VECTOR–JACOBIAN ENDPOINT GEOMETRY FOR INVERSE PROBLEMS: EXPLOITING THE LOCAL STRUCTURE OF FLOW PREDICTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained flow-matching models provide generative priors for inverse problems, but treating their clean predictions only as point estimates leaves local structure unused during measurement correction. We introduce Vector–Jacobian Endpoint Geometry (VEG), a clean-image correction that uses both measurements and local flow structure to determine where to move, how far to move, and whether to accept the move. We call the local variation among clean endpoints compatible with the current flow state *endpoint geometry*. VEG uses a clean-prediction vector–Jacobian product for direction, norm matching for proposal scale, and safeguarded line search for step selection. Our analysis characterizes how measurement evidence should shift the predicted clean image among these plausible endpoints, and shows that this shift is shaped by their local variation. Experiments on nonlinear and linear inverse problems show improved reconstruction quality over ordinary clean-prediction correction with modest additional local computation, while controlled interventions demonstrate that endpoint geometry can guide image changes that the measurement alone cannot determine.

1 INTRODUCTION

Pretrained diffusion and flow models provide learned image priors for inverse problems (Chung et al., 2023; Kim et al., 2025a; Martin et al., 2025; Pourya et al., 2026). A central challenge is to exploit the structure of these models without making reconstruction prohibitively expensive. Many methods therefore make local measurement-driven corrections during generation, either to the current noisy state or to a predicted clean image. Clean-prediction correction is especially simple: the pretrained model supplies a clean estimate, and the measurement loss determines how that estimate should be changed. This is computationally attractive, but uses the prediction only as a point estimate and ignores how it varies around the current flow state. At the other extreme, whole-trajectory optimization differentiates through the full generation path at substantially higher cost (Ben-Hamu et al., 2024). We ask whether a local clean-image correction can use this additional flow structure while retaining a lightweight per-step procedure.

A clean prediction is more than a point estimate. At a given flow state, multiple clean endpoints can remain plausible; we call the local variation among them *endpoint geometry*. We introduce VEG, which combines endpoint geometry with measurement information to choose the correction direction, proposal scale, and accepted step. A clean-prediction vector–Jacobian product provides the correction direction, norm matching controls its proposal scale, and a descent test with safeguarded line search determines the accepted step. Our analysis characterizes how measurement evidence shifts the predicted clean image among plausible endpoints. A local approximation shows that this change is shaped by endpoint covariance, whose action is accessible through the clean-prediction differential for the ideal linear flow-matching path. This provides a local, matrix-free way to use more of the pretrained flow than its point prediction alone.

Our experiments show that VEG achieves a practical balance between reconstruction quality and computation across nonlinear and linear inverse problems. On the held-out nonlinear benchmark, it improves PSNR by about 1 dB over matched ordinary clean-prediction correction, and remains higher PSNR than substantially deeper ordinary refinement. Compared with published solvers, VEG operates

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.

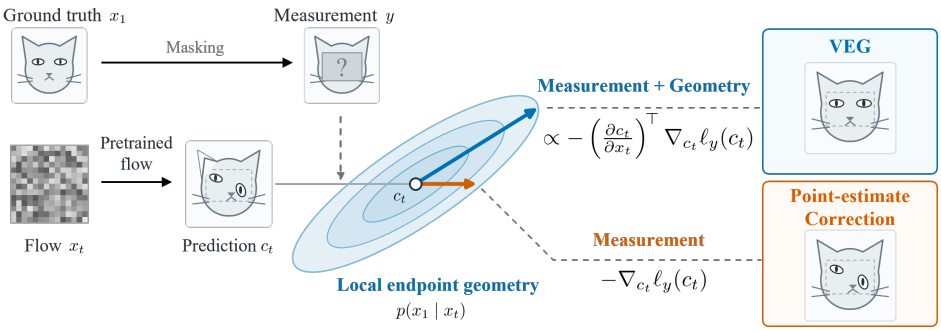


Figure 1: Conceptual illustration of VEG. Given a flow state x_t, the pretrained flow predicts a clean image c_t. VEG uses the local variation of plausible clean endpoints under $p(x_1 \mid x_t)$ to reshape the measurement gradient into a geometry-informed correction direction $\propto -(\partial c_t/\partial x_t)^\top \nabla_{c_t} \ell_y(c_t)$. Here, ℓ_y denotes the measurement loss. VEG applies this direction to the predicted clean image, combining measurement information with local endpoint geometry. The ellipses depict local variation of plausible clean endpoints, and the reconstruction images are schematic.

in a markedly lower-compute regime than full-trajectory optimization; D-Flow (Ben-Hamu et al., 2024) achieves stronger fidelity on several metrics but requires roughly 20–30$\times$ the mean runtime across our nonlinear and linear benchmarks. Without retuning, VEG transfers to Gaussian deblurring, $4\times$ super-resolution, and box inpainting, with particularly strong results on super-resolution and inpainting. Controlled ablations and interventions investigate the mechanism behind these gains, showing that the flow-derived correction can guide image changes that the measurement objective alone cannot distinguish. The PSNR effect also transfers to an independently trained JiT-B/16 model (Li & He, 2026) on ImageNet (Deng et al., 2009).

Contributions.

1. **Deriving the endpoint-geometry view of measurement correction.** We connect measurement-driven changes among plausible clean endpoints to their local covariance structure, whose action is accessible through the clean-prediction differential for the ideal linear flow-matching path.
2. **Constructing a safeguarded one-VJP clean-image update.** VEG uses one clean-prediction VJP per positive-time interval to choose the correction direction, controls its proposal scale by norm matching, and uses a descent test with safeguarded line search to determine whether and how far to move before continuing generation. It requires neither retraining nor full-trajectory differentiation.
3. **Characterizing empirical gains, mechanism, and transfer.** Across the primary nonlinear and linear benchmarks, VEG achieves higher PSNR than ordinary clean-prediction correction at moderate additional computation. Targeted ablations and interventions probe the source of these gains, including cases where multiple image changes are indistinguishable to the measurement objective. The PSNR effect also persists with an independently trained predictor.

2 Background

2.1 Flow matching

Flow matching learns a time-dependent velocity field that transports a simple noise distribution to a data distribution (Lipman et al., 2023). Let p_{data} be the distribution of clean images in $\mathbb{R}^d$. Independently drawing $X_0 \sim \mathcal{N}(0, I_d)$ and $X_1 \sim p_{\text{data}}$ as Gaussian noise and a clean image, respectively, we form the linear probability path

$$X_t = tX_1 + (1-t)X_0, \qquad t \in [0,1]. \tag{1}$$

Along each paired path, the velocity is $\dot{X}_t = X_1 - X_0$. Since the current state X_t does not uniquely determine the underlying pair (X_0, X_1), the ideal velocity field is the conditional mean $v^\star(x,t) = \mathbb{E}[X_1 - X_0 \mid X_t = x]$.

A neural velocity field $v_\theta(x,t)$ approximates $v^\star(x,t)$. After training, samples are generated from $x_0 \sim \mathcal{N}(0, I_d)$ by integrating $\dot{x}_t = v_\theta(x_t, t)$ from $t = 0$ to 1. For the linear path, v_θ is trained with the conditional flow-matching (CFM) objective

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,X_0,X_1}\left[\left\|v_\theta(X_t,t) - (X_1 - X_0)\right\|^2\right], \qquad t \sim \mathrm{Unif}[0,1], \tag{2}$$

whose squared-error minimizer is $v^\star$. The learned velocity also induces the clean-image prediction

$$c_t = D_t(x_t) := x_t + (1-t)v_\theta(x_t,t). \tag{3}$$

2.2 Flow-based inverse problems

An inverse problem seeks an unknown clean image $x^\star \in \mathbb{R}^d$ from a measurement $y \in \mathbb{R}^m$. A known forward operator $h : \mathbb{R}^d \to \mathbb{R}^m$ describes the measurement process, such as blur, downsampling, or clipping. We use the observation model

$$y = h(x^\star) + \eta, \qquad \eta \sim \mathcal{N}(0, \Gamma), \tag{4}$$

where $\Gamma \in \mathbb{R}^{m\times m}$ is a positive-definite noise covariance. With the weighted squared norm $\|r\|_A^2 := r^\top A r$, the Gaussian noise model gives the measurement loss

$$\ell_y(x) = \frac{1}{2}\left\|h(x) - y\right\|_{\Gamma^{-1}}^2. \tag{5}$$

The corresponding likelihood is $p(y \mid x) \propto \exp[-\ell_y(x)]$. A generative prior represents plausible clean images, while the likelihood measures how well they match the observation.

Along the flow trajectory, the clean-prediction map D_t produces a clean-image prediction $c_t = D_t(x_t)$ at each state x_t. A simple local reconstruction step applies ordinary gradient correction using the measurements:

$$g_t = \nabla \ell_y(c_t), \qquad c_t^+ = c_t - \alpha_t g_t, \tag{6}$$

where $\alpha_t \geq 0$ is a step length. For the linear flow-matching path in (1), the ideal clean-prediction map satisfies

$$D_t^\star(x_t) = x_t + (1-t)v^\star(x_t,t) = \mathbb{E}[X_1 \mid X_t = x_t]. \tag{7}$$

Thus, the ideal flow prediction is the conditional mean of possible clean images given the current flow state, and the neural map D_t approximates this mean.

Ordinary gradient correction uses the flow to obtain the current prediction, but chooses how to move it using only the measurement gradient. We next ask whether the local variation of D_t can also inform that direction.

3 Flow Prediction Geometry for Measurement Correction

3.1 How measurements change the conditional mean

Fix a state x_t with $t \in (0,1)$, and let $q_t(x_1 \mid x_t)$ denote the conditional distribution of the clean image X_1 given $X_t = x_t$ under the linear probability path in (1). The clean image is the *endpoint* of this noise-to-image path. Given the current state x_t, $q_t(\cdot \mid x_t)$ describes the remaining uncertainty over that endpoint; we refer to the local structure of this uncertainty as *endpoint geometry*. Conditioning additionally on the measurement y gives, by Bayes' rule,

$$p(x_1 \mid x_t, y) \propto q_t(x_1 \mid x_t)\, p(y \mid x_1) \propto q_t(x_1 \mid x_t) \exp[-\ell_y(x_1)]. \tag{8}$$

Appendix A.1 gives the derivation under the observation model.

To isolate the initial effect of the measurement on the conditional mean, we connect the original endpoint distribution to the measurement-conditioned one using likelihood tempering (Friel & Pettitt, 2008), an exponential reweighting strategy also used in recent diffusion sampling (Kim et al., 2025b):

$$\pi_{t,\beta}(x_1) = \frac{q_t(x_1 \mid x_t)\exp[-\beta \ell_y(x_1)]}{Z_t(\beta)}, \qquad m_t(\beta) = \mathbb{E}_{\pi_{t,\beta}}[X_1], \tag{9}$$

where $Z_t(\beta)$ normalizes the distribution and $\beta \geq 0$ controls the influence of the measurement. At $\beta = 0$, $\pi_{t,0} = q_t(\cdot \mid x_t)$, while at $\beta = 1$, $\pi_{t,1} = p(\cdot \mid x_t, y)$. Thus, $m_t'(0)$ describes the initial change in the conditional endpoint mean as the measurement is introduced. This change has the following exact form.

Proposition 1 (Measurement-induced change in the conditional endpoint mean)**.** *Assume the relevant moments exist and differentiation under the integral is valid. Then,*

$$\left.\frac{\mathrm{d}m_t(\beta)}{\mathrm{d}\beta}\right|_{\beta=0} = m_t'(0) = -\operatorname{Cov}_{q_t}(X_1, \ell_y(X_1))\,. \tag{10}$$

Here, $\operatorname{Cov}_\mu(U, V) := \mathbb{E}_\mu[(U - \mathbb{E}_\mu[U])(V - \mathbb{E}_\mu[V])]$ *for vector-valued* U *and scalar* V*.*

Proof. As β increases, the factor $e^{-\beta \ell_y(x_1)}$ reduces the weight of endpoints with larger measurement loss more strongly. Since $m_t(\beta)$ is the mean under these normalized weights, differentiating it gives $m_t'(\beta) = -\mathbb{E}_{\pi_{t,\beta}}[X_1 \ell_y(X_1)] + \mathbb{E}_{\pi_{t,\beta}}[X_1]\mathbb{E}_{\pi_{t,\beta}}[\ell_y(X_1)] = -\operatorname{Cov}_{\pi_{t,\beta}}(X_1, \ell_y(X_1))$. At $\beta = 0$, $\pi_{t,0} = q_t$, yielding (10). The full derivation is given in Appendix A.2. □

The covariance captures which endpoint variations tend to increase or decrease the measurement loss; the negative sign moves the mean toward variations with lower loss. We next approximate the measurement loss locally to turn this mean change into a practical correction direction.

3.2 The ideal covariance-shaped direction

To express the measurement-induced mean change locally, let $\bar{c}_t = \mathbb{E}[X_1 \mid X_t = x_t]$ be the ideal conditional mean, $C_t = \operatorname{Cov}(X_1 \mid X_t = x_t)$ its covariance, and $\bar{g}_t = \nabla \ell_y(\bar{c}_t)$ the measurement gradient at that mean. Linearizing the measurement loss around $\bar{c}_t$ gives

$$\ell_y(X_1) = \ell_y(\bar{c}_t) + \bar{g}_t^\top (X_1 - \bar{c}_t) + r_t(X_1), \tag{11}$$

where r_t is the Taylor remainder. Substituting this expansion into (10) and using the linearity of covariance, the constant term vanishes, while $\operatorname{Cov}_{q_t}(X_1, \bar{g}_t^\top (X_1 - \bar{c}_t)) = C_t \bar{g}_t$ by the definitions of $\bar{c}_t$ and C_t. Hence,

$$m_t'(0) = -C_t \bar{g}_t - \operatorname{Cov}_{q_t}(X_1, r_t(X_1)). \tag{12}$$

Neglecting the Taylor-remainder term gives

$$m_t'(0) \approx -C_t \bar{g}_t. \tag{13}$$

The covariance therefore reshapes the measurement gradient according to how plausible endpoints vary around their conditional mean, changing both its direction and scale. If C_t is a positive multiple of the identity, this reduces to ordinary descent $-\bar{g}_t$. This argument applies whenever the measurement loss is differentiable around $\bar{c}_t$; Appendix A.3 derives the remainder and its exact form for linear measurements.

The covariance can couple changes across image coordinates. For inpainting, the measurement gradient vanishes on missing pixels, but conditional correlations can produce a nonzero correction there. Section 6.3 tests whether these components of the learned correction improve reconstruction. However, the ideal direction $-C_t \bar{g}_t$ is not directly available because the state-dependent covariance C_t is high-dimensional and not explicitly represented by the pretrained flow. We next obtain its action on the gradient from the clean-prediction map without estimating or storing C_t.

