LARC: A GENERAL LATE-STAGE ADAPTIVE RECONSTRUCTION CORRECTOR FOR DIFFUSION INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models provide powerful generative priors for solving imaging inverse problems, enabling high-quality reconstructions from incomplete or corrupted measurements. However, further improving their reconstruction quality can be computationally expensive, particularly when refinement involves repeated inner sampling or optimization across noise levels. Motivated by the coarse-to-fine behavior of reverse diffusion sampling, we introduce the Late-stage Adaptive Reconstruction Corrector (LARC), a general corrector that refines image details based on the structure established during early sampling. LARC can be integrated into a broad range of diffusion inverse solvers, preserving the host solver's early sampling trajectory and refining its later stages. At each selected noise level, LARC performs a two-stage correction. The first is measurement-guided initialization, which solves a regularized minimization problem to obtain an initial estimate guided by the observation. The second is preconditioned score refinement, which further refines the initial estimate through score updates with direction-dependent step sizes. Together, these stages combine measurement guidance with the diffusion prior to refine image details through a small number of local updates. Experiments on FFHQ and ImageNet across inpainting, super-resolution, and deblurring demonstrate that LARC outperforms competing baselines in reconstruction accuracy and consistently improves the tested host solvers with limited additional computation.

1 INTRODUCTION

Inverse problems arise in a wide range of imaging applications, including image deblurring, super-resolution, and inpainting (Zhu et al., 2023; Alkhouri et al., 2025). These problems aim to recover an unknown image $x_0 \in \mathbb{R}^d$ from measurements $y \in \mathbb{R}^m$ generated by

$$y = Ax_0 + n, \tag{1}$$

where A is a known linear measurement operator and n denotes measurement noise. However, recovering x_0 from y is often ill-posed, as the reconstruction may be non-unique or highly sensitive to measurement noise (Hansen, 2010).

Recent diffusion-based methods address this ill-posedness by using a generative model pretrained on clean images as an image prior (Song et al., 2021). During inference, the measurement y is incorporated into the reverse diffusion process to guide sampling toward a reconstruction that is both realistic and consistent with the observation (Chung et al., 2023). In score-based posterior sampling, this involves combining the pretrained prior score with a measurement-dependent likelihood score, which is generally unavailable in closed form at intermediate noise levels.

Earlier diffusion-based solvers exploit the structure of the measurement operator to incorporate observations into reverse sampling (Kawar et al., 2022; Wang et al., 2023), or approximate the intractable likelihood score to guide the sampling process (Chung et al., 2023; Song et al., 2023). Other approaches formulate reconstruction as an optimization problem that balances data fidelity with a diffusion-based image prior (Zhu et al., 2023; Mardani et al., 2024). More recently, methods such as DAPS and SITCOM further refine intermediate reconstructions through inner sampling or optimization before proceeding to the next noise level (Zhang et al., 2025; Alkhouri et al., 2025).

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.

These methods show the benefits of intermediate refinement, but repeated inner updates across noise levels can be costly.

Recent analyses and experiments suggest that reverse diffusion sampling generally follows a coarse-to-fine pattern, with global structure emerging early and finer details developing later (Wang & Vastola, 2023; Sclocchi et al., 2025). This suggests that later sampling stages offer an opportunity to refine image details based on an already informative reconstruction. Therefore, we seek a general corrector that can be plugged into different diffusion inverse solvers to refine their late-stage sampling trajectories with limited additional computation.

In this paper, we introduce the Late-stage Adaptive Reconstruction Corrector (LARC), a general and efficient sampling corrector that can be attached to a broad range of existing diffusion inverse solvers to improve their reconstruction quality at little additional cost. LARC retains the early sampling trajectory of the host solver and is activated after an informative intermediate reconstruction has been obtained. At each selected noise level, LARC performs a two-stage correction. The first is measurement-guided initialization, which solves a regularized minimization problem to obtain an initial estimate. The second is preconditioned score refinement, which further refines this initial estimate through a few adaptive score updates. Specifically, this adaptive refinement uses a measurement-dependent matrix to adapt the step size across directions, taking smaller steps in well-observed directions and relatively larger steps in weakly observed directions. Together, these stages combine measurement guidance with the diffusion prior to refine image details through a small number of local updates. Further theoretical analysis establishes prior-energy descent and bounds the accompanying increase in local correction energy under an exact-score assumption and suitable smoothness and step-size conditions. Experiments on FFHQ and ImageNet demonstrate that LARC outperforms competing baselines in reconstruction accuracy and consistently improves the tested host solvers across multiple inverse problems with limited additional computation. Our main contributions are summarized as follows:

- We introduce a general late-stage corrector that can be integrated into different diffusion inverse solvers. LARC preserves the host solver's early sampling trajectory and improves reconstruction through subsequent trajectory correction.
- We develop a two-stage local update that combines measurement-guided initialization with preconditioned score refinement. This design combines measurement information and the diffusion prior to improve intermediate reconstructions through a small number of local updates.
- We evaluate LARC on FFHQ and ImageNet across inpainting, super-resolution, and deblurring tasks using multiple host solvers. The results demonstrate that LARC outperforms competing baselines in reconstruction accuracy and consistently improves the tested host solvers across multiple inverse problems with limited additional computation.

2 Background and Related Work

2.1 Diffusion Preliminaries

Pretrained diffusion models generate images by reversing a predefined noising process (Ho et al., 2020; Song et al., 2021). Under the variance-preserving stochastic differential equation formulation, the forward and reverse processes are

$$\mathrm{d}x_t = -\frac{1}{2}\beta(t)x_t\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}w_t, \tag{2}$$

$$\mathrm{d}x_t = -\beta(t)\left[\frac{1}{2}x_t + \nabla_{x_t}\log p_t(x_t)\right]\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\bar{w}_t, \tag{3}$$

where $\beta(t)$ controls the rate of noise injection, w_t and $\bar{w}_t$ are forward and reverse Wiener processes, and p_t is the distribution of the noisy image at time t. The reverse process is integrated from large to small t, with the score approximated by a pretrained network $s_\theta(x_t, t)$.

For a discrete noise schedule, let $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$. The forward process is then

$$x_t = \sqrt{\bar{\alpha}_t}\,x_0 + \sqrt{1-\bar{\alpha}_t}\,\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I). \tag{4}$$

Under this perturbation model, Tweedie's formula gives the following clean-image estimate when the score is approximated by s_θ (Chung et al., 2023):

$$\widehat{x}_0(x_t) = \frac{x_t + (1-\bar{\alpha}_t)s_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}. \tag{5}$$

This estimate approximates the conditional mean $\mathbb{E}[x_0 \mid x_t]$ and is widely used to introduce diffusion prior information during reconstruction.

For posterior sampling in inverse problems, the unconditional score in equation (3) is replaced by the conditional score, which, by Bayes' rule, is

$$\nabla_{x_t} \log p_t(x_t \mid y) = \underbrace{\nabla_{x_t} \log p_t(x_t)}_{\text{prior score}} + \underbrace{\nabla_{x_t} \log p_t(y \mid x_t)}_{\text{likelihood score}}. \tag{6}$$

The pretrained network supplies the prior score, but the likelihood score at specific noise levels is generally unavailable in closed form (Chung et al., 2023).

2.2 Related Work

Diffusion priors for inverse problems. Diffusion-based inverse solvers combine observations with a pretrained image prior through operator-based corrections, measurement guidance, or optimization. For linear measurement models, DDRM uses a singular value decomposition of the operator and incorporates observations into the reverse updates along the resulting singular directions (Kawar et al., 2022). In the noiseless setting, DDNM uses $A^\dagger$ to enforce data consistency in each clean prediction while preserving its null-space component (Wang et al., 2023).

A second approach constructs measurement guidance from the current noisy state and its clean-image estimate. DPS approximates $p_t(y \mid x_t)$ by $p\big(y \mid \widehat{x}_0(x_t)\big)$ and differentiates this approximation with respect to x_t to guide reverse sampling (Chung et al., 2023). MCG also uses a Tweedie-based estimate, introducing a manifold-constrained gradient correction to reduce departures from the learned data manifold (Chung et al., 2022). ΠGDM accounts for uncertainty in the clean-image estimate through an isotropic Gaussian approximation to $p(x_0 \mid x_t)$, leading to pseudoinverse-based measurement guidance (Song et al., 2023). Optimization-based methods combine data fidelity and a diffusion prior within a reconstruction objective function. DiffPIR uses half-quadratic splitting to separate measurement correction from diffusion denoising (Zhu et al., 2023). At each reverse timestep, it corrects a clean-image prediction by solving a data-fidelity subproblem and uses the corrected estimate to construct the next noisy state. RED-Diff optimizes an image estimate under a variational objective, using denoisers across noise levels as regularizers weighted by the signal-to-noise ratio (Mardani et al., 2024).

Intermediate refinement during sampling. In addition to transitions between noise levels, diffusion sampling can include updates at a fixed noise level. In predictor-corrector sampling, a numerical predictor takes one step of the reverse-time process, while a corrector applies score-based Markov chain Monte Carlo such as Langevin dynamics at the current noise level (Song et al., 2021). For inverse reconstruction, several methods further separate measurement correction from diffusion sampling or perform multiple updates within each reconstruction stage. DCDP alternates between data-fidelity optimization and diffusion purification, which adds noise to the current reconstruction and then denoises it using the pretrained model (Li et al., 2024). DAPS uses Langevin dynamics to sample a clean image from an approximate conditional distribution given the current noisy state and the measurement, then adds noise to the sample to reach the next noise level (Zhang et al., 2025). SITCOM performs multiple gradient updates on the input of the pretrained network so that its clean-image prediction fits the measurement, while regularization keeps this input close to the current noisy state (Alkhouri et al., 2025). It then resamples this prediction at the next noise level.

Feature evolution during diffusion. Theoretical and empirical studies also examine feature evolution during diffusion. Using a Gaussian model of the data distribution, Wang & Vastola (2023) analyze probability-flow trajectories and relate the order of feature emergence to variance: high-variance features tend to emerge before low-variance details. Sclocchi et al. (2025) study a hierarchical data model and ImageNet images through forward–backward experiments, in which an image is corrupted to a chosen noise level and then denoised. They find a threshold in the recovery of high-level features such as image class, while lower-level features change more gradually with the starting noise level. These findings suggest a coarse-to-fine pattern in reverse diffusion sampling, motivating late-stage detail refinement for inverse problems.

3 METHODOLOGY

In this section, we introduce LARC, a general corrector that can be integrated into different diffusion inverse solvers. Specifically, LARC preserves the host solver's early sampling trajectory and refines its later stages to improve image details.

