medical imaging problem that aims to shorten the acquisition time of MRI scanning by subsampling. Specifically, CS-MRI takes in only a subset of the measurement space (k-space) and solves an inverse problem to reconstruct the whole source image in high resolution. Suppose the underlying source image is $\mathbf{x}_0 \in \mathbb{C}^n$. The forward function of CS-MRI can be written as

$$\mathbf{y} = \boldsymbol{P}\boldsymbol{F}\boldsymbol{S}\mathbf{x}_0 + \mathbf{n}, \tag{50}$$

where $\boldsymbol{P}$ is a subsampling operator with only zeros and ones on its diagonal, $\boldsymbol{F}$ is the Fourier transform matrix, and $\boldsymbol{S}$ is the multi-coil sensitivity map so that $\boldsymbol{S} = [\boldsymbol{S}_1, \dots, \boldsymbol{S}_c]$ for c coils.

Experimental setup. We preprocess the fastMRI dataset [57] by calculating the magnitude images of the minimal variance unbiased estimator (MVUE), and resize them into 320×320 grayscale images. This pipeline is the same as [23]. We use a diffusion model trained with EDM framework [26] on the preprocessed data. We compare DAPS with DPS [13] and DiffPIR [59] designed for general image restoration tasks, and also ScoreMRI [10] and CSGM [23] which are specifically designed for solving MRI with diffusion models.

Results. We include the quantitative results in Tab. 2 in the main text. Here, we provide some visual results of the reconstructions in Appendix H, which validates the capability of DAPS in solving real-world medical imaging inverse problems.

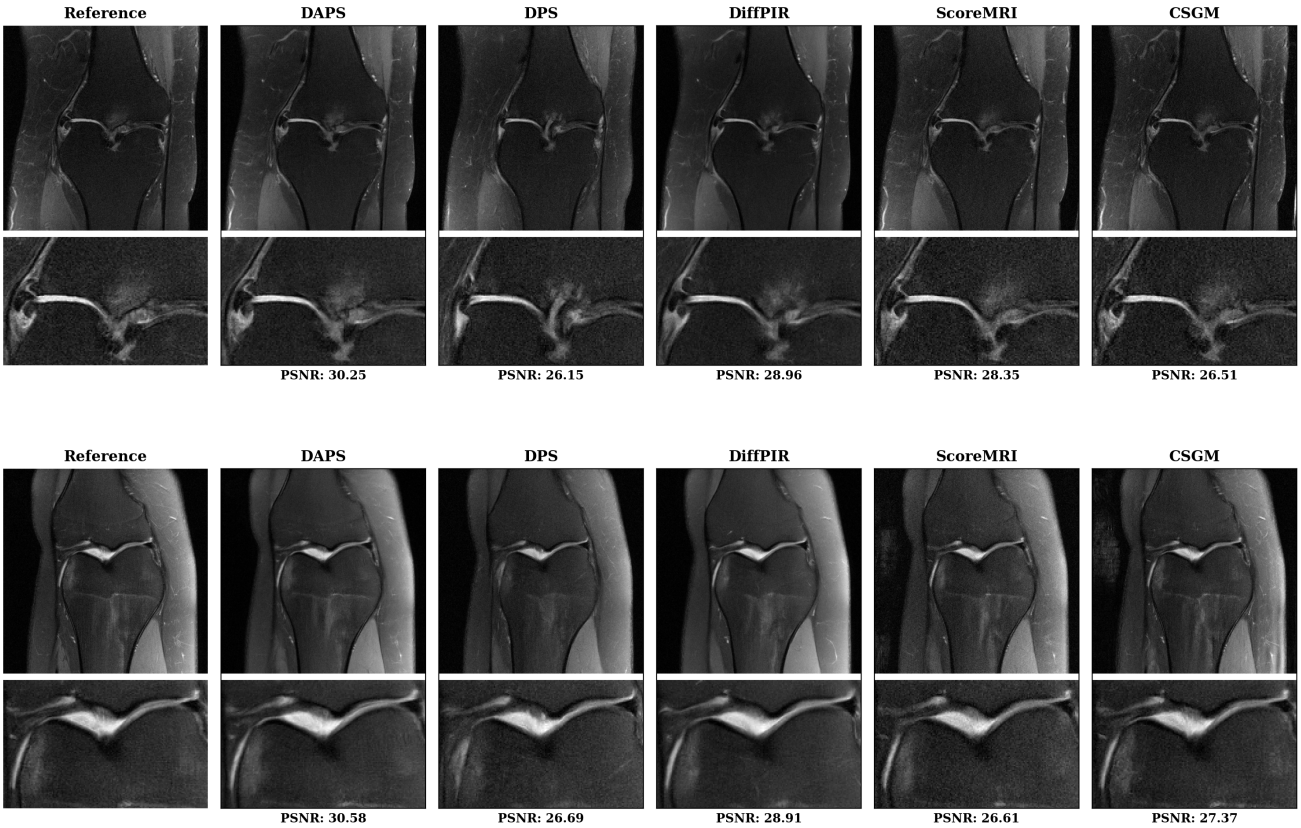


Figure 12. **Qualitative results** of inverse problem solvers with diffusion models on CS-MRI.

I. Additional Results

I.1. More Ablation Study

Effectiveness of different MCMC samplers. We conduct a detailed comparison of the three proposed MCMC samplers in DAPS. To represent linear and nonlinear inverse problems, we select super resolution 4× and high dynamic range, evaluating performance using FFHQ 256 images. Hyperparameter tuning is performed on 5 validation images, and results are reported on 10 test images. The comparison is illustrated in Tab. 9. HMC achieves the best balance between time efficiency and generation quality among the methods. Langevin dynamics, while simpler to implement and requiring fewer tunable hyperparameters, delivers competitive results. Although Metropolis Hastings shows slightly inferior performance, it has the advantage of being extendable to problems where gradient information is unavailable.

Table 9. **Quantitative comparison on different MCMC samplers.** HMC achieves the best balance between time efficiency and generation quality among the methods.