3.3 Obtaining the covariance-shaped direction from the flow prediction

To compute the covariance-shaped direction $-C_t \bar{g}_t$ without forming C_t, we use the Jacobian of the ideal clean-prediction map $D_t^\star$ in (7). For fixed t, let $J_{D_t^\star}(x_t) := \partial D_t^\star(x_t)/\partial x_t$. A backward pass computes $J_{D_t^\star}(x_t)^\top a$ for a vector a without forming the Jacobian explicitly; this is a vector–Jacobian product (VJP).

Proposition 2 (Jacobian–covariance identity for the linear flow-matching path)**.** *For the linear flow-matching path in* (1)*, with independent Gaussian noise* X_0 *and clean images* X_1*,*

$$J_{D_t^\star}(x_t) = \frac{t}{(1-t)^2} C_t \quad \Longrightarrow \quad C_t \bar{g}_t = \frac{(1-t)^2}{t} J_{D_t^\star}(x_t)^\top \bar{g}_t, \qquad t \in (0, 1). \tag{14}$$

Proof. Under the linear flow-matching path, the conditional endpoint density can be written as an exponential family, i.e., $q_t(x_1 \mid x_t) \propto h_t(x_1)\exp(\theta_t^\top x_1)$, with $\theta_t = tx_t/(1-t)^2$. If $A_t(\theta)$ denotes the log normalizer of this density, standard exponential-family identities give $\nabla_\theta A_t(\theta_t) = D_t^\star(x_t)$ and $\nabla_\theta^2 A_t(\theta_t) = C_t$. Since θ_t is linear in x_t, the chain rule yields $J_{D_t^\star}(x_t) = tC_t/(1-t)^2$. Since C_t is symmetric, the second relation follows. An affine-Gaussian form of this Jacobian–covariance identity was derived in Proposition 4.1 of Ben-Hamu et al. (2024); Appendix A.4 gives the full exponential-family derivation for the linear path considered in our paper. □

This identity turns the ideal covariance-shaped correction into a quantity accessible by automatic differentiation:

$$m_t'(0) = -\operatorname{Cov}_{q_t}(X_1, \ell_y(X_1)) \approx -C_t\bar{g}_t \propto -J_{D_t^\star}(x_t)^\top \bar{g}_t.$$

The first equality is exact, whereas the approximation $m_t'(0) \approx -C_t\bar{g}_t$ comes from locally linearizing the measurement loss around the conditional mean. Thus, the covariance-shaped direction can be obtained locally through a VJP of the clean-prediction map, without forming either C_t or the full Jacobian.

4 Vector–Jacobian Endpoint Geometry

4.1 Choosing and scaling the correction direction

In practice, we replace the ideal map $D_t^\star$ with the pretrained clean-prediction map D_t in (3). Given the current flow state x_t, the pretrained neural map predicts $c_t = D_t(x_t)$, with measurement gradient $g_t = \nabla\ell_y(c_t)$. We compute the flow-derived vector by a vector–Jacobian product (VJP),

$$v_t = J_{D_t}(x_t)^\top g_t = \nabla_{x_t}\ell_y(D_t(x_t)), \tag{15}$$

which reverse-mode automatic differentiation computes without forming the Jacobian. In the linear flow-matching setting of Section 3, our endpoint-mean analysis independently arrives at the same chain-rule form used in DPS (Chung et al., 2023): after locally linearizing the measurement loss, the measurement-induced change in the conditional endpoint mean is directed by $-J_{D_t^\star}(x_t)^\top\bar{g}_t$ up to a positive scalar. DPS uses this vector to update the current noisy state, whereas VEG applies it directly to the predicted clean image. These two placements generally induce different clean-image changes: to first order, a noisy-state update gives a change proportional to $-J_{D_t}(x_t)J_{D_t}(x_t)^\top g_t$, whereas direct clean-image correction uses $-J_{D_t}(x_t)^\top g_t$. Under the ideal linear-path identity, these correspond to $-C_t^2\bar{g}_t$ and $-C_t\bar{g}_t$, respectively. In our image-space setting, the flow state and predicted clean image share coordinates, allowing the latter direction to be applied directly to the clean prediction.

For the pretrained neural map, however, the neural Jacobian may differ from the ideal covariance in scale and symmetry, so we use it to choose the direction and control the update through norm matching and safeguarded acceptance. With a numerical stabilizer $\epsilon > 0$, VEG defines

$$d_t^{\text{VEG}} = -\|g_t\|\,\frac{v_t}{\|v_t\| + \epsilon}. \tag{16}$$

This *norm matching* preserves the flow-derived direction while controlling its scale; Appendix B.4 compares the direction and scaling choices.

4.2 Safeguarding the correction and resuming generation

Norm matching controls the scale of the flow-derived direction, but a neural predictor need not preserve the ideal covariance geometry exactly. We therefore safeguard each finite correction with a local objective, a descent test, and a line search. Let N be the number of outer intervals and $t_n = n/N$ for $n = 0, \ldots, N$. At interval $n = 0, \ldots, N-1$, let x_n be the current flow state and $c_n = D_{t_n}(x_n)$ its predicted clean image. To discourage large departures from this prediction while leaving the gradient at c_n unchanged, we use

$$U_n(c) = \ell_y(c) + \frac{\lambda_n}{2}\|c - c_n\|^2, \qquad \lambda_n > 0. \tag{17}$$

At c_n, the quadratic term has zero gradient, so $g_n = \nabla U_n(c_n) = \nabla \ell_y(c_n)$ remains the measurement gradient. For $t_n > 0$, we compute $v_n = J_{D_{t_n}}(x_n)^\top g_n$. We use the norm-matched VEG direction only when g_n and v_n are finite and nonzero and $g_n^\top v_n > 0$. Since VEG moves opposite to v_n, this condition gives $g_n^\top d_n^{\text{VEG}} < 0$, so a sufficiently small step along the VEG direction decreases the local objective. At $t_0 = 0$, where Proposition 2 does not apply, and whenever the VJP is nonfinite, zero, or fails this descent test, we instead use ordinary descent $-g_n$. Appendix A.5 gives the derivation.

The descent test screens out directions that would not decrease the local objective near c_n, while a safeguarded line search determines how far to move by requiring the chosen finite step to decrease U_n. If the VEG direction does not yield an acceptable step, we repeat the line search with ordinary descent; if that also fails, the prediction remains unchanged. Let d_n denote the resulting direction and $\alpha_n \geq 0$ the accepted step length, giving the corrected prediction $\widetilde{c}_n = c_n + \alpha_n d_n$. Appendix B.1 specifies the schedule for λ_n and the line-search details. After the correction, we resume the flow by combining the corrected clean prediction with fresh Gaussian noise at the next flow time. Drawing $z_{n+1} \sim \mathcal{N}(0, I)$, we form

$$x_{n+1} = t_{n+1}\widetilde{c}_n + (1 - t_{n+1})z_{n+1}, \qquad t_{n+1} = \frac{n+1}{N}. \tag{18}$$

This continuation mixes the corrected clean prediction with fresh noise at the next flow time, allowing VEG to modify the local correction while retaining the underlying generative progression. Algorithm 1 in Appendix B gives the complete procedure.

Computational cost. Over N outer intervals, VEG adds one VJP through the clean-prediction map at each positive-time interval and permits at most one accepted clean-image correction per interval. It does not form a covariance matrix, solve a linear system, differentiate through the full trajectory, or optimize the initial noise; exact evaluated counts appear in table 9.

5 Related Work

Local correction and trajectory optimization. Local inverse solvers differ in where and how measurement information is applied. DPS differentiates a measurement loss through a clean-image estimate and uses the resulting gradient to update the noisy state (Chung et al., 2023). Clean-prediction methods instead refine an estimated clean image before continuing generation (Kim et al., 2025a; Martin et al., 2025; Pourya et al., 2026). VEG also operates on the predicted clean image, but uses a clean-prediction VJP to shape the correction direction. Whole-trajectory methods such as D-Flow optimize through the full generator (Ben-Hamu et al., 2024), while DAPS and ReSample use additional reconstruction or sampling procedures for posterior inference (Zhang et al., 2025; Song et al., 2024).

Prior work on Jacobian-aware conditioning. Diffusion methods already exploit structure beyond isotropic uncertainty approximations (Peng et al., 2024; Boys et al., 2024; Rout et al., 2024; Rissanen et al., 2025; Zirvi et al., 2025; Dou et al., 2025). Most directly, JAPS differentiates a diffusion clean-image estimator to construct posterior guidance along a diffusion trajectory (Hen et al., 2026). VEG instead connects a clean-prediction VJP to a local approximation of the measurement-induced change in the conditional endpoint mean and applies the resulting direction in clean-image space. Appendix B.7 gives a detailed comparison.

Flow prediction and uncertainty. FLOWER refines a flow-predicted clean image through a measurement-aware proximal step before returning to the flow (Pourya et al., 2026). Ben-Hamu et al. (2024) derive a Jacobian–covariance identity for affine Gaussian probability paths in their analysis of full-trajectory optimization. Our analysis starts from the measurement-induced change in the conditional endpoint mean; locally linearizing the measurement loss yields a covariance-shaped correction direction, which the linear-path identity then relates to the differential of the clean-prediction map. Xing et al. (2026) study the same linear-path identity for uncertainty quantification.

6 Experiments

We evaluate VEG through reconstruction quality and computational cost, matched controls and interventions that probe the flow-derived direction, and transfer to independently trained predictors.

Table 1: Nonlinear reconstruction by dataset and task (30 held-out cases per group). Time is mean GH200 seconds per case. FLOWER, D-Flow, and DAPS use our same-prior nonlinear adaptations. Bold and underlining mark the best and second-best results within each group.

Method	Nonlinear blur				HDR saturation			
	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓
(a) AFHQ-Cat (256^2)								
VEG (Ours)	26.2540	**0.7023**	0.33897	**21.81**	<u>26.4524</u>	**0.8714**	<u>0.12034</u>	**9.10**
FLOWER-NL	21.4791	0.5676	0.50031	1438.30	24.8909	0.8089	0.21247	55.47
D-Flow-NL	**26.8585**	0.6778	**0.15397**	<u>293.36</u>	**27.5683**	<u>0.8481</u>	**0.08816**	304.46
DAPS (adapt.)	<u>26.2597</u>	<u>0.6832</u>	<u>0.20584</u>	627.22	24.7409	0.8253	0.13674	<u>47.61</u>
(b) CelebA (128^2)								
VEG (Ours)	27.7335	<u>0.8281</u>	0.10978	**26.61**	<u>23.7148</u>	**0.8553**	<u>0.06694</u>	**10.92**
FLOWER-NL	27.1466	0.8039	0.18269	1452.33	21.7830	0.7961	0.12558	59.57
D-Flow-NL	**29.3154**	0.8165	**0.05003**	<u>373.48</u>	**27.1637**	<u>0.8518</u>	**0.04368**	363.67
DAPS (adapt.)	<u>28.6581</u>	**0.8345**	<u>0.05456</u>	637.88	22.3796	0.8040	0.08753	<u>54.83</u>

6.1 EXPERIMENTAL SETUP

We evaluate frozen image-space rectified-flow models on AFHQ-Cat (Choi et al., 2020) and CelebA (Liu et al., 2015) without finetuning. The primary nonlinear benchmark contains 120 cases held out from method development, with 30 images per dataset evaluated under both nonlinear blur and HDR saturation. Linear transfer uses the same 10 images per dataset across Gaussian deblurring, $4\times$ super-resolution, and box inpainting, without retuning. Nonlinear blur uses the fixed pretrained KernelWizard model (Tran et al., 2021), while HDR uses elementwise clipping; all observations have Gaussian noise $\sigma_y = 0.05$. All paired comparisons use the same images, observations, pretrained models, and random draws. We report PSNR, SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018), and algorithm runtime. Appendix D gives further operator and baseline settings, metric conventions, measurement NRMSE, paired uncertainty estimates, and additional qualitative comparisons.

6.2 RECONSTRUCTION QUALITY AND COMPUTATIONAL COST

Tables 1 and 2 evaluate VEG as a reconstruction method. On the nonlinear benchmark, D-Flow-NL attains higher PSNR and lower LPIPS at substantially greater computational cost. Across the four equally weighted dataset–task groups, VEG uses about 5% of D-Flow-NL's mean runtime and has higher mean PSNR than FLOWER-NL and the DAPS adaptation using the same pretrained flow; Table 1 shows the task-specific tradeoffs. Without retuning, VEG has the best displayed PSNR and LPIPS for $4\times$ super-resolution and box inpainting on both datasets. Gaussian deblurring is mixed: FLOWER leads PSNR, while LPIPS favors D-Flow on AFHQ-Cat and FLOWER on CelebA. Figure 2a illustrates representative nonlinear and linear reconstructions, and Appendix D.3 reports the 400-case extension.

6.3 WHAT INFORMATION DOES ENDPOINT GEOMETRY ADD?

Direction and depth. Table 3 summarizes matched controls that test whether the gain comes from the flow-derived direction rather than additional local refinement. The ordinary-gradient control replaces VEG's direction with the gradient of the same local objective while keeping the predictor, acceptance rule, continuation, and random draws fixed. The PSNR advantage persists against deeper and restarted ordinary refinement; relative to ten ordinary corrections, VEG uses $0.642\times$ the runtime, while the LPIPS difference is unresolved. Angle-matched random directions do not reproduce the gain.

The 400-case extension retains PSNR gains of 0.93 dB and 0.47 dB over two- and ten-correction ordinary refinement, respectively. On the newly added images alone, the gains remain 0.87 dB and 0.43 dB. Appendix D.3 reports the full larger-scale evaluation. Appendix B.4 compares VJP scaling and alternative Jacobian products, and Appendix B.5 examines noisy-state correction and continuation choices.

Table 2: Untuned linear transfer with 10 paired cases per dataset and task. Time is mean GH200 seconds per case. Bold and underlining mark the best and second-best values within each group. Appendix D.2 gives measurement NRMSE and additional paired comparisons.

