Exact Posterior Score Estimation for Solving Linear Inverse Problems

Abbas Mammadov* University of Oxford
Ozgur Kara* UIUC
Kaan Oktay fal
Iskander Azangulov University of Oxford
Adil Kaan Akan fal
Hyungjin Chung Korea University
James Matthew Rehg UIUC
Yee Whye Teh University of Oxford

Abstract

Diffusion and flow-based models learn powerful data priors by training a denoiser to reverse Gaussian corruption. To use this prior to solve a linear inverse problem, one needs to sample from the posterior, but the score that the prior provides is the unconditional score, not the posterior score. Existing methods either steer a fixed pretrained denoiser with approximate measurement-matching corrections, or train a conditional restoration model that abandons the denoising structure of the prior. We derive the exact posterior score in closed form for linear Gaussian inverse problems under general Gaussian interpolants, and show that posterior sampling reduces to a denoising problem at an operator-dependent shifted pivot under an anisotropic noise covariance. We turn this identity into **Exact Posterior Score (EPS)**, a denoising training objective whose population minimizer is the exact posterior denoiser. It preserves the input/output structure of standard pretraining and can therefore be trained from scratch or fine-tuned from a pretrained denoiser. At inference, EPS uses the same sampler as the underlying backbone, with no likelihood gradients or projections. We evaluate EPS on five linear inverse problems across FFHQ and ImageNet, where it outperforms training-free and training-based baselines on fidelity, perceptual, and distributional metrics, while using roughly an order of magnitude fewer denoiser evaluations than gradient-based posterior samplers. The recipe scales to 1024×1024, where EPS, fine-tuned from a pretrained flow model, outperforms zero-shot flow-based solvers that use the same model on every task and metric.

Code: github.com/abbasmammadov/EPS **Project page:** abbasmammadov.github.io/EPS

1 Introduction

Linear inverse problems, in which an unknown signal x_0 must be recovered from a noisy linear measurement $y = Ax_0 + \eta$ with known forward operator A and observation noise η, are pervasive across imaging and the sciences, including compressive sensing [1, 2], accelerated medical imaging [3, 4], super-resolution [5, 6], deblurring [7, 8], and inpainting in computational photography [9, 10]. The forward operator A is typically ill-conditioned or rank-deficient, so many candidate signals are consistent with the same observation, and the right object to recover is the posterior $p(x_0|y)$ rather than any single point estimate. The posterior captures uncertainty over reconstructions, supports downstream decisions, and exposes the trade-off between data fidelity and prior plausibility.

Diffusion and flow-based generative models offer powerful, expressive data priors for this task, learning a denoising trajectory from noise back to clean samples [11–17]. The central question is

*Equal contribution. Correspondence to abbas.mammadov@stats.ox.ac.uk and y.w.teh@stats.ox.ac.uk.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

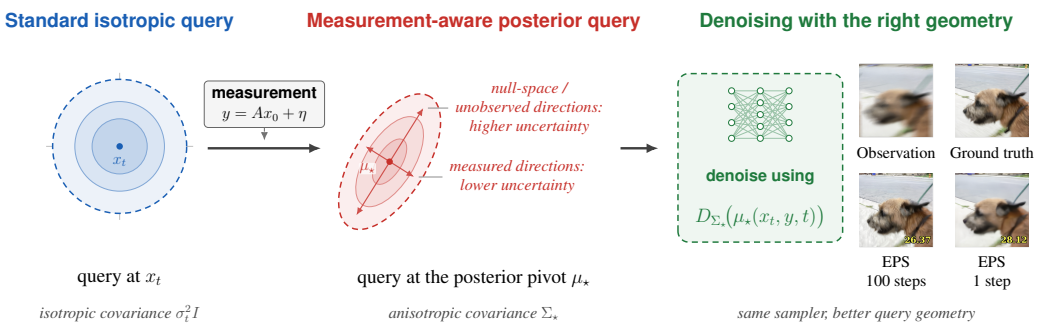

Figure 1: **EPS turns posterior sampling into denoising with the right query geometry.** Instead of denoising an isotropic query at x_t, the measurement shifts the query to the posterior pivot $\mu_\star$ and reshapes the noise into an anisotropic covariance $\Sigma_\star$. Measured directions become more certain, while unobserved directions remain uncertain. EPS trains a denoiser for this anisotropic geometry and reuses the backbone's unconditional sampler unchanged. The first sampler step estimates the posterior mean $\mathbb{E}[x_0|y]$, which typically has higher PSNR but is over-smoothed; the final sample (here after 100 steps) has more detail.

how to turn this trajectory into a sampler from $p(x_0|y)$. The reverse-time sampler needs the *posterior* score $\nabla_{x_t} \log p(x_t|y)$, not the unconditional prior score $\nabla_{x_t} \log p(x_t)$ that diffusion training provides. Replacing the former by an approximation introduces bias at every step, which compounds into oversmoothing, hallucinated structure, or poorly calibrated uncertainty. Existing methods fall into two broad camps.

Training-free methods keep a pretrained denoising backbone fixed and add a measurement-matching update at each reverse step. The prototypical example is Diffusion Posterior Sampling (DPS) [18], which differentiates a measurement loss through the unconditional denoiser, with variants using projections, denoised estimates, or task-specific correction rules [18–25]. This route is attractive because it is zero-shot and inherits the strong unconditional prior of a pretrained model. However, the added update is only an approximation to the true measurement-matching score, and even moment-matching variants that track anisotropic uncertainty in $p(x_0|x_t)$ [26–28] only refine the unconditional denoising query. Asymptotically exact alternatives based on sequential Monte Carlo [29–31] avoid this approximation, but at the cost of running many particle trajectories per observation.

Training-based methods sidestep the approximation question by training a new model specifically for the inverse problem, with the measurement y as input. This family includes conditional diffusion models that learn a measurement-conditional score [32–34], bridge-based methods that build a trajectory directly from y to the data [35, 36], and methods that distill a posterior sampler from a pretrained diffusion prior [37–39]. In all cases, the network is exposed to the raw measurement rather than to the geometry of the exact posterior denoising query, so it must learn the operator dependence end-to-end.

In contrast, we observe that for linear Gaussian inverse problems the exact posterior score has a closed form, with a simple structural meaning. As Figure 1 illustrates, posterior sampling is still a denoising problem, but with a measurement-aware input and an operator-dependent anisotropic noise covariance. We use this identity to define **Exact Posterior Score (EPS)**, a denoising training objective whose target and loss match those of standard pretraining, with the input replaced by a measurement-dependent pivot. EPS can therefore be trained from scratch or fine-tuned efficiently from a pretrained denoiser. At inference, it runs the underlying backbone's sampler unchanged, with no likelihood gradients, projections, or inner optimization. Throughout, "exact" refers to the identity and to the population minimizer of the objective; the trained network approximates it, as any learned denoiser does.

Our contributions are as follows.

- We derive the exact posterior score for general linear Gaussian inverse problems (including rank-deficient, blur, and downsampling operators) under general Gaussian interpolants, and show that posterior sampling reduces to denoising at an operator-dependent shifted pivot under an anisotropic covariance. We also pinpoint where existing approximate-guidance methods deviate from this exact identity.

- We turn the identity into EPS, a denoising training objective and sampling algorithm that preserves the structure of standard pretraining while incorporating the exact posterior geometry. EPS can be trained from scratch or fine-tuned from a pretrained checkpoint, and it uses the underlying backbone's sampler at inference.
- We evaluate EPS on five linear inverse problems across FFHQ and ImageNet, reporting pointwise fidelity, perceptual quality, and distributional calibration metrics, and find consistent improvements over training-free and training-based baselines at substantially smaller sampling budgets, and over zero-shot flow-based solvers at 1024×1024.

2 Background

2.1 Generative Models as Denoising Trajectories

Let $x_0 \sim p_{\mathrm{data}}$ and $\epsilon \sim \mathcal{N}(0, I_d)$. A stochastic interpolant defines

$$x_t = \alpha_t x_0 + \beta_t \epsilon, \qquad t \in [0, T], \tag{1}$$

with $(\alpha_0, \beta_0) = (1, 0)$ and $(\alpha_T, \beta_T) = (0, 1)$ in the data-to-noise convention. Different choices of (α_t, β_t) recover variance-preserving diffusion [12, 13], rectified flow [16], and EDM [17], with corresponding network parameterizations as a score, noise predictor, velocity, or EDM-preconditioned denoiser; the conversion between them is deterministic once (α_t, β_t) are fixed, so we write all backbones through a learned denoiser D_θ. The forward kernel is

$$p(x_t|x_0) = \mathcal{N}(x_t; \alpha_t x_0, \beta_t^2 I_d). \tag{2}$$

The marginal score $s_t(x) = \nabla_x \log p(x)$ is equivalent to the optimal denoiser through Tweedie's identity,

$$D_t(x) \coloneqq \mathbb{E}[x_0|x_t = x] = \frac{1}{\alpha_t}\big(x + \beta_t^2 s_t(x)\big), \tag{3}$$

and the reverse velocity follows from the same identity:

$$v_t(x) = \mathbb{E}[\dot\alpha_t x_0 + \dot\beta_t \epsilon \mid x_t = x] = \frac{\dot\beta_t}{\beta_t}\, x + \left(\dot\alpha_t - \frac{\alpha_t \dot\beta_t}{\beta_t}\right) D_t(x). \tag{4}$$

This viewpoint sets up EPS: if the posterior denoiser is known, the posterior score and posterior velocity follow immediately from the same identities, and the base model's sampler can be reused.

2.2 Inverse Problems and Approximate Posterior Sampling

We observe

$$y = Ax_0 + \eta, \qquad \eta \sim \mathcal{N}(0, \sigma_y^2 I_m), \tag{5}$$

for a known linear operator $A \in \mathbb{R}^{m\times d}$. This notation covers masks, downsampling, and convolutional blur operators, and includes rank-deficient settings where many signals are consistent with the same observation. The target is the posterior $p(x_0|y) \propto p(y|x_0)p_{\mathrm{data}}(x_0)$. A reverse sampler should therefore use

$$\nabla_{x_t} \log p(x_t|y) = \nabla_{x_t} \log p(x_t) + \nabla_{x_t} \log p(y|x_t), \tag{6}$$

where the second term is the measurement-matching score. Zero-shot solvers approximate it with the template $\nabla_{x_t} \log p(y|x_t) \approx -L_t M_t / G_t$ [28], where M_t is a measurement residual, L_t lifts it back to sample space, and G_t is the guidance strength.

3 Exact Posterior Score (EPS)

We now derive the posterior score and convert it into a training objective. The derivation has two pieces. First, a Gaussian product identifies the correct pivot and covariance. Second, an anisotropic Tweedie identity turns the resulting smoothed density into a denoiser.

3.1 Anisotropic Tweedie Identity

For a positive definite covariance matrix Σ, define the Gaussian-smoothed data density

$$p_{\text{data}}^{\Sigma}(\mu) = \int \mathcal{N}(\mu; x_0, \Sigma)\, p_{\text{data}}(x_0)\, \mathrm{d}x_0, \tag{7}$$

and the corresponding optimal denoiser

$$D_{\Sigma}(\mu) = \mathbb{E}[x_0 | x_0 + \xi = \mu], \qquad \xi \sim \mathcal{N}(0, \Sigma). \tag{8}$$

Then, by the anisotropic form of Tweedie's formula [40, 41],

$$D_{\Sigma}(\mu) = \mu + \Sigma \nabla_{\mu} \log p_{\text{data}}^{\Sigma}(\mu), \tag{9}$$

which is the usual Tweedie formula when Σ is a scalar multiple of the identity. EPS uses this identity with a covariance that is not chosen by hand, but rather is derived from the inverse problem formulation.

3.2 Closed-Form Posterior Score

The posterior marginal can be written as

$$p(x_t|y) \;\propto\; \int p(x_t|x_0) p(y|x_0) p_{\text{data}}(x_0)\, \mathrm{d}x_0. \tag{10}$$

Both factors inside the integral are Gaussian in x_0. Completing the square gives the following result.

Theorem 1 (Exact posterior score)**.** *Under the linear Gaussian inverse problem* (5) *and the interpolant* (1)*, the posterior score at time* t *is*

$$\nabla_{x_t} \log p(x_t|y) = \frac{1}{\beta_t^2}\Big(\alpha_t D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) - x_t\Big), \tag{11}$$

where

$$\Sigma_\star(t) = \left(\frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A\right)^{-1}, \qquad \mu_\star(x_t, y, t) = \Sigma_\star(t)\left(\frac{\alpha_t}{\beta_t^2} x_t + \frac{1}{\sigma_y^2} A^\top y\right). \tag{12}$$

Equivalently, $D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) = \mathbb{E}[x_0 | x_t, y]$.

The proof is given in Appendix A.2. The theorem says that posterior sampling is still a denoising problem, but not the isotropic one seen in unconditional pretraining. The denoiser must be queried at a measurement-aware input $\mu_\star$ under a measurement-aware anisotropic noise covariance $\Sigma_\star$.

Proposition 2 (Posterior velocity)**.** *The posterior velocity associated with the interpolant* (1) *is*

$$v_t^y(x_t) = \mathbb{E}[\dot\alpha_t x_0 + \dot\beta_t \epsilon | x_t, y] = \frac{\dot\beta_t}{\beta_t} x_t + \left(\dot\alpha_t - \frac{\alpha_t \dot\beta_t}{\beta_t}\right) D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)). \tag{13}$$

Thus estimating the exact posterior denoiser is equivalent to estimating the exact posterior flow.

The proof is given in Appendix A.3.

Posterior pivot. We call $\mu_\star$ the *posterior pivot* because the proof of Theorem 1 (Appendix A.2) shows that the joint quadratic form in (x_t, y, x_0) pivots about $\mu_\star(x_t, y, t)$: Completing the square sends the entire dependence on x_0 into a single Gaussian centered at $\mu_\star$ with covariance $\Sigma_\star(t)$, while x_t and y enter only through this pivot and a multiplicative normalizer. Equivalently, $\mu_\star$ is the precision-weighted Bayesian fusion of the current state and the measurement under the two Gaussian likelihoods, before the data prior p_{data} is folded in by the denoiser $D_{\Sigma_\star(t)}$. The pivot is therefore the only summary statistic of (x_t, y) that the posterior denoiser needs to see.

Computing $\mu_\star$. Although (12) involves inverting a $d \times d$ matrix in general, every operator we consider admits a fast structured solve. For binary inpainting masks $A^\top A$ is diagonal, for downsampling it is block-diagonal, and for circular convolutions used in deblurring it is diagonalized by the FFT. The per-step cost of computing $\mu_\star$ is therefore negligible relative to a denoiser forward pass. For more general operators, $\mu_\star$ can still be obtained efficiently via conjugate gradient applied to the symmetric positive-definite system $(\alpha_t^2/\beta_t^2\, I + \sigma_y^{-2} A^\top A)\, \mu_\star = (\alpha_t/\beta_t^2)\, x_t + \sigma_y^{-2} A^\top y$, which only requires matrix-vector products with A and $A^\top$. We measure these costs directly in Appendix D.12, where the structured $\mu_\star$ solve adds only sub-millisecond overhead per sampling step.

3.3 What was missing in Training-Free Methods

Theorem 1 also pinpoints what existing training-free methods miss. Combining (11) with the unconditional Tweedie identity (3), the measurement-matching score can be written as a difference of two denoisers,

$$\nabla_{x_t} \log p(y|x_t) \;=\; \nabla_{x_t} \log p(x_t|y) - \nabla_{x_t} \log p(x_t) \;=\; \frac{\alpha_t}{\beta_t^2}\Big(D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) \;-\; D_t(x_t) \Big). \quad (14)$$

The exact guidance is the gap between the *posterior* denoiser evaluated at the pivot $\mu_\star$ under the anisotropic covariance $\Sigma_\star(t)$, and the unconditional denoiser evaluated at x_t. Methods that follow the template from [28], including DPS, DDNM, ΠGDM, and moment-matching variants [18–20, 26, 27], all approximate the first denoiser using only the second, by differentiating a measurement loss through $D_t(x_t)$, projecting $D_t(x_t)$ onto an affine subspace, or fitting a Gaussian to $p(x_0|x_t)$ and comparing it to y. They thus evaluate the network at a *different input* than the exact identity, querying D_θ at x_t rather than at the pivot $\mu_\star$, which itself depends on y. Even moment-matching methods that use anisotropic information approximate $p(x_0|x_t)$, the denoising distribution *before* the measurement is incorporated, whereas the exact object is $p(x_0|x_t, y)$, the denoising distribution *after* the measurement is fused into the kernel. The two coincide only in degenerate cases such as isotropic $A^\top A$ or high-noise limits, but not generically.

3.4 The EPS Training Objective

Theorem 1 reduces posterior sampling to a single object, the anisotropic posterior denoiser $D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t))$. Two of the three quantities involved are analytically tractable. Given (x_t, y, t) and the operator parameters (A, σ_y), the pivot $\mu_\star(x_t, y, t)$ and covariance $\Sigma_\star(t)$ are deterministic, closed-form functions defined by (12), and for the structured operators we consider, both can be computed in essentially the cost of an FFT or an element-wise solve (Section 3.2). What we cannot compute analytically is the denoiser itself. The expression $D_{\Sigma_\star(t)}(\mu) = \mathbb{E}[x_0|x_0 + \xi = \mu]$ with $\xi \sim \mathcal{N}(0, \Sigma_\star(t))$ requires the data distribution p_{data}, which is accessible only through data samples, while the pretrained unconditional denoiser was trained with isotropic noise so is not directly applicable (but can be used to initialize the EPS training).

Following standard approaches for training diffusion models, we therefore learn it by regression. To enable efficient noising of x_0 using the anisotropic noise covariance, we note the following result:

Proposition 3 (Isotropic simulation of the anisotropic pivot)**.** *Let $x_t = \alpha_t x_0 + \beta_t \epsilon$ with $\epsilon \sim \mathcal{N}(0, I_d)$ and $y = Ax_0 + \sigma_y \eta$ with $\eta \sim \mathcal{N}(0, I_m)$, independently. Define $\mu_\star(x_t, y, t)$ and $\Sigma_\star(t)$ as in* (12)*. Then, conditional on x_0,*

$$\mu_\star(x_t, y, t)|x_0 \sim \mathcal{N}(x_0, \Sigma_\star(t)). \quad (15)$$

Thus the anisotropic corruption required by the exact posterior denoiser is induced by the closed-form pivot construction itself; it is not necessary to sample anisotropic noise directly.

The proof is given in Appendix A.4. This result is related to existing GP and linear regression literature (Appendix A.7), and shows that the required anisotropic noising can be computed efficiently.

The objective. EPS regresses a denoising network D_θ onto clean targets given the pivot input,

$$\mathcal{L}_{\text{EPS}}(\theta) = \mathbb{E}_{x_0, y, t, \epsilon}\Big[w(t) \; \|D_\theta(\mu_\star(x_t, y, t),\, y,\, t) - x_0\|^2 \Big], \quad (16)$$

where $x_t = \alpha_t x_0 + \beta_t \epsilon$ and $y \sim \mathcal{N}(Ax_0, \sigma_y^2 I_m)$. Standard arguments show that the squared-loss minimizer of (16) is $\mathbb{E}[x_0|\mu_\star, y, t]$, which by Theorem 1 equals $D_{\Sigma_\star(t)}(\mu_\star) = \mathbb{E}[x_0|x_t, y]$. Once trained, the posterior score and posterior velocity follow without further effort from (11) and (13). While strictly unnecessary, note that EPS also passes y to the learned denoiser so that the network is explicitly conditioned on the observed measurement while learning the posterior denoising map; for a fixed operator, the closed-form pivot is a sufficient statistic, but conditioning on y makes the dependence on the particular inverse-problem instance explicit in the learned model, and slightly improves results (Appendix D.1).

The upshot. The minimizer of (16) has the same target type (a clean image $x_0 \in \mathbb{R}^d$) and the same squared-loss regression structure as the standard pretrained denoiser $D_{\theta_0}(x_t, t)$. The

Algorithm 1 EPS training

Require: Pretrained denoiser D_{θ_0} (or random init), data distribution p_{data}, operator distribution $p(A)$, observation noise σ_y, noise schedule (α_t, β_t).

Initialize $\theta \leftarrow \theta_0$ (or randomly).
while not converged **do**
  Sample $x_0 \sim p_{\mathrm{data}}$, $A \sim p(A)$, $t \sim p(t)$, $\epsilon \sim \mathcal{N}(0, I_d)$, and $\eta \sim \mathcal{N}(0, I_m)$.
  Form $y \leftarrow Ax_0 + \sigma_y \eta$ and $x_t \leftarrow \alpha_t x_0 + \beta_t \epsilon$.
  Compute $\Sigma_\star(t)$ and $\mu_\star(x_t, y, t)$ from (12) via the structured solve for A.
  Evaluate the posterior denoising loss $\mathcal{L} = w(t)\|D_\theta(\mu_\star, y, t) - x_0\|^2$.
  Update θ by gradient descent on $\mathcal{L}$, and update EMA weights if used by the base sampler.
end while
return Trained denoiser D_θ.

structural changes are (i) the input is the pivot $\mu_\star$ rather than x_t, and (ii) implicitly, through the pivot construction, the input is corrupted by anisotropic noise of covariance $\Sigma_\star(t)$ rather than isotropic noise of scale β_t. Why does this matter? At the same input $\mu_\star$, the unconditional denoiser D_{θ_0} would return a biased estimate, because it implicitly assumes its input is corrupted by isotropic noise, whereas the true noise on $\mu_\star$ is operator-dependent and anisotropic. The network must therefore learn how to denoise under this measurement-induced anisotropic geometry.

Empirically, when warm-started from the pretrained unconditional denoiser, EPS converges in a small fraction of the iterations needed by other training-based posterior solvers (Appendix D.3). We attribute this to its proximity to the pretraining task. Conditional methods such as Palette [32] and InvFusion [34] preserve the noise schedule and noisy intermediates of pretraining but condition the score on the raw measurement, so the network must spend capacity learning the operator dependence on top of the pretrained denoising prior. Bridge methods such as InDI [35] and I2SB [36] go further by replacing the noise-to-data forward process with a measurement-to-data one, so the network must learn a different conditional mapping from scratch. EPS preserves the forward process and the per-time denoising query, and only adapts to operator-induced anisotropic geometry $\Sigma_\star(t)$.

3.5 Sampling

At inference, EPS uses the same deterministic or stochastic sampler as the underlying diffusion backbone, replacing every denoiser call by $D_\theta(\mu_\star(x_t, y, t), y, t)$. No likelihood gradient, projection, or inner optimization is required during sampling. Since $\mu_\star$ is obtained by a structured linear solve, the per-step overhead is negligible relative to a denoiser forward pass (see Section 3.2).

The High-Noise Limit. Theorem 1 also characterizes the sampler at the start of the reverse trajectory, where the noise scale σ_t is largest. In this regime both the pivot and its anisotropic denoiser take a particularly simple form:

Observation 4 (High-noise posterior-mean limit)**.** *With EDM parameterization $\alpha_t = 1$, $\beta_t = \sigma_t$, and let $P_{\mathcal{N}(A)} = I - A^\dagger A$ be the orthogonal projector onto the nullspace of A. Then, as $\sigma_t \to \infty$,*

$$\mu_\star(x_t, y, t) \longrightarrow A^\dagger y + P_{\mathcal{N}(A)} x_t. \tag{17}$$

Moreover, the corresponding anisotropic denoiser satisfies

$$D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) \longrightarrow \mathbb{E}[x_0|y]. \tag{18}$$

Thus a single high-noise EPS denoiser evaluation is a posterior-mean estimator.

The proof is given in Appendix A.5. The pivot limit (17) is the EPS-specific part of the statement: at the start of sampling, the network is queried at the pseudo-inverse reconstruction $A^\dagger y$ plus pure noise in the nullspace of A. The posterior-mean limit (18), by contrast, is generic rather than a contribution of EPS: by Theorem 1 it is equivalent to $\mathbb{E}[x_0|x_t, y] \to \mathbb{E}[x_0|y]$, which holds simply because x_t carries vanishing information about x_0 as $\sigma_t \to \infty$. Under perfect learning, the same is therefore true of any training-based method whose denoiser targets $\mathbb{E}[x_0|x_t, y]$ along the same path, e.g., Palette [32]; Appendix D.11 confirms this empirically. We nevertheless find it a useful way to read the sampling path of all such methods: it starts at the posterior mean $\mathbb{E}[x_0|y]$ and ends at a sample from $p(x_0|y)$.

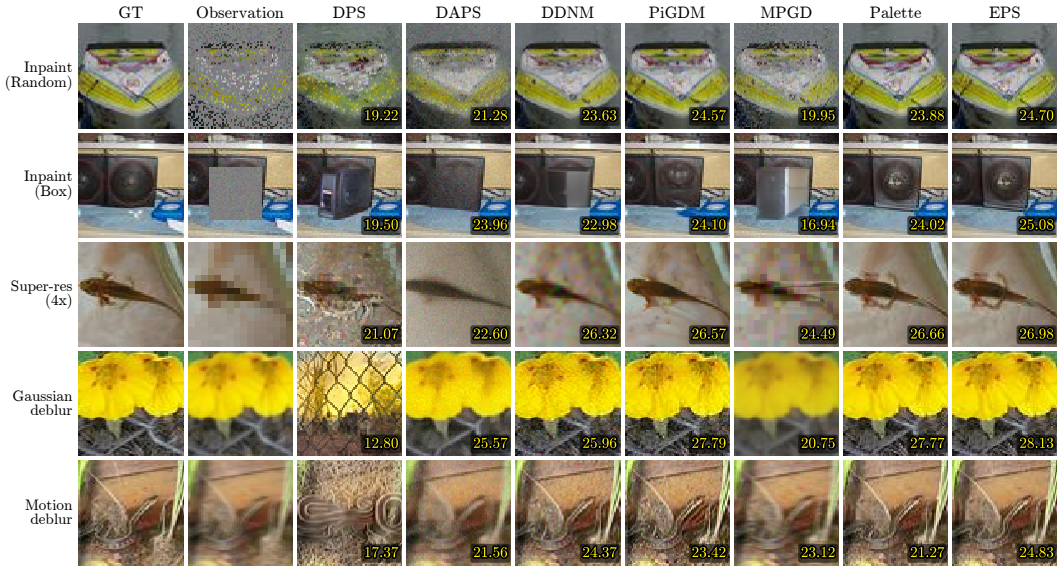

Figure 2: **Qualitative reconstructions across the five inverse problems.** Numbers indicate PSNR values.

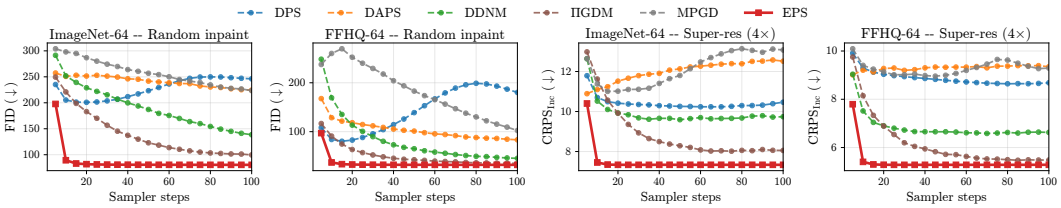

Figure 3: **EPS converges in ∼20 NFE; baselines never catch up.** Sampling-step sensitivity for FID and CRPS-Inception on random inpainting and $4\times$ super-resolution, across both ImageNet-64 and FFHQ-64. EPS plateaus within ∼20 NFE on every panel, while DPS, DAPS, DDNM, ΠGDM, and MPGD continue to improve out to 100 NFE without reaching the EPS asymptote.

4 Experiments

We evaluate EPS on five linear inverse problems across two datasets. The main comparison is at 64×64, where every baseline is run under the same backbone, task, and evaluation protocol and where distributional metrics can be computed fairly. Results at 1024×1024 are in Section 4.3, results at 256×256 in Appendix D.10, and high-resolution text-guided editing in Appendix F.3.

4.1 Experimental Setup

Datasets, backbones, and tasks. We use FFHQ [42] and ImageNet [43]. For ImageNet we use the publicly available class-conditional EDM [17] checkpoint released with the original codebase. For FFHQ we train an EDM checkpoint from scratch because the released FFHQ model does not reserve images for validation. All methods that require training or fine-tuning use the same backbone as EPS, and all zero-shot solvers use the same pretrained denoiser. We consider five linear inverse problems: random inpainting with 70% missing pixels, centered box inpainting, $4\times$ super-resolution, Gaussian deblurring, and motion deblurring. In all cases we add Gaussian observation noise with standard deviation $\sigma_y = 0.05$. Operator details and randomization protocols are in Appendix B.

Baselines and metrics. The sampling-based family comprises DPS [18], DAPS [22], DDNM [19], ΠGDM [20], and MPGD [23]. The training-based family is represented by Palette [32], implemented under the same EDM backbone and compute budget; Palette can be viewed as the EPS pipeline with x_t replacing $\mu_\star$, isolating the contribution of the shifted pivot. We report PSNR and SSIM [44],

Table 1: **Quantitative comparison on ImageNet-64.** Five linear inverse problems, 100 images $\times$ 10 seeds. Baselines follow the sampler and hyperparameters from their respective papers; Palette and EPS use the EDM Euler sampler at 1 NFE per step. Best in **bold**, second-best underlined; EPS rows highlighted in light pink. † The NFE=1 row applies a single Tweedie evaluation $D_\theta(\mu_\star, \sigma_{\max})$, returning the conditional posterior mean $\mathbb{E}[x_0 \mid y]$ in one shot rather than a posterior sample.