	Super resolution 4×		High dynamic range		Number of forward function callings
	PSNR ↑	LPIPS ↓	PSNR ↑	LPIPS ↓	
DAPS+HMC	**28.17**	**0.160**	**27.68**	0.163	10
DAPS+Langevin dynamics	28.15	0.166	27.57	**0.162**	100
DAPS+Metropolis hasting	27.46	0.192	25.21	0.224	50000

Effectiveness of the number of function evaluations. To better understand how the number of function evaluations (NFE) of the diffusion model influences the performance, we evaluate the performance of DAPS with different configurations. Recall that we use an ODE sampler in each inner loop to compute $\hat{\mathbf{x}}_0(\mathbf{x}_t)$, the total NFE for DAPS is the number of inner ODE steps times the number of noise annealing steps. We evaluate DAPS using NFE ranging from 50 to 4k, with configurations as specified in Appendix E.1. As indicated by Fig. 13, DAPS achieves relatively decent performance even with small NFE.

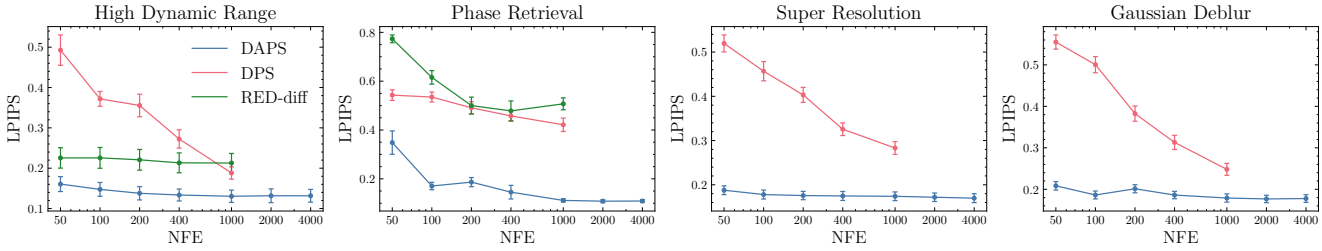


Figure 13. **Quantitative evaluations of image quality for different number of function evaluations (NFE).** Experiments are conducted on the FFHQ 256 dataset for four different tasks.

Effectiveness of the number of ODE steps. Recall that we use an ODE sampler to compute $\hat{\mathbf{x}}_0(\mathbf{x}_t)$, the estimated mean of the approximated distribution $p(\mathbf{x}_0 \mid \mathbf{x}_t)$. We use the same number of function evaluations in our ODE sampler throughout the entire algorithm. To test how the number of function evaluations (NFE) in the ODE sampler influences the performance, we try different NFE on two linear tasks and one nonlinear task. In particular, when NFE is 1, the ODE sampler is equivalent to computing $\mathbb{E}[\mathbf{x}_0 \mid \mathbf{x}_t]$ via Tweedie's formula. As shown in Fig. 14, increasing NFE in the ODE sampler consistently improves the overall image perceptual quality, but also at the cost of a slightly lower PSNR. This trade-off between PSNR and LPIPS is also observed in [35]. Perceptually, we notice that increasing ODE steps adds more fine-grained details to the produced images, which improves LPIPS but decreases PSNR. This finding is corroborated by Fig. 15, where the reconstructed images appear less blurry and show high-frequency details as the number of ODE steps increases.

Effectiveness of annealing noise scheduling step. To better understand how the scheduling of sigma influences performance, we also evaluate the effects of sampling with varying noise scheduling steps. A larger number of scheduling steps implies a denser discretization grid between $\sigma_{\max}$ and $\sigma_{\min}$. The quantitative results are shown in Fig. 16. The performance of

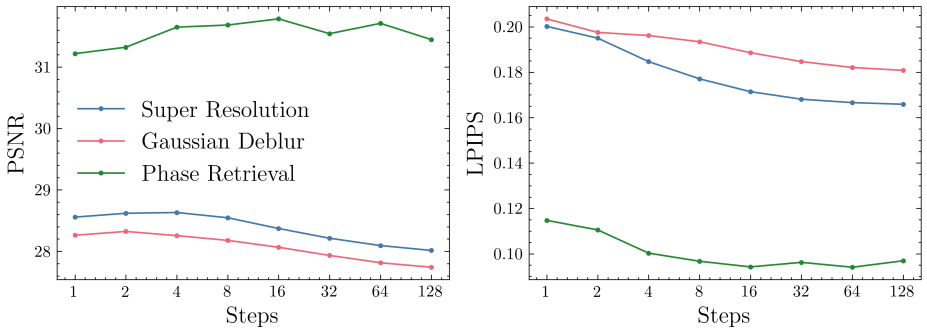


Figure 14. The effect of the **number of ODE steps** for denoisers.

Figure 15. **Qualitative ablation studies on the number of ODE steps**. We run DAPS with different numbers of ODE steps on super-resolution 4× task. More details are observed as the number of ODE steps increases.

DAPS on linear tasks slightly increases as the number of annealing noise scheduling steps increases, while its performance on nonlinear tasks (*e.g.*, phase retrieval) increases dramatically with the number of scheduling steps. However, DAPS achieves a near-optimal sample quality when the number of noise scheduling steps is larger than 200.

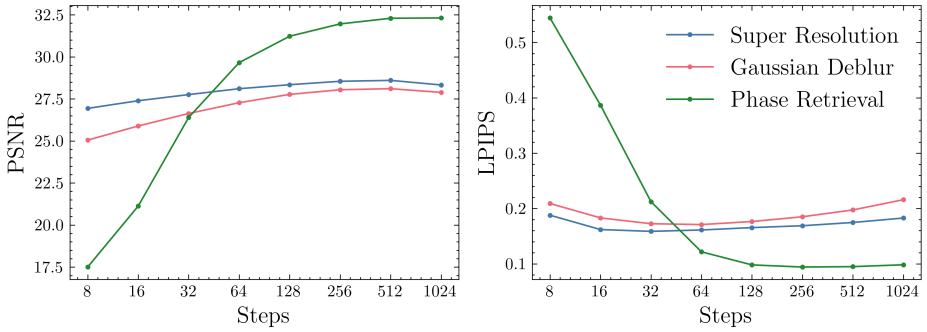


Figure 16. The effect of the number of **annealing noise scheduling steps**.