Method	Gaussian deblurring				$4\times$ super-resolution				Box inpainting			
	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Time ↓
(a) AFHQ-Cat (256^2)												
VEG (Ours)	<u>28.6832</u>	<u>0.7709</u>	0.32556	<u>9.02</u>	**28.3395**	**0.8011**	**0.20883**	<u>8.73</u>	**27.2329**	**0.9240**	**0.05934**	<u>8.76</u>
FLOWER	**29.1593**	**0.7908**	0.29074	**6.35**	<u>25.3190</u>	<u>0.7073</u>	0.31737	**3.69**	26.1211	<u>0.9209</u>	<u>0.06800</u>	**3.69**
PnP-Flow	28.6308	0.7686	0.39338	17.22	23.4164	0.6510	0.42651	17.10	26.5385	0.9098	0.14047	17.24
D-Flow	27.7652	0.7003	**0.21277**	437.43	25.0373	0.6115	<u>0.25123</u>	436.86	<u>26.7514</u>	0.8533	0.08739	191.91
Flow Priors	26.8045	0.7032	<u>0.23564</u>	65.39	23.7670	0.5894	0.32867	65.38	26.6139	0.8203	0.14235	65.40
(b) CelebA (128^2)												
VEG (Ours)	<u>35.0641</u>	**0.9506**	<u>0.03522</u>	<u>10.89</u>	**27.3621**	**0.8439**	**0.09019**	<u>10.47</u>	**31.8756**	**0.9540**	**0.01761**	<u>10.70</u>
FLOWER	**35.0842**	<u>0.9488</u>	**0.02802**	**7.08**	23.0522	<u>0.6820</u>	0.15009	**4.43**	<u>31.8066</u>	<u>0.9488</u>	<u>0.01878</u>	**4.36**
PnP-Flow	34.9594	0.9408	0.05445	20.63	20.7165	0.5971	0.21750	20.84	31.3422	0.9458	0.03295	20.60
D-Flow	31.1520	0.8468	0.03933	188.65	<u>24.5692</u>	0.6579	<u>0.10071</u>	266.93	30.9373	0.9053	0.02159	239.40
Flow Priors	31.4555	0.8567	0.06553	61.31	22.4823	0.5441	0.20054	61.82	29.9532	0.8574	0.04796	61.83

Table 3: Direction/depth controls and a noisy-state guidance comparison (ΔPSNR: VEG minus the listed variant). Counts denote cases in separate held-out and development cohorts. Brackets give 95% CIs; the random-direction row gives the range across repeated angle-matched random controls. Correction counts are maxima per interval.

Control	Cohort (cases)	ΔPSNR (dB)
Ordinary gradient, 1 correction	Held-out 120	$+1.069$ $[0.828, 1.334]$
Ordinary gradient, 10 corrections	Held-out 120	$+0.576$ $[0.405, 0.749]$
Ordinary gradient, 10 corrections + restart	Development 40	$+1.089$ $[0.777, 1.361]$
Angle-matched random direction	Development 40	$+2.054$–$+2.182$
Normalized noisy-state guidance	Development 40	$+1.945$ $[0.903, 2.988]$

Guidance where measurements are ambiguous. The previous controls show that the flow-derived direction matters. We next ask what information it contributes when the measurement objective cannot distinguish among image changes. In inpainting, changes to missing pixels leave the observation unchanged; in HDR, values can move within the same clipping plateau. We call the corresponding part of an update the *measurement-indifferent component*. Keep, Remove, and Reverse alter only this component of an accepted VEG update. At the same state, all candidates share the measurement prediction, while Keep and Reverse also match in update norm and local quadratic penalty.

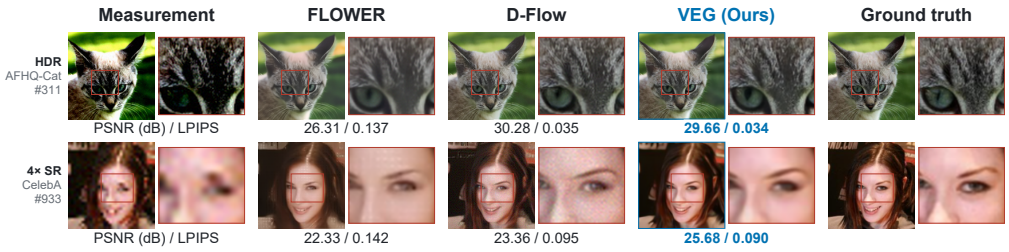
Table 4 shows that retaining the flow-selected component improves mean reconstruction quality over removing it, whereas reversing its direction causes a larger degradation. The aggregate PSNR and region-MSE effects have 95% confidence intervals excluding zero for both tasks and both interventions. Appendix C reports the full intervention results and dataset-specific effects, and Figure 2b illustrates representative examples. In these two tasks, endpoint geometry therefore provides useful information for choosing among image changes that the local measurement objective cannot distinguish. Appendix B gives the full solver controls.

6.4 TRANSFER AND SCOPE

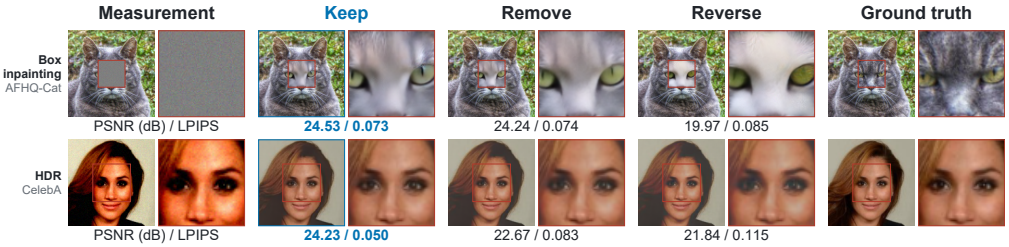
To test transfer beyond the primary pretrained flows, we apply VEG to the independently trained class-conditional JiT-B/16 ImageNet predictor (Li & He, 2026), keeping its weights fixed and using ground-truth class labels for all compared methods. Across 40 previously unused images evaluated on inpainting and HDR, VEG improves PSNR over one ordinary correction by 1.475 dB, with a 95% confidence interval of $[0.851, 2.127]$. The gain remains 1.52 dB relative to up to ten ordinary corrections and 1.11 dB relative to ten corrections with restarted line searches. These results support transfer of the PSNR effect to an independently trained predictor. The aggregate LPIPS difference is unresolved, while inpainting LPIPS worsens; Appendix E.2 reports the full paired results. Appendix E.3 gives the broader five-task comparison with published JiT-based solvers and the resulting task-dependent quality–runtime tradeoffs.

Table 4: Intervention effects (Keep minus the alternative) on 40 previously unused images per task, averaged equally over AFHQ-Cat and CelebA. Brackets give 95% PSNR CIs. Region MSE evaluates missing pixels or ground-truth saturated coordinates in raw solver coordinates. Ground truth is used only for evaluation; Appendix C gives full intervals and dataset-specific effects.

Task	Contrast	ΔPSNR (dB) ↑	Δregion MSE ↓	ΔLPIPS ↓
Inpainting	Keep–Remove	$+0.788$ $[0.433, 1.188]$	-0.0102	-0.00534
	Keep–Reverse	$+4.557$ $[3.834, 5.287]$	-0.1057	-0.02252
HDR	Keep–Remove	$+0.886$ $[0.573, 1.194]$	-0.0054	-0.01605
	Keep–Reverse	$+2.620$ $[1.994, 3.293]$	-0.0132	-0.05291



(a) Reconstruction with published methods



(b) Keeping, removing, or reversing the measurement-indifferent component

Figure 2: Reconstruction quality and the contribution of endpoint geometry. (a) HDR on AFHQ-Cat #311 and untuned $4\times$ super-resolution on CelebA #933, comparing FLOWER, D-Flow, and VEG; HDR uses the nonlinear adaptations. (b) Inpainting (AFHQ-Cat) and HDR (CelebA): Keep retains the accepted VEG update, while Remove and Reverse alter only its measurement-indifferent component. Each full image has an identical-region zoom to its right; labels give full-image PSNR (dB) / LPIPS. Appendix Figures 5 and 6 give additional intervention and reconstruction examples.

Correction depth and scope. Additional correction depth provides a separate quality–compute tradeoff. With $K = 10$ selected on separate JiT development images, repeated VEG improves the five-task evaluation mean PSNR by 0.84 dB over one-correction VEG, at $5.04\times$ its runtime (Appendix E.4). In the decoded latent SiT depth study (Ma et al., 2024), deeper local refinement benefits both direction families, while their PSNR difference at $K = 10$ remains unresolved ($+0.13$ dB, 95% CI $[-0.07, +0.36]$; Appendix F.2). Appendix F.1 gives the pixel-space depth sweeps.

7 Conclusion

A pretrained flow's clean prediction is more than a point estimate: its local variation contains information about how plausible clean endpoints can change. VEG connects this structure to the measurement-induced change in the conditional endpoint mean and turns it into a one-VJP correction of the predicted clean image. Matched controls on pixel-space flows show PSNR gains beyond deeper ordinary refinement, while inpainting and HDR interventions show that the flow-derived direction selects useful image changes that the measurement loss alone cannot distinguish. The PSNR effect also transfers to an independently trained ImageNet predictor, while the latent-flow study identifies a setting in which the geometry-specific benefit is weaker. These results show that the local structure of a pretrained flow prediction can provide a useful conditioning signal for inverse reconstruction.

Reproducibility Statement

The main text specifies the proposed method and evaluation protocol, while the appendix provides implementation details, solver configurations, baseline settings, paired evaluation procedures, and additional ablations.

Ethics Statement

This work studies general-purpose computational imaging algorithms using established public image datasets and pretrained models. It does not involve human-subject experiments or the collection of sensitive personal data. Inverse-problem reconstructions can be visually plausible while incorrect, especially under ambiguous measurements; applications in high-stakes settings therefore require domain-specific validation and uncertainty assessment.

AI Use Statement

Generative AI tools were used to assist with language editing, literature search, code implementation, and refining the presentation of mathematical derivations and figures. All core conceptual ideas, theoretical frameworks, mathematical claims, experimental designs, analyses, and conclusions were developed and independently verified by the authors. The authors take full responsibility for the entire content of the manuscript.

References

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*, 2024.

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*, September 2024.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *Conference on Computer Vision and Pattern Recognition*, 2020.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Conference on Computer Vision and Pattern Recognition*, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

Hongkun Dou, Zeyu Li, Jinyang Du, Lijun Yang, Wen Yao, and Yue Deng. Hybrid regularization improves diffusion-based inverse problem solving. In *International Conference on Learning Representations*, 2025.

Nial Friel and Anthony N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(3):589–607, 2008.

Liav Hen, Tom Tirer, Raja Giryes, and Shady Abu-Hussein. Jacobian-aware posterior sampling for inverse problems. *Transactions on Machine Learning Research*, June 2026.

Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. In *International Conference on Computer Vision*, 2025a.

Sunwoo Kim, Minkyu Kim, and Dongmin Park. Test-time alignment of diffusion models without reward over-optimization. In *International Conference on Learning Representations*, 2025b.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Conference on Computer Vision and Pattern Recognition*, 2026.

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

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

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. SiT: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In *European Conference on Computer Vision*, 2024.

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*, 2025.

Xinyu Peng, Ziyang Zheng, Wenrui Dai, Nuoqian Xiao, Chenglin Li, Junni Zou, and Hongkai Xiong. Improving diffusion models for inverse problems using optimal posterior covariance. In *International Conference on Machine Learning*, 2024.

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

Severi Rissanen, Markus Heinonen, and Arno Solin. Free hunch: Denoiser covariance estimation for diffusion models without extra costs. In *International Conference on Learning Representations*, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Conference on Computer Vision and Pattern Recognition*, 2022.

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 *Conference on Computer Vision and Pattern Recognition*, 2024.

Bowen Song, Soo Min Kwon, Zecheng Zhang, Xinyu Hu, Qing Qu, and Liyue Shen. Solving inverse problems with latent diffusion models via hard data consistency. In *International Conference on Learning Representations*, 2024.

Stability AI. sd-vae-ft-mse: Improved autoencoders. Hugging Face model card, 2022. URL `https://huggingface.co/stabilityai/sd-vae-ft-mse`.

Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In *Conference on Computer Vision and Pattern Recognition*, 2021.

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.

Jiarui Xing, Song Wang, and Jian Wang. Uncertainty in a single pass: A closed-form identity for one-step flow matching. *arXiv preprint arXiv:2605.00941*, 2026. URL `https://arxiv.org/abs/2605.00941`. Version 5; concurrent preprint, earlier circulated as "Divergence is Uncertainty".

Jian Xu, Delu Zeng, John Paisley, and Qibin Zhao. What do flow-based inverse solvers approximate? a posterior-transport view. *arXiv preprint arXiv:2606.24516*, 2026. URL `https://arxiv.org/abs/2606.24516`. Version 2; concurrent preprint.

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *Conference on Computer Vision and Pattern Recognition*, 2025.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition*, 2018.

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

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

APPENDIX

Appendix A gives mathematical details; Appendix B gives solver implementation and component analyses; Appendix C gives measurement-indifferent interventions; Appendix D gives evaluation protocols, supplementary metrics, and the scale extension; Appendix E gives the independent JiT evaluation; and Appendix F gives correction-depth and scope studies.

A MATHEMATICAL DETAILS

This section proves the ideal-model identities used to motivate VEG and checks when its proposed direction decreases the local objective.

A.1 BAYESIAN INTERPRETATION OF THE TEMPERED PATH

Under the observation model in (4), the clean endpoint X_1 takes the role of the unknown image $x^\star$, and the measurement is generated by

$$Y = h(X_1) + \eta, \qquad \eta \sim \mathcal{N}(0, \Gamma).$$

The observation noise is independent of (X_0, X_1), so Y and the intermediate flow state X_t are conditionally independent given X_1. Bayes' rule therefore gives

$$p(x_1 \mid x_t, y) \propto p(x_1 \mid x_t)\, p(y \mid x_1) = q_t(x_1 \mid x_t)\, p(y \mid x_1).$$

For the Gaussian observation model,

$$p(y \mid x_1) = \frac{1}{(2\pi)^{m/2}|\Gamma|^{1/2}} \exp\left[-\frac{1}{2}\,\|h(x_1) - y\|^2_{\Gamma^{-1}}\right] \propto \exp[-\ell_y(x_1)].$$

Thus measurement conditioning reweights the conditional endpoint law according to how well the clean endpoints match the observation. Raising the likelihood to a power $\beta \geq 0$ gives the tempered path in (9):

$$\pi_{t,\beta}(x_1) = \frac{q_t(x_1 \mid x_t) \exp[-\beta \ell_y(x_1)]}{Z_t(\beta)}.$$

At $\beta = 0$, this is the original conditional endpoint law $q_t(\cdot \mid x_t)$; at $\beta = 1$, it is the Bayesian measurement-conditioned distribution $p(\cdot \mid x_t, y)$. Varying β from 0 to 1 therefore introduces the measurement likelihood continuously. Under this law, $m_t(\beta) = \mathbb{E}_{\pi_{t,\beta}}[X_1]$ is the conditional endpoint mean. The derivative at $\beta = 0$ describes the initial change in the mean clean image as measurement information is introduced.