Task	Method	NFE	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)	MMD_{pix} (↓)	MMD_{Inc} (↓)	CRPS_{pix} (↓)	CRPS_{Inc} (↓)
Inpaint (random)	DPS	250	21.56	0.6573	0.2158	183.94	-4.88e-03	2.51e-02	5.88	9.89
	DAPS	100	20.89	0.5555	0.3168	224.39	-2.22e-03	4.94e-02	6.91	12.09
	DDNM	100	22.63	0.7127	0.1758	138.53	-5.78e-03	7.29e-03	5.23	7.99
	ΠGDM	100	23.95	0.7780	0.1198	99.60	-6.40e-03	-2.22e-03	4.54	6.32
	MPGD	100	19.62	0.5447	0.3151	223.43	2.56e-03	4.92e-02	7.81	11.94
	Palette	100	24.09	0.7869	0.1011	81.88	-6.50e-03	-4.41e-03	4.16	5.52
	EPS (ours)	100	24.34	0.7948	0.0979	79.60	**-6.52e-03**	**-4.51e-03**	4.04	**5.41**
	EPS (ours)	20	24.87	0.8122	**0.0910**	**77.06**	**-6.52e-03**	**-4.51e-03**	**4.02**	5.44
	EPS (ours)†	1	**26.60**	**0.8580**	0.0933	88.59	-6.15e-03	-4.84e-04	4.98	8.32
Inpaint (box)	DPS	250	19.44	0.6587	0.1891	142.05	-5.63e-03	1.01e-02	7.25	8.19
	DAPS	100	20.62	0.6600	0.2100	161.62	-5.00e-03	2.38e-02	7.36	9.57
	DDNM	100	20.38	0.7049	0.1824	125.82	-5.37e-03	4.18e-03	7.00	7.54
	ΠGDM	100	20.53	0.7292	0.1490	104.50	-5.87e-03	-2.69e-03	6.59	6.50
	MPGD	100	19.35	0.6467	0.2463	167.19	-3.44e-03	1.93e-02	8.67	9.68
	Palette	100	21.12	0.7541	0.1218	92.73	-6.10e-03	-4.12e-03	5.92	5.93
	EPS (ours)	100	21.24	0.7569	0.1196	91.07	**-6.11e-03**	**-4.23e-03**	5.87	**5.84**
	EPS (ours)	20	21.72	0.7667	**0.1166**	**90.12**	-6.08e-03	-4.14e-03	**5.86**	5.90
	EPS (ours)†	1	**23.60**	**0.7908**	0.1514	129.11	-5.60e-03	7.34e-03	7.16	10.38
Super-res. (4×)	DPS	250	19.59	0.4511	0.3367	200.98	-5.06e-03	2.67e-02	7.77	10.36
	DAPS	100	18.46	0.3324	0.4772	258.25	6.09e-03	8.54e-02	10.01	12.52
	DDNM	100	21.10	0.5523	0.3055	169.94	-5.10e-03	1.41e-02	7.32	9.74
	ΠGDM	100	20.25	0.5318	0.2499	144.36	-5.78e-03	4.61e-03	6.76	8.05
	MPGD	100	20.57	0.5103	0.3426	221.50	-3.20e-03	6.97e-02	9.02	13.08
	Palette	100	20.24	0.5364	0.2220	**128.76**	**-5.86e-03**	-2.79e-03	**6.50**	**7.33**
	EPS (ours)	100	20.25	0.5369	0.2207	128.80	**-5.86e-03**	**-2.84e-03**	6.52	7.35
	EPS (ours)	20	20.90	0.5692	**0.2127**	129.13	-5.72e-03	-1.98e-03	6.51	7.54
	EPS (ours)†	1	**22.78**	**0.6530**	0.2455	182.92	-4.97e-03	1.98e-02	8.06	13.47
Gaussian deblur	DPS	250	25.05	0.7804	0.1511	158.46	-5.24e-03	1.57e-02	4.48	8.96
	DAPS	100	25.97	0.7680	0.1466	152.47	-6.50e-03	2.78e-02	3.69	9.58
	DDNM	100	26.28	0.7903	0.1227	101.61	-6.71e-03	9.01e-03	2.92	7.18
	ΠGDM	100	28.13	0.8785	0.0742	66.10	-6.70e-03	-2.49e-03	2.91	5.68
	MPGD	100	20.74	0.6375	0.2396	175.42	2.24e-02	2.13e-02	8.01	10.20
	Palette	100	29.15	0.9010	0.0491	46.62	**-6.83e-03**	**-5.56e-03**	2.26	**4.11**
	EPS (ours)	100	29.18	0.9015	0.0486	46.55	**-6.83e-03**	-5.55e-03	2.25	**4.11**
	EPS (ours)	20	**29.68**	0.9108	**0.0449**	**44.25**	**-6.83e-03**	-5.55e-03	**2.24**	4.13
	EPS (ours)†	1	28.82	**0.9194**	0.0606	56.53	-5.14e-03	-3.46e-03	4.09	7.73
Motion deblur	DPS	250	25.63	0.7981	0.1398	162.66	-6.41e-03	2.58e-02	4.42	9.82
	DAPS	100	24.07	0.6859	0.1863	190.39	-6.06e-03	3.99e-02	4.68	11.19
	DDNM	100	25.73	0.7758	0.1299	109.80	-6.68e-03	7.60e-03	3.16	7.21
	ΠGDM	100	25.89	0.8205	0.1031	93.75	-6.46e-03	-6.33e-04	3.72	6.63
	MPGD	100	25.90	0.8114	0.1525	183.11	-6.09e-03	3.34e-02	5.28	12.49
	Palette	100	27.02	0.8582	0.0680	62.86	-6.73e-03	-5.02e-03	2.93	4.81
	EPS (ours)	100	27.62	0.8661	0.0647	61.29	**-6.77e-03**	**-5.11e-03**	**2.69**	**4.70**
	EPS (ours)	20	**28.17**	**0.8791**	**0.0606**	**59.11**	**-6.77e-03**	-5.05e-03	**2.69**	4.76
	EPS (ours)†	1	26.56	0.8613	0.1079	97.59	-6.03e-03	-7.30e-04	5.25	9.91

LPIPS [45], and FID [46] for pointwise and perceptual quality, and CRPS [47] and MMD [48] in pixel and Inception feature space [49, 50] for distributional calibration. We use raw Inception features rather than the L2-normalized features of Mammadov et al. [49], so absolute values differ but relative comparisons across methods are preserved. Definitions are in Appendix I.

4.2 Main Results

Tables 1 and 8 report all five tasks, all baselines, and all metrics in a single view per dataset. We include EPS at 100, 20, and 1 NFE: the first two correspond to the posterior-sampling regime at different budgets, and the 1-NFE row tests the posterior-mean prediction from Section 3.5.

Quantitative comparison and sampling efficiency. EPS attains the best or second-best score on every metric and task in Tables 1 and 8 , across pointwise fidelity (PSNR, SSIM), perceptual quality (LPIPS, FID), and distributional calibration (CRPS, MMD). The closest competitor is usually Palette, which shares the EPS pipeline except that the network input is x_t rather than $\mu_\star$; holding

Table 2: **EPS scales to 1024×1024.** FFHQ-1024, 100 held-out faces $\times$ 10 seeds, all methods on one pretrained flow prior (EPS fine-tuned, baselines zero-shot). All metrics, error bars, and the gradient-guided baselines (ΠGDM, OPC, PSLD, all below 14 dB) are in Table 20.

Method	SR 16×			SR 64×			Box inpaint (40%)			Gaussian deblur		
	PSNR	LPIPS	FID	PSNR	LPIPS	FID	PSNR	LPIPS	FID	PSNR	LPIPS	FID
FlowDPS	23.84	0.528	132.9	18.93	0.578	345.4	15.73	0.411	209.9	27.83	0.351	32.7
FlowChef	22.57	0.582	137.6	17.66	0.627	318.0	15.40	0.389	216.0	27.80	0.370	39.6
EPS (ours)	**25.31**	**0.361**	**42.7**	**19.56**	**0.479**	**91.1**	**18.62**	**0.278**	**56.1**	**30.33**	**0.240**	**10.7**

the backbone, compute budget, and training data fixed, EPS still outperforms Palette on every task, isolating the contribution of the shifted pivot. Sampling-based baselines that approximate the measurement-matching score (ΠGDM, DPS, DDNM, DAPS, MPGD) mostly lag further behind, even at substantially larger NFE budgets (100 NFE; DPS at 250), and the gap widens where the operator is most ill-conditioned, namely random inpainting with 70% missing pixels and $4\times$ super-resolution. Figure 3 confirms the same trend across budgets: EPS plateaus within ~ 20 NFE, while every baseline keeps improving out to 100 NFE without reaching the EPS asymptote. The 1-NFE row is generally the strongest pointwise estimator on PSNR and SSIM, consistent with Observation 4, trading distributional calibration for pointwise sharpness, in line with the perception-distortion trade-off [51].

More baselines, error bars, and calibration. Appendix D.15 adds OPC [52] and an InvFusion-style model [34] (operator given as $A^\top A$, no pivot) and reports error bars. EPS beats OPC in FID, CRPS, and MMD on every task and by 1.6 and 5.7 dB in PSNR on deblurring, and at a matched training budget it beats the InvFusion-style model on every task, by up to 4.0 dB. On most tasks, EPS has the best distributional metrics (FID, CRPS, and MMD). EPS has the lowest rank-calibration χ^2 in 8 of 9 settings and passes in 6, while no baseline passes in any (Appendix E.2), and human raters prefer it in about a third of judgments, against at most 20% for any other method (Appendix D.17).

Qualitative comparison and ablations. Figure 2 shows reconstructions on the five tasks. The largest visual gaps appear under aggressive inpainting and deblurring, where DPS, DAPS, and MPGD oversmooth or introduce texture inconsistent with the measurement, and DDNM and ΠGDM match measured directions but leave the unmeasured subspace blurry. EPS preserves sharp prior structure while matching the observation, since the pivot $\mu_\star$ explicitly separates measured and unmeasured directions and $\Sigma_\star(t)$ specifies how much to denoise along each. Appendix D studies the input pivot, zero-shot behavior, warm-start convergence, sampling-step sensitivity, amortization across tasks, and 256×256 scaling, and confirms the two central mechanisms: the shifted pivot is the right input, and preserving pretrained denoising marginals explains the fast convergence.

4.3 Scaling to 1024×1024

We fine-tune EPS on FFHQ-1024 from a pretrained pixel-space rectified-flow model (MiniT2I-B/16 [53]) and compare it with zero-shot solvers on the same model: FlowDPS [54], FlowChef [55], ΠGDM, OPC, and PSLD [56]. On $16\times$ and $64\times$ super-resolution (the latter from a 16×16 image), 40% box inpainting, and Gaussian deblurring, with 100 held-out faces and 10 samples each, EPS is best on every task and metric (Table 2), with FID three to four times lower than the best baseline; the gradient-guided solvers fail to constrain the prior at this scale. On deblurring, with each baseline's guidance tuned on held-out faces, EPS is 2.5 dB above FlowDPS and better on 99 of 100 faces. Appendix F adds the protocol, figures, and single models for all super-resolution factors at 512 and 1024. Fine-tuned with captions, EPS also lets a text prompt edit details the measurement leaves open, such as glasses or a beard, while the output stays consistent with y (Appendix F.3).

5 Related Work

Posterior sampling with pretrained generative priors. A large literature uses pretrained diffusion or flow models as priors for inverse problems. Explicit guidance methods, including Score-SDE/ALD, RePaint, DDNM, DDRM, DPS, ΠGDM, DAPS, MPGD, and FlowDPS [57, 58, 19, 21, 18, 20, 22, 23, 54, 28], approximate the measurement-matching score $\nabla_{x_t}\log p(y|x_t) \approx -L_t M_t/G_t$ [28], where M_t is a measurement residual, L_t lifts it back to the sample space, and G_t controls the guidance

strength. They differ in the form of M_t, L_t, and G_t, instantiating the template via projections, denoised estimates, Jacobian corrections, or moment approximations of $p(x_0|x_t)$. Moment-matching variants [26, 27] go beyond first-order Tweedie by approximating $p(x_0|x_t)$ with an anisotropic Gaussian, and OPC [52] uses the optimal covariance for this Gaussian. Section 3.3 makes precise the gap between all of these and the exact posterior score, namely that each method substitutes the unconditional denoising query for the posterior denoising query, querying the network at x_t rather than at the pivot $\mu_\star$. GLASS [59] is the closest training-free relative, and its equivalent-time formula coincides with EPS in the special case where $A^\top A$ is a scalar multiple of the identity, equivalently when $\Sigma_\star(t)$ is isotropic (Appendix A.6). EPS handles the general operator-dependent anisotropic case at the cost of a training step.

Exact posterior scores and denoiser priors. Kadkhodaie and Simoncelli [60] sample noiseless linear inverse problems with an isotropic denoiser prior and a projection onto the measurement, i.e., the unconditional query of Section 3.3, with no problem-specific training. Bellchambers [61] derives the posterior score for pure denoising (our equivalent-time case, Appendix A.6), expresses inpainting through a non-isotropic score that is not generally available, and uses it training-free to correct DPS step sizes. Our Theorem 1 covers general linear operators and interpolants, and Proposition 3 makes the missing denoiser trainable. Unrolled and plug-and-play solvers [62–65] embed learned denoisers or proximal modules in a fixed-depth data-consistency iteration and target a point estimate. EPS instead derives the corruption in closed form and targets posterior samples, and its depth is the sampler budget.

Conditional training and restoration bridges. Palette and conditional image-to-image diffusion models train a network directly on (x_t, y) pairs [32–34], amortized variational methods such as APS [37] and DAVI [39] train a network that maps y to posterior samples under a diffusion prior, and bridge-based restoration methods such as InDI [35] and image-to-image Schrödinger bridges [36] construct trajectories from the measurement distribution to the data distribution. These methods are expressive and avoid hand-designed corrections, but they expose the network to a conditional path whose intermediate marginals do not match the unconditional denoising marginals of the prior, so they cannot leverage a pretrained denoiser as a warm start in a structurally aligned way. Zilberstein et al. [66] learn one normalized energy model and regularize its marginals to agree across operators. EPS instead derives the conditional path induced by the exact linear-Gaussian posterior kernel, which preserves the input/output type of standard denoising pretraining (Section 3.4). This makes both random initialization and warm-starting from a pretrained checkpoint natural training options. We view bridge-based methods as complementary, since they define useful restoration dynamics for general degradations, while EPS identifies the bridge that solves the linear-Gaussian posterior exactly.

6 Conclusion

We derived the exact posterior score for linear Gaussian inverse problems and showed that posterior sampling reduces to a denoising problem at a measurement-induced pivot $\mu_\star$ under an operator-dependent anisotropic covariance $\Sigma_\star$. We turned this identity into EPS, a denoising training objective whose input/output structure matches standard pretraining, so it can be trained from scratch or fine-tuned efficiently from a pretrained checkpoint. EPS samples with the underlying backbone's sampler, requires no measurement gradients or projections, and admits a one-evaluation posterior-mean estimator in the high-noise limit. Empirically, EPS improves both reconstruction fidelity and distributional calibration over sampling-based and conditional-training baselines on five linear inverse problems on FFHQ and ImageNet, and outperforms zero-shot flow-based solvers at 1024×1024.

Limitations. The theory assumes a known linear operator and Gaussian noise, so nonlinear and blind problems are future work. Nonlinear operators could use local linearization or the true likelihood in training, and blind problems would pair EPS with operator and noise estimation (Appendix D.16 studies a mis-specified σ_y). EPS trains per operator family, which can be broad (Appendix D.5); a new family needs a short fine-tune. Unstructured operators can use conjugate gradients or a one-time factorization, but a very large dense A with neither structure nor an affordable factorization is a real cost, shared with solvers that use the pseudo-inverse $A^\dagger$, such as DDNM and ΠGDM. Latent backbones need care, since the decoder makes a linear pixel-space operator nonlinear in latent space. Appendix G discusses when training pays off.

Acknowledgments and Disclosure of Funding

AM is supported by the Clarendon Fund Scholarship, University of Oxford. We thank fal for the compute grants that supported this research. The authors acknowledge the support of the Health Care Engineering Systems Center within the Coordinated Science Lab at UIUC. This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) through the AI Hub in Generative Models [grant number EP/Y028805/1]. The authors acknowledge the use of resources provided by the Isambard-AI National AI Research Resource (AIRR) [67]. Isambard-AI is operated by the University of Bristol and is funded by the UK Government's Department for Science, Innovation and Technology (DSIT) via UK Research and Innovation; and the Science and Technology Facilities Council [ST/AIRR/I-A-I/1023].

References

[1] David L Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4): 1289–1306, 2006.

[2] Emmanuel J Candès, Justin Romberg, and Terence Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[3] Michael Lustig, David Donoho, and John M Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, 2007.

[4] Florian Knoll, Jure Zbontar, Anuroop Sriram, Matthew J Muckley, Mary Bruno, Aaron Defazio, Marc Parente, Krzysztof J Geras, Joe Katsnelson, Hersh Chandarana, et al. fastMRI: A publicly available raw k-space and DICOM dataset of knee images for accelerated MR image reconstruction using machine learning. *Radiology: Artificial Intelligence*, 2(1):e190007, 2020.

[5] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199. Springer, 2014.

[6] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-ESRGAN: Training real-world blind super-resolution with pure synthetic data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1905–1914, 2021.

[7] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3883–3891, 2017.

[8] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. DeblurGAN: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8183–8192, 2018.

[9] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424, 2000.

[10] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016.

[11] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 2015.

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

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

[14] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

[15] Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209): 1–80, 2025.

[16] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

[17] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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

[19] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *International Conference on Learning Representations*, 2023.

[20] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=9_gsMA8MRKQ`.

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

[22] Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20895–20905, 2025.

[23] Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J. Zico Kolter, Ruslan Salakhutdinov, and Stefano Ermon. Manifold preserving guided diffusion. In *International Conference on Learning Representations*, 2024.

[24] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *Advances in Neural Information Processing Systems*, 35:25683–25696, 2022.

[25] Morteza Mardani, Jiaming Song, Jan Kautz, and Arash Vahdat. A variational perspective on solving inverse problems with diffusion models. In *International Conference on Learning Representations*, 2024.

[26] François Rozet, Gérôme Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. *Advances in Neural Information Processing Systems*, 37:87647–87682, 2024.

[27] Litu Rout, Yujia Chen, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. Beyond first-order Tweedie: Solving inverse problems using latent diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9472–9481, 2024.

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

[29] Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36:31372–31403, 2023.

[30] Gabriel Cardoso, Yazid Janati El Idrissi, Sylvain Le Corff, and Eric Moulines. Monte Carlo guided denoising diffusion models for Bayesian linear inverse problems. In *International Conference on Learning Representations*, 2024.

[31] Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

[32] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022.

[33] Georgios Batzolis, Jan Stanczuk, Carola-Bibiane Schönlieb, and Christian Etmann. Conditional image generation with score-based diffusion models. *arXiv preprint arXiv:2111.13606*, 2021.

[34] Noam Elata, Hyungjin Chung, Jong Chul Ye, Tomer Michaeli, and Michael Elad. InvFusion: Bridging supervised and zero-shot diffusion for inverse problems. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

[35] Mauricio Delbracio and Peyman Milanfar. Inversion by direct iteration: An alternative to denoising diffusion for image restoration. *Transactions on Machine Learning Research*, 2023.

[36] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2SB: Image-to-image Schrödinger bridge. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, 2023.

[37] Abbas Mammadov, Hyungjin Chung, and Jong Chul Ye. Amortized posterior sampling with diffusion prior distillation. *arXiv preprint arXiv:2407.17907*, 2024.

[38] Berthy T Feng, Jamie Smith, Michael Rubinstein, Huiwen Chang, Katherine L Bouman, and William T Freeman. Score-based diffusion models as principled priors for inverse imaging. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10520–10531, 2023.

[39] Sojin Lee, Dogyun Park, Inho Kong, and Hyunwoo J. Kim. Diffusion prior-based amortized variational inference for noisy inverse problems. In *European Conference on Computer Vision*, 2024.

[40] Herbert E Robbins. An empirical Bayes approach to statistics. In *Breakthroughs in Statistics: Foundations and Basic Theory*, pages 388–394. Springer, 1992.

[41] Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011.

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

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

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

[45] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

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

[47] Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151, 2014.

[48] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(1):723–773, 2012.

[49] Abbas Mammadov, So Takao, Bohan Chen, Ricardo Baptista, Morteza Mardani, Yee Whye Teh, and Julius Berner. Variational flow maps: Make some noise for one-step conditional generation. *arXiv preprint arXiv:2603.07276*, 2026.

[50] Valentin De Bortoli, Alexandre Galashov, J. Swaroop Guntupalli, Guangyao Zhou, Kevin Murphy, Arthur Gretton, and Arnaud Doucet. Distributional diffusion models with scoring rules, 2025. URL https://arxiv.org/abs/2502.02483.

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

[52] Xinyu Peng, Ziyang Zheng, Wenrui Dai, Nuoqian Xiao, Chenglin Li, Junni Zou, and Hongkai Xiong. Improving diffusion models for inverse problems using optimal posterior covariance. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 40347–40370, 2024.

[53] Xianbang Wang, Hanhong Zhao, Yiyang Lu, Kangyang Zhou, Linrui Ma, and Kaiming He. MiniT2I: A minimalist baseline for text-to-image synthesis. https://peppaking8.github.io/#/post/minit2i, 2026.

[54] Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12328–12337, 2025.

[55] Maitreya Patel, Song Wen, Dimitris N Metaxas, and Yezhou Yang. FlowChef: Steering of rectified flow models for controlled generations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15308–15318, 2025.

[56] Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alex Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:49960–49990, 2023.

[57] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

[58] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. RePaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[59] Peter Holderrieth, Uriel Singer, Tommi Jaakkola, Ricky T. Q. Chen, Yaron Lipman, and Brian Karrer. GLASS flows: Efficient inference for reward alignment of flow and diffusion models. In *International Conference on Learning Representations*, 2026.

[60] Zahra Kadkhodaie and Eero P. Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. In *Advances in Neural Information Processing Systems*, volume 34, pages 13242–13254, 2021.

[61] Gregory Bellchambers. Exploiting the exact denoising posterior score in training-free guidance of diffusion models. *arXiv preprint arXiv:2506.13614*, 2025.

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

[63] Kai Zhang, Yawei Li, Wangmeng Zuo, Lei Zhang, Luc Van Gool, and Radu Timofte. Plug-and-play image restoration with deep denoiser prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6360–6376, 2022.

[64] Hemant K. Aggarwal, Merry P. Mani, and Mathews Jacob. MoDL: Model-based deep learning architecture for inverse problems. *IEEE Transactions on Medical Imaging*, 38(2):394–405, 2019.

[65] Vishal Monga, Yuelong Li, and Yonina C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[66] Nicolas Zilberstein, Santiago Segarra, Eero Simoncelli, and Florentin Guth. Learning normalized energy models for linear inverse problems. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, 2026.

[67] Simon McIntosh-Smith, Sadaf R Alam, and Christopher Woods. Isambard-AI: A leadership class supercomputer optimised specifically for artificial intelligence, 2024. URL `https://arxiv.org/abs/2410.11199`.

[68] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[69] Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. PixelDiT: Pixel diffusion transformers for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

[70] Jordan Meyer, Nick Padgett, Cullen Miller, and Laura Exline. Public domain 12M: A highly aesthetic image-text dataset with novel governance mechanisms. *arXiv preprint arXiv:2410.23144*, 2024.

[71] Haozhe Zhao, Xiaojian Ma, Liang Chen, Shuzheng Si, Rujie Wu, Kaikai An, Peiyu Yu, Minjia Zhang, Qing Li, and Baobao Chang. UltraEdit: Instruction-based fine-grained image editing at scale. In *Advances in Neural Information Processing Systems*, 2024.

[72] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The Open Images dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *International Journal of Computer Vision*, 128(7):1956–1981, 2020.

[73] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[74] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[75] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems*, 2021.

[76] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. MaskGAN: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[77] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision*, 2014.

[78] Agrim Gupta, Piotr Dollár, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Appendix Contents

A Theory and Derivations **18**
- A.1 Anisotropic Tweedie Formula . . . 18
- A.2 Proof of Theorem 1 . . . 18
- A.3 Proof of Proposition 2 . . . 19
- A.4 Proof of Proposition 3 . . . 20
- A.5 Proof of Observation 4 . . . 20
- A.6 Equivalent-Time and GLASS Limit . . . 21
- A.7 Connection to ridge regression and Gaussian processes. . . . 21

B Implementation Details **22**

C Broader Impact **23**

D Additional Experiments **24**
- D.1 Input Configuration Ablation . . . 24
- D.2 Zero-Shot Pivoting . . . 27
- D.3 Palette vs EPS . . . 28
- D.4 Sampling Efficiency . . . 29
- D.5 Amortized Variant Across All Five Tasks . . . 32
- D.6 One-Step Posterior Mean Check . . . 33
- D.7 Additional 64×64 Results . . . 34
- D.8 Extreme Tasks 64×64 . . . 37
- D.9 OOD Mask-Density Experiments . . . 39
- D.10 Additional 256×256 Results . . . 41
- D.11 Palette vs. EPS at Matched NFE . . . 43
- D.12 Runtime Analysis . . . 45
- D.13 Posterior Diversity . . . 46
- D.14 Sampling Budget . . . 48
- D.15 Additional Baselines and Error Bars . . . 49
- D.16 Sensitivity to the Measurement Noise Level . . . 51
- D.17 Human Preference Study . . . 52

E Exactness and Posterior Calibration **54**
- E.1 Gaussian-Mixture Benchmark . . . 54
- E.2 Calibration of the Posterior Samples . . . 54

F High-Resolution Experiments **56**
- F.1 Controlled Comparison at 1024×1024 . . . 56
- F.2 General-Purpose Models at 512×512 and 1024×1024 . . . 59
- F.3 Text-Guided Editing . . . 61

G Practical Considerations **65**

H Baseline Configurations **67**

I Metric Definitions **68**

A Theory and Derivations

This appendix gives the full derivation of the EPS score identity, the posterior velocity, the equivalent-time special case, and the one-step posterior-mean limit. We keep the assumptions of the main text: $x_t = \alpha_t x_0 + \beta_t \epsilon$ with $\epsilon \sim \mathcal{N}(0, I_d)$ and $y = Ax_0 + \eta$ with $\eta \sim \mathcal{N}(0, \sigma_y^2 I_m)$.

A.1 Anisotropic Tweedie Formula

For any positive definite Σ, define

$$p_{\mathrm{data}}^{\Sigma}(\mu) = \int \mathcal{N}(\mu; x_0, \Sigma) p_{\mathrm{data}}(x_0)\, \mathrm{d}x_0, \qquad D_\Sigma(\mu) = \mathbb{E}[x_0 | x_0 + \xi = \mu], \quad \xi \sim \mathcal{N}(0, \Sigma). \tag{19}$$

Then

$$D_\Sigma(\mu) = \mu + \Sigma \nabla_\mu \log p_{\mathrm{data}}^{\Sigma}(\mu). \tag{20}$$

Indeed, differentiating under the integral gives

$$\nabla_\mu p_{\mathrm{data}}^{\Sigma}(\mu) = \int \nabla_\mu \mathcal{N}(\mu; x_0, \Sigma) p_{\mathrm{data}}(x_0)\, \mathrm{d}x_0 \tag{21}$$

$$= \int \mathcal{N}(\mu; x_0, \Sigma) \Sigma^{-1}(x_0 - \mu) p_{\mathrm{data}}(x_0)\, \mathrm{d}x_0. \tag{22}$$

Dividing by $p_{\mathrm{data}}^{\Sigma}(\mu)$ yields

$$\nabla_\mu \log p_{\mathrm{data}}^{\Sigma}(\mu) = \Sigma^{-1} \left(\mathbb{E}[x_0 | x_0 + \xi = \mu] - \mu \right), \tag{23}$$

which proves (20). The key point for EPS is that Σ is a full covariance matrix fixed by the inverse problem, not a scalar noise level chosen to match an unconditional diffusion time.

A.2 Proof of Theorem 1

Theorem 1 restated. *Under the linear Gaussian inverse problem* (5) *and the interpolant* (1)*, the posterior score at time t is*

$$\nabla_{x_t} \log p(x_t | y) = \frac{1}{\beta_t^2} \Big(\alpha_t D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) - x_t \Big), \tag{24}$$

where

$$\Sigma_\star(t) = \left(\frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A \right)^{-1}, \qquad \mu_\star(x_t, y, t) = \Sigma_\star(t) \left(\frac{\alpha_t}{\beta_t^2} x_t + \frac{1}{\sigma_y^2} A^\top y \right). \tag{25}$$

Equivalently, $D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) = \mathbb{E}[x_0 | x_t, y]$.

Proof. By conditional independence of x_t and y given x_0,

$$p(x_t | y) = \frac{1}{p(y)} \int p(x_t | x_0) p(y | x_0) p_{\mathrm{data}}(x_0)\, \mathrm{d}x_0. \tag{26}$$

The two Gaussian factors are

$$p(x_t | x_0) = (2\pi\beta_t^2)^{-d/2} \exp\left(-\frac{\|x_t - \alpha_t x_0\|^2}{2\beta_t^2} \right), \tag{27}$$

$$p(y | x_0) = (2\pi\sigma_y^2)^{-m/2} \exp\left(-\frac{\|y - Ax_0\|^2}{2\sigma_y^2} \right). \tag{28}$$

Collect the exponent terms that depend on x_0:

$$-\frac{\|x_t - \alpha_t x_0\|^2}{2\beta_t^2} - \frac{\|y - Ax_0\|^2}{2\sigma_y^2} \tag{29}$$

$$= -\frac{1}{2} x_0^\top \underbrace{\left(\frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A \right)}_{\Lambda_t} x_0 + \underbrace{\left(\frac{\alpha_t}{\beta_t^2} x_t + \frac{1}{\sigma_y^2} A^\top y \right)^\top}_{b_t^\top} x_0 - \frac{\|x_t\|^2}{2\beta_t^2} - \frac{\|y\|^2}{2\sigma_y^2}. \tag{30}$$

Because $\beta_t > 0$, the matrix Λ_t is positive definite even when A is rank deficient. Let $\Sigma_\star = \Lambda_t^{-1}$ and $\mu_\star = \Sigma_\star b_t$. Completing the square in (30),

$$-\frac{1}{2}x_0^\top \Lambda_t x_0 + b_t^\top x_0 = -\frac{1}{2}(x_0 - \mu_\star)^\top \Sigma_\star^{-1}(x_0 - \mu_\star) + \frac{1}{2}b_t^\top \Sigma_\star b_t. \tag{31}$$

Therefore the product of the two likelihoods can be factored as

$$p(x_t|x_0)p(y|x_0) = C_t(x_t, y)\,\mathcal{N}(x_0; \mu_\star, \Sigma_\star), \tag{32}$$

where all dependence on x_0 is contained in the displayed Gaussian and

$$\log C_t(x_t, y) = \text{const}(t) - \frac{\|x_t\|^2}{2\beta_t^2} - \frac{\|y\|^2}{2\sigma_y^2} + \frac{1}{2}b_t^\top \Sigma_\star b_t. \tag{33}$$

Substituting (32) into (26) gives

$$p(x_t|y) = \frac{C_t(x_t, y)}{p(y)}\, p_{\text{data}}^{\Sigma_\star}(\mu_\star), \qquad p_{\text{data}}^{\Sigma_\star}(\mu_\star) = \int \mathcal{N}(x_0; \mu_\star, \Sigma_\star) p_{\text{data}}(x_0)\,\mathrm{d}x_0. \tag{34}$$

Now differentiate (34) with respect to x_t. Since $\Sigma_\star$ does not depend on x_t and $\partial b_t/\partial x_t = (\alpha_t/\beta_t^2)I_d$,

$$\frac{\partial \mu_\star}{\partial x_t} = \frac{\alpha_t}{\beta_t^2}\Sigma_\star. \tag{35}$$

The normalizer derivative is

$$\nabla_{x_t} \log C_t(x_t, y) = -\frac{x_t}{\beta_t^2} + \frac{1}{2}\nabla_{x_t}\left(b_t^\top \Sigma_\star b_t\right) \tag{36}$$

$$= -\frac{x_t}{\beta_t^2} + \frac{\alpha_t}{\beta_t^2}\Sigma_\star b_t = -\frac{x_t}{\beta_t^2} + \frac{\alpha_t}{\beta_t^2}\mu_\star. \tag{37}$$

The smoothed-density derivative is, by the chain rule and (35),

$$\nabla_{x_t} \log p_{\text{data}}^{\Sigma_\star}(\mu_\star) = \frac{\alpha_t}{\beta_t^2}\Sigma_\star \nabla_\mu \log p_{\text{data}}^{\Sigma_\star}(\mu)\big|_{\mu=\mu_\star}. \tag{38}$$

Using the anisotropic Tweedie identity (20),

$$\Sigma_\star \nabla_\mu \log p_{\text{data}}^{\Sigma_\star}(\mu_\star) = D_{\Sigma_\star}(\mu_\star) - \mu_\star. \tag{39}$$

Combining (37), (38), and (39), the pivot terms cancel:

$$\nabla_{x_t} \log p(x_t|y) = -\frac{x_t}{\beta_t^2} + \frac{\alpha_t}{\beta_t^2}\mu_\star + \frac{\alpha_t}{\beta_t^2}\left(D_{\Sigma_\star}(\mu_\star) - \mu_\star\right) \tag{40}$$

$$= \frac{1}{\beta_t^2}\left(\alpha_t D_{\Sigma_\star}(\mu_\star) - x_t\right). \tag{41}$$

Finally, under the joint density proportional to $\mathcal{N}(x_0; \mu_\star, \Sigma_\star)p_{\text{data}}(x_0)$, the posterior mean of x_0 is exactly $D_{\Sigma_\star}(\mu_\star)$, so $D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) = \mathbb{E}[x_0|x_t, y]$. □

A.3 Proof of Proposition 2

For brevity in this proof we write $D^\star(x_t, y, t) \coloneqq D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t))$, which by Theorem 1 equals $\mathbb{E}[x_0|x_t, y]$. The interpolant satisfies $x_t = \alpha_t x_0 + \beta_t \epsilon$, hence conditional on (x_t, y),

$$\mathbb{E}[\epsilon|x_t, y] = \frac{x_t - \alpha_t \mathbb{E}[x_0|x_t, y]}{\beta_t} = \frac{x_t - \alpha_t D^\star(x_t, y, t)}{\beta_t}. \tag{42}$$

The posterior velocity is the conditional expectation of the path derivative:

$$v_t^y(x_t) = \mathbb{E}[\dot\alpha_t x_0 + \dot\beta_t \epsilon|x_t, y] \tag{43}$$

$$= \dot\alpha_t D^\star(x_t, y, t) + \dot\beta_t \frac{x_t - \alpha_t D^\star(x_t, y, t)}{\beta_t} \tag{44}$$

$$= \frac{\dot\beta_t}{\beta_t}x_t + \left(\dot\alpha_t - \frac{\alpha_t \dot\beta_t}{\beta_t}\right) D^\star(x_t, y, t). \tag{45}$$

Thus estimating the exact posterior denoiser is equivalent to estimating the exact posterior velocity for the interpolant.

