HIGH-ORDER DIFFUSION SOLVERS FOR LINEAR INVERSE PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion models (DMs) are effective generative priors but typically require hundreds to thousands of sampling steps. Recent high-order solvers accelerate unconditional generation by exploiting the semilinear structure of diffusion ordinary differential equations (ODEs) and stochastic differential equations (SDEs), integrating linear terms exactly and reducing sampling to 5 to 20 steps while maintaining generation quality. DMs have also become effective generative priors for inverse problems, where measurement conditioning introduces an intractable measurement matching score. Efficient diffusion inverse solvers typically approximate the measurement matching score in various forms. However, to our knowledge, none of these methods can effectively exploit the semilinear structure of the corresponding differential equations to derive conditional high-order solvers. To address this issue, we identify a class of inverse solvers with structured measurement matching scores that incorporate the model prediction while retaining semilinearity. However, the corresponding integral equations contain dense matrices in the linear terms, making direct computation inefficient. We utilize the singular value decomposition of the forward operator to diagonalize the dense matrices in the spectral domain, enabling efficient computation of the linear terms and high-order approximation of the remaining nonlinear terms. We further introduce guidance strength and stochasticity to control measurement influence and sampling noise, respectively. Experiments on CelebA and ImageNet demonstrate that our method achieves improved reconstruction quality over baselines across linear inverse problems with 5 to 10 sampling steps.

1 INTRODUCTION

Linear inverse problems arise in a wide range of applications, including image deblurring (Tran et al., 2021), super-resolution (Park et al., 2003), inpainting (Elharrouss et al., 2020), and compressed sensing (Wang & Zhao, 2016). Given corrupted measurements

$$\mathbf{y} = \mathbf{A}\mathbf{x}_0 + \boldsymbol{\epsilon}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{M\times N}$ is the forward operator and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_y^2\mathbf{I}_M)$, the objective is to recover the unknown signal $\mathbf{x}_0 \in \mathbb{R}^N$. Linear inverse problems are typically ill-posed, particularly when $M \ll N$, making prior information essential for reconstruction. Several approaches impose structural priors such as sparsity (Chen et al., 2013; Li, 2013; Nguyen & Tran, 2013; Xu et al., 2012; Nguyen & Tran, 2012), while learned generative models capture data distributions utilizing variational autoencoders (VAEs) (Kingma & Welling, 2014) and generative adversarial networks (GANs) (Goodfellow et al., 2014). More recently, diffusion models (DMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020) have become effective generative priors for inverse problems. However, diffusion sampling typically requires hundreds to thousands of steps, resulting in high computational cost for DM-based inverse solvers.

DDPM (Ho et al., 2020) generates samples through a Markovian reverse diffusion process that progressively removes noise over successive sampling steps. DDIM (Song et al., 2021a) introduces non-Markovian sampling processes that enable generation with fewer steps and has been adopted by inverse solvers such as DDRM (Kawar et al., 2022), DDNM (Wang et al., 2023), and DiffPIR (Zhu et al., 2023), which typically require 20 to 100 neural function evaluations (NFEs) to maintain

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.

reconstruction quality. For unconditional generation, recent high-order diffusion solvers, including DPM-Solver (Lu et al., 2022), DPM-Solver++ (Lu et al., 2025), UniPC (Zhao et al., 2023), and SA-Solver (Xue et al., 2023), reduce sampling to 5 to 20 steps by formulating diffusion ordinary differential equations (ODEs) or stochastic differential equations (SDEs) as integral equations. The solvers exploit the semilinear structure of diffusion ODEs and SDEs to integrate linear terms exactly and apply high-order approximation to the neural prediction, enabling high-quality generation within this sampling range.

Extending high-order diffusion solvers to inverse problems is challenging, as measurement conditioning introduces an additional measurement matching score into the conditional ODEs and SDEs. This score is generally intractable, and efficient diffusion inverse solvers therefore rely on approximations with different structures. Approximations that exclude the model prediction yield simple linear corrections but can be less effective with 5 to 10 sampling steps, whereas those that incorporate the model prediction through nonlinear operations make it difficult for high-order diffusion solvers to exploit the semilinear structure.

To address this issue, we identify a class of efficient inverse solvers with structured measurement matching scores that incorporate the model prediction while retaining semilinearity. However, the corresponding integral equations contain dense matrices in the linear terms, making direct computation inefficient. We utilize the singular value decomposition (SVD) of the forward operator to diagonalize the dense matrices in the spectral domain, enabling efficient computation of the linear terms and high-order approximation of the remaining nonlinear terms. Based on the spectral formulation, we develop high-order solvers for conditional ODE and SDE sampling and introduce guidance strength and stochasticity to control measurement influence and sampling noise, respectively. Experiments on CelebA and ImageNet demonstrate improved reconstruction quality over baselines across linear inverse problems with 5 to 10 sampling steps.

1.1 Related work

We review related work on high-order diffusion solvers for unconditional generation, DM-based solvers for inverse problems, and acceleration techniques for DM-based inverse solvers.

High-order diffusion solvers for unconditional DMs. High-order diffusion solvers reduce NFEs by numerically integrating diffusion ODEs or SDEs. DPM-Solver (Lu et al., 2022) integrates the linear terms of the diffusion ODE exactly and applies high-order approximation to the nonlinear neural prediction. DPM-Solver++ (Lu et al., 2025) extends high-order sampling to data-prediction models, while DEIS (Zhang & Chen, 2023) derives exponential-integrator updates for diffusion ODEs. UniPC (Zhao et al., 2023) constructs predictor-corrector updates utilizing model evaluations from previous steps. For diffusion SDEs, SA-Solver (Xue et al., 2023) develops high-order solvers with controllable stochasticity. The high-order solvers above consider unconditional diffusion sampling, whereas inverse problems introduce an additional measurement matching score into the conditional ODEs and SDEs. Our work identifies measurement matching scores whose structure allows the additional measurement-related terms required for high-order sampling to be computed efficiently.

DM-based solvers for inverse problems. DM-based inverse solvers incorporate measurements into reverse sampling through different forms of measurement matching. Score-ALD (Jalal et al., 2021), MCG (Chung et al., 2022a), and SNIPS (Kawar et al., 2021) incorporate measurement constraints into Langevin-based sampling. DPS (Chung et al., 2023), DMPS (Meng & Kabashima, 2025), and DAPS (Zhang et al., 2025a) construct different measurement corrections during reverse sampling. For linear inverse problems, DDRM (Kawar et al., 2022) performs spectral-domain updates, while DDNM (Wang et al., 2023) uses a range–null-space decomposition. DiffPIR (Zhu et al., 2023) combines diffusion denoising with data-consistency updates, while DDS (Chung et al., 2024) solves the corresponding optimization problem using conjugate gradient (CG). Other formulations include pseudo-inverse guidance in ΠGDM (Song et al., 2023) and variational inference in RED-diff (Mardani et al., 2024). Latent diffusion methods, including ReSample (Song et al., 2024), PSLD (Rout et al., 2023), and Flash-Diffusion (Fabian et al., 2024), perform inverse reconstruction utilizing latent representations. Such methods require transformations between latent and pixel spaces, which can introduce additional reconstruction error and computational cost (Song et al.,

2024). We focus on pixel-space DMs and measurement matching scores that incorporate the model prediction while allowing efficient computation of the linear terms required by high-order solvers.

Acceleration techniques for DM-based inverse solvers. Recent studies reduce the sampling cost of DM-based inverse solvers through different mechanisms. SSD (Liu et al., 2024) introduces shortcut sampling and typically uses 30 to 100 NFEs. We compare SSD in the main experiments. CCDF (Chung et al., 2022b) shortens reverse sampling through measurement-dependent initialization and is developed for specific forward operators under noiseless measurements. C-ΠGDM (Pandey et al., 2024) uses conjugate integration for few-step inverse sampling, with the reported noisy-task results requiring more sampling steps than the 5 to 10 NFEs considered in our experiments. Recent work also reports difficulty in obtaining plausible C-ΠGDM reconstructions for random inpainting after several hyperparameter tuning (Pandey et al., 2025). We therefore provide comparisons with CCDF and C-ΠGDM in the appendix. LLE (Zhang et al., 2025b) uses historical model predictions through learned linear extrapolation and requires additional training. We also compare with LLE-based inverse solvers in the appendix, while our solvers directly utilize pretrained DMs without additional training.

1.2 Contributions

Our contributions are summarized as follows:

- We identify a class of measurement matching scores underlying DDNM (Wang et al., 2023) and DiffPIR (Zhu et al., 2023) that incorporate the model prediction while retaining the semilinear structure of the conditional diffusion ODEs and SDEs. The corresponding integral equations contain dense matrices in the linear terms. We show that the SVD of the forward operator diagonalizes the dense matrices in the spectral domain, enabling efficient computation of the linear terms while high-order approximation is applied only to the neural prediction.
- We introduce guidance strength and stochasticity to control measurement influence and sampling noise, respectively, and formulate the corresponding conditional SDE. We derive an integral representation that supports high-order conditional ODE and SDE sampling within the corresponding formulation.
- Experiments on CelebA and ImageNet across linear inverse problems demonstrate our method achieves improved reconstruction quality over baselines with only 5 to 10 NFEs.

