PERTURB-AND-SOLVE: EFFICIENT LEARNED-OPERATOR CONDITIONING FOR LATENT DIFFUSION INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent diffusion models serve as powerful priors for solving inverse problems in image restoration, such as deblurring, inpainting, and super-resolution. **Current methods have a trade-off between generality and efficiency.** Solvers that are restricted to a fixed set of degradation operators are fast and efficient. Methods that support arbitrary degradation operators are slow and require gradients through the diffusion network. To break this bottleneck, we introduce PASEO (**P**erturb-**A**nd-**S**olve for **E**fficient **O**perator conditioning), a method that uses a small ($\sim$1M parameters) learned network to degrade diffusion model predictions in latent space. **PASEO supports learned degradation operators without back-propagating through the diffusion network.** We efficiently sample reconstructions from an approximate posterior by combining the diffusion model's prediction with the observed image. We do this by adding noise and solving linear equations based on a local linear approximation of the learned network, without building or inverting large covariance matrices. Across super-resolution, deblurring, and inpainting on FFHQ and COCO, PASEO achieves strong perceptual quality while running up to $19\times$ faster and using up to 34% less peak memory than the tested baselines, with the same or fewer model evaluations.

1 INTRODUCTION

In image and signal processing, a signal of interest often cannot be observed directly and therefore must instead be recovered from indirect, incomplete or noisy measurements. This is known as an *inverse problem*, where we seek to recover an *unknown signal* $\mathbf{x}$ from an *indirect measurement* $\mathbf{y}$ of that signal. The same measurement can come from different signals: for example, different sharp images can produce similar blurred observations. We therefore need additional information about what the original signal could be. The Bayesian formulation expresses this knowledge through a prior $p(\mathbf{x})$ and uses a likelihood $p(\mathbf{y} \mid \mathbf{x})$ to describe how a candidate signal could produce the observed measurement. Together, they define the posterior distribution $p(\mathbf{x} \mid \mathbf{y}) \propto p(\mathbf{y} \mid \mathbf{x})p(\mathbf{x})$, which describes plausible signals given the measurement. Inverse problems with diffusion models aim at generating plausible reconstructions through posterior sampling. For image reconstruction, the prior must capture the structure and detail of natural images, motivating the use of expressive generative models.

Latent diffusion models (LDMs) (Rombach et al., 2022) provide effective image priors for inverse problems (Rout et al., 2023; Song et al., 2024; Raphaeli et al., 2025). They learn the structure and detail of natural images and generate images by working in a compact latent space. A pretrained autoencoder connects this space to images: its encoder maps an image to a latent representation, and its decoder maps that representation back to an image. Working in this smaller space reduces the cost of image generation.

For image reconstruction, generating realistic images is not enough. The reconstructed image must also be similar to the measurement $\mathbf{y}$. In the literature, this similarity is often called *data consistency*. Enforcing data consistency is harder with LDMs because the measurement $\mathbf{y}$ is in pixel space, while the model works in latent space. Many solvers therefore enforce data consistency by repeatedly running the encoder and decoder and back-propagating (Rout et al., 2023; Song et al., 2024; Rout et al., 2024; Chung et al., 2024b; Raphaeli et al., 2025), increasing runtime and memory.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 1: Visual comparison on COCO with SD3.5-medium prior.

Recent methods aim to improve reconstruction quality and reduce these costs. LD-SMC (Achituve et al., 2025) and DDSMC (Ekström Kelvinius et al., 2025) focus on more accurate posterior sampling, while SITCOM (Alkhouri et al., 2025), learnable linear extrapolation (Zhang et al., 2025b), SPIN (Shtanchaev et al., 2026), and CWGF (Spagnoletti et al., 2026) create faster samplers. Other approaches use the structure of linear measurements, as in DMPS (Meng & Kabashima, 2025) and pDDIM (Jiao et al., 2026); modify the sampling procedure, as in DAPS (Zhang et al., 2025a); or extend reconstruction to flow-based generative models, as in PNP-FLOW (Martin et al., 2025) and FLOWDPS (Kim et al., 2025). We build on the following two methods for reconstruction in latent space. DING (Moufad et al., 2026) samples exactly from approximate posterior transitions without back-propagating through the denoising network. Its latent-space formulation is limited to inpainting, where a mask separates observed and missing regions and enables a simple exact sampling update. Other degradations generally require a different update. On the other hand, SILO (Raphaeli et al., 2025) uses a small trained network to describe more general degradation processes in latent space. We denote this learned operator by $\mathbf{H}_\psi$, where ψ denotes its learned parameters. From a clean latent, the network predicts the encoding of the corresponding degraded image. This avoids repeatedly running the encoder and decoder during sampling, but SILO still back-propagates through the learned operator and the denoising network at every step to guide the reconstruction process. Our aim is to combine these benefits to support learned degradation operators in latent space while avoiding propagating gradients through the denoiser network.

PASEO efficiently updates its estimate toward an approximate posterior by first forming a local linear approximation of the learned network $\mathbf{H}_\psi$ around the current clean-latent prior estimate obtained from the diffusion step. To avoid forming the posterior covariance matrix and computing matrix inverses, we add independent Gaussian noise to the encoded measurement $\mathbf{y}_z$ and the mean of the clean-latent prior. We then solve the resulting linear equations using conjugate gradients (CG) to obtain a *corrected* clean latent. For a linear operator, this basic construction gives an exact posterior sample when the equations are solved exactly. With the nonlinear learned network, the result is an approximate posterior sample, because we linearize the network locally and solve the equations with only a few CG iterations. The solver requires derivatives only through $\mathbf{H}_\psi$, without back-propagating through the denoiser or decoder. The encoder and decoder are each used only once, to encode the degraded image $\mathbf{y}$ and decode the reconstruction. We make three ***contributions:***

1 A back-propagation-free sampler for learned operators. We turn each denoising step into a Gaussian inference problem in latent space and sample it by perturbing and solving. For a linear operator and an exact solve, this sample is exact; for a nonlinear one, we linearize the operator locally,

and the sample is approximate. This takes back-propagation-free latent updates beyond inpainting, to any degradation with a learned latent operator.

2 Fast and memory-efficient by design. The update needs only a few CG iterations through the small learned network (∼1M parameters). It never back-propagates through the denoiser, never forms a Jacobian or covariance matrix, and runs the autoencoder only twice per reconstruction.

3 Strong reconstruction quality at lower cost. Across five restoration tasks on FFHQ and COCO, with a diffusion prior (RV-v5.1) and a flow prior (SD3.5), PASEO ranks first or second in LPIPS on almost every task. With SD3.5, it runs up to 19× faster and uses up to 34% less peak memory than the baselines, with the same or fewer denoiser evaluations.

2 Background

Diffusion and flow-matching priors. Denoising diffusion models (Song & Ermon, 2019; Ho et al., 2020; Song et al., 2021b) construct a probability path $(p_t)_{t\in[0,1]}$ between the data distribution p_0 and a reference distribution $p_1 := \mathbf{N}(0, \mathbf{I}_{d_\mathbf{x}})$. A conditional path $p_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ induces the marginal $p_t(\mathbf{x}_t) = \int p_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)p_0(\mathbf{x}_0)\, d\mathbf{x}_0$. Although this density is generally intractable, samples are obtained by drawing $\mathbf{x}_0 \sim p_0$ followed by $\mathbf{x}_t \sim p_{t|0}(\cdot \mid \mathbf{x}_0)$. For the Gaussian conditional path $p_{t|0}(\cdot \mid \mathbf{x}_0) = \mathbf{N}(\alpha_t\mathbf{x}_0, \sigma_t^2\mathbf{I}_{d_\mathbf{x}})$, this becomes

$$\mathbf{x}_t = \alpha_t\mathbf{x}_0 + \sigma_t\mathbf{x}_1, \qquad \mathbf{x}_0 \sim p_0, \quad \mathbf{x}_1 \sim p_1, \quad \mathbf{x}_0 \perp\!\!\!\perp \mathbf{x}_1. \tag{2.1}$$

where schedules α_t, σ_t satisfy $(\alpha_0, \sigma_0) = (1, 0)$ and $(\alpha_1, \sigma_1) = (0, 1)$, with α_t non-increasing and σ_t non-decreasing. Variance-preserving diffusion uses $\alpha_t^2 + \sigma_t^2 = 1$ (Ho et al., 2020), whereas flow matching commonly uses $(\alpha_t, \sigma_t) = (1 - t, t)$ (Lipman et al., 2023).