A.4 Proof of Proposition 3

Let

$$\Lambda_t = \frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A, \qquad \Sigma_\star(t) = \Lambda_t^{-1}. \tag{46}$$

Using $x_t = \alpha_t x_0 + \beta_t \epsilon$ and $y = Ax_0 + \sigma_y \eta$, the pivot is

$$\mu_\star = \Sigma_\star \left(\frac{\alpha_t}{\beta_t^2} x_t + \frac{1}{\sigma_y^2} A^\top y \right) \tag{47}$$

$$= \Sigma_\star \left[\left(\frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A \right) x_0 + \frac{\alpha_t}{\beta_t} \epsilon + \frac{1}{\sigma_y} A^\top \eta \right] \tag{48}$$

$$= x_0 + \Sigma_\star \left(\frac{\alpha_t}{\beta_t} \epsilon + \frac{1}{\sigma_y} A^\top \eta \right). \tag{49}$$

The noise term is Gaussian with mean zero and covariance

$$\Sigma_\star \left(\frac{\alpha_t^2}{\beta_t^2} I_d + \frac{1}{\sigma_y^2} A^\top A \right) \Sigma_\star = \Sigma_\star \Lambda_t \Sigma_\star = \Sigma_\star. \tag{50}$$

Therefore $\mu_\star | x_0, t \sim \mathcal{N}(x_0, \Sigma_\star(t))$, as claimed.

A.5 Proof of Observation 4

For EDM, $\alpha_t = 1$ and $\beta_t = \sigma_t$. Let

$$A = U_r S V_r^\top \tag{51}$$

be the compact SVD of A, where $S = \mathrm{diag}(s_1, \ldots, s_r)$ contains the positive singular values. Let V_0 be an orthonormal basis for $\mathcal{N}(A)$. Then

$$P_{\mathcal{N}(A)} = V_0 V_0^\top, \qquad A^\dagger = V_r S^{-1} U_r^\top. \tag{52}$$

With $\lambda = \sigma_y^2 / \sigma_t^2$, the pivot can be written as

$$\mu_\star = V_r (S^2 + \lambda I)^{-1} \left(S U_r^\top y + \lambda V_r^\top x_t \right) + V_0 V_0^\top x_t. \tag{53}$$

Taking $\lambda \to 0$ gives

$$\mu_\star \longrightarrow V_r S^{-1} U_r^\top y + V_0 V_0^\top x_t = A^\dagger y + P_{\mathcal{N}(A)} x_t. \tag{54}$$

It remains to show that the corresponding anisotropic denoiser converges to the posterior mean $\mathbb{E}[x_0|y]$. In the same SVD coordinates, the covariance is

$$\Sigma_\star(t) = \left(\frac{1}{\sigma_t^2} I + \frac{1}{\sigma_y^2} A^\top A \right)^{-1} \tag{55}$$

$$= \sigma_y^2 V_r (S^2 + \lambda I)^{-1} V_r^\top + \sigma_t^2 V_0 V_0^\top. \tag{56}$$

Thus, as $\sigma_t \to \infty$, the row-space covariance converges to

$$\sigma_y^2 V_r S^{-2} V_r^\top = \sigma_y^2 (A^\top A)^\dagger, \tag{57}$$

while the nullspace variance diverges. Therefore the limiting denoising query contains finite information only in the row space of A and no information in the nullspace.

More explicitly, the limiting finite-noise observation is

$$z = A^\dagger y = A^\dagger A x_0 + A^\dagger \eta = P_{\mathcal{R}(A^\top)} x_0 + \zeta, \qquad \zeta \sim \mathcal{N}\big(0, \sigma_y^2 (A^\top A)^\dagger\big). \tag{58}$$

Since $A^\dagger$ is a deterministic function of y, conditioning on y implies conditioning on z. Conversely, for Gaussian measurement noise with known covariance, $z = A^\dagger y$ is a sufficient statistic for y with respect to x_0: the remaining component of y orthogonal to $\mathcal{R}(A)$ carries no information about x_0. Hence

$$\mathbb{E}[x_0|z] = \mathbb{E}[x_0|y]. \tag{59}$$

The limiting anisotropic denoiser is exactly the Bayes estimator associated with this row-space observation and infinite nullspace uncertainty. Therefore

$$\lim_{\sigma_t \to \infty} D_{\Sigma_\star(t)}(\mu_\star(x_t, y, t)) = \mathbb{E}[x_0|y]. \tag{60}$$

This proves Observation 4.

A.6 Equivalent-Time and GLASS Limit

When $A^\top A = \gamma^2 I_d$, the covariance in (12) is isotropic:

$$\Sigma_\star(t) = \sigma_\star^2(t) I_d, \qquad \sigma_\star^2(t) = \frac{\beta_t^2 \sigma_y^2}{\alpha_t^2 \sigma_y^2 + \beta_t^2 \gamma^2}. \tag{61}$$

In this special case the posterior denoising query can be represented by an equivalent scalar noise level. If the base denoiser is trained for isotropic corruptions with effective noise ratio β_s^2/α_s^2, we choose $s = t^\star$ such that

$$\frac{\beta_{t^\star}^2}{\alpha_{t^\star}^2} = \sigma_\star^2(t). \tag{62}$$

Then $D_{\Sigma_\star(t)}(\mu_\star)$ can be approximated by the pretrained isotropic denoiser at time $t^\star$. This is the setting in which an equivalent-time, training-free reduction such as GLASS [59] is available. For a general inverse problem, however, $A^\top A$ has different eigenvalues and often a nontrivial nullspace; $\Sigma_\star$ is then anisotropic and no single scalar time can represent the posterior denoising kernel. EPS fine-tuning is introduced precisely to learn this missing anisotropic denoising geometry.

A.7 Connection to ridge regression and Gaussian processes.

Equation (12) is exactly the linear-Gaussian Bayesian update familiar from ridge regression and Gaussian process regression. Here $\sigma_y^{-2} A^\top A$ plays the role of the data precision, $(\alpha_t^2/\beta_t^2) I_d$ plays the role of the prior precision (set by the current diffusion noise level), and $\mu_\star$ is their precision-weighted mean. As σ_t grows the prior precision vanishes and $\mu_\star$ converges to the data-only ridge solution, while as σ_t shrinks $\mu_\star$ collapses onto x_t. The covariance $\Sigma_\star(t)$ is the corresponding posterior covariance, which shrinks along measured directions (large eigenvalues of $A^\top A$) and remains diffuse along weakly observed directions, mirroring the heteroscedastic uncertainty of GP posteriors [68].

B Implementation Details

Architecture. We use the EDM-ADM checkpoint [17] for ImageNet-64×64 (`edm-imagenet-64x64-cond-adm.pkl`, $\sim$296M parameters, class-conditional with 1000-way one-hot embedding), and an EDM-DDPM++ checkpoint we trained from scratch for FFHQ-64×64. EPS extends the first conv from 3 to 6 input channels: the mean pivot $\mu_\star$ (3 channels) concatenated with a task-specific observation tensor (3 channels). The added input channels are zero-initialised so the network reproduces the unconditional pretrained mapping at step zero of fine-tuning. The observation tensor is the masked observation y for inpainting, the nearest-neighbor upsampling of y for super-resolution, and the blurred observation y for deblurring; the total input width is therefore 6 channels for every task.

Pivot solver. For binary masks $\Sigma_\star$ is diagonal and the pivot solve is element-wise. For $4\times$ super-resolution the structured solve uses average-pool / nearest-upsample primitives in $O(d)$. For circular blur kernels we diagonalize $A^\top A$ by the 2D FFT; the per-step solve is then a complex element-wise divide plus an inverse FFT, $O(d \log d)$. Empirically the pivot solve contributes <1 ms per step compared to a U-Net forward of $\sim$19 ms at batch 1.

Optimization. We use the EDM optimizer stack unchanged: Adam (β_1=0.9, β_2=0.999, ε=10^{-8}); log-normal noise sampling with $P_{\text{mean}}=-1.2$, P_{std}=1.2, σ_{data}=0.5; schedule extrema $\sigma_{\min}$=0.002, $\sigma_{\max}$=80, ρ=7. Learning rate is 10^{-4} with a 2×10^6-image linear warm-up. EMA uses a half-life of 500 kimg with a 5% ramp-up ratio. We weight the loss by the standard EDM weighting $\lambda(\sigma) = (\sigma^2+\sigma_{\text{data}}^2)/(\sigma\,\sigma_{\text{data}})^2$. Per-task fine-tuning runs for 10 epochs ($\sim$25k iterations) on ImageNet-64 at batch 128 across 4 NVIDIA B200 GPUs (gradient accumulation=1); FFHQ-64 trains for the same iteration budget at batch 192. End-to-end fine-tuning takes $\sim$24 h on ImageNet and $\sim$10 h on FFHQ per task.

Operator randomization during training. We re-sample the operator at every minibatch step. Random inpainting samples a per-pixel Bernoulli mask with mask-density $\sim \mathcal{U}(50\%, 70\%)$ (default training density 70%). Box inpainting samples a uniformly random rectangle whose side lengths are drawn from $\mathcal{U}([H/4, H/2]) \times \mathcal{U}([W/4, W/2])$ with margins $H/16$ at every edge. Motion-deblur kernels are generated with random length $\in\{7, 9, 11, 13, 15\}$ and random angle $\in[0, 180^\circ]$; Gaussian-deblur kernels use a fixed bandwidth σ_{blur}=0.75 at length 11. Super-resolution applies a fixed $4\times$ average-pool. Observation noise is fixed at σ_y=0.05 throughout both training and evaluation.

Sampling at inference. We use the deterministic EDM Euler ODE sampler for the 20- and 100-NFE variants (`second_order=False`). The 1-NFE variant evaluates the denoiser once at the highest noise level $\sigma_{\max}$ via the high-noise pivot of Observation 4; the resulting single forward pass returns the conditional MMSE estimator $\mathbb{E}[x_0|y]$ directly. Class labels at inference time use the ground-truth ImageNet-1k class for ImageNet-64 and are not used for FFHQ-64.

Reproducibility. All random seeds are fixed; the same 100 evaluation images and 10 posterior seeds per image are used for every method. For code and model checkpoints, see `https://github.com/abbasmammadov/EPS`.

C Broader Impact

EPS provides a principled, calibrated approach to posterior sampling for linear inverse problems. Because it preserves the input/output structure of standard denoising pretraining, an existing pretrained prior can be repurposed into an uncertainty-aware posterior sampler with a lightweight fine-tune rather than a from-scratch retraining. This makes the method straightforward to adapt across scientific imaging applications where reliable reconstructions and quantified posterior uncertainty are valuable, and the closed-form pivot and covariance offer a transparent handle for analyzing the sampler in any downstream pipeline. Beyond these benefits, our work does not introduce societal impacts that go meaningfully beyond those of the existing generative diffusion priors it builds on; the standard considerations around dual-use of high-fidelity image generation and the demographic or domain biases of the underlying training data continue to apply.

D Additional Experiments

We collect ablations and extended tables that support the main claims.

D.1 Input Configuration Ablation

The central input ablation compares raw-state conditioning to shifted-pivot conditioning while keeping the backbone, compute budget, and EDM warm start fixed. We evaluate four input streams to the denoiser: (a) $[x_t, y, t]$ (Palette-style), the standard conditional baseline that feeds the noised latent alongside the observation and exposes no closed-form posterior structure; (b) $[\mu_\star, t]$, which replaces x_t with the closed-form posterior mean $\mu_\star(x_t, y, \sigma_t)$ obtained by Gaussian-merging $p(x_t \mid x_0)$ and $p(y \mid x_0)$ inside the integral, dropping y from the input; (c) $[\mu_\star, \Sigma_\star, t]$, which additionally passes the per-component posterior covariance $\Sigma_\star$ as a side channel, giving the network explicit access to the local anisotropic uncertainty; and (d) $[\mu_\star, y, t]$ (EPS, ours), the full EPS input where the posterior mean is concatenated with the raw observation, allowing the network to use y both directly and through the analytical pivot.

Table 3 reports this ablation on FFHQ-64 (DDPM++/EDM backbone) and ImageNet-64 (EDM-ADM backbone) at NFE=100 with the EDM Euler sampler. Both backbones are warm-started from the same pretrained checkpoint and fine-tuned under matched protocols. Averaged over tasks, EPS is the best of the four inputs on every metric on both datasets, with $[\mu_\star, \Sigma_\star, t]$ a close second. Replacing x_t by $\mu_\star$ explains most of the gain, and the auxiliary observation channel provides an additional anchor.

All five tasks and operator-aware inputs. We extend the ablation to all five tasks and to inputs that pass the operator explicitly: $[\mu_\star, y, A^\top A]$, $[\mu_\star, y, x_t]$, and the InvFusion-style input without a pivot, $[x_t, y, A^\top A]$. Motion deblurring is the most informative case, because its kernel is redrawn at every training step (random length and angle), so the network cannot rely on a fixed operator and must read it from the measurement or the pivot. Each arm starts from the EPS checkpoint of its task and changes only the network input; a control arm that keeps the EPS input is fine-tuned for the same budget, so no arm is credited for training longer. Table 4 shows that every input that keeps $\mu_\star$ lands within 0.06 dB of the control on every task, so $\Sigma_\star$, $A^\top A$, and x_t carry no information that the network cannot already read from the pivot. Removing the pivot is never better, even though this arm also starts from the EPS weights. It costs 0.19, 0.55, and 0.8 dB on super-resolution and the two inpainting tasks, and 3.3 and 5.9 dB on Gaussian and motion deblurring, where the operator is least informative.

Table 3: **The shifted pivot $\mu_\star$ is the right input.** Input-configuration ablation on FFHQ-64 (top) and ImageNet-64 (bottom) at NFE=100 with the EDM Euler sampler. Averaged over tasks, EPS is best on every metric on both datasets; the pivot alone improves PSNR over Palette but not every metric. Best in **bold**, second-best underlined; the EPS row is highlighted in light pink.

Task	Input	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
				FFHQ 64×64					
Average	$[x_t, y, t]$ (Palette)	26.03	0.8590	0.0626	31.50	-6.6e-03	-4.7e-03	3.43	3.86
	$[\mu_\star, t]$	26.39	0.8583	0.0632	31.34	-6.7e-03	-4.6e-03	3.30	3.81
	$[\mu_\star, \Sigma_\star, t]$	26.62	0.8636	0.0603	30.20	-6.7e-03	-4.8e-03	3.27	3.76
	$[\mu_\star, y, t]$ (EPS)	**26.69**	**0.8661**	**0.0590**	**29.94**	**-6.7e-03**	**-4.9e-03**	**3.24**	**3.73**
Random inpaint	$[x_t, y, t]$ (Palette)	25.76	0.8809	0.0593	33.30	-6.7e-03	-4.8e-03	3.31	3.87
	$[\mu_\star, t]$	24.87	0.8564	0.0698	37.92	-6.7e-03	-3.9e-03	3.39	4.08
	$[\mu_\star, \Sigma_\star, t]$	26.00	0.8845	0.0546	32.10	-6.7e-03	-5.0e-03	3.22	3.76
	$[\mu_\star, y, t]$ (EPS)	**26.16**	**0.8879**	**0.0533**	**31.87**	**-6.7e-03**	**-5.0e-03**	**3.16**	**3.75**
Box inpaint	$[x_t, y, t]$ (Palette)	24.18	0.8426	0.0577	25.09	-6.6e-03	-5.3e-03	4.02	3.47
	$[\mu_\star, t]$	24.14	0.8431	0.0575	24.94	**-6.6e-03**	-5.4e-03	4.02	3.45
	$[\mu_\star, \Sigma_\star, t]$	24.17	0.8430	0.0574	24.88	-6.6e-03	**-5.4e-03**	4.03	3.45
	$[\mu_\star, y, t]$ (EPS)	**24.23**	**0.8448**	**0.0567**	**24.74**	-6.6e-03	-5.4e-03	**4.01**	**3.45**
Super-res (4×)	$[x_t, y, t]$ (Palette)	21.95	0.7220	0.1273	**49.28**	-6.3e-03	-3.0e-03	**5.22**	**5.29**
	$[\mu_\star, t]$	21.89	0.7179	0.1300	49.85	-6.3e-03	-2.8e-03	5.29	5.36
	$[\mu_\star, \Sigma_\star, t]$	21.86	0.7162	0.1308	50.14	-6.2e-03	-2.7e-03	5.30	5.39
	$[\mu_\star, y, t]$ (EPS)	**21.96**	**0.7232**	**0.1262**	49.29	**-6.3e-03**	**-3.0e-03**	5.23	5.30
Gaussian deblur	$[x_t, y, t]$ (Palette)	30.47	0.9397	0.0286	21.63	-6.9e-03	-5.6e-03	1.91	3.06
	$[\mu_\star, t]$	30.82	0.9408	**0.0273**	20.72	**-6.9e-03**	-5.7e-03	1.84	2.99
	$[\mu_\star, \Sigma_\star, t]$	30.82	0.9408	0.0273	20.70	-6.9e-03	**-5.7e-03**	1.84	2.99
	$[\mu_\star, y, t]$ (EPS)	**30.82**	**0.9408**	0.0273	**20.68**	-6.9e-03	-5.7e-03	**1.84**	**2.99**
Motion deblur	$[x_t, y, t]$ (Palette)	27.79	0.9099	0.0404	28.23	-6.8e-03	-5.0e-03	2.67	3.59
	$[\mu_\star, t]$	30.23	0.9334	0.0314	23.24	-6.9e-03	-5.4e-03	1.94	3.18
	$[\mu_\star, \Sigma_\star, t]$	30.25	0.9337	0.0312	23.16	-6.9e-03	-5.4e-03	1.94	3.18
	$[\mu_\star, y, t]$ (EPS)	**30.27**	**0.9339**	**0.0311**	**23.10**	**-6.9e-03**	**-5.4e-03**	**1.94**	**3.18**
				ImageNet 64×64					
Average	$[x_t, y, t]$ (Palette)	24.32	0.7673	0.1124	82.57	-6.4e-03	-4.4e-03	4.35	5.54
	$[\mu_\star, t]$	24.36	0.7627	0.1151	83.64	-6.4e-03	-4.3e-03	4.31	5.55
	$[\mu_\star, \Sigma_\star, t]$	24.52	0.7699	0.1115	82.06	-6.4e-03	-4.4e-03	4.29	5.50
	$[\mu_\star, y, t]$ (EPS)	**24.53**	**0.7712**	**0.1103**	**81.46**	**-6.4e-03**	**-4.4e-03**	**4.27**	**5.48**
Random inpaint	$[x_t, y, t]$ (Palette)	24.09	0.7869	0.1011	81.88	-6.5e-03	-4.4e-03	4.16	5.52
	$[\mu_\star, t]$	23.53	0.7588	0.1173	88.24	-6.5e-03	-4.2e-03	4.18	5.66
	$[\mu_\star, \Sigma_\star, t]$	**24.35**	0.7944	0.0986	80.19	-6.5e-03	-4.4e-03	4.05	5.43
	$[\mu_\star, y, t]$ (EPS)	24.34	**0.7948**	**0.0979**	**79.60**	**-6.5e-03**	**-4.5e-03**	**4.04**	**5.41**
Box inpaint	$[x_t, y, t]$ (Palette)	21.12	0.7541	0.1218	92.73	-6.1e-03	-4.1e-03	5.92	5.93
	$[\mu_\star, t]$	21.19	0.7552	0.1209	91.74	-6.1e-03	-4.2e-03	5.88	5.86
	$[\mu_\star, \Sigma_\star, t]$	21.18	0.7544	0.1212	**91.03**	-6.1e-03	**-4.2e-03**	5.92	**5.82**
	$[\mu_\star, y, t]$ (EPS)	**21.24**	**0.7569**	**0.1196**	91.07	**-6.1e-03**	-4.2e-03	**5.87**	5.84
Super-res (4×)	$[x_t, y, t]$ (Palette)	20.24	0.5364	0.2220	**128.76**	**-5.9e-03**	-2.8e-03	**6.50**	**7.33**
	$[\mu_\star, t]$	20.20	0.5308	0.2246	130.53	-5.8e-03	-2.6e-03	6.55	7.39
	$[\mu_\star, \Sigma_\star, t]$	20.20	0.5319	0.2250	131.17	-5.8e-03	-2.7e-03	6.54	7.40
	$[\mu_\star, y, t]$ (EPS)	**20.25**	**0.5369**	**0.2207**	128.80	-5.9e-03	**-2.8e-03**	6.52	7.35
Gaussian deblur	$[x_t, y, t]$ (Palette)	29.15	0.9010	0.0491	46.62	**-6.8e-03**	**-5.6e-03**	2.26	**4.11**
	$[\mu_\star, t]$	29.18	0.9014	**0.0486**	46.76	-6.8e-03	-5.5e-03	2.25	4.12
	$[\mu_\star, \Sigma_\star, t]$	29.18	0.9014	0.0487	46.73	-6.8e-03	-5.5e-03	2.25	4.13
	$[\mu_\star, y, t]$ (EPS)	**29.18**	**0.9015**	0.0486	**46.55**	-6.8e-03	-5.6e-03	**2.25**	4.11
Motion deblur	$[x_t, y, t]$ (Palette)	27.02	0.8582	0.0680	62.86	-6.7e-03	-5.0e-03	2.93	4.81
	$[\mu_\star, t]$	**27.68**	**0.8674**	**0.0641**	**60.94**	**-6.8e-03**	**-5.1e-03**	**2.68**	4.70
	$[\mu_\star, \Sigma_\star, t]$	27.67	0.8673	0.0642	61.19	-6.8e-03	-5.1e-03	2.68	**4.70**
	$[\mu_\star, y, t]$ (EPS)	27.62	0.8661	0.0647	61.29	-6.8e-03	-5.1e-03	2.69	4.70

Table 4: **Input ablation on all five tasks** (FFHQ-64, 100 NFE, 100 images $\times$ 10 samples). Each arm starts from the EPS checkpoint of its task and is fine-tuned for the same budget (batch 32, 5000 steps) with only the network input changed; the $[\mu_\star, y]$ arm keeps the EPS input and is the control. Cells are per-sample PSNR↑ / LPIPS↓ / FID↓ (standard errors over images are 0.14 to 0.24 dB for every PSNR entry). Every input that keeps $\mu_\star$ is within 0.06 dB of the control, far inside the standard errors, so $\Sigma_\star$, $A^\top A$ and x_t add nothing once the pivot is present. Dropping the pivot (the InvFusion-style input) is never better and costs most where the operator is least informative, up to 5.9 dB on motion deblurring.

Task	$[\mu_\star, y]$ (control)	$[\mu_\star, y, \Sigma_\star]$	$[\mu_\star, y, A^\top A]$	$[\mu_\star, y, x_t]$	$[x_t, y, A^\top A]$ (no pivot)
SR 4×	21.98 / 0.125 / 47.5	21.97 / 0.126 / 47.4	21.97 / 0.125 / 47.4	21.98 / 0.126 / 47.4	21.79 / 0.131 / 47.6
Inpaint (box)	23.95 / 0.059 / 23.1	24.00 / 0.059 / 23.2	23.97 / 0.059 / 23.2	23.94 / 0.059 / 23.2	23.40 / 0.074 / 28.5
Inpaint (random)	26.17 / 0.053 / 30.0	26.19 / 0.053 / 30.1	26.22 / 0.053 / 30.1	26.18 / 0.053 / 30.1	25.36 / 0.067 / 32.8
Gaussian deblur	30.87 / 0.027 / 18.6	30.87 / 0.027 / 18.6	30.87 / 0.027 / 18.7	30.88 / 0.027 / 18.6	27.61 / 0.061 / 33.0
Motion deblur	30.33 / 0.031 / 21.1	30.30 / 0.031 / 21.1	30.30 / 0.031 / 21.1	30.30 / 0.031 / 21.1	24.40 / 0.078 / 38.2

D.2 Zero-Shot Pivoting

Before fine-tuning, we can feed $\mu_\star$ directly to the pretrained denoiser (with the closest available EDM noise level, or the exact equivalent time in isotropic cases). This is not exact for general A because the pretrained model has not learned anisotropic denoising, but it tests whether the pivot already carries useful posterior information.

Table 5 compares zero-shot pivoting to fine-tuned EPS and to Palette under the same 100-step Euler sampler on FFHQ-64 and ImageNet-64. Zero-shot pivoting feeds $\mu_\star$ directly to the pretrained EDM denoiser at the current σ_t (no fine-tuning) and runs the standard 100-step Euler loop; fine-tuned EPS uses the same sampler but with the denoiser adapted to take $[\mu_\star, y]$ on each task. Zero-shot pivoting underperforms both Palette and fine-tuned EPS on every task, confirming that the pretrained denoiser does not natively handle the anisotropic geometry of $\Sigma_\star(t)$, and that the EPS fine-tuning step is what unlocks the benefit of the pivot.

Table 5: **The pivot needs the fine-tuning step.** Zero-shot pivoting vs. fine-tuned EPS at NFE=100 (Euler sampler) on FFHQ-64 (top) and ImageNet-64 (bottom). Feeding $\mu_\star$ to the pretrained denoiser without adaptation underperforms fine-tuned EPS on every task and metric, and Palette everywhere except PSNR and SSIM on FFHQ-64 motion deblurring. Best in **bold**, second-best underlined; the EPS row is highlighted in light pink.

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
					FFHQ 64×64				
Average	Palette	26.03	0.8590	0.0626	31.50	-6.65e-03	-4.74e-03	3.43	3.86
	Zero-shot EPS pivot	22.97	0.7386	0.1503	70.88	-3.14e-03	3.56e-02	5.61	6.68
	EPS fine-tuned	**26.69**	**0.8661**	**0.0590**	**29.94**	**-6.69e-03**	**-4.89e-03**	**3.24**	**3.73**
Random inpaint	Palette	25.76	0.8809	0.0593	33.30	-6.71e-03	-4.76e-03	3.31	3.87
	Zero-shot EPS pivot	16.36	0.5302	0.2985	121.98	6.58e-03	1.02e-01	9.08	8.66
	EPS fine-tuned	**26.16**	**0.8879**	**0.0533**	**31.87**	**-6.74e-03**	**-5.00e-03**	**3.16**	**3.75**
Box inpaint	Palette	24.18	0.8426	0.0577	25.09	-6.58e-03	-5.30e-03	4.02	3.47
	Zero-shot EPS pivot	18.75	0.7065	0.1699	84.24	-5.19e-03	4.32e-02	6.71	6.69
	EPS fine-tuned	**24.23**	**0.8448**	**0.0567**	**24.74**	**-6.59e-03**	**-5.35e-03**	**4.01**	**3.45**
Super-res (4×)	Palette	21.95	0.7220	0.1273	**49.28**	-6.29e-03	-2.98e-03	**5.22**	**5.29**
	Zero-shot EPS pivot	20.50	0.6234	0.1713	57.45	-3.68e-03	4.23e-03	6.62	6.20
	EPS fine-tuned	**21.96**	**0.7232**	**0.1262**	49.29	**-6.30e-03**	**-3.00e-03**	5.23	5.30
Gaussian deblur	Palette	30.47	0.9397	0.0286	21.63	-6.90e-03	-5.65e-03	1.91	3.06
	Zero-shot EPS pivot	29.79	0.9197	0.0540	43.77	-6.73e-03	1.31e-02	2.76	5.79
	EPS fine-tuned	**30.82**	**0.9408**	**0.0273**	**20.68**	**-6.91e-03**	**-5.73e-03**	**1.84**	**2.99**
Motion deblur	Palette	27.79	0.9099	0.0404	28.23	-6.77e-03	-4.98e-03	2.67	3.59
	Zero-shot EPS pivot	29.42	0.9135	0.0579	46.95	-6.69e-03	1.54e-02	2.89	6.05
	EPS fine-tuned	**30.27**	**0.9339**	**0.0311**	**23.10**	**-6.90e-03**	**-5.39e-03**	**1.94**	**3.18**
					ImageNet 64×64				
Average	Palette	24.32	0.7673	0.1124	82.57	-6.40e-03	-4.38e-03	4.35	5.54
	Zero-shot EPS pivot	22.02	0.6398	0.2461	146.88	2.48e-03	1.06e-02	6.96	8.89
	EPS fine-tuned	**24.53**	**0.7712**	**0.1103**	**81.46**	**-6.42e-03**	**-4.45e-03**	**4.27**	**5.48**
Random inpaint	Palette	24.09	0.7869	0.1011	81.88	-6.50e-03	-4.41e-03	4.16	5.52
	Zero-shot EPS pivot	16.92	0.4168	0.4350	223.68	2.61e-02	4.15e-02	10.14	11.23
	EPS fine-tuned	**24.34**	**0.7948**	**0.0979**	**79.60**	**-6.52e-03**	**-4.51e-03**	**4.04**	**5.41**
Box inpaint	Palette	21.12	0.7541	0.1218	92.73	-6.10e-03	-4.12e-03	5.92	5.93
	Zero-shot EPS pivot	18.70	0.6442	0.2333	139.38	-1.38e-03	2.45e-03	8.56	8.10
	EPS fine-tuned	**21.24**	**0.7569**	**0.1196**	**91.07**	**-6.11e-03**	**-4.23e-03**	**5.87**	**5.84**
Super-res (4×)	Palette	20.24	0.5364	0.2220	**128.76**	**-5.86e-03**	-2.79e-03	**6.50**	**7.33**
	Zero-shot EPS pivot	19.85	0.4663	0.3035	160.80	-9.25e-04	6.29e-03	7.91	8.72
	EPS fine-tuned	**20.25**	**0.5369**	**0.2207**	128.80	-5.86e-03	**-2.84e-03**	6.52	7.35
Gaussian deblur	Palette	29.15	0.9010	0.0491	46.62	**-6.83e-03**	**-5.56e-03**	2.26	**4.11**
	Zero-shot EPS pivot	27.83	0.8507	0.1211	97.87	-5.85e-03	7.94e-04	3.87	7.97
	EPS fine-tuned	**29.18**	**0.9015**	**0.0486**	**46.55**	-6.83e-03	-5.55e-03	**2.25**	4.11
Motion deblur	Palette	27.02	0.8582	0.0680	62.86	-6.73e-03	-5.02e-03	2.93	4.81
	Zero-shot EPS pivot	26.82	0.8212	0.1374	112.64	-5.59e-03	2.19e-03	4.31	8.42
	EPS fine-tuned	**27.62**	**0.8661**	**0.0647**	**61.29**	**-6.77e-03**	**-5.11e-03**	**2.69**	**4.70**

D.3 Palette vs EPS

Figure 4 reports fine-tuning convergence curves for EPS and Palette warm-started from the same pretrained EDM checkpoint. Rows index five (dataset, task) pairs (ImageNet-64 random inpainting, ImageNet-64 motion deblurring, FFHQ-64 random inpainting, FFHQ-64 motion deblurring, FFHQ-64 Gaussian deblurring); columns track training loss, PSNR, SSIM, LPIPS, and FID against the number of fine-tuning iterations under the matched 100-step Euler sampler. Three patterns hold across all rows. First, EPS starts from a much better initialization on every metric: at iteration zero, the shifted pivot $\mu_\star$ already encodes enough of the measurement geometry that PSNR and SSIM are within a few units of their converged values, while Palette has to climb from the unconditional prior. Second, EPS reaches its asymptotic LPIPS and FID in a small fraction of the iterations Palette needs and stays at least as low for the rest of training. Third, the gap is largest on the deblurring tasks, where the pivot provides the strongest measurement signal at initialization, and on the perceptual and distributional metrics (LPIPS, FID), where the conditioning structure matters most. This is consistent with the structural-locality argument: EPS preserves the input/output type and Gaussian-corrupted-target geometry of the pretrained denoising task, and only adapts to the operator-induced anisotropic covariance $\Sigma_\star(t)$.