First, we express the sampling trajectory in normalized coordinates for simplicity. Using the forward perturbation model introduced in Section 2.1, we write

$$u_t = \frac{x_t}{\sqrt{\bar{\alpha}_t}} = x_0 + \sqrt{\lambda_t}\epsilon, \qquad \lambda_t = \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t}, \tag{7}$$

where $\epsilon \sim \mathcal{N}(0, I)$. Here, u_t denotes the normalized sampling state, and λ_t is the corresponding noise variance. The normalized score is then $s_{\theta,t}(u) = \sqrt{\bar{\alpha}_t}\, s_\theta(\sqrt{\bar{\alpha}_t}\, u, t)$. The normalized and original coordinates coincide at $t = 0$ since $\bar{\alpha}_0 = 1$.

Let t_s denote the noise level at which LARC is activated. The overall reverse sampling process can be summarized as

$$\underbrace{u_T \to \cdots \to u_{t_\mathrm{s}}}_{\text{Host solver's trajectory}} \to \overbrace{\underbrace{\tilde{u}_{t_\mathrm{s}}^{(0)} \to \cdots \to \tilde{u}_{t_\mathrm{s}}^{(K)}}_{\text{Late-stage sampling with LARC}}}^{K \text{ local updates}} \to u_{t_\mathrm{s}-1} \to \cdots \to u_0\,. \tag{8}$$

Starting from the host solver's state u_{t_s}, LARC refines the subsequent sampling trajectory. Within each selected late sampling step, LARC performs K local updates for refinement. We detail the correction procedure in the following subsection.

3.1 LATE-STAGE ADAPTIVE RECONSTRUCTION CORRECTOR

In this subsection, we detail the LARC correction procedure, which consists of two stages, performed at a fixed noise level t. The first stage, **measurement-guided initialization**, constructs an initial estimate from the observation and the current sampling state. The second stage, **preconditioned score refinement**, further refines this estimate through score updates with direction-dependent step sizes. Figure 1 provides a geometric illustration of the two-stage correction and the subsequent transition to the next noise level.

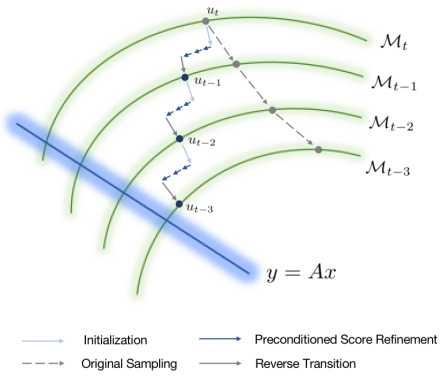

Figure 1: LARC mechanism. $\mathcal{M}_t$ denotes the diffusion manifold at step t.

Stage 1: Measurement-guided initialization. Using Tweedie's formula, we obtain the clean prediction

$$\hat{u}_0(u_t) = u_t + \lambda_t s_{\theta,t}(u_t). \tag{9}$$

Although this prediction incorporates information from the diffusion prior, it may not be consistent with the observation y. We therefore initialize the inner updates by minimizing the following objective, which balances measurement fidelity with proximity to both the clean prediction and the current sampling state:

$$F_t(\widetilde{u}) = \frac{1}{2\sigma^2}\|A\widetilde{u} - y\|^2 + \frac{1}{2\tau_t}\|\widetilde{u} - \hat{u}_0(u_t)\|^2 + \frac{1}{2\ell_t}\|\widetilde{u} - u_t\|^2. \tag{10}$$

Here, $\sigma > 0$ is the observation noise scale, and $\tau_t, \ell_t > 0$ control the strengths of the two regularization terms. The first term promotes consistency with the observation, the second anchors the estimate to the clean prediction, and the third penalizes departures from the current sampling state, keeping the initialization local to the host trajectory. We refer to F_t as the local correction energy. Its minimizer provides the initial estimate:

$$\widetilde{u}_t^{(0)} = \arg\min_{\widetilde{u}} F_t(\widetilde{u}) = G_t\left(\frac{\tau_t}{\sigma^2}A^\top y + \frac{\tau_t}{\ell_t}u_t + \hat{u}_0(u_t)\right), \tag{11}$$

where $G_t = [\frac{\tau_t}{\sigma^2}A^\top A + \left(1 + \frac{\tau_t}{\ell_t}\right) I]^{-1}$. For common structured degradation operators, G_t can be applied efficiently without forming a dense inverse, using elementwise operations for inpainting or FFT-based filtering for circular-convolution deblurring (Zhu et al., 2023).

The following proposition connects the initialization to the local curvature of F_t.

Proposition 1 (Initialization and correction geometry). *For $\sigma, \tau_t, \ell_t > 0$, the matrix G_t is symmetric positive definite, and $\widetilde{u}_t^{(0)}$ is the unique minimizer of F_t. Moreover,*

$$\nabla^2 F_t = \frac{1}{\sigma^2} A^\top A + \left(\frac{1}{\tau_t} + \frac{1}{\ell_t}\right) I = \tau_t^{-1} G_t^{-1}, \tag{12}$$

and, for any estimate $\widetilde{u}$,

$$F_t(\widetilde{u}) - F_t(\widetilde{u}_t^{(0)}) = \frac{1}{2\tau_t} \|\widetilde{u} - \widetilde{u}_t^{(0)}\|^2_{G_t^{-1}}, \tag{13}$$

where $\|v\|_M^2 = v^\top M v$.

The proof is given in Appendix A.1.1. Thus, G_t is a scaled inverse Hessian of the local correction energy, motivating its use in the subsequent score refinement to take smaller steps along directions of higher curvature.

Stage 2: Preconditioned score refinement. Although the initialization in Stage 1 incorporates prior information through the fixed clean prediction $\hat{u}_0(u_t)$, measurement corrections may still introduce artifacts, since proximity to this prediction does not ensure consistency with the learned image prior (Alkhouri et al., 2025). We therefore further refine $\widetilde{u}_t^{(0)}$ using the diffusion prior, reevaluating the score at each inner iterate to obtain prior guidance at the current estimate. To limit the changes in directions strongly constrained by the measurements, we precondition these updates with the matrix G_t from Stage 1.

Specifically, we perform K preconditioned score updates at the fixed noise level t:

$$\widetilde{u}_t^{(k+1)} = \widetilde{u}_t^{(k)} + \eta_t G_t s_{\theta,t}\left(\widetilde{u}_t^{(k)}\right), \qquad k = 0, \dots, K-1, \tag{14}$$

where $\eta_t > 0$ is the scalar refinement step size.

The matrix G_t scales the score updates according to the curvature of the same local correction energy used for initialization. By Proposition 1, $G_t = (\tau_t \nabla^2 F_t)^{-1}$, so directions with greater local curvature receive smaller update scales. Specifically, let q_i be a unit eigenvector of $A^\top A$ with eigenvalue $\nu_i \geq 0$, so that $A^\top A q_i = \nu_i q_i$. Then

$$G_t q_i = \frac{1}{1 + \tau_t/\ell_t + \tau_t \nu_i/\sigma^2}\, q_i. \tag{15}$$

Since $\nu_i = \|Aq_i\|^2$, larger eigenvalues correspond to directions to which the measurements are more sensitive. Thus, score components are attenuated more strongly in well-observed directions with larger ν_i, while relatively larger updates are allowed in weakly observed directions with smaller ν_i. This directional scaling allows the diffusion prior to refine image details while limiting changes to components strongly constrained by the measurements.

Algorithmic summary. After the K inner updates, we blend the final refined estimate $\widetilde{u}_t^{(K)}$ with the original clean prediction $\hat{u}_0(u_t)$ to obtain the final reconstruction estimate at the current noise level:

$$\hat{u}_0^{\mathrm{LARC}}(u_t) = (1 - \rho_t)\hat{u}_0(u_t) + \rho_t \widetilde{u}_t^{(K)}, \qquad \rho_t \in [0, 1]. \tag{16}$$

The weight ρ_t interpolates between the original denoiser prediction ($\rho_t = 0$) and the final refined estimate ($\rho_t = 1$).

To advance to the next noise level, we compute the noise estimate associated with the corrected prediction and use it in the following reverse transition:

$$\hat{\epsilon}_t = \frac{u_t - \hat{u}_0^{\mathrm{LARC}}(u_t)}{\sqrt{\lambda_t}}, \tag{17}$$

$$u_{t-1} = \hat{u}_0^{\mathrm{LARC}}(u_t) + \sqrt{\lambda_{t-1}}\left[\sqrt{1-\zeta}\,\hat{\epsilon}_t + \sqrt{\zeta}\,\epsilon_t\right], \tag{18}$$

where $\epsilon_t \sim \mathcal{N}(0, I)$ is sampled independently and $\zeta \in [0, 1]$ controls the contribution of fresh noise. When $\zeta = 0$, the transition is deterministic given u_t; when $\zeta = 1$, the next state is sampled by adding Gaussian noise to the corrected clean prediction. Here $t - 1$ denotes the next level on the sampling schedule; the corresponding state in the original coordinates is $x_{t-1} = \sqrt{\bar{\alpha}_{t-1}} u_{t-1}$.

Algorithm 1 summarizes the LARC correction at a fixed noise level t. Starting from the host solver's state at the activation noise level t_s, LARC applies Algorithm 1 at each remaining sampling step until the final reconstruction is obtained. This design improves reconstruction accuracy by refining image details based on the structure established by the host solver, with limited additional computation.

Algorithm 1 LARC at a selected late sampling step t

Require: u_t, $\lambda_t > 0$, $\lambda_{t-1} \geq 0$, (A, y), $s_{\theta,t}$, $\sigma, \tau_t, \ell_t, \eta_t > 0$, integer $K \geq 1$, and $\rho, \zeta \in [0, 1]$.
1: $\hat{u}_0(u_t) \leftarrow u_t + \lambda_t s_{\theta,t}(u_t)$.
2: **Stage 1: Measurement-guided initialization.**
3: $G_t \leftarrow \left[\frac{\tau_t}{\sigma^2} A^\top A + (\frac{\tau_t}{\ell_t} + 1) I\right]^{-1}$, $\widetilde{u}_t^{(0)} \leftarrow G_t \left(\frac{\tau_t}{\sigma^2} A^\top y + \frac{\tau_t}{\ell_t} u_t + \hat{u}_0(u_t)\right)$.
4: **Stage 2: Preconditioned score refinement.**
5: **for** $k = 0, \ldots, K-1$ **do**
6: $\quad \widetilde{u}_t^{(k+1)} \leftarrow \widetilde{u}_t^{(k)} + \eta_t G_t s_{\theta,t}(\widetilde{u}_t^{(k)})$.
7: **end for**
8: $\hat{u}_0^{\mathrm{LARC}}(u_t) \leftarrow (1 - \rho_t)\hat{u}_0(u_t) + \rho_t \widetilde{u}_t^{(K)}$.
9: $\hat{\epsilon}_t \leftarrow \big(u_t - \hat{u}_0^{\mathrm{LARC}}(u_t)\big)/\sqrt{\lambda_t}$; draw $\epsilon_t \sim \mathcal{N}(0, I)$ independently.
10: $u_{t-1} \leftarrow \hat{u}_0^{\mathrm{LARC}}(u_t) + \sqrt{\lambda_{t-1}}\big[\sqrt{1-\zeta}\,\hat{\epsilon}_t + \sqrt{\zeta}\,\epsilon_t\big]$.
11: **return** u_{t-1}.

