BETTER TRAJECTORIES, NOT MORE: PAIRED POSTERIOR CORRECTION FOR FLOW-MATCHING INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose *PPC-Flow* (Paired Posterior Correction), a training-free flow-matching solver for linear image inverse problems. The strongest flow-matching solvers re-inject noise and average independent trajectories, which multiplies their cost and gives up the speed advantage of flow priors. We trace their error to two sources, a bias that builds up late in the trajectory and the variance of the re-injected noise. The bias arises because the proximal weight follows a fixed schedule that vanishes near the terminal time, which leaves the pull of the network toward the prior uncorrected in the final steps. Averaging independent trajectories reduces only the variance, and only at the Monte Carlo rate. *PPC-Flow* counters the bias with a posterior velocity correction, which scales up the proximal correction, and suppresses the variance by antithetic pairing of the re-injection noise, both without additional network evaluations. On the CelebA and AFHQ-Cat benchmarks, *PPC-Flow* achieves state-of-the-art PSNR and SSIM among flow-matching solvers while using only 40% of the NFE budget. Code and video are available at https://anonymous.4open.science/r/ppc-flow-C40F.

1 INTRODUCTION

Image inverse problems are central to computational imaging and computer vision (Li et al., 2025), arising in medical imaging, microscopy, and computational photography (Lustig et al., 2007; McCann & Unser, 2019; Ongie et al., 2020). They include standard tasks such as denoising, deblurring, super-resolution, and inpainting, all modeled by the forward measurement

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \eta, \qquad \eta \sim \mathcal{N}(\mathbf{0}, \sigma_n^2 I), \tag{1}$$

with a known linear forward operator $\mathbf{H} \in \mathbb{R}^{m\times d}$. The maximum-a-posteriori (MAP) estimator

$$\hat{\mathbf{x}}_{\mathrm{MAP}} = \arg\min_{\mathbf{x}} \Big\{ \tfrac{1}{2\sigma_n^2} \|\mathbf{H}\mathbf{x} - \mathbf{y}\|^2 \; - \; \log p(\mathbf{x}) \Big\} \tag{2}$$

combines the explicit Gaussian likelihood with an image prior $\log p(\mathbf{x})$ that is intractable for natural images. Classical regularizers and plug-and-play methods substitute the prior step with a pretrained denoiser (Hurault et al., 2022; Romano et al., 2017; Venkatakrishnan et al., 2013). Modern training-free solvers extend this to generative priors, where one pretrained model serves many forward operators, provided the per-image solver budget stays small enough to make sharing worthwhile. In diffusion models, measurement information is typically incorporated during sampling via gradient guidance, projection, or proximal updates (Chung et al., 2023; Kawar et al., 2022; Mardani et al., 2024; Song et al., 2023; Wang et al., 2023; Zhu et al., 2023), which we review in Section 4.

Flow matching offers a different route to training-free inverse solving. Unlike diffusion samplers, which require many score evaluations, a flow-matching velocity network (Albergo et al., 2025; Lipman et al., 2023; Liu et al., 2023; Tong et al., 2024) transports noise to data along an ODE trajectory in far fewer steps. The main cost is therefore the number of velocity-network evaluations.

The OT-flow parameterization (Lipman et al., 2023; Tong et al., 2024) further exposes a useful endpoint structure. At any intermediate state $\mathbf{x}_t$, the velocity directly gives the predicted endpoint $m_{\theta,t} = \mathbf{x}_t + (1-t)v_\theta(\mathbf{x}_t, t)$, the flow counterpart of the denoised estimate in diffusion solvers (Pokle

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.

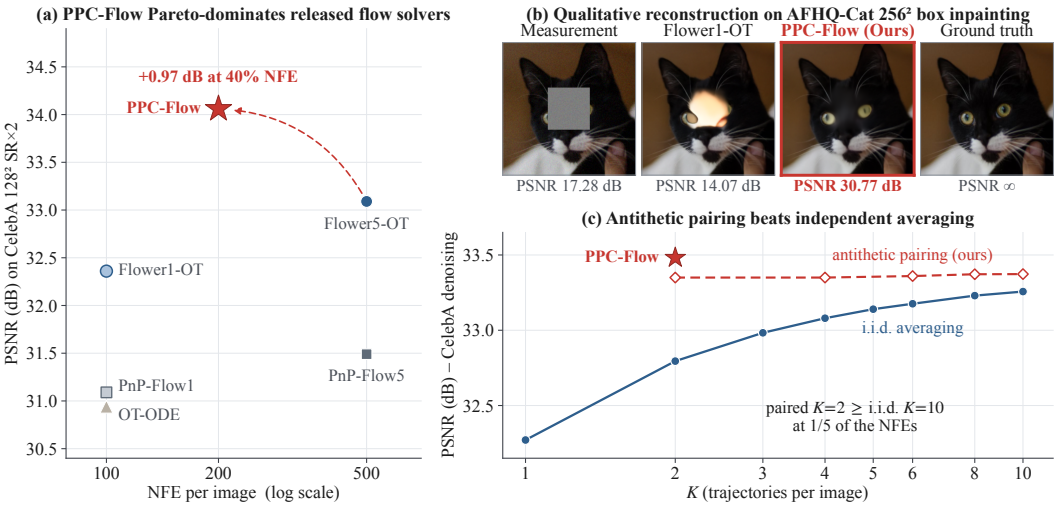

Figure 1: *PPC-Flow* spends network evaluations on correcting and pairing trajectories instead of averaging independent ones. **(a)** Compute and quality. With the velocity correction, *PPC-Flow* at $2T$ reaches 34.06 dB on CelebA SR×2, 0.97 dB above Flower5-OT at $5T$. **(b)** Hard AFHQ-Cat box-inpainting case. **(c)** On CelebA denoising, antithetic pairing with $K = 2$ exceeds independent averaging with $K = 10$ by reducing the variance of the terminal reconstruction.

et al., 2024; Pourya et al., 2026). This property enables flow-based solvers to directly incorporate measurement guidance, bypassing the Tweedie-type score-to-endpoint conversion (Efron, 2011; Robbins, 1956; Song et al., 2023) required by diffusion models. Existing methods exploit this structure through conditional velocities, flow-map optimization, MAP subproblems, or endpoint denoising (Ben-Hamu et al., 2024; Kim et al., 2025; Martin et al., 2025; Pokle et al., 2024; Pourya et al., 2026; Zhang et al., 2024).

However, fast single-trajectory sampling does not make reconstruction cheap. PnP-Flow and Flower re-inject noise during reconstruction to map corrected estimates back onto the flow path, and average independently sampled estimates to reduce the resulting variance (Martin et al., 2025; Pourya et al., 2026). This averaging spends the compute that flow matching saves. It is also an inefficient remedy, since it reduces the variance only at the Monte Carlo rate $1/K$ and leaves any bias shared by all estimates untouched. This raises a sharper question. At a fixed number of network function evaluations (NFEs), should the extra compute buy more independent samples, or better ones?

We answer it by tracing the error of these solvers to two sources, a late-stage bias and the variance of the re-injected noise. The velocity network pulls the predicted endpoint back toward the prior. While the proximal step is designed to counter this pull, its step size decays with the prior uncertainty rather than the residual misfit, so it vanishes near the terminal time. Yet the final steps are the ones that count, because the solver forgets corrections made earlier in the trajectory. The reconstruction therefore keeps a bias toward the prior. The second source is the variance of the re-injected noise, which independent averaging reduces only at the rate $1/K$. Treating both sources yields substantial gains in reconstruction fidelity and compute efficiency (Fig. 1).

We propose *PPC-Flow*, which counters the late-stage bias and suppresses the variance of the re-injected noise without any additional network evaluation. For the bias, a posterior velocity correction uses the measurement residual of the predicted endpoint to adjust the velocity before the proximal step. We prove that the velocity correction followed by the proximal step is an over-relaxed proximal step with factor $\omega_t \geq 1$, so the endpoint overshoots the ordinary proximal solution and counters the next pull toward the prior. We further derive a closed-form cap on ω_t that keeps every observed component contractive. Unlike gradient-based guidance, the velocity correction requires no network Jacobian and no covariance estimate. For the variance, antithetic pairing runs two trajectories with sign-flipped re-injection noise. By the classical antithetic-variates identity (Hammersley & Morton, 1956; Owen, 2013), the paired average has lower variance than the average of two independent trajectories whenever the odd part of the noise response dominates the even part. We verify by direct measurement that this condition holds with a wide margin.

Our contributions are as follows.

1. A bias-variance analysis of stochastic flow-matching inverse solvers. We trace their error to a late-stage bias and the variance of the re-injected noise, and we measure both directly.
2. *PPC-Flow*, a solver that counters the bias with a posterior velocity correction and suppresses the variance with antithetic pairing, at no extra network cost. We prove that the velocity correction over-relaxes the proximal step and derive a closed-form cap that keeps every observed residual component contractive.
3. State-of-the-art PSNR and SSIM among flow-matching solvers on CelebA and AFHQ-Cat, obtained with only 40% of the NFE budget.

2 Background

2.1 Flow Matching with the OT Path

Let $\mathbf{x}_1 \sim p_{\text{data}}$ denote a clean data sample and $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, I)$ denote the initial Gaussian latent. In the optimal-transport (OT) flow used throughout this paper (Lipman et al., 2023; Tong et al., 2024), the latent state at time $t \in [0, 1]$ is

$$\mathbf{x}_t = (1 - t)\mathbf{x}_0 + t\mathbf{x}_1, \tag{3}$$

which moves along a straight line from Gaussian noise at $t{=}0$ to clean data at $t{=}1$. We call this path the OT bridge. Along it, the target velocity is the conditional expectation

$$v^\star(\mathbf{x}_t, t) := \mathbb{E}[\mathbf{x}_1 - \mathbf{x}_0 \mid \mathbf{x}_t], \tag{4}$$

and flow matching trains a velocity network v_θ to approximate $v^\star$. Sampling integrates the ODE

$$\frac{d\mathbf{x}_t}{dt} = v_\theta(\mathbf{x}_t, t), \qquad \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, I). \tag{5}$$

Along the OT bridge, the velocity also gives a direct estimate of the clean endpoint,

$$m_{\theta,t} := \mathbf{x}_t + (1 - t)v_\theta(\mathbf{x}_t, t), \tag{6}$$

which equals the conditional mean $\mathbb{E}[\mathbf{x}_1 \mid \mathbf{x}_t]$ when $v_\theta = v^\star$. This predicted endpoint is the prior guess that an inverse solver corrects toward the measurement $\mathbf{y}$.

2.2 Bayesian Setup of Linear Inverse Problems

Training-free inverse solving. We observe

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \eta, \qquad \eta \sim \mathcal{N}(\mathbf{0}, \sigma_n^2\mathbf{I}), \tag{7}$$

and seek a reconstruction that is both consistent with the measurement and plausible under the pretrained flow prior. The Gaussian noise model in Eq. (7) gives the likelihood $p(\mathbf{y} \mid \mathbf{x}_1) \propto \exp\!\big(-\|\mathbf{y} - \mathbf{H}\mathbf{x}_1\|^2/2\sigma_n^2\big)$. With the data prior p_{data}, the Bayes rule gives the posterior target

$$p(\mathbf{x}_1 \mid \mathbf{y}) \;\propto\; \underbrace{p(\mathbf{y} \mid \mathbf{x}_1)}_{\text{Gaussian likelihood}} \; \underbrace{p_{\text{data}}(\mathbf{x}_1)}_{\text{flow prior}} \;=\; \exp\!\left(-\frac{\|\mathbf{y} - \mathbf{H}\mathbf{x}_1\|^2}{2\sigma_n^2}\right) p_{\text{data}}(\mathbf{x}_1). \tag{8}$$

Since the data prior is available only through the velocity network, training-free flow solvers approximate the posterior velocity or endpoint implied by Eq. (8).

Stochastic flow inverse solvers and per-image evaluation. A stochastic flow inverse solver draws its randomness $\tau := (\mathbf{x}_0, \varepsilon_0, \ldots, \varepsilon_{T-1})$, which consists of the initial latent and the re-injection noise of T integration steps. A multi-trajectory solver runs K independent copies $\tau^{(1)}, \ldots, \tau^{(K)}$, each consuming T calls to v_θ, and returns the average of their terminal reconstructions

$$\hat{\mathbf{x}}_{\text{out}} \;=\; \frac{1}{K}\sum_{k=1}^{K} \mathbf{x}_T^{(k)}, \tag{9}$$

where $\mathbf{x}_T^{(k)}$ is the k-th terminal reconstruction. This independent averaging is the standard multi-trajectory baseline of state-of-the-art flow-matching inverse solvers such as PnP-Flow and

Flower (Martin et al., 2025; Pourya et al., 2026), typically with $K{=}5$. We measure the cost of a flow solver by its NFE budget $B = KT$ and report the operator overhead separately in Appendix B.3. For a single image, the mean squared error then splits into bias and variance,

$$\mathbb{E}_\tau\big[\|\hat{\mathbf{x}}_{\text{out}} - \mathbf{x}_1\|^2 \,\big|\, \mathbf{x}_1, \mathbf{y}\big] = \underbrace{\big\|\mathbb{E}_\tau[\hat{\mathbf{x}}_{\text{out}} \mid \mathbf{x}_1, \mathbf{y}] - \mathbf{x}_1\big\|^2}_{\text{bias}} + \underbrace{\mathrm{Var}_\tau[\hat{\mathbf{x}}_{\text{out}} \mid \mathbf{x}_1, \mathbf{y}]}_{\text{trajectory variance}}, \tag{10}$$

with $\mathrm{Var}_\tau[\,\cdot\,] := \mathrm{tr}\,\mathrm{Cov}_\tau[\,\cdot\,]$. Conditioning on $(\mathbf{x}_1, \mathbf{y})$ isolates the error of the solver. The bias is shared by all trajectories, whereas the variance comes from the initial latent and the re-injection noise. Independent averaging reduces only the variance, and only at the Monte Carlo rate $1/K$. Antithetic variates (Hammersley & Morton, 1956; Owen, 2013) instead average negatively correlated pairs, and Section 3.3 applies this idea to the re-injection noise.

3 Method

PPC-Flow works within a proximal reconstruction framework (Pourya et al., 2026; Xu & Chi, 2024) and targets the two terms of Eq. (10) separately. Posterior velocity correction counters the bias by strengthening the proximal step, and antithetic pairing suppresses the variance by driving two trajectories with sign-flipped re-injection noise. The three steps below run on both trajectories at every time t, starting from independent initial latents, and Algorithm 1 gives the full procedure. Each step evaluates the velocity network once per trajectory, so a solver with K trajectories and T integration steps uses KT NFEs. These evaluations dominate the runtime, and beyond them *PPC-Flow* adds only one application each of $\mathbf{H}$ and $\mathbf{H}^\top$ per step and a single offline power iteration per run (Appendix B.3).

1. **Posterior velocity correction.** At each step, the velocity network gives the predicted endpoint $m_{\theta,t}$ of Eq. (6). We use its measurement residual, $\mathbf{y} - \mathbf{H}m_{\theta,t}$, to correct the velocity without differentiating through the network, which gives

$$v_{\text{corr}}(\mathbf{x}_t, t) = v_\theta(\mathbf{x}_t, t) + \alpha_t \frac{\mathbf{H}^\top(\mathbf{y} - \mathbf{H}\, m_{\theta,t})}{\sigma_n^2}, \quad \hat{\mathbf{x}}_1^{\textit{PPC-Flow}} = \mathbf{x}_t + (1-t)\, v_{\text{corr}}(\mathbf{x}_t, t), \tag{11}$$

   where $\alpha_t \geq 0$ controls the correction strength. Section 3.2 gives its schedule and shows that this step followed by the proximal step equals the ordinary proximal correction multiplied by a factor $\omega_t \geq 1$.

2. **Proximal data-consistency correction.** The corrected endpoint is anchored to the observation through the proximal solve

$$\mathbf{x}^\star = \mathrm{prox}_{\lambda_t F_{\mathbf{y}}}\big(\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}\big), \qquad F_{\mathbf{y}}(\mathbf{x}) := \tfrac{1}{2\sigma_n^2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2, \tag{12}$$

   which reduces to a linear system that we solve with conjugate gradients (Pourya et al., 2026). The re-injection step (Eq. (13)) then maps $\mathbf{x}^\star$ to $\mathbf{x}_{t+\delta}$, and Section 3.1 recalls the proximal step and its contraction property.

3. **Re-injection step with antithetic pairing.** Following the re-injection strategy of flow-based inverse solvers (Martin et al., 2025; Pourya et al., 2026), we advance each trajectory to time $t{+}\delta$ by interpolating the proximal-corrected endpoint $\mathbf{x}^\star$ with fresh noise $\varepsilon_t \sim \mathcal{N}(\mathbf{0}, I)$ along the OT bridge. Antithetic pairing assigns the sign-flipped noise sequences (ε_t) and $(-\varepsilon_t)$ to two trajectories that share the same update rule and step schedule. The two updates read

$$\begin{aligned} \mathbf{x}^+_{t+\delta} &\leftarrow (t+\delta)\, \mathbf{x}^+_{\star,t} + (1-t-\delta)\, \varepsilon_t, \\ \mathbf{x}^-_{t+\delta} &\leftarrow (t+\delta)\, \mathbf{x}^-_{\star,t} - (1-t-\delta)\, \varepsilon_t. \end{aligned} \tag{13}$$

   The next iterates $\mathbf{x}^\pm_{t+\delta}$ re-enter step 1 at time $t{+}\delta$, and the terminal reconstruction is the paired average $\hat{\mathbf{x}} = \frac{1}{2}(\mathbf{x}^+_{\text{term}} + \mathbf{x}^-_{\text{term}})$, where $\mathbf{x}^\pm_{\text{term}} := \mathbf{x}^\pm_{t=1}$. This step targets the trajectory-variance term of Eq. (10), and Section 3.3 gives the corresponding variance-reduction result (Proposition 2).

Fig. 2 sketches the two mechanisms. We first recall the proximal step as a reference, then analyze how the velocity correction modifies it and how pairing affects trajectory variance.

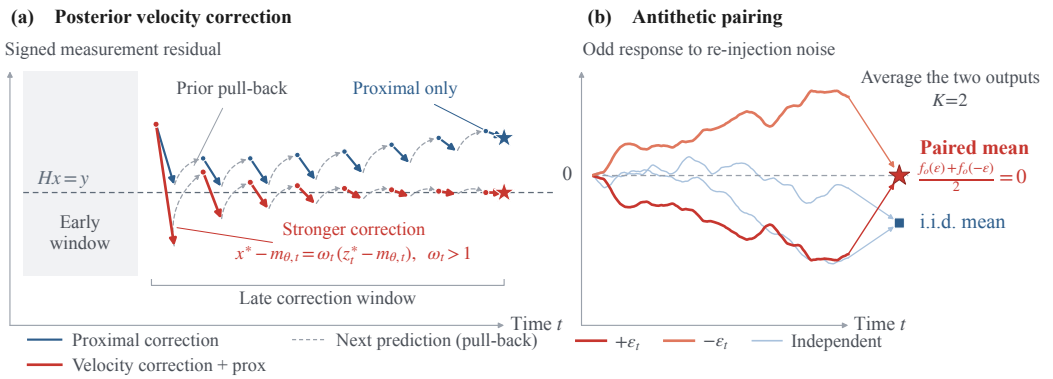

Figure 2: Mechanism of *PPC-Flow* (schematic). (a) Velocity correction enlarges the ordinary proximal displacement (red versus blue solid arrows), counteracting the prior pull-back at the next prediction (dashed arrows). The dashed horizontal line marks data consistency, $\mathbf{Hx} = \mathbf{y}$, and the early window is shaded and its trajectories omitted. (b) Opposite re-injection noise produces sign-reversed odd responses (the two red curves), which cancel in the terminal paired average (star). Light-blue curves and the square show the independent-averaging reference at the same $K = 2$.

3.1 Preliminary: the inherited proximal data-consistency step

Proximal methods are commonly used to enforce data consistency in inverse problems (Chung et al., 2024; Pourya et al., 2026; Xu & Chi, 2024). The standard quadratic proximal step balances measurement fidelity against proximity to the corrected prediction $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$ and reads

$$\mathbf{x}^\star = \arg\min_{\mathbf{z}} \left\{ \tfrac{1}{2\sigma_n^2} \|\mathbf{y} - \mathbf{Hz}\|^2 + \tfrac{1}{2\lambda_t} \|\mathbf{z} - \hat{\mathbf{x}}_1^{\textit{PPC-Flow}}\|^2 \right\}, \tag{14}$$

where λ_t sets how far the step moves toward the measurement, and smaller values keep the output closer to its input. Appendix A.4 shows how this step contracts the residual.