2 Background

We first review diffusion ODEs and SDEs and the corresponding reverse sampling processes. We then formulate linear inverse problems and introduce posterior sampling with measurement matching scores in DM-based inverse solvers.

2.1 Diffusion models

DMs generate samples by reversing a forward process that progressively perturbs data with noise. Given $\mathbf{x}_0 \sim p_{\mathrm{data}}$, the forward process over $t \in [0, T]$ is defined by

$$p_{0t}(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \alpha_t \mathbf{x}_0, \sigma_t^2 \mathbf{I}_N), \tag{2}$$

where α_t and σ_t are non-negative differentiable functions and the signal-to-noise ratio (SNR) α_t^2/σ_t^2 decreases with t. Let p_t denote the marginal distribution of $\mathbf{x}_t$, with p_T approaching a standard Gaussian distribution. The forward process can be represented by the SDE (Song et al., 2021b)

$$\mathrm{d}\mathbf{x}_t = f(t)\mathbf{x}_t \mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t, \tag{3}$$

where $\mathbf{w}_t$ is a standard Wiener process. The drift and diffusion coefficients satisfy $f(t) = \mathrm{d}\log(\alpha_t)/\mathrm{d}t$ and $g^2(t) = -2\sigma_t^2 \mathrm{d}\lambda_t/\mathrm{d}t$, where $\lambda_t := \log(\alpha_t/\sigma_t)$ denotes the half-log SNR.

Sampling starts from $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ and follows the reverse-time SDE (Song et al., 2021b)

$$\mathrm{d}\mathbf{x}_t = \big(f(t)\mathbf{x}_t - g^2(t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\big)\,\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t, \tag{4}$$

where $\bar{\mathbf{w}}_t$ is a reverse-time Wiener process and $\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$ is the score function. The associated probability-flow ODE is

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = f(t)\mathbf{x}_t - \frac{1}{2} g^2(t) \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t). \tag{5}$$

The reverse-time SDE and probability-flow ODE have identical marginal distributions p_t as the forward SDE (Song et al., 2021b).

The score is approximated by a neural network $\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ trained utilizing denoising score matching (Song & Ermon, 2019; Song et al., 2021b). The score network can be parameterized through a data-prediction network $\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ as (Kingma et al., 2021; Salimans & Ho, 2022)

$$\mathbf{s}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \frac{\alpha_t \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) - \mathbf{x}_t}{\sigma_t^2}. \tag{6}$$

Tweedie's formula (Efron, 2011) gives

$$\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t] = \frac{\mathbf{x}_t + \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)}{\alpha_t}, \tag{7}$$

which relates the data prediction $\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ to the posterior mean $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]$.

2.2 Diffusion models for linear inverse problems

For linear inverse problems, diffusion sampling targets the posterior distribution $p(\mathbf{x}_0|\mathbf{y})$, where $\mathbf{y} \in \mathbb{R}^M$ denotes the measurements in Eq. (1). The conditional reverse-time SDE is

$$\mathrm{d}\mathbf{x}_t = \left(f(t)\mathbf{x}_t - g^2(t)\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})\right) \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t. \tag{8}$$

By Bayes' rule, the conditional score decomposes as

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y}) = \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t), \tag{9}$$

where the first term is the unconditional score and the second term is the measurement matching score that incorporates the measurements into conditional sampling.

The conditional Tweedie's formula (Daras et al., 2024) gives

$$\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] = \frac{\mathbf{x}_t + \sigma_t^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y})}{\alpha_t}. \tag{10}$$

Combining Eqs. (7), (9), and (10) yields

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) = \frac{\alpha_t}{\sigma_t^2} \left(\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] - \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]\right). \tag{11}$$

Equation (11) relates the measurement matching score to the difference between the conditional and unconditional posterior means. $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]$ is approximated in practice by the model prediction $\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$, and DM-based inverse solvers approximate $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}]$ (Daras et al., 2024).

DiffPIR (Zhu et al., 2023) and DDS (Chung et al., 2024) approximate the conditional posterior mean through the following data-consistency problem (Daras et al., 2024)

$$\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] \approx \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{\rho_t}{2}\|\mathbf{x} - \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]\|_2^2, \tag{12}$$

where $\rho_t > 0$ controls the regularization strength. DDS approximately solves Eq. (12) using a few conjugate-gradient (CG) steps with a fixed regularization parameter, whereas DiffPIR uses the time-dependent schedule $\rho_t = \lambda \sigma_y^2 \alpha_t^2 / \sigma_t^2$, where $\lambda > 0$ controls the regularization strength. In this work, we use the DiffPIR formulation to approximate the measurement matching score. For the linear forward operator considered here, Eq. (12) admits the closed-form solution

$$\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] \approx \left(\mathbf{A}^\top \mathbf{A} + \rho_t \mathbf{I}_N\right)^{-1} \left(\mathbf{A}^\top \mathbf{y} + \rho_t \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]\right). \tag{13}$$

DDNM (Wang et al., 2023) uses a range–null-space decomposition to approximate the conditional posterior mean (Daras et al., 2024)

$$\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] \approx \left(\mathbf{I}_N - \boldsymbol{\Psi}_t \mathbf{A}^\dagger \mathbf{A}\right) \mathbb{E}[\mathbf{x}_0|\mathbf{x}_t] + \boldsymbol{\Psi}_t \mathbf{A}^\dagger \mathbf{y}, \tag{14}$$

where $\mathbf{A}^\dagger$ denotes the Moore–Penrose pseudoinverse of $\mathbf{A}$ and $\boldsymbol{\Psi}_t \in \mathbb{R}^{N \times N}$ is a time-dependent weighting operator defined in the spectral basis of the forward operator.

Although DiffPIR and DDNM construct the conditional posterior mean differently, both approximations have a structured dependence on the unconditional posterior mean. Section 3 uses the shared structure to formulate measurement matching scores for high-order conditional sampling.

3 METHOD

We develop high-order solvers for DM-based linear inverse problems in four stages. We first formulate a class of structured measurement matching scores, then utilize the SVD of the forward operator to derive the conditional ODEs in the spectral domain, enabling efficient computation of the linear terms. We further extend the formulation to guided conditional SDEs with controllable guidance strength and stochasticity, and finally construct high-order discretizations by approximating the remaining nonlinear terms.

3.1 STRUCTURED MEASUREMENT MATCHING SCORE

Utilizing the model prediction $\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ to approximate $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]$ in Eq. (11) gives

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx \frac{\alpha_t}{\sigma_t^2} \left(\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}] - \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right), \tag{15}$$

where the measurement matching score depends on the approximation of the conditional posterior mean $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t, \mathbf{y}]$. For few-step high-order sampling, a main challenge is to identify measurement matching score approximations that are able to exploit semi-linearity in the corresponding differential equations, allowing efficient computation of the linear terms introduced by measurement conditioning while applying high-order approximation to the nonlinear model prediction. Recent efficient diffusion inverse solvers approximate the measurement matching score in various forms. SNIPS and DMPS exclude the model prediction, yielding simple linear corrections but potentially limiting reconstruction quality with fewer sampling steps. DPS and ΠGDM incorporate the model prediction through nonlinear operations, including Jacobian-related computations, making it difficult for high-order diffusion solvers to exploit the semilinear structure. DDRM uses componentwise updates, which do not directly produce the continuous measurement matching score adopted in our formulation. We identify that DDNM and DiffPIR incorporate the model prediction through linear corrections capable of exploiting the semi-linear structure of the corresponding differential equations. Substituting the corresponding posterior-mean approximations in Eqs. (13) and (14) into Eq. (15) reveals the shared form

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx \frac{\alpha_t}{\sigma_t^2} \mathbf{M}_t \big(\mathbf{y} - \mathbf{A}\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\big), \tag{16}$$

where $\mathbf{M}_t \in \mathbb{R}^{N\times M}$ depends on the inverse solver. For DiffPIR, Eq. (13) gives $\mathbf{M}_t = (\mathbf{A}^\top\mathbf{A} + \rho_t\mathbf{I}_N)^{-1}\mathbf{A}^\top$. For DDNM, Eq. (14) gives $\mathbf{M}_t = \boldsymbol{\Psi}_t\mathbf{A}^\dagger$. The shared form in Eq. (16) incorporates the model prediction through a linear form and retains the semi-linear structure exploited by high-order diffusion solvers. However, the corresponding conditional diffusion ODE contains dense matrix operations in the linear terms, making direct computation inefficient. Section 3.2 shows how the SVD of the forward operator diagonalizes the corresponding matrices for both choices of $\mathbf{M}_t$, allowing the linear terms to be computed efficiently in the spectral domain.