3.2 Theoretical Analysis

We further analyze the preconditioned score refinement to establish prior-energy descent and quantify the departure from the measurement-guided initialization. At a fixed noise level t, let $\Phi_t(\widetilde{u}) = -\log p_t^u(\widetilde{u})$, where p_t^u is the prior density at noise level t in normalized coordinates. For this analysis, we assume an exact score $s_{\theta,t} = -\nabla\Phi_t$ and that $\nabla\Phi_t$ is L_t-Lipschitz on a convex region containing all inner iterates, with $L_t > 0$.

Theorem 1 (Prior-energy descent)**.** *Under the above assumptions, let $\sigma, \tau_t, \ell_t > 0$ and choose*

$$0 < \eta_t < \frac{2}{L_t \lambda_{\max}(G_t)}, \qquad c_t := 1 - \frac{L_t \eta_t \lambda_{\max}(G_t)}{2} > 0, \tag{19}$$

where $\lambda_{\max}(G_t)$ denotes the largest eigenvalue of G_t. Then each update in equation (14) satisfies

$$\Phi_t(\widetilde{u}_t^{(k+1)}) \leq \Phi_t(\widetilde{u}_t^{(k)}) - \eta_t c_t \|\nabla\Phi_t(\widetilde{u}_t^{(k)})\|_{G_t}^2. \tag{20}$$

Thus, the prior energy is non-increasing and decreases strictly whenever $\nabla\Phi_t(\widetilde{u}_t^{(k)}) \neq 0$. Since $\widetilde{u}_t^{(0)}$ minimizes F_t, any departure from this initialization increases the local correction energy relative to its minimum. The following theorem bounds this increase in terms of the cumulative decrease in prior energy.

Theorem 2 (Controlled increase in local correction energy)**.** *Under the conditions of Theorem 1, the iterates initialized by equation (11) satisfy, for any integer $K \geq 1$,*

$$0 \leq F_t(\widetilde{u}_t^{(K)}) - F_t(\widetilde{u}_t^{(0)}) \leq \frac{K\eta_t}{2\tau_t c_t} \left[\Phi_t(\widetilde{u}_t^{(0)}) - \Phi_t(\widetilde{u}_t^{(K)})\right]. \tag{21}$$

Together, these results show that, under the stated assumptions, the refinement decreases the prior energy while bounding its departure from the measurement-guided initialization in the G_t^{-1} metric. The bound also makes explicit the dependence on the number of inner updates and the refinement step size. Proofs are provided in Appendices A.1.2 and A.1.3.

4 Experiments

4.1 Experimental setup

We apply LARC to four host solvers: DPS (Chung et al., 2023), DiffPIR (Zhu et al., 2023), DAPS (Zhang et al., 2025), and FAST-DIPS (Kim et al., 2026b). Once each host solver has completed its early sampling phase, LARC is activated at the late stage and applied continuously to refine the reconstruction. The starting point is tuned on 10 development images. We compare each host solver with and without LARC and include SITCOM (Alkhouri et al., 2025), DDiff (Kim et al., 2026a), and the pseudoinverse-guided diffusion method ΠGDM (Song et al., 2023) as additional baselines. We use the official implementations of all host solvers and baselines. All experiments were performed on a single NVIDIA RTX 5880 GPU.

We evaluate on 100 FFHQ (Karras et al., 2019) and 100 ImageNet (Deng et al., 2009) images at 256×256 resolution. The tasks include random inpainting with observed-pixel ratios $r = 0.3$,

Table 1: Best and second-best results are **bold** and underlined, respectively. Green shading marks LARC variants, and gray shading marks sampling time. Dashes indicate unavailable results.

		FFHQ					ImageNet				
Task	Method	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓	Run-time (s)	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓	Run-time (s)
Inpaint ($r = 0.3$)	DPS	29.63	0.889	0.071	0.175	31.485	26.01	0.802	0.110	0.269	127.152
	DiffPIR	33.13	0.925	0.046	0.110	1.412	29.11	0.874	0.079	0.147	6.193
	ΠGDM	31.57	0.912	0.055	0.097	5.192	26.08	0.800	0.116	0.185	14.462
	DAPS	31.11	0.868	0.057	0.177	14.460	27.39	0.809	0.095	0.213	63.690
	SITCOM	33.61	0.937	0.043	0.097	22.239	29.42	0.891	0.078	0.119	97.203
	FAST-DIPS	31.31	0.888	0.056	0.188	2.413	27.71	0.822	0.091	0.222	5.847
	DDiff	30.02	0.912	0.068	0.103	25.931	26.71	0.821	0.102	0.206	136.131
	LARC (DPS)	33.98	0.948	0.042	0.084	12.244	30.66	0.913	**0.068**	0.112	67.475
	LARC (DiffPIR)	34.27	0.952	0.040	0.083	1.405	30.24	0.905	0.070	0.120	6.223
	LARC (DAPS)	32.78	0.941	0.048	0.102	4.362	29.27	0.879	0.076	0.169	25.153
	LARC (FAST-DIPS)	**34.73**	**0.956**	**0.038**	**0.081**	3.593	**30.70**	**0.915**	0.068	**0.109**	15.837
Inpaint ($r = 0.5$)	DPS	32.10	0.935	0.055	0.116	31.569	28.41	0.883	0.085	0.180	156.890
	DiffPIR	36.38	0.960	0.031	0.058	1.415	32.55	0.936	0.054	0.074	6.069
	ΠGDM	35.62	0.960	0.035	0.046	5.190	30.86	0.912	0.069	0.078	14.436
	DAPS	33.76	0.917	0.042	0.124	14.520	30.42	0.890	0.067	0.134	63.690
	SITCOM	36.90	0.967	0.030	0.047	23.549	33.00	0.945	0.052	0.059	100.322
	FAST-DIPS	33.62	0.922	0.043	0.148	2.039	30.48	0.889	0.066	0.152	5.863
	DDiff	32.79	0.948	0.052	0.060	26.752	29.56	0.899	0.077	0.110	136.285
	LARC (DPS)	37.55	0.974	0.028	0.039	12.244	34.59	0.960	**0.044**	0.051	65.625
	LARC (DiffPIR)	37.99	0.977	0.026	0.039	1.411	34.17	0.956	0.046	0.053	6.096
	LARC (DAPS)	36.93	0.970	0.030	0.048	4.311	32.07	0.933	0.055	0.097	17.748
	LARC (FAST-DIPS)	**38.34**	**0.978**	**0.025**	**0.038**	3.608	**34.63**	**0.960**	0.044	**0.050**	15.782
Inpaint ($r = 0.7$)	DPS	34.50	0.965	0.044	0.063	31.845	31.11	0.939	0.064	0.100	126.751
	DiffPIR	39.79	0.980	0.021	0.028	1.414	36.18	0.969	0.036	0.035	6.194
	ΠGDM	39.66	0.983	0.022	0.020	5.196	34.80	0.955	0.047	0.043	14.416
	DAPS	36.73	0.954	0.030	0.076	14.410	33.74	0.943	0.046	0.078	63.760
	SITCOM	40.57	0.985	0.019	0.020	24.285	36.87	0.975	0.034	0.027	102.133
	FAST-DIPS	35.10	0.939	0.036	0.125	2.055	32.51	0.922	0.052	0.119	5.828
	DDiff	35.09	0.969	0.042	0.038	25.923	32.25	0.940	0.059	0.067	136.950
	LARC (DPS)	41.19	0.987	0.018	0.017	12.184	38.53	0.982	0.028	0.022	48.537
	LARC (DiffPIR)	41.75	0.989	0.017	0.016	1.405	38.17	0.981	0.030	0.023	6.225
	LARC (DAPS)	40.85	0.987	0.019	0.019	4.309	35.99	0.971	0.036	0.037	17.763
	LARC (FAST-DIPS)	**42.05**	**0.990**	**0.016**	**0.016**	3.601	**38.68**	**0.982**	**0.028**	**0.021**	15.774
SR 2×	DPS	26.51	0.790	0.099	0.243	32.141	22.82	0.626	0.157	0.407	128.065
	DiffPIR	34.10	0.936	0.041	0.080	1.471	30.54	0.896	0.068	0.115	6.062
	ΠGDM	34.68	0.950	0.039	**0.068**	5.206	30.51	0.900	0.071	**0.099**	14.392
	DAPS	33.32	0.897	0.044	0.133	15.600	30.17	0.867	0.068	0.155	67.540
	SITCOM	34.07	0.929	0.041	0.165	22.866	30.95	0.899	0.063	0.162	97.985
	FAST-DIPS	33.98	0.927	0.041	0.152	2.459	30.43	0.885	0.067	0.175	6.168
	DDiff	30.90	0.918	0.064	0.166	27.201	26.02	0.767	0.115	0.307	136.615
	LARC (DPS)	**36.32**	**0.963**	**0.032**	0.078	7.805	31.11	0.912	0.066	0.137	63.239
	LARC (DiffPIR)	34.92	0.952	0.038	0.079	1.479	31.09	0.911	0.065	0.118	6.072
	LARC (DAPS)	35.99	0.962	0.033	0.079	5.248	**31.47**	**0.919**	**0.061**	0.127	18.625
	LARC (FAST-DIPS)	36.21	0.961	0.032	0.085	4.816	31.11	0.901	0.063	0.126	9.955
SR 4×	DPS	23.79	0.685	0.133	0.300	32.345	20.62	0.512	0.198	0.471	130.716
	DiffPIR	29.16	0.842	0.072	0.176	1.465	25.65	0.740	0.114	0.271	6.179
	ΠGDM	29.31	0.857	0.071	**0.151**	5.210	24.63	0.721	0.133	**0.219**	14.374
	DAPS	29.28	0.805	0.071	0.238	15.500	25.66	0.707	0.113	0.331	64.760
	SITCOM	28.49	0.818	0.077	0.291	16.361	25.44	0.719	0.114	0.369	75.262
	FAST-DIPS	30.16	0.861	0.064	0.243	2.468	26.12	0.748	0.108	0.331	6.134
	DDiff	28.47	0.868	0.080	0.224	27.468	24.44	0.710	0.131	0.350	137.052
	LARC (DPS)	30.33	0.884	0.063	0.213	8.492	26.24	0.767	0.107	0.314	27.856
	LARC (DiffPIR)	29.97	0.873	0.066	0.183	1.469	26.41	0.768	0.105	0.287	6.026
	LARC (DAPS)	30.40	0.884	0.062	0.212	4.377	26.23	0.768	0.107	0.316	16.026
	LARC (FAST-DIPS)	**30.75**	**0.887**	**0.060**	0.212	4.842	**26.43**	**0.773**	**0.105**	0.289	10.005
Gaussian deblur	DPS	23.92	0.679	0.130	0.285	32.423	21.00	0.516	0.188	0.419	128.662
	DiffPIR	27.27	0.766	0.089	**0.237**	1.447	23.41	0.594	0.145	0.364	6.100
	ΠGDM	25.87	0.730	0.105	0.257	5.209	22.06	0.568	0.169	**0.355**	14.439
	DAPS	27.60	0.750	0.085	0.292	35.490	24.17	0.620	0.133	0.430	85.590
	SITCOM	28.13	0.803	0.081	0.291	19.490	24.73	0.674	0.125	0.391	83.544
	FAST-DIPS	28.03	0.773	0.081	0.335	2.333	24.05	0.598	0.133	0.476	5.036
	DDiff	25.31	0.648	0.110	0.437	27.618	24.11	0.639	0.135	0.435	73.813
	LARC (DPS)	27.43	0.794	0.088	0.288	11.420	21.32	0.507	0.184	0.437	73.442
	LARC (DiffPIR)	28.07	0.794	0.081	0.248	5.282	**24.87**	**0.676**	**0.123**	0.375	6.103
	LARC (DAPS)	28.67	**0.822**	0.076	0.291	1.695	24.46	0.649	0.128	0.439	17.430
	LARC (FAST-DIPS)	**28.69**	0.820	**0.076**	0.285	1.877	24.68	0.667	0.126	0.409	3.906
Motion deblur	DPS	24.75	0.718	0.119	0.266	32.066	21.96	0.584	0.168	0.400	128.421
	DiffPIR	29.08	0.812	0.072	**0.205**	1.455	26.51	0.750	0.102	**0.284**	6.219
	ΠGDM	25.42	0.738	0.111	0.252	5.209	21.81	0.607	0.177	0.325	14.491
	DAPS	28.72	0.793	0.075	0.249	34.890	25.51	0.701	0.115	0.353	84.470
	SITCOM	29.92	0.848	0.066	0.248	21.612	26.86	0.776	0.098	0.295	91.531
	FAST-DIPS	27.53	0.676	0.085	0.324	2.312	26.33	0.750	0.104	0.330	3.182
	DDiff	25.66	0.649	0.106	0.405	27.129	26.07	0.729	0.106	0.350	73.823
	LARC (DPS)	30.73	0.859	0.060	0.230	11.161	26.58	0.756	0.102	0.345	27.551
	LARC (DiffPIR)	30.35	0.854	0.062	0.219	1.489	27.21	0.780	0.095	0.304	6.046
	LARC (DAPS)	30.66	0.866	0.060	0.230	4.601	27.17	0.781	0.095	0.336	14.912
	LARC (FAST-DIPS)	**30.77**	**0.866**	**0.060**	0.248	2.398	**27.27**	**0.791**	**0.094**	0.292	3.908