A.2 PROOF OF PROPOSITION 1

Define the normalizer

$$Z_t(\beta) = \int q_t(x_1 \mid x_t) e^{-\beta \ell_y(x_1)}\, \mathrm{d}x_1, \tag{19}$$

and the unnormalized first moment

$$N_t(\beta) = \int x_1 q_t(x_1 \mid x_t) e^{-\beta \ell_y(x_1)}\, \mathrm{d}x_1, \qquad m_t(\beta) = \frac{N_t(\beta)}{Z_t(\beta)}.$$

Under the moment and differentiation assumptions of the proposition,

$$Z_t'(\beta) = -\int \ell_y(x_1) q_t(x_1 \mid x_t) e^{-\beta \ell_y(x_1)}\, \mathrm{d}x_1,$$

$$N_t'(\beta) = -\int x_1 \ell_y(x_1) q_t(x_1 \mid x_t) e^{-\beta \ell_y(x_1)}\, \mathrm{d}x_1.$$

The quotient rule separates the weighted-numerator term from the normalizer term:

$$\begin{aligned} m_t'(\beta) &= \frac{N_t'(\beta)}{Z_t(\beta)} - \frac{N_t(\beta) Z_t'(\beta)}{Z_t(\beta)^2} \\ &= -\mathbb{E}_{\pi_{t,\beta}}[X_1 \ell_y(X_1)] + \mathbb{E}_{\pi_{t,\beta}}[X_1]\mathbb{E}_{\pi_{t,\beta}}[\ell_y(X_1)] \qquad (20) \\ &= -\operatorname{Cov}_{\pi_{t,\beta}}(X_1, \ell_y(X_1)). \qquad (21) \end{aligned}$$

Since $\pi_{t,0} = q_t$, evaluating this identity at $\beta = 0$ yields

$$m_t'(0) = -\operatorname{Cov}_{q_t}(X_1, \ell_y(X_1)),$$

which proves Proposition 1.

A.3 Local covariance approximation and remainder

Let $\delta = X_1 - \bar{c}_t$, where $\bar{c}_t = \mathbb{E}_{q_t}[X_1]$, and let $\bar{g}_t = \nabla \ell_y(\bar{c}_t)$. From (11),

$$\ell_y(X_1) = \ell_y(\bar{c}_t) + \bar{g}_t^\top \delta + r_t(X_1).$$

Using the linearity of covariance, $\mathbb{E}_{q_t}[\delta] = 0$, and $C_t = \mathbb{E}_{q_t}[\delta\delta^\top]$, we obtain

$$\operatorname{Cov}_{q_t}(X_1, \ell_y(X_1)) = C_t \bar{g}_t + \mathbb{E}_{q_t}[\delta\, r_t(X_1)].$$

Together with Proposition 1, this gives

$$m_t'(0) = -C_t \bar{g}_t - \mathbb{E}_{q_t}[\delta\, r_t(X_1)].$$

Thus, (13) drops exactly the final term, and its accuracy depends on how the Taylor remainder correlates with endpoint deviations from their conditional mean.

Remark 1 (Linear measurements). For $h(x_1) = Hx_1$, let $Q = H^\top \Gamma^{-1} H$. Since the measurement loss is quadratic, its Taylor remainder is exactly

$$r_t(X_1) = \frac{1}{2}\delta^\top Q \delta,$$
$$\mathbb{E}_{q_t}[\delta\, r_t(X_1)] = \frac{1}{2}\mathbb{E}_{q_t}[\delta(\delta^\top Q \delta)].$$

The omitted term therefore depends on third central moments of the conditional endpoint distribution. It vanishes when this distribution is centrally symmetric, including for a Gaussian conditional distribution, but linearity of the forward operator alone does not imply such symmetry. Hence, (13) need not be exact even for linear measurements.

A.4 Proof of Proposition 2

For the linear probability path in (1),

$$q_t(x_1 \mid x_t) \propto p_{\mathrm{data}}(x_1) \exp\left(\frac{t}{(1-t)^2} x_t^\top x_1 - \frac{t^2}{2(1-t)^2} \|x_1\|^2 \right).$$

Define

$$\theta_t := \frac{t}{(1-t)^2} x_t, \qquad h_t(x_1) := p_{\mathrm{data}}(x_1) \exp\left(-\frac{t^2}{2(1-t)^2} \|x_1\|^2 \right).$$

Then the conditional endpoint distribution can be written in exponential-family form,

$$q_t(x_1 \mid x_t) = h_t(x_1) \exp\big(\theta_t^\top x_1 - A_t(\theta_t)\big),$$

where the log-partition function is

$$A_t(\theta) := \log \int h_t(x_1) \exp(\theta^\top x_1)\, \mathrm{d}x_1.$$

Standard exponential-family identities therefore give

$$D_t^\star(x_t) = \mathbb{E}[X_1 \mid X_t = x_t] = \nabla A_t(\theta_t), \qquad C_t = \nabla^2 A_t(\theta_t).$$

Since $\theta_t = t x_t/(1-t)^2$, the chain rule yields

$$J_{D_t^\star}(x_t) = \frac{t}{(1-t)^2} \nabla^2 A_t(\theta_t) = \frac{t}{(1-t)^2} C_t.$$

Since C_t is symmetric, so is $J_{D_t^\star}(x_t)$, and therefore

$$C_t \bar{g}_t = \frac{(1-t)^2}{t} J_{D_t^\star}(x_t)^\top \bar{g}_t.$$

This proves Proposition 2.

A.5 Neural nonideality, descent, and symmetry

For the ideal linear-path predictor, $J_{D_t^\star}$ is proportional to the conditional covariance and is therefore symmetric and positive semidefinite. A neural predictor only approximates this map, so its Jacobian need not preserve the ideal scale or symmetry. Measurement-driven corrections can also move the encountered states away from those seen during training. Thus, the exact covariance interpretation need not hold for the neural Jacobian.

The descent test ensures that an eligible VEG direction decreases the local objective to first order. For nonzero g_t and v_t satisfying $g_t^\top v_t > 0$,

$$g_t^\top d_t^{\mathrm{VEG}} = -\|g_t\| \frac{g_t^\top v_t}{\|v_t\| + \epsilon} < 0.$$

If $J_{D_t}(x_t) = \lambda_t I$ with $\lambda_t > 0$, then $v_t = \lambda_t g_t$, so d_t^{VEG} is a positive rescaling of the ordinary descent direction $-g_t$; when the numerical stabilizer is omitted, $d_t^{\mathrm{VEG}} = -g_t$ exactly.

To examine the effect of neural asymmetry, write $J_D := J_{D_t}(x_t) = S_D + A_D$, where $S_D = (J_D + J_D^\top)/2$ and $A_D = (J_D - J_D^\top)/2$. Then, we have $J_D g = S_D g + A_D g$, $J_D^\top g = S_D g - A_D g$. Since $g^\top A_D g = 0$, we also obtain $g^\top J_D g = g^\top J_D^\top g = g^\top S_D g$. Thus, only the symmetric part determines the sign of the descent test, while the antisymmetric part can still change the proposed direction. In particular, $J_D^\top g - S_D g = -A_D g$ and $J_D g - J_D^\top g = 2A_D g$ hold. For the ideal predictor, $A_D = 0$; for a neural predictor, these directions can differ while sharing the same descent test. Appendix B.4 compares them empirically.

Finally, $J_{D_t}(x_t)^\top g_t$ is naturally expressed in the coordinates of the flow state. In the primary image-space setting, the flow state and predicted clean image share the same coordinates, allowing this direction to be applied directly to the predicted image.

B Solver Implementation and Component Analyses

This appendix gives the shared solver and controls for correction depth, direction, and continuation used in Sections 6.2 and 6.3. Appendix C gives the inpainting and HDR interventions. Paired runs of Algorithm 1 share the same Gaussian noise draws.

B.1 Objective, step acceptance, and noise mixing

For the primary image-space experiments, the local objective in (17) takes the form

$$U_n(c) = \frac{1}{2\kappa_n}\|c - c_n\|^2 + \frac{B_n}{2(\sigma_y^2 + \delta_{\mathrm{cov}})}\|h(c) - y\|^2, \qquad \lambda_n = \kappa_n^{-1}. \tag{22}$$

We use $\kappa_n = \max\{(1-t_n)^2/[t_n^2 + (1-t_n)^2], 10^{-6}\}$, with $t_n = n/N$, $N = 100$, and total measurement weight $B = 100$ distributed uniformly across intervals, so $B_n = B/N = 1$. Measurements are generated with noise standard deviation $\sigma_y = 0.05$. The solver adds the small numerical floor $\delta_{\mathrm{cov}} = 10^{-6}$ to the measurement variance when evaluating the loss; this changes only the numerical weighting used by the solver, not the generated measurements. VEG and the ordinary-gradient controls use the same objective, penalty schedule, observations, and paired Gaussian noise draws.

The safeguarded line search starts from $\alpha = 1$, halves the step after a rejection, and accepts the first candidate satisfying

$$U_n(c_n + \alpha d_n) \le U_n(c_n) + 10^{-4}\alpha g_n^\top d_n. \tag{23}$$

The search stops when the step falls below 10^{-8} or after 30 backtracks. If the VEG direction does not yield an acceptable step, the same search is repeated with ordinary descent $-g_n$; if that also fails, no correction is applied. The resulting clean prediction is returned to the next flow state through the noise-mixing update in (18).

B.2 Ordinary-gradient comparison: procedure and paired effects

Ordinary-gradient control. This comparison tests whether the PSNR gain comes from the direction itself or can be obtained simply by repeating ordinary correction. We replace VEG's normalized

Algorithm 1 Vector–Jacobian Endpoint Geometry (VEG)

Require: State x_0, clean-prediction map D_t, objectives U_n, intervals N, stabilizer ϵ, fixed noise stream

for $n = 0, \dots, N-1$ **do**
  $t_n \leftarrow n/N$; $c_n \leftarrow D_{t_n}(x_n)$; $g_n \leftarrow \nabla U_n(c_n)$
  if $t_n > 0$ **then**
    $v_n \leftarrow J_{D_{t_n}}(x_n)^\top g_n$ ▷ one positive-time VJP
  end if
  if $t_n > 0$, g_n and v_n finite, $\|g_n\| > 0$, $\|v_n\| > 0$, and $g_n^\top v_n > 0$ **then**
    $d_n \leftarrow -\|g_n\| v_n/(\|v_n\| + \epsilon)$
    $\alpha_n \leftarrow$ SAFEGUARDEDLINESEARCH(U_n, c_n, g_n, d_n)
    if the search returns no step **then**
      $d_n \leftarrow -g_n$
      $\alpha_n \leftarrow$ SAFEGUARDEDLINESEARCH(U_n, c_n, g_n, d_n)
    end if
  else
    $d_n \leftarrow -g_n$ ▷ $t = 0$ or VJP direction not used
    $\alpha_n \leftarrow$ SAFEGUARDEDLINESEARCH(U_n, c_n, g_n, d_n)
  end if
  if the safeguarded search returns no step **then**
    $\alpha_n \leftarrow 0$
  end if
  $\widetilde{c}_n \leftarrow c_n + \alpha_n d_n$
  draw z_{n+1} from the fixed noise stream; $t_{n+1} \leftarrow (n+1)/N$
  $x_{n+1} \leftarrow t_{n+1}\widetilde{c}_n + (1 - t_{n+1})z_{n+1}$
end for

VJP direction with the ordinary local-objective gradient, keeping the pretrained predictor, objective, line-search acceptance rule, noise-mixing step, and random draws unchanged. The control takes up to $K = 1, 2, 10$ corrections within each flow interval. K counts inner corrections, not flow intervals; all variants use the same 100-interval time grid. These comparisons isolate the direction and depth components of the solver.

Within an interval, the objective U_n remains centered at the original prediction c_n. Each additional ordinary correction recomputes the full gradient ∇U_n, including the quadratic penalty. The first line search starts at 1; later inner corrections carry forward the accepted step length. VEG accepts at most one correction per interval. The one-correction comparison therefore changes the direction component, while the two- and ten-correction comparisons also increase the control's optimization depth.

Table 5 reports the absolute depth-control results.

Relative to the one-correction ordinary-gradient control, VEG gains 1.069 dB in PSNR with image-clustered 95% CI $[0.828, 1.334]$ and a 91.7% win rate. Relative to the two-correction control, it gains 1.067 dB with CI $[0.896, 1.250]$ and an 88.3% win rate. It also exceeds the ten-correction control by 0.576 dB, CI $[0.405, 0.749]$, while using $0.642\times$ its runtime. The ten-correction control has lower measurement NRMSE, while its LPIPS difference from VEG is unresolved ($\Delta\mathrm{LPIPS}_{\mathrm{VEG}-K10} = +0.00012$, CI $[-0.00550, +0.00542]$; VEG has lower LPIPS on 74/120 cases and 37/60 images after averaging over tasks). Thus the PSNR and runtime improvement cannot be explained by giving ordinary correction more steps.

The longer control also spends more time evaluating the local objective. Appendix B.6 gives operation counts and measured runtime, separating the additional flow VJP from repeated measurement-gradient computation.

Across the 60 untuned linear cases, VEG exceeds the two-correction ordinary control by 2.611 dB (image-clustered 95% CI $[+2.251, +3.019]$; 78.3% of cases have higher PSNR). These linear-transfer cases are separate from the nonlinear summary above; Appendix D.2 reports measurement fit and comparisons with published methods.

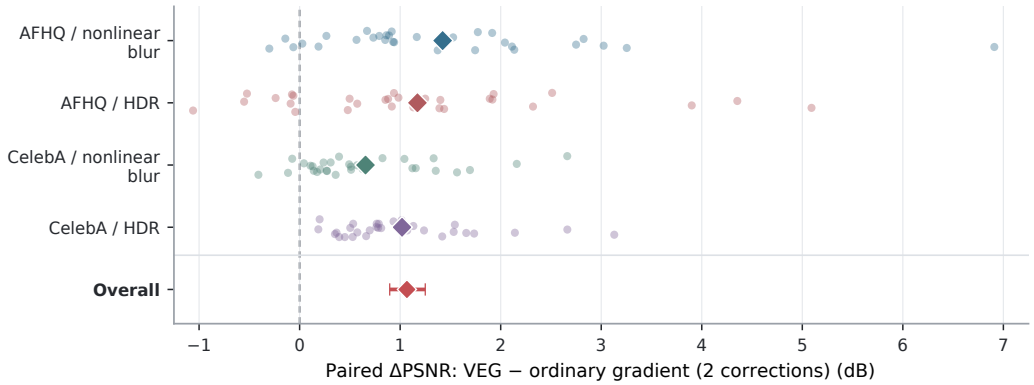


Figure 3: Paired PSNR gains from the flow-derived direction relative to the ordinary-gradient control with up to two corrections per interval on held-out nonlinear tasks. Each point represents one of 30 cases in a dataset–task group; diamonds show group means. The Overall whisker is the image-clustered 95% CI, obtained by resampling image identities within each dataset while keeping their nonlinear-blur and HDR outcomes paired. All 120 cases, including negative gains, are shown. The equal-group mean is $+1.0672$ dB and the win rate is 88.3%.