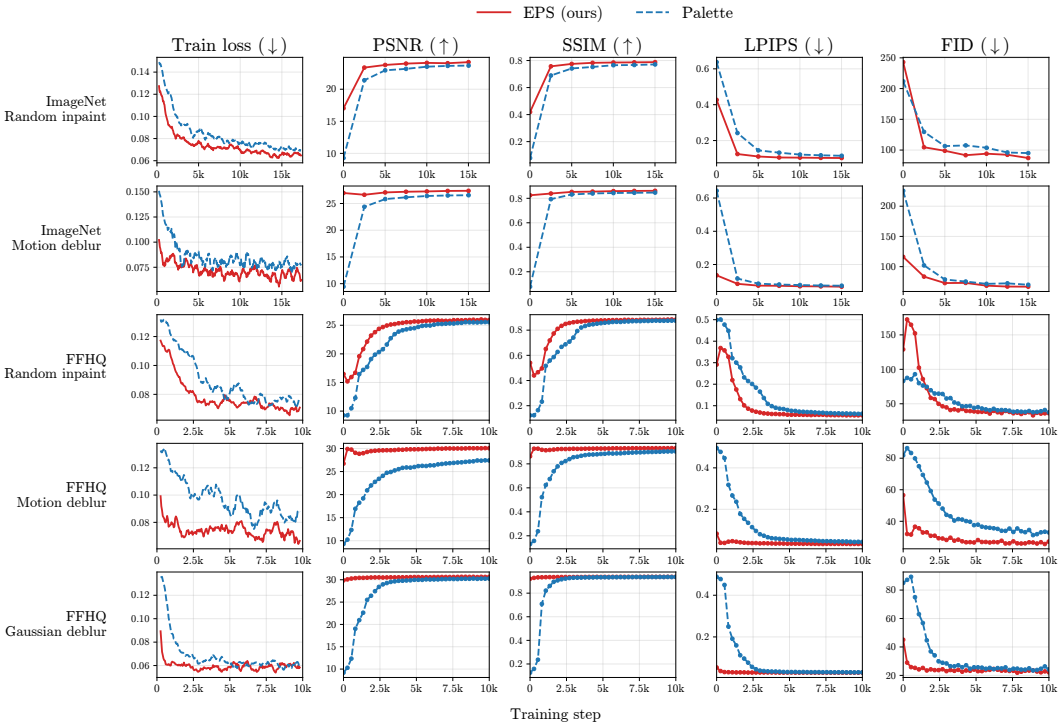

Figure 4: **EPS converges faster than Palette from the same warm start.** Fine-tuning curves for EPS and Palette, both initialized from the same pretrained EDM checkpoint, on five (dataset, task) pairs (rows) and five metrics (columns: training loss, PSNR, SSIM, LPIPS, FID) under the matched 100-step Euler sampler. EPS starts from a markedly better initialization on every metric and reaches its asymptote in a small fraction of the iterations Palette needs.

D.4 Sampling Efficiency

We sweep NFE at inference from 5 to 100 across all five tasks, with all methods using a 1-NFE-per-step Euler sampler under matched conditions.

Figures 5 and 6 report PSNR, FID, and Inception-feature CRPS as a function of sampler iterations on FFHQ-64 and ImageNet-64 respectively. Rows index the five tasks (random inpaint, box inpaint, $4\times$ super-resolution, Gaussian deblur, motion deblur); columns track the three reported metrics. EPS reaches its asymptotic FID and CRPS within roughly 15-20 steps on every task and stays flat thereafter, while sampling-based baselines either fail to reach the same level (DPS, MPGD) or are slower to converge (DDNM, ΠGDM). The flat right tail of the EPS curves is the practical justification for the 20-NFE setting reported in the main results (Tables 1 and 8), and the gap to the strongest sampling-based baseline (ΠGDM) widens on the deblurring tasks under the more diverse ImageNet distribution.

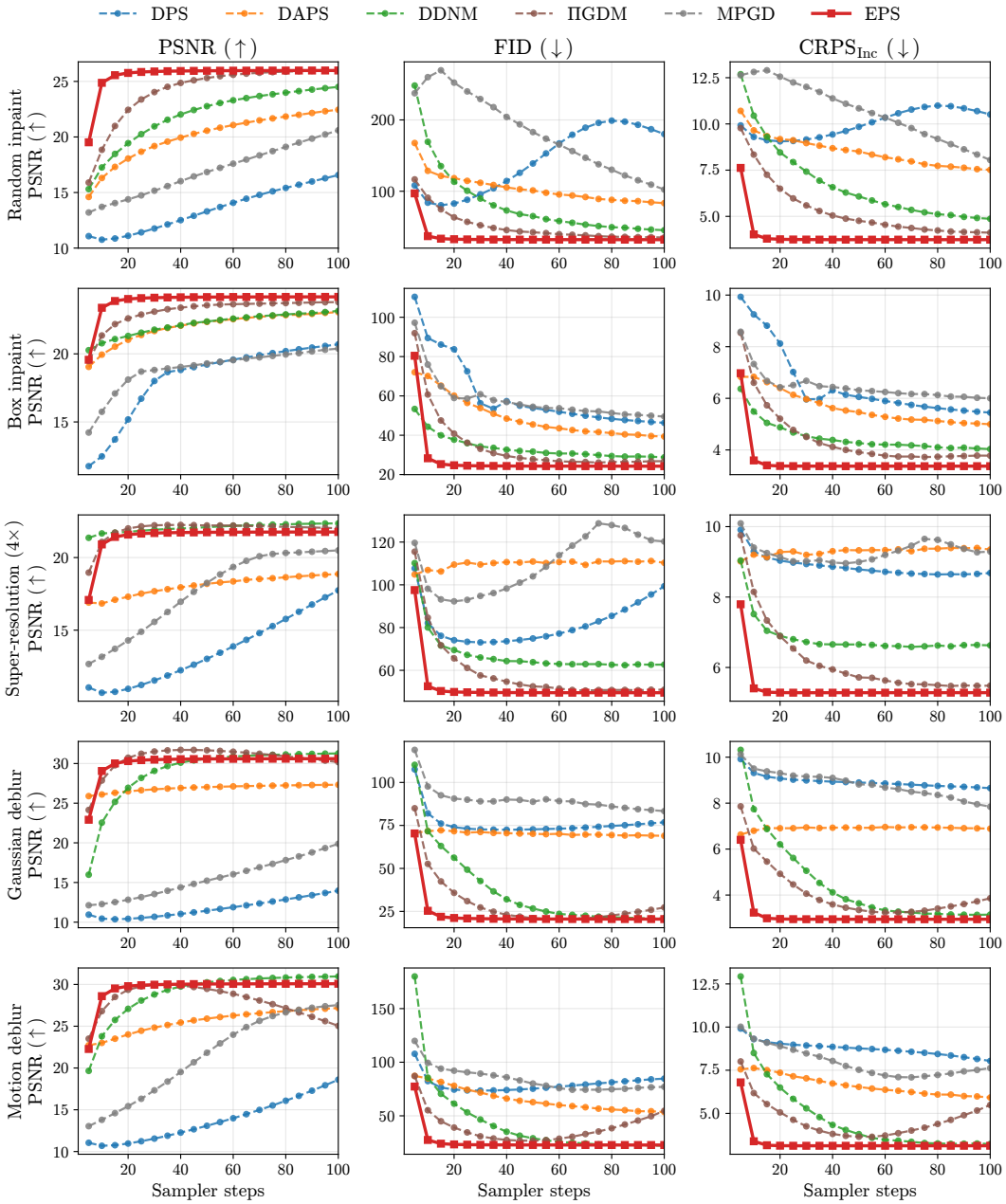

Figure 5: **EPS plateaus by 15-20 steps on FFHQ-64.** Sampling-step sensitivity on FFHQ-64 across the five tasks (rows). Columns report PSNR (↑), FID (↓), and Inception-feature CRPS (↓) versus sampler iterations under a 1-NFE-per-step Euler sampler. EPS reaches its asymptote within roughly 15-20 steps on every task and remains best or tied-best on FID and CRPS thereafter.

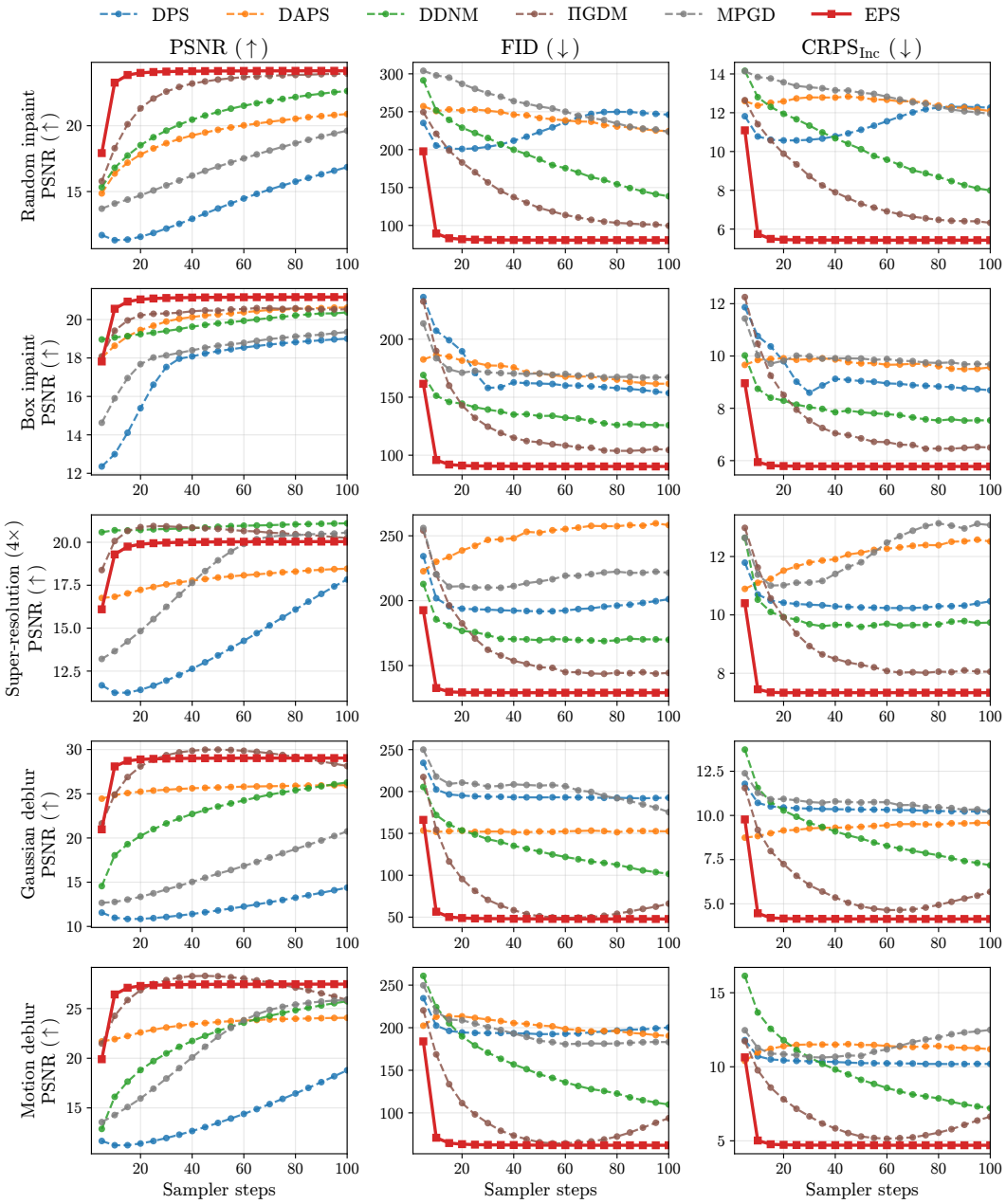

Figure 6: **The plateau transfers to ImageNet-64.** Sampling-step sensitivity on ImageNet-64; same layout as Fig. 5. EPS plateaus at the same step count on a more diverse class-conditional distribution, and the gap between EPS and the strongest sampling-based baseline (ΠGDM) widens on the deblurring tasks.

D.5 Amortized Variant Across All Five Tasks

As a deployment-friendly alternative to per-task fine-tuning, we train a single EPS checkpoint across all five tasks using uniformly sampled operators per training step and no task indicator at the input.

Table 6 compares per-task EPS, amortized EPS, and Palette under matched compute on FFHQ-64 (top) and ImageNet-64 (bottom). On FFHQ-64 we report two amortized snapshots, at 55k and 160k training steps; the 160k checkpoint surpasses per-task EPS on every task and every metric, indicating that a single network can absorb all five operators without loss. On ImageNet-64, the amortized model is within 0.2 dB PSNR of per-task EPS and, on average over tasks, matches or beats Palette on the distributional metrics, while requiring $5\times$ less storage and a single set of weights at deployment.

Table 6: **Amortization works.** Amortized EPS vs. per-task EPS on FFHQ-64 (top) and ImageNet-64 (bottom) at NFE$=$100 with the EDM Euler sampler. The FFHQ amortized model is reported at two training-step snapshots (55k, 160k); the 160k snapshot surpasses per-task EPS across every task and metric. The ImageNet amortized model (single 296M-param ADM checkpoint, $\sim$60 epochs) is within 0.2 dB PSNR of per-task EPS and, on average over tasks, matches or beats Palette on the distributional metrics. Best in **bold**, second-best underlined; amortized rows are highlighted in light pink.

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
				FFHQ 64×64					
Average	Palette	26.03	0.8590	0.0626	31.50	-6.65e-03	-4.74e-03	3.43	3.86
	Per-task EPS	26.69	0.8661	0.0590	29.94	-6.69e-03	-4.89e-03	3.24	3.73
	Amortized EPS (55k steps)	26.95	0.8745	0.0552	27.86	-6.71e-03	-5.20e-03	3.10	3.51
	Amortized EPS (160k steps)	**27.31**	**0.8830**	**0.0513**	**26.16**	**-6.74e-03**	**-5.25e-03**	**2.98**	**3.38**
Random inpaint	Palette	25.76	0.8809	0.0593	33.30	-6.71e-03	-4.76e-03	3.31	3.87
	Per-task EPS	26.16	0.8879	0.0533	31.87	-6.74e-03	-5.00e-03	3.16	3.75
	Amortized EPS (55k steps)	26.27	0.8920	0.0519	29.76	-6.74e-03	-5.20e-03	3.10	3.56
	Amortized EPS (160k steps)	**26.73**	**0.9001**	**0.0478**	**27.57**	**-6.77e-03**	**-5.34e-03**	**2.96**	**3.40**
Box inpaint	Palette	24.18	0.8426	0.0577	25.09	-6.58e-03	-5.30e-03	4.02	3.47
	Per-task EPS	24.23	0.8448	0.0567	24.74	-6.59e-03	-5.35e-03	4.01	3.45
	Amortized EPS (55k steps)	24.75	0.8580	0.0517	22.25	-6.64e-03	-5.60e-03	3.74	3.18
	Amortized EPS (160k steps)	**25.14**	**0.8663**	**0.0486**	**21.07**	**-6.67e-03**	**-5.68e-03**	**3.61**	**3.07**
Super-res (4×)	Palette	21.95	0.7220	0.1273	49.28	-6.29e-03	-2.98e-03	5.22	5.29
	Per-task EPS	21.96	0.7232	0.1262	49.29	-6.30e-03	-3.00e-03	5.23	5.30
	Amortized EPS (55k steps)	22.33	0.7433	0.1158	45.53	-6.36e-03	**-3.80e-03**	4.96	4.89
	Amortized EPS (160k steps)	**22.74**	**0.7633**	**0.1072**	**43.42**	**-6.42e-03**	-3.66e-03	**4.74**	**4.74**
Gaussian deblur	Palette	30.47	0.9397	0.0286	21.63	-6.90e-03	-5.65e-03	1.91	3.06
	Per-task EPS	30.82	0.9408	0.0273	20.68	**-6.91e-03**	-5.73e-03	1.84	2.99
	Amortized EPS (55k steps)	31.01	0.9433	0.0260	19.54	**-6.92e-03**	-5.82e-03	1.79	2.87
	Amortized EPS (160k steps)	**31.19**	**0.9451**	**0.0250**	**18.57**	**-6.92e-03**	**-5.87e-03**	**1.76**	**2.80**
Motion deblur	Palette	27.79	0.9099	0.0404	28.23	-6.77e-03	-4.98e-03	2.67	3.59
	Per-task EPS	30.27	0.9339	0.0311	23.10	-6.90e-03	-5.39e-03	1.94	3.18
	Amortized EPS (55k steps)	30.37	0.9358	0.0307	22.21	**-6.91e-03**	-5.59e-03	1.90	3.03
	Amortized EPS (160k steps)	**30.74**	**0.9402**	**0.0281**	**20.17**	**-6.91e-03**	**-5.71e-03**	**1.83**	**2.88**
				ImageNet 64×64					
Average	Palette	24.32	0.7673	0.1124	82.57	-6.40e-03	-4.38e-03	4.35	5.54
	Per-task EPS	**24.53**	**0.7712**	**0.1103**	**81.46**	**-6.42e-03**	**-4.45e-03**	**4.27**	**5.48**
	Amortized EPS	24.41	0.7677	0.1116	81.82	-6.41e-03	-4.39e-03	4.29	5.50
Random inpaint	Palette	24.09	0.7869	0.1011	81.88	-6.50e-03	-4.41e-03	4.16	5.52
	Per-task EPS	**24.34**	**0.7948**	**0.0979**	**79.60**	**-6.52e-03**	**-4.51e-03**	**4.04**	**5.41**
	Amortized EPS	24.14	0.7890	0.1007	81.27	**-6.52e-03**	-4.42e-03	4.08	5.48
Box inpaint	Palette	21.12	0.7541	0.1218	92.73	-6.10e-03	-4.12e-03	5.92	5.93
	Per-task EPS	**21.24**	**0.7569**	**0.1196**	91.07	**-6.11e-03**	-4.23e-03	**5.87**	5.84
	Amortized EPS	21.12	0.7549	0.1199	**89.59**	-6.10e-03	**-4.34e-03**	5.88	**5.81**
Super-res (4×)	Palette	20.24	0.5364	0.2220	**128.76**	**-5.86e-03**	-2.79e-03	**6.50**	**7.33**
	Per-task EPS	**20.25**	**0.5369**	**0.2207**	128.80	**-5.86e-03**	**-2.84e-03**	6.52	7.35
	Amortized EPS	20.15	0.5308	0.2238	130.55	-5.85e-03	-2.54e-03	6.56	7.38
Gaussian deblur	Palette	29.15	0.9010	0.0491	46.62	**-6.83e-03**	-5.56e-03	2.26	4.11
	Per-task EPS	29.18	0.9015	0.0486	46.55	**-6.83e-03**	-5.55e-03	2.25	4.11
	Amortized EPS	**29.19**	**0.9017**	**0.0478**	**45.33**	**-6.83e-03**	**-5.61e-03**	**2.24**	**4.06**
Motion deblur	Palette	27.02	0.8582	0.0680	62.86	-6.73e-03	-5.02e-03	2.93	4.81
	Per-task EPS	**27.62**	**0.8661**	**0.0647**	**61.29**	**-6.77e-03**	**-5.11e-03**	**2.69**	**4.70**
	Amortized EPS	27.44	0.8619	0.0657	62.35	-6.76e-03	-5.07e-03	2.72	4.74

D.6 One-Step Posterior Mean Check

Section 3.5 predicts that a single high-noise evaluation returns a posterior-mean estimator. We compare EPS at 1 NFE (a single direct Tweedie call $D_\theta(\mu_\star, \sigma_{\max})$ at $\sigma_{\max}$=80, no sampler loop) to the empirical mean of J=10 multi-step posterior samples drawn with the standard 100-step Euler sampler (1000 NFE per image), and to a single posterior sample from the same multi-step sampler (the EPS NFE=100 row used in our main results).

Table 7 reports this comparison on FFHQ-64 (top) and ImageNet-64 (bottom) across all five tasks. The 1-NFE Tweedie call recovers most of the PSNR/SSIM gain that the multi-step empirical mean attains while using $1000\times$ fewer denoiser evaluations, confirming the high-noise posterior-mean prediction. As expected, both rows that target the conditional mean (the 1-NFE row and the empirical-mean row) score better on distortion (PSNR, SSIM) than the single-sample row, while the single-sample row scores better on perceptual and distributional metrics (LPIPS, FID, CRPS, MMD). Note that MMD/CRPS for the empirical-mean row degenerate to deterministic distances since the per-image ensemble has size J=1 after averaging.

Table 7: **One Tweedie call recovers most of the multi-step gain on inpainting and super-resolution.** One-step EPS vs. empirical posterior mean from multi-step EPS samples on FFHQ-64 (top) and ImageNet-64 (bottom). The 1-NFE row is a single direct Tweedie call at $\sigma_{\max}$=80 (no sampler loop). The posterior-mean row averages 10 independent 100-step Euler samples (1000 NFE per image). The single-sample row reports per-seed metrics from the same 100-step sampler (matching the main-text NFE=100 row). On inpainting and super-resolution the 1-NFE row is within 0.8 dB of the empirical-mean row on PSNR while using $1000\times$ fewer denoiser calls; on deblurring the gap is larger. Best in **bold**, second-best underlined.

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
				FFHQ 64×64					
Average	EPS, 1 NFE	27.34	0.8887	0.0699	46.30	-6.31e-03	1.13e-02	4.67	6.91
	EPS, posterior mean from samples	**29.10**	**0.9095**	**0.0480**	34.17	**-1.22e-02**	-2.77e-03	4.35	6.46
	EPS, single posterior sample	26.69	0.8661	0.0590	**29.94**	-6.69e-03	**-4.89e-03**	**3.24**	**3.73**
Random inpaint	EPS, 1 NFE	27.90	0.9148	0.0549	39.71	-6.63e-03	2.18e-03	3.94	5.75
	EPS, posterior mean from samples	**28.66**	**0.9293**	**0.0399**	34.17	**-1.23e-02**	**-5.15e-03**	4.26	6.48
	EPS, single posterior sample	26.16	0.8879	0.0533	**31.87**	-6.74e-03	-5.00e-03	**3.16**	**3.75**
Box inpaint	EPS, 1 NFE	26.01	0.8706	0.0644	32.90	-6.36e-03	-8.49e-04	5.05	5.62
	EPS, posterior mean from samples	**26.72**	**0.8906**	**0.0464**	26.75	**-1.20e-02**	**-8.08e-03**	5.38	5.93
	EPS, single posterior sample	24.23	0.8448	0.0567	**24.74**	-6.59e-03	-5.35e-03	**4.01**	**3.45**
Super-res (4×)	EPS, 1 NFE	24.21	0.7999	0.1265	77.97	-5.63e-03	4.09e-02	6.57	9.27
	EPS, posterior mean from samples	**24.21**	**0.8042**	**0.1087**	64.71	**-1.14e-02**	1.35e-02	7.03	9.36
	EPS, single posterior sample	21.96	0.7232	0.1262	**49.29**	-6.30e-03	**-3.00e-03**	**5.23**	**5.30**
Gaussian deblur	EPS, 1 NFE	30.86	0.9503	0.0369	31.39	-6.62e-03	2.35e-03	3.21	6.15
	EPS, posterior mean from samples	**33.18**	**0.9637**	**0.0211**	21.15	**-1.26e-02**	**-7.75e-03**	2.48	5.05
	EPS, single posterior sample	30.82	0.9408	0.0273	**20.68**	-6.91e-03	-5.73e-03	**1.84**	**2.99**
Motion deblur	EPS, 1 NFE	27.73	0.9078	0.0669	49.55	-6.31e-03	1.21e-02	4.60	7.76
	EPS, posterior mean from samples	**32.71**	**0.9597**	**0.0239**	24.06	**-1.25e-02**	**-6.39e-03**	2.61	5.46
	EPS, single posterior sample	30.27	0.9339	0.0311	**23.10**	-6.90e-03	-5.39e-03	**1.94**	**3.18**
				ImageNet 64×64					
Average	EPS, 1 NFE	25.67	0.8165	0.1318	110.95	-5.58e-03	4.49e-03	5.91	9.96
	EPS, posterior mean from samples	**26.97**	**0.8330**	**0.0999**	90.93	**-1.16e-02**	**-5.77e-03**	5.74	9.76
	EPS, single posterior sample	24.53	0.7712	0.1103	**81.46**	-6.42e-03	-4.45e-03	**4.27**	**5.48**
Random inpaint	EPS, 1 NFE	26.60	0.8580	0.0933	88.59	-6.15e-03	-4.84e-04	4.98	8.32
	EPS, posterior mean from samples	**26.81**	**0.8651**	**0.0790**	81.97	**-1.19e-02**	**-7.41e-03**	5.44	9.35
	EPS, single posterior sample	24.34	0.7948	0.0979	**79.60**	-6.52e-03	-4.51e-03	**4.04**	**5.41**
Box inpaint	EPS, 1 NFE	23.60	0.7908	0.1514	129.11	-5.60e-03	7.34e-03	7.16	10.38
	EPS, posterior mean from samples	**23.65**	**0.7985**	**0.1149**	111.46	**-1.10e-02**	-3.94e-03	7.86	11.12
	EPS, single posterior sample	21.24	0.7569	0.1196	**91.07**	-6.11e-03	**-4.23e-03**	**5.87**	**5.84**
Super-res (4×)	EPS, 1 NFE	**22.78**	**0.6530**	0.2455	182.92	-4.97e-03	1.98e-02	8.06	13.47
	EPS, posterior mean from samples	22.57	0.6447	**0.2078**	158.62	**-1.03e-02**	1.40e-03	8.75	13.68
	EPS, single posterior sample	20.25	0.5369	0.2207	**128.80**	-5.86e-03	**-2.84e-03**	**6.52**	**7.35**
Gaussian deblur	EPS, 1 NFE	28.82	0.9194	0.0606	56.53	-5.14e-03	-3.46e-03	4.09	7.73
	EPS, posterior mean from samples	**31.66**	**0.9399**	**0.0404**	**41.79**	**-1.24e-02**	**-1.02e-02**	3.03	6.63
	EPS, single posterior sample	29.18	0.9015	0.0486	46.55	-6.83e-03	-5.55e-03	**2.25**	**4.11**
Motion deblur	EPS, 1 NFE	26.56	0.8613	0.1079	97.59	-6.03e-03	-7.30e-04	5.25	9.91
	EPS, posterior mean from samples	**30.17**	**0.9167**	**0.0573**	**60.82**	**-1.23e-02**	**-8.67e-03**	3.63	8.03
	EPS, single posterior sample	27.62	0.8661	0.0647	61.29	-6.77e-03	-5.11e-03	**2.69**	**4.70**

D.7 Additional 64×64 Results

For completeness, Table 8 reproduces the FFHQ-64 main-table comparison from the body of the paper, broken out by task with all metrics. EPS at NFE=20 is the strongest configuration on perceptual and distributional metrics across all five tasks, while the NFE=1 Tweedie variant trades distributional fidelity for distortion (PSNR, SSIM), as predicted by the high-noise posterior-mean limit of Section 3.5.

Table 8: **Detailed FFHQ-64 results.** Quantitative comparison across the five inverse problems on FFHQ 64×64. All methods use a 1-NFE-per-step Euler sampler; reported NFE equals the number of sampler iterations. Best in **bold**, second-best underlined; EPS rows highlighted in light pink. † The NFE=1 row evaluates the deterministic high-noise posterior-mean limit (one direct Tweedie call $D_\theta(\mu_\star, \sigma_{\max})$); MMSE-optimal in pixel space but does not produce posterior samples, hence its strong PSNR/SSIM but weaker distributional metrics. Error bars for EPS and the sampling-based methods at 100 NFE, together with the OPC and InvFusion-style baselines, are in Table 14.