0.5, and 0.7, $2\times$ and $4\times$ super-resolution with bicubic downsampling, Gaussian deblurring with a 61×61 kernel of standard deviation 3, and motion deblurring with a 61×61 kernel of intensity 0.5. Both deblurring tasks use Gaussian measurement noise with standard deviation 0.05 on the $[0, 1]$ image scale, while inpainting and super-resolution are noise-free. We report PSNR, SSIM (Wang et al., 2004), NRMSE, LPIPS (Zhang et al., 2018), and average run time per image.

We use DPS's FFHQ checkpoint (Chung et al., 2023) and the unconditional ImageNet checkpoint of Dhariwal & Nichol (2021). See Appendix A.2 for fixed experimental configurations.

4.2 Main quantitative results

Table 1 reports quantitative results and runtime in seconds. Columns 3–7 (resp. 8–12) show the metrics for FFHQ (resp. ImageNet). The table covers 7 tasks, 4 restoration quality metrics, and 2 datasets, totaling 56 results. LARC ranks first on 48 results and second on 4.

Overall, LARC provides efficient reconstruction refinement across the tested restoration tasks. For example, on ImageNet motion deblurring, LARC (FAST-DIPS) achieves 27.27 dB in 3.908s per image. On ImageNet Gaussian deblurring, LARC (DiffPIR) achieves 24.87 dB and an LPIPS of 0.375 in 6.103s per image. These results illustrate the reconstruction quality attainable at practical runtime costs.

Compared with their corresponding baselines, LARC variants generally improve reconstruction accuracy at practical computational budgets. On $4\times$ super-resolution with FFHQ, LARC (DPS) raises PSNR from 23.79 to 30.33 dB using 250 network forward evaluations (NFE), and LARC (DAPS) raises PSNR from 29.28 to 30.40 dB using 210 NFE. On motion deblurring with ImageNet, LARC (DiffPIR) improves PSNR from 26.51 to 27.21 dB with a runtime of 6.046 s per image, and LARC (FAST-DIPS) improves PSNR from 26.33 to 27.27 dB with a runtime of 3.908 s per image. The NFE configurations are listed in Appendix A.2.

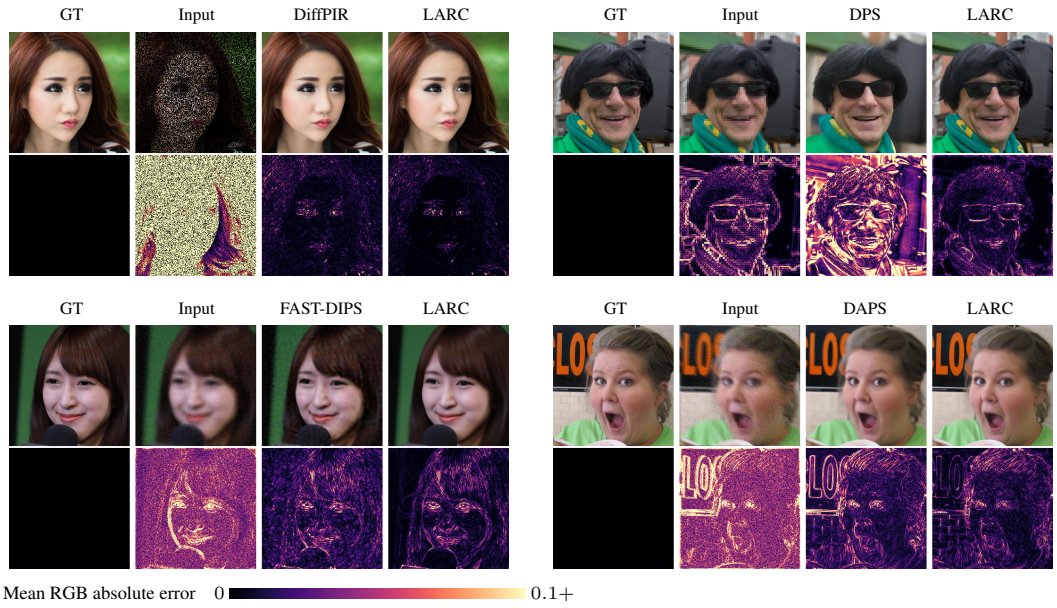

Figure 2: FFHQ reconstructions. Top left: inpainting ($r = 0.3$); top right: $4\times$ super-resolution; bottom left: Gaussian deblurring; bottom right: motion deblurring. LARC refines the baseline shown in the same panel. Error maps show mean RGB absolute error, saturated at 0.1.

4.3 Box inpainting on FFHQ

To further examine LARC's ability to directly generate content in unknown regions, we conduct box inpainting on the FFHQ dataset, where we mask randomly positioned 128×128 regions and reconstruct the missing content. Quantitative results are reported in Table 2. LARC (FAST-DIPS) achieves leading values on most metrics, whereas DPS obtains a better LPIPS. Figure 3 visualizes the results of DPS and LARC (FAST-DIPS). LARC produces reconstructions closer to the original image, while DPS may generate expressions that are not faithful to the ground truth.

Table 2: FFHQ box inpainting (100 images). Notation follows Table 1; hybrid timing uses batch size 1.

Method	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓	Run-time (s)
DPS	23.38	0.8719	0.1434	**0.0889**	31.426
DiffPIR	22.55	0.8643	0.1601	0.0961	1.400
ΠGDM	23.22	0.8656	0.1458	0.0892	5.200
DAPS	24.41	0.8783	0.1288	0.0897	14.380
SITCOM	24.45	0.8741	0.1263	0.1131	24.274
FAST-DIPS	24.22	0.8531	0.1320	0.1786	2.044
DDiff	23.28	0.8769	0.1458	0.1288	25.745
LARC (DPS)	24.36	0.8811	0.1297	0.1141	12.280
LARC (DiffPIR)	24.80	0.8876	0.1236	0.0903	1.423
LARC (DAPS)	24.61	0.8814	0.1258	0.0940	14.055
LARC (FAST-DIPS)	**25.23**	**0.8922**	**0.1159**	0.1018	4.019

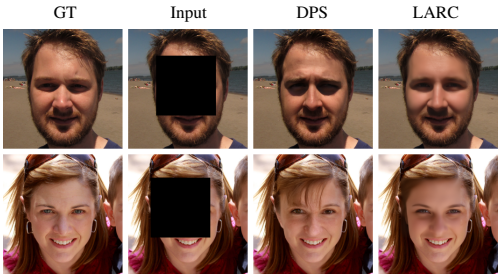

Figure 3: DPS and LARC (FAST-DIPS) on two FFHQ box-inpainting examples.