The conditional path describes how a clean image produces a noisy state. Conversely, the distribution of the clean image given that noisy state follows from Bayes' rule: $p_{0|t}(\mathbf{x}_0 \mid \mathbf{x}_t) \propto p_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)p_0(\mathbf{x}_0)$. A learned denoiser approximates the mean of this reverse conditional distribution: $\hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t) \approx \hat{\mathbf{x}}_0(\mathbf{x}_t, t) := \mathbb{E}[\mathbf{x}_0 \mid \mathbf{x}_t] = \int \mathbf{x}_0 p_{0|t}(\mathbf{x}_0 \mid \mathbf{x}_t) d\mathbf{x}_0$.

To generate samples, we follow this probability path in reverse, starting from a noise sample $\mathbf{x}_1 \sim p_1$ and moving toward the data distribution p_0. Learned transitions approximate this reverse sampling process. DDIM (Song et al., 2021a) defines one such transition:

$$\mathbf{x}_t = \alpha_t\hat{\mathbf{x}}_0^\theta(\mathbf{x}_{t+1}, t+1) + (\sigma_t^2 - \eta_t^2)^{1/2}\hat{\mathbf{x}}_1^\theta(\mathbf{x}_{t+1}, t+1) + \eta_t\mathbf{w}, \qquad \mathbf{w} \sim \mathbf{N}(0, \mathbf{I}_{d_\mathbf{x}}). \tag{2.2}$$

Here $0 \le \eta_t \le \sigma_t$, and $\hat{\mathbf{x}}_1^\theta(\mathbf{x}_t, t) := (\mathbf{x}_t - \alpha_t\hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t))/\sigma_t$ estimates the noise coordinate. This follows from $\mathbb{E}[\mathbf{x}_1 \mid \mathbf{x}_t] = (\mathbf{x}_t - \alpha_t\hat{\mathbf{x}}_0(\mathbf{x}_t, t))/\sigma_t$; equivalently, the clean estimate is recovered from the noise one as $\hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t) = (\mathbf{x}_t - \sigma_t\hat{\mathbf{x}}_1^\theta(\mathbf{x}_t, t))/\alpha_t$, an identity used in equation 2.7 and again in Section 3. The update is deterministic when $\eta_t = 0$ and stochastic otherwise. We write $p_{t|t+1}^{\eta,\theta}(\mathbf{x}_t \mid \mathbf{x}_{t+1})$ for the law of equation 2.2.

Inverse problems. We consider the forward model

$$\mathbf{y} = \mathcal{A}(\mathbf{x}) + \mathbf{n}, \qquad \mathbf{n} \sim \mathbf{N}(0, \sigma_\mathbf{y}^2\mathbf{I}_{d_\mathbf{y}}). \tag{2.3}$$

where $\mathbf{x} \in \mathbb{R}^{d_\mathbf{x}}$ is the *unknown signal*, $\mathbf{y} \in \mathbb{R}^{d_\mathbf{y}}$ is the *measurement*, and the known degradation operator $\mathcal{A} : \mathbb{R}^{d_\mathbf{x}} \to \mathbb{R}^{d_\mathbf{y}}$ may represent masking, blur, downsampling, or compression. The posterior introduced earlier combines the image prior p_0 with a likelihood, denoted here by ℓ_0, that describes how a candidate image could produce the observation. The Gaussian likelihood is $\ell_0(\mathbf{y} \mid \mathbf{x}) = \mathbf{N}(\mathbf{y}; \mathcal{A}(\mathbf{x}), \sigma_\mathbf{y}^2\mathbf{I}_{d_\mathbf{y}})$, and the resulting posterior distribution is

$$p_0(\mathbf{x}_0 \mid \mathbf{y}) \propto \ell_0(\mathbf{y} \mid \mathbf{x}_0)p_0(\mathbf{x}_0). \tag{2.4}$$

To draw approximate samples from this posterior, we use the observed image $\mathbf{y}$ to guide the pretrained diffusion sampler, whose transitions are $p_{t|t+1}^{\eta,\theta}(\mathbf{x}_t \mid \mathbf{x}_{t+1})$. This requires the likelihood of $\mathbf{y}$ given the current noisy state $\mathbf{x}_t$, denoted by $\ell_t(\mathbf{y} \mid \mathbf{x}_t)$. Computing this likelihood requires averaging over the possible clean images given that noisy state: $\ell_t(\mathbf{y} \mid \mathbf{x}_t) := \int \ell_0(\mathbf{y} \mid \mathbf{x}_0)\, p_{0|t}(\mathbf{x}_0 \mid \mathbf{x}_t)\, d\mathbf{x}_0$ (Daras et al., 2024, Equation 2.20). Evaluating this integral over the high-dimensional space of clean images is generally intractable.

To avoid computing this integral exactly, diffusion solvers use an approximate correction to the denoiser's clean-image prediction. The posterior denoiser averages the possible clean images given both the noisy state $\mathbf{x}_t$ and the observed image $\mathbf{y}$: $\hat{\mathbf{x}}_0(\mathbf{x}_t, t \mid \mathbf{y}) := \mathbb{E}[\mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y}]$. For the Gaussian noising model above, this average equals the estimate based only on $\mathbf{x}_t$, plus a correction using $\mathbf{y}$ (Daras et al., 2024, Equations 2.15 and 2.17):

$$\hat{\mathbf{x}}_0(\mathbf{x}_t, t \mid \mathbf{y}) = \hat{\mathbf{x}}_0(\mathbf{x}_t, t) + \alpha_t^{-1}\sigma_t^2 \nabla_{\mathbf{x}_t} \log \ell_t(\mathbf{y} \mid \mathbf{x}_t). \tag{2.5}$$

The pretrained denoiser $\hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t)$ approximates the first term, $\hat{\mathbf{x}}_0(\mathbf{x}_t, t)$. The gradient $\nabla_{\mathbf{x}_t} \log \ell_t(\mathbf{y} \mid \mathbf{x}_t)$ is called the *guidance term*. To approximate it, a standard approach evaluates the likelihood at the denoiser's clean-image prediction instead of averaging over the possible clean images (Ho et al., 2022; Chung et al., 2023):

$$\ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t) := \ell_0\big(\mathbf{y} \mid \hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t)\big)\,. \tag{2.6}$$

This approximation ignores uncertainty about the clean image but lets solvers guide sampling with the observation, without further training the denoiser.