Different Measurement Noise Level When subjected to varying levels of measurement noise, the quality of solutions to inverse problems can differ significantly. To evaluate the performance of DAPS under different noise conditions, we present the results in Fig. 17. DAPS is robust to small noise levels ($\sigma < 0.05$) and degrades almost linearly as σ continues to increase.

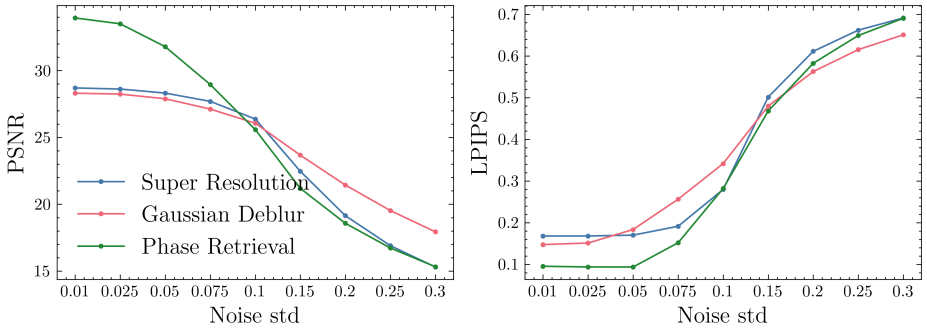


Figure 17. The effect of the **measurement noise level** $\beta_{\mathbf{y}}$.

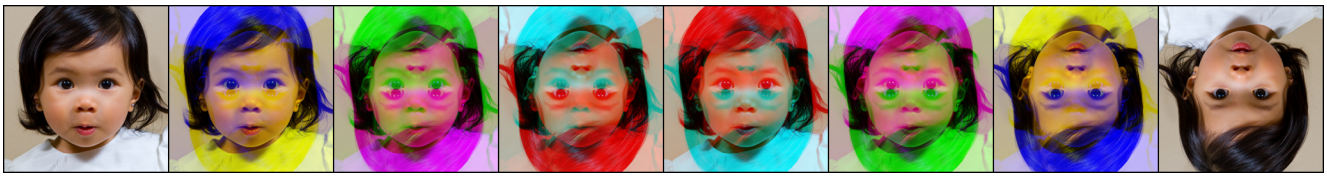


Figure 18. Eight images with exactly the **same measurement** for the phase retrieval task.

I.2. More Discussion on Phase Retrieval

Compared to the baselines, DAPS exhibits significantly better sample quality and stability in the phase retrieval task. Unlike other selected tasks, phase retrieval is more ill-posed, meaning that images with the same measurements can appear quite different perceptually. Specifically, there are multiple disjoint modes with exactly the same measurement for phase retrieval, while for other tasks, such as super resolution and deblurring, the subset of images with low measurement error is a continuous set. We show in Fig. 18 eight images with disparate perceptual features but with exactly the same measurement in phase retrieval. To mitigate this issue, oversampling is often used to reduce the ill-posedness of the phase retrieval problem. We present quantitative results in Tab. 10 using different oversampling ratios in phase retrieval. These results further demonstrate the strength of DAPS in addressing complex, ill-posed inverse problems.

Table 10. Phase retrieval of **different oversampling ratios** with DAPS.

Oversample	2.0	1.5	1.0	0.5	0.0
LPIPS	0.117	0.131	0.235	0.331	0.489
PSNR	30.26	29.17	24.87	21.60	16.02

I.3. More Analysis on Sampling Trajectory

Here we show a longer trajectory of phase retrieval in Figs. 19 to 21. The $\hat{\mathbf{x}}_0(\mathbf{x}_t)$ evolves from unconditional samples from the model to the posterior samples while $\mathbf{x}_{0|\mathbf{y}}$ evolves from noisy conditioned samples to the posterior samples. These two trajectories converge to the same sample as noise annealing down.

I.4. More Qualitative Samples

We show a full stack of phase retrieval samples in 4 runs without manual post-selection in Figs. 22 and 23. More samples for other tasks are shown in Figs. 24 and 25. The more diverse samples from box inpainting of size 192×192 and super resolution of factor 16 are shown in Figs. 26 and 27. More samples for CS-MRI are shown in Fig. 28.

(a) the estimated means of $p(\mathbf{x}_t \mid \mathbf{x}_0)$ as $\hat{\mathbf{x}}_0(\mathbf{x}_t)$

(b) the samples $\mathbf{x}_{0|y} \sim p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$

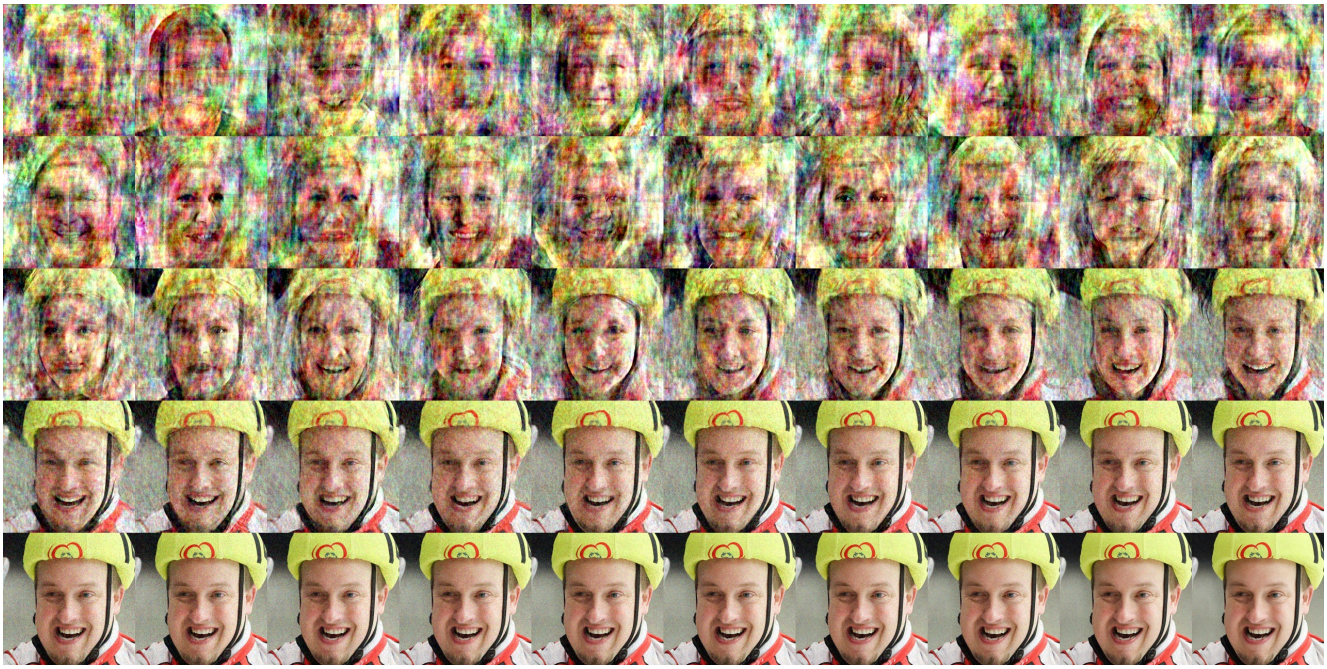
Figure 19. **DAPS trajectory for phase retrieval.** The images are selected from 200 annealing steps, evenly spaced in DAPS-1k configuration.