4.4 Ablation

Ablation on LARC Insertion. We compare DPS, DAPS, DiffPIR, and FAST-DIPS with and without LARC under shared early diffusion trajectories on inpainting (observed-pixel ratio $r = 0.3$), $4\times$ super-resolution, and motion deblurring. Figure 4 plots PSNR against NFE; the gray vertical dashed line marks LARC activation. Across all tested hosts, LARC improves PSNR over the original solvers after activation, showing the effectiveness of late-stage trajectory correction.

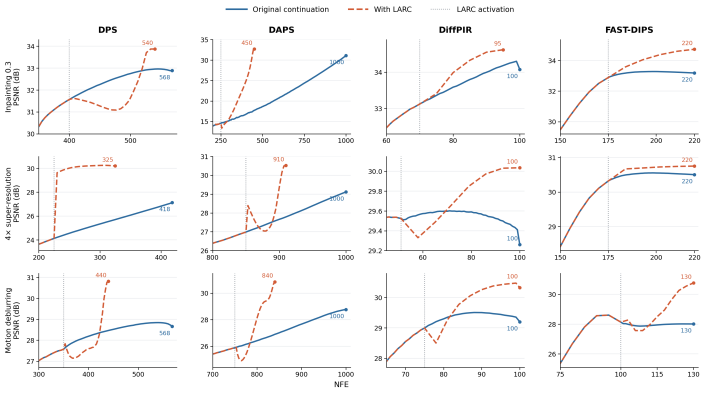

Figure 4: LARC insertion across four baselines.

Initialization and preconditioning. We ablate initialization and compare score refinement using G_t, $\bar{g}_t I = \mathrm{tr}(G_t)I/d$ (isotropic updates with matched average scaling), and $\mathbf{0}$ (no refinement). Table 3 shows that combining initialization with preconditioning achieves the best performance. For inpainting, adding initialization to isotropic refinement increases PSNR from 30.03 to 31.82 dB, and preconditioning further raises it to 34.27 dB.

Table 3: Initialization and preconditioning ablations for LARC (DiffPIR) on FFHQ (100 images per task). $\mathbf{0}$ disables refinement ($K = 0$), retaining initialization, the original outer schedules, and all other settings. Best values are bold.

		Inpainting ($r = 0.3$)				SR 4×				Motion deblurring			
Init.	Precond.	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓
Off	$\bar{g}_t I$	30.03	0.869	0.064	0.260	29.57	0.865	0.068	0.195	27.39	0.784	0.087	0.327
Off	G_t	31.35	0.894	0.056	0.213	29.79	0.860	0.067	0.204	24.81	0.579	0.117	0.547
On	$\bar{g}_t I$	31.82	0.907	0.053	0.186	29.65	0.867	0.068	0.193	29.97	0.847	0.065	0.252
On	G_t	**34.27**	**0.952**	**0.040**	**0.083**	**29.97**	**0.873**	**0.066**	**0.183**	**30.35**	**0.854**	**0.062**	**0.219**
On	$\mathbf{0}$	32.04	0.907	0.052	0.150	24.27	0.491	0.124	0.396	13.25	0.109	0.440	0.746

See Appendix A.3 for further ablations on inner iteration counts and correction schedules.

5 Conclusion

We introduced LARC, a general late-stage corrector for diffusion inverse solvers. By combining measurement-guided initialization with preconditioned score refinement, LARC improves image details based on the structure established by the host solver. Experiments on FFHQ and ImageNet demonstrate that LARC consistently improves the tested host solvers with limited additional computation and outperforms competing baselines in reconstruction accuracy. Future work will investigate adaptive selection of the activation noise level to reduce manual tuning.

REFERENCES

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

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *Advances in Neural Information Processing Systems*, 2022.

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 *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794, 2021.

Per Christian Hansen. *Discrete Inverse Problems: Insight and Algorithms*. Society for Industrial and Applied Mathematics, 2010.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4396–4405, 2019.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Advances in Neural Information Processing Systems*, 2022.

Minseo Kim, Axel Levy, and Gordon Wetzstein. Dual ascent diffusion for inverse problems. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23622–23631, 2026a.

Minwoo Kim, Seunghyeok Shin, and Hongki Lim. FAST-DIPS: Adjoint-free analytic steps and hard-constrained likelihood correction for diffusion-prior inverse problems. *arXiv preprint arXiv:2603.01591*, 2026b.

Xiang Li, Soo Min Kwon, Ismail R. Alkhouri, Saiprasad Ravishankar, and Qing Qu. Decoupled data consistency with diffusion purification for image restoration. *arXiv preprint arXiv:2403.06054*, 2024.

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

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

Antonio Sclocchi, Alessandro Favero, and Matthieu Wyart. A phase transition in diffusion models reveals the hierarchical nature of data. *Proceedings of the National Academy of Sciences*, 122(1): e2408799121, 2025.

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

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

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

Binxu Wang and John J. Vastola. Diffusion models generate images like painters: an analytical theory of outline first, details later. *arXiv preprint arXiv:2303.02490*, 2023.

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

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.

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

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

Yuanzhi Zhu, Kai Zhang, Jingyun Liang, Jiezhang Cao, Bihan Wen, Radu Timofte, and Luc Van Gool. Denoising diffusion models for plug-and-play image restoration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1219–1229, 2023.

A APPENDIX

AI USE STATEMENT

We used generative AI tools to assist with drafting and polishing parts of the manuscript, translation, literature retrieval, reference checks, and feedback on methodological exposition. The authors reviewed all AI-assisted content and take responsibility for the final manuscript.

A.1 PROOFS

We fix t, y, u_t, and $\hat{u}_0(u_t)$ throughout each sequence of inner updates and use the local energy F_t defined in equation (10). We take $\sigma, \tau_t, \ell_t > 0$ and write $\|v\|_M^2 = v^\top M v$ for a symmetric positive definite matrix M. The two theorems additionally assume the exact-score relation and local smoothness stated in Section 3.2.

A.1.1 PROOF OF PROPOSITION 1

Proof. Differentiating the local energy gives

$$\nabla F_t(\widetilde{u}) = \frac{1}{\sigma^2} A^\top (A\widetilde{u} - y) + \frac{1}{\tau_t}(\widetilde{u} - \hat{u}_0(u_t)) + \frac{1}{\ell_t}(\widetilde{u} - u_t), \tag{22}$$

$$H_t := \nabla^2 F_t = \frac{1}{\sigma^2} A^\top A + \left(\frac{1}{\tau_t} + \frac{1}{\ell_t}\right) I = \tau_t^{-1} G_t^{-1}. \tag{23}$$

For any nonzero vector v,

$$v^\top H_t v = \frac{\|Av\|^2}{\sigma^2} + \left(\frac{1}{\tau_t} + \frac{1}{\ell_t}\right) \|v\|^2 > 0.$$

Thus $H_t \succ 0$, $G_t = (\tau_t H_t)^{-1} \succ 0$, and F_t has a unique minimizer. Setting the gradient to zero and multiplying by τ_t yields

$$G_t^{-1} \widetilde{u}_t^{(0)} = \frac{\tau_t}{\sigma^2} A^\top y + \frac{\tau_t}{\ell_t} u_t + \hat{u}_0(u_t),$$

which gives the initialization in equation (11). Since F_t is quadratic and $\nabla F_t(\widetilde{u}_t^{(0)}) = 0$, its exact expansion about the minimizer is

$$\begin{aligned} F_t(\widetilde{u}) - F_t(\widetilde{u}_t^{(0)}) &= \frac{1}{2}(\widetilde{u} - \widetilde{u}_t^{(0)})^\top H_t (\widetilde{u} - \widetilde{u}_t^{(0)}) \\ &= \frac{1}{2\tau_t} \|\widetilde{u} - \widetilde{u}_t^{(0)}\|_{G_t^{-1}}^2. \end{aligned} \tag{24}$$

This proves both the Hessian identity and the correction-energy identity in the proposition. □

A.1.2 PROOF OF THEOREM 1

Proof. Let $g_t^{(k)} = \nabla \Phi_t(\widetilde{u}_t^{(k)})$. The exact-score relation turns equation (14) into

$$\widetilde{u}_t^{(k+1)} - \widetilde{u}_t^{(k)} = -\eta_t G_t g_t^{(k)}.$$

The segment between consecutive iterates lies in the convex region on which $\nabla \Phi_t$ is L_t-Lipschitz, so the descent lemma gives

$$\begin{aligned} \Phi_t(\widetilde{u}_t^{(k+1)}) &\le \Phi_t(\widetilde{u}_t^{(k)}) - \eta_t \|g_t^{(k)}\|_{G_t}^2 + \frac{L_t \eta_t^2}{2} \|G_t g_t^{(k)}\|^2 \\ &\le \Phi_t(\widetilde{u}_t^{(k)}) - \eta_t \left(1 - \frac{L_t \eta_t \lambda_{\max}(G_t)}{2}\right) \|g_t^{(k)}\|_{G_t}^2. \end{aligned}$$

The second inequality uses $G_t^2 \preceq \lambda_{\max}(G_t) G_t$, which follows from the positive eigenvalues of G_t. Under equation (19), the parenthesized factor is $c_t > 0$, yielding equation (20). □

A.1.3 Proof of Theorem 2

Proof. Using the same notation, the fixed matrix G_t allows us to sum the inner updates as

$$\widetilde{u}_t^{(K)} - \widetilde{u}_t^{(0)} = -\eta_t G_t \sum_{k=0}^{K-1} g_t^{(k)}.$$

Substituting this identity into equation (24) and applying the Cauchy–Schwarz inequality in the G_t-weighted norm gives

$$\begin{aligned} F_t(\widetilde{u}_t^{(K)}) - F_t(\widetilde{u}_t^{(0)}) &= \frac{\eta_t^2}{2\tau_t} \left\| \sum_{k=0}^{K-1} g_t^{(k)} \right\|_{G_t}^2 \\ &\le \frac{K\eta_t^2}{2\tau_t} \sum_{k=0}^{K-1} \|g_t^{(k)}\|_{G_t}^2. \end{aligned} \tag{25}$$

Summing the bound in Theorem 1 over the same K updates yields

$$\Phi_t(\widetilde{u}_t^{(0)}) - \Phi_t(\widetilde{u}_t^{(K)}) \ge \eta_t c_t \sum_{k=0}^{K-1} \|g_t^{(k)}\|_{G_t}^2.$$

Combining the two inequalities proves the upper bound in equation (21). The lower bound follows because $\widetilde{u}_t^{(0)}$ minimizes F_t. The proof requires smoothness of Φ_t, but not convexity. □

Effect of output blending. For the prediction in equation (16), define

$$d_t^{\text{base}} = \hat{u}_0(u_t) - \widetilde{u}_t^{(0)}, \qquad d_t^{\text{ref}} = \widetilde{u}_t^{(K)} - \widetilde{u}_t^{(0)}.$$

Its displacement from initialization is $(1-\rho_t)d_t^{\text{base}} + \rho_t d_t^{\text{ref}}$. The quadratic identity in equation (13) therefore gives

$$F_t(\hat{u}_0^{\text{LARC}}(u_t)) - F_t(\widetilde{u}_t^{(0)}) = \frac{1}{2\tau_t} \left\| (1-\rho_t)d_t^{\text{base}} + \rho_t d_t^{\text{ref}} \right\|_{G_t^{-1}}^2. \tag{26}$$

By convexity of the squared norm, this implies

$$\begin{aligned} &F_t(\hat{u}_0^{\text{LARC}}(u_t)) - F_t(\widetilde{u}_t^{(0)}) \\ &\quad\le (1-\rho_t)\big[F_t(\hat{u}_0(u_t)) - F_t(\widetilde{u}_t^{(0)})\big] \\ &\qquad + \rho_t\big[F_t(\widetilde{u}_t^{(K)}) - F_t(\widetilde{u}_t^{(0)})\big]. \end{aligned}$$

Theorem 2 bounds the second energy gap. Anchoring at the original denoiser prediction requires retaining the first gap; pure ρ_t^2 scaling holds only if $d_t^{\text{base}} = 0$.