Guidance in latent space. We use a pretrained latent diffusion model with encoder $\mathcal{E}$ and decoder $\mathcal{D}$ (Rombach et al., 2022). From this point on, the $\mathbf{x}$-*family* denotes *latent diffusion states* and their clean and noise estimates in $\mathbb{R}^d$, so equation 2.2 is read as a latent transition. We color the diffusion state **red** $\mathbf{x}_t$, the proxy **blue** $\mathbf{x}_t^{\text{pxy}}$, and the encoded measurement **orange** $\mathcal{E}(\mathbf{y}) = \mathbf{y}_z$. In this setting, the two solvers we build on use different approximations to the intermediate likelihood $\ell_t(\mathbf{y} \mid \mathbf{x}_t)$. DING (Moufad et al., 2026) avoids differentiating through the denoiser. It rewrites the likelihood equation 2.6 in terms of the noise prediction and evaluates that prediction at an independent proxy $\mathbf{x}_t^{\text{pxy}} := \mathbf{m}_t + \eta_t \mathbf{w}', \quad \mathbf{w}' \sim \mathbf{N}(0, \mathbf{I})$ with $\mathbf{m}_t$ the mean of the prior transition in equation 2.2, giving

$$\begin{aligned} \ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t) = \ell_0\big(\mathbf{y} \mid \hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t)\big) &\overset{\text{(i)}}{=} \ell_0\big(\mathbf{y} \,\big|\, (\mathbf{x}_t - \sigma_t \hat{\mathbf{x}}_1^\theta(\mathbf{x}_t, t))/\alpha_t\big) \\ &\approx \ell_0\big(\mathbf{y} \,\big|\, (\mathbf{x}_t - \sigma_t \hat{\mathbf{x}}_1^\theta(\mathbf{x}_t^{\text{pxy}}, t))/\alpha_t\big) =: \ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t, \mathbf{x}_t^{\text{pxy}}). \end{aligned} \tag{2.7}$$

Here, (i) uses the relation between the clean and noise predictions stated after equation 2.2. The proxy noise $\mathbf{x}_t^{\text{pxy}}$ is fixed with respect to $\mathbf{x}_t$, making the likelihood argument *affine*. Therefore, evaluating the gradient $\nabla_{\mathbf{x}_t} \log \ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t, \mathbf{x}_t^{\text{pxy}})$ does not require differentiation through the denoiser $\hat{\mathbf{x}}_1^\theta(\mathbf{x}_t^{\text{pxy}}, t)$. Because $\mathbf{x}_t^{\text{pxy}}$ is fixed and, for inpainting, the mask only selects the observed coordinates, combining the Gaussian likelihood $\ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t, \mathbf{x}_t^{\text{pxy}})$ with the Gaussian prior transition $p_{t|t+1}^{\eta,\theta}(\mathbf{x}_t \mid \mathbf{x}_{t+1})$ gives another Gaussian. DING uses this construction to sample exactly from the resulting surrogate posterior transition $p_{t|t+1}^\theta(\mathbf{x}_t \mid \mathbf{x}_{t+1}, \mathbf{x}_t^{\text{pxy}}, \mathbf{y}) \propto \ell_t^\theta(\mathbf{y} \mid \mathbf{x}_t, \mathbf{x}_t^{\text{pxy}})\, p_{t|t+1}^{\eta,\theta}(\mathbf{x}_t \mid \mathbf{x}_{t+1})$.

SILO (Raphaeli et al., 2025) uses the same likelihood approximation in equation 2.6 but replaces decoding, applying the degradation $\mathcal{A}$, and re-encoding with a learned latent operator $\mathbf{H}_\psi \approx \mathcal{E} \circ \mathcal{A} \circ \mathcal{D}$. With the encoded measurement $\mathcal{E}(\mathbf{y}) \in \mathbb{R}^d$, it uses

$$\ell_t^{\theta,\psi}(\mathbf{y} \mid \mathbf{x}_t) := \mathbf{N}\big(\mathcal{E}(\mathbf{y}); \mathbf{H}_\psi(\hat{\mathbf{x}}_0^\theta(\mathbf{x}_t, t)), r^2 \mathbf{I}_d\big)\,. \tag{2.8}$$

Here, r is the assumed noise scale of the latent forward degradation model. This permits any degradation in the latent space for which an accurate $\mathbf{H}_\psi$ can be trained, while evaluating $\mathcal{E}$ and $\mathcal{D}$ only once per trajectory. Its guidance $\nabla_{\mathbf{x}_t} \log \ell_t^{\theta,\psi}(\cdot)$ nevertheless backpropagates through both $\mathbf{H}_\psi$ and $\hat{\mathbf{x}}_0^\theta$ at every step.

3 METHOD

We begin with the observation that equation 2.7 does not produce a Gaussian surrogate posterior transition that can be sampled in closed form when the degradation operator is a learned network $\mathbf{H}_\psi$. PASEO uses the proxy construction from equation 2.7, which avoids gradients through the denoiser, and extends it to equation 2.8. This yields a Gaussian approximation to the surrogate posterior transition. We achieve this by (i) locally approximating the learned operator with a linear map, and (ii) using perturbations to sample the surrogate transition efficiently without forming or inverting its covariance matrix. Figure 2 gives an overview and Algorithm 1 the pseudocode.

Training the latent operator. Before describing the sampler, we briefly explain how $\mathbf{H}_\psi$ is trained. Following Raphaeli et al. (2025), we train a small network offline for each degradation and latent

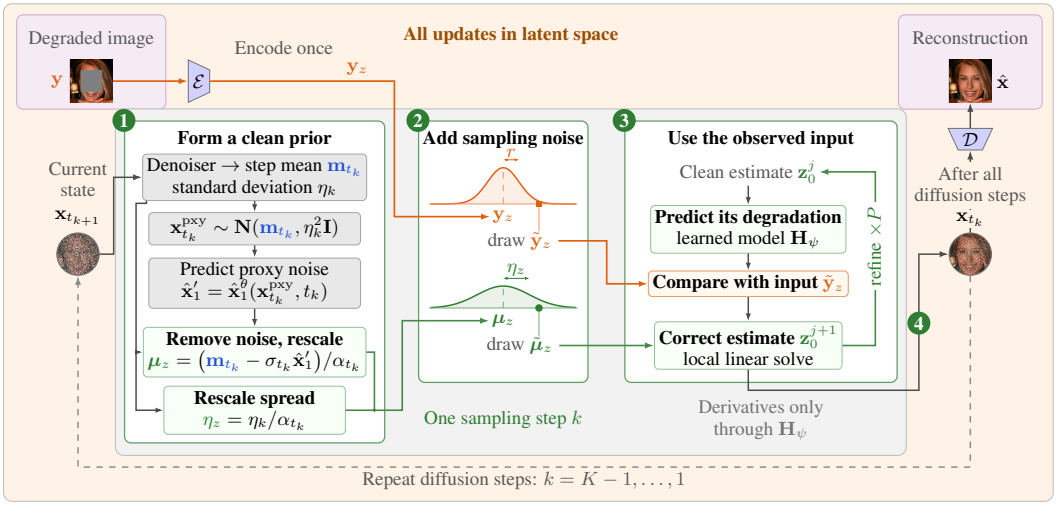


Figure 2: **Overview of PASEO.** Each diffusion step is conditioned on the measurement through a small learned operator $\mathbf{H}_\psi$ in latent space. Only $\mathbf{H}_\psi$ is differentiated, *never the denoiser*, and the autoencoder runs only once. Numbers match Steps 1–4 (Algorithm 1).

space to predict the encoded degraded measurement from the encoded clean image, minimizing an ℓ_1 prediction loss. The diffusion model and autoencoder remain fixed, and $\mathbf{H}_\psi$ is kept fixed during sampling. Full training details are given in Appendix E.1.