(a) the estimated means of $p(\mathbf{x}_t \mid \mathbf{x}_0)$ as $\hat{\mathbf{x}}_0(\mathbf{x}_t)$

(b) the samples $x_{0|y} \sim p(x_0 \mid x_t, y)$

Figure 20. **DAPS trajectory for phase retrieval.** The images are selected from 200 annealing steps, evenly spaced in DAPS-1k configuration.

(a) the estimated means of $p(\mathbf{x}_t \mid \mathbf{x}_0)$ as $\hat{\mathbf{x}}_0(\mathbf{x}_t)$



(b) the samples $\mathbf{x}_{0|\mathbf{y}} \sim p(\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y})$

Figure 21. **DAPS trajectory for phase retrieval.** The images are selected from 200 annealing steps, evenly spaced in DAPS-1k configuration.

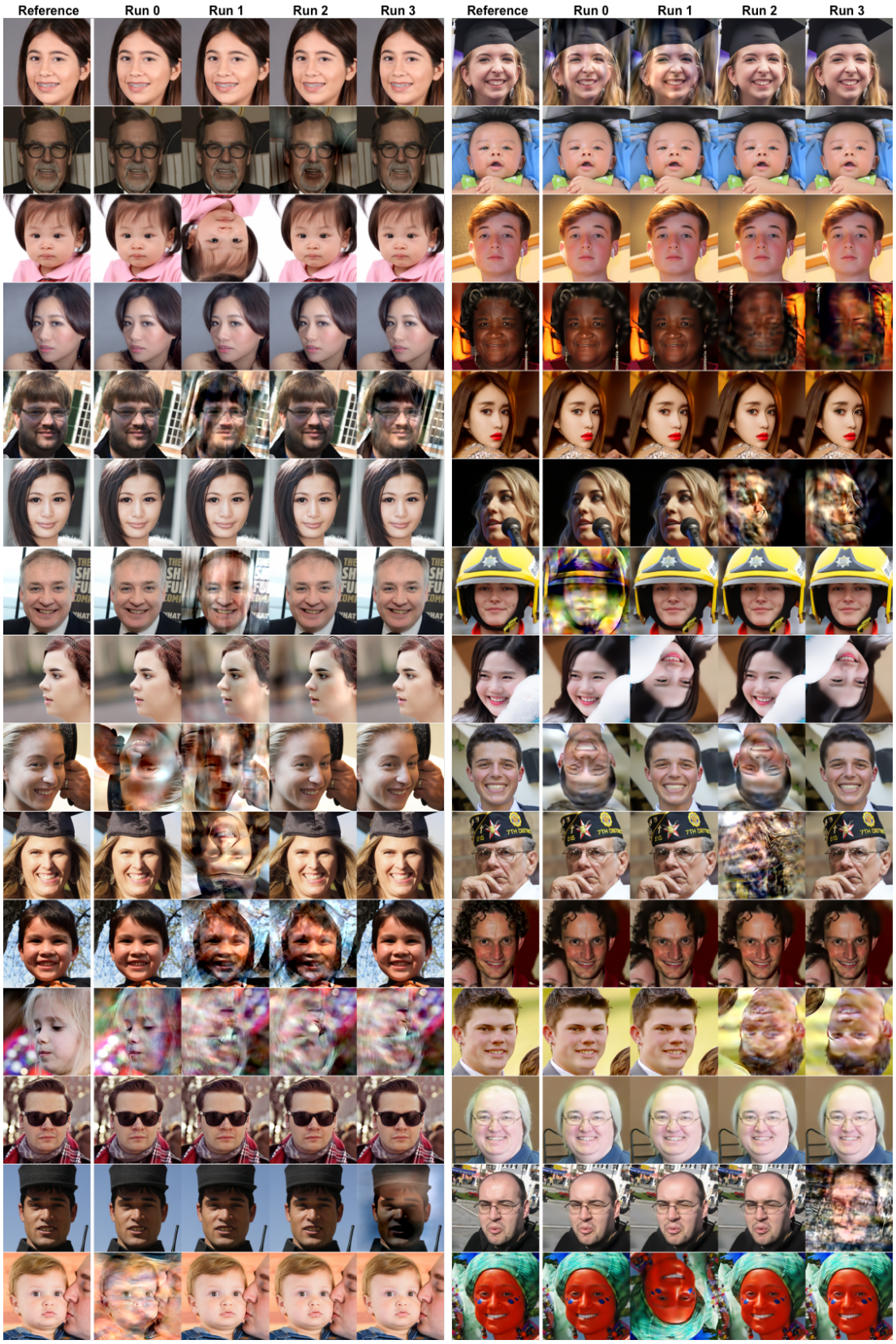


Figure 22. **Phase retrieval samples from DAPS (left) and LatentDAPS (right) in four independent runs on FFHQ.** No manual selection was performed to better visualize the success rate and sample quality.