Task	Method	NFE	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
Random inpaint	DPS	250	23.09	0.8003	0.1080	70.27	-6.04e-03	3.00e-02	4.56	6.33
	DAPS	100	22.45	0.7576	0.1519	83.31	-4.74e-03	4.95e-02	5.47	7.52
	DDNM	100	24.50	0.8566	0.0766	45.40	-6.44e-03	4.88e-04	4.03	4.87
	ΠGDM	100	26.00	0.8898	0.0556	35.17	-6.69e-03	-3.83e-03	3.44	4.12
	MPGD	100	20.60	0.7097	0.1755	102.37	-1.67e-03	7.62e-02	6.29	8.05
	Palette	100	25.76	0.8809	0.0593	33.30	-6.71e-03	-4.76e-03	3.31	3.87
	EPS (ours)	100	26.16	0.8879	0.0533	31.87	**-6.74e-03**	**-5.00e-03**	3.16	**3.75**
	EPS (ours)	20	26.75	0.9006	**0.0489**	**31.56**	**-6.74e-03**	-3.94e-03	**3.15**	3.89
	EPS (ours)†	1	**27.90**	**0.9148**	0.0549	39.71	-6.63e-03	2.18e-03	3.94	5.75
Box inpaint	DPS	250	22.33	0.7692	0.0946	41.58	-6.30e-03	5.29e-03	5.09	5.08
	DAPS	100	23.06	0.7968	0.0900	39.35	-6.01e-03	6.47e-03	5.09	5.00
	DDNM	100	23.15	0.8231	0.0737	28.75	-6.20e-03	-4.28e-03	4.82	4.03
	ΠGDM	100	23.80	0.8359	0.0623	26.81	-6.51e-03	-4.33e-03	4.36	3.78
	MPGD	100	20.39	0.7211	0.1229	49.51	-4.55e-03	9.16e-03	6.98	6.00
	Palette	100	24.18	0.8426	0.0577	25.09	-6.58e-03	-5.30e-03	4.02	3.47
	EPS (ours)	100	24.23	0.8448	0.0567	24.74	**-6.59e-03**	**-5.35e-03**	4.01	**3.45**
	EPS (ours)	20	24.70	0.8560	**0.0536**	**24.45**	**-6.59e-03**	-4.77e-03	**3.99**	3.55
	EPS (ours)†	1	**26.01**	**0.8706**	0.0644	32.90	-6.36e-03	-8.49e-04	5.05	5.62
Super-res ($4\times$)	DPS	250	20.43	0.5968	0.2162	90.79	-5.71e-03	4.85e-02	6.62	7.69
	DAPS	100	18.87	0.5112	0.2658	110.31	-1.81e-03	1.15e-01	8.28	9.35
	DDNM	100	22.35	0.7201	0.1592	62.67	-6.06e-03	1.90e-02	5.79	6.63
	ΠGDM	100	22.02	0.7242	0.1281	50.92	-6.28e-03	-1.23e-03	5.39	5.48
	MPGD	100	20.49	0.5809	0.2496	120.27	-4.69e-03	1.05e-01	7.59	9.29
	Palette	100	21.95	0.7220	0.1273	**49.28**	-6.29e-03	-2.98e-03	**5.22**	**5.29**
	EPS (ours)	100	21.96	0.7232	0.1262	49.29	**-6.30e-03**	**-3.00e-03**	5.23	5.30
	EPS (ours)	20	22.56	0.7480	**0.1188**	50.11	-6.24e-03	1.01e-03	5.24	5.50
	EPS (ours)†	1	**24.21**	**0.7999**	0.1265	77.97	-5.63e-03	4.09e-02	6.57	9.27
Gaussian deblur	DPS	250	25.89	0.8530	0.0901	68.78	-5.78e-03	2.85e-02	3.75	6.40
	DAPS	100	27.32	0.8494	0.0771	68.96	-6.80e-03	5.00e-02	2.75	6.89
	DDNM	100	31.27	0.9454	0.0273	20.96	**-6.91e-03**	-5.45e-03	1.87	3.15
	ΠGDM	100	30.18	0.9332	0.0339	27.30	-6.85e-03	-8.48e-04	2.26	3.86
	MPGD	100	19.88	0.6716	0.1770	83.31	2.72e-02	5.93e-02	7.59	7.84
	Palette	100	30.47	0.9397	0.0286	21.63	-6.90e-03	-5.65e-03	1.91	3.06
	EPS (ours)	100	30.82	0.9408	0.0273	20.68	**-6.91e-03**	**-5.73e-03**	**1.84**	**2.99**
	EPS (ours)	20	**31.47**	0.9488	**0.0249**	**20.36**	**-6.91e-03**	-4.97e-03	**1.84**	3.13
	EPS (ours)†	1	30.86	**0.9503**	0.0369	31.39	-6.62e-03	2.35e-03	3.21	6.15
Motion deblur	DPS	250	27.85	0.8694	0.0752	70.24	-6.76e-03	4.68e-02	3.29	7.20
	DAPS	100	27.18	0.8703	0.0732	53.53	-6.70e-03	1.85e-02	3.12	5.92
	DDNM	100	**30.96**	0.9411	0.0297	**22.06**	**-6.91e-03**	-5.32e-03	**1.93**	3.23
	ΠGDM	100	25.04	0.8529	0.0940	54.68	-6.23e-03	1.24e-02	4.10	5.49
	MPGD	100	27.53	0.8764	0.0746	77.02	-6.63e-03	5.17e-02	3.62	7.63
	Palette	100	27.79	0.9099	0.0404	28.23	-6.77e-03	-4.98e-03	2.67	3.59
	EPS (ours)	100	30.27	0.9339	0.0311	23.10	-6.90e-03	**-5.39e-03**	1.94	**3.18**
	EPS (ours)	20	30.88	**0.9422**	**0.0285**	22.79	**-6.91e-03**	-4.44e-03	1.94	3.32
	EPS (ours)†	1	27.73	0.9078	0.0669	49.55	-6.31e-03	1.21e-02	4.60	7.76

Figures 7 and 8 show qualitative reconstructions on FFHQ-64 and ImageNet-64 across the five inverse problems, comparing EPS against DPS, DAPS, DDNM, ΠGDM, MPGD, and Palette. Two example observations per task; numbers in the bottom-right corner of each panel are per-image PSNR. On FFHQ-64, EPS recovers facial structure (eyes, mouth, hairline) under aggressive random inpainting and box inpainting, and produces sharper texture and edge geometry on super-resolution and deblurring than the sampling-based and Palette baselines. On ImageNet-64 the same pattern holds

on a more diverse class-conditional distribution, with EPS preserving operator-consistent structure where DPS, DAPS, and MPGD oversmooth or hallucinate texture inconsistent with the measurement.

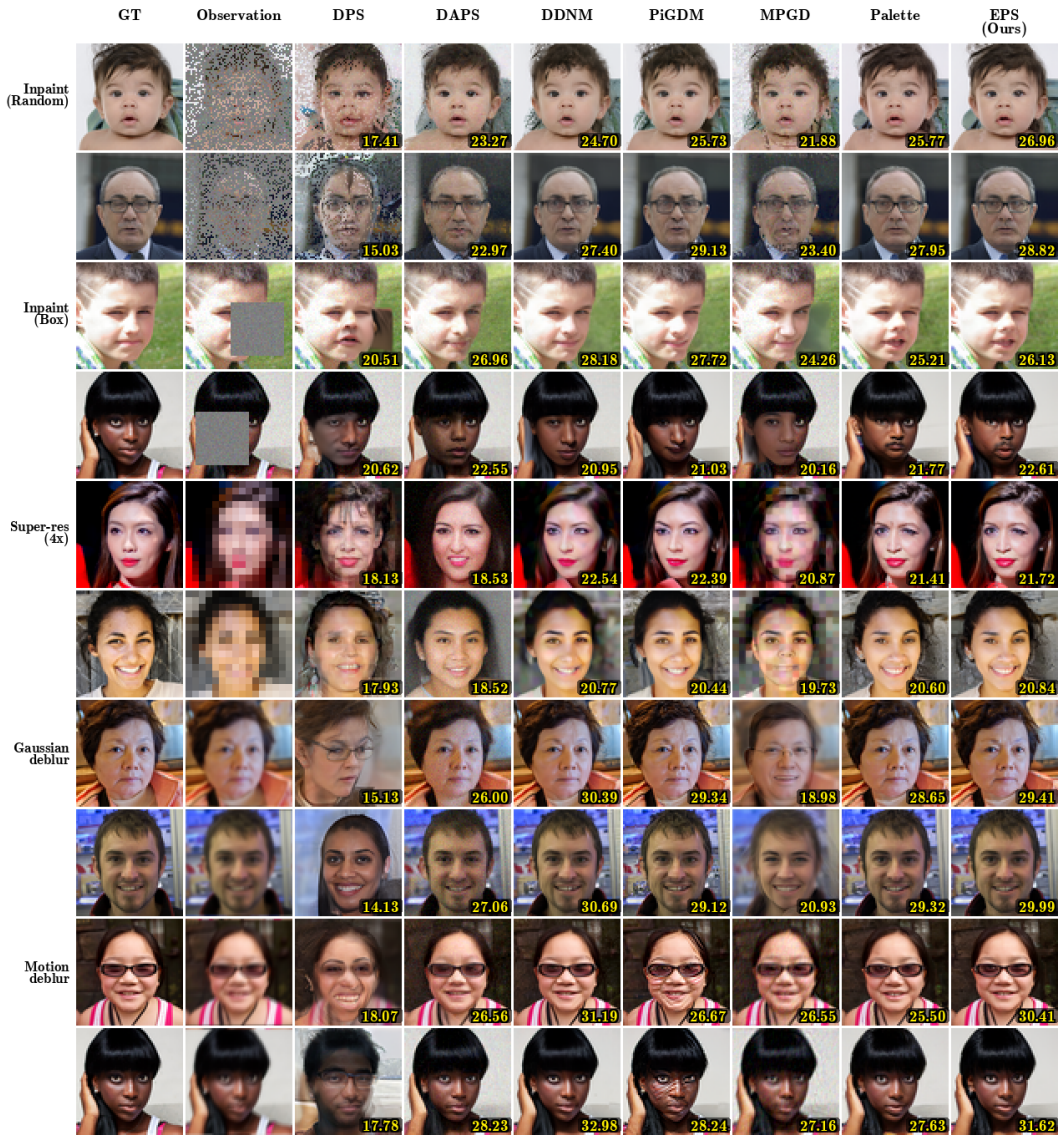

Figure 7: **Qualitative reconstructions on FFHQ-64.** Two example observations per task across the five inverse problems; numbers in the bottom-right corner of each panel are per-image PSNR. EPS recovers facial structure under aggressive random inpainting and box inpainting, and produces sharper texture and edge geometry on super-resolution and deblurring than the sampling-based and Palette baselines.

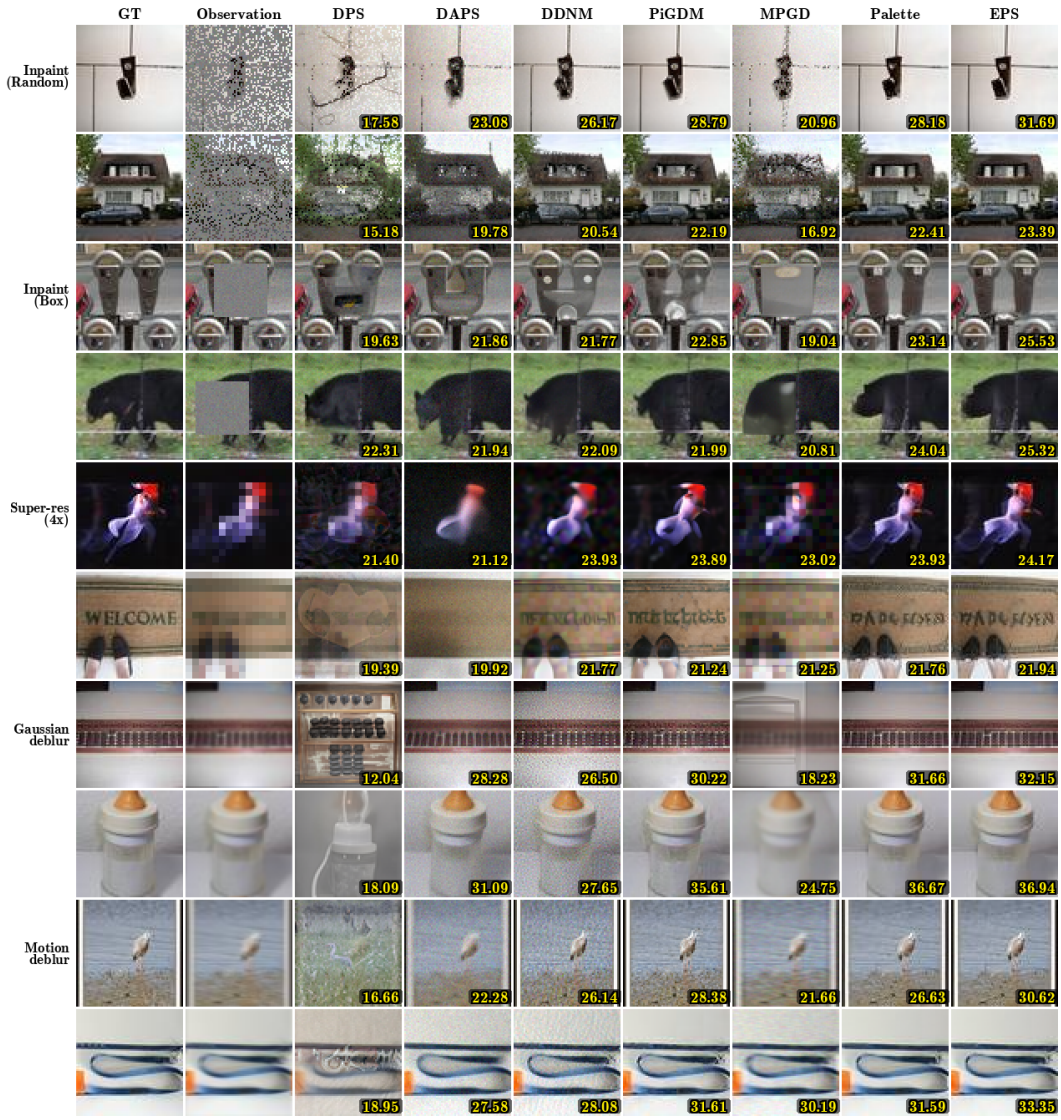

Figure 8: **Qualitative reconstructions on ImageNet-64.** Same layout as Fig. 7. EPS preserves operator-consistent structure where DPS, DAPS, and MPGD oversmooth or hallucinate texture inconsistent with the measurement.

D.8 Extreme Tasks 64×64

We test EPS on two extreme regimes that fall outside the main-text protocol: random inpainting with 95% of pixels missing (only 5% observed) and $16\times$ super-resolution (a 4×4 low-resolution observation upsampled to 64×64). Both push the operator nullspace to occupy almost the entire signal space, so the prior must do most of the reconstruction work and the measurement-matching score is correspondingly noisier.

Table 9 reports this comparison on ImageNet-64 against DPS, DAPS, DDNM, ΠGDM, and MPGD; Palette is omitted because no Palette checkpoint was trained for these regimes. EPS is the strongest method on perceptual and distributional metrics across both tasks (LPIPS, FID, MMD, CRPS), and is competitive on PSNR/SSIM with the strongest sampling-based baselines despite their having access to a full sampler trajectory. On 95% inpainting, EPS-20 reduces FID by roughly 25% over the best sampling-based baseline (ΠGDM at 195) and roughly 30% over DPS. On $16\times$ super-resolution, the perceptual gap is smaller because the measurement is a 4×4 image, one average per 16×16 block, which leaves little measurement signal in the pivot to exploit.

Table 9: **EPS holds up under extreme operator nullspace.** Quantitative comparison on ImageNet-64 in two extreme inverse-problem regimes: 95% random inpainting (only 5% of pixels observed) and $16\times$ super-resolution (a 4×4 low-resolution observation upsampled to 64×64). All baselines use a 1-NFE-per-step Euler/DDIM sampler; EPS uses the EDM Euler sampler. Reported NFE equals the number of sampler iterations. Best in **bold**, second-best underlined; EPS rows highlighted in light pink. Palette is omitted because no Palette checkpoint was trained for these regimes.

Task	Method	NFE	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
Inpaint (95% masked)	DPS	250	14.14	0.2101	0.5066	210.05	1.63e-02	2.48e-02	12.78	10.68
	DAPS	100	14.20	0.1753	0.5962	329.19	1.08e-01	2.14e-01	15.96	14.77
	DDNM	100	14.47	0.2316	0.5480	251.66	6.91e-02	7.34e-02	14.37	11.91
	ΠGDM	100	16.09	0.3015	0.4481	195.18	9.08e-03	1.50e-02	10.96	10.10
	MPGD	100	13.87	0.2073	0.5632	267.41	7.01e-02	9.36e-02	15.15	12.34
	EPS (ours)	100	17.85	0.4468	0.2813	145.17	**-5.27e-03**	**-2.11e-03**	8.38	**7.93**
	EPS (ours)	20	**18.41**	**0.4755**	**0.2712**	**144.18**	-5.08e-03	-1.68e-03	**8.34**	8.00
Super-res ($16\times$)	DPS	250	13.30	0.1171	0.5550	191.98	2.63e-03	7.74e-03	13.88	10.23
	DAPS	100	12.70	0.1437	0.6439	267.84	7.64e-02	9.77e-02	17.51	12.47
	DDNM	100	**15.84**	**0.2295**	0.5787	224.62	7.58e-03	5.73e-02	13.30	11.62
	ΠGDM	100	14.43	0.1703	0.5073	181.06	-2.85e-03	8.31e-03	12.71	9.72
	MPGD	100	13.84	0.1875	0.6176	233.46	8.89e-02	5.67e-02	15.85	11.61
	EPS (ours)	100	14.17	0.1585	0.5007	**175.76**	**-3.10e-03**	**3.34e-04**	12.71	**9.60**
	EPS (ours)	20	14.86	0.1822	**0.4948**	180.14	-6.49e-04	3.63e-03	**12.67**	9.81

Figure 9 shows EPS reconstructions across six independent latent seeds on three 95%-inpainting and three $16\times$ super-resolution observations. Samples agree on the broad spatial layout dictated by the few observed pixels but diverge sharply in fine structure and unobserved content (foreground identity, background texture, occluded geometry), which is the qualitative signature of a calibrated posterior in a regime where the operator nullspace dominates.

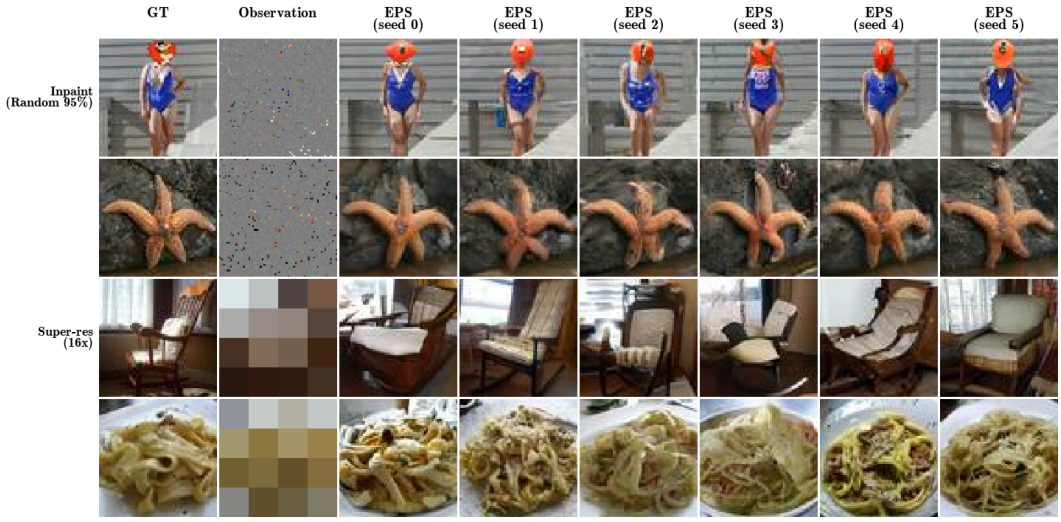

Figure 9: **Posterior diversity under extreme operator nullspace.** EPS reconstructions on ImageNet-64 across six independent latent seeds. Top three rows: 95% random inpainting (only 5% of pixels observed). Bottom three rows: $16\times$ super-resolution (a 4×4 low-resolution observation upsampled to 64×64). Samples agree on the broad spatial layout consistent with the measurement but vary substantially in unobserved directions, illustrating that EPS produces genuinely distinct posterior samples rather than near-duplicates of a single conditional mean.

D.9 OOD Mask-Density Experiments

We test how EPS and Palette generalize when the test-time mask density differs from training. Both checkpoints are trained on random inpainting at 70% masking and frozen; at evaluation we re-sample masks at five densities, ranging from 50% (easier than training) through 70% (in-distribution) to 90% (much harder than training). All results use NFE=100 with the EDM Euler sampler over 100 images $\times$ 10 seeds, with σ_y=0.05.

Table 10 reports this comparison on ImageNet-64 (top) and FFHQ-64 (bottom). On ImageNet, EPS wins on every metric at every density except 80% (near-tie). On FFHQ, EPS dominates in- and near-distribution (50%-70%) but degrades faster than Palette on PSNR/SSIM at heavily-OOD densities (80%-90%); EPS still wins on the distributional metrics (MMD-pix, CRPS) at every density. The PSNR/SSIM crossover at high mask fraction is consistent with the closed-form pivot $\mu_\star$ extrapolating poorly when the operator shifts far from training at test time, since the precision weighting of the pivot is calibrated to a 70% mask under σ_y=0.05 and is mismatched when the actual mask density changes.

Table 10: **OOD generalization across mask density.** Both checkpoints are trained on random inpainting at 70% masking and frozen; evaluation re-samples masks at five densities (50%, 60%, 70% in-distribution, 80%, 90%) on the same eval x_0 at σ_y=0.05. ImageNet-64 (top): EPS wins on every metric at every density except 80% (near-tie). FFHQ-64 (bottom): EPS leads on PSNR, SSIM, LPIPS and FID in and near the training distribution but degrades faster than Palette on PSNR/SSIM at heavily-OOD densities, while still winning on MMD-pix and CRPS-pix at every density. All numbers at NFE=100 (EDM Euler). Best per row in **bold**; EPS rows highlighted in light pink.

Mask %	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
				ImageNet 64×64					
50%	Palette	27.10	0.8820	0.0608	55.12	-6.71e-03	-5.26e-03	2.87	4.41
	EPS (ours)	**27.88**	**0.8969**	**0.0545**	**51.79**	**-6.76e-03**	**-5.32e-03**	**2.69**	**4.31**
60%	Palette	25.62	0.8415	0.0778	65.85	-6.63e-03	-5.01e-03	3.44	4.83
	EPS (ours)	**26.07**	**0.8526**	**0.0729**	**62.98**	**-6.66e-03**	**-5.10e-03**	**3.31**	**4.73**
70% (in-dist.)	Palette	24.09	0.7869	0.1011	81.88	-6.50e-03	-4.41e-03	4.16	5.52
	EPS (ours)	**24.34**	**0.7948**	**0.0979**	**79.60**	**-6.52e-03**	**-4.51e-03**	**4.04**	**5.41**
80%	Palette	22.29	**0.7084**	**0.1399**	**98.49**	-6.27e-03	**-3.69e-03**	5.14	6.06
	EPS (ours)	**22.30**	0.7080	0.1403	98.91	**-6.28e-03**	-3.61e-03	**5.08**	**6.04**
90%	Palette	18.60	0.5135	0.2635	152.54	-3.54e-03	3.38e-03	8.04	8.33
	EPS (ours)	**19.06**	**0.5258**	**0.2456**	**144.18**	**-5.01e-03**	**6.27e-04**	**7.19**	**7.78**
				FFHQ 64×64					
50%	Palette	27.95	0.9171	0.0414	25.07	-6.47e-03	**-5.32e-03**	2.50	**3.32**
	EPS (ours)	**29.64**	**0.9409**	**0.0307**	**22.92**	**-6.86e-03**	-3.85e-03	**2.19**	3.34
60%	Palette	27.11	0.9061	0.0470	28.16	-6.70e-03	**-5.11e-03**	2.77	3.55
	EPS (ours)	**28.05**	**0.9206**	**0.0393**	**26.19**	**-6.82e-03**	-4.48e-03	**2.58**	**3.49**
70% (in-dist.)	Palette	25.76	0.8809	0.0593	33.30	-6.71e-03	-4.76e-03	3.31	3.87
	EPS (ours)	**26.16**	**0.8879**	**0.0533**	**31.87**	**-6.74e-03**	**-5.00e-03**	**3.16**	**3.75**
80%	Palette	**23.78**	**0.8321**	**0.0826**	**40.23**	-6.42e-03	-3.44e-03	4.28	4.51
	EPS (ours)	23.57	0.8250	0.0838	40.83	**-6.53e-03**	**-3.58e-03**	**4.19**	**4.45**
90%	Palette	**18.34**	**0.6023**	0.2286	**83.35**	4.20e-03	**2.17e-02**	8.68	7.09
	EPS (ours)	17.82	0.5683	**0.2284**	90.78	**-1.54e-03**	3.45e-02	**7.71**	**7.08**

Figure 10 shows qualitative reconstructions from EPS and Palette across the five mask densities on both datasets. The visual gap between the two methods is largest in the in-distribution regime and narrows at the extremes: at 50% masking both methods recover most of the image structure, while at 90% both methods struggle and the reconstructions diverge sharply from the ground truth.

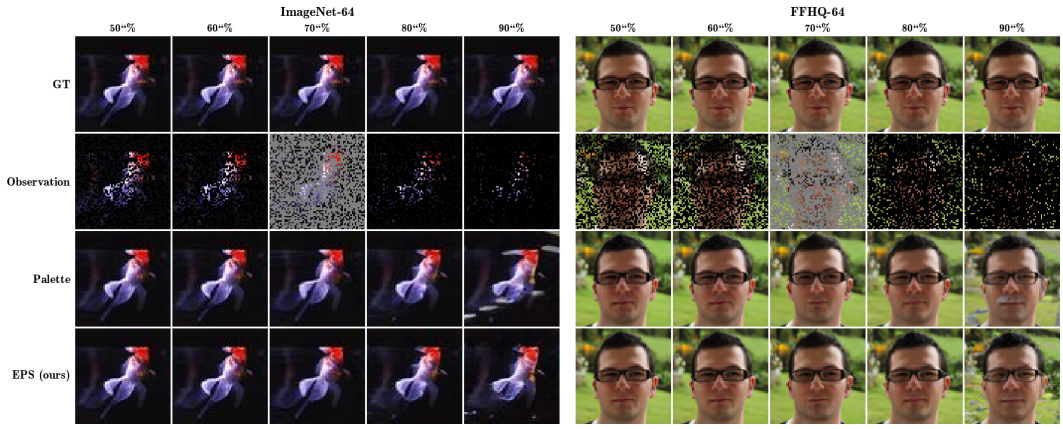

Figure 10: **Qualitative OOD generalization across mask density.** EPS and Palette reconstructions on ImageNet-64 and FFHQ-64 across five mask densities (50%, 60%, 70% in-distribution, 80%, 90%). Both checkpoints were trained on 70% masking and frozen at evaluation. EPS preserves operator-consistent structure best in- and near-distribution; degradation at heavily-OOD densities (80%, 90%) is consistent with the pivot $\mu_\star$ being calibrated to a 70% mask.

D.10 Additional 256×256 Results

Table 11 reports EPS at NFE$\in \{1, 20, 100\}$ against sampling-based and training-based baselines on ImageNet 256×256 across five inverse problems. EDM-DDPM++ architecture is used, and trained from scratch on each task (no pretrained backbone), because no pretrained 256×256 checkpoint exists under our codebase and validation-split protocol. Baseline numbers are taken directly from the DAPS paper [22]; we did not re-run them under our protocol, and distributional metrics (CRPS, MMD) are not reported because they are not available in the source paper. This comparison is therefore not controlled. The quoted baselines use strong pretrained priors, while EPS uses a weaker prior trained from scratch. The controlled high-resolution comparison, in which every method uses the same pretrained prior, is the 1024×1024 study of Section 4.3 and Appendix F. EPS leads on both inpainting tasks and on Gaussian deblurring, particularly on perceptual metrics (LPIPS, FID), and is competitive with DAPS on motion deblurring (taking second place on SSIM, LPIPS, and FID at NFE=20). DAPS retains an edge on $4\times$ super-resolution. The 1-NFE EPS row is the strongest EPS variant on PSNR/SSIM on every task except motion deblurring, mirroring the high-noise posterior-mean check of Section 3.5.

Table 11: **ImageNet** 256×256 **results.** EPS vs. sampling-based and training-based baselines on five linear inverse problems at 256×256 resolution. Baseline numbers are taken from the DAPS paper [22]. Best in **bold**, second-best underlined; EPS rows highlighted in light pink. † The NFE=1 row applies a single direct Tweedie evaluation $D_\theta(\mu_\star, \sigma_{\max})$, returning the conditional posterior mean rather than a posterior sample.

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓
Super-res ($4\times$)	DAPS	**25.89**	0.694	0.276	**83.57**
	DPS	21.13	0.489	0.361	106.32
	DDRM	22.62	0.521	0.324	103.85
	DDNM	23.96	0.604	0.475	98.62
	DCDP	–	–	–	–
	FPS-SMC	24.82	**0.703**	0.313	97.51
	DiffPIR	23.18	–	0.371	106.32
	EPS, NFE=100	19.60	0.520	0.294	138.55
	EPS, NFE=20	20.32	0.561	0.277	130.66
	EPS, NFE=1†	22.65	0.671	**0.254**	116.82
Gaussian deblur	DAPS	26.15	0.684	0.253	75.68
	DPS	20.31	0.598	0.397	116.42
	DDRM	21.26	0.564	0.443	146.89
	DDNM	**28.06**	0.703	0.278	81.43
	DCDP	–	–	–	–
	FPS-SMC	23.91	0.601	0.387	91.72
	DiffPIR	22.80	–	0.355	93.36
	EPS, NFE=100	24.34	0.640	0.224	72.38
	EPS, NFE=20	24.92	0.666	**0.222**	**71.31**
	EPS, NFE=1†	25.90	**0.704**	0.249	100.10
Motion deblur	DAPS	**27.86**	**0.766**	**0.196**	**61.83**
	DPS	18.96	0.629	0.423	137.81
	DCDP	–	–	–	–
	FPS-SMC	24.52	0.647	0.326	87.43
	DiffPIR	24.01	–	0.366	94.63
	EPS, NFE=100	23.65	0.616	0.254	79.79
	EPS, NFE=20	24.39	0.655	0.240	75.52
	EPS, NFE=1†	24.12	0.647	0.318	137.59

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓
Inpaint (box)	DAPS	21.43	0.725	0.214	109.85
	DPS	18.94	0.722	0.257	126.52
	DDRM	18.63	0.733	0.254	116.37
	DDNM	21.64	0.748	0.319	103.97
	DCDP	–	–	–	–
	FPS-SMC	**22.16**	0.726	0.208	111.58
	EPS, NFE=100	19.57	0.785	0.154	100.30
	EPS, NFE=20	20.12	0.795	**0.149**	**94.22**
	EPS, NFE=1†	22.13	**0.809**	0.177	113.50
Inpaint (random)	DAPS	28.44	0.775	0.135	54.25
	DPS	23.52	0.745	0.297	87.53
	DDNM	**31.16**	0.841	0.191	63.84
	DCDP	–	–	–	–
	FPS-SMC	24.52	0.701	0.316	79.12
	EPS, NFE=100	27.26	0.817	0.113	21.39
	EPS, NFE=20	27.90	0.836	0.104	**20.17**
	EPS, NFE=1†	29.73	**0.876**	**0.103**	24.03

Figure 11 shows qualitative reconstructions of EPS on ImageNet 256×256 across the five inverse problems, with one example observation per task. EPS recovers sharp object structure under aggressive random inpainting and box inpainting, and produces texture and edge geometry consistent with the measurement on super-resolution and deblurring at this higher resolution.