3.2 Bias: posterior velocity correction

The proximal step alone leaves the reconstruction biased toward the prior, and Fig. 2(a) shows how this bias arises. The proximal step moves the predicted endpoint toward the data (solid arrows), but the next network evaluation pulls it back toward a prediction favored by the learned image prior (dashed arrows). Like a stretched spring recoiling, this pull-back partially undoes the preceding data-consistency correction. Two properties of the solver keep it from correcting this bias. First, the proximal weight follows a fixed schedule. The existing proximal formulation sets λ_t to the variance $r_t := (1-t)^2/[t^2 + (1-t)^2]$ of a local Gaussian endpoint approximation (Pourya et al., 2026) (Appendices A.1 and A.2), and we retain this schedule. Because this variance vanishes as $t \to 1$, the proximal correction fades near the terminal time regardless of the remaining misfit. Second, the solver forgets early corrections. In a Gaussian bridge model, an early perturbation reaches the terminal state multiplied by a product of per-step retention factors (Lemma 2), and our time-window ablation confirms that only late corrections measurably change the terminal reconstruction (Tab. 12). Both properties therefore call for a stronger correction in the late steps.

To isolate the velocity correction, we disable it and pass the unmodified prediction $m_{\theta,t}$ to Eq. (14) in place of $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$. For $\lambda_t = r_t$, the resulting endpoint is the ordinary proximal solution $z_t^\star = \arg\min_z \Phi_t(z)$ with the objective (Parikh & Boyd, 2014; Stuart, 2010)

$$\Phi_t(z) \;:=\; \frac{1}{2\sigma_n^2}\|\mathbf{y} - \mathbf{H}z\|^2 \;+\; \frac{1}{2r_t}\|z - m_{\theta,t}\|^2. \tag{15}$$

The ordinary proximal solution $z_t^\star$ also admits a probabilistic interpretation. If the endpoint distribution is approximated by a Gaussian with mean $m_{\theta,t}$ and covariance $r_t I$, then under the Gaussian observation model $z_t^\star$ is the maximum a posteriori (MAP) estimate. Pseudoinverse guidance (Song et al., 2023), OT-ODE (Pokle et al., 2024), and Flower (Pourya et al., 2026) use the same local Gaussian approximation, and Appendix A.2 gives this estimate in closed form.

Proposition 1 (Velocity correction as a scaled proximal step). *Let $0 \le t < 1$, $\sigma_n^2 > 0$, $\alpha_t \ge 0$, and $\lambda_t = r_t$ in Eq. (12). With an exact proximal solve, the anchored endpoint produced by Eqs. (11) and (12) equals*

$$\mathbf{x}^\star \;=\; m_{\theta,t} + \omega_t \left(z_t^\star - m_{\theta,t}\right), \qquad \omega_t \;:=\; 1 + \frac{(1-t)\,\alpha_t}{r_t}, \tag{16}$$

so $\alpha_t = 0$ recovers the ordinary proximal step and $\alpha_t > 0$ scales its displacement by $\omega_t > 1$. The velocity correction is therefore an over-relaxation of the proximal step with factor ω_t.

Let $\sigma_j > 0$ be a nonzero singular value of $\mathbf{H}$ and $w_j(t) := r_t\sigma_j^2/(r_t\sigma_j^2 + \sigma_n^2)$ the corresponding proximal gain. The update multiplies that component of the residual $\mathbf{y} - \mathbf{H}m_{\theta,t}$ by $1 - \omega_t w_j(t)$. It contracts every observed component if and only if $\omega_t w_{\max}(t) < 2$, where $w_{\max}(t) := \max_j w_j(t)$.

The velocity correction therefore keeps the direction of the proximal correction and scales its length by ω_t. The extra length is a deliberate overshoot. The endpoint moves past the ordinary proximal solution, so that the next pull toward the prior brings the prediction onto the data instead of leaving it on the prior side. The overshoot is proportional to $1 - t$ and lies in the range of $\mathbf{H}^\top$, so it vanishes at the terminal boundary and leaves the null-space components, which the generative prior determines, untouched. Measured endpoint errors confirm this mechanism (Fig. 5), and Section 5.2 shows that the velocity correction lowers the bias term of Eq. (10) while leaving the variance essentially unchanged.

We use the bridge-aligned schedule

$$\alpha_t = \min\{a,\ \bar{a}_t\}\, r_t, \qquad \text{so that} \qquad \omega_t = 1 + (1-t)\min\{a,\ \bar{a}_t\}, \tag{17}$$

where $a \ge 0$ is the correction strength. The schedule strengthens the proximal step in the late window and tapers the extra correction to zero at the terminal boundary. The cap $\bar{a}_t$ enforces $\omega_t w_{\max}(t) \le 1.9$, below the threshold of 2 in Proposition 1, which guarantees that every observed residual component contracts at every step (Appendix A.2). The cap is essential in practice. Without it, sufficiently strong corrections cross the contraction threshold and diverge, whereas the gain of the capped schedule saturates (Appendix A.3).

3.3 Variance: Antithetic Pairing

At the same two-trajectory budget, *PPC-Flow* replaces independent averaging with sign-flipped re-injection noise (Eq. (13)). Pairing leaves the marginal distribution of each trajectory, and hence the mean reconstruction, unchanged, so it changes only the variance term in Eq. (10). This is classical antithetic sampling (Hammersley & Morton, 1956; Owen, 2013) on the full re-injection sequence.

For a fixed observation $\mathbf{y}$, let $\boldsymbol{\varepsilon} = (\varepsilon_0, \dots, \varepsilon_{T-1})$ be the re-injection sequence with a symmetric law $(-\boldsymbol{\varepsilon} \overset{d}{=} \boldsymbol{\varepsilon})$, and let ζ collect independent randomness, including the initial latent. Write the square-integrable terminal reconstruction as $\mathbf{x}_T(\boldsymbol{\varepsilon}; \zeta)$. Averaging over ζ gives $f(\boldsymbol{\varepsilon}) := \mathbb{E}_\zeta[\mathbf{x}_T(\boldsymbol{\varepsilon}; \zeta)]$ and the parity decomposition

$$f(\boldsymbol{\varepsilon}) \;=\; f_e(\boldsymbol{\varepsilon}) + f_o(\boldsymbol{\varepsilon}), \qquad f_e(-\boldsymbol{\varepsilon}) = f_e(\boldsymbol{\varepsilon}), \quad f_o(-\boldsymbol{\varepsilon}) = -f_o(\boldsymbol{\varepsilon}), \tag{18}$$

where f_e is unchanged by a sign flip and f_o changes sign, so the latter cancels in the paired average illustrated in Fig. 2(b). These components are orthogonal under the symmetric law. We write $\mathrm{Var}(\cdot) := \mathrm{tr}\,\mathrm{Cov}(\cdot)$ for trace variance.

Proposition 2 (Antithetic variance identity for paired trajectories). *The paired and i.i.d. estimators at matched compute,*

$$T_p := \tfrac{1}{2}\big(\mathbf{x}_T(\boldsymbol{\varepsilon}; \zeta_+) + \mathbf{x}_T(-\boldsymbol{\varepsilon}; \zeta_-)\big),$$
$$T_i := \tfrac{1}{2}\big(\mathbf{x}_T(\boldsymbol{\varepsilon}_a; \zeta_a) + \mathbf{x}_T(\boldsymbol{\varepsilon}_b; \zeta_b)\big),$$

with (ζ_+, ζ_-) independent and $(\boldsymbol{\varepsilon}_a, \zeta_a), (\boldsymbol{\varepsilon}_b, \zeta_b)$ i.i.d., satisfy the exact identity

$$\mathrm{Var}(T_i) - \mathrm{Var}(T_p) \;=\; \tfrac{1}{2}\big(\mathrm{Var}(f_o) - \mathrm{Var}(f_e)\big) \;=\; -\tfrac{1}{2}\,\mathrm{tr}\,\mathrm{Cov}\big(f(\boldsymbol{\varepsilon}),\ f(-\boldsymbol{\varepsilon})\big). \tag{19}$$

In particular, pairing strictly reduces trace variance if and only if

$$\mathrm{tr}\,\mathrm{Cov}\big(f(\boldsymbol{\varepsilon}),\ f(-\boldsymbol{\varepsilon})\big) \;<\; 0, \qquad \textit{equivalently} \qquad \mathrm{Var}(f_o) \;>\; \mathrm{Var}(f_e). \tag{20}$$

Pairing cancels the odd component of the averaged response, while independently drawn initial latents contribute equally to both estimators. Appendix A.4.1 proves the identity in our setting. The condition reflects the response symmetry of the solver. Odd-order terms in the expansion of the trajectory response belong to f_o and even-order terms to f_e, so pairing reduces the variance whenever the response to the re-injection noise is predominantly linear. We measure both variances directly and find that the condition holds with a wide margin in every setting, and the identity then predicts the measured gains of pairing over independent averaging (Appendix B.11). Pairing the initial latents has no effect, since the solver forgets them (Tab. 12), and the gain of pairing persists with and without the velocity correction (Appendix B.9 and Tab. 17).

4 Related Work

Training-free solvers with diffusion priors. Diffusion solvers use the measurement at sampling time in three ways. Guidance methods differentiate a data-fidelity loss through the denoised estimate (Chung et al., 2022; 2023; Rout et al., 2023; Song et al., 2023), and later work refines the resulting gradient (Boys et al., 2024; He et al., 2024; Rout et al., 2024; Zirvi et al., 2025). Most of these methods back-propagate through the network at every step. Projection methods avoid this cost by exploiting the SVD, null space, or mask of $\mathbf{H}$ (Kawar et al., 2021; 2022; Lugmayr et al., 2022; Wang et al., 2023), but they need such a decomposition of $\mathbf{H}$ to be available. Splitting methods (Chung et al., 2024; Xu & Chi, 2024; Zhang et al., 2025; Zhu et al., 2023) build on plug-and-play and half-quadratic schemes (Geman & Yang, 1995; Romano et al., 2017; Venkatakrishnan et al., 2013) and alternate a prior step with a data step. The data step needs only products with $\mathbf{H}$ and $\mathbf{H}^\top$, and *PPC-Flow* follows the same design. These works improve a single sampling path, whereas we study how to split a fixed NFE budget between correcting one path and averaging several.

Flow-matching inverse solvers. Flow models with straight paths give a predicted endpoint at every state (Lipman et al., 2023; Liu et al., 2023; Tong et al., 2024), so the measurement can act on the endpoint directly. OT-ODE and FlowDPS adapt pseudoinverse and posterior-sampling guidance to flows (Kim et al., 2025; Pokle et al., 2024). D-Flow optimizes the initial latent through the ODE solve (Ben-Hamu et al., 2024) and Flow-Priors uses the trace of the velocity Jacobian (Zhang et al., 2024), so both cost substantially more than a single sampling pass. PnP-Flow uses the flow as a plug-and-play denoiser (Martin et al., 2025), LFlow works in a latent space (Askari et al., 2025), and Flower alternates endpoint prediction, a proximal step, and noise re-injection (Pourya et al., 2026). Flower is the closest work to ours. Its refinement is the ordinary proximal step, the special case $\omega_t{=}1$ of Proposition 1. We show that this step fades near the terminal time, exactly where it matters, and that a capped over-relaxation restores it without extra network evaluations. Over-relaxation itself is classical in linear solvers and proximal algorithms (Eckstein & Bertsekas, 1992; Young, 1971).

Averaging and variance reduction. Stochastic solvers average K independent evaluations or trajectories (Martin et al., 2025; Pourya et al., 2026), which costs K times more and reduces the variance only at the rate $1/K$. Antithetic variates instead average negatively correlated pairs (Hammersley & Morton, 1956; Owen, 2013). Ruipérez-Campillo et al. (2026) pair the per-step noise of a conditional diffusion model trained to denoise cardiac signals. We carry this classical variance reduction over to training-free flow solvers, where the paired average serves as the reconstruction itself. The position of the pairing is decisive. The solver forgets its initial latent, so pairing the initial noise has no effect, whereas pairing the re-injection noise cancels the odd component of the response and lowers the variance (Tab. 12 and Appendix B.11).

5 Experiments

5.1 Setup

Datasets and tasks. We evaluate on the standard OT-flow inverse benchmark (Martin et al., 2025; Pourya et al., 2026), which covers CelebA (128^2) (Liu et al., 2015) and AFHQ-Cat (256^2) (Choi et al., 2020) and five linear inverse problems, namely denoising, Gaussian deblurring, super-resolution, random inpainting, and box inpainting. We report PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) on the released 100-image test splits (Appendix B.1).

Table 1: Quantitative comparison on CelebA 128^2 (100 test images). The best value in each column is bold and the second best is underlined.

Method	Denoising			Deblurring			SR×2			Rand inp			Box inp		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Degraded	20.00	0.348	0.372	27.83	0.740	0.126	10.26	0.183	0.827	11.95	0.196	1.041	22.27	0.742	0.214
PnP-GS	32.64	0.910	0.035	34.03	0.924	0.041	31.32	0.892	0.065	29.23	0.876	0.069	—	—	—
DiffPIR	31.20	0.885	0.060	32.77	0.912	0.060	31.52	0.895	0.034	31.74	0.917	0.025	—	—	—
OT-ODE	30.54	0.859	**0.032**	33.01	0.921	0.029	30.94	0.895	0.034	28.68	0.871	0.051	29.40	0.920	0.038
D-Flow	26.03	0.607	0.093	31.27	0.858	0.038	30.46	0.844	**0.027**	33.65	0.942	**0.015**	30.60	0.898	0.027
Flow-Priors	29.34	0.768	0.134	31.54	0.858	0.056	28.41	0.715	0.100	32.88	0.943	0.019	30.07	0.858	0.048
PnP-Flow1	31.80	0.905	0.044	34.48	0.936	0.040	31.09	0.902	0.045	33.05	0.944	0.018	30.47	0.933	0.037
Flower1-OT	32.28	0.914	0.033	34.98	0.947	**0.026**	32.36	0.923	0.034	33.08	0.944	0.018	31.19	0.945	**0.022**
PnP-Flow5	32.30	0.911	0.056	34.80	0.941	0.047	31.49	0.907	0.056	33.98	0.953	0.022	31.09	0.940	0.043
Flower5-OT	33.14	0.926	0.038	35.67	0.954	0.032	33.09	0.932	0.040	33.95	0.953	0.020	31.87	0.952	0.024
***PPC-Flow* (ours)**	**33.48**	**0.930**	0.033	**35.89**	**0.956**	0.030	**34.06**	**0.945**	0.029	**34.91**	**0.961**	0.018	**32.08**	**0.953**	0.023

Table 2: Quantitative comparison on AFHQ-Cat 256^2 (100 test images). The best value in each column is bold and the second best is underlined.

Method	Denoising			Deblurring			SR×4			Rand inp			Box inp		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Degraded	20.00	0.314	0.509	23.94	0.517	0.444	11.70	0.208	0.873	13.36	0.223	1.081	21.80	0.740	0.198
PnP-GS	32.58	0.894	**0.078**	28.14	0.759	0.344	24.26	0.637	0.353	29.59	0.838	0.119	—	—	—
DiffPIR	30.64	0.834	0.196	27.77	0.733	0.336	23.69	0.623	0.397	31.84	0.882	0.059	—	—	—
OT-ODE	30.13	0.816	0.079	27.26	0.718	**0.124**	26.12	0.720	**0.110**	29.57	0.842	0.086	24.56	0.874	0.093
D-Flow	26.20	0.574	0.169	28.06	0.730	0.166	24.79	0.606	0.180	32.31	0.894	0.044	26.42	0.841	0.082
Flow-Priors	29.42	0.760	0.151	26.66	0.705	0.185	23.69	0.573	0.263	32.35	0.905	0.052	26.24	0.817	0.117
PnP-Flow1	31.26	0.863	0.143	28.09	0.766	0.301	27.14	0.767	0.169	33.18	0.919	**0.040**	26.08	0.896	0.112
Flower1-OT	31.75	0.879	0.114	28.17	0.769	0.280	26.40	0.745	0.275	33.15	0.919	0.044	26.28	0.915	**0.066**
PnP-Flow5	31.50	0.864	0.177	28.39	0.771	0.329	**27.57**	0.778	0.179	33.90	0.929	0.048	26.88	0.900	0.129
Flower5-OT	32.40	0.890	0.128	28.58	0.778	0.296	26.71	0.753	0.285	33.84	0.928	0.050	26.93	0.922	0.069
***PPC-Flow* (ours)**	**32.92**	**0.904**	0.084	**28.71**	**0.786**	0.286	27.47	**0.782**	0.256	**34.26**	**0.934**	0.048	**27.13**	**0.924**	0.069

Baselines. The baselines, checkpoints, and evaluation splits are those released with PnP-Flow and Flower (Martin et al., 2025; Pourya et al., 2026), so the comparison follows the protocol these two papers established. We compare against PnP-GS (Hurault et al., 2022), DiffPIR (Zhu et al., 2023), OT-ODE (Pokle et al., 2024), D-Flow (Ben-Hamu et al., 2024), Flow-Priors (Zhang et al., 2024), the single-trajectory PnP-Flow1 and Flower1-OT, and their five-trajectory versions PnP-Flow5 and Flower5-OT. Three of them represent the diffusion and plug-and-play line of work on this benchmark. DiffPIR is a diffusion inverse solver, PnP-GS is a plug-and-play method with a learned denoiser, and OT-ODE is the flow counterpart of the pseudoinverse guidance of Song et al. (2023). The flow-based baselines share the OT-flow checkpoints, while PnP-GS and DiffPIR use the priors of the released benchmark. Appendix B.1 lists the checkpoints, operator settings, and step counts.

5.2 Main Results

On CelebA (Tab. 1), *PPC-Flow* achieves the highest PSNR and SSIM on all five tasks while using only 40% of the NFE budget of Flower5-OT ($2T$ vs. $5T$). Tab. 3 averages the five tasks. At $2T$, *PPC-Flow* outperforms both single-trajectory baselines at T and both five-trajectory baselines at $5T$, so it dominates multi-trajectory averaging in quality and cost at once. Fig. 3 shows the two effects on SR×2. Pairing shrinks the spread of the two-trajectory estimate, and velocity correction moves its center toward the clean image. The largest gains over Flower5-OT are on SR×2 and random inpainting (0.97 dB and 0.96 dB). Both tasks leave most pixels unobserved, and although velocity correction acts in the range of $\mathbf{H}^\top$, most of its error reduction lands on the unobserved pixels (Tab. 4). Box inpainting, whose unobserved region is a single contiguous block, still gains 0.21 dB. Paired-bootstrap tests (Appendix B.5) and an evaluation on $1{,}000$ images (Appendix B.6) confirm that these gains are significant and stable.

On AFHQ-Cat (Tab. 2), *PPC-Flow* outperforms Flower5-OT on all five tasks while using the same 40% of its NFE budget, and it achieves the highest SSIM on every task. The largest PSNR gains are on SR×4 ($+0.76$ dB) and denoising ($+0.52$ dB). The gains extend to perceptual quality, as *PPC-Flow* improves or matches the LPIPS of Flower5-OT on every task. In Fig. 4, *PPC-Flow* recovers sharper facial and fur structure with fewer artifacts, while D-Flow and Flow-Priors show visible artifacts or lost detail. Appendix B.13 extends the visual comparison to all ten task–dataset cells.

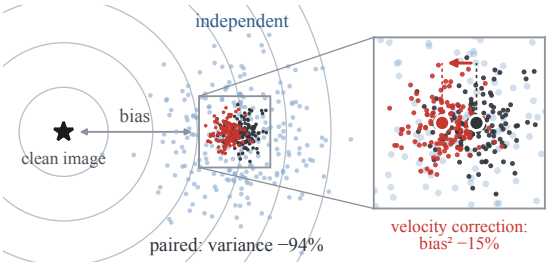

Figure 3: Measured bias and variance on CelebA SR×2, to scale. Dots are Gaussian samples with the moments of each $K=2$ estimator (Appendix B.11).

Table 3: CelebA 5-task mean (PSNR / SSIM) across NFE budgets.