1 The denoiser only sets the prior. We now turn to sampling with the trained operator, which acts on clean latents; we denote them $\mathbf{z}_0$ and color them **green**. Diffusion sampling, however, starts from a noisy state $\mathbf{x}_t$, so we express the diffusion state at t as a distribution over the clean latent. From the current state $\mathbf{x}_{t+1}$, the DDIM transition equation 2.2 proposes the next state $\mathbf{x}_t = \mathbf{m}_t + \eta_t \mathbf{w}$ with $\mathbf{w} \sim \mathbf{N}(0, \mathbf{I}_d)$, where $\mathbf{m}_t := \alpha_t \hat{\mathbf{x}}_0^\theta(\mathbf{x}_{t+1}, t+1) + (\sigma_t^2 - \eta_t^2)^{1/2} \hat{\mathbf{x}}_1^\theta(\mathbf{x}_{t+1}, t+1)$. As in equation 2.7, PASEO also draws an independent proxy state $\mathbf{x}_t^{\text{pxy}} = \mathbf{m}_t + \eta_t \mathbf{w}'$, evaluates the denoiser once on it, and keeps its noise prediction $\hat{\mathbf{x}}_1' := \hat{\mathbf{x}}_1^\theta(\mathbf{x}_t^{\text{pxy}}, t)$ fixed for the rest of the step. With $\hat{\mathbf{x}}_1'$ fixed, a state and its clean latent are linked by the simple relation $\mathbf{x}_t = \alpha_t \mathbf{z}_0 + \sigma_t \hat{\mathbf{x}}_1'$, as in equation 2.1. Solving it for $\mathbf{z}_0$ and substituting the next-state draw gives the *clean-latent prior*:

$$\mathbf{z}_0 = \frac{\mathbf{x}_t - \sigma_t \hat{\mathbf{x}}_1'}{\alpha_t} = \frac{\mathbf{m}_t - \sigma_t \hat{\mathbf{x}}_1'}{\alpha_t} + \frac{\eta_t}{\alpha_t}\mathbf{w} = \boldsymbol{\mu}_z + \eta_z \mathbf{w} \quad \Rightarrow \quad \mathbf{z}_0 \sim \mathbf{N}(\boldsymbol{\mu}_z, \eta_z^2 \mathbf{I}_d). \tag{3.1}$$

This is where the denoiser's role ends. Its two forward passes set $\boldsymbol{\mu}_z$ (the schedule sets η_z), and nothing below differentiates through it.

2 Perturb and solve: a posterior sample from one linear solve. Before turning to the learned network, it helps to look at a linear latent degradation $H \in \mathbb{R}^{d \times d}$. Here the posterior has a closed form, which makes it easy to see how the measurement corrects the prior and why sampling from it is expensive. Step 3 then carries this over to $\mathbf{H}_\psi$. The forward degradation model is $\mathbf{y}_z = H\mathbf{z}_0 + \epsilon$, with $\mathbf{y}_z := \mathcal{E}(\mathbf{y})$ and $\epsilon \sim \mathbf{N}(0, r^2 \mathbf{I}_d)$, where r is the latent measurement scale in equation 2.8. It tells how likely a measurement $\mathbf{y}_z$ is for a given clean latent $\mathbf{z}_0$: $\mathbf{N}(\mathbf{y}_z; H\mathbf{z}_0, r^2\mathbf{I}_d)$. Multiplying this by the clean-latent prior equation 3.1 gives the posterior, $p(\mathbf{z}_0 \mid \mathbf{y}_z) \propto \mathbf{N}(\mathbf{y}_z; H\mathbf{z}_0, r^2\mathbf{I}_d)\,\mathbf{N}(\mathbf{z}_0; \boldsymbol{\mu}_z, \eta_z^2\mathbf{I}_d)$. Because H is linear, this product is again a Gaussian, $\mathbf{N}(\mathbf{m}_p, \boldsymbol{\Sigma}_p)$ (Bishop, 2006, Eq. 2.116). Written with the Woodbury identity (Henderson & Searle, 1981), its mean is

$$\mathbf{m}_p = \boldsymbol{\mu}_z + K(\mathbf{y}_z - H\boldsymbol{\mu}_z), \qquad K := \eta_z^2 H^T S^{-1}, \qquad S := r^2\mathbf{I}_d + \eta_z^2 H H^T \tag{3.2}$$

(Appendix A.2). The formula starts from the prior mean, compares the measurement with what the operator predicts there, and corrects by the mismatch. The gain K decides how much to trust the measurement: a broad prior (large η_z) or a precise measurement (small r) moves the result further toward the measurement.

Computing this mean does not require inverting S: we solve the linear system $S\mathbf{v} = \mathbf{y}_z - H\boldsymbol{\mu}_z$ for $\mathbf{v}$ and set $\mathbf{m}_p = \boldsymbol{\mu}_z + \eta_z^2 H^T \mathbf{v}$. Drawing a sample is harder. For a mask, S is diagonal and

the sample follows coordinate by coordinate. For a general H, S couples all coordinates and the covariance $\mathbf{\Sigma}_p$ is a dense $d \times d$ matrix. A direct sample would take the form $\mathbf{z}_0 = \mathbf{m}_p + \mathbf{\Sigma}_p^{1/2}\boldsymbol{\xi}$, with $\boldsymbol{\xi} \sim \mathbf{N}(0, \mathbf{I}_d)$, which needs this matrix and its square root; both are out of reach at latent sizes (with SD3.5, $d = 65{,}536$, so $\mathbf{\Sigma}_p$ has about 4.3 billion entries).

PASEO gets a sample without ever building the covariance. It adds noise to the two inputs of the mean formula, each at its own level: the prior mean gets the prior's spread η_z, and the measurement gets the measurement noise r (Step 2 in Figure 2). It then applies the same formula to the noisy inputs (Step 3). With independent $\boldsymbol{\xi}_z, \boldsymbol{\xi}_y \sim \mathbf{N}(0, \mathbf{I}_d)$,

$$\boxed{\mathbf{z}_0^+ = \tilde{\boldsymbol{\mu}}_z + K\big(\tilde{\mathbf{y}}_z - H\tilde{\boldsymbol{\mu}}_z\big), \quad \tilde{\boldsymbol{\mu}}_z = \boldsymbol{\mu}_z + \eta_z\boldsymbol{\xi}_z, \quad \tilde{\mathbf{y}}_z = \mathbf{y}_z + r\boldsymbol{\xi}_y} \tag{3.3}$$

Algorithm 1 PASEO - Pseudocode

Input: decreasing timesteps $(t_k)_{k=K}^0$; latent measurement $\mathbf{y}_z$; latent operator $\mathbf{H}_\psi$; noise level r; DDIM parameters $(\eta_k)_{k=K}^0$; damping λ; relaxation ρ; inner steps P; CG budget C.

$\mathbf{x} \sim \mathbf{N}(0, \mathbf{I}_d)$
for $k = K-1$ **to** 1 **do**
  $\hat{\mathbf{x}}_0 \leftarrow \hat{\mathbf{x}}_0^\theta(\mathbf{x}, t_{k+1})$
  $\hat{\mathbf{x}}_1 \leftarrow \big(\mathbf{x} - \alpha_{t_{k+1}}\hat{\mathbf{x}}_0\big)/\sigma_{t_{k+1}}$
  $\mathbf{m} \leftarrow \alpha_{t_k}\hat{\mathbf{x}}_0 + \big(\sigma_{t_k}^2 - \eta_k^2\big)^{1/2}\hat{\mathbf{x}}_1$
  ① /* Form clean-latent prior */
  $\mathbf{x}_{t_k}^{\text{pxy}} \leftarrow \mathbf{m} + \eta_k\mathbf{w}', \quad \mathbf{w}' \sim \mathbf{N}(\mathbf{0}, \mathbf{I}_d)$
  $\hat{\mathbf{x}}_1' \leftarrow \hat{\mathbf{x}}_1^\theta(\mathbf{x}_{t_k}^{\text{pxy}}, t_k)$
  $\boldsymbol{\mu}_z \leftarrow (\mathbf{m} - \sigma_{t_k}\hat{\mathbf{x}}_1')/\alpha_{t_k}, \quad \eta_z := \eta_k/\alpha_{t_k}$
  $\mathbf{z}_0^0 \leftarrow (\mathbf{x}_{t_k}^{\text{pxy}} - \sigma_{t_k}\hat{\mathbf{x}}_1')/\alpha_{t_k}$
  ② /* Perturb-and-optimize targets */
  $\boldsymbol{\xi}_y \sim \mathbf{N}(\mathbf{0}, \mathbf{I}_d), \quad \boldsymbol{\xi}_z \sim \mathbf{N}(\mathbf{0}, \mathbf{I}_d)$
  $\tilde{\mathbf{y}}_z \leftarrow \mathbf{y}_z + r\,\boldsymbol{\xi}_y, \quad \tilde{\boldsymbol{\mu}}_z \leftarrow \boldsymbol{\mu}_z + \eta_z\,\boldsymbol{\xi}_z$
  ③ /* Local measurement update */
  for $j = 0$ **to** $P-1$ **do**
    $\mathbf{h} \leftarrow \mathbf{H}_\psi(\mathbf{z}_0^j)$
    $\mathbf{J} \leftarrow \partial\mathbf{H}_\psi/\partial\mathbf{z}_0\big|_{\mathbf{z}_0^j}$
    $\mathbf{b} \leftarrow \tilde{\mathbf{y}}_z - \mathbf{h} - \mathbf{J}(\tilde{\boldsymbol{\mu}}_z - \mathbf{z}_0^j)$
    solve $\big((r^2+\lambda)\mathbf{I}_d + \eta_z^2\mathbf{J}\mathbf{J}^\top\big)\mathbf{v} = \mathbf{b}$
    $\mathbf{z}_0^\star \leftarrow \tilde{\boldsymbol{\mu}}_z + \eta_z^2\mathbf{J}^\top\mathbf{v}$
    $\mathbf{z}_0^{j+1} \leftarrow \mathbf{z}_0^j + \rho\,(\mathbf{z}_0^\star - \mathbf{z}_0^j)$
  $\mathbf{x} \leftarrow \alpha_{t_k}\mathbf{z}_0^P + \sigma_{t_k}\hat{\mathbf{x}}_1'$ ▷ back to the diffusion state ④
Return: $\mathcal{D}(\hat{\mathbf{x}}_0^\theta(\mathbf{x}, t_1))$