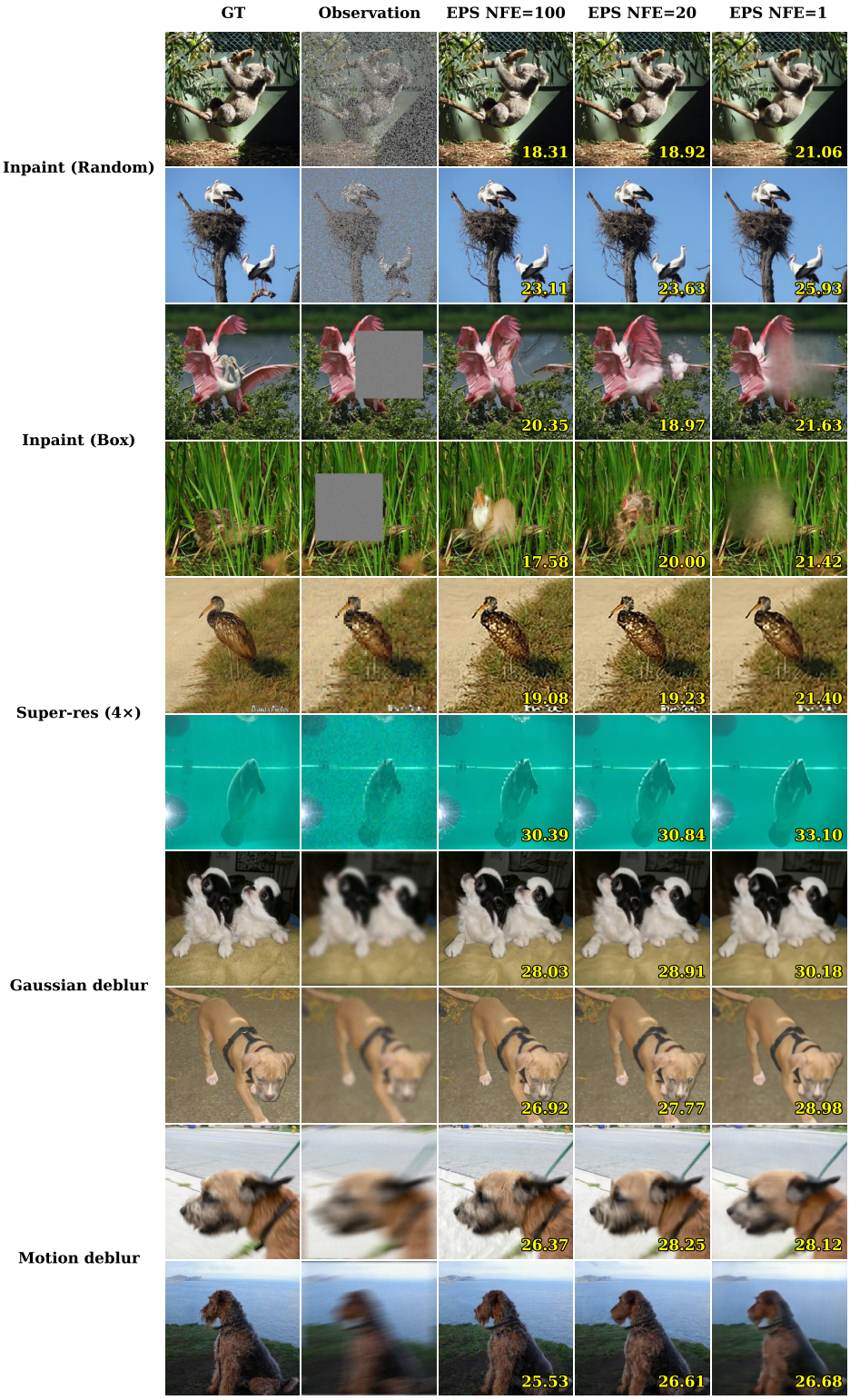

Figure 11: **Qualitative reconstructions on ImageNet-256.** EPS reconstructions across the five inverse problems on ImageNet 256×256. Numbers in the bottom-right corner of each panel are per-image PSNR. EPS preserves operator-consistent structure under aggressive inpainting and produces sharp texture and edges on super-resolution and deblurring at this higher resolution.

D.11 Palette vs. EPS at Matched NFE

The one-step posterior-mean check in Section D.6 showed that a single Tweedie call recovers most of the multi-step PSNR/SSIM gain. Here we ask the matched question for Palette: does the same one-step shortcut close the gap to EPS? Concretely, we evaluate Palette $[x_t, y, t]$ and EPS $[\mu_\star, y, t]$ at NFE=1 (a single direct denoiser call at $\sigma_{\max}$=80, no sampler loop) and at NFE=100 (EDM Euler sampler), on both datasets and all five tasks.

Table 12 reports this comparison on FFHQ-64 (top) and ImageNet-64 (bottom). Two patterns dominate. First, at NFE=1 the two methods largely coincide, as predicted by Observation 4: at $\sigma_{\max}$ the noisy state carries vanishing information about x_0, so both the Palette input (x_t, y) and the EPS input $(\mu_\star, y)$ are informationally equivalent to y alone, and both networks estimate $\mathbb{E}[x_0|y]$. Second, at NFE=100 the EPS pivot becomes informative and EPS dominates Palette across distortion and distributional metrics on most tasks. Both 1-NFE rows trade perceptual quality for distortion: PSNR/SSIM jump sharply because the conditional mean is the MMSE-optimal estimator under squared error, but FID, LPIPS, CRPS, and MMD all degrade because no actual sample is produced.

Table 12: **Palette vs. EPS at NFE=1 and NFE=100.** One-step (single Tweedie call at $\sigma_{\max}$=80) vs. 100-step (EDM Euler) generation for Palette $[x_t, y, t]$ and EPS $[\mu_\star, y, t]$ on FFHQ-64 (top) and ImageNet-64 (bottom). At NFE=1, both inputs carry essentially only the information in y (Observation 4), and the two methods agree on inpainting and super-resolution while Palette is better on deblurring; at NFE=100 the pivot becomes informative and EPS is ahead in PSNR on every task. Best in **bold**, second-best underlined; EPS rows highlighted in light pink.

Task	Method	PSNR ↑	SSIM ↑	LPIPS ↓	FID ↓	MMD-pix ↓	MMD-Inc ↓	CRPS-pix ↓	CRPS-Inc ↓
				FFHQ 64×64					
Average	Palette (NFE=100)	26.03	0.8590	0.0626	31.50	-6.6e-03	-4.7e-03	3.43	3.86
	Palette (NFE=1)	**27.77**	**0.8920**	0.0638	41.93	-6.4e-03	8.0e-03	4.36	6.21
	EPS (NFE=1)	27.34	0.8887	0.0699	46.30	-6.3e-03	1.1e-02	4.67	6.91
	EPS (NFE=100)	26.69	0.8661	**0.0590**	**29.94**	**-6.7e-03**	**-4.9e-03**	**3.24**	**3.73**
Random inpaint	Palette (NFE=100)	25.76	0.8809	0.0593	33.30	-6.7e-03	-4.8e-03	3.31	3.87
	Palette (NFE=1)	27.58	0.9095	0.0551	39.20	-6.5e-03	1.1e-03	4.24	5.96
	EPS (NFE=1)	**27.90**	**0.9148**	0.0549	39.71	-6.6e-03	2.2e-03	3.94	5.75
	EPS (NFE=100)	26.16	0.8879	**0.0533**	**31.87**	**-6.7e-03**	**-5.0e-03**	**3.16**	**3.75**
Box inpaint	Palette (NFE=100)	24.18	0.8426	0.0577	25.09	-6.6e-03	-5.3e-03	4.02	3.47
	Palette (NFE=1)	**26.14**	**0.8727**	0.0639	32.60	-6.3e-03	-1.2e-03	5.01	5.51
	EPS (NFE=1)	26.01	0.8706	0.0644	32.90	-6.4e-03	-8.5e-04	5.05	5.62
	EPS (NFE=100)	24.23	0.8448	**0.0567**	**24.74**	**-6.6e-03**	**-5.4e-03**	**4.01**	**3.45**
Super-res (4×)	Palette (NFE=100)	21.95	0.7220	0.1273	**49.28**	-6.3e-03	-3.0e-03	**5.22**	**5.29**
	Palette (NFE=1)	**24.23**	**0.8009**	**0.1255**	77.61	-5.6e-03	4.1e-02	6.56	9.31
	EPS (NFE=1)	24.21	0.7999	0.1265	77.97	-5.6e-03	4.1e-02	6.57	9.27
	EPS (NFE=100)	21.96	0.7232	0.1262	49.29	**-6.3e-03**	**-3.0e-03**	5.23	5.30
Gaussian deblur	Palette (NFE=100)	30.47	0.9397	0.0286	21.63	-6.9e-03	-5.6e-03	1.91	3.06
	Palette (NFE=1)	**32.04**	**0.9529**	0.0285	24.49	-6.8e-03	-1.9e-03	2.40	4.61
	EPS (NFE=1)	30.86	0.9503	0.0369	31.39	-6.6e-03	2.4e-03	3.21	6.15
	EPS (NFE=100)	30.82	0.9408	**0.0273**	**20.68**	**-6.9e-03**	**-5.7e-03**	**1.84**	**2.99**
Motion deblur	Palette (NFE=100)	27.79	0.9099	0.0404	28.23	-6.8e-03	-5.0e-03	2.67	3.59
	Palette (NFE=1)	28.87	0.9240	0.0461	35.76	-6.6e-03	1.4e-03	3.59	5.68
	EPS (NFE=1)	27.73	0.9078	0.0669	49.55	-6.3e-03	1.2e-02	4.60	7.76
	EPS (NFE=100)	**30.27**	**0.9339**	**0.0311**	**23.10**	**-6.9e-03**	**-5.4e-03**	**1.94**	**3.18**
				ImageNet 64×64					
Average	Palette (NFE=100)	24.32	0.7673	0.1124	82.57	-6.4e-03	-4.4e-03	4.35	5.54
	Palette (NFE=1)	**26.73**	**0.8289**	0.1215	104.38	-6.1e-03	4.3e-03	5.35	9.37
	EPS (NFE=1)	25.67	0.8165	0.1318	110.95	-5.6e-03	4.5e-03	5.91	9.96
	EPS (NFE=100)	24.53	0.7712	**0.1103**	**81.46**	**-6.4e-03**	**-4.4e-03**	**4.27**	**5.48**
Random inpaint	Palette (NFE=100)	24.09	0.7869	0.1011	81.88	-6.5e-03	-4.4e-03	4.16	5.52
	Palette (NFE=1)	26.50	0.8550	**0.0926**	91.79	-6.4e-03	4.3e-04	5.16	8.90
	EPS (NFE=1)	**26.60**	**0.8580**	0.0933	88.59	-6.1e-03	-4.8e-04	4.98	8.32
	EPS (NFE=100)	24.34	0.7948	0.0979	**79.60**	**-6.5e-03**	**-4.5e-03**	**4.04**	**5.41**
Box inpaint	Palette (NFE=100)	21.12	0.7541	0.1218	92.73	-6.1e-03	-4.1e-03	5.92	5.93
	Palette (NFE=1)	**23.60**	**0.7926**	0.1539	131.54	-5.5e-03	8.7e-03	7.19	10.67
	EPS (NFE=1)	23.60	0.7908	0.1514	129.11	-5.6e-03	7.3e-03	7.16	10.38
	EPS (NFE=100)	21.24	0.7569	**0.1196**	**91.07**	**-6.1e-03**	**-4.2e-03**	**5.87**	**5.84**
Super-res (4×)	Palette (NFE=100)	20.24	0.5364	0.2220	**128.76**	**-5.9e-03**	-2.8e-03	**6.50**	**7.33**
	Palette (NFE=1)	**22.79**	**0.6538**	0.2452	185.15	-4.9e-03	1.9e-02	8.05	13.63
	EPS (NFE=1)	22.78	0.6530	0.2455	182.92	-5.0e-03	2.0e-02	8.06	13.47
	EPS (NFE=100)	20.25	0.5369	**0.2207**	128.80	-5.9e-03	**-2.8e-03**	6.52	7.35
Gaussian deblur	Palette (NFE=100)	29.15	0.9010	0.0491	46.62	-6.8e-03	**-5.6e-03**	2.26	**4.11**
	Palette (NFE=1)	**31.73**	**0.9407**	**0.0441**	**43.40**	**-6.8e-03**	-4.2e-03	2.74	6.15
	EPS (NFE=1)	28.82	0.9194	0.0606	56.53	-5.1e-03	-3.5e-03	4.09	7.73
	EPS (NFE=100)	29.18	0.9015	0.0486	46.55	-6.8e-03	-5.6e-03	**2.25**	4.11
Motion deblur	Palette (NFE=100)	27.02	0.8582	0.0680	62.86	-6.7e-03	-5.0e-03	2.93	4.81
	Palette (NFE=1)	**29.04**	**0.9023**	0.0715	70.01	-6.7e-03	-2.3e-03	3.63	7.51
	EPS (NFE=1)	26.56	0.8613	0.1079	97.59	-6.0e-03	-7.3e-04	5.25	9.91
	EPS (NFE=100)	27.62	0.8661	**0.0647**	**61.29**	**-6.8e-03**	**-5.1e-03**	**2.69**	**4.70**

D.12 Runtime Analysis

We measure single-image wall-clock sampling latency on ImageNet 64×64 at NFE$=100$ with batch size 1. All methods use the same EDM-ADM [17] denoiser checkpoint (`edm-imagenet-64x64-cond-adm.pkl`, $\sim$296M parameters) on a single NVIDIA B200 GPU, with class labels set to the evaluation-set ground-truth ImageNet-1k classes. Each cell in Table 13 is the mean of five independent sampling runs after two warm-up runs that amortise CUDA kernel JIT and cuDNN auto-tuning. All methods use the Euler ODE schedule (`second_order=False`) so NFE equals the number of sampler steps; for DAPS we set annealing_steps$=20$ and ode_steps$=5$ so total NFE matches.

The dominant cost in all methods is the U-Net forward, and for DPS and ΠGDM the U-Net backward as well. EDM (uncond.) runs the bare pretrained denoiser with no measurement-aware updates and is task-independent. DPS and ΠGDM require a backward pass per step (likelihood gradient / Jacobian-vector product), $2.3\times$ and $2.4\times$ the EDM unconditional cost. DAPS runs a nested ODE rollout plus Langevin correction at each annealing step, $2.2\times$ the unconditional cost. DDNM and MPGD ($1.3\times$) add a closed-form nullspace or manifold projection on top of a single denoiser forward. EPS is the fastest sampler that solves the inverse problem. At batch 1 it runs at $1.006\times$ the wall-clock of EDM unconditional, because the structured solve for $\mu_\star$ (FFT for deblurring; element-wise for inpainting and super-resolution) is sub-millisecond. EPS runtime is essentially task-independent: all five tasks land within ±0.04 s of the average. Palette shares EPS's architecture and per-step forward cost, so its runtime is well approximated by the EPS row. At batch 8, EPS is about 3.5% slower per image than EDM unconditional, so its overhead also stays small in throughput terms.

Table 13: **EPS matches the bare denoiser in wall-clock cost.** Per-image sampling latency (s, lower is better) on ImageNet-64 at NFE$=100$, batch size 1, on a single B200 GPU. Among methods that solve the inverse problem, EPS is fastest on every task and on average, at essentially the same cost as the bare EDM unconditional sampler ($1.006\times$). Sampling-based baselines that require a backward pass (DPS, ΠGDM) are $\sim 2.3\times$ slower; nested-rollout methods (DAPS) are $\sim 2.2\times$ slower. † EDM (uncond.) is shown as a reference: it runs the bare pretrained denoiser without any measurement-aware update, so it does not actually solve the inverse problem.

Method	Inpaint (Random)	Inpaint (Box)	Super-res ($4\times$)	Gaussian deblur	Motion deblur	Avg	$\times$ EDM
EDM (uncond.)†	*1.87*	*1.87*	*1.87*	*1.87*	*1.87*	*1.87*	$1.00\times$
DPS	4.61	4.17	4.05	4.14	4.42	4.28	$2.29\times$
DAPS	3.98	4.02	3.99	4.19	4.20	4.07	$2.18\times$
DDNM	2.52	2.50	2.49	2.51	2.51	2.51	$1.34\times$
ΠGDM	4.55	4.55	4.53	4.54	4.55	4.54	$2.43\times$
MPGD	2.51	2.51	2.52	2.53	2.54	2.52	$1.35\times$
EPS (ours)	**1.87**	**1.87**	**1.88**	**1.89**	**1.89**	**1.88**	$\mathbf{1.006}\times$

D.13 Posterior Diversity

Figures 12 and 13 show four EPS reconstructions of the same observation drawn with independent latent seeds, alongside the ground truth, on box inpainting and $4\times$ super-resolution. Samples agree on observed structure while differing in the unobserved directions (skin texture, hair detail, background, occluded foreground content), which is the qualitative signature of a calibrated posterior under a non-trivial operator nullspace.

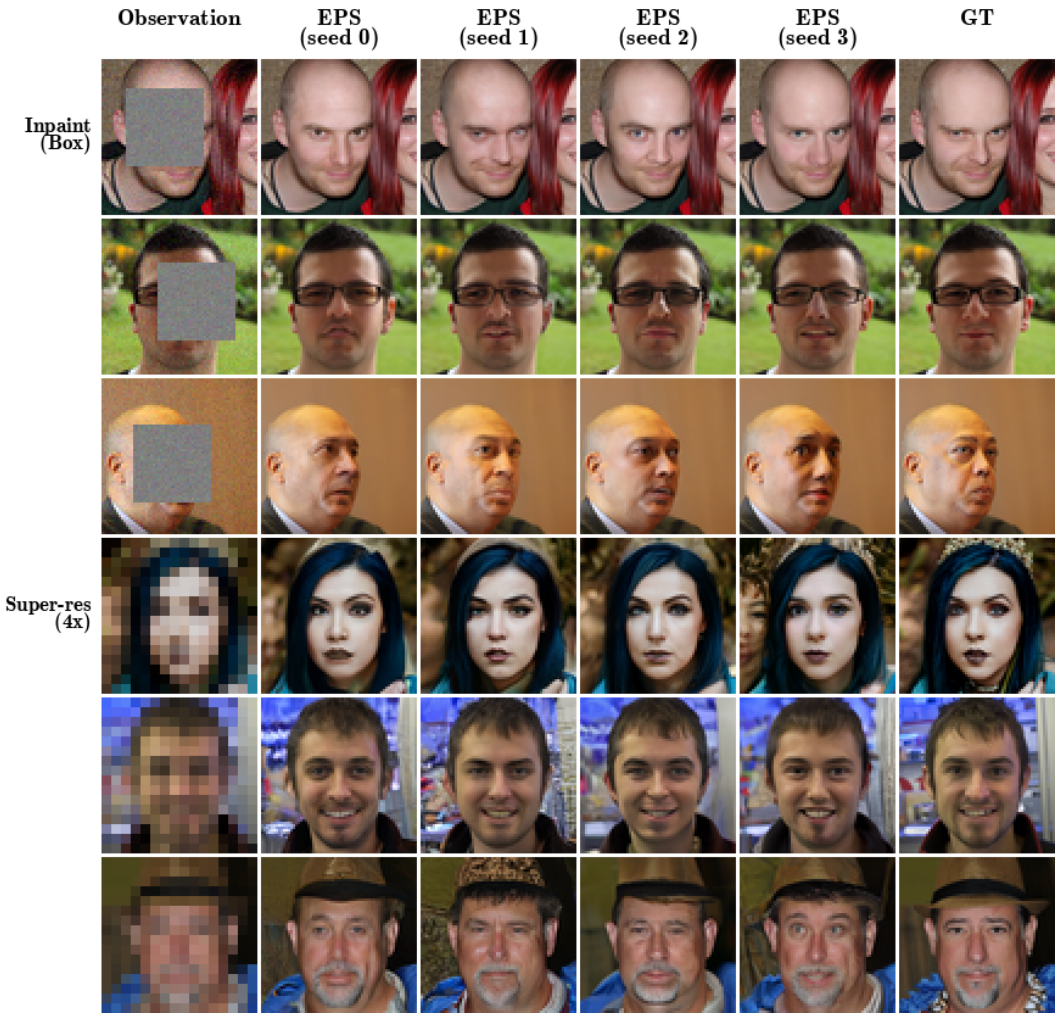

Figure 12: **Posterior diversity from EPS on FFHQ-64.** Four reconstructions per observation drawn with independent latent seeds, alongside the ground truth, on box inpainting and $4\times$ super-resolution. Samples agree on observed structure while differing in unobserved directions (skin texture, hair detail, background), the qualitative signature of a calibrated posterior under a non-trivial operator nullspace.

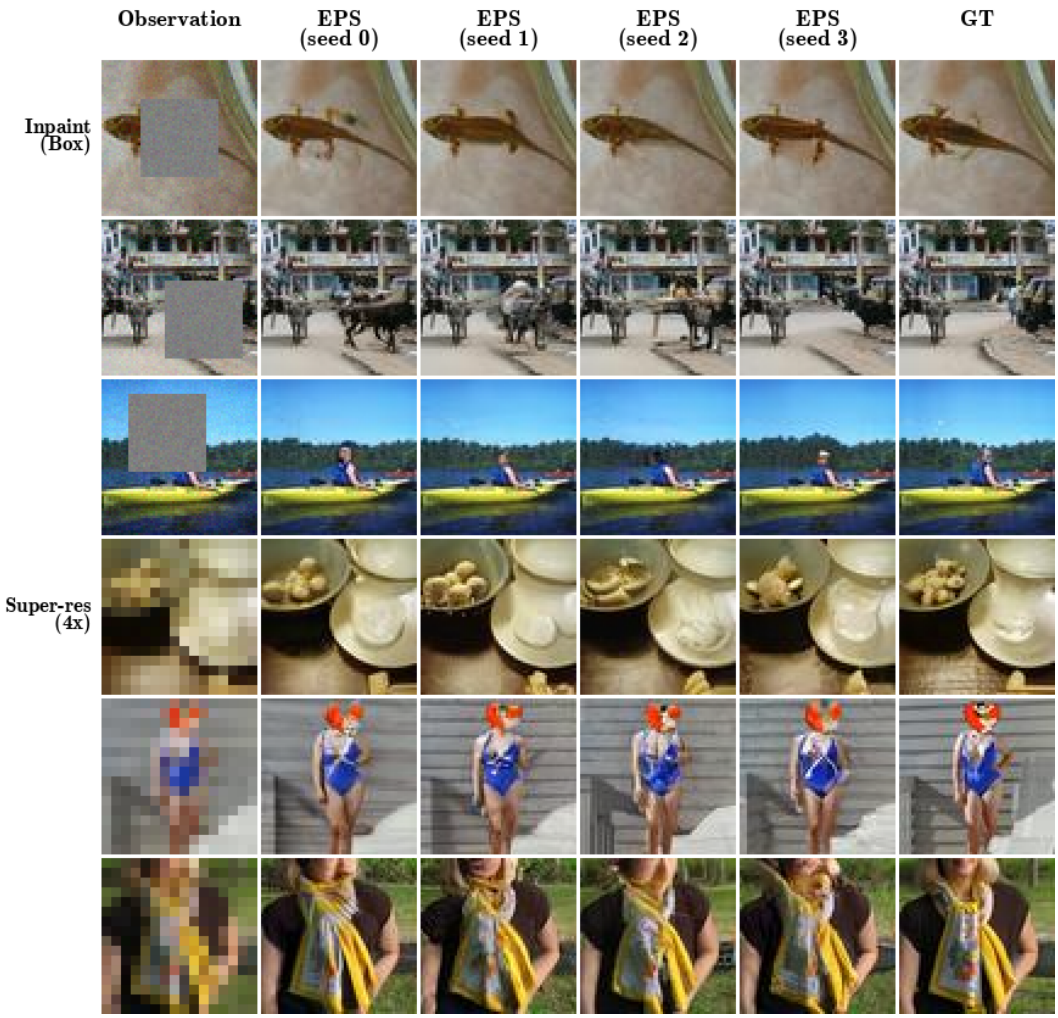

Figure 13: **Posterior diversity from EPS on ImageNet-64.** Same layout as Fig. 12. Diversity is concentrated in the operator nullspace: occluded foreground content varies under box inpainting, while sharp high-frequency detail varies under $4\times$ super-resolution.

D.14 Sampling Budget

Figure 14 compares EPS reconstructions at NFE=1, 20, and 100 on the same observation across all five tasks, on both ImageNet-64 (left) and FFHQ-64 (right). The NFE=1 column is the deterministic high-noise posterior-mean limit (Section 3.5), which is MMSE-optimal in pixel space and yields the highest per-image PSNR. The NFE=20 and NFE=100 columns target posterior samples and trade pointwise fidelity for distributional sharpness, in line with the perception-distortion pattern visible in Table 7.

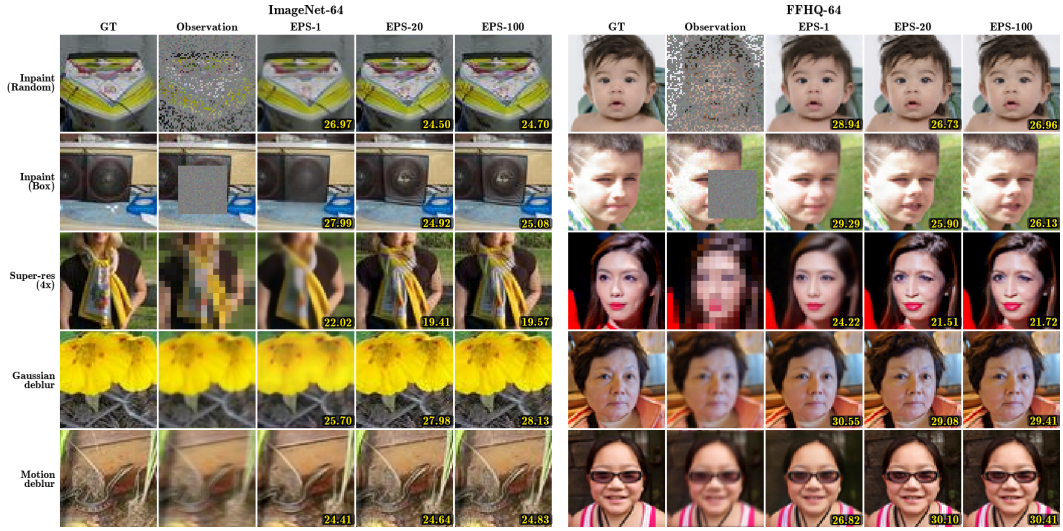

Figure 14: **EPS reconstructions at varying sampling budgets.** ImageNet-64 (left) and FFHQ-64 (right) across the five inverse problems. For each observation, columns show NFE=1, 20, and 100. The NFE=1 column is the deterministic high-noise posterior-mean limit (Section 3.5), MMSE-optimal in pixel space; NFE=20 and NFE=100 target posterior samples and trade pointwise fidelity for distributional sharpness.

D.15 Additional Baselines and Error Bars

Table 14 adds OPC and an InvFusion-style baseline on all five FFHQ-64 tasks and reports error bars for EPS and every sampling-based method of Table 8. OPC [52] uses the same pretrained denoiser as ΠGDM and replaces its scalar posterior variance with the optimal one; it runs at 100 NFE like the other sampling-based methods. The InvFusion-style baseline [34] is a conditional diffusion model that sees the operator. Its input is $[x_t, y, A^\top A]$, where the operator channel is the mask for inpainting, the kernel autocorrelation for deblurring, and the autocorrelation of the averaging kernel for super-resolution. It is trained from the same base checkpoint as EPS, for 40 epochs at batch size 128 (9.4k steps). Retraining it with the exact diagonal of $A^\top A$ for super-resolution, and with the kernel of each batch for motion deblurring, raises its per-sample PSNR by only 0.04 and 0.17 dB, so the table reports the original runs, for which all metrics are available. Its budget is shorter than that of the EPS models in Table 14, which therefore overstates the gap between them. Against EPS models trained for a comparable 9k steps (the left half of Table 16), the InvFusion-style model is behind by 0.06, 1.13, 0.03, 0.67, and 4.00 dB in per-sample PSNR on the five tasks, and it is also behind in SSIM and LPIPS on every task.

Per-image metrics (PSNR, SSIM, LPIPS) carry the standard error of the mean over the 100 evaluation images. Set-level metrics (FID, CRPS, MMD) carry the standard deviation over 1000 bootstrap resamples of the images, in which each image keeps its 10 samples; we report this standard deviation rather than percentile intervals, because resampling biases FID and MMD upward and the percentile intervals can exclude the point estimate. Within these error bars the picture of Table 8 holds. EPS has the best CRPS and MMD on every task and the best FID on every task except motion deblurring, where DDNM is lower by less than one bootstrap standard deviation. EPS outperforms OPC in FID, CRPS, and MMD on every task, and in PSNR it is ahead by 1.6 and 5.7 dB on the deblurring tasks and within one standard error on super-resolution and inpainting. Since all methods are evaluated on the same images and measurements, paired differences between methods are more precise than the per-method error bars suggest.

Table 14: **Additional baselines and error bars on FFHQ-64** (100 NFE, the 100 evaluation images $\times$ 10 posterior samples of Table 8). PSNR, SSIM and LPIPS are per-sample values averaged over the 10 seeds, the estimator of Table 8, $\pm$ the standard error over images; ΠGDM, DDNM, DAPS and MPGD reproduce their Table 8 rows exactly, and the EPS row re-evaluates the paper checkpoints (within 0.05 dB). FID, CRPS (Inception) and MMD^2 (Inception) use all 1000 samples against the 100 evaluation images, $\pm$ the standard deviation over 1000 bootstrap resamples of the images (each image keeps its 10 samples); the smaller reference set shifts FID slightly from Table 8. OPC [52] replaces the scalar posterior variance of ΠGDM with its optimal value and uses the same pretrained denoiser; the InvFusion-style baseline [34] is a conditional model with input $[x_t, y, A^\top A]$ and no pivot, trained from the same base checkpoint for 40 epochs at batch size 128, a shorter budget than the EPS models of this table (Appendix D.15 compares it with EPS at a matched budget). Palette is not included. †DPS at 100 NFE (250 in Table 8).