A.1.4 Equivalent PnP splitting formulation

At a fixed noise level t, separate the measurement and forward-consistency terms from the clean-prediction anchor by defining

$$\Psi_t(u) := \frac{1}{2\sigma^2}\|Au - y\|^2 + \frac{1}{2\ell_t}\|u - u_t\|^2, \qquad P_t(v) := \operatorname{prox}_{\tau_t \Psi_t}(v).$$

The quadratic split energy $\Psi_t(u) + \Phi_t(v) + \|u - v\|^2/(2\tau_t)$ motivates separating the explicit data operator from the score-based prior update, following the proximal and PnP frameworks (Parikh & Boyd, 2014; Venkatakrishnan et al., 2013). The data subproblem has the exact solution

$$P_t(v) = G_t\left(v + \frac{\tau_t}{\sigma^2}A^\top y + \frac{\tau_t}{\ell_t}u_t\right), \qquad DP_t(v) = G_t. \tag{27}$$

Initialize $v_t^{(0)} = \hat{u}_0(u_t)$ and consider the following PnP splitting iteration:

$$\begin{aligned} \widetilde{u}_t^{(k)} &= P_t(v_t^{(k)}), \\ v_t^{(k+1)} &= v_t^{(k)} + \eta_t s_{\theta,t}(\widetilde{u}_t^{(k)}), \qquad k = 0, \dots, K-1. \end{aligned} \tag{28}$$

The first line at $k = 0$ is exactly the initialization in equation (11). Since P_t is affine, applying it to the second line yields

$$\begin{aligned}\widetilde{u}_t^{(k+1)} - \widetilde{u}_t^{(k)} &= P_t(v_t^{(k+1)}) - P_t(v_t^{(k)}) \\ &= G_t(v_t^{(k+1)} - v_t^{(k)}) = \eta_t G_t s_{\theta,t}(\widetilde{u}_t^{(k)}).\end{aligned}$$

Thus eliminating $v_t^{(k)}$ recovers equation (14) exactly, for every finite K. Using the same blending and reverse step then gives the LARC algorithm in the main text. The prior update uses the specified score correction, not exact minimization over v. This equivalence holds for the iterations without treating the pretrained score as an exact proximal map.

A.2 Experimental settings

A.2.1 LARC parameters

We set $\tau_t = \alpha\lambda_t$, $\ell_t = \beta\lambda_t$, and $\eta_t = \gamma\lambda_t$. Task-specific parameters are listed below. In the parameter tables, N is the number of executed outer steps, $j \in \{1, \dots, N\}$ indexes these steps in execution order, and j_{start} is the first active correction step. This execution index is distinct from the noise-level index t used in the method. Each active step performs K inner score updates. DAPS can use multiple network evaluations per outer step, so N is not its NFE count. IP denotes inpainting with observed-pixel ratio r, and SR denotes super-resolution. For noiseless tasks, we use $\sigma = 0.001$ in super-resolution for numerical stability and solve inpainting using the $\sigma \to 0$ limit of the Gaussian data term.

Table 4: LARC (DiffPIR) parameters.

	FFHQ									ImageNet								
Task	N	j_{start}	K	α	β	γ	ρ	ζ	NFE	N	j_{start}	K	α	β	γ	ρ	ζ	NFE
IP ($r = 0.3$)	60	56	8	0.5	1.0	0.8	0.6	1.0	100	60	56	8	0.5	1.0	0.8	0.6	1.0	100
IP ($r = 0.5$)	60	56	8	0.5	1.0	0.8	0.6	1.0	100	60	56	8	0.5	1.0	0.8	0.6	1.0	100
IP ($r = 0.7$)	60	56	8	0.5	1.0	0.8	0.6	1.0	100	60	56	8	0.5	1.0	0.8	0.6	1.0	100
SR 2×	85	81	3	1.0	1.0	1.5	0.4	0.3	100	85	81	3	1.0	1.0	1.5	0.4	0.3	100
SR 4×	60	56	8	1.2	1.0	1.7	0.5	0.3	100	80	77	5	1.2	1.0	4.5	0.7	0.4	100
Gaussian	88	83	2	1.8	1.2	2.3	0.7	0.6	100	88	85	3	0.2	1.2	0.5	0.5	0.8	100
Motion	76	65	2	16.0	8.0	3.2	1.0	0.4	100	84	77	2	6.4	8.0	1.3	1.0	0.7	100
Box	80	71	2	0.5	1.0	0.4	0.4	1.0	100	–	–	–	–	–	–	–	–	–

Table 5: LARC (DPS) parameters.

	FFHQ									ImageNet								
Task	N	j_{start}	K	α	β	γ	ρ	ζ	NFE	N	j_{start}	K	α	β	γ	ρ	ζ	NFE
IP ($r = 0.3$)	370	351	4	1.0	1.0	1.5	0.8	0.0	450	360	351	8	1.0	1.0	2.3	1.0	0.7	440
IP ($r = 0.5$)	370	351	4	1.0	1.0	1.5	0.8	0.0	450	360	351	8	1.0	1.0	2.3	1.0	0.7	440
IP ($r = 0.7$)	370	351	4	1.0	1.0	1.5	0.8	0.0	450	360	351	8	1.0	1.0	2.3	1.0	0.7	440
SR 2×	220	201	4	1.0	1.0	1.5	0.8	0.0	300	360	351	8	1.0	1.0	0.8	1.0	0.7	440
SR 4×	210	201	4	0.6	1.0	0.4	0.8	0.3	250	210	201	4	1.2	1.0	1.0	1.0	0.7	250
Gaussian	210	201	2	0.9	1.2	0.6	1.0	0.7	230	560	551	2	0.1	1.2	0.1	0.2	0.3	580
Motion	360	351	2	32.0	8.0	6.3	0.8	0.3	380	210	201	4	12.8	8.0	1.3	0.8	0.3	250
Box	370	351	4	1.0	1.0	2.3	0.5	0.0	450	–	–	–	–	–	–	–	–	–

Table 6: LARC (DAPS) parameters.

	FFHQ									ImageNet								
Task	N	j_{start}	K	α	β	γ	ρ	ζ	NFE	N	j_{start}	K	α	β	γ	ρ	ζ	NFE
IP ($r = 0.3$)	60	31	4	0.5	1.0	0.8	1.0	0.7	300	80	51	4	0.5	1.0	1.1	1.0	1.0	400
IP ($r = 0.5$)	70	41	2	0.5	1.0	0.8	1.0	1.0	290	60	41	4	0.5	1.0	1.1	1.0	1.0	300
IP ($r = 0.7$)	70	41	2	0.5	1.0	0.8	1.0	1.0	290	60	41	4	0.5	1.0	1.1	1.0	1.0	300
SR 2×	70	31	2	8.0	4.0	3.0	0.7	0.4	270	70	41	2	8.0	4.0	3.0	1.0	1.0	290
SR 4×	50	31	2	4.6	4.0	1.7	0.7	0.4	210	50	31	4	2.4	2.0	2.0	1.0	0.7	250
Gaussian	30	11	2	14.4	4.8	4.5	0.7	0.4	110	50	41	8	3.8	9.6	2.3	1.0	0.0	290
Motion	70	41	2	16.0	8.0	3.2	0.6	0.7	290	50	21	4	25.6	8.0	3.0	1.0	0.7	250
Box	190	171	2	0.5	1.0	0.4	0.4	1.0	910	–	–	–	–	–	–	–	–	–

Table 7: LARC (FAST-DIPS) parameters.

	FFHQ									ImageNet								
Task	N	j_{start}	K	α	β	γ	ρ	ζ	NFE	N	j_{start}	K	α	β	γ	ρ	ζ	NFE
IP ($r = 0.3$)	180	176	8	0.5	1.0	1.1	0.6	1.0	220	185	176	8	0.5	1.0	1.1	0.6	1.0	265
IP ($r = 0.5$)	180	176	8	0.5	1.0	1.1	0.6	1.0	220	185	176	8	0.5	1.0	1.1	0.6	1.0	265
IP ($r = 0.7$)	180	176	8	0.5	1.0	1.1	0.6	1.0	220	185	176	8	0.5	1.0	1.1	0.6	1.0	265
SR 2×	180	176	8	1.0	1.0	0.9	0.5	0.3	220	120	101	2	1.0	1.0	0.9	0.8	0.3	160
SR 4×	180	176	8	1.2	1.0	1.0	0.5	0.3	220	120	101	2	1.2	1.0	1.2	0.8	0.3	160
Gaussian	90	86	2	1.8	1.2	2.3	1.0	0.7	100	45	41	4	0.5	2.4	0.5	1.0	0.7	65
Motion	110	101	2	16.0	8.0	4.7	0.6	0.7	130	45	41	4	6.4	8.0	1.9	1.0	0.0	65
Box	230	226	2	0.5	1.0	1.1	0.4	0.7	240	–	–	–	–	–	–	–	–	–

A.3 Schedule ablations

A.3.1 Schedules under a fixed NFE budget

We compare LARC (DiffPIR) schedules on 100 FFHQ inpainting images with observed-pixel ratio $r = 0.3$, using shared masks and a fixed budget of 100 network evaluations. Here N is the total number of outer sampling steps, M is the number of late steps with LARC enabled, and K is the number of inner score iterations per active step. Each outer step uses one network evaluation and each inner iteration adds one, so $\text{NFE} = N + MK = 100$. We vary M and K and set $N = 100 - MK$ to compare budget allocations. Figure 5 shows that concentrating refinement at fewer late steps performs better in this sweep, with the highest PSNR at $(N, M, K) = (60, 5, 8)$.