For a linear H, $\mathbf{z}_0^+$ is an exact draw from $\mathbf{N}(\mathbf{m}_p, \mathbf{\Sigma}_p)$ (Appendix A.4). This works because the formula is linear in its two inputs: noise that enters at the prior's and the measurement's levels leaves with exactly the posterior's spread. The perturb-and-solve identity itself is known in the sampling literature (Papandreou & Yuille, 2010; Bardsley et al., 2014). Our observation is that, once the proxy fixes the noise prediction, conditioning a latent diffusion step reduces to exactly this problem; Step 3 extends it to the learned operator by local linearization. Two consequences follow.

A sample costs the same as the mean. The sample uses the same solve as the mean, only with the noisy inputs: $S\mathbf{v} = \tilde{\mathbf{y}}_z - H\tilde{\boldsymbol{\mu}}_z$ gives $\mathbf{z}_0^+ = \tilde{\boldsymbol{\mu}}_z + \eta_z^2 H^T\mathbf{v}$. The conjugate gradient (CG) method (Hestenes & Stiefel, 1952) solves it using only products with H and H^T, so neither S nor $\mathbf{\Sigma}_p$ is ever built (Appendix A.3).

The denoiser is never differentiated. It enters only through the fixed value $\boldsymbol{\mu}_z$.

For a mask, equation 3.3 reduces to the closed-form step of equation 2.7 (Appendix D.1).

③ Solving with the learned operator. So far the operator was a fixed linear map. In practice it is the learned network $\mathbf{H}_\psi$, and the forward degradation model becomes $\mathbf{y}_z = \mathbf{H}_\psi(\mathbf{z}_0) + \epsilon$. The identity equation 3.3 needs a linear map, so we cannot apply it to $\mathbf{H}_\psi$ directly. Instead, we use the fact that close to the current estimate $\mathbf{z}_0^{(j)}$, the network behaves almost like a linear map,

$$\mathbf{H}_\psi(\mathbf{z}_0) \approx \mathbf{H}_\psi(\mathbf{z}_0^{(j)}) + \mathbf{J}_j\big(\mathbf{z}_0 - \mathbf{z}_0^{(j)}\big), \tag{3.4}$$

where the Jacobian $\mathbf{J}_j$ tells how the predicted measurement changes when the latent moves away from $\mathbf{z}_0^{(j)}$. Within this approximation we can use equation 3.3 with $\mathbf{J}_j$ in place of H. The update compares the noisy measurement $\tilde{\mathbf{y}}_z$ with what the linearized network predicts at $\tilde{\boldsymbol{\mu}}_z$, and uses the mismatch to correct the estimate, giving an approximate posterior draw $\mathbf{z}_0^\star$:

$$\boxed{\mathbf{z}_0^\star = \tilde{\boldsymbol{\mu}}_z + \eta_z^2\mathbf{J}_j^T\mathbf{v}, \quad S_j\mathbf{v} = \tilde{\mathbf{y}}_z - \mathbf{H}_\psi(\mathbf{z}_0^{(j)}) - \mathbf{J}_j\big(\tilde{\boldsymbol{\mu}}_z - \mathbf{z}_0^{(j)}\big)} \tag{3.5}$$

with $S_j := r^2\mathbf{I}_d + \eta_z^2\mathbf{J}_j\mathbf{J}_j^T$. The approximation is only accurate near the point where it was built, and the correction moves the estimate away from that point. We therefore linearize again at the new

estimate and repeat the correction, P times in total (the "refine $\times P$" loop in Figure 2), always with the same noisy inputs $\tilde{\boldsymbol{\mu}}_z$ and $\tilde{\mathbf{y}}_z$. Each solve runs CG, which only needs the products $\mathbf{J}_j\mathbf{v}$ and $\mathbf{J}_j^T\mathbf{u}$. Automatic differentiation computes them through the small network $\mathbf{H}_\psi$, without forming $\mathbf{J}_j$ and without touching the denoiser (Appendix B.3). Because $\mathbf{H}_\psi$ is nonlinear, $\mathbf{z}_0^\star$ is an exact posterior sample only for the linearized model (Appendix B.2).

4 Back to the diffusion state. The corrections happen in the clean latent, but the sampler moves through noisy states, so the result has to become a state at time t before the next step. We use the relation that built the prior in Step 1, with the same fixed noise prediction $\hat{\mathbf{x}}_1'$, to get the noisy state $\mathbf{x}_t = \alpha_t \mathbf{z}_0^{(P)} + \sigma_t \hat{\mathbf{x}}_1'$. This undoes the change of variables of Step 1, so the corrected clean latent becomes the next diffusion state, and the next step starts from it. Over the whole run, the measurement is encoded once and the final clean estimate is decoded once.

Practical choices. A few settings keep the corrections of Step 3 cheap and stable. We start from the proxy's clean latent $\mathbf{z}_0^{(0)} := (\mathbf{x}_t^{\text{pxy}} - \sigma_t \hat{\mathbf{x}}_1')/\alpha_t$. We add $\lambda \mathbf{I}_d$ to S_j, with $\lambda \geq 0$, which keeps the solve well conditioned when r is small. We stop CG after at most C iterations, which bounds the cost and regularizes the update (Hanke, 1995); it also beats an exact solve (Section 4.2). With the denoiser anchor and no perturbation, the step solves the DDS (Chung et al., 2024a) problem (Appendix D.1.1). Because the linearization is only accurate nearby, each correction moves only part of the way, $\mathbf{z}_0^{(j+1)} = \mathbf{z}_0^{(j)} + \rho(\mathbf{z}_0^\star - \mathbf{z}_0^{(j)})$ with $0 < \rho \leq 1$. For inpainting, we weight the likelihood by $\mathbf{W} = \operatorname{diag}(w)$ with $w_i \propto \|(\mathbf{J}_j)_{i,:}\|$, the sensitivity of measurement i to the latent, so insensitive measurements get little weight; $\mathbf{W}$ multiplies $\mathbf{J}$ and $\mathbf{b}$ in lines 18–21 of Algorithm 2 (Appendix F.6.1).