We first formulate the conditional reverse process under deterministic sampling:

$$\mathrm{d}\mathbf{x}_t = \Big(f(t)\mathbf{x}_t - \frac{1}{2}g^2(t)\big(\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)\big)\Big)\mathrm{d}t. \tag{17}$$

3.2 CONDITIONAL ODES IN SPECTRAL DOMAINS

Substituting Eq. (16) and the unconditional score approximation in Eq. (6) into Eq. (17) gives

$$\mathrm{d}\mathbf{x}_t = \left(\Big(f(t) + \frac{g^2(t)}{2\sigma_t^2}\Big)\mathbf{x}_t - \frac{\alpha_t g^2(t)}{2\sigma_t^2}\big(\mathbf{S}_t\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \mathbf{g}_t\big)\right)\mathrm{d}t, \tag{18}$$

where $\mathbf{S}_t := \mathbf{I}_N - \mathbf{M}_t\mathbf{A}$ and $\mathbf{g}_t := \mathbf{M}_t\mathbf{y}$. For fixed measurements $\mathbf{y}$, $\mathbf{g}_t$ depends on time and can be precomputed.

In Eq. (18), the matrix $\mathbf{S}_t$ is generally dense, making direct computation of the linear terms in the corresponding integral equation inefficient. We therefore utilize the SVD of the forward operator to diagonalize the dense matrix in the spectral domain, enabling efficient computation of the linear terms.

Specifically, let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be the SVD of the forward operator, where $\mathbf{U} \in \mathbb{R}^{M\times M}$ and $\mathbf{V} \in \mathbb{R}^{N\times N}$ are orthogonal matrices, and $\mathbf{\Sigma} = [\mathrm{Diag}(s_1, \ldots, s_M), \mathbf{0}_{M\times(N-M)}] \in \mathbb{R}^{M\times N}$ is a rectangular diagonal matrix of singular values $s_1, \ldots, s_M$. Applying the SVD to Eq. (18) gives the following spectral form.

Proposition 1 (Spectral form of the conditional ODE)**.** *Define* $\tilde{\mathbf{S}}_t := \mathbf{V}^\top \mathbf{S}_t \mathbf{V} \in \mathbb{R}^{N\times N}$, $\tilde{\mathbf{g}}_t := \mathbf{V}^\top \mathbf{g}_t \in \mathbb{R}^N$, $\tilde{\mathbf{x}}_t := \mathbf{V}^\top \mathbf{x}_t \in \mathbb{R}^N$, *and* $\tilde{\mathbf{y}} := \mathbf{U}^\top \mathbf{y} \in \mathbb{R}^M$. *Let* $\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) := \mathbf{V}^\top \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{V}\tilde{\mathbf{x}}_t, t)$. *Then Eq.* (18) *can be expressed as*

$$\mathrm{d}\tilde{\mathbf{x}}_t = \left(\Big(f(t) + \frac{g^2(t)}{2\sigma_t^2} \Big) \tilde{\mathbf{x}}_t - \frac{\alpha_t g^2(t)}{2\sigma_t^2} \big(\tilde{\mathbf{S}}_t \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) + \tilde{\mathbf{g}}_t \big) \right) \mathrm{d}t. \tag{19}$$

For the $\mathbf{M}_t$ *corresponding to DiffPIR and DDNM,* $\tilde{\mathbf{S}}_t$ *is diagonal.*

Proposition 1 shows that the SVD of the forward operator diagonalizes $\mathbf{S}_t$ for DiffPIR and DDNM, enabling efficient coordinate-wise computation of the linear terms in the spectral domain. The proof and the explicit forms of $\tilde{\mathbf{S}}_t$ and $\tilde{\mathbf{g}}_t$ are provided in Appendix C.1.

We next derive the corresponding integral representation, where the linear terms can be computed utilizing the diagonal coefficients and high-order approximation is applied to the nonlinear model prediction. We reparameterize the transformed ODE utilizing the half-log SNR $\lambda_t := \log(\alpha_t/\sigma_t)$, which decreases monotonically with t and has inverse $t(\lambda)$. We define $\bar{\mathbf{x}}_\lambda := \tilde{\mathbf{x}}_{t(\lambda)}$, $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\cdot, \lambda) := \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\cdot, t(\lambda))$, $\bar{\mathbf{S}}_\lambda := \tilde{\mathbf{S}}_{t(\lambda)}$, and $\bar{\mathbf{g}}_\lambda := \tilde{\mathbf{g}}_{t(\lambda)}$. This gives the following integral representation.

Proposition 2 (Integral representation of the transformed ODE)**.** *Given* $\tilde{\mathbf{x}}_s$ *at time* s*, the solution to Eq.* (19) *at* $t \in [0, s]$ *admits the integral representation*

$$\tilde{\mathbf{x}}_t = \frac{\sigma_t}{\sigma_s} \tilde{\mathbf{x}}_s + \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda} \left(\bar{\mathbf{S}}_\lambda \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \bar{\mathbf{g}}_\lambda \right) \mathrm{d}\lambda. \tag{20}$$

In Proposition 2, the first term gives a closed-form transition from $\tilde{\mathbf{x}}_s$. The diagonal structure of $\bar{\mathbf{S}}_\lambda$ allows the linear terms in the integral to be computed coordinate-wise, while high-order approximation is applied only to the nonlinear model prediction. The proof is provided in Appendix C.2. The error analysis of the high-order solver corresponding to the deterministic formulation is provided in Appendix E.

3.3 Guided conditional SDEs in spectral domains

Inspired by classifier-free guidance in conditional generation (Rombach et al., 2022; Ho & Salimans, 2022), we introduce a guidance strength $\gamma_t > 0$ to control the contribution of the measurement matching score and refer to $\gamma_t \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t)$ as the *guidance term*. We use a time-dependent linear schedule for γ_t to vary measurement influence during sampling. Motivated by the stochastic sampling used in recent DM-based inverse solvers (Kawar et al., 2022; Wang et al., 2023; Zhu et al., 2023), we further introduce a stochasticity parameter $\tau \geq 0$ to control the sampling noise in the conditional reverse process. Combining guidance strength and stochasticity gives

$$\mathrm{d}\mathbf{x}_t = \Big(f(t)\mathbf{x}_t - \frac{1+\tau^2}{2} g^2(t) \big(\nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) + \gamma_t \nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \big) \Big) \mathrm{d}t + \tau g(t) \mathrm{d}\bar{\mathbf{w}}_t. \tag{21}$$

In Eq. (21), $\tau = 0$ yields a deterministic conditional ODE, while $\tau = 1$ recovers the standard conditional reverse-time SDE when $\gamma_t = 1$. The guidance strength γ_t controls the contribution of the measurement matching score, while τ controls the sampling noise.

The SVD of the forward operator also diagonalizes the linear terms introduced by measurement conditioning in Eq. (21), enabling their efficient computation in the spectral domain. The resulting conditional SDE admits the following integral representation.

Proposition 3 (Integral representation of the transformed SDE)**.** *Given* $\tilde{\mathbf{x}}_s$ *at time* s*, the solution to the transformed conditional SDE at* $t \in [0, s]$ *admits the integral representation*

$$\begin{aligned} \tilde{\mathbf{x}}_t = {} & \left(\frac{\alpha_t}{\alpha_s} \right)^{1-\kappa} \left(\frac{\sigma_t}{\sigma_s} \right)^{\kappa} \tilde{\mathbf{x}}_s + \kappa \alpha_t^{1-\kappa} \sigma_t^{\kappa} \int_{\lambda_s}^{\lambda_t} e^{\kappa\lambda} \big(\bar{\mathbf{S}}_{\gamma_{t(\lambda)},\lambda} \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \bar{\mathbf{g}}_{\gamma_{t(\lambda)},\lambda} \big) \mathrm{d}\lambda \\ & + \alpha_t^{1-\kappa} \sigma_t^{\kappa} \sqrt{e^{2(\kappa-1)\lambda_t} - e^{2(\kappa-1)\lambda_s}}\, \mathbf{z}_s, \end{aligned} \tag{22}$$

where $\kappa = 1 + \tau^2 \geq 1$, $\mathbf{z}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$, *and* $\bar{\mathbf{S}}_{\gamma_{t(\lambda)},\lambda}$ *and* $\bar{\mathbf{g}}_{\gamma_{t(\lambda)},\lambda}$ *denote the corresponding coefficients under the guidance strength schedule.*

In Proposition 3, the first term gives a closed-form transition from $\tilde{\mathbf{x}}_s$, the diagonal coefficients in the second term allow efficient coordinate-wise computation of the linear terms, and the third term introduces stochastic noise that can be sampled directly. The explicit coefficient forms are provided in Appendix C.1, and the derivation is provided in Appendix C.3.

3.4 Numerical Implementation

We discretize the sampling interval utilizing $T = t_I > t_{I-1} > \cdots > t_0 = \epsilon$, with the corresponding half-log SNRs satisfying $\lambda_{t_I} < \lambda_{t_{I-1}} < \cdots < \lambda_{t_0}$. [1] Starting from $\tilde{\mathbf{x}}_{t_I} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$, we compute $\tilde{\mathbf{x}}_{t_{i-1}}$ from $\tilde{\mathbf{x}}_{t_i}$ utilizing the integral representation in Proposition 3. The spectral coefficients are computed coordinate-wise, allowing efficient computation of the linear terms, while high-order approximation is applied only to the nonlinear model prediction.