Metric	Method	SR 4×	Inpaint (box)	Inpaint (random)	Gaussian deblur	Motion deblur
PSNR↑	EPS (ours)	21.99 ± 0.19	24.19 ± 0.23	26.17 ± 0.24	30.86 ± 0.17	30.29 ± 0.17
	OPC	21.92 ± 0.19	24.19 ± 0.26	26.25 ± 0.25	29.28 ± 0.16	24.61 ± 0.29
	InvFusion-style	20.97 ± 0.16	21.98 ± 0.19	24.30 ± 0.19	29.44 ± 0.16	25.01 ± 0.14
	ΠGDM	22.02 ± 0.18	23.80 ± 0.24	26.00 ± 0.23	30.18 ± 0.16	25.04 ± 0.32
	DDNM	22.35 ± 0.17	23.15 ± 0.24	24.50 ± 0.21	31.27 ± 0.16	30.96 ± 0.16
	DAPS	18.87 ± 0.13	23.06 ± 0.22	22.45 ± 0.16	27.32 ± 0.13	27.18 ± 0.16
	MPGD	20.49 ± 0.14	20.39 ± 0.18	20.60 ± 0.16	19.88 ± 0.12	27.53 ± 0.13
	DPS†	17.74 ± 0.10	20.70 ± 0.17	16.57 ± 0.11	13.97 ± 0.09	18.60 ± 0.10
SSIM↑	EPS (ours)	0.7247 ± 0.0058	0.8445 ± 0.0037	0.8882 ± 0.0035	0.9412 ± 0.0017	0.9343 ± 0.0019
	OPC	0.7194 ± 0.0058	0.8427 ± 0.0038	0.8937 ± 0.0034	0.9151 ± 0.0022	0.8251 ± 0.0059
	InvFusion-style	0.6542 ± 0.0053	0.7782 ± 0.0032	0.8390 ± 0.0035	0.9272 ± 0.0016	0.8543 ± 0.0025
	ΠGDM	0.7242 ± 0.0055	0.8359 ± 0.0037	0.8898 ± 0.0033	0.9332 ± 0.0018	0.8529 ± 0.0059
	DDNM	0.7201 ± 0.0048	0.8231 ± 0.0039	0.8566 ± 0.0037	0.9454 ± 0.0016	0.9411 ± 0.0017
	DAPS	0.5112 ± 0.0071	0.7968 ± 0.0044	0.7576 ± 0.0045	0.8494 ± 0.0031	0.8703 ± 0.0028
	MPGD	0.5809 ± 0.0041	0.7211 ± 0.0046	0.7097 ± 0.0050	0.6716 ± 0.0046	0.8764 ± 0.0023
	DPS†	0.4545 ± 0.0047	0.7295 ± 0.0043	0.5037 ± 0.0052	0.3653 ± 0.0046	0.5874 ± 0.0049
LPIPS↓	EPS (ours)	0.1257 ± 0.0030	0.0569 ± 0.0019	0.0535 ± 0.0016	0.0274 ± 0.0008	0.0312 ± 0.0010
	OPC	0.1314 ± 0.0029	0.0598 ± 0.0020	0.0529 ± 0.0016	0.0483 ± 0.0014	0.1170 ± 0.0051
	InvFusion-style	0.1710 ± 0.0034	0.0912 ± 0.0026	0.0905 ± 0.0023	0.0396 ± 0.0013	0.0755 ± 0.0018
	ΠGDM	0.1281 ± 0.0030	0.0623 ± 0.0022	0.0556 ± 0.0018	0.0339 ± 0.0011	0.0940 ± 0.0048
	DDNM	0.1592 ± 0.0036	0.0737 ± 0.0031	0.0766 ± 0.0022	0.0273 ± 0.0009	0.0297 ± 0.0010
	DAPS	0.2658 ± 0.0057	0.0900 ± 0.0034	0.1519 ± 0.0041	0.0771 ± 0.0021	0.0732 ± 0.0020
	MPGD	0.2496 ± 0.0048	0.1229 ± 0.0043	0.1755 ± 0.0044	0.1770 ± 0.0039	0.0746 ± 0.0020
	DPS†	0.2827 ± 0.0044	0.1129 ± 0.0035	0.3313 ± 0.0054	0.3610 ± 0.0051	0.2323 ± 0.0047
FID↓	EPS (ours)	47.9 ± 2.7	22.6 ± 1.5	30.4 ± 2.2	18.8 ± 1.2	21.0 ± 1.3
	OPC	49.7 ± 2.7	25.0 ± 1.6	31.8 ± 2.1	36.7 ± 2.1	65.5 ± 3.8
	InvFusion-style	75.5 ± 3.4	37.7 ± 2.1	54.5 ± 3.0	31.6 ± 1.7	54.1 ± 2.8
	ΠGDM	49.3 ± 2.8	25.1 ± 1.7	33.5 ± 2.4	25.5 ± 1.5	53.1 ± 3.3
	DDNM	61.0 ± 3.1	27.1 ± 1.9	43.7 ± 2.9	19.2 ± 1.2	20.3 ± 1.2
	DAPS	108.9 ± 4.8	37.7 ± 2.4	81.7 ± 4.5	67.4 ± 2.9	51.9 ± 2.8
	MPGD	119.0 ± 4.8	47.9 ± 2.5	100.9 ± 4.6	82.0 ± 4.0	75.5 ± 3.2
	DPS†	98.1 ± 5.0	44.7 ± 2.1	178.9 ± 5.6	75.4 ± 4.0	83.4 ± 4.4
CRPS_{Inc} ↓	EPS (ours)	5.28 ± 0.13	3.37 ± 0.11	3.74 ± 0.11	2.95 ± 0.07	3.12 ± 0.08
	OPC	5.35 ± 0.12	3.75 ± 0.11	3.95 ± 0.11	4.80 ± 0.12	6.32 ± 0.17
	InvFusion-style	7.07 ± 0.15	4.68 ± 0.13	5.56 ± 0.14	4.10 ± 0.10	5.62 ± 0.15
	ΠGDM	5.48 ± 0.14	3.78 ± 0.12	4.12 ± 0.12	3.86 ± 0.09	5.49 ± 0.15
	DDNM	6.63 ± 0.15	4.03 ± 0.15	4.87 ± 0.14	3.15 ± 0.07	3.23 ± 0.08
	DAPS	9.35 ± 0.20	5.00 ± 0.16	7.52 ± 0.19	6.89 ± 0.14	5.92 ± 0.15
	MPGD	9.29 ± 0.16	6.00 ± 0.16	8.05 ± 0.17	7.84 ± 0.17	7.62 ± 0.15
	DPS†	8.68 ± 0.17	5.45 ± 0.13	10.51 ± 0.15	8.65 ± 0.16	8.04 ± 0.17
MMD^2 ↓	EPS (ours)	-0.0033 ± 0.0008	-0.0055 ± 0.0004	-0.0050 ± 0.0004	-0.0057 ± 0.0003	-0.0056 ± 0.0003
	OPC	0.0001 ± 0.0010	-0.0030 ± 0.0007	-0.0030 ± 0.0006	0.0093 ± 0.0014	0.0276 ± 0.0028
	InvFusion-style	0.0222 ± 0.0031	-0.0010 ± 0.0009	0.0064 ± 0.0014	-0.0022 ± 0.0006	0.0063 ± 0.0015
	ΠGDM	-0.0012 ± 0.0011	-0.0043 ± 0.0006	-0.0038 ± 0.0006	-0.0008 ± 0.0006	0.0124 ± 0.0018
	DDNM	0.0190 ± 0.0023	-0.0043 ± 0.0007	0.0005 ± 0.0011	-0.0054 ± 0.0003	-0.0053 ± 0.0003
	DAPS	0.1154 ± 0.0067	0.0065 ± 0.0015	0.0495 ± 0.0038	0.0500 ± 0.0037	0.0185 ± 0.0019
	MPGD	0.1048 ± 0.0046	0.0092 ± 0.0017	0.0761 ± 0.0045	0.0592 ± 0.0047	0.0516 ± 0.0030
	DPS†	0.0607 ± 0.0045	0.0070 ± 0.0015	0.1741 ± 0.0056	0.0237 ± 0.0039	0.0399 ± 0.0039

D.16 Sensitivity to the Measurement Noise Level

The noise level σ_y enters EPS at test time only through the closed-form pivot (12), so it is a scalar to set rather than a quantity the network must infer. Table 15 evaluates the models trained at $\sigma_y = 0.05$ in two ways. In the *matched* setting the pivot uses the true σ_y. Below the training level, the sampler stays stable down to $\sigma_y = 0.001$ on every task and improves as the problem becomes easier, up to 46.45 dB on Gaussian deblurring (against 30.86 dB at $\sigma_y = 0.05$). In the *blind* setting the pivot keeps assuming 0.05. Below the training level, the blind sampler still improves on its result at 0.05, but less than the matched one (Gaussian deblurring at $\sigma_y = 0.01$: 32.38 against 36.98 dB), so setting the pivot to the true noise level is worthwhile when it is known. Above the training level, both settings degrade because the model never saw such noise during training. At $\sigma_y = 0.1$ the loss is 1.5 to 2.7 dB on every task except Gaussian deblurring, the most nearly invertible one, which drops by more than 10 dB.

Amortizing over σ_y flattens this curve. One model per task trained with $\sigma_y \sim \mathcal{U}[0.01, 0.2]$ and evaluated with the pivot at the true σ_y loses 1.0 to 5.6 dB from $\sigma_y = 0.05$ to 0.2, against 5.5 to 15.9 dB for the models trained at 0.05; at $\sigma_y = 0.2$ it is ahead on four of the five tasks (Gaussian deblurring: 24.43 against 14.95 dB), and it extrapolates below its training range (47.12 dB at $\sigma_y = 0.001$). These amortized models were trained with a shorter budget than the paper models, which lowers their values at small σ_y, so Table 16 compares them with models trained at $\sigma_y = 0.05$ under the same budget. At $\sigma_y = 0.05$, amortization costs 0.06 to 0.16 dB on four tasks and 1.27 dB on box inpainting.

Table 15: **Sensitivity to the measurement noise level** (FFHQ-64, 100 NFE, 100 images $\times$ 10 seeds, per-sample PSNR↑ / LPIPS↓). *Matched*: the model trained at $\sigma_y = 0.05$ is evaluated with the pivot set to the true σ_y. *Blind*: the pivot keeps assuming 0.05 while the data carry the true σ_y. *Amortized*: one model per task trained with $\sigma_y \sim \mathcal{U}[0.01, 0.2]$ (a shorter budget than the paper models, see Table 16), pivot set to the true σ_y; * below the training range.

Matched: true σ_y in the pivot, model trained at $\sigma_y = 0.05$							
Task	0.001	0.005	0.01	0.025	0.05	0.1	0.2
SR 4×	22.72 / 0.096	22.70 / 0.099	22.66 / 0.103	22.48 / 0.111	21.99 / 0.126	19.65 / 0.234	14.67 / 0.580
Inpaint (box)	24.45 / 0.047	24.43 / 0.047	24.42 / 0.049	24.36 / 0.052	24.19 / 0.057	22.31 / 0.142	18.30 / 0.314
Inpaint (random)	26.87 / 0.038	26.85 / 0.040	26.81 / 0.042	26.65 / 0.046	26.17 / 0.053	23.64 / 0.135	17.56 / 0.420
Gaussian deblur	46.45 / 0.001	39.42 / 0.006	36.98 / 0.011	33.98 / 0.019	30.86 / 0.027	19.57 / 0.352	14.95 / 0.588
Motion deblur	49.47 / 0.001	40.38 / 0.004	36.78 / 0.008	32.89 / 0.018	30.29 / 0.031	27.61 / 0.055	24.83 / 0.099
Blind: pivot fixed at $\sigma_y = 0.05$; columns give the true σ_y							
Task	0.01	0.025	0.05	0.075	0.1	0.2	
SR 4×	22.44 / 0.116	22.34 / 0.117	21.99 / 0.126	21.20 / 0.157	19.73 / 0.236	14.89 / 0.572	
Inpaint (box)	24.27 / 0.055	24.27 / 0.055	24.19 / 0.057	23.71 / 0.082	22.67 / 0.125	19.16 / 0.266	
Inpaint (random)	26.61 / 0.048	26.53 / 0.049	26.17 / 0.053	25.36 / 0.078	23.84 / 0.128	18.00 / 0.393	
Gaussian deblur	32.38 / 0.025	32.09 / 0.025	30.86 / 0.027	26.10 / 0.118	20.06 / 0.338	14.67 / 0.582	
Motion deblur	30.75 / 0.028	30.71 / 0.028	30.29 / 0.031	29.56 / 0.037	28.43 / 0.052	19.41 / 0.325	
Amortized over $\sigma_y \in [0.01, 0.2]$: true σ_y in the pivot							
Task	0.001*	0.005*	0.01	0.025	0.05	0.1	0.2
SR 4×	21.53 / 0.130	21.51 / 0.133	21.48 / 0.137	21.34 / 0.147	21.00 / 0.165	20.25 / 0.197	18.91 / 0.253
Inpaint (box)	21.99 / 0.079	21.98 / 0.079	21.97 / 0.080	21.93 / 0.083	21.80 / 0.091	21.50 / 0.110	20.76 / 0.151
Inpaint (random)	24.62 / 0.066	24.61 / 0.067	24.59 / 0.068	24.47 / 0.074	24.15 / 0.087	23.29 / 0.119	21.57 / 0.178
Gaussian deblur	47.12 / 0.001	39.76 / 0.003	36.63 / 0.006	32.77 / 0.018	30.04 / 0.035	27.32 / 0.068	24.43 / 0.130
Motion deblur	48.95 / 0.001	39.47 / 0.005	35.75 / 0.012	31.57 / 0.026	28.86 / 0.046	26.42 / 0.077	24.04 / 0.127

Table 16: **Amortizing over σ_y at a matched budget.** Both models are retrained from the same initialization for the same number of steps (about 9k) and scored at $\sigma_y = 0.05$ (100 images $\times$ 5 seeds, per-sample PSNR with the standard error over images); the shorter budget puts both 0.7 to 2.4 dB below the paper models of Table 8.

	Trained at $\sigma_y = 0.05$			Amortized, $\sigma_y \sim \mathcal{U}[0.01, 0.2]$		
Task	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
SR 4×	21.03 ± 0.16	0.6579	0.1675	20.97 ± 0.16	0.6573	0.1666
Inpaint (box)	23.11 ± 0.22	0.8151	0.0717	21.84 ± 0.19	0.7779	0.0913
Inpaint (random)	24.33 ± 0.20	0.8418	0.0836	24.16 ± 0.20	0.8369	0.0870
Gaussian deblur	30.11 ± 0.15	0.9312	0.0350	30.02 ± 0.15	0.9297	0.0358
Motion deblur	29.01 ± 0.15	0.9154	0.0450	28.85 ± 0.15	0.9129	0.0462

D.17 Human Preference Study

To complement the automatic metrics on the distortion and perception trade-off, we ran a human preference study in two settings, ImageNet-64 $4\times$ super-resolution and FFHQ-64 random inpainting, with 76 raters and 1,872 judgments (936 per setting) over 40 evaluation images per setting. In each judgment a rater picks one of the eight methods' reconstructions, so the shares in Table 17 sum to 100% in each setting, and a uniform choice would give 12.5%. EPS is chosen most often in both settings, in about a third of all judgments (32.1% and 33.2%) against at most 19.6% for any other method, and its lead over every method is significant at the 95% level.

Table 17: **Human preference study.** 76 raters, 1,872 judgments (936 per setting) over 40 images per setting. *Share*: percentage of judgments choosing each method (the shares sum to 100% in each setting; a uniform choice gives 12.5%), with a 95% confidence interval for EPS. *EPS − method*: difference in share, in percentage points, with its 95% interval. Intervals use 5,000 bootstrap resamples over images; every difference is significant at the 95% level.

	ImageNet-64, SR 4×		FFHQ-64, random inpainting	
Method	Share (%)	EPS − method	Share (%)	EPS − method
EPS (ours)	32.1 [27.9, 36.5]		33.2 [29.7, 36.8]	
ΠGDM	19.6	+12.5 [6.1, 18.7]	11.9	+21.4 [16.6, 26.1]
Palette	17.8	+14.2 [8.0, 20.5]	14.9	+18.4 [14.1, 22.7]
DDRM	12.6	+19.4 [13.2, 25.5]	14.9	+18.4 [12.7, 23.9]
MPGD	6.5	+25.5 [21.1, 30.1]	4.3	+29.0 [25.4, 32.4]
DDNM	5.9	+26.2 [21.5, 31.0]	12.6	+20.6 [15.7, 25.3]
DPS	3.6	+28.4 [24.0, 33.0]	3.1	+30.1 [26.4, 33.8]
DAPS	1.9	+30.1 [25.8, 34.5]	5.2	+28.0 [24.0, 32.0]

Protocol. Each screen showed the measurement (the 16×16 low-resolution input, or the masked image) above the eight reconstructions of the same image, one per method, numbered 1 to 8 in a 2×4 grid; the ground truth and the method names were not shown. The question on each screen was "Which image is the most realistic high-resolution version of the small image?" for super-resolution and "Which image is the most realistic completion of the masked image?" for inpainting, and raters answered by clicking an image or pressing its number, after which the next screen loaded. Before the first screen, the instructions (Figure 15) asked raters to weigh two criteria equally: whether a reconstruction looks like a real photograph, with natural detail and texture rather than blur, smearing, speckle, or artificial smoothness, and whether it plausibly shows the same scene as the input, since sharpness alone is not the goal; they suggested about 9 seconds per screen. A session had 24 screens, 12 per setting in two blocks, and showed the 64×64 images magnified 2 to 5 times (128 to 320 pixels), depending on the window size.

Statistics. Confidence intervals use 5,000 bootstrap resamples over the 40 images of each setting, keeping all judgments of a resampled image, so that the correlation between judgments of the same image is accounted for. A difference between EPS and another method is significant when the 95% bootstrap interval of the difference in shares excludes zero. Each setting compares EPS with seven methods, 14 comparisons in total, and every difference stays significant even after a Bonferroni correction for these 14 comparisons.

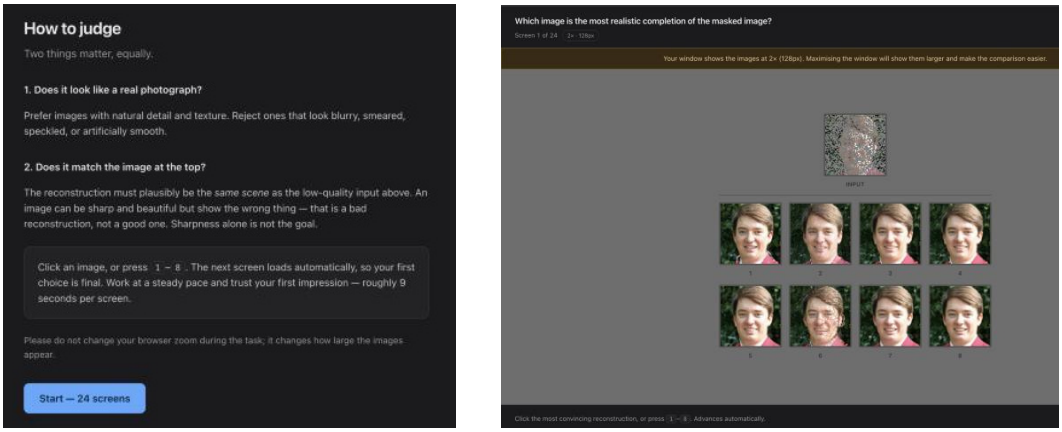

Figure 15: **Human-study interface.** Left: the instructions shown before the first screen. Right: a screen for FFHQ-64 random inpainting, with the measurement on top and the eight reconstructions, numbered 1 to 8, below.

E Exactness and Posterior Calibration

"Exact" in EPS refers to two statements that involve no approximation: the identity of Theorem 1, which writes the posterior score through the anisotropic denoiser at the pivot, and the fact that the population minimizer of (16) is that denoiser. A trained network approximates this minimizer under finite capacity, data, and optimization, as any learned denoiser does. This appendix checks both parts on a problem where the true posterior is known (Appendix E.1), and then checks the calibration of the learned samplers on images (Appendix E.2).

E.1 Gaussian-Mixture Benchmark

We use a prior with $K = 5$ Gaussian components in $d = 64$ dimensions and a random operator $A \in \mathbb{R}^{24\times 64}$ with $\sigma_y = 0.05$. The component means differ only inside the nullspace of A, so several components explain the same measurement, and the true posterior is a reweighted mixture available in closed form. For this prior, the unconditional denoiser D_t and the anisotropic denoiser $D_{\Sigma_\star(t)}$ are also available in closed form (the oracle denoisers), and the two sides of Theorem 1 agree to a maximum absolute error of 1.1×10^{-12}, the floating-point floor. Computing $\mathbb{E}[x_0|x_t, y]$ directly by Gaussian conditioning, for the EDM interpolant at noise levels from 0.01 to 10 and for a flow interpolant ($\alpha_t = 1 - t$, $\beta_t = t$) at t from 0.05 to 0.95, gives the same agreement (relative error below 5×10^{-11}). We then compare EPS with DPS, ΠGDM, and OPC under the same EDM sampler ($\sigma \in [0.002, 20]$), first with the oracle denoisers and then with learned ones: the EPS denoiser and the unconditional denoiser used by the baselines are MLPs of the same size (three layers of width 512 with SiLU activations), each trained with Adam (learning rate 10^{-3}, batch size 512) for 1000 steps. This configuration was selected from a small sweep (widths 256 to 1024, 1000 to 16000 steps) by the distance of the learned EPS sampler at 20 NFE, and the unconditional denoiser reuses it. We draw 2000 samples from each sampler and from the analytic posterior and report the sliced-Wasserstein distance between them; two independent sets of analytic draws are 0.063 apart, which is the Monte Carlo floor.

Table 18 shows that EPS is closest to the true posterior at every number of steps, both with oracle denoisers (0.18 to 0.23, against 0.27 to 0.33 for the baselines) and with learned ones (0.30 to 0.37, against 0.41 to 0.53). Learning adds an error of similar size to every method, so the ordering is preserved, and the bottom of the table shows the same ordering for the recovered mode weights, the posterior mean, and the posterior covariance. ΠGDM and OPC differ only in a scalar variance that has almost no effect at this small σ_y, so the benchmark does not separate them; with oracle denoisers both trail EPS at every budget, so neither isotropic choice closes the gap to the exact anisotropic query at the pivot.

E.2 Calibration of the Posterior Samples

On images the true posterior is unknown, so we test two necessary conditions for calibration.

Rank calibration. If the samples are draws from the true posterior, the ground truth is exchangeable with them. For each of the 100 FFHQ-64 test images, we project the 10 posterior samples and the ground truth onto 100 random unit directions, drawn inside the nullspace of A or inside its range, and record the rank of the ground truth among the 11 projected values; under calibration the rank is uniform. Table 19 reports the χ^2 statistic of the pooled rank histogram against the uniform distribution (10 degrees of freedom, 95% critical value 18.3); Gaussian deblurring has no nullspace at our kernel, so only its range test is defined. EPS passes the test in 6 of the 9 cells (both subspaces of both inpainting tasks and the range of both deblurring tasks) and has the lowest statistic in 8 of them, while no baseline passes in any cell (statistics from 27.7 to 19570). The two EPS failures on super-resolution (47 in the range and 45 in the nullspace) are still six to seven times below the best baseline in the same cells (275 and 305); the motion-deblurring nullspace is the one cell where EPS (710) is behind ΠGDM (116) and DDNM (394).

Measurement residual. An exact posterior sample $\hat{x}$ explains the measurement only up to the noise, so the residual $y - A\hat{x}$ is distributed like the measurement noise and its RMS should equal σ_y. A smaller RMS means that the sampler also fits the noise in y. For the super-resolution models of

Table 18: **Distance to the analytic posterior on a Gaussian-mixture benchmark** ($d = 64$, $m = 24$, $K = 5$ components whose means differ only in the nullspace of A, $\sigma_y = 0.05$). Sliced-Wasserstein distance between 2000 sampler draws and 2000 analytic-posterior draws (lower is better; the Monte Carlo floor of two independent analytic draws is 0.063). With the oracle denoisers, EPS uses the exact $D_{\Sigma_\star}$ and the baselines the exact unconditional D_t; with learned denoisers, both are MLPs of the same capacity and training budget. ΠGDM and OPC differ only in a scalar prior variance, which has almost no effect on their updates at this small σ_y, so their distances agree to within 10^{-4}.

	Oracle denoiser				Learned denoiser			
NFE	EPS	DPS	ΠGDM	OPC	EPS	DPS	ΠGDM	OPC
5	**0.226**	0.333	0.334	0.334	**0.300**	0.479	0.462	0.462
10	**0.197**	0.290	0.292	0.292	**0.326**	0.437	0.411	0.411
20	**0.184**	0.277	0.280	0.280	**0.346**	0.455	0.422	0.422
50	**0.188**	0.275	0.275	0.275	**0.362**	0.494	0.440	0.440
100	**0.188**	0.278	0.275	0.275	**0.367**	0.531	0.448	0.448

Method	Denoiser	Sliced-W↓	Mode TV↓	Mean error↓	Covariance error↓
EPS	oracle	0.188	0.068	1.81	0.127
EPS	learned	0.367	0.077	2.77	0.225
DPS	oracle	0.278	0.102	2.75	0.196
DPS	learned	0.531	0.117	3.61	0.380
ΠGDM	oracle	0.275	0.100	2.72	0.195
ΠGDM	learned	0.448	0.081	3.00	0.302
OPC	oracle	0.275	0.100	2.72	0.195
OPC	learned	0.448	0.081	3.01	0.302

Bottom: all four distances at NFE=100: sliced-Wasserstein, total variation between the empirical and the analytic mode weights, ℓ_2 error of the mean, and relative Frobenius error of the covariance.

Table 19: **Rank calibration of the posterior samples** (FFHQ-64, 100 images, 10 samples each, 100 NFE). The 10 samples and the ground truth are projected onto 100 random unit directions inside the nullspace of A (or inside its range), and the rank of the truth among the samples is recorded; a calibrated posterior gives a flat rank histogram. Entries are χ^2 statistics against uniform with 10 degrees of freedom (95% critical value 18.3; lower is better). Gaussian deblurring has no nullspace at this kernel, so its nullspace entry is undefined.

Subspace	Method	SR 4×	Inpaint (box)	Inpaint (random)	Gaussian deblur	Motion deblur
Nullspace	EPS (ours)	45.4	7.8	13.6	n/a	710.3
	ΠGDM	305.1	108.9	358.0	n/a	116.2
	DDNM	2081.2	408.5	347.7	n/a	394.3
	DPS	582.0	237.1	244.8	n/a	1467.9
	DAPS	1944.2	1056.7	1464.0	n/a	1221.3
	MPGD	4689.5	1215.8	384.5	n/a	7079.0
Range	EPS (ours)	46.8	16.1	7.7	9.5	5.4
	ΠGDM	6271.9	360.1	411.8	1724.4	828.2
	DDNM	19569.6	293.0	332.9	572.0	441.9
	DPS	275.2	659.3	323.4	161.1	554.7
	DAPS	1062.1	707.5	1631.7	27.7	1097.2
	MPGD	11179.8	2006.6	455.8	2623.3	8602.2

Appendix F.2 at $\sigma_y = 0.05$, the RMS of the block-averaged samples minus y is 0.053 at both 512 and 1024 (Table 24).

F High-Resolution Experiments

F.1 Controlled Comparison at 1024×1024

Setup. All methods share one pretrained prior, the MiniT2I-B/16 text-to-image rectified-flow model [53], which operates in pixel space on 16×16 patches. For each task, EPS fine-tunes this model on FFHQ-1024 training images, with the pivot $\mu_\star$ and the measurement as additional input channels, for 400k training images (batch size 4 on two GPUs, learning rate 10^{-4}); the Gaussian deblurring model sees the same 400k images at batch size 8 on four GPUs, in half the optimizer steps. This fine-tuning pairs the images with unrelated captions, so these models do not use the text input, and every method samples with the same fixed prompt. The baselines run zero-shot on the same pretrained model: FlowDPS [54], FlowChef [55], ΠGDM [20], OPC [52], and PSLD [56]. Every method uses 28 sampling steps, classifier-free guidance 2.5, and $\sigma_y = 0.05$, on 100 held-out FFHQ-1024 faces with 10 seeds each. Super-resolution averages pixel blocks, so the $64\times$ task observes a 16×16 image; box inpainting hides a box covering 40% of the image; and Gaussian deblurring uses a circular 61×61 kernel with standard deviation 3 pixels, the standard setting of DPS and FlowDPS. We tuned the hyperparameters of every baseline and report its best results. PSNR, SSIM, and LPIPS use the per-sample estimator of the 64×64 tables, and FID, CRPS, and MMD use all 1000 samples.

Results. Table 20 reports all metrics. EPS is best on every task and metric. FlowDPS and FlowChef follow the measurement but leave noise and texture artifacts at $16\times$, and they fail to produce a coherent face at $64\times$ or inside the box (Figure 16), which shows in FIDs three to four times higher than those of EPS. The gradient-guided solvers ΠGDM, OPC, and PSLD, which backpropagate the measurement residual through the network, do not constrain this prior and stay below 14 dB on every task. On deblurring, paired over images, EPS is 2.49 dB above the strongest baseline, FlowDPS, and better on 99 of the 100 faces; the mean of its 10 samples reaches 31.93 dB (Table 21). Figure 17 contrasts the one-step posterior-mean estimate of Observation 4 with posterior samples. The one-step output is smooth, and the samples add the detail that the measurement leaves open while agreeing on what it determines.

Table 20: **All metrics at 1024×1024.** FFHQ-1024, 100 held-out faces $\times$ 10 seeds, per-sample PSNR, SSIM and LPIPS with the standard error over images; FID, CRPS (Inception) and MMD^2 over all 1000 samples. Every method uses the same pretrained MiniT2I-B/16 flow prior; EPS is fine-tuned from it and the baselines run zero-shot on it.

Task	Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	CRPS_{Inc} ↓	MMD^2 ↓
SR $16\times$	EPS (ours)	25.31 ± 0.21	0.7689 ± 0.0061	0.3609 ± 0.0029	42.7	4.526	-2.89e-03
	FlowDPS	23.84 ± 0.18	0.6540 ± 0.0059	0.5277 ± 0.0027	132.9	10.106	8.23e-02
	FlowChef	22.57 ± 0.14	0.5103 ± 0.0037	0.5815 ± 0.0021	137.6	10.429	8.98e-02
	ΠGDM	9.89 ± 0.12	0.1382 ± 0.0062	0.7536 ± 0.0047	365.9	15.525	3.43e-01
	OPC	9.29 ± 0.11	0.0867 ± 0.0050	0.7506 ± 0.0036	397.2	14.686	2.67e-01
	PSLD	12.64 ± 0.14	0.3931 ± 0.0080	0.7034 ± 0.0045	272.9	12.822	2.30e-01
SR $64\times$	EPS (ours)	19.56 ± 0.16	0.5096 ± 0.0088	0.4792 ± 0.0036	91.1	7.585	9.55e-03
	FlowDPS	18.93 ± 0.14	0.4260 ± 0.0066	0.5781 ± 0.0026	345.4	14.209	2.40e-01
	FlowChef	17.66 ± 0.13	0.2954 ± 0.0053	0.6266 ± 0.0022	318.0	14.093	2.08e-01
	ΠGDM	8.88 ± 0.08	0.0761 ± 0.0040	0.7587 ± 0.0034	427.1	15.178	2.98e-01
	OPC	9.95 ± 0.14	0.1291 ± 0.0050	0.7298 ± 0.0036	415.5	14.162	2.27e-01
	PSLD	12.36 ± 0.13	0.3841 ± 0.0079	0.7080 ± 0.0046	270.6	12.739	2.35e-01
Inpaint (box, 40%)	EPS (ours)	18.62 ± 0.20	0.7396 ± 0.0044	0.2775 ± 0.0027	56.1	5.420	2.72e-04
	FlowDPS	15.73 ± 0.15	0.6750 ± 0.0040	0.4114 ± 0.0034	209.9	11.328	1.25e-01
	FlowChef	15.40 ± 0.15	0.6603 ± 0.0036	0.3893 ± 0.0034	216.0	11.019	1.16e-01
	ΠGDM	9.29 ± 0.11	0.0876 ± 0.0044	0.7783 ± 0.0037	370.4	15.630	3.90e-01
	OPC	8.98 ± 0.08	0.0684 ± 0.0041	0.7892 ± 0.0033	363.6	15.103	3.69e-01
	PSLD	12.95 ± 0.17	0.3504 ± 0.0091	0.6724 ± 0.0042	318.4	12.624	1.58e-01
Gaussian deblur	EPS (ours)	30.33 ± 0.19	0.9558 ± 0.0016	0.2400 ± 0.0025	10.7	2.500	-5.28e-03
	FlowDPS	27.83 ± 0.28	0.9020 ± 0.0057	0.3512 ± 0.0062	32.7	4.634	-3.18e-05
	FlowChef	27.80 ± 0.23	0.8867 ± 0.0051	0.3698 ± 0.0047	39.6	5.183	2.01e-03
	ΠGDM	10.14 ± 0.09	0.1123 ± 0.0031	0.7620 ± 0.0041	368.1	15.378	3.69e-01
	OPC	9.20 ± 0.10	0.0700 ± 0.0032	0.7828 ± 0.0035	373.2	15.028	3.62e-01
	PSLD	13.86 ± 0.12	0.4102 ± 0.0095	0.6751 ± 0.0037	288.5	13.037	2.00e-01

Table 21: **Gaussian deblurring at** 1024×1024 (61×61 kernel, $\sigma = 3$ px). PSNR of a single sample (per-sample) and of the mean of the 10 samples (posterior mean).