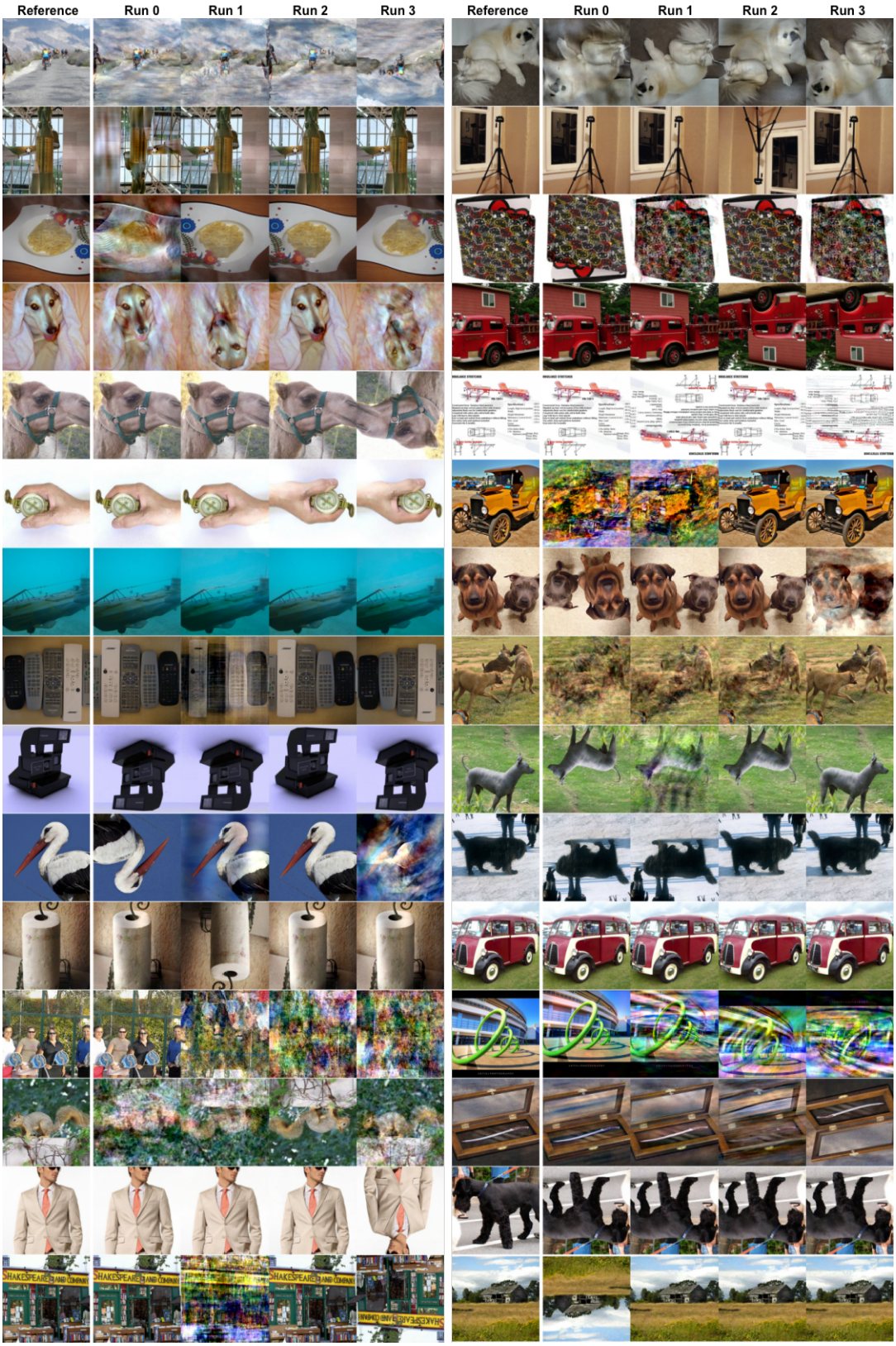
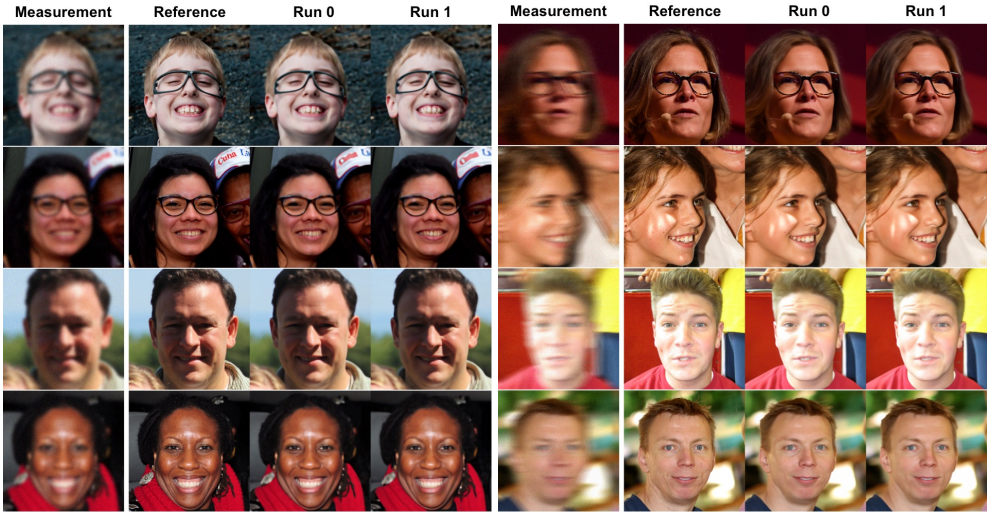
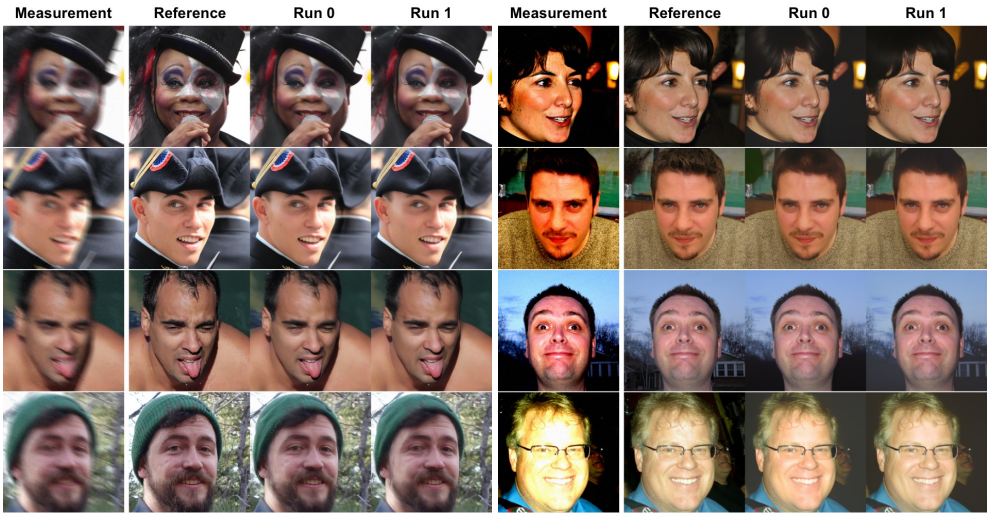