For the model prediction, we adopt the first- and second-order approximation strategies of DPM-Solver++ (Lu et al., 2025) and the second-order predictor-corrector strategy of UniPC (Zhao et al., 2023), resulting in DPM-1, DPM-2, and UniPC-2, respectively. These approximation strategies are applied to the model prediction in our spectral integral formulation as follows.

For $\lambda \in [\lambda_{t_i}, \lambda_{t_{i-1}}]$, DPM-1 uses the constant approximation

$$\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) \approx \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_i}}, \lambda_{t_i}). \tag{23}$$

DPM-2 uses the linear approximation

$$\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) \approx \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_i}}, \lambda_{t_i}) + (\lambda - \lambda_{t_i})\mathbf{c}_i, \quad \mathbf{c}_i = \frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_i}}, \lambda_{t_i}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_{i+1}}}, \lambda_{t_{i+1}})}{\lambda_{t_i} - \lambda_{t_{i+1}}}. \tag{24}$$

The first sampling step uses DPM-1 when no previous model prediction is available.

UniPC-2 follows the predictor–corrector construction of UniPC (Zhao et al., 2023). The predictor uses the first-order approximation to obtain $\tilde{\mathbf{x}}^{\text{pred}}_{t_{i-1}}$, at which the model prediction is evaluated. The corrector uses the linear approximation in Eq. (24), with

$$\mathbf{c}_i = \frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}^{\text{pred}}_{t_{i-1}}, \lambda_{t_{i-1}}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}^{\text{pred}}_{t_i}, \lambda_{t_i})}{\lambda_{t_{i-1}} - \lambda_{t_i}}, \tag{25}$$

where the model prediction at $\tilde{\mathbf{x}}^{\text{pred}}_{t_i}$ is cached from the previous interval. Each predicted-endpoint evaluation is reused in the subsequent step, avoiding an additional NFE at the corrected state. On the final interval, the predictor is used directly without an endpoint evaluation. The overall sampling procedure is summarized in Algorithm 1. We also extend these numerical approximations to third order and evaluate their reconstruction quality with 5 to 10 sampling steps in Appendix B.

4 Experiments

In this section, we evaluate the performance of our method on linear inverse problems.

4.1 Experimental setup

Datasets and metrics. All experiments utilize pretrained DMs at a resolution of 256×256. We utilize the CelebA model from (Choi et al., 2021) and the unconditional ImageNet model from (Dhariwal & Nichol, 2021), with datasets from (Karras et al., 2018; Deng et al., 2009). For each dataset, we select 100 test images scaled to $[-1, 1]$ to match the input range of the pretrained DMs and utilize the same images across all methods. We report PSNR, SSIM, LPIPS (Zhang et al., 2018), and FID (Heusel et al., 2017).

[1] In practice, t_0 is set to a sufficiently small positive value to avoid numerical instability, following (Lu et al., 2022).

Algorithm 1 High-order sampling with SVD-based spectral operations

Require: Forward operator $\mathbf{A} \in \mathbb{R}^{M\times N}$, measurements $\mathbf{y} \in \mathbb{R}^M$, guidance schedule $\{\gamma_{t_i}\}_{i=0}^I$, stochasticity $\tau \geq 0$, time steps $\{t_i\}_{i=0}^I$, noise schedule $\{\alpha_{t_i}, \sigma_{t_i}\}_{i=0}^I$, model prediction network $\mathbf{x}_{\boldsymbol{\theta}}$, inverse solver $\mathcal{G} \in \{\text{DiffPIR}, \text{DDNM}\}$, numerical solver $\mathcal{T} \in \{\text{DPM-1}, \text{DPM-2}, \text{UniPC-2}\}$

Compute $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$, $\tilde{\mathbf{y}} = \mathbf{U}^\top\mathbf{y}$, $\kappa = 1 + \tau^2$, and $\{\lambda_{t_i}\}_{i=0}^I$ from $\{\alpha_{t_i}, \sigma_{t_i}\}_{i=0}^I$
Compute $\{\bar{\mathbf{S}}_i, \bar{\mathbf{g}}_i\}_{i=0}^I$ from $\boldsymbol{\Sigma}$, $\tilde{\mathbf{y}}$, $\{\gamma_{t_i}\}_{i=0}^I$, and $\mathcal{G}$ using Proposition 1
Define $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}, t) = \mathbf{V}^\top\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{V}\tilde{\mathbf{x}}, t)$
Sample $\tilde{\mathbf{x}}_{t_I} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$; cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t_I}, t_I)$ if $\mathcal{T} =$ UniPC-2
for $i = I, I-1, \ldots, 1$ **do**
    Sample $\mathbf{z}_{t_i} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$
    if $\mathcal{T} =$ DPM-1 **then**
        Evaluate $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t_i}, t_i)$; update $\tilde{\mathbf{x}}_{t_{i-1}}$ using Eqs. (22) and (23)
    else if $\mathcal{T} =$ DPM-2 **then**
        Evaluate and cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t_i}, t_i)$; update $\tilde{\mathbf{x}}_{t_{i-1}}$ using Eqs. (22) and (24), or Eq. (23) for $i = I$
    else
        Predict $\tilde{\mathbf{x}}_{t_{i-1}}^{\text{pred}}$ using Eqs. (22) and (23) with the cached model prediction
        Evaluate and cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_{t_{i-1}}^{\text{pred}}, t_{i-1})$; correct using Eqs. (22), (24), and (25)
    end if
end for
return $\mathbf{x}_{t_0} = \mathbf{V}\tilde{\mathbf{x}}_{t_0}$

Inverse tasks and measurements. We consider inpainting with a random mask removing 70% of pixels, compressed sensing with Walsh–Hadamard measurements at a 50% sampling ratio, and motion deblurring with a normalized 61×61 motion kernel of intensity 0.5. We utilize a closed-form SVD for inpainting and a permuted Fast Walsh–Hadamard Transform for compressed sensing. Additive Gaussian measurement noise has standard deviation $\sigma_y = 0.05$.

Table 1: Quantitative comparison on CelebA at NFE = 5 for inpainting with a 70% random mask and compressed sensing. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDRM	17.03±1.29	0.535±0.072	0.574±0.078	100.27	19.06±2.33	0.630±0.069	0.362±0.049	76.18
DDNM	17.09±1.44	0.533±0.074	0.570±0.090	101.62	19.16±2.49	0.634±0.074	0.363±0.052	74.90
DiffPIR	17.53±1.49	0.378±0.050	0.635±0.055	148.70	19.75±2.39	0.515±0.047	0.464±0.047	128.63
DDS	13.54±1.67	0.348±0.065	0.781±0.096	188.32	16.93±1.92	0.521±0.059	0.520±0.045	158.34
SSD	15.93±1.06	0.249±0.040	0.706±0.049	169.82	18.21±2.17	0.450±0.046	0.512±0.039	184.81
DDNM-Solver (DPM-1)	26.64±1.55	0.785±0.047	0.260±0.031	70.38	18.89±2.68	0.648±0.070	0.421±0.050	81.44
DDNM-Solver (DPM-2)	27.10±1.48	0.790±0.044	**0.254**±0.030	68.59	18.53±2.17	0.633±0.073	0.470±0.097	82.36
DDNM-Solver (UniPC-2)	**27.67**±1.49	**0.797**±0.044	**0.254**±0.029	**68.34**	**20.91**±2.67	**0.689**±0.070	**0.324**±0.050	**71.42**
DiffPIR-Solver (DPM-1)	26.10±1.52	0.776±0.047	0.263±0.030	70.48	19.82±2.45	0.655±0.062	0.373±0.062	77.53
DiffPIR-Solver (DPM-2)	26.54±1.40	0.780±0.046	0.260±0.031	69.57	19.70±2.33	0.646±0.073	0.411±0.071	79.08
DiffPIR-Solver (UniPC-2)	27.13±1.45	0.788±0.047	0.262±0.031	69.51	20.88±2.66	0.687±0.070	**0.324**±0.050	72.42

Baselines and our method. We compare with DDRM (Kawar et al., 2022), DDNM (Wang et al., 2023), DiffPIR (Zhu et al., 2023), and the accelerated inverse solvers DDS (Chung et al., 2024) and SSD (Liu et al., 2024). Applying our formulation to the guidance terms of DiffPIR and DDNM gives DiffPIR-Solver and DDNM-Solver, respectively. For each solver, we evaluate the DPM-1, DPM-2, and UniPC-2 discretizations described in Section 3.4. We utilize $I + 1$ uniformly spaced time nodes over $[10^{-3}, 1]$, corresponding to I integration intervals. Complete experimental settings are provided in Appendix A.