Table 5: Matched direction/depth controls on 120 held-out nonlinear cases. (a) Equal-group averages with maximum corrections per interval. (b) Ordinary-control breakdown, with GH200 seconds per case; VEG quality and runtime appear in Table 1, and dataset-specific measurement NRMSE in Table 13. (c) Paired VEG-minus-two-correction PSNR gains. All controls share the predictor, objective, acceptance rule, noise mixing, and random draws.

(a) Aggregate nonlinear comparison

Variant	Max. corrections per interval	PSNR ↑	LPIPS ↓	NRMSE ↓	sec./case ↓
Ordinary gradient	1	24.9695	0.18942	0.08717	13.14
Ordinary gradient	2	24.9714	0.18066	0.08986	14.80
VEG	1	26.0387	0.15901	0.08752	17.11
Ordinary gradient	10	25.4625	0.15889	0.08243	26.64

(b) Ordinary controls by dataset and task

Control	Dataset / task	PSNR	LPIPS	NRMSE	Time (s)
Ordinary (2)	AFHQ-Cat / nonlinear blur	24.831	0.3538	0.1139	21.44
Ordinary (2)	AFHQ-Cat / HDR	25.280	0.1500	0.0830	6.11
Ordinary (2)	CelebA / nonlinear blur	27.078	0.1274	0.1006	24.88
Ordinary (2)	CelebA / HDR	22.696	0.0914	0.0619	6.77
Ordinary (10)	AFHQ-Cat / nonlinear blur	26.115	0.3003	0.1059	42.52
Ordinary (10)	AFHQ-Cat / HDR	25.471	0.1368	0.0700	7.26
Ordinary (10)	CelebA / nonlinear blur	27.681	0.1109	0.0982	48.80
Ordinary (10)	CelebA / HDR	22.583	0.0876	0.0557	8.00

(c) VEG minus two ordinary corrections

Dataset / task	ΔPSNR (dB)	Win rate
AFHQ-Cat / nonlinear blur	+1.4227	90.0%
AFHQ-Cat / HDR	+1.1720	73.3%
CelebA / nonlinear blur	+0.6555	90.0%
CelebA / HDR	+1.0188	100.0%

B.3 Stronger ordinary refinement with line-search restarts

This control tests whether deeper ordinary correction is held back by a line search that keeps starting from a previously accepted small step. Restarting lets each new correction explore a larger step again. The comparison reuses 10 AFHQ-Cat and 10 CelebA development images, each under nonlinear blur and HDR saturation.

Table 6: Dataset-balanced paired restart effects on development cases. PSNR entries include 95% CIs from a dataset-stratified bootstrap of image identities, keeping both nonlinear tasks together. Differences are first variant minus second; negative LPIPS and runtime differences favor the first.

Contrast	K	ΔPSNR (dB) [95% CI]	ΔLPIPS	Δsec./case
Restart–carry	2	+0.037 [-0.034,+0.103]	-0.00272	+3.86
Restart–carry	5	-0.447 [-0.643,-0.286]	-0.00805	+15.33
Restart–carry	10	-0.629 [-0.759,-0.496]	-0.00605	+34.71
VEG–restart	2	+0.923 [+0.631,+1.170]	-0.01919	-1.44
VEG–restart	5	+1.086 [+0.884,+1.285]	+0.00177	-17.74
VEG–restart	10	+1.089 [+0.777,+1.361]	+0.00654	-44.53

In the carry-forward control, the first search in each interval starts at 1 and later searches start at the previously accepted length. In the restarted control, every inner search starts at 1. At matched K, this starting-step policy is the only intended change: the predictor, local objective, isotropic quadratic penalty, acceptance rule, noise-mixing step, and random draws are fixed. Both recompute the full ordinary gradient after each accepted inner correction. The $K = 1$ policies coincide. VEG uses its unchanged single-correction algorithm, and no setting is retuned.

Table 6 summarizes paired effects across the four equally weighted dataset–task groups. The PSNR effect of restarting is unresolved at $K = 2$; at $K = 5$ and $K = 10$, restarting trades lower PSNR for improved LPIPS and higher runtime. VEG retains higher PSNR than every restarted control and takes less time. The $K = 5$ and $K = 10$ restarted controls have numerically lower mean LPIPS than VEG, but both paired differences are unresolved: VEG minus restart is $+0.00177$ (CI $[-0.00696, +0.01074]$) and $+0.00654$ (CI $[-0.00035, +0.01372]$), respectively. Thus these comparisons establish a PSNR–compute advantage without establishing a perceptual advantage in either direction.

B.4 Comparing directions and update scales

We compare whether the particular Jacobian product and its scaling affect reconstruction. For brevity, write $J_D := J_{D_t}(x_t)$ and $g := g_t$ at one interval. We consider the forward product $a_{\mathrm{JVP}} = J_D g$, the reverse product $a_{\mathrm{VJP}} = J_D^\top g$ used by VEG, and their symmetric average $a_{\mathrm{SYM}} = (J_D g + J_D^\top g)/2$. The ordinary correction uses $-g$. Each Jacobian-based vector is converted to a correction direction by norm matching,

$$\mathrm{NM}(a) = -\|g\| \frac{a}{\max(\|a\|, \epsilon)}. \tag{24}$$

The primary VEG implementation instead uses the additive stabilizer $\|a\|+\epsilon$ in (16). Both conventions preserve the proposed direction; when $\|a\| \gg \epsilon$, their scales are effectively identical. We use $\epsilon = 10^{-12}$ throughout these comparisons.

To test whether performance can be explained only by the angle between the VEG correction and ordinary descent, we also construct a random direction with the same norm and angle. Let $u = g/\|g\|$ and $a = J_D^\top g/\left\|J_D^\top g\right\|$, and set $\rho = u^\top a$. For a Gaussian vector r, remove its component along u and normalize the remainder as $\widehat{r}_\perp = (r - (u^\top r)u)/\left\|r - (u^\top r)u\right\|$. We then define

$$d_{\mathrm{rand}} = -\|g\| \left(\rho u + \sqrt{\max(1-\rho^2, 0)}\, \widehat{r}_\perp\right), \tag{25}$$

which has the same norm as ordinary descent and the same angle to it as the VEG direction.

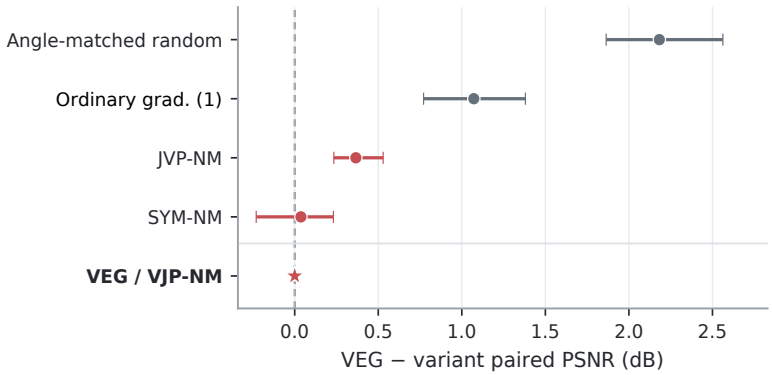


Figure 4: Direction ablation on 40 development cases. Positive paired PSNR differences favor VEG's norm-matched VJP. JVP-NM uses the forward product, SYM-NM averages the forward and reverse products, and the angle-matched random control preserves the update norm and angle to ordinary descent. Whiskers show image-clustered 95% CIs.

Table 7: Update scale and step-acceptance comparisons on 40 development cases. Raw VJP uses the unnormalized product; theory-scaled VJP uses the ideal-model scale. Image NMSE is $\|\hat{x} - x^\star\|^2 / \|x^\star\|^2$ in raw solver coordinates, averaged equally over the four dataset–task groups. The three scaling rules give similar PSNR, while removing backtracking causes catastrophic divergence.

Variant	PSNR ↑	LPIPS ↓	Image NMSE ↓	Diverged?
Raw VJP	25.554	0.1592	0.0444	No
Theory-scaled VJP	25.483	0.1592	0.0454	No
Norm-matched VJP (VEG)	25.384	0.1633	0.0466	No
Norm-matched VJP, no backtracking	-34.744	1.1603	46833.9073	Yes

All variants share the descent test, observations, noise draws, line search, and rule for resuming generation. Figure 4 summarizes the direction comparison. On 40 development cases, VEG's norm-matched VJP improves PSNR by 1.072 dB over one ordinary-gradient correction and by 0.366 dB over the norm-matched JVP. Its difference from the symmetric average is $+0.037$ dB (95% CI $[-0.230, +0.232]$), which is unresolved. Against the angle-matched random control, the gain remains between 2.054 and 2.182 dB across five random draws. Thus, matching only the update norm and its angle to ordinary descent is insufficient to reproduce the gain.

Across the evaluated development intervals, the forward and reverse products are closely aligned, with an overall median cosine similarity of 0.9583.

The three VJP scale choices give similar PSNR with backtracking. Without backtracking, the norm-matched update diverges catastrophically. We therefore use norm matching to control the update scale and retain backtracking as a stability safeguard.

B.5 Additional correction strategies and continuation

Normalized noisy-state guidance. This comparison uses the same pretrained model, clean images, observations, initial Gaussian noise, and 100-interval grid on 40 development cases. The normalized noisy-state guidance variant implements our nonlinear extension of the velocity-correction update in Xu et al. (2026). At $t_n = n/100$, it computes the clean prediction $c_n = D_{t_n}(x_n)$ and differentiates the unweighted squared measurement residual through this prediction:

$$w_n = \nabla_{x_n}\left[\frac{1}{2}\left\|h(D_{t_n}(x_n)) - y\right\|^2\right], \qquad x_{n+1} = x_n + \Delta t\, v_\theta(x_n, t_n) - \gamma \frac{w_n}{\|w_n\| + 10^{-12}},$$

where $\Delta t = 1/100$, $\gamma = 1$, and $n = 0, \ldots, 99$. Each update is applied directly, without a line search or local quadratic penalty, and continues by deterministic Euler integration without fresh-noise

Table 8: Correction direction and continuation on 40 development cases. All rows use one correction per interval and equally weight the four dataset–task groups. FLOWER denotes noise mixing; Euler uses the corrected velocity. Measurement NRMSE is $\|h(\hat{x}) - y\| \,/\, \|y\|$.

Correction	Continuation	PSNR	LPIPS	SSIM	Meas. NRMSE
Ordinary gradient	FLOWER	24.312	0.19358	0.7538	0.08717
VEG	FLOWER	25.384	0.16335	0.7892	0.08249
Ordinary gradient	Euler	14.300	0.36355	0.3529	0.43824
VEG	Euler	19.272	0.23149	0.5483	0.19696

mixing. The clean prediction is recomputed from the updated state at the next interval; the returned reconstruction is the final state x_{100}. Thus, the comparison evaluates complete correction strategies with different update scales, acceptance rules, and continuations.

Correcting the clean image with VEG exceeds this noisy-state strategy by 1.945 dB (95% CI $[0.903, 2.988]$). The separate FLOWER reference comparison gives a gain of 2.549 dB ($[2.058, 3.101]$); that reference uses local Langevin corrections followed by noise mixing. These results complement the direction and refinement-depth controls in Table 3.

Corrected-velocity Euler continuation. This 2×2 comparison crosses ordinary-gradient or VEG correction with the noise-mixing update used in FLOWER in (18) or deterministic Euler continuation. It uses the same 40 nonlinear development cases, pretrained predictor, 100-interval grid, objective, quadratic penalty, line search, initial noise, and one correction per interval, without retuning. Given the corrected clean prediction $\widetilde{c}_n$, the Euler alternative uses

$$v_n^{\mathrm{corr}} = \frac{\widetilde{c}_n - x_n}{1 - t_n}, \qquad x_{n+1} = x_n + (t_{n+1} - t_n) v_n^{\mathrm{corr}}.$$

This continuation draws no fresh noise and requires no additional pretrained ODE evaluation. Both correction rules retain their original acceptance procedure.

The dataset-balanced VEG-minus-ordinary PSNR effect is $+1.072$ dB with the noise-mixing update, 95% CI $[+0.767, +1.383]$, and $+4.972$ dB with Euler, CI $[+3.937, +6.024]$. However, switching VEG from the noise-mixing update to Euler reduces its absolute PSNR by 6.111 dB, CI $[-7.367, -4.890]$. Euler gives a larger relative direction gain but lower absolute reconstruction quality. These intervals resample identities within datasets, keeping both task outcomes together.

Runtime is omitted because concurrent GPU work may have affected these measurements.

B.6 When ordinary descent is used and what computation costs

On the held-out nonlinear evaluation, VEG uses ordinary descent in 181 of 12,000 intervals (1.51%), including the mandatory $t = 0$ correction.

B.7 Detailed Comparison with JAPS

JAPS backpropagates a vector through its diffusion clean-image estimator, giving an exact $J^\top v$ product for that neural map. It applies projection and diffusion-schedule scaling, updates the diffusion state, and aims at posterior sampling on nonlinear deblurring and HDR tasks (Hen et al., 2026). Table 10 compares these construction choices with VEG.

Table 9: Computation per 100-interval reconstruction. Local-method runtime uses the held-out nonlinear evaluation. Peak memory and operation counts for VEG and the one-correction ordinary-gradient control come from 40 development cases. D-Flow backpropagates through the full Euler flow; its count varies with L-BFGS line-search evaluations. Parentheses give the maximum corrections per interval for ordinary-gradient controls. Unmeasured memory is left blank.