Figure 23. **Phase retrieval samples from DAPS in four independent runs on ImageNet.** The success rate of ImageNet is less than FFHQ and contains more samples with 180 degree rotation. No manual selection was performed to better visual ize the success rate and sample quality.

(a) Super resolution 4× (b) Inpaint (box)

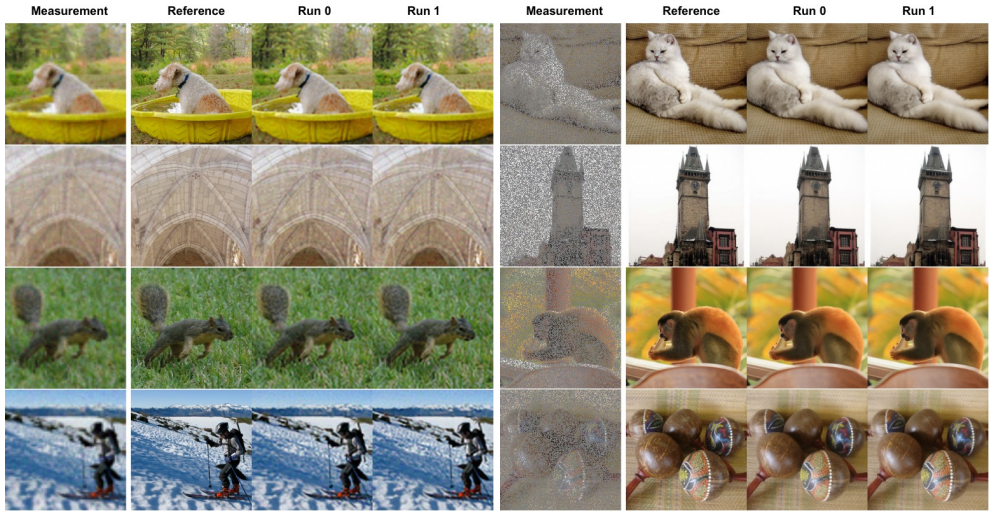


(c) Gaussian deblurring (d) Motion deblurring

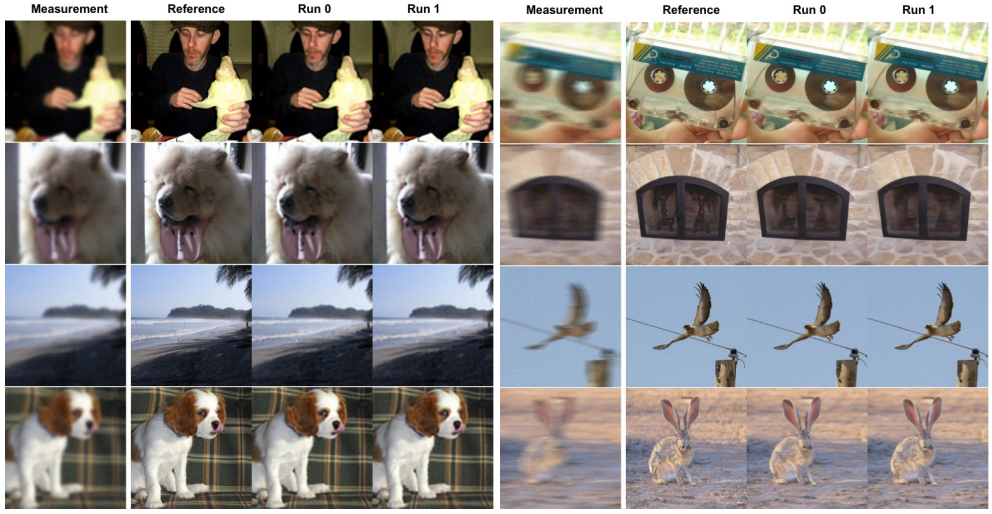


(e) Nonlinear deblurring (f) High Dynamic Range

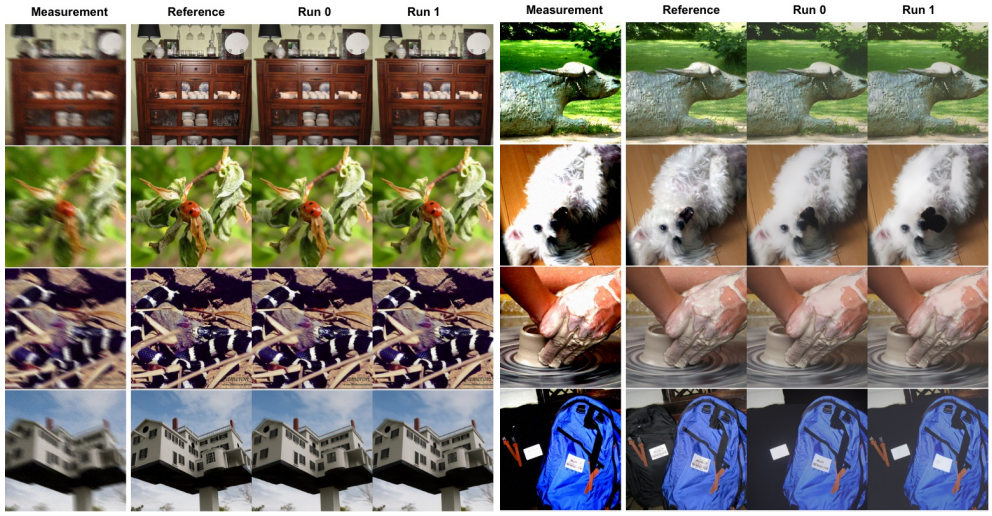
Figure 24. **DAPS samples for various tasks on FFHQ.** DAPS can obtain visually better samples for the above linear and nonlinear tasks.