For motion deblurring, the forward operator is not explicitly constructed. The considered implementations of DDRM, DDNM, and SSD require an explicit pseudoinverse or SVD and therefore do not directly apply to this setting, whereas DiffPIR and DDS can be implemented utilizing only the forward operation. Our method exploits the Fourier-domain diagonalization of the convolutional forward operator, allowing the required spectral operations to be efficiently implemented utilizing FFTs without explicitly constructing the operator matrix or its SVD, as detailed in Appendix D. We therefore compare with DiffPIR and DDS for motion deblurring. DPS (Chung et al., 2023) and DAPS (Zhang et al., 2025a) utilize iterative gradient-based measurement guidance and typically require hundreds to thousands of sampling steps in their reported experiments, differing from the few-

step high-order sampling considered here, and are therefore not included in the main comparisons. Other accelerated inverse solvers utilize different sampling settings, as discussed in Section 1.1. Comparisons with C-ΠGDM, CCDF, and LLE-based variants of DDRM, DDNM, and DiffPIR are provided in Appendix F.2.

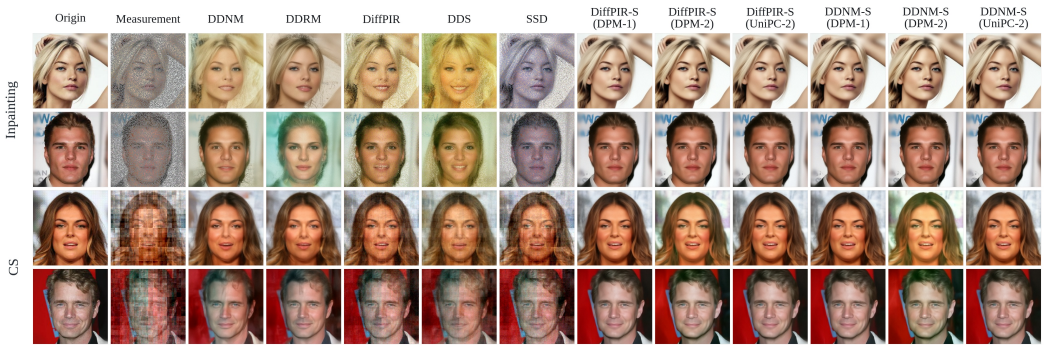

Figure 1: Qualitative results for 70% random inpainting and compressed sensing (CS) on CelebA at $\mathrm{NFE} = 5$. DDNM-S and DiffPIR-S denote their solver variants with DPM-1, DPM-2, or UniPC-2.

4.2 Results and Analysis

Main results. Table 1 compares reconstruction quality on CelebA inpainting and compressed sensing at $\mathrm{NFE} = 5$, with qualitative results in Fig. 1. The proposed solvers generally achieve better reconstruction quality than the baselines across the reported metrics with 5 sampling steps.[2] Additional results on ImageNet, motion deblurring, and $\mathrm{NFE} = 10$ are provided in Appendix F.1, with additional accelerated baselines in Appendix F.2 and higher-order discretizations in Appendix B.

Ablation study. We examine stochasticity τ and initial guidance strength γ_0 using a linear guidance schedule with $\gamma_1 = 1$. Random inpainting benefits from stronger initial guidance, whereas compressed sensing generally favors moderate guidance. With suitable guidance strengths, increasing stochasticity generally improves perceptual quality in the evaluated few-step settings, motivating its use in the main experiments. The joint ablation of guidance strength and stochasticity is provided in Appendix F.3.

Runtime analysis. We evaluate runtime over 100 images, with detailed runtime and implementation comparisons provided in Appendix F.4. The forward-operator operations for inpainting and compressed sensing and the FFT-based operations for motion deblurring account for only a small fraction of the overall sampling time.

5 Conclusion

We developed high-order diffusion solvers for linear inverse problems by identifying a class of structured measurement matching scores that incorporate the model prediction while retaining the semilinear structure exploited by high-order solvers. The corresponding integral equations contain dense matrices in the linear terms. We utilize the SVD of the forward operator to diagonalize these matrices in the spectral domain, enabling efficient computation of the linear terms while applying high-order approximation only to the nonlinear model prediction. Based on this formulation, we developed DiffPIR-Solver and DDNM-Solver for conditional ODE and SDE sampling, with guidance strength and stochasticity controlling measurement influence and sampling noise, respectively. Experiments on CelebA and ImageNet demonstrate improved reconstruction quality over baselines with 5 to 10 sampling steps. Our current framework focuses on linear inverse problems, and its extension to nonlinear inverse problems is left for future work, where the corresponding differential equations may not admit the semilinear structure that can be efficiently exploited by our high-order solvers.

[2] DiffPIR and DDS share the objective in Eq. (12), but use a time-dependent parameter and a fixed parameter with CG, respectively, leading to different quantitative results as reported in Table 1.

AI USE STATEMENT

Generative AI tools were used to assist with and polish the writing of this paper. All AI-assisted text was reviewed by the authors. The authors take full responsibility for the final content of the manuscript.

ETHICS STATEMENT

This paper presents work advancing efficient inverse problem solving. We do not foresee specific negative societal impacts beyond standard considerations for generative models.

REPRODUCIBILITY STATEMENT

We provide detailed implementation details, hyperparameter settings, and baseline configurations in the appendix to facilitate reproducibility of our results.

REFERENCES

Yudong Chen, Ali Jalali, Sujay Sanghavi, and Constantine Caramanis. Low-rank matrix recovery from errors and erasures. *IEEE Transactions on Information Theory*, 59(7):4324–4337, 2013.

Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. ILVR: Conditioning method for denoising diffusion probabilistic models. In *ICCV*, 2021.

Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *NeurIPS*, 2022a.

Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In *CVPR*, pp. 12413–12422, 2022b.

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

Hyungjin Chung, Suhyeon Lee, and Jong Chul Ye. Decomposed diffusion sampler for accelerating large-scale inverse problems. In *ICLR*, 2024.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *NeurIPS*, 2021.

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

Omar Elharrouss, Noor Almaadeed, Somaya Al-Maadeed, and Younes Akbari. Image inpainting: A review. *Neural Processing Letters*, 51(2):2007–2028, 2020.

Zalan Fabian, Berk Tinaz, and Mahdi Soltanolkotabi. Adapt and diffuse: Sample-adaptive reconstruction via latent diffusion models. In *ICML*, volume 235, pp. 12723, 2024.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NeurIPS*, 2017.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020.

Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G Dimakis, and Jon Tamir. Robust compressed sensing MRI with deep generative priors. In *NeurIPS*, 2021.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *ICLR*, 2018.

Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. In *NeurIPS*, volume 34, pp. 21757–21769, 2021.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *NeurIPS*, 2022.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. In *NeurIPS*, 2021.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *ICLR*, 2014.

Xiaodong Li. Compressed sensing and matrix completion with constant proportion of corruptions. *Constructive Approximation*, 37(1):73–99, 2013.

Gongye Liu, Haoze Sun, Jiayi Li, Fei Yin, and Yujiu Yang. Accelerating diffusion models for inverse problems through shortcut sampling. In *IJCAI*, pp. 1101–1109, 2024.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *NeurIPS*, 2022.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver++: Fast solver for guided sampling of diffusion probabilistic models. *Machine Intelligence Research*, 2025.

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

Xiangming Meng and Yoshiyuki Kabashima. Diffusion model based posterior sampling for noisy linear inverse problems. In *ACML*, 2025.

Nam H Nguyen and Trac D Tran. Robust Lasso with missing and grossly corrupted observations. *IEEE Transactions on Information Theory*, 59(4):2036–2058, 2012.

Nam H. Nguyen and Trac D. Tran. Exact recoverability from dense corrupted observations via ℓ_1-Minimization. *IEEE Transactions on Information Theory*, 59(4):2017–2035, 2013.

Kushagra Pandey, Ruihan Yang, and Stephan Mandt. Fast samplers for inverse problems in iterative refinement models. In *NeurIPS*, volume 37, pp. 26872–26914, 2024.

Kushagra Pandey, Farrin Marouf Sofian, Felix Draxler, Theofanis Karaletsos, and Stephan Mandt. Variational control for guidance in diffusion models. In *ICML*, 2025.

Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. Super-resolution image reconstruction: a technical overview. *IEEE Signal Processing Magazine*, 20(3):21–36, 2003.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, pp. 10684–10695, 2022.

Litu Rout, Negin Raoof, Giannis Daras, Constantine Caramanis, Alex Dimakis, and Sanjay Shakkottai. Solving linear inverse problems provably via posterior sampling with latent diffusion models. In *NeurIPS*, volume 36, pp. 49960–49990, 2023.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

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 *ICLR*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021a.

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *ICLR*, 2023.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

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

Phong Tran, Anh Tuan Tran, Quynh Phung, and Minh Hoai. Explore image deblurring via encoded blur kernel space. In *CVPR*, 2021.

Le Wang and Shengmei Zhao. Fast reconstructed and high-quality ghost imaging with fast walsh–hadamard transform. *Photonics Research*, 4(6):240–244, 2016.

Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *ICLR*, 2023.

Huan Xu, Constantine Caramanis, and Shie Mannor. Outlier-robust PCA: The high-dimensional case. *IEEE Transactions on Information Theory*, 59(1):546–572, 2012.