Method	Clean-image corr./interval	Prediction forwards	Meas.-loss BP	Prediction VJPs	JVPs	Full-flow BP
Ordinary grad. (1)	≤ 1	100	100	0	0	No
Ordinary grad. (2)	≤ 2	100	≤ 200	0	0	No
VEG	≤ 1	100	100	99	0	No
Ordinary grad. (10)	≤ 10	100	≤ 1000	0	0	No
D-Flow	n/a	$2(S-1)$/eval.	–	–	0	Yes

Method	Measurement evaluations	sec./case	Peak GiB
Ordinary grad. (1)	variable (63,360†)	13.14	6.38†
Ordinary grad. (2)	variable	14.80	–
VEG	variable (63,335†)	17.11	6.38†
Ordinary grad. (10)	variable	26.64	–
D-Flow	1/evaluation	333.74	–

BP denotes a backward pass. Measurement-loss BP computes g_n; a VJP through the clean-prediction map computes $J_D(x_n)^\top g_n$. †Counts cover 40 development reconstructions; divide evaluation totals by 40 for a per-case mean. VEG uses 99 clean-prediction VJPs per reconstruction. S is the number of Euler steps in D-Flow.

Table 10: Construction-level comparison with JAPS, the closest published Jacobian-aware conditioning method. Both compute an exact transpose-Jacobian product for a neural map. They differentiate different predictors and differ in the statistical target, location of the correction, scaling and acceptance, and rule for continuing generation.

Item	JAPS (Hen et al., 2026)	VEG (ours)
Generative map	Pretrained diffusion model with DDIM sampling	Pretrained flow model
Differentiated map	Tweedie/DDIM clean estimator	Clean-prediction map D_t
Jacobian action	Denoiser $J^\top v$ of a detached proximal vector	Clean-prediction $J_D^\top g$
Correction space	Diffusion trajectory state	Predicted clean image
Scale/nonideality	u–v projection, diffusion schedule, released task scale	Direction-only norm match and safeguarded acceptance
Propagation/return	Conditional DDIM trajectory update	Unchanged noise-mixing step
Inference objective	Posterior sampling	Local reconstruction correction
Additional training	None; released pretrained diffusion checkpoint	None; frozen pretrained flow checkpoint
Per-step geometry	Denoiser VJP; nonlinear proximal solve uses operator JVP/VJP CG	One additional clean-prediction VJP per positive-time interval

C MEASUREMENT-INDIFFERENT INTERVENTIONS

C.1 SHARED PROTOCOL

With the intervention rules fixed, we evaluate Keep, Remove, and Reverse on 20 AFHQ-Cat and 20 CelebA images per task. All variants share the pretrained predictor, 100 flow intervals, observations with $\sigma_y = 0.05$, and paired noise streams. At each variant's current state, the usual VEG

proposal is accepted before its selected component is retained, removed, or reversed. The resulting candidate enters the unchanged noise-mixing continuation without a second line search, clipping, or rescaling. Same-state equalities concern these candidates; separately continued trajectories may diverge. Regional MSE uses raw solver coordinates before display clipping. Confidence intervals resample identities within datasets, weight the datasets equally, and use 20,000 replicates. Secondary and subgroup intervals are nominal 95% intervals.

The evaluation images are separate from the development images used to set these rules. No ground truth is used to form or choose the interventions. Table 4 summarizes the aggregate effects.

C.2 Inpainting construction

Inpainting leaves the observation unchanged when only missing pixels change. We use centered 80×80 and 40×40 masks at 256^2 and 128^2 resolution, respectively.

Let $P_O = P_O^\top = P_O^2$ select observed image coordinates and $P_M = I - P_O$ select the masked coordinates, so $P_O P_M = 0$. The forward operator satisfies $h(c) = P_O c$. At the current clean prediction c_n, the quadratic penalty is centered at c_n and the measurement gradient has no masked component: $P_M g_n = 0$. Ordinary descent therefore cannot directly update these coordinates within this local correction. The flow-derived direction can couple observed and unobserved coordinates.

With accepted displacement $\Delta c = \widetilde{c}_n - c_n$, set $\Delta c_O = P_O \Delta c$ and $\Delta c_M = P_M \Delta c$. Keep, Remove, and Reverse give the following candidate clean predictions:

$$\begin{aligned} c^{\mathrm{Keep}} &= c_n + \Delta c_O + \Delta c_M, \\ c^{\mathrm{Remove}} &= c_n + \Delta c_O, \\ c^{\mathrm{Reverse}} &= c_n + \Delta c_O - \Delta c_M. \end{aligned}$$

All three candidates share the same observed coordinates and measurement prediction. Keep and Reverse also have equal displacement norm because

$$\|\Delta c_O + \Delta c_M\|^2 = \|\Delta c_O\|^2 + \|\Delta c_M\|^2 = \|\Delta c_O - \Delta c_M\|^2 .$$

For the isotropic quadratic penalty in (17), this gives equal penalty and local-objective values. Since $P_M g_n = 0$, the Armijo linear term is also equal:

$$g_n^\top(\Delta c_O + \Delta c_M) = g_n^\top \Delta c_O = g_n^\top(\Delta c_O - \Delta c_M).$$

These equalities concern candidates formed at the same state, whereas separately continued trajectories can have different objectives. Remove retains the observation but reduces the penalty whenever the removed component is nonzero.

For a final reconstruction $\widehat{c}$ and ground truth c_{gt}, masked-region MSE is

$$\mathrm{MSE}_M(\widehat{c}, c_{\mathrm{gt}}) = \frac{\|P_M(\widehat{c} - c_{\mathrm{gt}})\|^2}{\mathrm{tr}(P_M)}.$$

The projector includes all color channels, and regional MSE uses raw solver coordinates before display clipping.

C.3 HDR construction

For HDR, the observation stays fixed only while each changed value remains on its original clipping plateau. We therefore choose coordinates where the original prediction, the accepted VEG proposal, and its reversed change all stay on that same plateau. The HDR forward map is $h(c) = \mathrm{clip}(2c, -1, 1)$, applied to each color coordinate, followed by Gaussian measurement noise. Let d be the accepted VEG displacement from prediction c. With fixed margin $\varepsilon_{\mathrm{sat}} = 10^{-4}$, the selected set is

$$S(c, d) = \left\{ i : \min(c_i, c_i + d_i, c_i - d_i) > \tfrac{1}{2} + \varepsilon_{\mathrm{sat}} \text{ or } \max(c_i, c_i + d_i, c_i - d_i) < -\tfrac{1}{2} - \varepsilon_{\mathrm{sat}} \right\}.$$

Writing P_S for this coordinate projector, the three candidates are

$$c^{\mathrm{Keep}} = c + d, \qquad c^{\mathrm{Remove}} = c + d - P_S d, \qquad c^{\mathrm{Reverse}} = c + d - 2P_S d.$$

Table 11: Final intervention evaluation on 20 previously unused images per dataset and task. Region MSE is evaluated on the known missing region for inpainting and on ground-truth saturated coordinates for HDR. Ground-truth image values are used only for evaluation, not to construct or select the interventions.

Task	Dataset	Variant	PSNR ↑	LPIPS ↓	Region MSE ↓
Inpainting	AFHQ-Cat	Keep	27.149	0.06273	0.08933
		Remove	26.575	0.07140	0.10303
		Reverse	22.303	0.09939	0.25881
Inpainting	CelebA	Keep	32.529	0.01544	0.02082
		Remove	31.527	0.01745	0.02753
		Reverse	28.262	0.02382	0.06272
HDR	AFHQ-Cat	Keep	27.129	0.10193	0.02908
		Remove	26.374	0.11947	0.03256
		Reverse	23.693	0.17200	0.04204
HDR	CelebA	Keep	23.590	0.07337	0.05177
		Remove	22.574	0.08793	0.05903
		Reverse	21.788	0.10911	0.06522

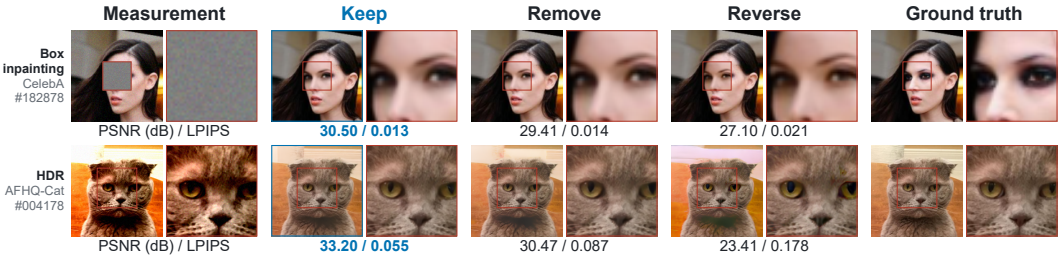


Figure 5: Additional intervention examples: CelebA inpainting and AFHQ-Cat HDR, complementing the other two dataset–task groups in Figure 2b. Columns show measurement, Keep, Remove, Reverse, and ground truth with identical zoom regions. Labels report full-image PSNR (dB) / standard LPIPS. Images retain the fixed intensity mapping without per-image contrast normalization.

The selector uses only c and d, without the ground truth or measurement values. All candidates coincide outside S, and all remain on the same clipping plateau inside it, giving identical forward predictions and measurement losses. Since $P_S \nabla \ell_y(c) = 0$, Keep and Reverse also share the Armijo term. Orthogonality of P_S and $I - P_S$ gives equal displacement norms and hence equal isotropic quadratic penalties and local-objective values. Remove has no larger quadratic penalty.

Across intervention states, selected coordinates occupy 34.19% of image coordinates on average and contain 2.11% of the accepted update's squared norm (median 1.42%). The selected displacement is exactly zero in 1.00% of states. Saturation-region MSE is evaluated on $T = \{i : |c_{\mathrm{gt},i}| \geq 1/2\}$ as $\sum_{i \in T} (\widehat{c}_i - c_{\mathrm{gt},i})^2 / |T|$. This ground-truth evaluation mask is distinct from the state-dependent intervention set S and is never supplied to the solver.

C.4 Evaluation results and additional examples

Tables 11 and 12 report absolute quality and paired effects on the previously unused images. Aggregate PSNR and region-MSE intervals exclude zero for both tasks and contrasts. The AFHQ-Cat inpainting Keep–Remove PSNR and region-MSE intervals include zero. For HDR, the Keep–Remove measurement-NRMSE effect is unresolved ($+0.00041$, 95% CI $[-0.00254, +0.00486]$).

Table 12: Paired effects of Keep minus Remove or Reverse. (a) Dataset-specific effects with nominal 95% confidence intervals and higher-PSNR counts. (b) Effects averaged equally over datasets, with dataset-stratified 95% intervals. Positive PSNR and negative region MSE or LPIPS favor Keep.

(a) Dataset-specific effects

Task	Dataset	Contrast	ΔPSNR (dB)	Δregion MSE	PSNR wins
Inpainting	AFHQ-Cat	Keep–Remove	$+0.575$ $[-0.120, +1.361]$	-0.0137 $[-0.0347, +0.0047]$	12/20
		Keep–Reverse	$+4.847$ $[+3.463, +6.250]$	-0.1695 $[-0.2233, -0.1154]$	19/20
Inpainting	CelebA	Keep–Remove	$+1.001$ $[+0.841, +1.173]$	-0.0067 $[-0.0090, -0.0049]$	20/20
		Keep–Reverse	$+4.266$ $[+3.839, +4.724]$	-0.0419 $[-0.0537, -0.0333]$	20/20
HDR	AFHQ-Cat	Keep–Remove	$+0.755$ $[+0.269, +1.213]$	-0.0035 $[-0.0056, -0.0014]$	17/20
		Keep–Reverse	$+3.437$ $[+2.426, +4.543]$	-0.0130 $[-0.0191, -0.0060]$	19/20
HDR	CelebA	Keep–Remove	$+1.016$ $[+0.641, +1.444]$	-0.0073 $[-0.0093, -0.0053]$	19/20
		Keep–Reverse	$+1.803$ $[+1.135, +2.586]$	-0.0134 $[-0.0179, -0.0094]$	19/20

(b) Equal-dataset aggregate effects

Task	Contrast	ΔPSNR (dB)	Δregion MSE	ΔLPIPS
Inpainting	Keep–Remove	+0.788 [+0.433,+1.188]	-0.0102 [-0.0207,-0.0009]	-0.00534 [-0.00796,-0.00306]
	Keep–Reverse	+4.557 [+3.834,+5.287]	-0.1057 [-0.1333,-0.0780]	-0.02252 [-0.02694,-0.01795]
HDR	Keep–Remove	+0.886 [+0.573,+1.194]	-0.0054 [-0.0068,-0.0039]	-0.01605 [-0.02131,-0.01174]
	Keep–Reverse	+2.620 [+1.994,+3.293]	-0.0132 [-0.0170,-0.0092]	-0.05291 [-0.06388,-0.04256]

D Evaluation Protocols and Supplementary Reconstruction Results

D.1 Evaluation and reporting protocols

The primary evaluation uses the optimal-transport-coupled flow checkpoints distributed with FLOWER (Pourya et al., 2026), for AFHQ-Cat at 256×256 and CelebA at 128×128, with weights held fixed. It uses 30 images per dataset and nonlinear task; linear transfer uses 10 per group. Measurements have Gaussian noise $\sigma_y = 0.05$. FLOWER-NL, D-Flow-NL, and the DAPS adaptation use the same pretrained flow and operators as VEG.

Nonlinear blur uses a pretrained KernelWizard network with its weights and kernel code held fixed, as in DAPS. Linear PnP-Flow uses its released task-family settings, including the super-resolution settings on the common $4\times$ operator; Flow Priors (Zhang et al., 2024) uses the PnP-Flow implementation. PSNR and SSIM measure image fidelity and structure; LPIPS measures perceptual discrepancy. Measurement NRMSE is $\|h(\hat{x}) - y\| \,/\, \|y\|$. Task-level measurement-fit summaries give equal weight to the two datasets. Published reconstruction methods are compared separately from the ordinary-gradient comparisons in Appendix B.2. In compact table labels, "Ordinary grad. (K)" means at most K corrections per interval.

VEG and its ordinary-gradient controls use 100 outer intervals, one trajectory, and the correction budgets specified in Appendix B.1. Nonlinear FLOWER averages five trajectories with 100 outer intervals and 200 inner Langevin steps for nonlinear blur or 100 for HDR. The DAPS adaptation uses 200 annealing states, five diffusion-ODE steps per state, and the same task-specific inner-step counts. Nonlinear D-Flow uses a midpoint transport on six time-grid points and 20 L-BFGS calls, each allowing up to 20 iterations with strong-Wolfe search and regularization weight 0.01.

For linear tasks, FLOWER uses 100 intervals and one trajectory; PnP-Flow uses 100 iterations with five samples per step, and Flow Priors uses 100 iterations. Linear D-Flow uses task-specific settings, with 7–20 L-BFGS calls and up to 20 iterations per call. Primary runtimes are measured with CUDA-synchronized algorithm timers on a GH200 at batch size one, excluding model loading and quality-metric evaluation. D-Flow timing covers source optimization and reconstruction after inverse-flow initialization.