(a) Super resolution 4× (b) Inpaint (box)



(c) Gaussian deblurring (d) Motion deblurring



(e) Nonlinear deblurring (f) High Dynamic Range

Figure 25. **DAPS samples for various tasks on ImageNet.** DAPS can obtain visually better samples for the above linear and nonlinear tasks.

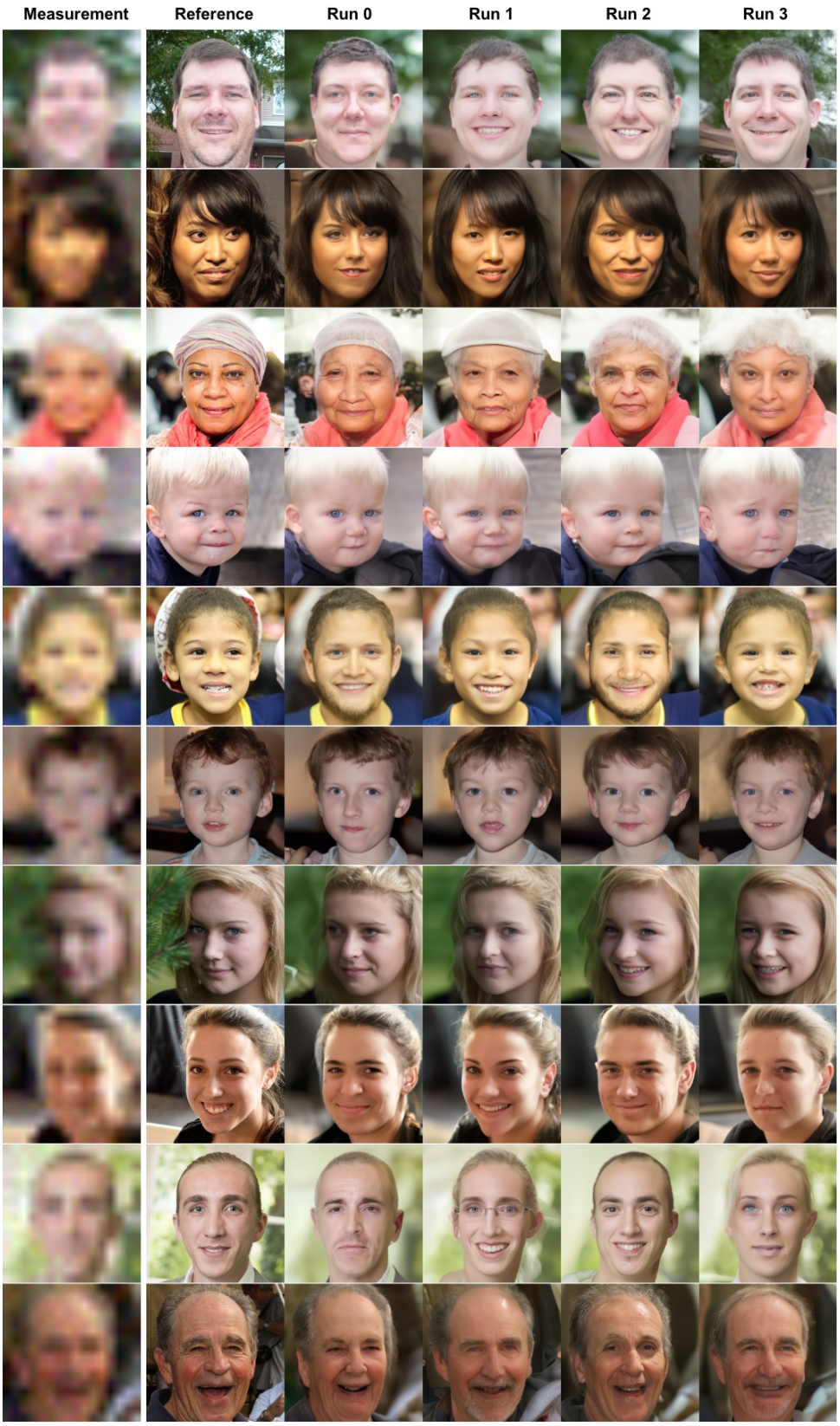


Figure 26. **More samples for super resolution 16×.** DAPS is able to generate diverse samples when the posterior distribution is multi-modal.