Shuchen Xue, Mingyang Yi, Weijian Luo, Shifeng Zhang, Jiacheng Sun, Zhenguo Li, and Zhi-Ming Ma. SA-Solver: Stochastic Adams solver for fast sampling of diffusion models. In *NeurIPS*, 2023.

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *CVPR*, 2025a.

Jiawei Zhang, Ziyuan Liu, Leon Yan, Gen Li, and Yuantao Gu. Improving diffusion-based inverse algorithms under few-step constraint via learnable linear extrapolation. In *NeurIPS*, 2025b.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *ICLR*, 2023.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *NeurIPS*, 2023.

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 *CVPRW*, 2023.

A HYPERPARAMETERS

We provide the hyperparameter settings for all baseline methods and our proposed solvers. For each method, we present the relevant update formulas to clarify the role of each parameter.

DDRM DDRM (Kawar et al., 2022) operates in the spectral domain of the forward operator $\mathbf{A}$. Let $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ be the SVD of the forward operator, where $\mathbf{U} \in \mathbb{R}^{M\times M}$ and $\mathbf{V} \in \mathbb{R}^{N\times N}$ are orthogonal matrices, and $\mathbf{\Sigma} = [\mathrm{Diag}(s_1, \ldots, s_M), \mathbf{0}_{M\times(N-M)}] \in \mathbb{R}^{M\times N}$ is a rectangular diagonal matrix of singular values $s_1, \ldots, s_M$.

At each timestep t_i, the denoised estimate is first computed via Tweedie's formula:

$$\mathbf{x}_{0,t_i} = \frac{\mathbf{x}_{t_i} - \sigma_{t_i}\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_{t_i}, t_i)}{\alpha_{t_i}}. \tag{26}$$

The corrector step then applies a modified projection in the SVD basis. Denoting $\tilde{\mathbf{x}} = \mathbf{V}^\top\mathbf{x}$ and $\tilde{\mathbf{y}} = \mathbf{U}^\top\mathbf{y}$, the k-th component of the corrected estimate is given by the following piecewise function:

$$\tilde{x}'^{(k)}_{0,t_i} = \begin{cases} \tilde{x}^{(k)}_{0,t_i}, & s_k = 0, \\ \tilde{x}^{(k)}_{0,t_i} + \sqrt{1-\eta^2}\dfrac{\sigma_{t_{i-1}}/\alpha_{t_{i-1}}}{\sigma_y/s_k}\left(\dfrac{\tilde{y}^{(k)}}{s_k} - \tilde{x}^{(k)}_{0,t_i}\right), & \sigma_{t_{i-1}} < \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \\ (1-\eta_b)\tilde{x}^{(k)}_{0,t_i} + \eta_b\dfrac{\tilde{y}^{(k)}}{s_k}, & \sigma_{t_{i-1}} \geq \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}. \end{cases} \tag{27}$$

The noiser step adds noise according to a DDIM-like schedule

$$\tilde{x}^{(k)}_{t_{i-1}} = \begin{cases} \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \sqrt{1-\eta^2}\sigma_{t_{i-1}}\tilde{\epsilon}^{(k)}_{\boldsymbol{\theta}} + \eta\sigma_{t_{i-1}}z^{(k)}, & s_k = 0, \\ \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \eta\sigma_{t_{i-1}}z^{(k)}, & \sigma_{t_{i-1}} < \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \\ \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \sqrt{\sigma^2_{t_{i-1}} - \dfrac{\alpha^2_{t_{i-1}}\sigma^2_y}{s^2_k}\eta_b^2}\cdot z^{(k)}, & \sigma_{t_{i-1}} \geq \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \end{cases} \tag{28}$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The parameter η controls stochasticity in the null-space components, and η_b controls the balance between data fidelity and prior in the range-space components. Following the original paper, we set $\eta = 0.85$ and $\eta_b = 1.0$ for all experiments.

DDNM DDNM (Wang et al., 2023) utilizes Tweedie denoising. The corrector applies a time-dependent weighting

$$\mathbf{x}'_{0,t_i} = \mathbf{x}_{0,t_i} + \mathbf{\Psi}_{t_i}\mathbf{A}^\dagger(\mathbf{y} - \mathbf{A}\mathbf{x}_{0,t_i}), \tag{29}$$

where $\mathbf{\Psi}_{t_i} = \mathbf{V}\mathrm{Diag}(d_{t_i,1}, \ldots, d_{t_i,N})\mathbf{V}^\top$ is a diagonal matrix in the SVD basis. Our DDNM-Solver method specifically adopts the parameter setting with elements defined as:

$$d_{t_i,k} = \begin{cases} 1, & \sigma_{t_{i-1}} \geq \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \\ \dfrac{s_k\sigma_{t_{i-1}}\sqrt{1-\eta^2}}{\alpha_{t_{i-1}}\sigma_y}, & \sigma_{t_{i-1}} < \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \\ 1, & s_k = 0, \end{cases} \tag{30}$$

where the hyperparameter η in the original DDNM formulation. The noiser step follows a similar structure to DDRM

$$\tilde{x}^{(k)}_{t_{i-1}} = \begin{cases} \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \sqrt{1-\eta^2}\sigma_{t_{i-1}}\tilde{\epsilon}^{(k)}_{\boldsymbol{\theta}} + \eta\sigma_{t_{i-1}}z^{(k)}, & s_k = 0, \\ \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \eta\sigma_{t_{i-1}}z^{(k)}, & \sigma_{t_{i-1}} < \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}, \\ \alpha_{t_{i-1}}\tilde{x}'^{(k)}_{0,t_i} + \sqrt{\sigma^2_{t_{i-1}} - \dfrac{\alpha^2_{t_{i-1}}\sigma^2_y}{s^2_k}}\cdot z^{(k)}, & \sigma_{t_{i-1}} \geq \dfrac{\alpha_{t_{i-1}}\sigma_y}{s_k}. \end{cases} \tag{31}$$

We set $\eta = 0.85$ following the original paper.

DiffPIR DiffPIR (Zhu et al., 2023) alternates between a denoising step and a data-fidelity step. After obtaining the Tweedie estimate $\mathbf{x}_{0,t_i}$, the data-fidelity step solves the proximal problem

$$\mathbf{x}'_{0,t_i} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{\rho_{t_i}}{2}\|\mathbf{x} - \mathbf{x}_{0,t_i}\|_2^2, \tag{32}$$

where $\rho_{t_i} = \lambda\sigma_y^2\alpha_{t_i}^2/\sigma_{t_i}^2$ is a time-dependent regularization parameter. For a linear forward operator, the proximal problem in Eq. (32) admits the closed-form solution

$$\mathbf{x}'_{0,t_i} = (\mathbf{A}^\top\mathbf{A} + \rho_{t_i}\mathbf{I}_N)^{-1}(\mathbf{A}^\top\mathbf{y} + \rho_{t_i}\mathbf{x}_{0,t_i}). \tag{33}$$

Recall that $\tilde{\mathbf{x}} = \mathbf{V}^\top\mathbf{x}$ and $\tilde{\mathbf{y}} = \mathbf{U}^\top\mathbf{y}$. In the spectral domain, the components of the solution are given by

$$(\tilde{\mathbf{x}}'_{0,t_i})_\ell = \begin{cases} (s_\ell^2 + \rho_{t_i})^{-1}\left(s_\ell\tilde{y}_\ell + \rho_{t_i}(\tilde{\mathbf{x}}_{0,t_i})_\ell\right), & \ell \le M, \\ (\tilde{\mathbf{x}}_{0,t_i})_\ell, & \ell > M. \end{cases} \tag{34}$$

The subsequent noising step combines the corrected clean estimate with stochastic noise:

$$\mathbf{x}_{t_{i-1}} = \alpha_{t_{i-1}}\mathbf{x}'_{0,t_i} + \eta\sigma_{t_{i-1}}\boldsymbol{\epsilon} + \sqrt{1-\eta^2}\frac{\sigma_{t_{i-1}}}{\sigma_{t_i}}\left(\mathbf{x}_{t_i} - \alpha_{t_i}\mathbf{x}'_{0,t_i}\right), \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N), \tag{35}$$

where η controls the stochasticity of the sampling process. Since the original DiffPIR paper does not provide hyperparameter settings for noisy inpainting and compressed sensing, we follow the few-step DiffPIR configuration utilized in LLE (Zhang et al., 2025b), setting $\eta = 1.0$ and $\lambda = 7.0$ for these experiments.