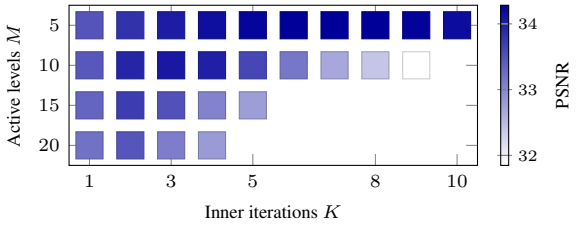

Figure 5: Inpainting PSNR under a fixed budget of 100 NFE.

A.3.2 Number of inner iterations

Using the same LARC (DiffPIR) schedule, we study how varying the number of inner iterations K affects restoration quality under a fixed outer schedule. Overall, the PSNR curves of these tasks exhibit an inverted-U shape with respect to K. Inpainting and super-resolution tend to peak at $K = 8$, while motion deblurring peaks at $K = 2$. This indicates that K typically reaches its performance peak within a limited number of iterations.

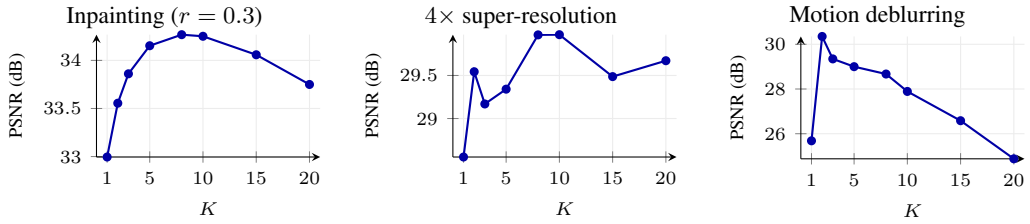

Figure 6: PSNR versus K on 100 FFHQ images, with fixed outer schedules.

A.3.3 Correction placement

For this placement ablation, we move LARC from its default terminal window to an early or middle window, using FAST-DIPS outside the correction window. We study where to insert LARC into FAST-DIPS on 100 FFHQ images per task, covering inpainting ($r = 0.3$), $4\times$ super-resolution, and motion deblurring. We place a fixed-length correction window at the beginning, after half of the total NFE budget has been consumed, or at the end of the trajectory. The window length, inner iterations, outer noise schedule, and total NFE remain unchanged across positions, with FAST-DIPS used outside the window. The budgets are 220 NFE for inpainting and super-resolution, and 130 NFE for motion deblurring.

Late-stage insertion achieves the best reconstruction quality across all three tasks. Early and middle insertion yield nearly identical final results under these schedules. We also compare against LARC applied throughout sampling without FAST-DIPS, adjusting the outer-step budget to match the total NFE. Late-stage insertion outperforms this standalone variant on all three tasks, supporting the use of LARC as a local corrector following the host solver's early trajectory.

Table 8: Correction placement on 100 FFHQ images per task. Middle insertion starts after half of the total NFE budget. Standalone LARC uses no FAST-DIPS steps. Seven decimal places distinguish the nearly identical early and middle means. Best values within each task are bold.

Placement / variant	PSNR ↑	SSIM ↑	NRMSE ↓	LPIPS ↓
Inpainting ($r = 0.3$), 220 NFE				
Early	33.2609450	0.9291565	0.0449537	0.1257478
Middle	33.2609215	0.9291564	0.0449538	0.1257468
Late	**34.7306883**	**0.9555151**	**0.0383523**	**0.0814711**
Standalone LARC	32.1472468	0.9154050	0.0512389	0.1527492
$4\times$ super-resolution, 220 NFE				
Early	30.6573235	0.8824856	0.0606332	0.2171480
Middle	30.6573557	0.8824860	0.0606331	0.2171494
Late	**30.7504842**	**0.8870674**	**0.0600092**	**0.2117187**
Standalone LARC	29.9411827	0.8675352	0.0655071	0.2131550
Motion deblurring, 130 NFE				
Early	27.0550214	0.6655927	0.0902317	0.3790685
Middle	27.0550290	0.6655922	0.0902316	0.3790648
Late	**30.7657958**	**0.8663863**	**0.0596213**	**0.2483248**
Standalone LARC	30.3623947	0.8328613	0.0623014	0.2631934

A.4 Additional qualitative comparisons

Figures 7–10 compare each solver with its LARC variant on selected examples from the same 100-image evaluations as Table 1. Examples illustrate improvements and are distinct within each dataset; aggregate results are reported in the main table. Each panel shows GT, the observation, the original solver, and LARC, with mean RGB absolute-error maps directly below. Error maps share a $[0, 0.2]$ scale (brighter means larger error).

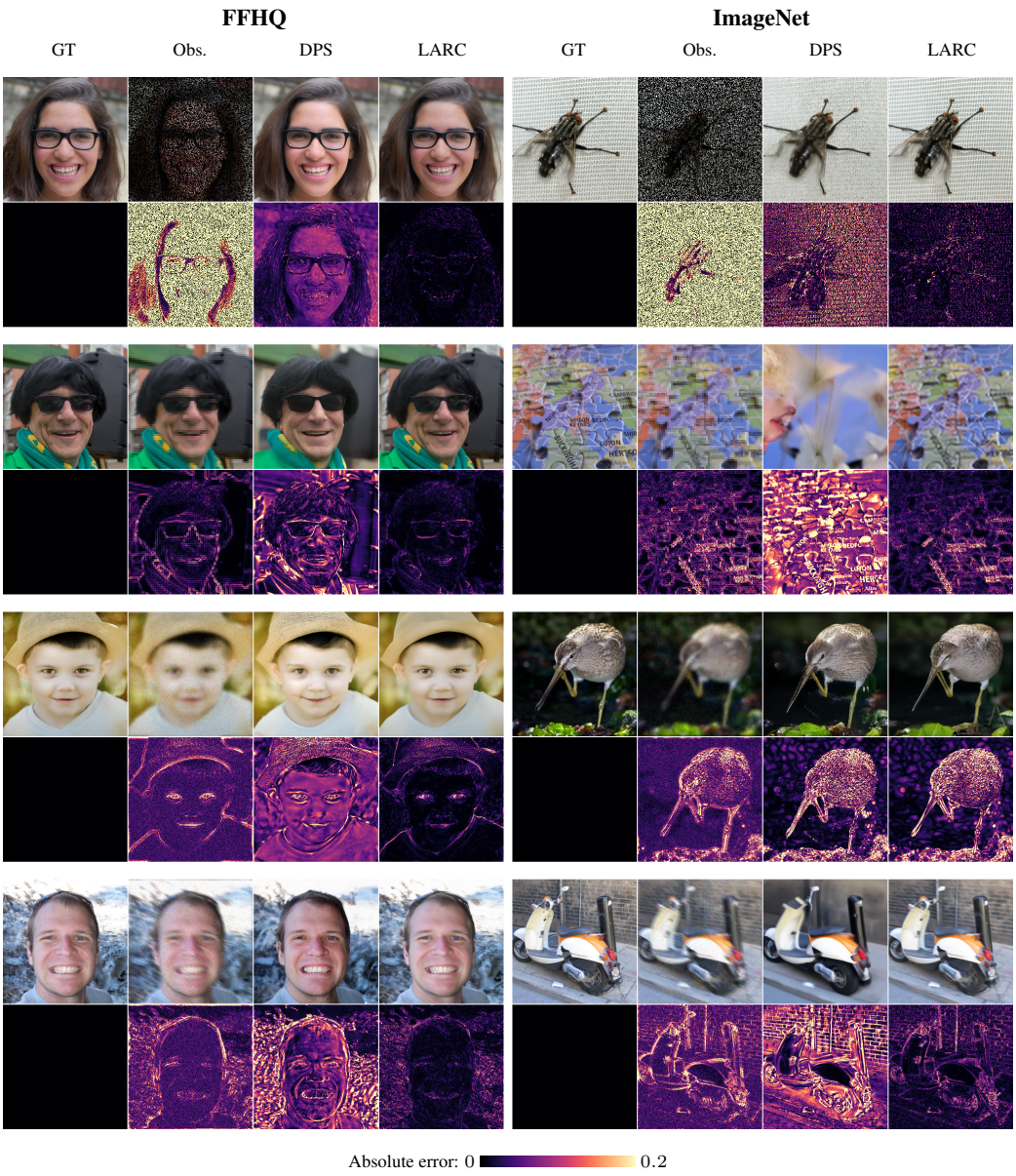

Figure 7: DPS and its LARC variant. Left: FFHQ; right: ImageNet. Rows from top to bottom: inpainting ($r = 0.3$), $4\times$ super-resolution, Gaussian deblurring, and motion deblurring.