Method	NFE	PSNR↑	SSIM↑
PnP-Flow1	T	32.18	0.924
Flower1-OT	T	32.78	0.935
PPC-Flow	$\mathbf{2T}$	**34.08**	**0.949**
PnP-Flow5	$5T$	32.73	0.930
Flower5-OT	$5T$	33.54	0.943

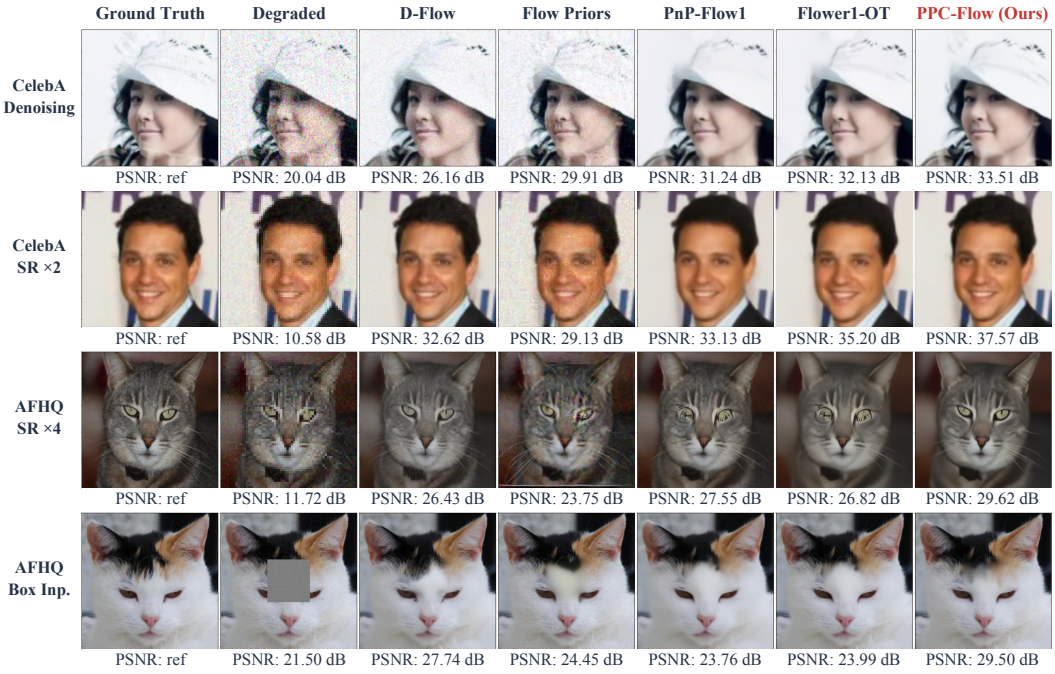

Figure 4: Qualitative comparison on four task–dataset cells (per-image PSNR below each tile).

Where the gains come from. A matched-budget ablation on CelebA at $2T$ separates the two components (Tab. 17). Pairing alone adds ≈ 0.5 dB on average, and velocity correction adds up to another 0.8 dB. The two gains add up, so the mechanisms act on separate error sources. Appendix B.7 repeats the velocity-correction ablation on all ten task–dataset cells. The bias-variance decomposition in Fig. 3 and Appendix B.11 shows why. A single trajectory spends 16–26% of its MSE on variance, and antithetic pairing leaves less than 2%. After pairing, the error is almost entirely bias, so further trajectories add little, and pairing two trajectories already beats averaging ten independent ones on denoising (Fig. 1(c)). Velocity correction, in turn, lowers the squared bias by up to 15% while leaving the variance essentially unchanged. Further ablations, wall-clock times, multi-seed runs, and extensions to nonlinear operators are reported in Appendices B.2, B.3, B.8 to B.10 and C.1 to C.3.

6 Conclusion

We introduced *PPC-Flow*, a training-free flow-matching solver for linear inverse problems. Our analysis examined the weakening of late-stage proximal correction and variance from noise re-injection. A posterior velocity correction addresses the first through a capped over-relaxation of the proximal step, and antithetic pairing suppresses the second, both without additional network evaluations. Our experiments demonstrated higher PSNR and SSIM than Flower5-OT across CelebA and AFHQ-Cat at 40% of its NFE budget, and the same steps extend to nonlinear operators (Appendix C.3).

AI USE STATEMENT

In this work, we used large-language-model assistants for error checking and for sentence-level editing and polishing of the manuscript. The research idea, the method, the derivations, the implementation, and the experiments are the authors' own work, and we have reviewed all AI-assisted text. We take responsibility for the final content of this work, including its text and claims.

ETHICS STATEMENT

This work proposes a training-free inference algorithm for linear image inverse problems and evaluates it on publicly available benchmark datasets (CelebA, AFHQ-Cat, and LSUN-Bedroom) under their respective licenses, using the released test splits and pretrained checkpoints of prior work. No new data were collected, no human subjects were involved, and no personally identifiable information beyond what is already contained in these public datasets is used or released. Image restoration methods, including inpainting, can be misused to alter images. Our method introduces no new capability of this kind beyond that of the pretrained generative priors it builds on, and it is evaluated only on standard benchmark degradations. The authors declare no conflicts of interest.

REPRODUCIBILITY STATEMENT

An anonymized implementation of *PPC-Flow*, including the evaluation scripts for every table and figure in the paper, is available at `https://anonymous.4open.science/r/ppc-flow-C40F`. All experiments reuse the publicly released OT-flow checkpoints and 100-image test splits of Pourya et al. (2026). The benchmark operators and noise levels, step counts, CG settings, antithetic pairing details, and RNG seeds are documented in Appendix B.1. Complete statements and proofs of Lemma 1, Theorem 1, and Propositions 1, 2 and 3 are given in Appendices A.1, A.2, A.4 and A.4.1, and the per-step pseudocode is given in Algorithm 1. Paired-bootstrap confidence intervals for all main-table comparisons are reported in Appendix B.5, and compute accounting and measured wall-clock times in Appendix B.3.

REFERENCES

Michael S. 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. arXiv:2303.08797; JMLR v26 (2025).

Hossein Askari, Yadan Luo, Hongfu Sun, and Fred Roosta. Latent refinement via flow matching for training-free linear inverse problem solving. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2511.06138.

Heli Ben-Hamu, Omri Puny, Itai Gat, Brian Karrer, Uriel Singer, and Yaron Lipman. D-Flow: Differentiating through flows for controlled generation. In *International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 3462–3483, 2024. arXiv:2402.14017.

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

Benjamin Boys, Mark Girolami, Jakiw Pidstrigach, Sebastian Reich, Alan Mosca, and Omer Deniz Akyildiz. Tweedie moment projected diffusions for inverse problems. *Transactions on Machine Learning Research (TMLR)*, 2024. arXiv:2310.06721.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8185–8194, 2020. doi: 10.1109/CVPR42600.2020.00821.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 25683–25696, 2022. arXiv:2206.00941.

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. arXiv:2209.14687.

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

Jonathan Eckstein and Dimitri P. Bertsekas. On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical Programming*, 55(1–3): 293–318, 1992.

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

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, 1993.

Donald Geman and Chengda Yang. Nonlinear image recovery with half-quadratic regularization. *IEEE Transactions on Image Processing*, 4(7):932–946, 1995.

John M. Hammersley and K. W. Morton. A new Monte Carlo technique: Antithetic variates. *Proceedings of the Cambridge Philosophical Society*, 52(3):449–475, 1956.

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 (ICLR)*, 2024. arXiv:2311.16424.

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.

Samuel Hurault, Arthur Leclaire, and Nicolas Papadakis. Proximal denoiser for convergent plug-and-play optimization with nonconvex regularization. In *International Conference on Machine Learning (ICML)*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9483–9505, 2022.

Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. arXiv:2105.14951.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 23593–23606, 2022. arXiv:2201.11793.

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 (ICCV)*, pp. 12328–12337, 2025.

Zixiang Li, Yue Song, Jianing Peng, Ting Liu, Jun Huang, Xiaochao Qu, Luoqi Liu, Wei Wang, Yao Zhao, and Yunchao Wei. On exact editing of flow-based diffusion models. *arXiv preprint arXiv:2512.24015*, 2025.

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. arXiv:2210.02747.

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 (ICLR)*, 2023. arXiv:2209.03003.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. RePaint: Inpainting using denoising diffusion probabilistic models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11461–11471, 2022. doi: 10.1109/CVPR52688.2022.01117. arXiv:2201.09865.

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.

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. arXiv:2305.04391.

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.

Michael T. McCann and Michael Unser. Biomedical image reconstruction: From the foundations to deep neural networks. *Foundations and Trends in Signal Processing*, 13(3):283–359, 2019.

Gregory Ongie, Ajil Jalal, Christopher A. Metzler, Richard G. Baraniuk, Alexandros G. Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

Art B. Owen. Monte Carlo theory, methods and examples. Chapter 8, §8.2 (antithetic sampling). https://artowen.su.domains/mc/, 2013.

Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3): 127–239, 2014.

Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *Transactions on Machine Learning Research (TMLR)*, 2024. arXiv:2310.04432.

Mehrsa Pourya, Bassam El Rawas, and Michael Unser. Flower: A flow-matching solver for inverse problems. In *International Conference on Learning Representations (ICLR)*, 2026.

Herbert Robbins. An empirical bayes approach to statistics. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 157–163, 1956.

Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

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, 2023. arXiv:2307.00619.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9472–9481, 2024. arXiv:2312.00852.

Samuel Ruipérez-Campillo, Pablo Blasco-Fernández, Moritz Rau, Prasanth Ganesan, Sabyasachi Bandyopadhyay, Charles Sillett, Lukas P. Arts, Esteban Peralta, Albert J. Rogers, Fleur V. Y. Tjong, Sanjiv M. Narayan, and Julia E. Vogt. Antithetic sampling enhanced probabilistic diffusion for denoising cardiac time series. *IEEE Journal of Biomedical and Health Informatics*, 2026. doi:10.1109/JBHI.2026.3705976.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations (ICLR)*, 2023.

Andrew M. Stuart. Inverse problems: A Bayesian perspective. *Acta Numerica*, 19:451–559, 2010.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research (TMLR)*, 2024. arXiv:2302.00482.

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 (GlobalSIP)*, pp. 945–948, 2013.

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. arXiv:2212.00490.

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.

Xingyu Xu and Yuejie Chi. Provably robust score-based diffusion posterior sampling for plug-and-play image reconstruction. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 36148–36184, 2024.

David M. Young. *Iterative Solution of Large Linear Systems*. Academic Press, 1971.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2015.

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)*, pp. 20895–20905, 2025. doi: 10.1109/CVPR52734.2025.01946. arXiv:2407.01521.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. doi: 10.1109/CVPR.2018.00068. arXiv:1801.03924.

Yasi Zhang, Peiyu Yu, Yaxuan Zhu, Yingshan Chang, Feng Gao, Ying Nian Wu, and Oscar Leong. Flow priors for linear inverse problems via iterative corrupted trajectory matching. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2405.18816.

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 (CVPR) Workshops*, 2023.

Rayhan Zirvi, Bahareh Tolooshams, and Anima Anandkumar. Diffusion state-guided projected gradient for inverse problems. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.03463.

APPENDIX ROADMAP

The appendix has three parts. Part A gives the theory and the algorithm, Part B the reproducibility details and the ablations, and Part C further tasks and domains.

Part A: Theory and Algorithms

What is proved and how the solver is implemented.

Appendix A.1 Local Gaussian model and its MAP estimate

Appendix A.2 Proofs of the local MAP estimate and the over-relaxation identity

Appendix A.3 Schedule rationale and step-size stability

Appendix A.4 Per-mode contraction and proof of the variance identity

Appendix A.5 *PPC-Flow* per-step algorithm

Part B: Ablations and Additional Results

How the empirical claims are checked.

Appendix B.1 Hyperparameter and reproducibility details

Appendix B.2 Scope and limitations

Appendix B.3 NFE and wall-clock analysis

Appendix B.4 Quality at a fixed total budget

Appendix B.5 Paired-bootstrap uncertainty quantification

Appendix B.6 Evaluation at a larger sample size

Appendix B.7 Cross-task velocity correction ablation

Appendix B.8 Cap, residual position, and time window

Appendix B.9 Component ablations at matched compute

Appendix B.10 Bridge-aligned schedule

Appendix B.11 Parity diagnostic for the paired-variance identity

Appendix B.12 Mechanism orthogonality

Appendix B.13 Extended qualitative comparison

Part C: Supplementary Datasets and Tasks

Further tasks and domains.

Appendix C.1 Free-form (paintbrush) inpainting

Appendix C.2 Cross-domain evaluation on LSUN-Bedroom

Appendix C.3 Extension to nonlinear forward operators

A Theory and Algorithms

A.1 Local Gaussian model and its MAP estimate

For the bridge $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ and $0 \le t < 1$, the posterior velocity is

$$v^\star(\mathbf{x}_t, t \mid \mathbf{y}) := \mathbb{E}[\mathbf{x}_1 - \mathbf{x}_0 \mid \mathbf{x}_t, \mathbf{y}] = \frac{\mathbb{E}[\mathbf{x}_1 \mid \mathbf{x}_t, \mathbf{y}] - \mathbf{x}_t}{1-t}. \tag{21}$$

We approximate the conditional endpoint distribution locally by a Gaussian. The learned flow supplies its mean, and the Gaussian bridge supplies its variance.

Lemma 1 (Conditional variance of the OT Gaussian bridge)**.** *For the OT bridge $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\,\mathbf{x}_1$ with $\mathbf{x}_0, \mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, I)$ independent, write $q_t := t^2 + (1-t)^2$. The conditional covariance of $\mathbf{x}_1$ given $\mathbf{x}_t$ is isotropic with scalar variance*

$$r_t \;:=\; \sigma_r^2(t) \;=\; \frac{(1-t)^2}{q_t} \;=\; \frac{(1-t)^2}{t^2 + (1-t)^2}. \tag{22}$$

We use r_t throughout as the variance of the local Gaussian model. The predicted endpoint $m_{\theta,t}$ of Eq. (6) equals the bridge conditional mean when v_θ is the exact conditional velocity of this Gaussian bridge.

Theorem 1 (Local MAP estimate)**.** *Let $0 \le t < 1$ and $\sigma_n^2 > 0$. With the predicted endpoint $m_{\theta,t}$ as mean, the bridge variance r_t of Lemma 1, and the Gaussian likelihood $\mathbf{y} \mid \mathbf{x}_1 \sim \mathcal{N}(\mathbf{H}\mathbf{x}_1, \sigma_n^2 I)$, the objective Eq. (15) has the unique minimizer*

$$\begin{aligned} z_t^\star &= m_{\theta,t} + r_t\,\mathbf{H}^\top \big(r_t\,\mathbf{H}\mathbf{H}^\top + \sigma_n^2 I\big)^{-1}(\mathbf{y} - \mathbf{H} m_{\theta,t}), \\ v_{\text{local}}^\star &= v_\theta + \frac{z_t^\star - m_{\theta,t}}{1-t}. \end{aligned} \tag{23}$$

Here $v_{\text{local}}^\star$ is the posterior velocity of the local Gaussian model.

This local Gaussian construction follows the standard Bayesian formulation of inverse problems (Stuart, 2010) and is related to pseudoinverse guidance (Song et al., 2023), OT-ODE (Pokle et al., 2024), and the endpoint correction of Flower (Pourya et al., 2026). The closed form is exact for the local Gaussian model. Posterior velocity correction followed by the exact proximal step over-relaxes this MAP step (Proposition 1). The cap uses only the operator norm $\sigma_{\max}(\mathbf{H})$, which we estimate once per run by power iteration, and the margin of $s = 1.9$ below the threshold of 2 absorbs the estimation error.

We also define the dimensionless ratio

$$\rho_t \;:=\; \frac{r_t\,\sigma_{\max}^2(\mathbf{H})}{\sigma_n^2}, \tag{24}$$

in terms of which the largest proximal gain is $w_{\max}(t) = \rho_t/(1+\rho_t) \le 1$, the quantity that enters the cap Eq. (25).

A.2 Proofs of the local MAP estimate and the over-relaxation identity

Proof of Lemma 1. Since $\mathbf{x}_0, \mathbf{x}_1 \sim \mathcal{N}(\mathbf{0}, I)$ are jointly Gaussian and independent, $(\mathbf{x}_t, \mathbf{x}_1)$ is jointly Gaussian with covariance

$$\mathrm{Cov}(\mathbf{x}_t, \mathbf{x}_t) = q_t\, I, \qquad \mathrm{Cov}(\mathbf{x}_t, \mathbf{x}_1) = t\, I, \qquad \mathrm{Cov}(\mathbf{x}_1, \mathbf{x}_1) = I,$$

where $q_t := t^2 + (1-t)^2$. Standard Gaussian conditioning gives the conditional covariance $r_t\, I$ with $r_t = 1 - t^2/q_t = (1-t)^2/q_t$. The predicted endpoint $m_{\theta,t} = \mathbf{x}_t + (1-t)v_\theta$ is defined directly from the learned flow. □

Proof of Theorem 1. The objective Φ_t in Eq. (15) is the negative log-posterior of the local Gaussian prior $\mathcal{N}(m_{\theta,t}, r_t I)$ combined with the Gaussian likelihood $\mathcal{N}(\mathbf{H}z, \sigma_n^2 I)$, up to an additive constant.

Setting $\nabla\Phi_t(z) = \sigma_n^{-2}\mathbf{H}^\top(\mathbf{H}z - \mathbf{y}) + r_t^{-1}(z - m_{\theta,t}) = 0$ yields the linear system $(\sigma_n^{-2}\mathbf{H}^\top\mathbf{H} + r_t^{-1}I)z_t^\star = \sigma_n^{-2}\mathbf{H}^\top\mathbf{y} + r_t^{-1}m_{\theta,t}$. The Woodbury identity (Bishop, 2006) gives

$$(\sigma_n^{-2}\mathbf{H}^\top\mathbf{H} + r_t^{-1}I)^{-1}\sigma_n^{-2}\mathbf{H}^\top \;=\; r_t\,\mathbf{H}^\top(r_t\,\mathbf{H}\mathbf{H}^\top + \sigma_n^2 I)^{-1},$$

and substituting back yields the closed form $z_t^\star = m_{\theta,t} + r_t\mathbf{H}^\top(r_t\mathbf{H}\mathbf{H}^\top + \sigma_n^2 I)^{-1}(\mathbf{y} - \mathbf{H}m_{\theta,t})$. The local posterior velocity then follows from $v_{\text{local}}^\star = v_\theta + (z_t^\star - m_{\theta,t})/(1-t)$. □

Proof of Proposition 1. Write $A_t := \sigma_n^{-2}\mathbf{H}^\top\mathbf{H} + r_t^{-1}I$ and $G_t := r_t\mathbf{H}^\top(r_t\mathbf{H}\mathbf{H}^\top + \sigma_n^2 I)^{-1} = A_t^{-1}\mathbf{H}^\top/\sigma_n^2$ by the Woodbury identity used in the proof of Theorem 1, so that $z_t^\star = m_{\theta,t} + G_t(\mathbf{y} - \mathbf{H}m_{\theta,t})$ and $I - G_t\mathbf{H} = r_t^{-1}A_t^{-1}$. In endpoint coordinates the velocity-corrected endpoint of Eq. (11) is

$$\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} \;=\; m_{\theta,t} + c_t\,r_t\,\mathbf{H}^\top(\mathbf{y} - \mathbf{H}m_{\theta,t})/\sigma_n^2, \qquad c_t := \frac{(1-t)\,\alpha_t}{r_t} = \omega_t - 1.$$

With $\lambda_t = r_t$, the proximal step Eq. (12) maps an input u to $A_t^{-1}(\mathbf{H}^\top\mathbf{y}/\sigma_n^2 + u/r_t) = (I - G_t\mathbf{H})\,u + G_t\mathbf{y}$, which is affine in u. Applying it to $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$ and using $(I - G_t\mathbf{H})\,r_t\mathbf{H}^\top/\sigma_n^2 = A_t^{-1}\mathbf{H}^\top/\sigma_n^2 = G_t$,

$$\mathbf{x}^\star \;=\; (I - G_t\mathbf{H})\,m_{\theta,t} + G_t\mathbf{y} \;+\; c_t\,G_t(\mathbf{y} - \mathbf{H}m_{\theta,t}) \;=\; z_t^\star + c_t\,(z_t^\star - m_{\theta,t}),$$

which is Eq. (16). For the per-mode statement let $\mathbf{H} = U\Sigma V^\top$. Then $\mathbf{H}G_t = U\,\mathrm{diag}\big(w_j(t)\big)U^\top$ with $w_j(t) = r_t\sigma_j^2/(r_t\sigma_j^2 + \sigma_n^2)$, so the post-step residual $\mathbf{y} - \mathbf{H}\mathbf{x}^\star = (I - \omega_t\mathbf{H}G_t)(\mathbf{y} - \mathbf{H}m_{\theta,t})$ has its j-th component scaled by $1 - \omega_t w_j(t)$. For the observed modes $\sigma_j > 0$, all factors satisfy $|1 - \omega_t w_j(t)| < 1$ if and only if $\omega_t w_j(t) < 2$ for every such j, that is, if and only if $\omega_t\,w_{\max}(t) < 2$ with $w_{\max}(t) = \rho_t/(1+\rho_t)$. Requiring $\omega_t w_{\max}(t) \le s$ with $s = 1.9$ and substituting $\omega_t = 1 + (1-t)\min\{a, \bar{a}_t\}$ gives the cap

$$\bar{a}_t \;=\; \frac{1}{1-t}\Big(\frac{s}{w_{\max}(t)} - 1\Big) \;=\; \frac{1}{1-t}\Big(\frac{s\,(1+\rho_t)}{\rho_t} - 1\Big), \qquad s = 1.9. \tag{25}$$

Since $w_{\max} \le 1$, we have $\bar{a}_t \ge (s-1)/(1-t) > 0$. The cap therefore never switches the correction off, and it becomes inactive once $w_{\max}(t)$ has decayed in the final steps. □

Relation to the current-residual form. Evaluating the residual at the current iterate instead of the endpoint, $\mathbf{y} - \mathbf{H}\mathbf{x}_t = (\mathbf{y} - \mathbf{H}m_{\theta,t}) + (1-t)\mathbf{H}v_\theta$, adds the term $c_t(1-t)\,G_t\mathbf{H}v_\theta$ to $\mathbf{x}^\star$. Under the uncapped schedule this additional term is proportional to $(1-t)^2$. The two choices behave similarly for weak correction, whereas the endpoint residual performs better as the correction strength grows.