One more choice concerns $\tilde{\boldsymbol{\mu}}_z$, the point each correction starts from, which we call the *anchor*. Early in diffusion sampling the prior equation 3.1 is wide, so the anchor can land far from the proxy's clean latent $\mathbf{z}_0^{(0)}$, where the first linearization is built. We found that reusing part of the proxy noise $\mathbf{w}'$ in place of the fresh noise $\boldsymbol{\xi}_z$ improves quality on most tasks at the same budget:

$$\tilde{\boldsymbol{\mu}}_z \leftarrow \boldsymbol{\mu}_z + \eta_z(c_1\mathbf{w}' + c_2\boldsymbol{\xi}_z), \qquad (c_1, c_2) = \big((1-\beta_k^2)^{1/2},\, \beta_k\big), \qquad \beta_k = \eta_{z,\min}/\eta_{z,k}, \quad (3.6)$$

where k indexes the current diffusion step, as in Algorithm 1, and $\eta_{z,\min}$ is the smallest prior standard deviation in the schedule. We call this the *arc anchor*: (c_1, c_2) lies on the unit circle, so the anchor keeps the prior's spread η_z. Since $\mathbf{z}_0^{(0)} = \boldsymbol{\mu}_z + \eta_z \mathbf{w}'$, it leans toward the proxy's clean latent while the prior is wide and recovers $\tilde{\boldsymbol{\mu}}_z$ when $\beta_k = 1$. Unlike the fixed anchor of Step 3, this anchor is recomputed from the current estimate at every correction (Algorithm 2); Algorithm 1 keeps the fixed original anchor for clarity. Appendices C and F.2 give the analysis and compare anchors.

4 Experiments

Models and datasets. We evaluate PASEO on five degradation tasks: $\times 4$ and $\times 8$ super-resolution, Gaussian blur (61×61, $\sigma = 3$), 256×256 center inpainting, and JPEG decompression at quality 10%, all at $\sigma_{\mathbf{y}} = 0.01$. We use two latent priors: diffusion-based RV-v5.1 (SG161222, 2023) and flow-based Stable Diffusion 3.5 (medium) (Esser et al., 2024). For sampler evaluation, we use 1000 held-out FFHQ (Karras et al., 2019) images and 1000 COCO-val2017 (Lin et al., 2014) images, all at 512×512.

Evaluation, metrics, and baselines. Following Raphaeli et al. (2025), we train a separate latent degradation operator $\mathbf{H}_\psi$ for each task. For FFHQ, we split the dataset into 68,000 training, 1000 validation, and 1000 evaluation images. For COCO, we train on the 84,990 LSDIR-train images (Li et al., 2023), validate on 250 LSDIR validation images, and evaluate on the first 1000 COCO-val2017 images. Appendix E.1 gives the training details. We report the following metrics: PSNR (Horé & Ziou, 2010) and LPIPS (Zhang et al., 2018), KID (Bińkowski et al., 2018) for distributional alignment, wall-clock seconds per image and peak device memory in GiB. We emphasize LPIPS, since PSNR favors overly smooth images when exact recovery is impossible (Janati et al., 2025, App. B.6). Across FFHQ and COCO, we compare PASEO against SILO, ReSample (Song et al., 2024), PSLD and GML-DPS (Rout et al., 2023), LDPS (Chung et al., 2023), DING, DAPS, RedDiff (Mardani et al., 2024), FlowDPS, FlowChef (Patel et al., 2025), and PnP-Flow. All our experiments are conducted on an AMD MI210 GPU. Appendix E.3 provides more details on the setup.

Hyperparameters. Our prompts are `A high quality photo of a face` for FFHQ and `A high quality photo` for COCO. The guidance scale is 1, except for SD3.5 on blur, JPEG, and FFHQ $\times 4$ super-resolution, where it is 4. Each reverse step calls the denoiser twice, at the current state and at the proxy sample, except the last two, which share one call; K steps thus cost $2K - 3$ NFEs. On FFHQ, both priors use 28 steps (53 NFEs). On COCO, we use 50 steps (97 NFEs) for $\times 4$ super-resolution, blur, and JPEG, and 100 steps (197 NFEs) for $\times 8$ super-resolution and inpainting.

Each step runs P inner corrections, with $P = 3$–5 for RV-v5.1 and, for SD3.5, $P = 1$–3 on FFHQ and 2–5 on COCO. The CG budget is $C = 1$–3 iterations, except for FFHQ inpainting ($C = 12$ for RV-v5.1, $C = 5$ for SD3.5). Appendix E.4 lists these and the remaining settings (noise level r, relaxation ρ, damping λ) for each task and prior, and Appendix F.3 examines their sensitivity.

Table 1: FFHQ restoration. Left: PASEO with RV-v5.1. Right: SD3.5-medium, baselines at 79 NFEs and PASEO at 53. Lower is better for all metrics. Bold/underline mark best/second-best. RES. = RESAMPLE; GML = GML-DPS

Task	Method	RV-v5.1 LPIPS	RV-v5.1 KID	RV-v5.1 Time	RV-v5.1 Mem	SD3.5-medium LPIPS	SD3.5-medium KID	SD3.5-medium Time	SD3.5-medium Mem
SR $\times 4$	PASEO	0.139	**1.81**	**11.7**	**3.6**	**0.182**	10.2	**10.6**	**16.8**
	PASEO†	**0.130**	2.96	14.4	**3.6**	0.221	9.49	25.9	**16.8**
	SILO	0.182	5.10	149	7.4	0.303	29.3	20	21.2
	RES.	0.433	25.5	1418	7.0	**0.182**	**7.70**	130	22.5
	PSLD	0.249	10.1	390	7.9	0.494	47.2	47	23.4
	GML	0.247	9.05	389	7.9	0.373	46.3	48	23.4
	LDPS	0.281	11.6	331	7.0	0.423	57.1	39	22.5
SR $\times 8$	PASEO	**0.212**	5.99	**11.6**	**3.6**	**0.330**	**28.8**	**8.8**	**16.8**
	PASEO†	0.241	13.13	14.4	**3.6**	0.420	30.3	10.7	**16.8**
	SILO	0.226	**5.23**	149	7.4	0.501	62.8	20	21.2
	RES.	0.575	118	1418	7.0	0.378	31.6	130	22.5
	PSLD	0.320	14.9	390	7.9	0.453	60.9	48	23.4
	GML	0.327	12.9	389	7.9	0.439	57.1	48	23.4
	LDPS	0.343	12.9	331	7.0	0.456	60.2	39	22.5
Gaussian blur	PASEO	**0.162**	**2.09**	**17.3**	**3.6**	**0.200**	**8.32**	**8.6**	**16.8**
	PASEO†	0.172	6.66	121	**3.6**	0.215	10.7	10.1	**16.8**
	SILO	0.222	8.21	149	7.4	0.399	47.9	20	21.2
	RES.	0.253	10.7	1418	7.0	0.206	9.37	138	22.5
	PSLD	0.288	12.2	390	7.9	0.450	49.1	69	25.6
	GML	0.309	16.5	389	7.9	0.449	57.3	48	23.4
	LDPS	0.327	19.9	331	7.0	0.467	61.4	40	22.5
Inpainting	PASEO	0.144	14.00	38.4	**3.6**	**0.118**	**2.71**	13.6	**16.8**
	PASEO†	0.147	13.36	31.2	**3.6**	0.144	11.4	**11.0**	**16.8**
	SILO	**0.139**	**1.80**	149	7.4	0.430	115	20	21.2
	DING	0.159	4.59	**8.8**	**3.6**	0.134	5.89	12.6	**16.8**
	RES.	0.273	119	1418	7.0	0.214	55.6	267	22.5
	PSLD	0.357	17.2	390	7.9	0.459	69.9	48	23.4
	GML	0.356	16.5	389	7.9	0.509	79.2	48	23.4
	LDPS	0.368	16.0	331	7.0	0.507	78.5	39	22.5
JPEG q=10	PASEO	0.258	26.53	**8.2**	**3.6**	0.350	71.2	**13.9**	**16.8**
	PASEO†	0.238	22.73	14.0	**3.6**	0.400	58.7	60.5	**16.8**
	SILO	**0.203**	**4.21**	138	7.4	0.503	77.4	20	21.2
	RES.	0.456	20.1	2438	7.0	**0.196**	**19.4**	140	22.5
	GML	0.268	7.54	402	7.9	0.413	50.1	49	23.4
	LDPS	0.373	17.7	412	7.0	0.424	54.4	40	22.5

4.1 MAIN RESULTS

We report two versions of PASEO. Our analysis uses the original anchor of Algorithm 1, a fresh draw from the prior. Early in sampling, however, the prior is wide, and this anchor can land far from the proxy's clean latent. Our default, PASEO, therefore uses the arc anchor of equation 3.6, which reuses part of the proxy noise to start near that latent. We also report PASEO†, which keeps the original anchor.