D-Flow radial regularization. Our D-Flow adaptations follow the objective as written in the D-Flow paper (Ben-Hamu et al., 2024), including its radial-regularizer sign. Appendix E.1 specifies the exact objective used for JiT.

Metric conventions and uncertainty estimates. Confidence intervals for the primary evaluation and the separate inpainting/HDR experiments use 20,000 bootstrap replicates, resampling image identities within datasets and keeping their task outcomes together. Development LPIPS intervals use 10,000 replicates. Image-space AFHQ-Cat/CelebA comparisons, development controls, and the mechanism experiments on previously unused images use standard AlexNet (Krizhevsky et al., 2012) LPIPS v0.1. Full-image reconstructions and clean targets are supplied with `normalize=False`; the solver uses the nominal $[-1, 1]$ coordinate system, and outputs are not clipped or renormalized for this metric. The same raw-output convention applies to JiT and the matched-objective SiT depth sweep.

D.2 Measurement fit and paired reconstruction differences

Table 13 supplements the image-quality and runtime results in Tables 1 and 2 with measurement NRMSE. The task-level panels pool the two datasets, while the final panel compares VEG and DAPS separately for each nonlinear dataset–task group. Ordinary controls are analyzed in Appendix B.2. Table 14 gives the paired uncertainty for linear comparisons with published methods.

D.3 Larger evaluation

The larger evaluation addresses two distinct questions from Sections 6.2 and 6.3: whether the gain over ordinary correction persists with more images, and how VEG compares with published methods that spend more computation. We report absolute scores and paired confidence intervals for the full cohort and the additional images.

On 400 cases, the paired PSNR gains are $+0.929$ dB over the two-correction ordinary-gradient control (95% CI $[+0.838, +1.019]$, 89.75% wins) and $+0.474$ dB over the ten-correction control ($[+0.375, +0.567]$, 70.0% wins). A win means higher PSNR on a paired reconstruction case. The 70 additional images in each dataset (70+70 images) retain gains of $+0.870$ and $+0.430$ dB.

The paired LPIPS difference between VEG and the ten-correction ordinary control is -0.00010 (95% CI $[-0.00296, +0.00274]$; 241/400 lower-LPIPS cases and lower LPIPS on 108/200 images after averaging over tasks), with an interval that includes zero.

Relative to VEG, D-Flow-NL changes PSNR by $+1.851$ dB (95% CI $[+1.519, +2.159]$) at $19.2\times$ the mean runtime, the DAPS adaptation by -0.389 dB ($[-0.483, -0.294]$), and FLOWER-NL by -2.065 dB ($[-2.199, -1.930]$). D-Flow leads PSNR and LPIPS, while DAPS improves LPIPS and measurement fit relative to VEG. The methods thus offer different quality–cost tradeoffs. FLOWER-NL uses five stochastic reconstructions, 500 flow-model evaluations, and nested nonlinear likelihood evaluations per case, explaining why a per-step method can still have a high total cost.

D.4 Additional qualitative comparisons

Figure 6 gives ten additional paired examples covering the two datasets and five reconstruction tasks.

Table 13: Measurement NRMSE (↓) on nonlinear and linear reconstruction tasks. (a,b) Task means pool 30 images per dataset for nonlinear tasks and the same 10 images per dataset across linear tasks. (c) Dataset-specific nonlinear VEG–DAPS values. Quality and runtime appear in Tables 1 and 2; correction counts denote maxima per interval.

(a) Nonlinear task means

Method	Nonlinear blur	HDR
Ordinary grad. (1)	0.12147	0.05288
Ordinary grad. (2)	0.10725	0.07247
VEG	0.10949	0.06555
Ordinary grad. (10)	0.10204	0.06282
FLOWER-NL	0.16224	0.09266
D-Flow-NL	0.09824	0.08191
DAPS (adapt.)	0.09697	0.05575

(b) Linear task means

Method	Gaussian deblur	4× SR	Box inpainting
Ordinary grad. (2)	0.09538	0.22110	0.09034
VEG	0.09493	0.08302	0.08675
FLOWER	0.09477	0.15511	0.08387
PnP-Flow	0.09350	0.19732	0.10083
D-Flow	0.09739	0.01366	0.10370
Flow Priors	0.09693	0.09650	0.05026

(c) Dataset-specific nonlinear measurement fit

Dataset / task	VEG	DAPS adaptation
AFHQ / nonlinear blur	0.11683	0.09870
AFHQ / HDR	0.07176	0.06279
CelebA / nonlinear blur	0.10215	0.09524
CelebA / HDR	0.05933	0.04871

Table 14: Paired PSNR gains of VEG over published methods on 60 untuned linear cases. Image-clustered 95% intervals keep all three tasks from an image together; win rates count higher-PSNR cases. The ordinary-gradient comparison is in Appendix B.2.

Paired comparison	Mean ΔPSNR (dB)	95% CI	Win rate
VEG − FLOWER	+1.3359	[+0.9815, +1.7632]	61.7%
VEG − D-Flow	+2.0575	[+1.7123, +2.3859]	88.3%
VEG − Flow Priors	+2.9135	[+2.3320, +3.4521]	90.0%
VEG − PnP-Flow	+2.1589	[+1.7968, +2.5822]	80.0%

Table 15: Nonlinear evaluation on 100 images per dataset (400 AFHQ-Cat/CelebA cases under nonlinear blur and HDR). The upper rows compare one VEG correction with up to two or ten ordinary corrections per interval. The lower rows compare adaptations of published methods using the same pretrained flow. FLOWER-NL applies the nonlinear FLOWER procedure, the DAPS adaptation uses the published update, and D-Flow-NL optimizes the initial noise through the flow.

Method	PSNR ↑	LPIPS ↓	Meas. NRMSE ↓	sec./case ↓
Ordinary grad. (2)	24.839	0.1848	0.0883	**14.89**
VEG (Ours)	**25.768**	**0.1652**	0.0873	17.14
Ordinary grad. (10)	25.295	<u>0.1653</u>	**0.0813**	26.48
FLOWER-NL	23.704	0.2584	0.1268	755.50
D-Flow-NL	**27.619**	**0.0896**	0.0877	328.33
DAPS (adapt.)	25.379	<u>0.1259</u>	**0.0749**	332.41



Figure 6: Additional qualitative comparisons on AFHQ-Cat and CelebA. (a) Nonlinear blur and HDR saturation. (b) Gaussian deblurring, $4\times$ super-resolution, and box inpainting with untuned VEG. Columns pair each full image with a zoom immediately to its right; all methods share the input, ground truth, and crop region within each row. Values are full-image PSNR (dB) / LPIPS on the evaluated reconstructions. FLOWER and D-Flow use their nonlinear adaptations in (a). These selected examples illustrate reconstruction gains and task-dependent quality differences.

E INDEPENDENT JIT EVALUATION

E.1 PREDICTOR, OPERATORS, AND BASELINE IMPLEMENTATIONS

We use the released JiT-B/16 ImageNet-256^2 checkpoint (Li & He, 2026), ground-truth class conditioning, guidance scale 1.0, and FP32 arithmetic. Its direct clean-image prediction supplies the differentiated map. We retain 100 outer intervals, a total measurement weight of 100 allocated uniformly ($B_n = 1$), the local quadratic penalty, line-search acceptance, and noise-mixing continuation.

The checkpoint uses exponential-moving-average weights. Images use the RGB ADM center-crop procedure (Dhariwal & Nichol, 2021) at 256×256, followed by float32 conversion $2(u/255) - 1$ from uint8 pixels u, without flipping. The local solver evaluates $t_n = n/100$ and differentiates the direct clean-image map. Repeated correction holds the original interval state, Jacobian evaluation point, and quadratic-penalty center fixed, recomputes the full objective gradient after each correction, carries the accepted step length within an interval, and resets it to 1 at the next interval. Inner corrections do not allocate additional measurement weight.

The five tasks use Gaussian blur with standard deviation 3 and a 61×61 circular kernel, strided $4\times$ subsampling, a centered 80×80 missing box, nonlinear blur generated by the pretrained KernelWizard with fixed weights, and $h(c) = \mathrm{clip}(2c, -1, 1)$ for HDR. All physical observations have noise standard deviation 0.05. PSNR/SSIM use unclipped affine image coordinates, and standard LPIPS AlexNet v0.1 receives raw solver outputs with `normalize=False`. Clipping is used only to display these images.

FLOWER-JiT uses 100 intervals and five trajectories. Linear conditionals use conjugate gradients with a 50-iteration cap and a same-system FP64 spectral/diagonal solve when the accuracy check fails; the solution is returned in FP32. Nonlinear FLOWER uses a clean-conditional ULA adaptation with 100 inner steps, initial step 2×10^{-5}, minimum step ratio 0.01, and $\tau = 0.01$. Both nonlinear tasks use $50{,}000$ Langevin updates per image, but nonlinear blur repeatedly evaluates and differentiates the BKSE network, whereas HDR uses inexpensive clipping.

FlowDPS-JiT retains three endpoint updates with step size 30 and the common 100-interval JiT grid. Linear tasks use the adjoint residual; nonlinear tasks use a physical-residual extension. Where a velocity is required, the JiT interface uses $(D_t(x) - x)/\max(1 - t, 0.05)$. D-Flow-JiT optimizes the initial noise through a midpoint transport with six grid points: five intervals and ten model evaluations per transport. On linear tasks it uses inverse-adjoint initialization with `dopri5` and blend 0.1. On nonlinear blur and HDR it starts directly from the common Gaussian source, with no adjoint or inverse ODE. Both use regularization 0.01 and 20 L-BFGS calls with at most 20 iterations each and strong-Wolfe search. The nonlinear extension retains these settings on the same 40 evaluation images without parameter tuning. All methods share the pretrained weights, class labels, clean targets, operators, and observations; method-specific sampling and optimization rules remain distinct.

The nonlinear D-Flow-JiT objective is the sum of squared physical measurement residuals plus $0.01R(z)$ for the optimized source $z \in \mathbb{R}^d$. Our local port uses $R(z) = -\min(\|z\|_2^2/2, 10^6) + (d - 1)\log(\|z\|_2 + 10^{-5})$, following the sign printed in the published equation. This is the paper-based convention used in our primary D-Flow comparisons (Appendix D.1). L-BFGS uses learning rate 1 and at most 25 objective evaluations per call.

Runtime measurement. JiT runtime is averaged over four evaluation images and three repeats after a full untimed solve for each method and task. The common benchmark uses three A100 40GB GPUs on one node, with one repeat per GPU, batch size one, FP32, and one reconstruction at a time per GPU. Timing covers the complete solver call, including method-specific initialization, all trajectories and corrections, and final image formation; the reported FLOWER times include its implementation overhead. Model/input loading, quality metrics, disk I/O, and export are excluded.

Nonlinear D-Flow times marked † use a separate set of three A100 40GB GPUs with the same four images, three repeats, batch size, precision, warm-up, and timing boundaries. These measurements supplement the common benchmark; VEG/ordinary-refinement runtime ratios use only that benchmark.

Table 16: JiT reconstruction on 40 evaluation images per task. Bold and underlining rank complete quality means; NRMSE is measurement NRMSE. Time is A100 seconds per case, averaged over four fixed images and three repeats under Appendix E.1. D-Flow-JiT uses Gaussian initialization for nonlinear tasks; † marks its separate A100 timing run.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	NRMSE ↓	Time (s) ↓
Gaussian deblur					
VEG	24.555	0.6310	0.49762	**0.11293**	5.86
Repeated VEG	<u>24.874</u>	<u>0.6503</u>	<u>0.46337</u>	0.11354	29.53
FLOWER-JiT	**24.895**	**0.6532**	0.46730	0.11337	25.79
FlowDPS-JiT	23.355	0.5607	0.49776	0.12534	2.54
D-Flow-JiT	24.785	0.6480	**0.40066**	<u>0.11336</u>	392.21
4× **SR**					
VEG	**23.172**	**0.6489**	**0.38294**	<u>0.12090</u>	5.55
Repeated VEG	<u>23.097</u>	<u>0.6452</u>	<u>0.38793</u>	0.14431	27.95
FLOWER-JiT	22.176	0.5723	0.44901	0.22008	12.52
FlowDPS-JiT	15.749	0.2622	0.72361	0.38224	2.41
D-Flow-JiT	19.187	0.4392	0.54508	**0.11530**	394.09
Box inpainting					
VEG	24.930	0.8360	0.15461	<u>0.15848</u>	5.62
Repeated VEG	<u>25.084</u>	<u>0.8408</u>	<u>0.14550</u>	0.15939	28.35
FLOWER-JiT	**25.298**	**0.8414**	0.15101	**0.15787**	11.78
FlowDPS-JiT	21.420	0.6912	0.30355	0.21903	2.31
D-Flow-JiT	23.651	0.8010	**0.12539**	0.17033	378.48
Nonlinear blur					
VEG	23.821	0.6297	0.42118	0.14834	14.84
Repeated VEG	<u>25.411</u>	**0.7056**	<u>0.31619</u>	<u>0.13839</u>	75.19
FLOWER-JiT	**25.706**	<u>0.7052</u>	0.33372	**0.11244**	1001.62
FlowDPS-JiT	22.491	0.5548	0.45757	0.16550	7.91
D-Flow-JiT	25.302	0.6806	**0.29187**	0.14033	214.94†
HDR					
VEG	23.397	0.7720	0.19310	0.18240	5.73
Repeated VEG	<u>25.621</u>	<u>0.8226</u>	<u>0.13658</u>	<u>0.13233</u>	28.44
FLOWER-JiT	**26.283**	**0.8601**	**0.09055**	**0.06914**	45.33
FlowDPS-JiT	22.244	0.6976	0.27173	0.26200	2.36
D-Flow-JiT	24.218	0.7242	0.24958	0.18345	209.73†

E.2 PRESPECIFIED TWO-TASK TRANSFER

The primary transfer comparison tests whether the PSNR effect extends to JiT's independently trained pixel-space predictor, retaining the same type of clean-image correction.

Each of the 40 ImageNet validation images is evaluated under box inpainting and HDR with VEG and all five ordinary-gradient variants. These images are separate from the earlier evaluation set, and method settings are fixed before evaluation.

The primary comparison averages the two task differences within each image, then across all 40 images. VEG exceeds one ordinary gradient correction by $+1.475$ dB, with an image-level bootstrap 95% CI of $[+0.851, +2.127]$. The median difference is $+0.912$ dB: PSNR is higher on 28 images, equal on none, and lower on 12. Task-specific PSNR differences are $+0.420$ dB for inpainting (CI $[+0.124, +0.732]$; higher on 30/40 images) and $+2.529$ dB for HDR (CI $[+1.259, +3.889]$; higher on 27/40 images).