A.3 Schedule rationale and step-size stability

Cap behaviour along the trajectory. For a sufficiently large correction strength a, the cap binds while $w_{\max}(t)$ is close to one and releases once $w_{\max}(t)$ decays (Eq. (25)). The cap therefore bounds $\omega_t w_{\max}$ where the proximal gain is large and becomes inactive as $r_t \to 0$. Without the cap, strong corrections can cross the contraction threshold and diverge. The capped schedule remains stable across all ten task–dataset cells, and its gain saturates as the correction strength increases.

Where velocity correction contributes. The time-window ablation in Tab. 12 locates the benefit in the late steps. Applying the correction only for $t \le 0.6$ changes PSNR by $+0.003$ dB on SR×2 and by 0.000 dB on denoising, whereas applying it for $t \ge 0.6$ recovers the full gain. Lemma 2 predicts this behaviour. The solver forgets early corrections, so the bias forms in the late window, where a stronger data correction pays off.

Fig. 5 reports the corresponding endpoint-error curves for CelebA SR×2.

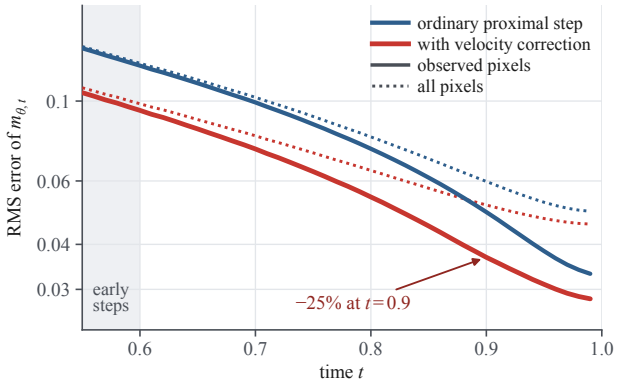

Figure 5: RMS error of the predicted endpoint $m_{\theta,t}$ against the clean image on 100 CelebA SR×2 test images, in the observed pixels (solid) and in all pixels (dotted), with the ordinary proximal step and with velocity correction. Grey shading marks the early interval $t \leq 0.6$ tested separately in Tab. 12.

The single-step analysis also does not predict where the gain lands. Velocity correction acts in the range of $\mathbf{H}^\top$, yet its gain appears mostly in unobserved pixels. Tab. 4 splits the squared error of the terminal reconstruction by the image-space observation mask, that is, the pixels kept by the decimation, the random mask, or the box mask. For these pixel-selection operators, this split coincides with the orthogonal projections $P_{\text{obs}} = \mathbf{H}^\dagger\mathbf{H}$ and $P_{\text{null}} = I - \mathbf{H}^\dagger\mathbf{H}$. On SR×2 the error falls by 25% in observed and by 14% in unobserved pixels, and since three quarters of the pixels are unobserved, 84% of the total reduction lands there. On random inpainting the observed pixels are already matched without velocity correction, and the entire gain is in the masked pixels. On box inpainting the masked block improves while the observed pixels, whose error is about eighty times smaller, change marginally. The network infers the unobserved content from the observed content, so a better-anchored observed part improves its prediction of the unobserved pixels.

Table 4: Where the gain of velocity correction lands. Squared error of the terminal reconstruction (K=2 paired, 100 CelebA test images, per pixel, $\times 10^{-3}$, image range $[-1, 1]$), split by the image-space observation mask, with and without velocity correction. The last column is the share of the total squared-error reduction that falls on unobserved pixels, with a 95% bootstrap interval over images.

	observed	observed pixels		unobserved pixels		share on
Task	fraction	off	on	off	on	unobserved
SR×2	25%	0.841	0.630	2.491	2.133	84% [83, 84]
Random inpainting	30%	0.067	0.066	2.446	2.136	100% [100, 100]
Box inpainting	90%	0.359	0.373	30.85	30.02	118% [110, 169]

Why $\sigma_r^2(t)$ as the envelope. The bridge variance $r_t = \sigma_r^2(t)$ is the conditional variance in the Gaussian bridge model of Lemma 1. For fixed a, this envelope gives $\alpha_t \to 0$ and $\omega_t \to 1$ as $t \to 1$, so the extra correction tapers off at the terminal boundary. We use this envelope for all reported configurations.

A.3.1 Effective memory of endpoint corrections

This subsection proves the retention statement used in Section 3.2. A correction applied at one step reaches the output multiplied by a product of per-step factors, each of magnitude below one, so early corrections are attenuated and the bias of the solver is created in the late window. We prove this statement in the Gaussian bridge model of Lemma 1, which replaces the learned endpoint predictor by the conditional mean of that bridge. With $q_t = t^2 + (1-t)^2$, the predictor and the variance are

$$D_t(x) = \frac{t}{q_t}x, \qquad r_t = \frac{(1-t)^2}{q_t},$$

both exact for the Gaussian bridge.

Fix a linear operator $\mathbf{H}$, an observation $\mathbf{y}$, $\sigma_n^2 > 0$, and times $0 \le t_0 < \cdots < t_T = 1$. At step $k < T$, let $m_k = D_{t_k}(\mathbf{x}_k)$ and apply the velocity correction followed by the exact proximal step, which by Proposition 1 gives

$$u_k = m_k + \omega_k G_k(\mathbf{y} - \mathbf{H}m_k), \qquad G_k = r_{t_k}\mathbf{H}^\top(r_{t_k}\mathbf{H}\mathbf{H}^\top + \sigma_n^2 I)^{-1},$$

where $\omega_k := \omega_{t_k} \ge 1$ is deterministic. Advance by $\mathbf{x}_{k+1} = t_{k+1}u_k + (1-t_{k+1})\varepsilon_k$. At $t_T = 1$, D_1 is the identity, so the terminal prediction equals u_{T-1}. For a singular pair $\mathbf{H}v_j = \sigma_j q_j$, $\mathbf{H}^\top q_j = \sigma_j v_j$ with $\sigma_j > 0$, define

$$e_{k,j} = q_j^\top(\mathbf{y} - \mathbf{H}m_k), \quad w_{k,j} = \frac{r_{t_k}\sigma_j^2}{r_{t_k}\sigma_j^2 + \sigma_n^2},$$
$$c_{k,j} = 1 - \omega_k w_{k,j}, \qquad \beta_{k+1} = \frac{t_{k+1}^2}{q_{t_{k+1}}} = 1 - r_{t_{k+1}}.$$

Here $c_{k,j}$ is the signed retention of the data correction and β_{k+1} that of the re-injection step followed by the next endpoint prediction.

Lemma 2 (Effective memory of a correction)**.** *Under the model and the updates above, each observed residual component satisfies the pathwise recurrence*

$$e_{k+1,j} = \beta_{k+1}c_{k,j}e_{k,j} + b_{k+1,j},$$
$$b_{k+1,j} = (1-\beta_{k+1})q_j^\top\mathbf{y} - \frac{t_{k+1}(1-t_{k+1})}{q_{t_{k+1}}}\sigma_j v_j^\top\varepsilon_k.$$

For $0 \le k \le \ell \le T$, define the signed memory kernel

$$M_j(k,\ell) = \prod_{s=k}^{\ell-1}\beta_{s+1}c_{s,j}, \qquad M_j(\ell,\ell) = 1.$$

Two runs that start from different predicted endpoints at step k and share the measurement, the strengths, and the subsequent noises satisfy $\Delta e_{\ell,j} = M_j(k,\ell)\,\Delta e_{k,j}$. Every factor has magnitude below one whenever $0 < \omega_s w_{s,j} < 2$, which the cap of Eq. (17) enforces. If $|\beta_{s+1}c_{s,j}| \le q < 1$ over L steps, then $|\Delta e_{k+L,j}| \le q^L\,|\Delta e_{k,j}|$.

Proof. Gaussian conditioning gives $D_t(x) = tx/q_t$ because $\mathrm{Cov}(\mathbf{x}_1, \mathbf{x}_t) = tI$ and $\mathrm{Cov}(\mathbf{x}_t) = q_t I$. Substituting the bridge update into this predictor yields

$$m_{k+1} = \beta_{k+1}u_k + \frac{t_{k+1}(1-t_{k+1})}{q_{t_{k+1}}}\varepsilon_k.$$

The singular-pair identities give $q_j^\top\mathbf{H}G_k = w_{k,j}q_j^\top$, so $q_j^\top(\mathbf{y} - \mathbf{H}u_k) = c_{k,j}e_{k,j}$. Subtracting $q_j^\top\mathbf{H}m_{k+1}$ from $q_j^\top\mathbf{y}$ proves the recurrence. The forcing $b_{k+1,j}$ is the same for the two runs, so subtraction and induction give the product formula. Since $0 < \beta_{s+1} \le 1$ and $|c_{s,j}| < 1$ under the stated condition, the product contracts, and bounding its L factors by q gives the inequality. The identities are pathwise and need no distributional assumption on the shared noise. □

The same expansion gives the full residual budget,

$$e_{T,j} = M_j(0,T)e_{0,j} + \sum_{k=0}^{T-1} M_j(k+1,T)b_{k+1,j}.$$

The first term retains the initial discrepancy and the sum retains the discrepancies introduced by later steps. For a tolerance $\tau \in (0,1)$ the effective memory window is

$$k_j^*(\tau) = \min\{k \in \{0,\dots,T\} : |M_j(k,T)| \ge \tau\}.$$

Under the contraction condition, $|M_j(k,T)|$ is nondecreasing in k and $M_j(T,T) = 1$, so the window is well defined. Its width depends on the operator, the noise level, the time grid, and the strengths. Corrections applied before this window are attenuated below τ by the time they reach the output.

What velocity correction adds. Compare two runs with the same initial prediction and the same noises, one with $\omega_k = 1$ and one with a deterministic schedule $\omega_k \geq 1$. Write their residuals as $e^0_{k,j}$ and $e^\omega_{k,j}$ and the kernel of the second as M^ω_j. The forcing terms are identical, so subtracting the recurrences and expanding from a zero initial difference gives

$$e^0_{T,j} - e^\omega_{T,j} = \sum_{k=0}^{T-1} M^\omega_j(k+1, T)\, \beta_{k+1}(\omega_k - 1) w_{k,j} e^0_{k,j}.$$

Each term is the extra correction applied at one step, multiplied by its retention through all later steps. The expansion makes the trade-off explicit. An early correction is attenuated before it reaches the output, while near $t = 1$ the ordinary proximal gain $w_{k,j}$ is already small. Velocity correction enlarges the correction inside the window where it survives, and the cap keeps every retention factor below one in magnitude.

Two consequences follow. First, the retention factor $c_{k,j} = 1 - \omega_k w_{k,j}$ can be negative under the cap. For $w = 0.99$ it is 0.01 without velocity correction and -0.9 at $\omega w = 1.9$, so the residual changes sign while its magnitude falls. Second, for a fixed correction strength the schedule of Eq. (17) gives $(\omega_t - 1) w_j(t) \to 0$ as $t \to 1$, so the useful window ends before the final step. A single velocity-corrected step reduces the residual magnitude below that of the ordinary proximal step exactly when $1 < \omega < 2/w - 1$, which is much stricter than the cap condition $\omega w < 2$. The empirical gain therefore comes from the residual budget over the whole memory window rather than from any single step.

Learned endpoint predictors. For a general differentiable predictor $D_s(x) = x + (1 - s) v_\theta(x, s)$, the first-order propagation of a perturbation of m_k contains the matrix

$$t_{k+1}\, J D_{t_{k+1}}(\mathbf{x}_{k+1})(I - \omega_k G_k \mathbf{H}),$$

which mixes singular directions. The scalar kernel above is exact under the Gaussian predictor, and the time-window ablation of Tab. 12 measures the same behaviour on the learned solver. On SR×2, a correction confined to $t \leq 0.6$ changes PSNR by $+0.003$ dB, and one confined to $t \geq 0.6$ recovers $+0.811$ dB, essentially the full $+0.812$ dB. On the null space $\sigma_j = 0$ the matrix $G_k \mathbf{H}$ vanishes, so the correction reaches those directions only through later network evaluations, whose effect Tab. 4 measures.

A.4 Per-mode contraction ratio and proof of the variance identity

Proposition 3 isolates the role of the proximal step, which multiplies every observed mode by an additional contraction factor after the velocity correction.

Proposition 3 (Per-mode contraction ratio). *Let $\eta_j := (1 - t)\alpha_t \sigma_j^2 / \sigma_n^2$ and $\kappa_j := \lambda_t \sigma_j^2 / \sigma_n^2$ be the per-mode correction and proximal parameters in the singular basis of $\mathbf{H}$. With $\lambda_t = r_t$ one has $\eta_j = (\omega_t - 1)\, \kappa_j$, so η_j is the per-mode over-relaxation excess of Proposition 1. For range-space modes $\sigma_j > 0$ in the contraction regime $|1 - \eta_j| < \kappa_j + 1$, that is, $\eta_j < \kappa_j + 2$ or equivalently $\omega_t w_j(t) < 2$, which the cap of Eq. (17) enforces with margin for every mode, the homogeneous contraction coefficient under PPC-Flow of the linearised range-space residual is*

$$\left|\rho_j^{\text{PPC-Flow}}\right| = \frac{|1 - \eta_j|}{\kappa_j + 1}, \qquad \left|\rho_j^{\text{DPS}}\right| = |1 - \eta_j|, \qquad \frac{\left|\rho_j^{\text{PPC-Flow}}\right|}{\left|\rho_j^{\text{DPS}}\right|} = \frac{1}{\kappa_j + 1}, \tag{26}$$

compared with a DPS-style residual update without the proximal step, where the ratio is defined for $\eta_j \neq 1$. If both homogeneous factors vanish, both updates eliminate the homogeneous component in one step. For fixed coefficients with $0 < |1 - \eta_j| < 1$, an idealized scalar recurrence without network or noise forcing has the step-count ratio

$$\frac{T_{\text{DPS}}}{T_{\text{PPC-Flow}}} = 1 + \frac{\log(\kappa_j + 1)}{\left|\log|1 - \eta_j|\right|}. \tag{27}$$

We prove Proposition 3. Let $\mathbf{H} = U \Sigma V^\top$, with σ_j the j-th singular value and $V_{(j)}, U_{(j)}$ the corresponding singular vectors. Restrict to range-space modes $\sigma_j > 0$ and the contraction regime $|1 - \eta_j| < \kappa_j + 1$.

We first project the velocity-corrected endpoint onto the singular basis. It is $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} = m_{\theta,t} + (1-t)\,\alpha_t\sigma_n^{-2}\mathbf{H}^\top(\mathbf{y} - \mathbf{H}m_{\theta,t})$ with $m_{\theta,t} = \mathbf{x}_t + (1-t)v_\theta$. Projecting onto $V_{(j)}$ gives

$$(V^\top \hat{\mathbf{x}}_1^{\textit{PPC-Flow}})_j = (1-\eta_j)(V^\top m_{\theta,t})_j + \eta_j \cdot \sigma_j^{-1}(U^\top \mathbf{y})_j, \tag{28}$$

with $\eta_j = (1-t)\alpha_t\sigma_j^2/\sigma_n^2$ and $(V^\top m_{\theta,t})_j = (V^\top \mathbf{x}_t)_j + (1-t)(V^\top v_\theta)_j$.

The proximal problem Eq. (14) has closed-form solution $\mathbf{z}^\star$ satisfying $(\sigma_n^{-2}\mathbf{H}^\top\mathbf{H} + \lambda_t^{-1}I)\mathbf{z}^\star = \sigma_n^{-2}\mathbf{H}^\top\mathbf{y} + \lambda_t^{-1}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$. Projecting onto $V_{(j)}$, using $(V^\top\mathbf{H}^\top\mathbf{H})_j = \sigma_j^2 V_{(j)}^\top$, and setting $\kappa_j = \lambda_t\sigma_j^2/\sigma_n^2$ gives

$$(V^\top \mathbf{z}^\star)_j = \frac{\kappa_j \cdot \sigma_j^{-1}(U^\top\mathbf{y})_j + (V^\top\hat{\mathbf{x}}_1^{\textit{PPC-Flow}})_j}{\kappa_j + 1}. \tag{29}$$

To compare against the clean signal, substitute $(U^\top\mathbf{y})_j = \sigma_j(V^\top\mathbf{x}_1)_j + (U^\top\eta)_j$ where $\mathbf{x}_1$ is the clean image and η is the measurement noise, and subtract $x_j^\star := (V^\top\mathbf{x}_1)_j$ from (29) and (28). Write the per-step residual as $e_{k,j} := (V^\top\mathbf{x}_{t_k})_j - x_j^\star$. The first line is the update of *PPC-Flow*, and the second is a FlowDPS-style residual update evaluated at the same endpoint.

$$(V^\top\mathbf{z}_k^\star)_j - x_j^\star = \frac{(1-\eta_j)\left[e_{k,j} + (1-t)(V^\top v_\theta)_j\right]}{\kappa_j+1} + \frac{(\kappa_j+\eta_j)(U^\top\eta)_j}{\sigma_j(\kappa_j+1)}, \tag{30}$$

$$(V^\top\mathbf{z}_k^{\star,\mathrm{DPS}})_j - x_j^\star = (1-\eta_j)\left[e_{k,j} + (1-t)(V^\top v_\theta)_j\right] + \eta_j\frac{(U^\top\eta)_j}{\sigma_j}. \tag{31}$$

The homogeneous coefficient of $e_{k,j}$ is $(1-\eta_j)/(\kappa_j+1)$ for *PPC-Flow* and $(1-\eta_j)$ for the update without the proximal step, which gives the ratio (26). The coefficient $(1-\eta_j)/(\kappa_j+1)$ equals $1-\omega_t w_j(t)$, the per-mode factor of Proposition 1. In the over-relaxed regime $\eta_j > 1$ the coefficient is negative, so the residual overshoots, and its modulus stays below one as long as $\eta_j < \kappa_j + 2$.