FFHQ. With both priors, PASEO ranks first or second in LPIPS on almost every task, while running faster and with less memory than every baseline except the inpainting-only DING (Table 1). With RV-v5.1, the baselines are published results that use far more denoiser evaluations (999 NFEs for SILO, against our 53), yet PASEO or PASEO† has the lowest LPIPS on both super-resolution tasks and blur; SILO keeps inpainting and JPEG. With SD3.5, where every baseline gets more NFEs, PASEO has the best LPIPS on four of the five tasks (Appendix G). Against the strongest baseline, RESAMPLE, it is 10–20$\times$ faster and uses about 25% less memory; the price is PSNR, which is higher for RESAMPLE on four tasks (Table 23). JPEG, our only nonlinear degradation, is the hardest case for our sampler with both priors. DING is faster on inpainting at the same memory, but it handles only that task, and with SD3.5 PASEO beats it on all three quality metrics.

COCO. At matched NFEs, PASEO beats the flow-matching baselines FLOWDPS, FLOWCHEF, and PNP-FLOW on every task in LPIPS, runtime, and memory (Table 2; reconstructions in Figure 1). Against all baselines (Table 25), PASEO or PASEO† still has the lowest LPIPS on every task except JPEG, which is again the hardest case. Across the 15 settings of Tables 1 and 2, PASEO† beats PASEO in 3 and is within 0.02 LPIPS in 5 more, so the original anchor also works in practice. We keep the arc anchor as the default because it is better in the remaining cases and up to 7$\times$ faster.

Why JPEG is hard. Each correction relies on one question: if the latent changes slightly, how does the measurement change? For blur, super-resolution, and inpainting, the answer is smooth and reliable, because these degradations are linear. JPEG instead rounds the image's frequency coefficients to fixed levels: small changes to the image leave the output unchanged, and the output changes only in abrupt jumps, when a coefficient crosses to the next level. There is no reliable local direction to follow, and the learned operator cannot supply one, so the corrections get little guidance. This makes JPEG our hardest case with both priors. We test this in Appendix E.2, where the learned operator's local directions for JPEG do not match those of the true degradation.

4.2 Ablations

Solve depth. We stop CG after a few iterations, far from an exact solve. To test whether this pulls the sampler away from the posterior, we replace $\mathbf{H}_\psi$ with an exact linear latent blur, whose exact update we can compute. We also draw the ground truth from the model's own prior, so that a calibrated sampler should see it as one more draw (Appendix F.1.2).

Table 2: COCO-val2017 with SD3.5-medium: PASEO versus flow-matching baselines at matched NFEs.

Task	Method	LPIPS ↓	KID ↓	Time ↓	Mem ↓
SR ×4	PASEO	0.208	1.14	**26.4**	**16.8**
	PASEO†	**0.196**	**0.73**	27.8	**16.8**
	FlowDPS	0.374	9.19	76.8	18.7
	FlowChef	0.345	8.03	228.9	18.7
	PnP-Flow	0.449	19.00	35.5	18.7
SR ×8	PASEO	**0.344**	**7.36**	**41.2**	**16.8**
	PASEO†	0.389	35.92	92.6	**16.8**
	FlowDPS	0.513	24.76	156.6	18.7
	FlowChef	0.550	24.13	468.3	18.7
	PnP-Flow	0.494	21.24	70.8	18.7
Blur	PASEO	**0.229**	**0.78**	22.7	**16.8**
	PASEO†	0.280	2.10	**18.3**	**16.8**
	FlowDPS	0.456	15.11	80.2	18.7
	FlowChef	0.438	15.01	240.2	18.7
	PnP-Flow	0.552	25.40	36.4	18.7
Inpaint	PASEO	**0.174**	**5.26**	58.7	**16.8**
	PASEO†	0.186	5.65	**40.3**	**16.8**
	FlowDPS	0.391	16.02	156.4	18.7
	FlowChef	0.391	27.47	468.1	18.7
	PnP-Flow	0.697	63.14	70.5	18.7
JPEG	PASEO	**0.302**	18.82	**24.8**	**16.8**
	PASEO†	0.307	17.27	112.6	**16.8**
	FlowDPS	0.378	21.56	80.7	18.7
	FlowChef	0.355	**9.91**	243.6	18.7
	PnP-Flow	0.457	26.70	36.9	18.7

Solving exactly adds a mesh-like texture (LPIPS 0.75–0.80), while a few iterations give sharp samples (Figure 3a). The arc anchor is best at $C = 3$ and the original anchor at $C = 8$ (LPIPS 0.06–0.07); at these budgets, the draws have the right amount of fine detail, and their spread matches their distance to the truth within 3%. With the arc anchor, the learned operator is also best at small budgets, $C = 2$–4 (Figure 3b). Stopping CG early is therefore not only cheaper: with an exact operator, it keeps the draws closest to the posterior. Like a step size, C and ρ are tuned per task. Appendix F.3 reports LPIPS at every budget, and Appendix F.2 compares the anchors.

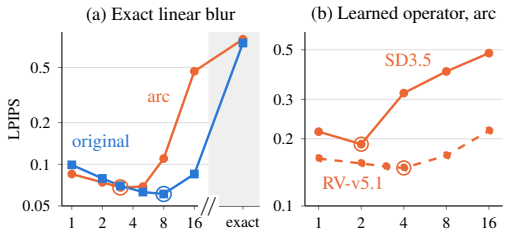


Figure 3: Few CG iterations are best: LPIPS against the number of CG iterations C; circles mark the best budget. (a) Exact linear latent blur, truths from the model's prior (64 images, Appendix F.1.2). (b) The learned operator on blur (Tables 12 and 13).

5 Conclusion

Conditioning a latent diffusion model on a learned operator does not require back-propagating through the denoiser. PASEO casts each step as Gaussian inference over the clean latent, so conditioning becomes a perturb-and-solve step that is exact for affine operators, forms no Jacobian, and runs the autoencoder only twice per reconstruction; a few CG iterations beat an exact solve. Across FFHQ and COCO, with the same or fewer denoiser evaluations than every baseline, PASEO ranks first or second in LPIPS on almost every task, while running up to 19× faster and using up to 34% less peak memory than the baselines.

Limitations. PASEO needs a trained latent operator for each degradation–prior pair, and its quality is bounded by that operator: on JPEG, where the learned operator's local directions are poorest, PASEO is weakest.

AI USE STATEMENT

In this work, we used generative AI tools to draft and edit the text of the paper, to help with some of the mathematical formulations, and, at times, to help run experiment jobs. We have reviewed and verified all AI-assisted work, including the mathematical formulations. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Idan Achituve, Hai Victor Habi, Amir Rosenfeld, Arnon Netzer, Idit Diamant, and Ethan Fetaya. Inverse problem sampling in latent space using sequential Monte Carlo. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 420–443. PMLR, 2025. URL https://proceedings.mlr.press/v267/achituve25a.html.

Ismail Alkhouri, Shijun Liang, Cheng-Han Huang, Jimmy Dai, Qing Qu, Saiprasad Ravishankar, and Rongrong Wang. SITCOM: Step-wise triple-consistent diffusion sampling for inverse problems. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 1128–1158. PMLR, 2025. URL https://proceedings.mlr.press/v267/alkhouri25b.html.