DDS. DDS (Chung et al., 2024) approximately solves the data-consistency problem using conjugate-gradient (CG) iterations. Given the clean prediction $\hat{\mathbf{x}}_t = (\mathbf{x}_t - \sigma_t\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t))/\alpha_t$, DDS computes

$$\mathbf{x}_t^{\mathrm{dc}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{\rho_{\mathrm{DDS}}}{2}\|\mathbf{x} - \hat{\mathbf{x}}_t\|_2^2, \tag{36}$$

where ρ_{DDS} is fixed across diffusion time. We use the noisy formulation of DDS and approximately solve Eq. (36) by CG initialized from $\hat{\mathbf{x}}_t$, followed by the DDIM update

$$\begin{aligned} \mathbf{x}_s &= \alpha_s\mathbf{x}_t^{\mathrm{dc}} + \sqrt{\sigma_s^2 - \eta^2\tilde{\beta}_{t,s}^2}\,\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \eta\tilde{\beta}_{t,s}\mathbf{z}, \\ \tilde{\beta}_{t,s}^2 &= \frac{\sigma_s^2}{\sigma_t^2}\left(1 - \frac{\alpha_t^2}{\alpha_s^2}\right), \qquad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N). \end{aligned} \tag{37}$$

Following the original noisy configuration, we use five CG iterations with the corresponding fixed regularization parameter and set $\eta = 0.15$.

SSD. SSD (Liu et al., 2024) accelerates inverse sampling through two stages. The first stage applies distortion-adaptive inversion from the measurement-based initialization $\mathbf{A}^\dagger\mathbf{y}$ to obtain a noisy state at a shortcut time t_{start}. Given the clean prediction $\hat{\mathbf{x}}_t = (\mathbf{x}_t - \sigma_t\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t))/\alpha_t$, the inversion step from t to $u > t$ is

$$\begin{aligned} \mathbf{x}_u &= \alpha_u\hat{\mathbf{x}}_t + \sqrt{\sigma_u^2 - \eta_1\beta_{t,u}}\,\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \sqrt{\eta_1\beta_{t,u}}\,\mathbf{z}, \\ \beta_{t,u} &= 1 - \frac{\alpha_u^2}{\alpha_t^2}, \qquad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N). \end{aligned} \tag{38}$$

The second stage performs generation from the shortcut state, alternating denoising with measurement projection:

$$\begin{aligned} \mathbf{x}_t^{\mathrm{dc}} &= \hat{\mathbf{x}}_t + \mathbf{A}^\dagger(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}_t), \\ \mathbf{x}_s &= \alpha_s\mathbf{x}_t^{\mathrm{dc}} + \sigma_s\left(\sqrt{1-\eta_2^2}\,\boldsymbol{\epsilon}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) + \eta_2\mathbf{z}\right). \end{aligned} \tag{39}$$

We utilize $t_{\mathrm{start}} = 0.55$, $\eta_1 = 0.4$, and $\eta_2 = 0.8$, following the configurations utilized in the original implementation.

C-ΠGDM C-ΠGDM (Pandey et al., 2024) accelerates ΠGDM by applying a conjugate transformation to the conditional diffusion dynamics. It builds on the measurement guidance of ΠGDM, which takes the form

$$\nabla_{\mathbf{x}_t} \log p_t(\mathbf{y}|\mathbf{x}_t) \approx \left(\frac{\partial \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)}{\partial \mathbf{x}_t}\right)^\top \mathbf{A}^\top \left(r_t^2 \mathbf{A}\mathbf{A}^\top + \sigma_y^2 \mathbf{I}_M\right)^{-1} \left(\mathbf{y} - \mathbf{A}\mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\right), \tag{40}$$

where r_t^2 represents the posterior uncertainty utilized in ΠGDM. The guidance combines the measurement residual with the forward operator, measurement noise, and the Jacobian of the clean prediction.

Rather than directly integrating the resulting conditional dynamics, C-ΠGDM introduces the transformed variable

$$\mathbf{u}_t = \mathbf{T}_t^{\sigma_y} \mathbf{x}_t, \tag{41}$$

where $\mathbf{T}_t^{\sigma_y}$ is a time-dependent transformation determined by the forward operator, the diffusion schedule, the guidance weight w, the transformation parameter λ, and the measurement noise level σ_y. For noisy measurements, the transformation includes an additional noise-dependent correction involving the pseudo-inverse of the forward operator.

The reverse process is initialized from a noisy pseudo-inverse reconstruction,

$$\mathbf{x}_{t_s} = \alpha_{t_s} \mathbf{A}^\dagger \mathbf{y} + \sigma_{t_s} \mathbf{z}, \qquad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N), \tag{42}$$

which is then mapped to the transformed coordinates by $\mathbf{u}_{t_s} = \mathbf{T}_{t_s}^{\sigma_y} \mathbf{x}_{t_s}$. During sampling, the current state is recovered through

$$\mathbf{x}_{t_i} = (\mathbf{T}_{t_i}^{\sigma_y})^{-1} \mathbf{u}_{t_i}, \tag{43}$$

and the reverse update is performed in the transformed coordinates.

The original paper does not provide hyperparameter settings for the noisy inpainting and compressed-sensing configurations considered in our experiments. Moreover, prior work reports difficulty in obtaining plausible reconstructions with C-ΠGDM for noisy inpainting after several hyperparameter tuning (Pandey et al., 2025). We include C-ΠGDM as an additional reference rather than a primary baseline. We use the default configuration with $\lambda = 0$ and $w = 1.0$ throughout these evaluations.

LLE LLE (Zhang et al., 2025b) accelerates DM-based inverse solvers by learning to extrapolate clean estimates utilizing information from previous sampling steps. At timestep t_i, the underlying inverse solver first produces a corrected clean estimate $\hat{\mathbf{x}}_{0,t_i}$. LLE combines this estimate with historical clean estimates as

$$\tilde{\mathbf{x}}_{0,t_i} = c_{t_i,I-i} \hat{\mathbf{x}}_{0,t_i} + \sum_{j=0}^{I-i-1} c_{t_i,j} \tilde{\mathbf{x}}_{0,t_{I-j}}, \tag{44}$$

where I is the total number of sampling steps and $c_{t_i,j}$ are learned extrapolation coefficients. The extrapolated estimate $\tilde{\mathbf{x}}_{0,t_i}$ is then utilized in the update of the underlying inverse solver to obtain the next sample. LLE therefore improves few-step sampling by incorporating historical clean estimates without changing the underlying inverse solver.

The extrapolation coefficients are learned before inference utilizing reference samples by minimizing a weighted combination of MSE and LPIPS losses between the extrapolated estimates and the corresponding clean images. We follow the training configuration of Zhang et al. (2025b). The learned coefficients are fixed during inference. Since the coefficients are learned for the corresponding inverse solver and task configuration, LLE requires additional training before few-step inference.

Our method We introduce a guidance strength schedule γ_t and a stochasticity parameter τ, as described in Section 3.1. The guided SDE takes the form given in Eq. (21), where the posterior score is approximated utilizing the linear guidance framework in Eq. (16).

The schedule γ_t controls the influence of the measurement guidance. Larger values impose stronger measurement consistency, while smaller values place more emphasis on the learned prior. Table 2 summarizes the task-specific settings of γ_t utilized for CelebA and ImageNet. For DiffPIR-Solver, we fix the regularization parameter to $\lambda = 7$ across all tasks and datasets. We set $\tau = 1$ and utilize a uniform time schedule in all experiments.

Table 2: Task-specific guidance strength γ_t utilized for CelebA and ImageNet.

	Inpainting (random)		Compressed sensing		Motion deblurring	
Method	CelebA	ImageNet	CelebA	ImageNet	CelebA	ImageNet
DiffPIR-Solver	4	4	2	2	2	2
DDNM-Solver	4.5	4.5	2	2	–	–

B THIRD-ORDER DISCRETIZATIONS

This section extends the numerical discretizations in Section 3.4 to third order. Following DPM-Solver++ (Lu et al., 2025) and UniPC (Zhao et al., 2023), we apply third-order approximation only to the nonlinear model prediction in the integral representation of Proposition 3, while the linear terms are computed coordinate-wise utilizing the spectral coefficients.

DPM-3. Extending DPM-2, DPM-3 uses a quadratic approximation based on the model predictions at the current step λ_{t_i} and the two preceding steps $\lambda_{t_{i+1}}$ and $\lambda_{t_{i+2}}$:

$$\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) \approx \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_i}}, \lambda_{t_i}) + (\lambda - \lambda_{t_i})\mathbf{c}_{i,1} + (\lambda - \lambda_{t_i})(\lambda - \lambda_{t_{i+1}})\mathbf{c}_{i,2}. \tag{45}$$

The coefficients are given by the divided differences

$$\begin{aligned} \mathbf{c}_{i,1} &= \frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_i}}, \lambda_{t_i}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_{i+1}}}, \lambda_{t_{i+1}})}{\lambda_{t_i} - \lambda_{t_{i+1}}}, \\ \mathbf{c}_{i,2} &= \frac{1}{\lambda_{t_i} - \lambda_{t_{i+2}}} \left(\mathbf{c}_{i,1} - \frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_{i+1}}}, \lambda_{t_{i+1}}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_{\lambda_{t_{i+2}}}, \lambda_{t_{i+2}})}{\lambda_{t_{i+1}} - \lambda_{t_{i+2}}} \right). \end{aligned} \tag{46}$$

DPM-3 requires two historical model predictions and therefore uses DPM-1 for the first step and DPM-2 for the second.