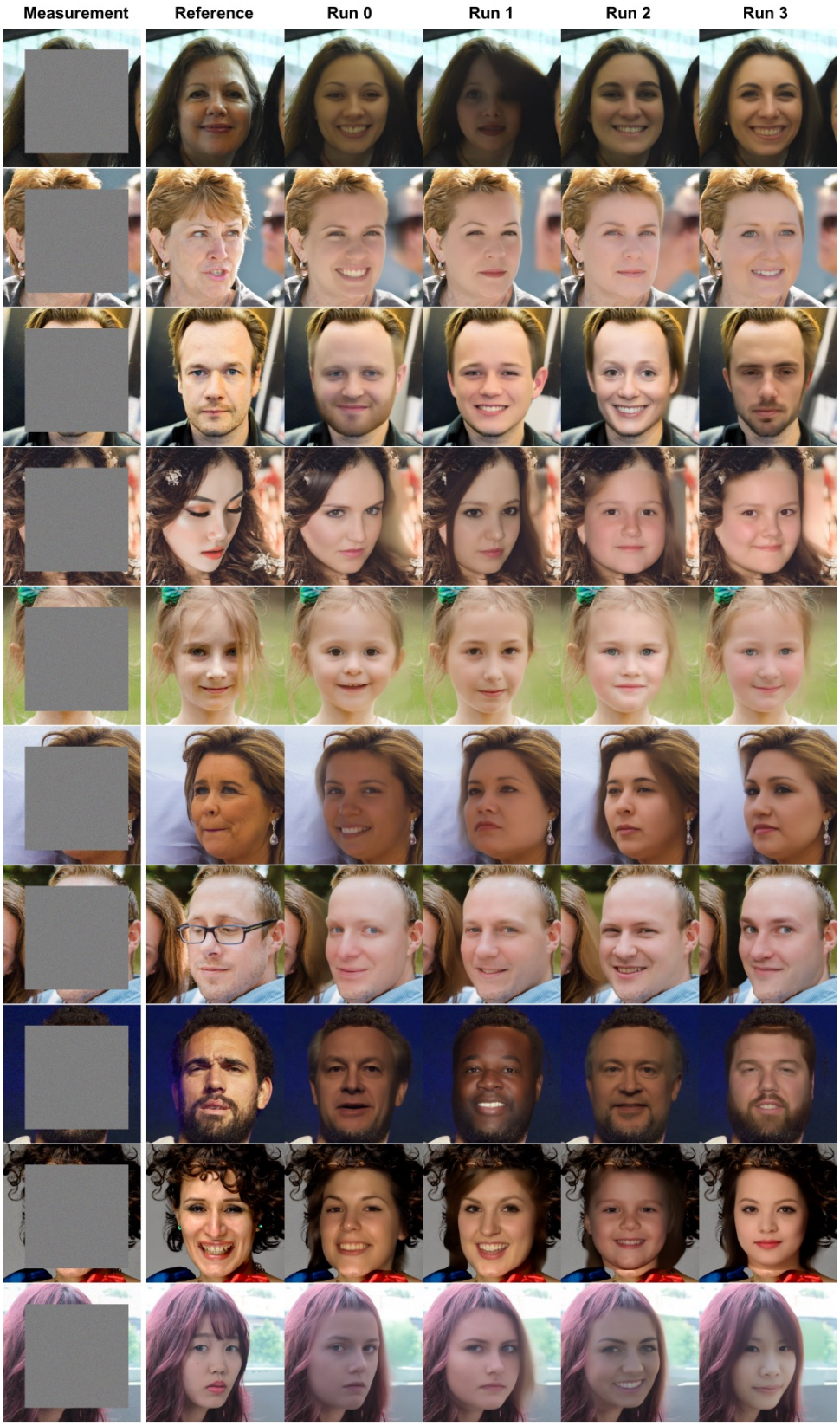


Figure 27. **More samples for inpainting of 192×192 box.** DAPS is able to generate diverse samples when the posterior distribution is multi-modal.

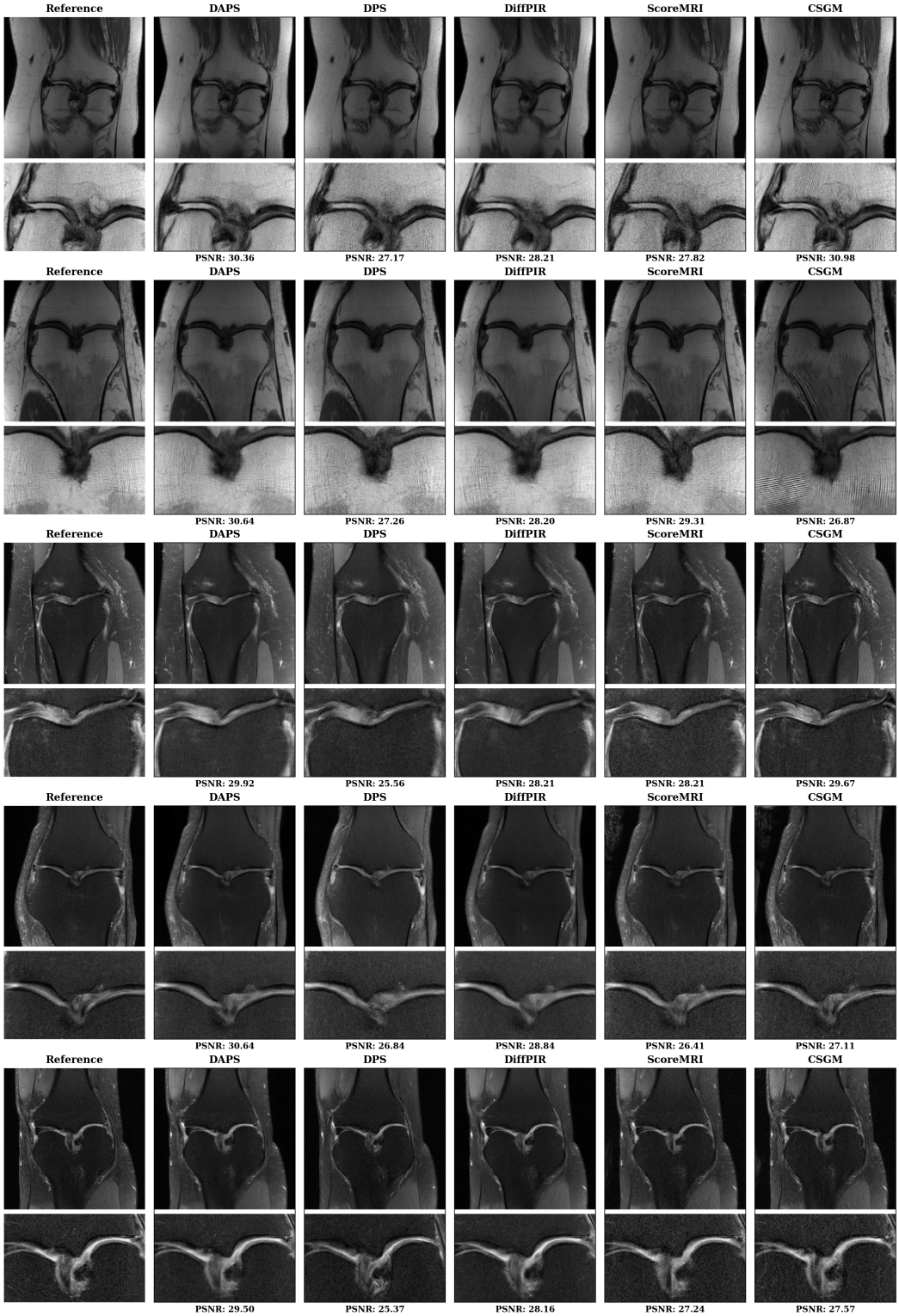


Figure 28. **More samples for CS-MRI.** DAPS is able to generate high-fidelity reconstructions for CS-MRI tasks.