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

Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=r1lUOzWCW.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

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 (ICLR)*, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k. Spotlight presentation.

Hyungjin Chung, Suhyeon Lee, and Jong Chul Ye. Decomposed diffusion sampler for accelerating large-scale inverse problems. In *International Conference on Learning Representations (ICLR)*, 2024a. arXiv:2303.05754.

Hyungjin Chung, Jong Chul Ye, Peyman Milanfar, and Mauricio Delbracio. Prompt-tuning latent diffusion models for inverse problems. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8941–8967. PMLR, 2024b. URL https://proceedings.mlr.press/v235/chung24b.html.

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. doi: 10.48550/arXiv.2410.00083. URL https://arxiv.org/abs/2410.00083.

Filip Ekström Kelvinius, Zheng Zhao, and Fredrik Lindsten. Solving linear-Gaussian Bayesian inverse problems with decoupled diffusion sequential Monte Carlo. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15148–15181. PMLR, 2025. URL https://proceedings.mlr.press/v267/ekstrom-kelvinius25b.html.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 12606–12633. PMLR, 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Martin Hanke. *Conjugate Gradient Type Methods for Ill-Posed Problems*, volume 327 of *Pitman Research Notes in Mathematics*. Longman Scientific & Technical, 1995.

H. V. Henderson and S. R. Searle. On deriving the inverse of a sum of matrices. *SIAM Review*, 23(1): 53–60, 1981. doi: 10.1137/1023004. URL https://doi.org/10.1137/1023004.

Magnus R. Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6):409–436, 1952. Research Paper 2379.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 8633–8646. Curran Associates, Inc., 2022. doi: 10.52202/068431-0628. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/39235c56aef13fb05a6adc95eb9d8d66-Abstract-Conference.html.

Alain Horé and Djemel Ziou. Image quality metrics: PSNR vs. SSIM. In *2010 20th International Conference on Pattern Recognition*, pp. 2366–2369. IEEE, 2010. doi: 10.1109/ICPR.2010.579. URL https://doi.org/10.1109/ICPR.2010.579.

Yazid Janati, Badr Moufad, Mehdi Abou El Qassime, Alain Oliviero Durmus, Eric Moulines, and Jimmy Olsson. A mixture-based framework for guiding diffusion models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 26830–26876. PMLR, 2025. URL https://proceedings.mlr.press/v267/janati25a.html.

Yuchen Jiao, Na Li, Changxiao Cai, Yuxin Chen, and Gen Li. Provable diffusion-based posterior sampling for linear inverse problems via DDIM. *arXiv preprint arXiv:2607.19333*, 2026. URL https://arxiv.org/abs/2607.19333.

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 (CVPR)*, pp. 4401–4410, June 2019. doi: 10.1109/CVPR.2019.00453. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Karras_A_Style-Based_Generator_Architecture_for_Generative_Adversarial_Networks_CVPR_2019_paper.html.

Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2503.08136.

Yawei Li, Kai Zhang, Jingyun Liang, Jiezhang Cao, Ce Liu, Rui Gong, Yulun Zhang, Hao Tang, Yun Liu, Denis Demandolx, Rakesh Ranjan, Radu Timofte, and Luc Van Gool. LSDIR: A large scale dataset for image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 1775–1787, June 2023. URL https://openaccess.thecvf.com/content/CVPR2023W/NTIRE/html/Li_LSDIR_A_Large_Scale_Dataset_for_Image_Restoration_CVPRW_2023_paper.html.

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 *Computer Vision – ECCV 2014*, volume 8693 of *Lecture Notes in Computer Science*, pp. 740–755. Springer, 2014. doi: 10.1007/978-3-319-10602-1_48. URL https://doi.org/10.1007/978-3-319-10602-1_48.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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 (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/7711026951f3dc3afb2be427bf98bc73-Abstract-Conference.html.

Ségolène Martin, Anne Gagneux, Paul Hagemann, and Gabriele Steidl. PnP-flow: Plug-and-play image restoration with flow matching. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.02423.

Xiangming Meng and Yoshiyuki Kabashima. Diffusion model based posterior sampling for noisy linear inverse problems. In *Asian Conference on Machine Learning (ACML)*, 2025. URL https://proceedings.mlr.press/v260/meng25a.html. PMLR 260:623–638; arXiv:2211.12343.

Badr Moufad, Navid Bagheri Shouraki, Alain Oliviero Durmus, Thomas Hirtz, Eric Moulines, Jimmy Olsson, and Yazid Janati. Efficient zero-shot inpainting with decoupled diffusion guidance. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=5F93RfQ12T. Poster presentation.

George Papandreou and Alan L. Yuille. Gaussian sampling by local perturbations. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 23. Curran Associates, Inc., 2010. URL https://proceedings.neurips.cc/paper/2010/hash/d09bf41544a3365a46c9077ebb5e35c3-Abstract.html.

Maitreya Patel, Song Wen, Dimitris N. Metaxas, and Yezhou Yang. FlowChef: Steering of rectified flow models for controlled generations. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. Preprint titled *Steering Rectified Flow Models in the Vector Field for Controlled Image Generation*, arXiv:2412.00100.

Ron Raphaeli, Sean Man, and Michael Elad. SILO: Solving inverse problems with latent operators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10570–10580, October 2025. doi: 10.1109/ICCV51701.2025.00984. URL https://openaccess.thecvf.com/content/ICCV2025/html/Raphaeli_SILO_Solving_Inverse_Problems_with_Latent_Operators_ICCV_2025_paper.html.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022. doi: 10.1109/CVPR52688.2022.01042. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alexandros G. Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 49960–49990. Curran Associates, Inc., 2023. doi: 10.52202/075280-2174. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/9c70cfa2e7d9328c649c94d50cbf8faf-Abstract-Conference.html.

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 (CVPR)*, pp. 9472–9481, June 2024. doi: 10.1109/CVPR52733.2024.00905. URL https://openaccess.thecvf.com/content/CVPR2024/html/Rout_Beyond_First-Order_Tweedie_Solving_Inverse_Problems_using_Latent_Diffusion_CVPR_2024_paper.html.

SG161222. Realistic vision v5.1 noVAE. Hugging Face model repository, 2023. URL https://huggingface.co/SG161222/Realistic_Vision_V5.1_noVAE. Accessed September 5, 2026.

Abduragim Shtanchaev, Albina Ilina, Yazid Janati, Arip Asadulaev, Martin Takáč, and Eric Moulines. Sparse scheduled diffusion guidance for inverse problems. *arXiv preprint arXiv:2603.07860*, 2026. doi: 10.48550/arXiv.2603.07860. URL https://arxiv.org/abs/2603.07860.

Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=j8hdRqOUhN.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

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

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pp. 11918–11930. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/3001ef257407d5a371a96dcd947c7d93-Abstract.html.

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 (ICLR)*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Alessio Spagnoletti, Tim Y. J. Wang, Marcelo Pereyra, and O. Deniz Akyildiz. Consistency regularised gradient flows for inverse problems. *arXiv preprint arXiv:2605.07907*, 2026. doi: 10.48550/arXiv.2605.07907. URL https://arxiv.org/abs/2605.07907.

Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *International Conference on Learning Representations (ICLR)*, 2023. Notable-top-25%; arXiv:2212.00490.

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a. arXiv:2407.01521.

Jiawei Zhang, Ziyuan Liu, Leon Yan, Gen Li, and Yuantao Gu. Improving diffusion-based inverse algorithms under few-step constraint via linear extrapolation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 35117–35171, 2025b. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/2ca73f0633760e2e2740ccb2a0d3f8a1-Abstract-Conference.html.

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 (CVPR)*, pp. 586–595, June 2018. doi: 10.1109/CVPR.2018.00068. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool. Denoising diffusion models for plug-and-play image restoration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2023. arXiv:2305.08995.