For the step-count ratio, consider the fixed-coefficient scalar recurrence under $0 < |1-\eta_j| < 1$, so both homogeneous factors lie strictly between zero and one in magnitude. Contracting a residual by a factor $\rho \in (0,1)$ requires $T \geq \log\rho/\log|\rho_j|$. Taking the ratio of these thresholds before integer rounding gives $T_{\mathrm{DPS}}/T_{\textit{PPC-Flow}} = \log|\rho_j^{\textit{PPC-Flow}}|/\log|\rho_j^{\mathrm{DPS}}|$. Substituting $\log|\rho_j^{\textit{PPC-Flow}}| = \log|1-\eta_j| - \log(\kappa_j+1)$ and $\log|\rho_j^{\mathrm{DPS}}| = \log|1-\eta_j|$ gives

$$\frac{T_{\mathrm{DPS}}}{T_{\textit{PPC-Flow}}} = 1 + \frac{\log(\kappa_j+1)}{\left|\log|1-\eta_j|\right|},$$

which is (27). □

On the null space. For $\sigma_j = 0$, the gradient $\mathbf{H}^\top(\mathbf{y} - \mathbf{H}m_{\theta,t})$ has no component in that direction, and both the velocity correction and the proximal step leave $(V^\top m_{\theta,t})_j$ unchanged. The null-space content improves through the next network evaluation, which infers it from the corrected observed content. Tab. 4 measures this effect.

Saturation. The gain from velocity correction first grows with the correction strength a and then saturates, because the cap holds $\omega_t w_{\max}(t)$ at 1.9 over most of the trajectory (Appendix A.3).

How the three steps interact. Posterior velocity correction adds a residual-based term to the predicted endpoint (Eq. (11)), and the exact proximal step contracts the remaining observed-mode residual by $1/(\kappa_j+1)$ (Eq. (14)). Together they over-relax the proximal step under the cap of Eq. (17). Antithetic pairing acts on the trajectory variance, with the reduction given by Proposition 2. The first two steps reduce the bias term of Eq. (10), and the third reduces its variance term.

A.4.1 Proof of paired variance reduction (Proposition 2)

We give the calculation behind the identity Eq. (19) of Proposition 2. The identity is the classical antithetic-variates result (Hammersley & Morton 1956; Owen 2013, §8.2), and the calculation below makes the marginalisation over the initial latents ζ explicit.

Let $f(\varepsilon) := \mathbb{E}_\zeta[\mathbf{x}_T(\varepsilon;\zeta)]$ and its parity components

$$f_e(\varepsilon) := \tfrac{1}{2}\big(f(\varepsilon)+f(-\varepsilon)\big), \qquad f_o(\varepsilon) := \tfrac{1}{2}\big(f(\varepsilon)-f(-\varepsilon)\big).$$

Under the symmetric law of ε, f_e and f_o are orthogonal in L^2, since $\mathbb{E}_{\boldsymbol{\varepsilon}}[f_o]=0$ and $\mathbb{E}_{\boldsymbol{\varepsilon}}[f_e^\top f_o]=0$, so $\mathrm{Var}(f)=\mathrm{Var}(f_e)+\mathrm{Var}(f_o)$. Define the ε-conditional residual variance $\sigma_\zeta^2(\varepsilon) := \mathbb{E}_\zeta\|\mathbf{x}_T(\varepsilon;\zeta)-f(\varepsilon)\|^2$, which has $\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(-\varepsilon)]=\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(\varepsilon)]$ by symmetry.

For the paired estimator, since $\mathbb{E}_{\zeta_+,\zeta_-}[T_p \mid \varepsilon] = \frac{1}{2}(f(\varepsilon)+f(-\varepsilon)) = f_e(\varepsilon)$ and the two trajectories are conditionally independent given ε,

$$\begin{aligned}\mathrm{Var}(T_p) &= \mathrm{Var}_{\boldsymbol{\varepsilon}}\big(\mathbb{E}_{\zeta_+,\zeta_-}[T_p \mid \varepsilon]\big) + \mathbb{E}_{\boldsymbol{\varepsilon}}\big[\mathrm{Var}_{\zeta_+,\zeta_-}(T_p \mid \varepsilon)\big] \\ &= \mathrm{Var}(f_e) + \tfrac{1}{2}\,\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(\varepsilon)]. \end{aligned} \tag{32}$$

For the independent-averaging estimator, by full independence of (ε_a,ζ_a) and (ε_b,ζ_b),

$$\begin{aligned}\mathrm{Var}(T_i) &= \tfrac{1}{2}\,\mathrm{Var}\big(f(\varepsilon)+\text{(zero-mean residual)}\big) \\ &= \tfrac{1}{2}\,\mathrm{Var}(f) + \tfrac{1}{2}\,\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(\varepsilon)] = \tfrac{1}{2}\big(\mathrm{Var}(f_e)+\mathrm{Var}(f_o)\big) + \tfrac{1}{2}\,\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(\varepsilon)]. \end{aligned} \tag{33}$$

Subtracting Eq. (32) from Eq. (33) cancels the common term $\frac{1}{2}\mathbb{E}_{\boldsymbol{\varepsilon}}[\sigma_\zeta^2(\varepsilon)]$ and gives

$$\mathrm{Var}(T_i) - \mathrm{Var}(T_p) = \tfrac{1}{2}\big(\mathrm{Var}(f_o) - \mathrm{Var}(f_e)\big). \tag{34}$$

To reach the trace-covariance form, use $\mathrm{Var}(f_e \pm f_o) = \mathrm{Var}(f_e)+\mathrm{Var}(f_o) \pm 2\,\mathrm{tr}\,\mathrm{Cov}(f_e,f_o) = \mathrm{Var}(f_e)+\mathrm{Var}(f_o)$ by orthogonality, and the identity $\mathrm{tr}\,\mathrm{Cov}(f(\varepsilon),f(-\varepsilon)) = \mathrm{Var}(f_e)-\mathrm{Var}(f_o)$, which follows from $f(\varepsilon)=f_e+f_o$ and $f(-\varepsilon)=f_e-f_o$. We obtain

$$\mathrm{Var}(f_o) - \mathrm{Var}(f_e) \;=\; -\,\mathrm{tr}\,\mathrm{Cov}\big(f(\varepsilon),f(-\varepsilon)\big),$$

and substituting it into Eq. (34) gives the second equality of Eq. (19). The condition for strict reduction, Eq. (20), follows directly. □

Wiener-chaos remark (Gaussian re-injection). For Gaussian re-injection noise the same condition can be read through the Hermite expansion. When ε is standard Gaussian, the Wiener-chaos expansion $f=\sum_{k\ge 0} f_k$ into homogeneous components yields

$$\mathrm{tr}\,\mathrm{Cov}\big(f(\varepsilon),f(-\varepsilon)\big) \;=\; \sum_{k\ge 0}(-1)^k\,\mathrm{Var}(f_k). \tag{35}$$

Antithetic pairing therefore reduces the trace variance whenever the odd-degree part $\sum_{k\ \mathrm{odd}}\mathrm{Var}(f_k)$ exceeds the even-degree part $\sum_{k\ \mathrm{even},\,k\ge 2}\mathrm{Var}(f_k)$. The linear term $k=1$ dominates when the re-injection noise perturbs the trajectory only slightly, which is the regime measured in Appendix B.11.

Measured sign of the trace covariance. Appendix B.11 measures the criterion Eq. (20) directly. On all five CelebA tasks, with and without velocity correction, $\mathrm{Var}(f_o)$ exceeds $\mathrm{Var}(f_e)$ by a factor of 10 to 49, so $\mathrm{tr}\,\mathrm{Cov}(f(\varepsilon),f(-\varepsilon)) < 0$ holds with a wide margin.

A.5 PPC-FLOW PER-STEP ALGORITHM

Algorithm 1 gives the deterministic version used in all experiments. The initial latents of the two trajectories are independent, while the re-injection noise sequence is shared with opposite signs, as in the variance analysis of Proposition 2.

Here $\hat{\mathbf{x}}^{\pm}_{1,t}$ is the velocity-corrected endpoint $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$ of the trajectory with sign $\pm$, and $\mathbf{x}^{\pm}_{\star,t}$ is its proximal-corrected endpoint. Each iteration ends with the re-injection step, which uses sign-flipped noise for the two trajectories. Neither the velocity correction nor the proximal step calls the velocity network. After the call to $v_\theta(\mathbf{x}_t^{\pm},t)$, the remaining operations use $\mathbf{H}$, $\mathbf{H}^\top$, vector updates, and the CG solve. The two terminal reconstructions are averaged pixelwise, which gives the paired estimator of Section 3.3.

Algorithm 1 *PPC-Flow* with Paired Posterior Correction (K=2 paired trajectories)

Input: measurement $\mathbf{y}$, operator $\mathbf{H}$, noise level σ_n, pretrained velocity v_θ, correction strength a, cap margin s=1.9, flow steps T
Output: terminal paired reconstruction $\hat{\mathbf{x}}$
Set step size $\delta \leftarrow 1/T$ and bridge-aligned correction schedule $\sigma_r^2(t) \leftarrow (1-t)^2/(t^2+(1-t)^2)$, and estimate $\sigma^2_{\max}(\mathbf{H})$ once by power iteration
Draw initial latents $\mathbf{x}_0^+, \mathbf{x}_0^- \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, I)$ and shared noises $\varepsilon_t \sim \mathcal{N}(\mathbf{0}, I)$ for each $t \in \{0, \delta, \dots, 1-\delta\}$
for $t = 0, \delta, 2\delta, \dots, 1-\delta$ **do**
  $\lambda_t \leftarrow \sigma_r^2(t),\ A_{\lambda_t} \leftarrow \mathbf{H}^\top\mathbf{H}/\sigma_n^2 + I/\lambda_t$
  $w_{\max} \leftarrow \lambda_t\sigma^2_{\max}(\mathbf{H})/(\lambda_t\sigma^2_{\max}(\mathbf{H}) + \sigma_n^2),\ \bar{a}_t \leftarrow (s/w_{\max} - 1)/(1-t),\ \alpha_t \leftarrow \min\{a, \bar{a}_t\}\,\sigma_r^2(t)$ *// capped strength*
  for each sign $\pm \in \{+, -\}$ **do**
    ▷ *Posterior velocity correction (Section 3.2)*
    $m_t^\pm \leftarrow \mathbf{x}_t^\pm + (1-t)\,v_\theta(\mathbf{x}_t^\pm, t),\ g_t^\pm \leftarrow \mathbf{H}^\top(\mathbf{y} - \mathbf{H}m_t^\pm)/\sigma_n^2$ *// endpoint residual*
    $v_t^\pm \leftarrow v_\theta(\mathbf{x}_t^\pm, t) + \alpha_t\, g_t^\pm$
    $\hat{\mathbf{x}}_{1,t}^\pm \leftarrow \mathbf{x}_t^\pm + (1-t)\,v_t^\pm$ *// velocity-corrected endpoint*
    ▷ *Proximal data-consistency correction (Section 3.1)*
    $b_t^\pm \leftarrow \mathbf{H}^\top\mathbf{y}/\sigma_n^2 + \hat{\mathbf{x}}_{1,t}^\pm/\lambda_t$
    $\mathbf{x}_{\star,t}^\pm \leftarrow \mathrm{CG}(A_{\lambda_t}\,\mathbf{z} = b_t^\pm)$ *// proximal-corrected endpoint*
  end for
  ▷ *Re-injection step with antithetic pairing (Section 3.3)*
  $\mathbf{x}_{t+\delta}^+ \leftarrow (t+\delta)\,\mathbf{x}_{\star,t}^+ + (1-t-\delta)\,\varepsilon_t$
  $\mathbf{x}_{t+\delta}^- \leftarrow (t+\delta)\,\mathbf{x}_{\star,t}^- - (1-t-\delta)\,\varepsilon_t$
end for
Set $\mathbf{x}_{\text{term}}^\pm \leftarrow \mathbf{x}_{t=1}^\pm$ and **return** $\hat{\mathbf{x}} = \frac{1}{2}(\mathbf{x}_{\text{term}}^+ + \mathbf{x}_{\text{term}}^-)$

Optional posterior sampling. The experiments use the deterministic proximal endpoint of Algorithm 1. The same local Gaussian model also gives a conjugate draw around this endpoint, which keeps the antithetic pairing and replaces the deterministic endpoint by a random one.

The only change relative to Algorithm 1 is a conjugate Gaussian draw between the proximal endpoint $\mathbf{x}^\star$ (Eq. (12)) and the re-injection step. We replace $\mathbf{x}^\star$ by the perturbed endpoint

$$\tilde{\mathbf{x}}_1 \;=\; \mathbf{x}^\star \;+\; \gamma_t\,\kappa_t, \quad \kappa_t \sim \mathcal{N}(\mathbf{0}, \Sigma_t), \quad \Sigma_t \;=\; \big(\sigma_r^{-2}I + \sigma_n^{-2}\mathbf{H}^\top\mathbf{H}\big)^{-1}, \tag{36}$$

where $\gamma_t \in [0, 1]$ controls the scale of the perturbation. The correction strength $a \geq 0$ is unchanged. It controls the mean, while κ_t carries the covariance implied by the local Gaussian model.

Sampling κ_t via one extra CG call. We avoid a Cholesky factor of Σ_t and draw κ_t by CG with a structured right-hand side. Sampling $\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, I_d)$ and $\mathbf{z}_2 \sim \mathcal{N}(\mathbf{0}, I_m)$ independently and forming

$$r \;=\; \mathbf{z}_1/\sigma_r \;+\; \mathbf{H}^\top\mathbf{z}_2/\sigma_n, \qquad \kappa_t \;=\; \Sigma_t\, r \;=\; \mathrm{CG}\big(\Sigma_t^{-1}\kappa_t = r,\ \text{init} = \mathbf{0}\big), \tag{37}$$

gives $\mathrm{Cov}(r) = \sigma_r^{-2}I + \sigma_n^{-2}\mathbf{H}^\top\mathbf{H} = \Sigma_t^{-1}$, hence $\mathrm{Cov}(\kappa_t) = \Sigma_t$ exactly. The CG call uses the operator Σ_t^{-1} of the proximal step, so the cost is one extra CG solve per step.

Setting $\gamma_t \equiv 0$ recovers Algorithm 1 exactly. Disabling velocity correction and setting $\gamma_t \equiv 1$ recovers the posterior draw of Pourya et al. (2026) under the same bridge covariance.

Proposition 4 (Posterior draw of the local Gaussian model)**.** *Under the velocity-corrected Gaussian model $\tilde{p}(\mathbf{x}_1 \mid \mathbf{x}_t) = \mathcal{N}(\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}, \sigma_r^2 I)$ and Gaussian likelihood $p(\mathbf{y} \mid \mathbf{x}_1) = \mathcal{N}(\mathbf{H}\mathbf{x}_1, \sigma_n^2 I)$, the endpoint draw $\tilde{\mathbf{x}}_1$ in Eq. (36) with γ_t=1 satisfies*

$$\tilde{\mathbf{x}}_1 \mid (\mathbf{x}_t, \mathbf{y}) \;\sim\; \mathcal{N}(\mu_t, \Sigma_t), \qquad \mu_t \;=\; \hat{\mathbf{x}}_1^{\textit{PPC-Flow}} \;+\; \Sigma_t\,\sigma_n^{-2}\mathbf{H}^\top(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}), \tag{38}$$

that is, the posterior of the local Gaussian model. Since the model is centred at the velocity-corrected endpoint $\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}$, the mean of the draw contains the correction of Eq. (11), which the γ=1 variant of Pourya et al. (2026) lacks. The construction separates the shift of the posterior mean, controlled by the velocity correction, from the posterior covariance, supplied by the Gaussian bridge.

Proof. Conjugacy of Gaussian prior and Gaussian likelihood gives precision $\Sigma_t^{-1} = \sigma_r^{-2}I + \sigma_n^{-2}\mathbf{H}^\top\mathbf{H}$ and Gaussian posterior mean $\Sigma_t\big(\sigma_r^{-2}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} + \sigma_n^{-2}\mathbf{H}^\top\mathbf{y}\big)$. The rearrangement

$$\begin{aligned}\Sigma_t(\sigma_r^{-2}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} + \sigma_n^{-2}\mathbf{H}^\top\mathbf{y}) &= \Sigma_t\big(\Sigma_t^{-1}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} - \sigma_n^{-2}\mathbf{H}^\top\mathbf{H}\,\hat{\mathbf{x}}_1^{\textit{PPC-Flow}} + \sigma_n^{-2}\mathbf{H}^\top\mathbf{y}\big)\\ &= \hat{\mathbf{x}}_1^{\textit{PPC-Flow}} + \Sigma_t\sigma_n^{-2}\mathbf{H}^\top(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_1^{\textit{PPC-Flow}}),\end{aligned}$$

yields the form of μ_t in Eq. (38), which coincides with the proximal solution $\mathbf{x}^\star$ of Eq. (12) when $\hat{\mathbf{x}}_1$ is replaced by $\hat{\mathbf{x}}_1^{PPC\text{-}Flow}$. The covariance of $\kappa_t = \Sigma_t r$ with $r = \mathbf{z}_1/\sigma_r + \mathbf{H}^\top \mathbf{z}_2/\sigma_n$ is $\Sigma_t \cdot \mathrm{Cov}(r) \cdot \Sigma_t^\top = \Sigma_t \cdot \Sigma_t^{-1} \cdot \Sigma_t = \Sigma_t$. Therefore $\tilde{\mathbf{x}}_1 = \mu_t + \kappa_t$ has the stated $\mathcal{N}(\mu_t, \Sigma_t)$ distribution. □

Mean shift under velocity correction. Theorem 1 defines the local MAP estimate, and Proposition 1 characterizes the shift of the mean under velocity correction and the proximal step. Proposition 4 adds Gaussian noise around this mean with the conjugate covariance.

B Ablations and Additional Results

B.1 Hyperparameter and reproducibility details

Benchmark operators and splits. The main benchmark uses CelebA (Liu et al., 2015) at 128^2 and AFHQ-Cat (Choi et al., 2020) at 256^2, both normalised to $[-1, 1]$, following the released OT-flow evaluation protocol used by PnP-Flow and Flower (Martin et al., 2025; Pourya et al., 2026). We evaluate each dataset on the released 100-image test split.

Table 5: Linear inverse-problem operators used in the main CelebA and AFHQ-Cat benchmarks.

Task	CelebA-128^2	AFHQ-Cat-256^2
Denoising	$\sigma_n = 0.2$	$\sigma_n = 0.2$
Gaussian deblurring	61×61 kernel, std. 1.0, $\sigma_n = 0.05$	61×61 kernel, std. 3.0, $\sigma_n = 0.05$
Super-resolution	$2\times$ pixel decimation, $\sigma_n = 0.05$	$4\times$ pixel decimation, $\sigma_n = 0.05$
Random inpainting	70% pixels removed, $\sigma_n = 0.01$	70% pixels removed, $\sigma_n = 0.01$
Box inpainting	centered 40×40 mask, $\sigma_n = 0.05$	centered 80×80 mask, $\sigma_n = 0.05$

For super-resolution, $\mathbf{H}$ retains the upper-left pixel of each non-overlapping $q \times q$ block ($q = 2$ or 4), and $\mathbf{H}^\top$ places the measurements back at the sampled locations, filling the remaining entries with zeros. The released benchmark baselines use the same measurement operator.

Backbone and hardware. Every run uses a single NVIDIA RTX 4090. We reuse the released OT-flow checkpoints of Pourya et al. (2026) for CelebA-128^2 and AFHQ-Cat-256^2, trained for 200 epochs with batch size 128 on CelebA and for 400 epochs with batch size 64 on AFHQ-Cat.

Flow-step counts. Denoising, CelebA deblurring, SR$\times$2, and box inpainting use T=100 flow steps. Random inpainting uses T=100 on CelebA and T=200 on AFHQ-Cat. AFHQ deblurring and SR$\times$4 use T=500. All counts follow Flower (Pourya et al., 2026).

The proximal step Eq. (14) uses conjugate gradients (Hestenes & Stiefel, 1952) with a maximum of 50 iterations and a relative residual tolerance of 10^{-5}, as in Flower.

Velocity correction and proximal schedules. We use the bridge-aligned, stability-capped schedule $\alpha_t = \min\{a, \bar{a}_t\}\, \sigma_r^2(t)$ with the cap of Eq. (25) ($s = 1.9$) throughout. The proximal weight is fixed to the same bridge variance, $\lambda_t = \sigma_r^2(t)$, as in the released Flower solver. The schedule and cap are shared across all reported configurations.

For the main benchmarks, we select a on held-out validation images (32 CelebA and 28 AFHQ-Cat) and fix it for test evaluation. In task order denoising, deblurring, super-resolution, random inpainting, and box inpainting, the values are $(2, 2, 12, 12, 3)$ for CelebA and $(6, 12, 12, 12, 0)$ for AFHQ-Cat, where $a = 0$ disables velocity correction.

Antithetic pairing. For K=2 antithetic pairing, we draw a single re-injection sequence $(\varepsilon_0, \ldots, \varepsilon_{T-1})$ for the first trajectory, then use $(-\varepsilon_0, \ldots, -\varepsilon_{T-1})$ for the second. The initial latent $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is sampled independently per trajectory from a seeded RNG. Only the re-injection noise is paired.