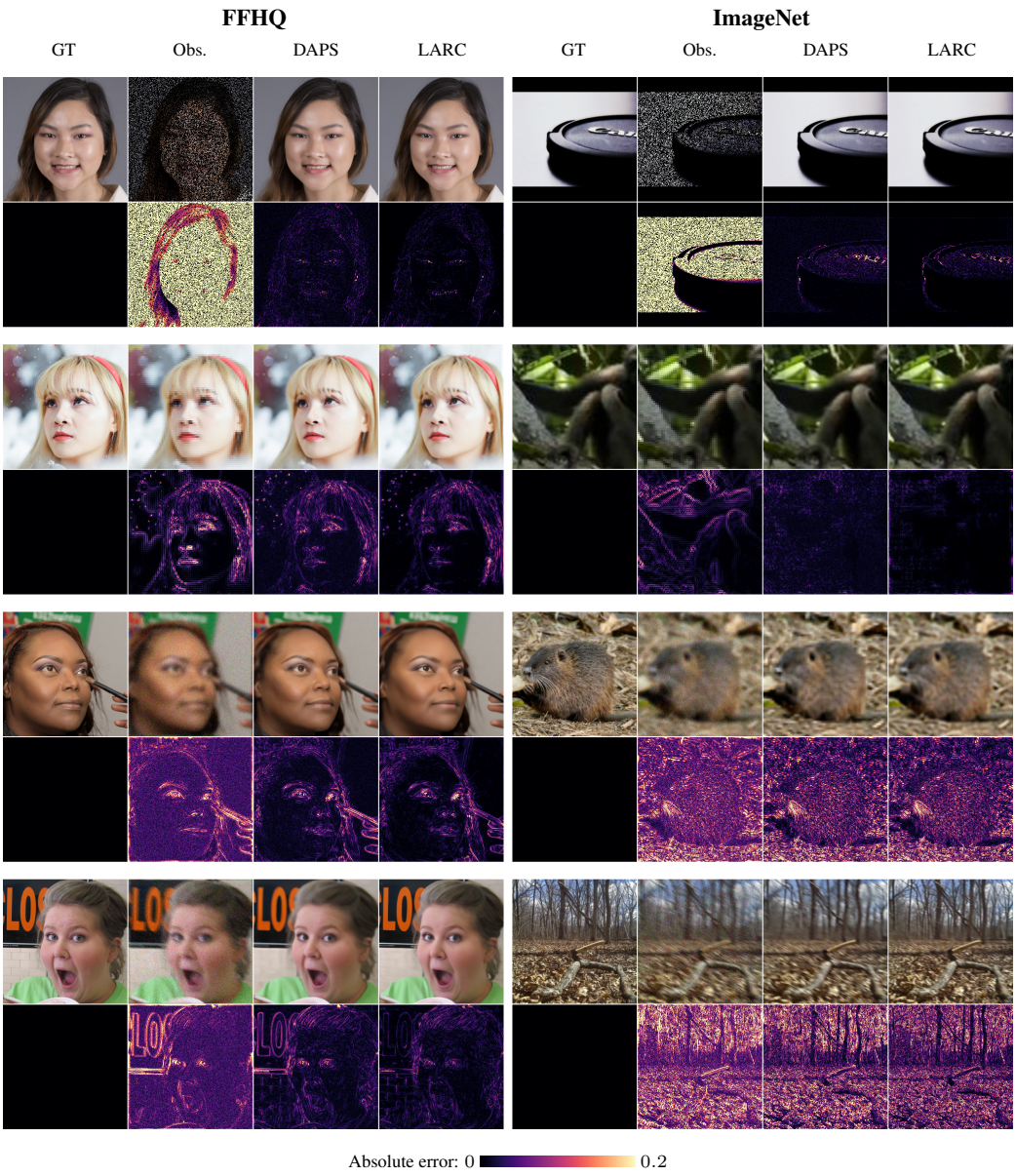

Figure 8: DAPS and its LARC variant. Left: FFHQ; right: ImageNet. Rows from top to bottom: inpainting ($r = 0.3$), $4\times$ super-resolution, Gaussian deblurring, and motion deblurring.

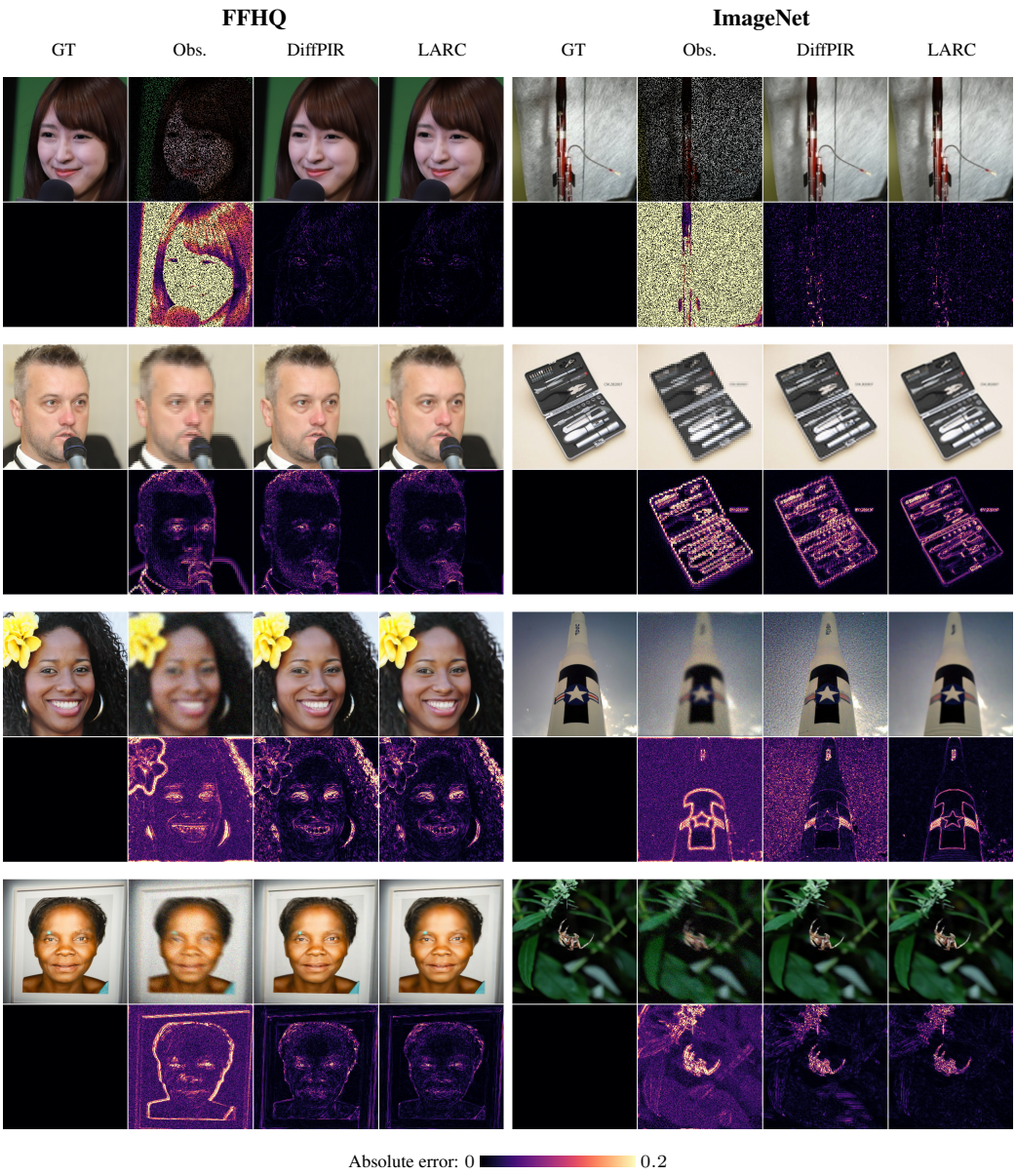

Figure 9: DiffPIR and its LARC variant. Left: FFHQ; right: ImageNet. Rows from top to bottom: inpainting ($r = 0.3$), $4\times$ super-resolution, Gaussian deblurring, and motion deblurring.

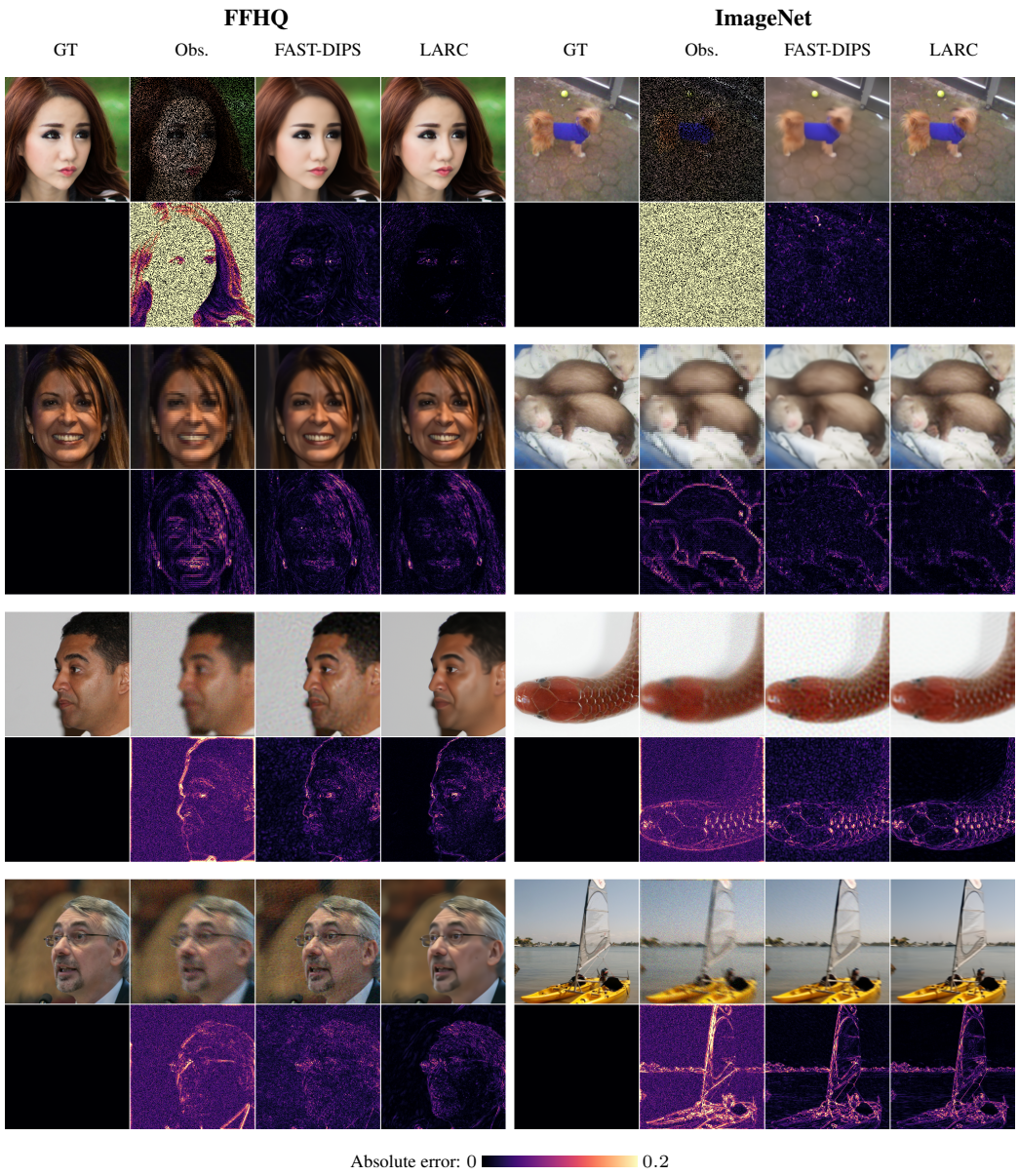

Figure 10: FAST-DIPS and its LARC variant. Left: FFHQ; right: ImageNet. Rows from top to bottom: inpainting ($r = 0.3$), $4\times$ super-resolution, Gaussian deblurring, and motion deblurring.