Method	PSNR (per-sample)	PSNR (post. mean)
EPS (ours)	30.33	31.93
FlowDPS	27.83	29.56
FlowChef	27.80	29.15
ΠGDM	10.14	12.53
OPC	9.20	11.41
PSLD	13.86	14.40



Figure 16: **Controlled comparison at** 1024×1024**.** Two held-out FFHQ-1024 faces per task, seed-0 samples of the runs in Table 20; every method uses the same pretrained flow prior. FlowDPS and FlowChef follow the measurement but leave artifacts and do not complete the face at $64\times$ or inside the box; ΠGDM, OPC, and PSLD do not constrain the prior at this scale.



Figure 17: **Posterior mean and posterior samples at** 1024×1024**.** One EPS evaluation from pure noise gives the posterior-mean estimate of Observation 4 (third column), which is smooth; full sampling (100 steps, three seeds) adds detail that the measurement does not determine, such as skin texture, hair strands, and the exact shape of the eyes, while agreeing on what it does.

F.2 General-Purpose Models at 512×512 and 1024×1024

Beyond the controlled study, we trained four larger models that cover whole operator families on general images. Super-resolution and inpainting at 512 are fine-tuned from MiniT2I-L/16 (912M parameters) [53], and at 1024 from PixelDiT-1.3B [69]; both are pixel-space text-to-image models. The 1024 models train on about one million PD12M [70] images together with FFHQ, on four H100 GPUs; the 512 super-resolution model trains on PD12M, source photographs of UltraEdit [71], Open Images [72], and FFHQ, and the 512 inpainting model on the UltraEdit corpus, whose targets include AI-edited images. Every training image has a caption. FFHQ has none of its own, so its captions are composed from appearance attributes (hair, eyes, mouth, accessories, clothing) that Qwen2.5-VL [73] extracts from each face, and the inpainting regions are captioned with Qwen2.5-VL and Qwen3-VL [74]. One super-resolution model covers every factor from $2\times$ to $64\times$ (to $128\times$ at 1024), and one inpainting model covers free-form, box, and outpainting masks. At 1024, the inpainting masks also include face parts from a SegFormer [75] face parser trained on CelebAMask-HQ [76] and object silhouettes from COCO [77], LVIS [78], and Open Images. The network input is the closed-form pivot $\mu_\star$ together with the measurement (and x_t for the velocity-predicting 1024 backbone). The 1024 models are trained for 40k steps (super-resolution) and 80k steps (inpainting). All evaluations use held-out images that no model saw, 24 sampling steps, text guidance 1.8, $\sigma_y = 0.05$, and the image's own caption as the prompt.

Table 22 shows that a single model per resolution handles every factor, with PSNR and SSIM decreasing smoothly as the factor grows. At the largest factors ($32\times$ and above at 1024, $64\times$ at 512), the samples score slightly below nearest-neighbor upsampling in PSNR. A sharp sample from a broad posterior is penalized by PSNR relative to a blurry estimate, as in the perception-distortion trade-off [51]. Table 23 and Figure 18 report the first 64 held-out faces of each evaluation set, without selection.

Table 22: **One model per resolution covers every super-resolution factor.** Held-out evaluation images (FFHQ faces and general images that no model saw), 24 steps, text guidance 1.8, $\sigma_y = 0.05$, the image's own caption as the prompt. "Nearest" is the PSNR of nearest-neighbor upsampling of y, for reference. At the largest factors, sharp posterior samples score below this blurry estimate in PSNR, as expected.

Model	Metric	$2\times$	$4\times$	$8\times$	$16\times$	$32\times$	$64\times$	$128\times$
SR-512 (MiniT2I-L/16)	PSNR↑	32.59	28.53	25.22	22.24	19.44	16.58	
	Nearest	28.49	25.88	23.56	21.35	19.23	17.32	
	SSIM↑	0.876	0.776	0.660	0.564	0.494	0.438	
SR-1024 (PixelDiT-1.3B)	PSNR↑	33.28	29.30	26.12	23.26	20.19	17.34	14.88
	Nearest	28.96	26.69	24.67	22.65	20.57	18.55	16.68
	SSIM↑	0.878	0.779	0.675	0.588	0.507	0.434	0.374
	LPIPS↓	0.129	0.214	0.285	0.353	0.439	0.531	0.616

Table 23: **Held-out FFHQ faces at 512 and 1024.** The first 64 faces of each evaluation pack, in pack order (no selection), general-purpose models, 24 steps, text guidance 1.8, $\sigma_y = 0.05$, each face's own caption as the prompt; metrics of seed 0. The eye and mouth rows use medium holes on the same faces.

Resolution	Task	PSNR↑	SSIM↑	LPIPS↓
512	SR $4\times$	29.39	0.8900	0.1905
512	SR $8\times$	26.46	0.7962	0.2613
512	SR $16\times$	23.61	0.6860	0.3250
1024	SR $8\times$	28.29	0.8897	0.2880
1024	SR $16\times$	25.58	0.7919	0.3429
1024	SR $32\times$	22.60	0.6651	0.3954
512	Inpainting (pack masks)	31.37	0.8898	0.0917
1024	Inpainting (pack masks)	28.42	0.8548	0.1446
1024	Inpainting (eye region, $\approx13\%$)	25.41	0.9236	0.0638
1024	Inpainting (mouth region, $\approx11\%$)	23.60	0.9199	0.0789



Figure 18: **Held-out faces at 512 and 1024, without selection.** The first 16 faces of each evaluation set, shown as triplets (measurement, EPS sample, ground truth); seed 0, 24 steps, text guidance 1.8, each face's own caption as the prompt. The measurements are shown upsampled to the output size.

F.3 Text-Guided Editing

A measurement leaves part of the image undetermined: the content of a hole, or the detail above the super-resolution cut-off. The posterior over that part is broad, and any single sample is one arbitrary choice from it. Because the general-purpose models keep the text input of their backbone, the prompt is a handle on this choice. We fix y and the seed and change only the prompt, appending an attribute to the image's own caption (for example ", with bright blue eyes" or ", wearing dark sunglasses"), with text guidance 5. For super-resolution at $64\times$, where the measurement is a 16×16 image (Figure 19), eye color, freckles, and lipstick change, while the output still reproduces the measurement. The RMS of the block-averaged output minus y stays at the noise level for every prompt (Table 24). For inpainting (Figure 20), the prompt decides what fills the eye or mouth region (sunglasses, glasses, or closed eyes; a beard, a smile, or lipstick), and the observed pixels are kept exactly. The prompt cannot override what y determines. With prompts that contradict a 32×32 measurement (Figure 21, bottom), pose, layout, and colors stay fixed, and the prompts change only detail that the measurement leaves open, such as stubble, wrinkles, or thin glasses; "a smiling child" does not turn an adult into a child, and "dark sunglasses" adds at most thin clear glasses, since the measurement shows the eyes. Text therefore chooses among the reconstructions that y allows, which is how a posterior sampler should behave. The edits are local. The LPIPS between the outputs of different prompts is 0.2 to 0.9 times the LPIPS between two seeds of one prompt (Table 24), because a new seed changes every undetermined detail while an attribute prompt changes one. Edits on general images are less reliable than on faces (Figures 19 and 21, top).

Table 24: **Text-guided editing keeps the measurement.** The measurement and the seed are fixed and only the prompt changes (guidance 5). For super-resolution, the residual RMS of the block-averaged output against y stays at the noise level $\sigma_y = 0.05$ for the image's own caption and for the edits; for inpainting, the observed pixels are kept exactly. The last column is the LPIPS between outputs of different prompts divided by the LPIPS between two seeds of one prompt (first 4 faces and first 4 non-face images of the evaluation pack, no selection). The control row uses prompts that contradict a 32×32 measurement; their effect on the output is of the same size as that of the $64\times$ attribute edits, and the residual stays at σ_y.

Model and setting	Residual RMS (own / edits)	LPIPS ratio (prompts / seeds)	
		faces	objects
SR-512, $32\times$	0.0534 / 0.0515	0.26	0.54
SR-1024, $64\times$	0.0534 / 0.0539	0.19	0.32
Inpaint-512, eye/mouth/center holes	kept exactly	0.74	0.88
Inpaint-1024, eye/mouth/center holes	kept exactly	0.60	0.42
SR-1024, $32\times$, contradicting prompts (guidance 1.8)	0.0522 / 0.0518	0.22	0.22



Figure 19: **Text-guided super-resolution at** 1024×1024 **from a** 16×16 **measurement** ($64\times$)**.** The first four held-out faces and the first four held-out PD12M images of the evaluation set, without selection; same measurement and seed in every row, and only the prompt changes. Faces append "with bright blue eyes", "with dark brown eyes", "with freckles across the cheeks", or "wearing bright red lipstick" to the image's own caption; other images append "with a rough weathered texture", "with a smooth polished surface", or "with intricate fine patterns". Identity and layout are fixed by the measurement, while the edits change detail that it leaves open.



Figure 20: **Text-guided inpainting at** 512×512**.** The first four held-out faces of the evaluation set, without selection, with a hole over the eyes (top) or the mouth (bottom); same measurement and seed in every row, and only the prompt changes. The appended attributes are "wearing dark sunglasses", "wearing round gold-rimmed glasses", and "with eyes closed" for the eyes, and "with a thick full beard", "smiling broadly with teeth showing", and "wearing bright red lipstick" for the mouth. The observed pixels are kept exactly.

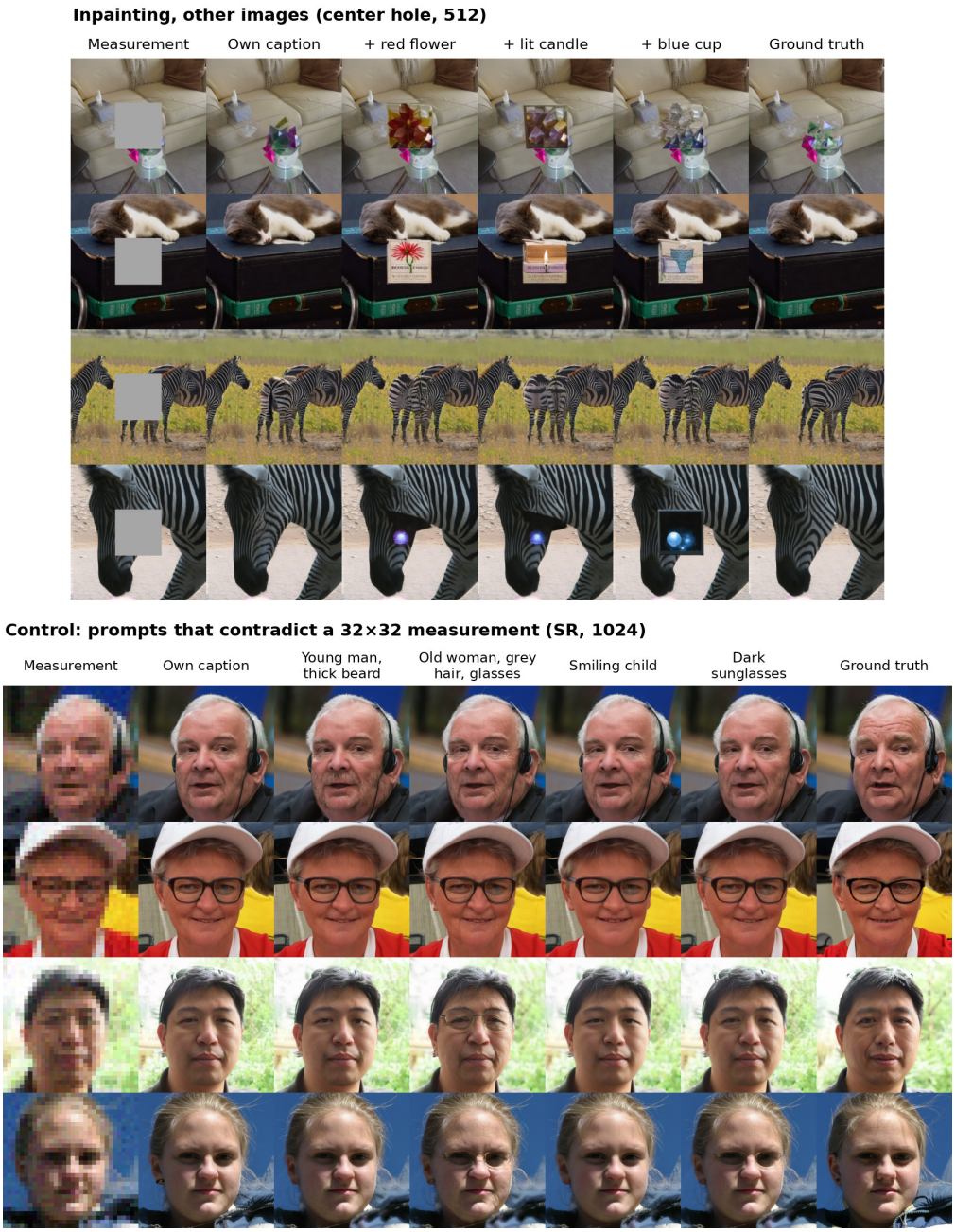

Figure 21: **Editing on general images, and a control.** Top: inpainting at 512×512 with a center hole on the first four held-out non-face images, appending "with a small red flower in the middle", "with a lit candle in the middle", or "with a blue ceramic cup in the middle"; the edits are less reliable than on faces. Bottom: super-resolution at 1024×1024 from a 32×32 measurement with prompts that contradict it (guidance 1.8). Pose, layout, and colors follow the measurement, and the prompts change only detail that it leaves open, such as stubble, wrinkles, or thin glasses.

G Practical Considerations

What the training buys. EPS pays once, at training time, for a model per operator family. In return, inference needs no gradient through the network, no projection, and no per-task guidance strength (the sampling-based baselines are tuned per task, Table 25), and it plateaus at about 20 NFE (Appendix D.4). On ImageNet-64, EPS at 20 NFE takes about 0.38 s per image (one fifth of its 100-NFE time in Table 13), against 4.54 s for ΠGDM at 100 NFE, about 12 times less, and at a matched budget EPS is also more accurate (Figure 3). The training cost is paid once per operator family, while the inference savings accrue with every image.

When the cost pays off. We distinguish three regimes. (i) *Fixed or calibrated operators.* In many imaging systems the forward model is fixed by the acquisition or calibrated once and then reused for a very large number of measurements. The undersampling pattern in MRI is set by the protocol, the system geometry in CT by the scanner, the point-spread function in deconvolution microscopy and astronomy by calibration, and the sensing matrix of compressed-sensing hardware by design. A one-time fine-tune then amortizes immediately, and EPS provides fast inference without backpropagation or per-task tuning, together with samples that are better calibrated than those of the baselines (Appendix E.2), so that the posterior uncertainty is usable downstream. (ii) *Operators within a parametric family.* When the operator varies within a family, no retraining is needed, because the operator enters only through the closed-form pivot. A single model amortized over all five operator families is within 0.2 dB of the per-task models or better (Appendix D.5), the general-purpose models cover every super-resolution factor from $2\times$ to $128\times$ (Appendix F.2), and σ_y can be amortized at a small cost on four of the five tasks (Appendix D.16). A new family needs a short fine-tune, which is cheap because the objective preserves the input/output structure of pretraining. (iii) *Blind or unseen operators.* When the operator is unknown at test time, EPS does not apply directly, but neither do the zero-shot baselines, which also need A and σ_y at test time. This is a limitation of the known-operator problem class rather than of EPS specifically. Pairing EPS with operator and noise estimation is a natural direction, for which the noise-level study of Appendix D.16 is a first step. A mis-specified σ_y costs little on the ill-posed tasks but much on near-invertible deblurring, and amortizing over σ_y reduces this sensitivity.

Computing the pivot. For the structured operators of this paper the pivot solve is exact and closed form (element-wise for masks, block averaging for super-resolution, FFT for convolutions), and the full EPS sampler costs the same as the bare denoiser to within 1% (Table 13). For other operators, the matrix $\alpha_t^2/\beta_t^2\, I + \sigma_y^{-2} A^\top A$ is symmetric positive definite with smallest eigenvalue at least α_t^2/β_t^2, so conjugate gradients apply with only products by A and $A^\top$. Moreover, these matrices share the eigenvectors of $A^\top A$ for every t, so a single eigendecomposition of $A^\top A$ per operator serves every sampling step and every image, which is the access already assumed by solvers that use the pseudo-inverse $A^\dagger$, such as DDNM and ΠGDM. A very large dense operator with neither structure nor an affordable factorization remains a real cost, for EPS and for those solvers alike.

Numerical stability. For rank-deficient A, $\Sigma_\star(t)$ approaches a singular matrix as $\beta_t \to 0$, and for small σ_y the term $\sigma_y^{-2} A^\top A$ becomes large. Neither causes instability, because the pivot can be computed without small divisors. Multiplying the linear system by $\beta_t^2 \sigma_y^2$ gives

$$\mu_\star = \big(\alpha_t^2 \sigma_y^2\, I + \beta_t^2 A^\top A\big)^{-1} \big(\alpha_t \sigma_y^2\, x_t + \beta_t^2 A^\top y\big), \tag{63}$$

in which β_t and σ_y appear only in products. For a unit eigenvector v of $A^\top A$ with eigenvalue λ, the component $v^\top \mu_\star$ is a convex combination of the state estimate $v^\top x_t/\alpha_t$ and the data estimate $v^\top A^\top y/\lambda$, with weights $\alpha_t^2\sigma_y^2/(\alpha_t^2\sigma_y^2 + \beta_t^2\lambda)$ and $\beta_t^2\lambda/(\alpha_t^2\sigma_y^2 + \beta_t^2\lambda)$; along a direction that A does not measure ($\lambda = 0$), the pivot simply copies $v^\top x_t/\alpha_t$. The limit $\sigma_y \to 0$ is therefore well defined. The pivot reproduces the pseudo-inverse solution $A^\dagger y$ on the row space of A and keeps x_t/α_t on the nullspace. In our implementation the solve is an element-wise or FFT division with strictly positive denominators, and the model trained at $\sigma_y = 0.05$ remains stable when evaluated with σ_y down to 10^{-3} (Table 15).

The resolution ladder. Together, the experiments trace EPS across resolutions and priors: controlled comparisons at 64×64 (Section 4.2), a model trained from scratch at 256×256 against

published baseline numbers (Appendix D.10), a controlled comparison on a shared pretrained flow prior at 1024×1024 (Section 4.3), and general-purpose models at 512 and 1024 that cover whole operator families on general images (Appendix F.2).

H Baseline Configurations

Table 25 reports per-task hyperparameters for the sampling-based and training-based baselines (DPS, DAPS, DDNM, ΠGDM, MPGD, Palette), including NFE, step size or guidance scale, noise schedule, projection or correction rule, and additional notes. Each baseline is tuned on a disjoint validation split, and the values reported are those used in the main and appendix tables. The additional 64×64 baselines of Appendix D.15 run at 100 NFE. OPC [52] uses the same pretrained denoiser as the other sampling-based methods, and the InvFusion-style model is trained as described there. The 1024×1024 baselines and their guidance strengths are described in Appendix F.1.

Table 25: **Baseline hyperparameter configurations.** Per-task hyperparameters for the sampling-based (DPS, DAPS, DDNM, ΠGDM, MPGD) and training-based (Palette) baselines. NFE is the number of denoiser forward passes per image (DAPS NFE=annealing×ode Euler steps). All samplers use the EDM VE noise schedule with $\sigma_{\max}=80$, $\sigma_{\min}=0.002$, $\rho=7$ unless noted. Observation noise is $\sigma_y=0.05$ for every cell. Identical settings are used on ImageNet-64 and FFHQ-64; only the underlying denoiser changes.

Task	Method	NFE	Step size / guidance	Noise schedule	Projection / correction	Notes
Random inpaint	DPS	250	$\zeta'=5.0$	EDM VE	likelihood grad, $\zeta'/\|r\|$	Euler, $S_{\text{churn}}=0$
Random inpaint	DAPS	500	mcmc 5×10^{-4}	EDM VE	nested ODE + Langevin	annealing=100, ode=5, mcmc=100, Euler
Random inpaint	DDNM	100	—	EDM VE	null-space $\lambda_t=\sigma_t^2/(\sigma_t^2+\sigma_y^2)$	DDIM-VE, $\eta=0.85$
Random inpaint	ΠGDM	100	$\zeta=1.0$	EDM VE	VJP guidance	DDIM-VE, $\eta=1.0$
Random inpaint	MPGD	100	$\zeta=20.0$	EDM VE	manifold projection	DDIM-VE, $\eta=0.85$
Random inpaint	Palette	100	—	EDM VE	—	training-based; per-task EPS backbone
Box inpaint	DPS	250	$\zeta'=10.0$	EDM VE	likelihood grad, $\zeta'/\|r\|$	Euler, $S_{\text{churn}}=0$
Box inpaint	DAPS	500	mcmc 5×10^{-4}	EDM VE	nested ODE + Langevin	annealing=100, ode=5, mcmc=100, Euler
Box inpaint	DDNM	100	—	EDM VE	null-space $\lambda_t=\sigma_t^2/(\sigma_t^2+\sigma_y^2)$	DDIM-VE, $\eta=0.85$
Box inpaint	ΠGDM	100	$\zeta=1.0$	EDM VE	VJP guidance	DDIM-VE, $\eta=1.0$
Box inpaint	MPGD	100	$\zeta=15.0$	EDM VE	manifold projection	DDIM-VE, $\eta=0.85$
Box inpaint	Palette	100	—	EDM VE	—	training-based; per-task EPS backbone
Super-res ($4\times$)	DPS	250	$\zeta'=10.0$	EDM VE	likelihood grad, $\zeta'/\|r\|$	Euler, $S_{\text{churn}}=0$
Super-res ($4\times$)	DAPS	500	mcmc 9×10^{-4}	EDM VE, $\sigma_{\max}=30$, $\sigma_{\min}=0.1$	nested ODE + Langevin	annealing=100, ode=5, mcmc=100, Euler
Super-res ($4\times$)	DDNM	100	—	EDM VE	null-space $\lambda_t=\sigma_t^2/(\sigma_t^2+\sigma_y^2)$	DDIM-VE, $\eta=0.85$, $A^\dagger$=nearest upsample
Super-res ($4\times$)	ΠGDM	100	$\zeta=1.0$	EDM VE	VJP guidance	DDIM-VE, $\eta=1.0$
Super-res ($4\times$)	MPGD	100	$\zeta=30.0$	EDM VE	manifold projection	DDIM-VE, $\eta=0.85$
Super-res ($4\times$)	Palette	100	—	EDM VE	—	training-based; per-task EPS backbone
Gaussian deblur	DPS	250	$\zeta'=1.5$	EDM VE	likelihood grad, $\zeta'/\|r\|$	Euler, $S_{\text{churn}}=0$
Gaussian deblur	DAPS	500	mcmc 9×10^{-4}	EDM VE, $\sigma_{\max}=30$, $\sigma_{\min}=0.1$	nested ODE + Langevin	annealing=100, ode=5, mcmc=100, Euler
Gaussian deblur	DDNM	100	—	EDM VE	per-freq Wiener correction	DDIM-VE, $\eta=0.85$, Wiener $\epsilon=10^{-3}$
Gaussian deblur	ΠGDM	100	$\zeta=1.0$	EDM VE	VJP guidance	DDIM-VE, $\eta=1.0$
Gaussian deblur	MPGD	100	$\zeta=5.0$	EDM VE	manifold projection	DDIM-VE, $\eta=0.85$
Gaussian deblur	Palette	100	—	EDM VE	—	training-based; per-task EPS backbone
Motion deblur	DPS	250	$\zeta'=2.5$	EDM VE	likelihood grad, $\zeta'/\|r\|$	Euler, $S_{\text{churn}}=0$
Motion deblur	DAPS	500	mcmc 5×10^{-4}	EDM VE	nested ODE + Langevin	annealing=100, ode=5, mcmc=100, Euler
Motion deblur	DDNM	100	—	EDM VE	per-freq Wiener correction	DDIM-VE, $\eta=0.85$, Wiener $\epsilon=10^{-3}$
Motion deblur	ΠGDM	100	$\zeta=1.0$	EDM VE	VJP guidance	DDIM-VE, $\eta=1.0$
Motion deblur	MPGD	100	$\zeta=9.0$	EDM VE	manifold projection	DDIM-VE, $\eta=0.85$
Motion deblur	Palette	100	—	EDM VE	—	training-based; per-task EPS backbone

I Metric Definitions

CRPS. For a scalar target z and a predictive distribution with CDF F, $\mathrm{CRPS}(F, z) = \int_{\mathbb{R}} (F(u) - \mathbf{1}\{u \geq z\})^2 \mathrm{d}u$. For our multivariate settings we report the average per-coordinate CRPS, in pixel space (CRPS-pixel) and in the Inception feature space used for FID (CRPS-inception), each averaged over the evaluation images and over the posterior samples drawn for each observation [47].

MMD. With a Gaussian kernel $k(u, v) = \exp(-\|u - v\|^2/(2\ell^2))$, we estimate $\mathrm{MMD}^2(P, Q)$ between the EPS sample distribution P and the empirical distribution Q of the evaluation ground truths using the unbiased U-statistic of Gretton et al. [48]. We report MMD-pixel (kernel in pixel space) and MMD-inception (kernel in Inception feature space). The bandwidth ℓ is chosen via the median heuristic on the combined sample.

Error bars. For per-image metrics (PSNR, SSIM, LPIPS) we report the standard error of the mean over the evaluation images, computed in the same pass as the mean. For set-level metrics (FID, CRPS, MMD) we resample the evaluation images with replacement 1000 times, keeping each image's posterior samples together, recompute the metric, and report the standard deviation of the replicates.

Sliced-Wasserstein distance. Between two sample sets of equal size, we project both onto 64 random unit directions and average the one-dimensional Wasserstein distances between the sorted projections.

Rank calibration. For each image and each random unit direction θ in the chosen subspace, we compute the rank $r \in \{1, \ldots, J{+}1\}$ of $\theta^\top x_0$ among $\theta^\top x_0$ and the projections $\theta^\top \hat{x}^{(j)}$ of the $J = 10$ posterior samples, pool the ranks into a histogram with $J{+}1$ bins, and report $\chi^2 = \sum_k (O_k - E_k)^2/E_k$ against the uniform expectation E_k.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and Section 1 state the contributions (closed-form posterior score for linear Gaussian inverse problems, the EPS training objective and sampler, the one-step high-noise posterior-mean limit, and empirical evaluation on five tasks across FFHQ and ImageNet together with a controlled 1024×1024 study), and these are the contributions established in Sections 3 and 4. The abstract and introduction also state that "exact" refers to the identity and the population minimizer, not to the trained network.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: It is discussed in the Section 6.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: Theorem and Proposition state their assumptions explicitly with full proofs in Appendix.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: Section 4.1 specifies the EDM backbone, the five linear operators, the observation noise level $\sigma_y = 0.05$, baselines, and metrics. Algorithm 1 gives the EPS training procedure, and Appendix B reports the operator construction, training schedule, optimizer, EMA, sampler, and per-baseline hyperparameters used to produce Tables.

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

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [Yes]

   Justification: For code and model checkpoints, see `https://github.com/abbasmammadov/EPS`, linked from the project page `https://abbasmammadov.github.io/EPS`. The paper and appendix give the full training and evaluation protocol.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We include all the details on data split, design choices and implementation details.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: Appendix D.15 (Table 14) reports error bars for EPS and every sampling-based FFHQ-64 method, plus OPC and the InvFusion-style baseline: the standard error of the mean over the 100 evaluation images for PSNR, SSIM, and LPIPS, and the standard deviation over 1000 image-level bootstrap resamples for FID, CRPS, and MMD (Appendix I). The 1024×1024 study reports standard errors over images (Table 20), and the human study reports 95% bootstrap confidence intervals for the share of EPS and for every difference between EPS and another method (Table 17).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Appendix B gives the GPUs and training time of the 64×64 models, Appendix D.12 the sampling cost, and Appendix F the GPUs and training budgets of the 512 and 1024 models.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We conform to the NeurIPS Code of Ethics.

   Guidelines:

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

10. **Broader impacts**

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

Answer: [Yes]

Justification: We discuss the broader impacts in the Appendix.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [N/A]

Justification: The paper has no such risks.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All assets are cited, and their licenses are: FFHQ (CC BY-NC-SA 4.0), ImageNet (research terms of access), PD12M (CDLA-Permissive-2.0; public-domain and CC0 images), UltraEdit (CC BY 4.0), Open Images (CC BY 4.0 annotations, CC BY 2.0 images), COCO and LVIS (CC BY 4.0 annotations, used only for mask shapes), the

EDM code and checkpoints (CC BY-NC-SA 4.0), MiniT2I (MIT) with its FLAN-T5 text encoder (Apache 2.0), PixelDiT (NVIDIA NSCL v1, non-commercial) with its Gemma-2 text encoder (Gemma terms of use), the captioning models Qwen2.5-VL and Qwen3-VL (Apache 2.0), and the face parser `jonathandinu/face-parsing`, a SegFormer trained on CelebAMask-HQ (non-commercial research use).

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: For code and model checkpoints and their documentation, see `https://github.com/abbasmammadov/EPS`. Our code is under the MIT license, and code and checkpoints derived from existing assets keep the licenses of those assets.

Guidelines:

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

14. **Crowdsourcing and research with human subjects**

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

Answer: [Yes]

Justification: We ran a simple human preference study. Appendix D.17 gives its details, including the question shown on every screen, and Figure 15 shows the instructions and a judgment screen.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [No]

    Justification: The study was a minimal-risk image-preference task (raters chose the most realistic of eight reconstructions of a 64×64 image) and recorded no personal information, so the paper does not discuss participant risks or IRB review.

    Guidelines:

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

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are used only for writing, editing, and formatting the paper, not to develop core method.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.