Determinism. The first-trajectory RNG seed is fixed as $10000 \cdot \text{batch_idx} + 13$ and the second-trajectory seed is $10000 \cdot \text{batch_idx} + 7919 + 13$.

Table 6: Per-image cost. T flow steps, K trajectories, $C \leq 50$ CG iterations.

Method	NFE	Operator calls	NFE ratio vs. F5-OT
Flower1-OT	T	$2CT$	0.20
Flower5-OT	$5T$	$10CT$	1.00
FlowDPS	T	$2T$	0.20
PPC-Flow $K=1$	T	$2CT + 2T$	0.20
PPC-Flow $K=2$	$\mathbf{2T}$	$\mathbf{4CT + 4T}$	**0.40**

Table 7: Wall-clock time per image in seconds on one RTX 4090 (20 CelebA and 12 AFHQ-Cat test images in batches of four, first batch excluded as warm-up). The last row gives the ratio of *PPC-Flow* to Flower5-OT.

Method	CelebA SR×2 $T=100$	AFHQ-Cat deblurring $T=500$	AFHQ-Cat SR×4 $T=500$
PnP-Flow1	0.75	5.12	5.34
Flower1-OT	1.00	17.29	7.29
PnP-Flow5	3.36	23.78	24.46
Flower5-OT	5.54	50.27	53.13
PPC-Flow ($K=2$)	**1.34**	**20.48**	**14.32**
PPC-Flow / Flower5-OT	0.24	0.41	0.27

B.2 Scope and limitations

The reported configuration has the following scope.

1. Problem scope. The main experiments assume a known linear forward operator and Gaussian noise. Appendix C.3 extends both data steps to differentiable nonlinear operators through a per-step Gauss–Newton linearisation. Latent-space flows with nonlinear decoders, blind degradations, and strongly nonconvex operators such as phase retrieval are left to future work.
2. Pairing. Only the re-injection noise is paired. The initial latents are independent, since pairing them has no effect (Tab. 12).
3. Datasets. The main experiments follow the public Flower benchmark, so that backbones, operators, and splits are matched across all baselines. Appendix C.2 adds a separate public Rectified Flow checkpoint and an indoor-scene domain.

B.3 Computational cost: NFE and wall-clock analysis

One application of $\mathbf{H}$ or $\mathbf{H}^\top$ costs less than 2% of a network forward pass, and Tab. 6 counts the NFEs and the operator calls separately. *PPC-Flow* with $K=2$ uses 40% of the NFEs of Flower5-OT.

Tab. 7 measures the wall-clock time on the tasks where the data step costs most, CelebA SR×2 and the two AFHQ-Cat tasks that run $T=500$ steps. *PPC-Flow* takes 24 to 41% of the time of Flower5-OT. On the two SR tasks the ratio falls below the NFE ratio because *PPC-Flow* evaluates the two paired trajectories as one batch.

B.4 Quality at a fixed total budget

The main tables fix the number of steps T and vary the number of trajectories K. A method can also spend a budget of $B=KT$ NFEs on fewer, longer trajectories, so we compare the best allocation of each method at the same B. On CelebA we run Flower and PnP-Flow with $(K, T) \in \{(1, 100), (1, 200), (2, 100), (1, 500), (2, 250), (5, 100)\}$ and *PPC-Flow* with $K=2$ and $T \in \{50, 100, 250\}$. On the two AFHQ-Cat tasks with $T=500$ in the main table we add Flower with $(1, 200), (1, 1000), (2, 250), (2, 500), (5, 100), (5, 200)$, PnP-Flow with $(2, 500), (1, 1000)$, and *PPC-Flow* with $T \in \{100, 250\}$. The strength a stays at its validation value for every T, and no other parameter is tuned. Tab. 8 reports the best PSNR each method reaches within the budget.

PPC-Flow has the highest PSNR at every budget. On CelebA its $B=100$ result already exceeds the best result of Flower at $B=500$ on four of the five tasks and is within 0.04 dB on box inpainting, so

Table 8: Best PSNR (dB) reached within a total budget of B NFEs per image, over the (K, T) allocations listed in Appendix B.4, on 100 test images. Each entry reads Flower / PnP-Flow / *PPC-Flow*, and — marks a budget below the smallest allocation run.

Dataset	Task	B=100	B=200	B=500
CelebA	Denoising	32.28 / 31.80 / **33.42**	32.81 / 32.30 / **33.48**	33.14 / 32.70 / **33.48**
	Deblurring	34.98 / 34.48 / **35.82**	35.40 / 34.70 / **35.89**	35.67 / 34.80 / **35.89**
	SR×2	32.36 / 31.09 / **33.83**	32.79 / 32.28 / **34.06**	33.09 / 32.91 / **34.06**
	Random inpainting	33.08 / 33.05 / **34.42**	33.69 / 33.72 / **34.91**	34.30 / 34.43 / **35.13**
	Box inpainting	31.19 / 30.47 / **31.83**	31.55 / 30.86 / **32.08**	31.87 / 31.09 / **32.08**
		B=200	B=500	B=1000
AFHQ-Cat	Deblurring	28.69 / — / **29.39**	29.21 / 28.09 / **29.39**	29.21 / 28.29 / **29.39**
	SR×4	26.00 / — / **27.67**	26.40 / 27.14 / **27.85**	26.60 / 27.44 / **27.85**

Table 9: Paired-bootstrap 95% CIs for *PPC-Flow* vs. Flower5-OT (re-run with matched measurements, K=5 i.i.d.) on all ten test cells.

Dataset	Task	*PPC-Flow* PSNR	F5-OT PSNR	Δ PSNR	95% CI	p
CelebA	Denoise	33.482	33.143	+0.340	[+0.326, +0.353]	< 0.001
CelebA	Deblur	35.889	35.661	+0.228	[+0.219, +0.237]	< 0.001
CelebA	SR×2	34.061	33.080	+0.981	[+0.931, +1.030]	< 0.001
CelebA	Rand inp	34.906	34.033	+0.873	[+0.818, +0.930]	< 0.001
CelebA	Box inp	32.084	31.751	+0.334	[+0.239, +0.434]	< 0.001
AFHQ-Cat	Denoise	32.916	32.404	+0.512	[+0.495, +0.529]	< 0.001
AFHQ-Cat	Deblur	28.712	28.563	+0.149	[+0.104, +0.187]	< 0.001
AFHQ-Cat	SR×4	27.470	26.696	+0.774	[+0.653, +0.922]	< 0.001
AFHQ-Cat	Rand inp	34.263	33.865	+0.399	[+0.370, +0.429]	< 0.001
AFHQ-Cat	Box inp	27.132	27.013	+0.119	[−0.019, +0.265]	0.098

the 40% ratio of the main tables understates the saving. Longer single trajectories do not help the baselines. Flower with $(1, 500)$ is below $(5, 100)$ on four of the five CelebA tasks. At B=2500, the budget of Flower5-OT and PnP-Flow5 in Tab. 2, Flower reaches 29.21 and 26.71 dB and PnP-Flow 28.39 and 27.57 dB on AFHQ-Cat deblurring and SR×4, both below *PPC-Flow* at B=500. On AFHQ-Cat deblurring both Flower and *PPC-Flow* do better with 100 steps than with the 500 steps that the main table inherits from the Flower protocol.

B.5 Paired-bootstrap uncertainty quantification

For every *PPC-Flow* vs. Flower5-OT comparison on a shared 100-image test set we compute a paired-bootstrap 95% confidence interval (Efron & Tibshirani, 1993) for the per-image PSNR difference. The bootstrap draws 10 000 resamples with replacement at the image level, and we report a two-sided bootstrap p-value for the null hypothesis of zero mean PSNR difference. To pair the two methods image by image, we re-run Flower5-OT in our code path as K=5 independent trajectories without velocity correction, with the same measurement realisations, image order, and CG configuration as *PPC-Flow*. Its per-cell means agree with the released numbers of Tabs. 1 and 2 within 0.12 dB. Tab. 9 reports the resulting per-cell intervals.

B.6 Evaluation at a larger sample size

The main tables follow the released benchmark of Pourya et al. (2026) and report the 100-image test split of each dataset. This section repeats the CelebA evaluation on 1,000 images from the official test split, ten times the benchmark size, to show that the reported ranking does not depend on the size of that split.

Protocol. Everything except the number of images is unchanged. The per-task strengths, the bridge-aligned schedule with s=1.9, the CG configuration, the measurement realisations and the image order are those of Tab. 1, and nothing is re-selected on the larger split. The baselines are the

Table 10: CelebA 128^2 evaluated on 1,000 test images instead of the benchmark 100. The last two columns compare *PPC-Flow* with its own 100-image result of Tab. 1 and with Flower5-OT at $5T$ NFE on the same 1,000 images. *PPC-Flow* uses $2T$ NFE throughout.

	PSNR↑ at N=1,000			SSIM↑ at N=1,000		*PPC-Flow* vs.	
Task	Flower1-OT	Flower5-OT	*PPC-Flow*	Flower5-OT	*PPC-Flow*	own N=100	Flower5-OT
Denoising	32.31	33.17	**33.50**	0.9256	**0.9296**	+0.02	+0.33
Deblurring	35.03	35.73	**35.95**	0.9539	**0.9554**	+0.06	+0.22
SR×2	32.45	33.19	**34.19**	0.9314	**0.9448**	+0.13	+1.00
Random inpainting	33.26	34.15	**35.06**	0.9532	**0.9613**	+0.15	+0.91
Box inpainting	—	31.85	**32.14**	0.9540	**0.9545**	+0.05	+0.29

Table 11: Velocity correction ablation on all ten cells (PSNR in dB, 100 test images per cell, K=2 paired). Δ is relative to no velocity correction, and the last column compares the capped schedule with the uncapped reference.

Dataset	Task	no correction	uncapped reference		capped (Eq. (17))		capped − uncapped
		PSNR	PSNR	Δ	PSNR	Δ [95% CI]	Δ [95% CI]
CelebA	Denoise	33.363	33.494	+0.132	33.482	+0.120 [+0.106, +0.133]	−0.012 [−0.024, −0.001]
CelebA	Deblur	35.837	35.882	+0.045	35.889	+0.052 [+0.043, +0.061]	+0.007 [+0.002, +0.013]
CelebA	SR×2	33.249	33.417	+0.168	34.061	+0.812 [+0.768, +0.858]	+0.644 [+0.604, +0.686]
CelebA	Rand inp	34.238	34.298	+0.060	34.906	+0.668 [+0.623, +0.714]	+0.608 [+0.568, +0.649]
CelebA	Box inp	31.933	32.016	+0.084	32.084	+0.152 [+0.095, +0.209]	+0.068 [+0.014, +0.124]
AFHQ-Cat	Denoise	32.562	32.658	+0.096	32.916	+0.354 [+0.334, +0.374]	+0.258 [+0.238, +0.278]
AFHQ-Cat	Deblur	28.577	28.541	−0.036	28.712	+0.135 [+0.082, +0.174]	+0.171 [+0.138, +0.205]
AFHQ-Cat	SR×4	26.720	26.801	+0.081	27.470	+0.750 [+0.615, +0.913]	+0.669 [+0.577, +0.765]
AFHQ-Cat	Rand inp	34.035	34.021	−0.014	34.263	+0.228 [+0.201, +0.255]	+0.242 [+0.214, +0.273]
AFHQ-Cat	Box inp	27.132	27.118	−0.014	27.132	0.000	+0.014 [−0.013, +0.042]

released solvers run on the same 1,000 images with the same seeds, so every column is measured on identical inputs.

Results. Tab. 10 shows that the 100-image split reports the method faithfully. Across the five tasks the PSNR of *PPC-Flow* moves by at most 0.15 dB when the evaluation grows to 1,000 images, and it moves upward on every task, so the benchmark split slightly understates the method. *PPC-Flow* keeps the highest PSNR and SSIM on all five tasks at 40% of the NFE budget of Flower5-OT, and the margin over Flower5-OT at the larger sample size is +0.22 to +1.00 dB. The two tasks with the largest null space, SR×2 and random inpainting, keep the largest margins, which is the behaviour that Tab. 4 explains.

B.7 Cross-task velocity correction ablation

Tab. 11 isolates the velocity correction step on all ten task–dataset cells within the same paired solver, with K=2, sign-flipped re-injection noise, the proximal step, and identical measurements and seeds. We compare three settings on the 100 test images of each cell. The first has no velocity correction, the second is an uncapped reference correction evaluated at $\mathbf{x}_t$, and the third is the capped endpoint correction of Eq. (17). Confidence intervals are per-image paired bootstrap with 5 000 resamples.

Three observations follow. First, the uncapped reference is a small and task-dependent correction, +0.04 to +0.17 dB on CelebA and within noise or negative on three AFHQ cells. Second, the capped schedule improves on no correction on nine of the ten cells and on the uncapped reference on eight, and the other two cells differ from the uncapped reference by at most 0.014 dB. Third, the gains concentrate where the proximal step is most under-relaxed in the late window, especially SR and random inpainting. On denoising and deblurring, over-relaxation also amplifies measurement noise and thus limits the gain. LPIPS moves in the same direction as PSNR on every cell where PSNR improves.

Table 12: Time-window ablation. The velocity correction is applied only on a sub-interval (PSNR in dB, Δ vs. no velocity correction with 95% CI). The last row pairs the initial latents in addition to the re-injection noise.

	SR×2		Denoising	
correction active on	PSNR	Δ [95% CI]	PSNR	Δ [95% CI]
none	33.249	—	33.363	—
$t \leq 0.6$ only	33.252	$+0.003$ $[+0.000, +0.007]$	33.363	0.000
$t \geq 0.9$ only	33.833	$+0.584$ $[+0.546, +0.622]$	33.450	$+0.088$ $[+0.081, +0.095]$
$t \geq 0.8$ only	34.020	$+0.771$ $[+0.726, +0.815]$	33.482	$+0.120$ $[+0.106, +0.133]$
$t \geq 0.6$ only	34.060	$+0.811$ $[+0.766, +0.856]$	33.482	$+0.120$ $[+0.106, +0.133]$
all t	34.061	$+0.812$ $[+0.768, +0.858]$	33.482	$+0.120$ $[+0.106, +0.133]$
initial-noise pairing	33.249	0.000	33.363	0.000

B.8 Cap, residual position, and time window

Unless stated otherwise, the experiments in this section use CelebA 128^2 SR×2 with T=100, K=2 paired, and 100 test images. Intervals are per-image paired bootstrap with 5 000 resamples against the configuration without velocity correction.

Role of the cap. Across the tested correction strengths, the capped schedule remains stable and its gain saturates, whereas sufficiently strong uncapped corrections can cross the contraction threshold and diverge. We use s=1.9 throughout, which keeps a margin of 0.1 below the theoretical threshold.

Residual position. Evaluating the residual at the current state $\mathbf{x}_t$ instead of the predicted endpoint $m_{\theta,t}$ adds the term $c_t(1-t)\,G_t\mathbf{H}v_\theta$ to the proximal endpoint (Appendix A.2). The two forms behave similarly for weak correction, but the additional term degrades the result as the correction becomes stronger. We therefore use the endpoint residual throughout.

Time window and initial-noise pairing. Tab. 12 restricts the correction to sub-intervals of the trajectory. A correction confined to $t \leq 0.6$ leaves the reconstruction essentially unchanged. It recovers $+0.003$ of the $+0.812$ dB on SR×2 and 0.000 dB on denoising. A correction confined to $t \geq 0.6$ recovers the full gain, and $t \geq 0.8$ still recovers 95% of the SR gain. Pairing the initial latents with opposite signs in addition to the re-injection noise changes neither task. Both results agree with Lemma 2. The solver forgets early corrections and its initial latent, so the correction must act late and the pairing must act on the re-injection noise.

Seed stability. Re-running CelebA SR×2 with disjoint random-number generator (RNG) seed offsets 0, 100 and 200 gives velocity-correction gains of $+0.812$, $+0.816$ and $+0.816$ dB over the matching runs without velocity correction. On AFHQ-Cat SR×4 a repeat with seed offset 100 gives $+0.733$ dB, against $+0.750$ dB for offset 0.

Paired momentum. On CelebA random inpainting, extrapolating the endpoint along the trend of the pair-averaged endpoint, $m^{\pm} \leftarrow m^{\pm} + \kappa\,(\bar{m}_t - \bar{m}_{t-\delta})$, adds $+0.08$ dB on top of the capped step, from 34.906 to 34.987 dB with κ=0.2 and a milder strength. The extrapolation acts in the null space, which the proximal step cannot reach, and its trend must come from the paired average. With a single trajectory the same extrapolation turns the $+0.51$ dB gain of velocity correction at K=1 into -0.10 dB. The extrapolation does not help on the other tasks and is not part of the reported method.

B.9 Component ablations and matched-compute details

This subsection collects the ablations of the two mechanisms of *PPC-Flow*, velocity correction and antithetic pairing. The proximal step is the backbone of the solver and stays on in every configuration. We start with a factorial ablation on CelebA 128^2 denoising (Tab. 13), then give the K-scaling sweep (Fig. 6) and per-image visualizations (Figs. 7 and 8).

Tab. 13 compares single- and paired-trajectory configurations. Velocity correction improves the single-trajectory estimate, and antithetic pairing reduces the variance of the terminal reconstruction

Table 13: Factorial ablation on CelebA 128^2 denoising (100 test images). Columns toggle posterior velocity correction and $K{=}2$ antithetic pairing, and every row keeps the proximal step. Rows (a) and (b) use $K{=}1$ with T NFEs and rows (c) and (d) use $K{=}2$ with $2T$ NFEs.

ID	Velocity correction	Paired ($K{=}2$)	PSNR	SSIM	Δ vs Full
(a)	✗	✗	32.29	0.914	-1.19
(b)	✓	✗	32.37	0.914	-1.11
(c)	✗	✓	33.36	0.929	-0.12
(d) **Full *PPC-Flow***	✓	✓	**33.48**	**0.930**	0 (ref)

Table 14: Velocity correction $\times$ trajectory configuration on CelebA 128^2 SR$\times$2 and random inpainting (PSNR in dB, 100 test images, $T{=}100$, $s{=}1.9$). Single-trajectory and paired rows use T and $2T$ NFEs, respectively. Velocity-correction gains are read along rows and gains over $K{=}1$ along columns.

Task	Trajectories	correction off	correction on	velocity correction gain
SR$\times$2	$K{=}1$ (single)	32.346	33.081	$+0.735$
	$K{=}2$ paired	33.249	34.061	$+0.812$
	gain over $K{=}1$	$+0.903$	$+0.980$	
Random inpainting	$K{=}1$ (single)	33.123	33.629	$+0.506$
	$K{=}2$ paired	34.238	34.906	$+0.668$
	gain over $K{=}1$	$+1.115$	$+1.277$	

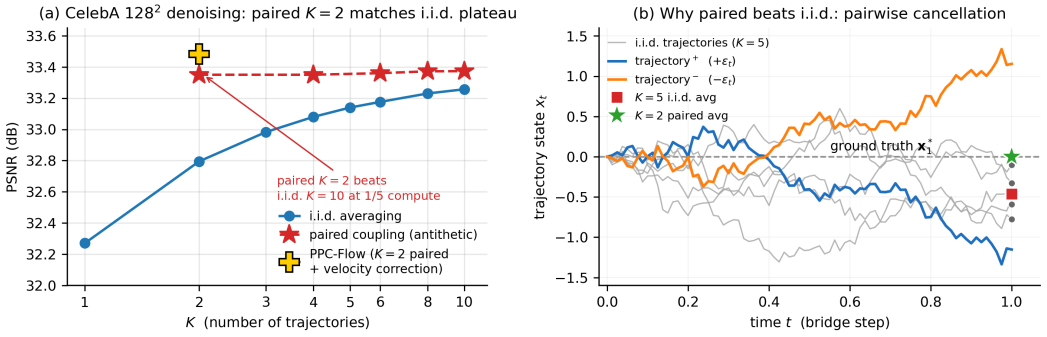

Figure 6: $K{=}2$ with antithetic pairing beats the baselines at T and at $5T$. (a) PSNR vs. number of trajectories K on CelebA 128^2 denoising. Antithetic pairing at $K{=}2$ already matches the plateau of independent averaging, and *PPC-Flow* ($K{=}2$ paired with velocity correction) exceeds it. (b) Schematic of the mechanism. Five independent trajectories (grey) drift toward the same reference and scatter at the endpoint, which leaves a residual in their average (red square). A pair (blue and orange) shares the re-injection noise sequence with opposite signs, so its average suppresses the odd noise response and moves closer to the reference. Tab. 3 in the main text reports the corresponding 5-task mean PSNR and SSIM at three NFE budgets.

at matched NFE (Tab. 17). The two mechanisms are complementary. The four-quadrant CelebA experiment in Tab. 17 shows that the gain of pairing over independent averaging persists when velocity correction is enabled, as the bias–variance decomposition in Eq. (10) predicts. The correction strength matters most on tasks with a large null space. Tab. 14 crosses velocity correction with pairing on SR$\times$2 and random inpainting. Velocity correction adds $+0.74$ and $+0.81$ dB on SR and $+0.51$ and $+0.67$ dB on random inpainting, on single and paired trajectories respectively. Moving from one trajectory to a paired two-trajectory average adds $+0.90$ and $+0.98$ dB on SR and $+1.12$ and $+1.28$ dB on random inpainting, without and with velocity correction. The two gains are additive with a small positive interaction.