The LPIPS difference from one ordinary gradient correction, averaged equally over the two tasks, is unresolved: -0.00412, CI $[-0.01938, +0.01007]$. Inpainting LPIPS worsens by $+0.02108$ (CI $[+0.01374, +0.02907]$), while HDR's mean improves by -0.02933 (CI $[-0.06075, -0.00209]$). LPIPS improves on 19/40 HDR images, and the median effect is $+0.00522$. Task and secondary intervals are nominal, without multiple-comparison adjustment.

Table 17: VEG minus each ordinary control on 40 images: paired PSNR and LPIPS effects with nominal 95% image-level bootstrap intervals. W/T/L counts higher/equal/lower-PSNR images. Task-level control runtimes use the common A100 benchmark; the final block reports five-task effects, distinct from the two-task primary comparison.

Control	ΔPSNR [95% CI]	ΔLPIPS [95% CI]	W/T/L	Time (s)
Gaussian deblur				
Ordinary (1)	+0.142 [+0.096, +0.203]	-0.01335 [-0.01635, -0.01078]	40/0/0	2.95
Ordinary (10)	-0.211 [-0.279, -0.116]	+0.02297 [+0.01885, +0.02706]	2/0/38	5.14
Ordinary (10) restart	-0.225 [-0.297, -0.124]	+0.02488 [+0.02051, +0.02921]	2/0/38	8.48
$4\times$ **SR**				
Ordinary (1)	+0.656 [+0.256, +1.087]	+0.00504 [-0.02176, +0.03234]	23/0/17	2.67
Ordinary (10)	+1.429 [+0.886, +2.004]	-0.08792 [-0.11421, -0.06068]	32/0/8	4.17
Ordinary (10) restart	+1.429 [+0.886, +2.004]	-0.08792 [-0.11421, -0.06068]	32/0/8	6.55
Box inpainting				
Ordinary (1)	+0.420 [+0.127, +0.740]	+0.02108 [+0.01373, +0.02910]	30/0/10	2.69
Ordinary (10)	+0.437 [+0.161, +0.726]	-0.00223 [-0.00857, +0.00481]	33/0/7	4.25
Ordinary (10) restart	+0.437 [+0.161, +0.726]	-0.00223 [-0.00857, +0.00481]	33/0/7	6.89
Nonlinear blur				
Ordinary (1)	+1.057 [+0.744, +1.381]	-0.06408 [-0.08013, -0.04927]	36/0/4	12.09
Ordinary (10)	-0.450 [-0.795, -0.088]	+0.03192 [+0.01265, +0.05062]	13/0/27	40.97
Ordinary (10) restart	-0.155 [-0.564, +0.287]	+0.04870 [+0.02614, +0.07089]	16/0/24	87.73
HDR				
Ordinary (1)	+2.529 [+1.261, +3.890]	-0.02932 [-0.06137, -0.00184]	27/0/13	2.76
Ordinary (10)	+2.606 [+1.351, +3.967]	-0.04534 [-0.07251, -0.02239]	30/0/10	4.34
Ordinary (10) restart	+1.792 [+0.909, +2.759]	-0.03535 [-0.05699, -0.01552]	28/0/12	7.22
Five-task mean				
Ordinary (1)	+0.961 [+0.664, +1.264]	-0.01613 [-0.02502, -0.00728]	34/0/6	–
Ordinary (10)	+0.762 [+0.448, +1.084]	-0.01612 [-0.02640, -0.00577]	32/0/8	–
Ordinary (10) restart	+0.656 [+0.392, +0.932]	-0.01038 [-0.02048, -0.00028]	31/0/9	–

The PSNR differences, averaged equally over the two tasks, relative to ordinary refinement with up to 2, 5, and 10 corrections are $+1.661$ (CI $[+1.002, +2.386]$), $+1.591$ ($[+0.844, +2.401]$), and $+1.522$ dB ($[+0.922, +2.180]$). Relative to the restarted ten-correction control the effect is $+1.114$ dB ($[+0.687, +1.573]$). These comparisons isolate changes in correction direction and depth.

E.3 Five-task reconstruction and component comparisons

Table 16 compares VEG and repeated VEG with published-method adaptations on the five tasks, retaining all quality metrics and runtime. Table 17 separately compares VEG with one or ten ordinary corrections and ten corrections with restarted line searches. Its five-task mean summarizes the broader evaluation; the prespecified inpainting/HDR average remains the primary transfer comparison in Appendix E.2. In Gaussian deblurring and nonlinear blur, deeper ordinary refinement can exceed one-correction VEG, so the two-task finding does not imply uniform dominance over deeper controls.

E.4 Repeated correction and computational cost

On 10 separate development images, we choose the smallest tested correction depth within 0.15 dB of the best PSNR attained by repeated VEG, averaged equally over tasks. This gives $K = 10$. We then evaluate repeated VEG at that depth on all 40 previously unused images and five tasks, using the same pretrained weights, classes, observations, initial noise, and continuation draws as VEG. Each interval retains the measurement weight $B_n = 1$, and line-search acceptance determines which correction attempts are applied.

The primary comparison averages the five PSNR differences between repeated VEG and VEG within each image, then across the 40 images. Confidence intervals use 20,000 bootstrap resamples of images, keeping all five task outcomes together. Repeated VEG has higher average PSNR on 38 images, ties on 0, and lower PSNR on 2. The effect depends on the task: Gaussian deblurring, nonlinear blur, and



Figure 7: JiT reconstructions for Gaussian deblurring (ImageNet #9), $4\times$ SR (#29), box inpainting (#28), nonlinear blur (#3), and HDR (#8). All methods share the measurement, ground truth, and zoom region within each row. Labels report full-image PSNR (dB) / standard LPIPS from raw solver outputs, displayed with a fixed intensity mapping. D-Flow-JiT uses Gaussian initialization on nonlinear tasks. Table 16 gives task-level results.

Table 18: Repeated VEG ($K = 10$) minus one-correction VEG on the 40 evaluation images. Depth is chosen on separate development images. Brackets give 95% image-level bootstrap intervals; task intervals are nominal. The equal-task row averages five differences within each image. Runtime ratios use unrounded means from the common A100 benchmark; absolute task times are in Table 16.

Task	ΔPSNR [95% CI]	ΔLPIPS [95% CI]	Runtime / VEG
Gaussian deblur	+0.319 [+0.255, +0.379]	-0.03424 [-0.03825, -0.03023]	5.04×
4× SR	-0.075 [-0.210, +0.051]	+0.00499 [-0.00148, +0.01172]	5.03×
Box inpainting	+0.154 [-0.145, +0.469]	-0.00911 [-0.01644, -0.00279]	5.05×
Nonlinear blur	+1.589 [+1.323, +1.858]	-0.10499 [-0.12209, -0.08810]	5.07×
HDR	+2.223 [+1.190, +3.412]	-0.05652 [-0.08172, -0.03584]	4.96×
Equal task	+0.842 [+0.596, +1.124]	-0.03997 [-0.04663, -0.03376]	5.04×

HDR have positive intervals; SR is slightly negative and unresolved, and inpainting is unresolved. Secondary intervals are nominal.

The aggregate quality improvement costs $5.04\times$ the runtime of one-correction VEG. Super-resolution measurement fit worsens with repetition: ΔNRMSE is $+0.02341$ (95% CI $[+0.02021, +0.02681]$), consistent with its task-dependent quality tradeoff. Figure 7 illustrates all five tasks.

F Correction Depth and Scope

F.1 Pixel-space correction depth

This development study evaluates ordinary and repeated VEG correction at $K \in \{1, 2, 3, 5, 10\}$. It uses 10 AFHQ-Cat and 10 CelebA development images on nonlinear blur and HDR, and 10 separate JiT development images on five tasks. Correction depth is chosen without using the 40 JiT evaluation images. Each interval retains $B_n = 1$, its original clean-prediction Jacobian and penalty center, and the same line-search safeguards. The accepted step length carries between inner corrections, with a reset at each outer interval.

For all three pixel-space predictors, geometry-guided refinement yields higher PSNR and lower LPIPS than ordinary refinement when both allow up to 10 corrections. These development-set results

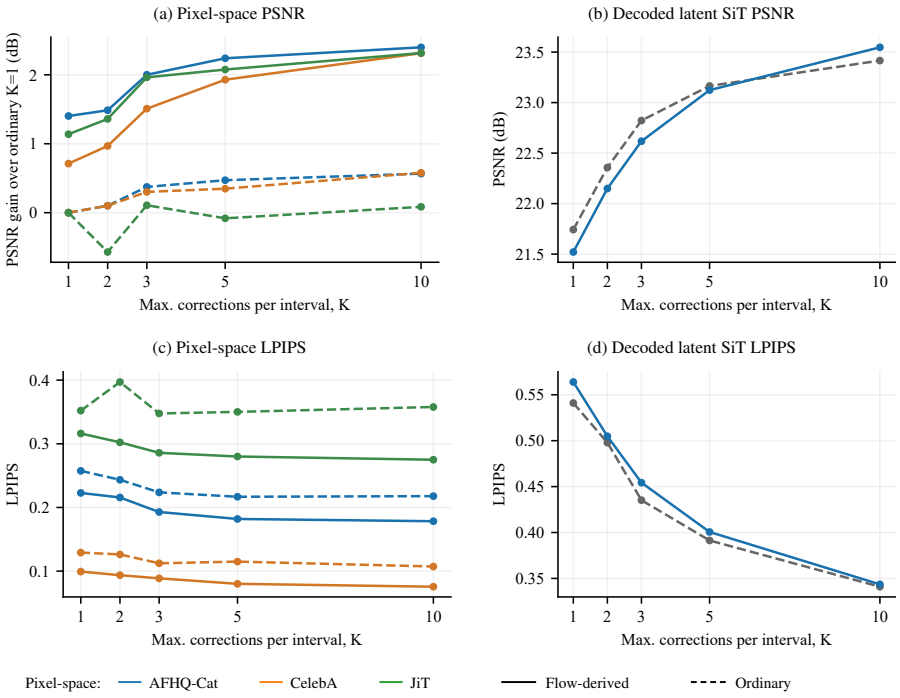


Figure 8: Correction depth on the development sets. Solid lines use flow-derived directions; dashed lines use ordinary gradients. Pixel-space panels compare AFHQ-Cat, CelebA, and JiT; their PSNR subtracts each predictor's one-ordinary-correction mean. SiT panels use raw decoded-image PSNR. All LPIPS values use raw solver or decoded outputs with `normalize=False`, and tasks are averaged equally within a predictor. Both SiT directions improve with depth; their $K = 10$ difference remains unresolved. Tables 19 and 20 report paired effects.

Table 19: Effect of the flow-derived direction at the same correction depth on the development sets: repeated VEG minus ordinary gradient refinement, both allowing up to 10 corrections. Positive PSNR and negative LPIPS favor the flow-derived direction. Brackets give the analysis intervals. These development images also determine the correction depth.

Predictor	ΔPSNR [95% CI]	ΔLPIPS [95% CI]
AFHQ-Cat	+1.832 [+1.429, +2.372]	-0.03940 [-0.05147, -0.02954]
CelebA	+1.734 [+1.055, +2.446]	-0.03172 [-0.04523, -0.01823]
JiT	+2.231 [+1.219, +3.252]	-0.08286 [-0.10761, -0.06090]

support a benefit from the correction direction beyond increasing local optimization depth. Repeated VEG adds clean-prediction VJPs at positive-time intervals. Figure 8 shows the depth trends, and Table 19 compares the directions with up to 10 corrections.

F.2 Matched-depth comparison with a decoded latent flow

This study uses the official SiT-XL/2 ImageNet-256^2 latent flow (Ma et al., 2024), true class labels, guidance scale 1, and the pretrained `stabilityai/sd-vae-ft-mse` decoder (Rombach et al., 2022; Stability AI, 2022) with its weights held fixed and latent scaling 0.18215. Both direction families use $N = 100$, $B_{\text{total}} = 100$, $B_n = 1$, summed Gaussian measurement loss with variance $0.05^2 + 10^{-12}$, and the same quadratic-penalty schedule. Raw decoded images enter the likelihood; the flow Jacobian remains at the original interval state while each inner gradient passes through the decoder.

We use FP32 and Armijo line search with initial step 1, shrink factor $1/2$, slope coefficient 10^{-4}, minimum step 10^{-8}, and a 30-backtrack cap. The accepted step is carried between inner corrections

Table 20: SiT correction-depth comparison with the same objective on 10 development images and five tasks, with $B_n = 1$ for both direction families. Each pair of quality columns gives means with equal weight for each task. The last column is paired VJP minus ordinary PSNR with a nominal 95% image-level bootstrap interval. The K10 difference remains unresolved.

K	Ord. PSNR	Ord. LPIPS	VJP PSNR	VJP LPIPS	ΔPSNR [95% CI]
1	21.743	0.54103	21.521	0.56388	-0.222 [-0.428, +0.008]
2	22.357	0.49787	22.149	0.50484	-0.208 [-0.293, -0.131]
3	22.822	0.43520	22.617	0.45431	-0.205 [-0.287, -0.111]
5	23.165	0.39135	23.123	0.40060	-0.042 [-0.174, +0.106]
10	23.416	0.34100	23.547	0.34354	+0.131 [-0.070, +0.361]

and reset at the next outer interval. At $t = 0$, geometry-guided refinement uses one ordinary correction as a fallback; no Jacobian is refreshed at a corrected state. Images are converted to RGB tensors, resized to 256, and center-cropped. SiT uses the DAPS bicubic SR operator, whereas the primary image-space and JiT evaluations use strided sampling. The super-resolution operators therefore differ across predictors.

The development comparison covers 10 images, five tasks, both direction families, and every tested correction depth. Development metrics use raw decoded outputs and standard LPIPS without clipping. Tasks receive equal weight, and nominal 95% bootstrap intervals keep all five task outcomes from an image together. Both direction families improve strongly with depth, but their difference at $K = 10$ is unresolved: $+0.131$ dB (95% CI $[-0.070, +0.361]$) and LPIPS $+0.00255$ ($[-0.01932, +0.02195]$). The negative PSNR intervals at $K = 2$ and $K = 3$, together with the unresolved difference at $K = 10$, show that additional local optimization and a geometry-specific benefit are distinct in this setting.

Scope of local reconstruction correction. Local reconstruction gains do not imply recovery of the full posterior distribution: exploratory Gaussian-mixture tests favor explicit posterior reweighting over local correction. Gains are weak or negative for tanh deblurring, and phase retrieval retains rare catastrophic failures despite improvements on typical cases. Additional Adam refinement did not remove those failures.