Fig. 8 shows reconstructions that correspond to the data points of Fig. 6(a). For the same CelebA 128^2 denoising test images, the columns are $K{=}1$ Flower1, $K{=}2$ independent averaging, $K{=}2$ antithetic pairing without velocity correction, $K{=}5$ Flower5, and *PPC-Flow*. Pairing at $K{=}2$ removes much of the visible stochastic degradation at 40% of the NFE budget of Flower5, and velocity correction further reduces the blur near high-frequency facial features.

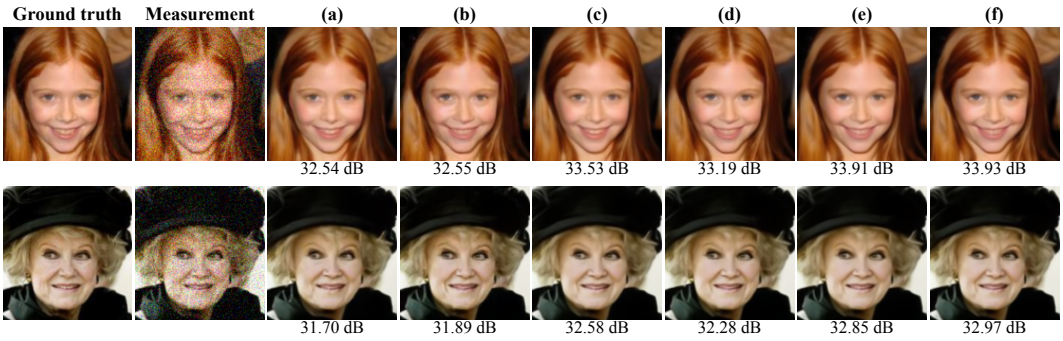

Figure 7: Trajectory configurations on two CelebA 128^2 denoising test images, with per-image PSNR. After the clean image and the measurement $\mathbf{y}$ come (a) Flower1-OT, (b) a single trajectory with velocity correction, (c) Flower5-OT, (d) two independent trajectories, (e) two paired trajectories without correction, and (f) *PPC-Flow*, two paired trajectories with velocity correction. All use T=100, with respective NFE budgets $T, T, 5T, 2T, 2T, 2T$. Panels (b) and (f) use the strength a=2 of Appendix B.1.

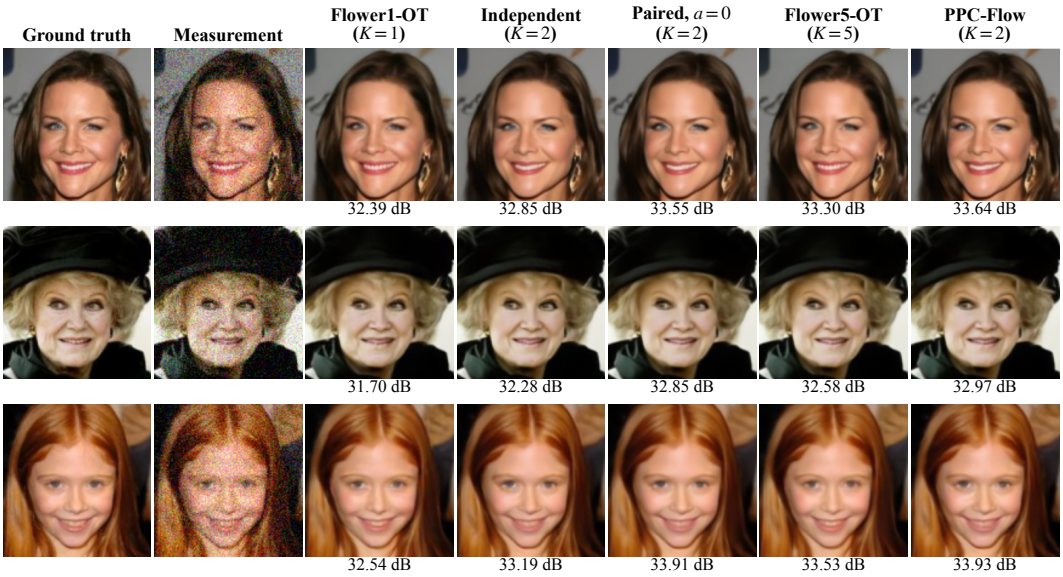

Figure 8: Reconstructions for the five compute configurations of Fig. 6(a) on three CelebA 128^2 denoising test images, with the per-image PSNR (dB) below each tile.

B.10 Bridge-aligned schedule

The schedule in Eq. (17) multiplies the capped strength $\min\{a, \bar{a}_t\}$ by the bridge variance $\sigma_r^2(t)$, the same envelope that attenuates the proximal weight of the benchmark solver. The envelope matches the scale of the endpoint uncertainty and makes the correction vanish at $t = 1$, and the cap (Appendix A.3) bounds the per-step over-relaxation. All reported configurations share both.

B.11 Parity diagnostic: measuring both sides of the paired-variance identity

Proposition 2 states that antithetic pairing reduces trace variance by exactly $\frac{1}{2}(\mathrm{Var}(f_o) - \mathrm{Var}(f_e))$, so the gain is positive whenever the odd part of the trajectory response to the re-injection noise carries more energy than the even part. This subsection makes the condition concrete in two ways. It identifies the regime in which the condition holds, and it measures both variances directly, so that we can check the identity, without any free parameter, against the gains of pairing over independent averaging in Tab. 17. The same runs also measure the two terms of Eq. (10).

Table 15: Parity diagnostic on CelebA 128^2 (100 test images, 8 seeds, K=2, T=100). Odd share is $\mathrm{Var}(f_o)/\mathrm{Var}(f)$, the ratio is $\mathrm{Var}(f_o)/\mathrm{Var}(f_e)$, and $\mathrm{Var}(T_p)/\mathrm{Var}(T_i) = 1 + \mathrm{tr}\,\mathrm{Cov}(A,B)/\mathrm{tr}\,\mathrm{Var}(A)$ is the ratio of paired to independent-averaging variance. Both ratios sum the traces over the 100 images before dividing, so each row satisfies $\mathrm{Var}(T_p)/\mathrm{Var}(T_i) = 2/(1+\mathrm{Var}(f_o)/\mathrm{Var}(f_e))$ up to the initial-latent term and can be recomputed from the two columns. The predicted gain follows from the identity with no free parameter. The measured gain is the single-seed gain of pairing over independent averaging on the same 100 test images, taken from Tab. 17. †Single-seed measurement on the noisiest cell, discussed in the text. In every configuration $\mathbb{E}[\sigma_\zeta^2]/\mathrm{tr}\,\mathrm{Var}(A) < 10^{-3}$, so the initial latent does not reach the endpoint.

Task	correction	odd share	$\mathrm{Var}(f_o)/\mathrm{Var}(f_e)$	$\mathrm{Var}(T_p)/\mathrm{Var}(T_i)$	predicted ΔPSNR [95% CI]	measured ΔPSNR
Denoising	off	0.978	44.2	0.044	+0.555 [+0.545, +0.566]	+0.559
Denoising	on	0.977	42.6	0.046	+0.571 [+0.561, +0.581]	+0.575
Deblurring	off	0.980	49.4	0.040	+0.441 [+0.425, +0.458]	—
Deblurring	on	0.979	47.1	0.042	+0.452 [+0.436, +0.468]	—
SR×2	off	0.966	28.3	0.068	+0.462 [+0.447, +0.477]	+0.455
SR×2	on	0.970	32.2	0.060	+0.509 [+0.493, +0.525]	+0.494
Random inpainting	off	0.952	19.9	0.095	+0.560 [+0.545, +0.576]	+0.572
Random inpainting	on	0.955	21.0	0.091	+0.657 [+0.638, +0.675]	+0.665
Box inpainting	off	0.915	10.8	0.170	+0.400 [+0.372, +0.428]	+0.514†
Box inpainting	on	0.910	10.2	0.179	+0.421 [+0.392, +0.450]	+0.503†

Linear-response reading of the condition. Expanding the ζ-averaged terminal response around $\varepsilon = \mathbf{0}$ gives $f(\varepsilon) = f(\mathbf{0}) + \mathbf{J}_f\varepsilon + \frac{1}{2}\varepsilon^\top \mathcal{Q}_f \varepsilon + \cdots$, where every odd-order term belongs to f_o and every even-order term to f_e. Hence $\mathrm{Var}(f_o)$ is, to leading order, the linear response $\|\mathbf{J}_f\|_F^2$ of the reconstruction to the injected noise, while $\mathrm{Var}(f_e)$ is the fluctuation of its second-order response, and the constant term $f(\mathbf{0})$ contributes to neither. The condition $\mathrm{Var}(f_o) > \mathrm{Var}(f_e)$ therefore states that the solver responds to the re-injection noise predominantly linearly. When f is affine in ε, $\mathrm{Var}(f_e) = 0$ and pairing eliminates the re-injection contribution to the variance of the paired estimator, so that $\mathrm{Var}(T_i) - \mathrm{Var}(T_p) = \frac{1}{2}\mathrm{Var}(f_o) = \frac{1}{2}\mathrm{Var}(f)$, half the corresponding single-trajectory variance. The nonlinearity of v_θ is what makes $\mathrm{Var}(f_e)$ nonzero. Pairing fails to strictly reduce variance when $\mathrm{Var}(f_e) \geq \mathrm{Var}(f_o)$, including the limiting case where the injected randomness does not reach the endpoint (see the initial-latent ablation in Tab. 12 and Appendix B.8).

Measuring the two variances. For a fixed measurement $\mathbf{y}$ we run, for each of S=8 seeds, three trajectories with independent initial latents $\zeta_1, \zeta_2, \zeta_3$ and re-injection sequences $(+\varepsilon, -\varepsilon, +\varepsilon)$, namely

$$A = \mathbf{x}_T(\varepsilon;\zeta_1), \qquad B = \mathbf{x}_T(-\varepsilon;\zeta_2), \qquad C = \mathbf{x}_T(\varepsilon;\zeta_3).$$

Because the initial latents are independent, the cross-seed trace covariances are exactly

$$\mathrm{tr}\,\mathrm{Cov}(A,B) = \mathrm{Var}(f_e) - \mathrm{Var}(f_o), \qquad \mathrm{tr}\,\mathrm{Cov}(A,C) = \mathrm{Var}(f_e) + \mathrm{Var}(f_o) = \mathrm{Var}(f),$$
$$\mathrm{tr}\,\mathrm{Var}(A) = \mathrm{Var}(f) + \mathbb{E}_\varepsilon[\sigma_\zeta^2],$$

so $\mathrm{Var}(f_o)$, $\mathrm{Var}(f_e)$ and the initial-latent residual $\mathbb{E}[\sigma_\zeta^2]$ are all identified without approximation. From the same runs we record the MSE of the paired K=2 estimate $\frac{1}{2}(A+B)$ against the clean image. Proposition 2 then predicts the MSE of independent averaging as $\mathrm{MSE}_{\mathrm{iid}} = \mathrm{MSE}_{\mathrm{paired}} + \frac{1}{2}(\mathrm{Var}(f_o) - \mathrm{Var}(f_e))$, and hence the PSNR gain of pairing, which we compare with the gain measured in independent runs (Tab. 17). The diagnostic uses the 100 CelebA test images, and confidence intervals are paired-bootstrap over images.

Results. Three facts emerge from Tab. 15. First, the condition of Proposition 2 holds with a wide margin on every task. The odd part carries 91 to 98% of the noise-induced variance $\mathrm{Var}(f)$, so $\mathrm{Var}(f_o)$ exceeds $\mathrm{Var}(f_e)$ by a factor of 10 to 49, and the paired K=2 estimator retains only 4 to 18% of the variance of independent averaging. The solver is close to the affine regime, in which pairing eliminates the re-injection contribution to the variance of the paired estimator. Second, the identity is quantitatively predictive. The PSNR gain of pairing computed from the measured variances agrees with the independently measured single-seed gain to within 0.02 dB on every configuration of denoising, SR×2 and random inpainting. Box inpainting is the one cell where the single-seed measurement exceeds the prediction, by about 0.1 dB. It is also the noisiest cell, with a prediction interval about 2.7 times wider than on denoising. The prediction averages over eight seeds and

Table 16: Per-image bias-variance decomposition of Eq. (10) on CelebA 128^2 (100 test images, 8 seeds). MSE, squared bias and variance of a single trajectory are per pixel, $\times 10^{-3}$, image range $[-1, 1]$. The variance share is the fraction of the MSE due to variance for a single trajectory and for the paired $K{=}2$ estimator. The last two columns give the relative change caused by velocity correction.

		single trajectory			variance share		effect of velocity correction	
Task	correction	MSE	bias^2	variance	single	paired	bias^2	variance
Denoising	off	2.440	1.898	0.542	22.2%	0.6%	—	—
	on	2.386	1.843	0.543	22.8%	0.7%	−2.9%	+0.3%
Deblurring	off	1.369	1.130	0.238	17.4%	0.4%	—	—
	on	1.356	1.113	0.242	17.9%	0.5%	−1.5%	+1.8%
SR×2	off	2.540	2.061	0.479	18.8%	0.8%	—	—
	on	2.182	1.743	0.438	20.1%	0.8%	−15.4%	−8.4%
Random inpainting	off	2.233	1.726	0.507	22.7%	1.4%	—	—
	on	2.023	1.503	0.519	25.7%	1.5%	−12.9%	+2.4%
Box inpainting	off	3.865	3.260	0.605	15.7%	1.6%	—	—
	on	3.812	3.177	0.635	16.7%	1.8%	−2.5%	+5.0%

Table 17: Mechanism orthogonality on CelebA 128^2 (100 images per cell, $T{=}100$). Each cell reports PSNR (dB) for $K{=}2$ at a fixed $2T$ NFE budget, and the four runs of a row share measurements and seeds. The pairing gain Δ_{P} persists with velocity correction.

	correction off		correction on		Δ_{P}	Δ_{P}
Task	i.i.d.	paired	i.i.d.	paired	off	on
Denoising	32.804	33.363	32.907	33.482	+0.559	+0.575
SR×2	32.794	33.249	33.567	34.061	+0.455	+0.494
Random inpainting	33.666	34.238	34.241	34.906	+0.572	+0.665
Box inpainting	31.419	31.933	31.581	32.084	+0.514	+0.503
Mean	—	—	—	—	**+0.525**	**+0.559**

the measurement uses one, so the prediction is the more precise of the two numbers. Third, the initial-latent residual $\mathbb{E}[\sigma_\zeta^2]$ is below 4×10^{-4} of the total trajectory variance in every configuration. The solver has no memory of its initial latent, so pairing the initial latents has no effect. Pairing the re-injection noise, which the late steps retain because the proximal contraction is weak there, removes almost all of the remaining variance. The measurements also explain the task ordering. The odd share is smallest on box inpainting, where the prior must synthesise a contiguous region and its nonlinear response to the noise is largest, and this is the task with the smallest predicted gain.

Bias and variance of Eq. (10). The same runs measure both terms of the per-image decomposition. For a single trajectory A we have $\mathbb{E}\|A - \mathbf{x}_1\|^2 = \|\mathbb{E}A - \mathbf{x}_1\|^2 + \operatorname{tr}\operatorname{Var}(A)$, so the squared bias is the measured MSE minus the measured variance, and the variance of the paired estimator is $\frac{1}{2}(\operatorname{tr}\operatorname{Var}(A) + \operatorname{tr}\operatorname{Cov}(A, B))$. Here, squared bias measures the error of the seed-averaged reconstruction relative to the clean image. Tab. 16 reports these quantities with and without velocity correction. A single trajectory spends 16 to 26% of its MSE on variance, and antithetic pairing leaves 0.4 to 1.8%. After pairing the error is therefore almost entirely bias, further trajectories add little, and the paired curve of Fig. 6 is flat in K. Velocity correction lowers the squared bias on every task, by 15% on SR×2 and 13% on random inpainting, and by 1.5 to 3% on the other tasks, while the variance changes by less than 9%, which is less than 2% of the MSE. The two mechanisms thus act on different terms, as stated in the main text.

B.12 Mechanism orthogonality: paired-vs-i.i.d. benefit

To test whether the gain of antithetic pairing over $K{=}2$ independent averaging survives velocity correction, Tab. 17 crosses the correction-off and correction-on configurations with the independent-averaging and paired estimators at fixed $K{=}2$ and a $2T$ NFE budget.

Table 18: LSUN-Bedroom 256^2 on 100 test images with a public Rectified Flow checkpoint. The strengths are transferred from AFHQ-Cat without any selection on this dataset. The a=0 column is *PPC-Flow* without the velocity correction, at the same $2T$ NFE budget. Flower5-OT uses $5T$.

Task	T	a	Flower1-OT	Flower5-OT	a=0	*PPC-Flow*
Denoising	100	6	22.84	23.02	23.07	**24.49**
Deblurring	500	12	23.46	23.68	23.73	**24.12**
SR×4	500	12	16.34	16.41	16.40	**16.51**
Random inpainting	200	12	21.03	21.19	21.22	**21.27**
Box inpainting	100	0	22.63	22.75	**22.81**	**22.81**

Read row by row, the pairing gain persists when velocity correction is switched on, with $+0.56$ vs. $+0.58$ dB on denoising, $+0.46$ vs. $+0.49$ on SR×2, $+0.57$ vs. $+0.67$ on random inpainting, and $+0.51$ vs. $+0.50$ on box inpainting. Read column by column, the gain of velocity correction is similar on independent and paired trajectories. Antithetic pairing adds $+0.53$ dB without velocity correction and $+0.56$ dB with it. The two mechanisms are complementary and target the two error sources of Propositions 1 and 2, with a small positive interaction on random inpainting.

B.13 Extended qualitative comparison across all ten task–dataset cells

Fig. 9 adds qualitative examples beyond the main-text comparison in Fig. 4. It covers all ten task–dataset cells, with rows 1–5 for CelebA and rows 6–10 for AFHQ-Cat. The examples are disjoint from the main-text figure, and the PSNR against the clean image is shown below each tile.

Fig. 10 shows five representative task cells with Flower5-OT and *PPC-Flow* side by side.

C Supplementary Datasets and Tasks

C.1 Free-form (paintbrush) inpainting

Martin et al. (2025) use free-form (paintbrush) inpainting as a qualitative example. We add a supplementary comparison on CelebA 128^2 using the official paintbrush mask generator from Martin et al. (2025), with $\sigma_n = 0.05$ Gaussian observation noise and T=100 flow steps. The mask removes a continuous irregular region that covers about 20% of the pixels and therefore complements the random-pixel and box-mask settings in the main text. On the 100 test images *PPC-Flow* reaches 27.76 dB at $2T$ NFEs. At $5T$, Flower5-OT and PnP-Flow5 reach 27.46 dB and 26.85 dB, and at T, Flower1-OT and PnP-Flow1 reach 26.87 dB and 26.39 dB. Fig. 11 shows representative reconstructions of *PPC-Flow* and Flower5-OT under this setting.

C.2 Cross-domain evaluation on LSUN-Bedroom

The two main benchmarks, CelebA 128^2 and AFHQ-Cat 256^2, follow the public Flower evaluation stack of Pourya et al. (2026). As a cross-domain check, we also run *PPC-Flow* on LSUN-Bedroom 256^2 (Yu et al., 2015), an indoor-scene dataset with a public Rectified Flow (Liu et al., 2023) checkpoint trained outside the Flower codebase. The check thus tests the solver beyond faces and cats and beyond the backbone of the Flower benchmark.

Transferred strengths. LSUN-Bedroom has no validation split that is disjoint from its 100 public test images, so selecting a strength on it would reuse the test images. We instead transfer the strengths of the AFHQ-Cat 256^2 tasks unchanged, since LSUN-Bedroom shares its resolution. No strength is selected on LSUN. The transferred values are 6, 12, 12, 12 and 0 for denoising, deblurring, SR×4, random inpainting and box inpainting. On box inpainting the transferred value is zero, so that row is the paired solver alone.

Results. Tab. 18 shows that both mechanisms carry over to a different image domain and a different flow checkpoint. *PPC-Flow* at $2T$ NFE has the highest PSNR on all five tasks, and on the four tasks with a nonzero transferred strength it improves on Flower5-OT at $5T$ by 0.08 to 1.46 dB. Without velocity correction, the paired solver at $2T$ (the a=0 column) exceeds Flower1-OT on every task and comes within 0.01 dB of Flower5-OT at $5T$ or exceeds it.



Figure 9: Qualitative comparison across all ten task–dataset cells, with the per-image PSNR (dB) below each tile.

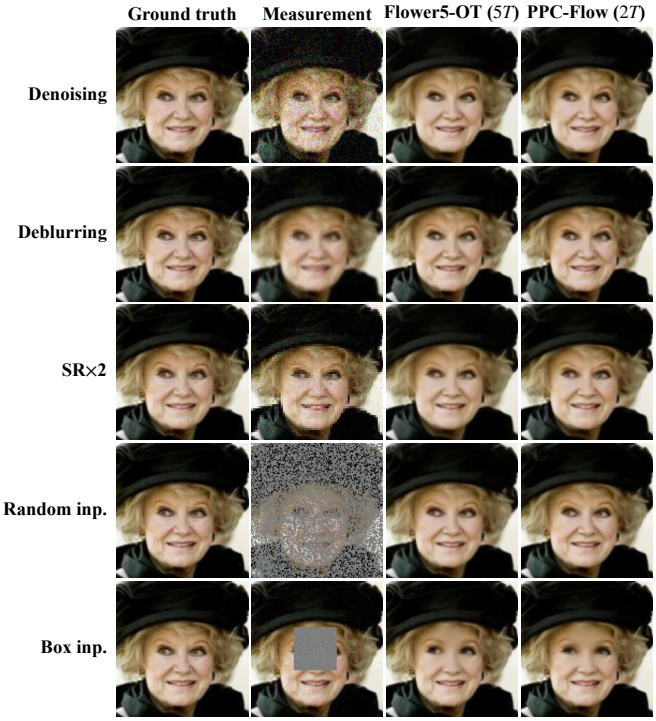

Figure 10: Additional qualitative comparison with Flower5-OT on five representative cells.

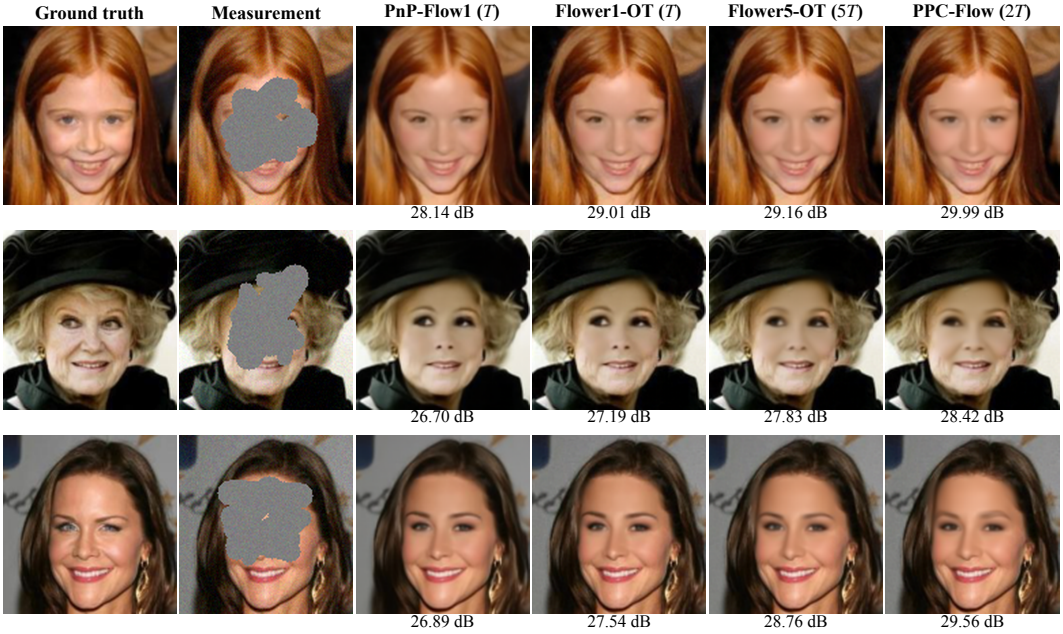

Figure 11: Free-form (paintbrush) inpainting on CelebA 128^2. Three representative cases compare *PPC-Flow* with Flower5-OT. The per-image PSNR (dB) is shown below each tile.

The transferred strength trades SSIM for PSNR on two tasks. On denoising it gains 1.42 dB of PSNR over $a{=}0$ and loses 0.032 of SSIM, the noise-amplification regime described in Appendix B.7. Denoising carries the highest measurement noise of the five tasks, $\sigma_n{=}0.2$, and over-relaxation amplifies that noise, which is why the validation search selects a small strength on CelebA denoising ($a{=}2$) rather than the $a{=}6$ that AFHQ-Cat selects. Transferring the larger value across datasets over-corrects. On SR$\times$4 it gains 0.11 dB and loses 0.014 of SSIM. On deblurring and random inpainting SSIM rises by 0.011 and 0.001.

The table reports PSNR only. The a=0 column and every SSIM value above come from our own runs. The two Flower columns come from the released evaluation code. Its PSNR agrees with our code path to within 0.001 dB at a=0 on all five tasks, whereas the two paths average SSIM differently and differ by about 0.002.

Fig. 12 shows representative LSUN-Bedroom 256^2 reconstructions from the public test split, with the same forward operators as in the main experiments.

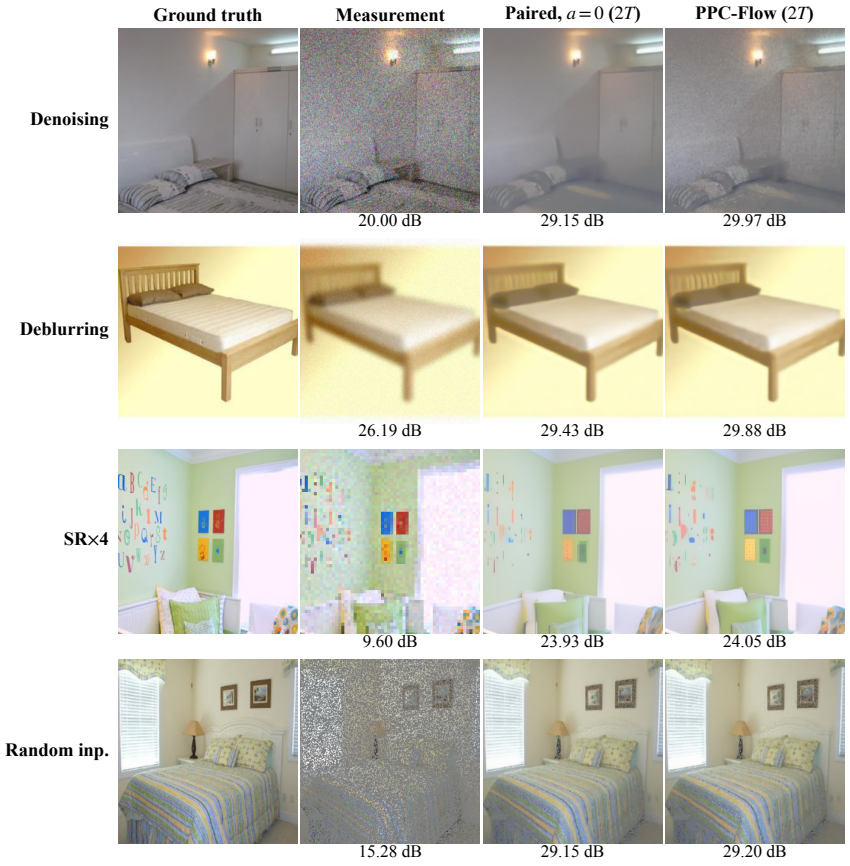

Figure 12: LSUN-Bedroom 256^2 qualitative comparison on four tasks. Each row shows a representative test image for one operator. The per-image PSNR (dB) is reported below each tile. The third column is the matched-K paired baseline without velocity correction (a=0).

C.3 Extension to nonlinear forward operators: blur with sensor saturation

The main text assumes a linear observation model $\mathbf{y} = \mathbf{H}\mathbf{x} + \sigma_n \boldsymbol{\epsilon}$. Linearity enters *PPC-Flow* in exactly two places, the residual term $\mathbf{H}^\top(\mathbf{y} - \mathbf{H}m_{\theta,t})$ of the velocity correction step (Eq. (11)) and the quadratic data term of the proximal step (Eq. (12)), whose minimiser is computed by conjugate gradients. Neither the antithetic pairing nor the variance identity of Proposition 2 uses linearity. This appendix extends both data steps to a differentiable nonlinear operator $\mathcal{A}$ through a per-step Gauss–Newton linearisation and evaluates the extension on blur followed by sensor saturation.

Gauss–Newton form of the two data steps. Let $\mathcal{A} : \mathbb{R}^d \to \mathbb{R}^M$ be differentiable with Jacobian $\mathbf{J}_\mathbf{x} := \partial\mathcal{A}(\mathbf{x})$. At the predicted endpoint $m_{\theta,t}$ we use the first-order model $\mathcal{A}(\mathbf{x}) \approx \mathcal{A}(m_{\theta,t}) + \mathbf{J}_m(\mathbf{x} - m_{\theta,t})$ with $\mathbf{J}_m := \mathbf{J}_{m_{\theta,t}}$. The velocity correction step keeps its form with the residual of the nonlinear model,

$$v_{\mathrm{corr}}(\mathbf{x}_t, t) = v_\theta(\mathbf{x}_t, t) + \alpha_t \frac{\mathbf{J}_m^\top\big(\mathbf{y} - \mathcal{A}(m_{\theta,t})\big)}{\sigma_n^2}, \tag{39}$$

Table 19: Gaussian blur followed by sensor saturation on CelebA 128^2 (100 test images, σ_n=0.05). GN marks the Gauss–Newton proximal step of Eq. (40). The ΔPSNR values are paired-bootstrap means with 95% CIs over images, and * marks intervals excluding zero.

Method	NFE	PSNR↑	SSIM↑	LPIPS↓	Δ vs Flower5-GN	Δ vs matched-K
PnP-Flow1 (Martin et al., 2025)	T	28.76	0.909	0.063	-1.12 $[-1.27, -0.99]^*$	-1.20 $[-1.35, -1.07]^*$
PnP-Flow5 (Martin et al., 2025)	$5T$	28.79	0.914	0.072	-1.10 $[-1.24, -0.97]^*$	-1.18 $[-1.32, -1.05]^*$
Flower1-GN	T	29.54	0.922	0.048	-0.35 $[-0.42, -0.28]^*$	-0.43 $[-0.51, -0.35]^*$
Flower5-GN (K=5 i.i.d.)	$5T$	29.89	0.930	0.054	0 (ref)	-0.08 $[-0.11, -0.06]^*$
Flower2-paired-GN (matched K, no velocity correction)	$2T$	29.97	0.931	0.056	$+0.08$ $[+0.06, +0.11]^*$	0 (ref)
PPC-Flow, K=1 (velocity correction only)	T	30.22	0.921	**0.038**	$+0.34$ $[+0.17, +0.51]^*$	$+0.25$ $[+0.08, +0.43]^*$
PPC-Flow (K=2)	$2T$	**30.76**	**0.933**	0.042	$+\mathbf{0.88}$ $[+0.75, +1.02]^*$	$+\mathbf{0.80}$ $[+0.66, +0.94]^*$
PPC-Flow, two GN iterations per step	$2T$	29.39	0.923	0.053	-0.50 $[-0.65, -0.36]^*$	-0.58 $[-0.72, -0.45]^*$

and the proximal step becomes the Gauss–Newton subproblem

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2\sigma_n^2}\left\|\tilde{\mathbf{y}} - \mathbf{J}_m\mathbf{x}\right\|^2 + \frac{1}{2\lambda_t}\left\|\mathbf{x} - \hat{\mathbf{x}}_1^{\textit{PPC-Flow}}\right\|^2, \qquad \tilde{\mathbf{y}} := \mathbf{y} - \mathcal{A}(m_{\theta,t}) + \mathbf{J}_m m_{\theta,t}, \quad (40)$$

which is the linear proximal problem of Eq. (12) with $(\mathbf{H}, \mathbf{y})$ replaced by $(\mathbf{J}_m, \tilde{\mathbf{y}})$ and is solved by the same matrix-free conjugate-gradient routine. Each CG iteration costs one Jacobian-vector and one vector-Jacobian product of the forward operator. The Jacobian of the velocity network is never formed, so the extension adds no network evaluation. With $\mathbf{J}_m$ in place of $\mathbf{H}$, Theorem 1 and Proposition 1 hold for the linearised model at each step, so the over-relaxation and its cap apply per step. Tab. 19 also reports a second linearisation at $\mathbf{x}^\star$, that is, two warm-started Gauss–Newton iterations per step.

Blur with saturation. We instantiate $\mathcal{A} = S \circ \mathbf{H}$ with $\mathbf{H}$ the Gaussian blur of the main deblurring task (σ_b=1, 61×61 kernel) and $S(z) = \min\{z, c\}$ a pixel-wise saturation clip at level c. The Jacobian is $\mathbf{J}_\mathbf{x} = \mathbf{D}_\mathbf{x}\mathbf{H}$ with $\mathbf{D}_\mathbf{x} = \mathrm{diag}\big(\mathbf{1}[\mathbf{H}\mathbf{x} < c]\big)$, so $\mathbf{J}_\mathbf{x}^\top u = \mathbf{H}^\top(\mathbf{D}_\mathbf{x} u)$ costs one blur adjoint. Since S is 1-Lipschitz, $0 \preceq \mathbf{D}_\mathbf{x} \preceq \mathbf{I}$ and therefore $\sigma_{\max}(\mathbf{J}_\mathbf{x}) \le \sigma_{\max}(\mathbf{H})$. The cap $\bar{a}_t$ of Eq. (17), computed once from $\mathbf{H}$ as in the linear case, remains valid for the nonlinear operator, and no per-step spectral estimate is needed. Pixels that the current endpoint predicts as saturated have zero Jacobian rows and drop out of the data term, so the saturated regions behave like an inpainting null space that the prior fills. Velocity correction gave its largest gains in the main text in exactly this inpainting-like regime. We use $c = 0.5$ in the $[-1, 1]$ image range, which saturates 17% of the blurred pixels on the CelebA test images, and $\sigma_n = 0.05$ as in the linear deblurring task.

Protocol. The correction strength is $a = 6$, selected on 32 CelebA validation images and fixed for test evaluation. We use CelebA 128^2 with 100 test images, T=100 flow steps, and the same backbone, seeds and evaluation code as Tab. 1. We compare against PnP-Flow and Flower, each at one and at five trajectories, and against a matched-budget paired baseline. PnP-Flow (Martin et al., 2025) applies to nonlinear operators through its gradient step $\mathbf{J}_\mathbf{x}^\top(\mathcal{A}(\mathbf{x}) - \mathbf{y})/\sigma_n^2$. Flower (Pourya et al., 2026) is restricted to linear operators, so we give it the same Gauss–Newton proximal step (Eq. (40)) and denote the result Flower-GN. The matched-budget baseline Flower2-paired-GN is *PPC-Flow* without velocity correction. All numbers are means over the 100 test images with paired-bootstrap 95% confidence intervals (Appendix B.5).

Results. Both mechanisms transfer. With the Gauss–Newton data steps, *PPC-Flow* at $2T$ NFE reaches 30.76 dB, which is +0.88 dB above Flower5-GN at $5T$ and +0.80 dB above the matched-budget paired baseline. Both intervals exclude zero. The per-image gain is positive on 93% of the test images with median +0.68 dB, and LPIPS improves from 0.056 for the paired baseline to 0.042. The two mechanisms split the gain as they do on the linear tasks. Velocity correction alone, at K=1 and T NFE, already exceeds the K=5 baseline by +0.34 dB, antithetic pairing adds a further +0.54 dB, and the paired baseline without velocity correction matches Flower5-GN at 40% of its NFE. PnP-Flow, whose gradient step is the standard way of handling a nonlinear operator without a proximal solve, trails Flower5-GN by 1.10 dB at the same $5T$ budget.

Two further results concern the linearisation and the saturation level. A second Gauss–Newton iteration per step lowers the PSNR of *PPC-Flow* from 30.76 to 29.39 dB. The hard clip makes the data term piecewise quadratic, and re-linearising at $\mathbf{x}^\star$ flips the saturation mask of the pixels near the clipping level, so the single linearisation at the predicted endpoint is both the cheaper and the better

choice. Per-image gains are uncorrelated with the saturated fraction, with correlation -0.08. Heavily saturated images, where more than 40% of the pixels are clipped, reach 16 to 23 dB for every method, since the clipped regions carry no measurement and are filled from the prior (Fig. 13, last row).

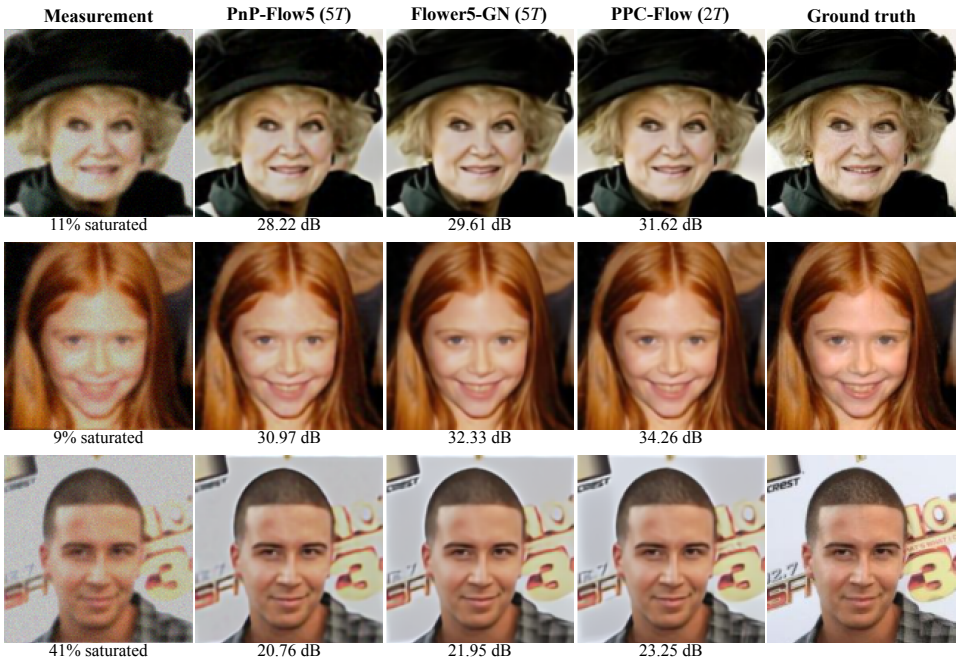

Figure 13: Blur with sensor saturation on CelebA 128^2 test images. The columns are the measurement, PnP-Flow5, Flower5-GN, *PPC-Flow* at $K{=}2$ and $2T$ NFE, and the clean image. The measurement uses Gaussian blur, a clip at $c{=}0.5$ in $[-1, 1]$, and $\sigma_n{=}0.05$. Its fraction of saturated pixels is given below it. Per-image PSNR (dB) is shown below each reconstruction. Saturated regions carry no measurement and are filled from the prior, so heavily saturated images reach low PSNR for every method.

Scope. The extension covers differentiable operators whose Jacobian-vector products are cheap relative to a velocity evaluation, which is the case for pixel-space sensor models such as blur with saturation, tone curves, and quantisation with a smooth surrogate. Latent-space flows with a nonlinear decoder and strongly nonconvex operators such as phase retrieval need several outer Gauss–Newton iterations per step and are left to future work.