UniPC-3. Extending UniPC-2, UniPC-3 uses the DPM-2 approximation in Eq. (24) to obtain $\tilde{\mathbf{x}}^{\mathrm{pred}}_{t_{i-1}}$, where the model prediction is evaluated. The corrector then uses a quadratic approximation based on the predicted endpoint and two historical predictions:

$$\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) \approx \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{\mathrm{pred}}_{\lambda_{t_i}}, \lambda_{t_i}) + (\lambda - \lambda_{t_i})\mathbf{c}_{i,1} + (\lambda - \lambda_{t_i})(\lambda - \lambda_{t_{i+1}})\mathbf{c}_{i,2}. \tag{47}$$

The coefficients are

$$\begin{aligned} \mathbf{c}_{i,1} &= \frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{\mathrm{pred}}_{\lambda_{t_i}}, \lambda_{t_i}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{\mathrm{pred}}_{\lambda_{t_{i+1}}}, \lambda_{t_{i+1}})}{\lambda_{t_i} - \lambda_{t_{i+1}}}, \\ \mathbf{c}_{i,2} &= \frac{1}{\lambda_{t_{i-1}} - \lambda_{t_{i+1}}} \left(\frac{\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{\mathrm{pred}}_{\lambda_{t_{i-1}}}, \lambda_{t_{i-1}}) - \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{\mathrm{pred}}_{\lambda_{t_i}}, \lambda_{t_i})}{\lambda_{t_{i-1}} - \lambda_{t_i}} - \mathbf{c}_{i,1} \right). \end{aligned} \tag{48}$$

As in UniPC-2, model predictions are evaluated only at predicted states and cached for subsequent steps. UniPC-3 uses the corresponding lower-order approximation when insufficient historical predictions are available.

Tables 3 and 4 compare the second- and third-order variants for CelebA inpainting and compressed sensing, while Table 5 reports motion deblurring results for DiffPIR-Solver. The third-order variants do not consistently improve reconstruction quality under the evaluated NFE budgets. We therefore utilize the DPM-1, DPM-2, and UniPC-2 discretizations in the main experiments.

C PROOFS OF MAIN THEORETICAL RESULTS

In this section, we provide detailed proofs of the main theoretical results and the error analysis.

Table 3: Quantitative comparison of higher-order numerical schemes utilizing the DDNM-Solver on CelebA at $\mathrm{NFE} = 5$ and $\mathrm{NFE} = 10$. We evaluate second- and third-order variants across inpainting and compressed sensing tasks. We mark **bold** the best scores and underline the second best.

Method	NFE	Inpainting (random)				Compressed sensing			
		PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDNM-Solver (DPM-2)	5	27.10±1.48	0.790±0.044	**0.254**±0.030	68.59	18.53±2.17	0.633±0.073	0.470±0.097	82.36
DDNM-Solver (DPM-3)	5	25.06±2.02	0.749±0.055	0.288±0.037	72.78	17.33±2.07	0.578±0.077	0.539±0.101	117.86
DDNM-Solver (UniPC-2)	5	**27.67**±1.49	**0.797**±0.044	**0.254**±0.029	**68.34**	**20.91**±2.67	**0.689**±0.070	**0.324**±0.050	71.42
DDNM-Solver (UniPC-3)	5	27.49±1.49	0.795±0.044	0.255±0.029	68.70	20.68±2.53	0.683±0.070	0.345±0.056	**71.30**
DDNM-Solver (DPM-2)	10	28.44±1.70	0.816±0.039	**0.221**±0.026	52.79	**23.38**±3.26	0.757±0.069	**0.258**±0.047	53.22
DDNM-Solver (DPM-3)	10	28.20±1.75	0.813±0.040	0.223±0.026	**52.71**	22.50±3.04	0.737±0.070	0.275±0.048	55.08
DDNM-Solver (UniPC-2)	10	**28.76**±1.63	**0.821**±0.038	0.222±0.026	53.96	23.12±3.16	0.755±0.067	0.262±0.046	**52.51**
DDNM-Solver (UniPC-3)	10	28.68±1.66	**0.821**±0.038	0.222±0.026	53.77	23.31±3.25	**0.758**±0.067	0.260±0.046	52.67

Table 4: Quantitative comparison of higher-order numerical schemes utilizing the DiffPIR-Solver on CelebA at $\mathrm{NFE} = 5$ and $\mathrm{NFE} = 10$. We evaluate second- and third-order variants across inpainting and compressed sensing. We mark **bold** the best scores and underline the second best.

Method	NFE	Inpainting (random)				Compressed sensing			
		PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DiffPIR-Solver (DPM-2)	5	26.54±1.40	0.780±0.046	**0.260**±0.031	69.57	19.70±2.33	0.646±0.073	0.411±0.071	79.08
DiffPIR-Solver (DPM-3)	5	25.26±1.37	0.761±0.048	0.284±0.034	69.98	18.57±2.18	0.594±0.077	0.482±0.082	109.37
DiffPIR-Solver (UniPC-2)	5	**27.13**±1.45	**0.788**±0.047	0.262±0.031	**69.51**	**20.88**±2.66	**0.687**±0.070	**0.324**±0.050	72.42
DiffPIR-Solver (UniPC-3)	5	26.95±1.44	0.786±0.047	0.262±0.030	69.77	20.82±2.57	0.682±0.070	0.337±0.052	**72.29**
DiffPIR-Solver (DPM-2)	10	27.89±1.65	0.807±0.043	**0.229**±0.027	55.63	**23.14**±3.18	0.749±0.071	**0.265**±0.047	56.25
DiffPIR-Solver (DPM-3)	10	27.62±1.68	0.803±0.044	0.231±0.027	**55.60**	22.17±2.92	0.728±0.071	0.283±0.048	57.83
DiffPIR-Solver (UniPC-2)	10	**28.16**±1.58	**0.811**±0.042	0.231±0.027	56.80	22.91±3.08	0.749±0.068	0.268±0.047	**55.70**
DiffPIR-Solver (UniPC-3)	10	28.10±1.61	**0.811**±0.042	0.230±0.027	56.96	23.09±3.17	**0.751**±0.069	0.266±0.046	56.22

Table 5: Quantitative comparison of higher-order numerical schemes utilizing the DiffPIR-Solver on CelebA at $\mathrm{NFE} = 5$ and $\mathrm{NFE} = 10$. We evaluate second- and third-order variants on motion deblurring. We mark **bold** the best scores and underline the second best.

Method	NFE	Motion deblurring			
		PSNR↑	SSIM↑	LPIPS↓	FID↓
DiffPIR-Solver (DPM-2)	5	27.81±1.58	0.773±0.041	0.264±0.028	75.11
DiffPIR-Solver (DPM-3)	5	27.75±1.58	0.771±0.040	0.265±0.028	75.24
DiffPIR-Solver (UniPC-2)	5	**28.20**±1.59	**0.783**±0.042	**0.262**±0.028	75.04
DiffPIR-Solver (UniPC-3)	5	28.16±1.59	0.781±0.042	0.263±0.028	**74.22**
DiffPIR-Solver (DPM-2)	10	28.25±1.84	0.800±0.046	**0.228**±0.026	51.54
DiffPIR-Solver (DPM-3)	10	27.99±1.91	0.795±0.048	0.229±0.026	52.93
DiffPIR-Solver (UniPC-2)	10	**28.94**±1.74	**0.809**±0.044	**0.228**±0.026	**51.39**
DiffPIR-Solver (UniPC-3)	10	28.78±1.79	0.808±0.044	**0.228**±0.026	51.66

C.1 Proof of Proposition 1

For the deterministic formulation with $\gamma_t = 1$ and $\tau = 0$, note that

$$\mathbf{V}^\top \mathbf{S}_t \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \mathbf{V}^\top \mathbf{S}_t \mathbf{V}\mathbf{V}^\top \mathbf{x}_{\boldsymbol{\theta}}(\mathbf{V}\tilde{\mathbf{x}}_t, t) \tag{49}$$

$$= \tilde{\mathbf{S}}_t \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t). \tag{50}$$

Multiplying both sides of Eq. (18) by $\mathbf{V}^\top$ therefore yields Eq. (19).

For DiffPIR, $\mathbf{M}_t = (\mathbf{A}^\top\mathbf{A} + \rho_t \mathbf{I}_N)^{-1}\mathbf{A}^\top$. Based on the SVD of $\mathbf{A}$, we obtain

$$\mathbf{M}_t\mathbf{A} = (\mathbf{A}^\top\mathbf{A} + \rho_t \mathbf{I}_N)^{-1}\mathbf{A}^\top\mathbf{A} \tag{51}$$

$$= \left(\mathbf{V}\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma}\mathbf{V}^\top + \rho_t \mathbf{I}_N\right)^{-1}\mathbf{V}\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma}\mathbf{V}^\top \tag{52}$$

$$= \mathbf{V}(\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma} + \rho_t \mathbf{I}_N)^{-1}\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma}\mathbf{V}^\top. \tag{53}$$

Then,

$$\tilde{\mathbf{S}}_t = \mathbf{V}^\top \mathbf{S}_t \mathbf{V} \tag{54}$$

$$= \mathbf{V}^\top(\mathbf{I}_N - \mathbf{M}_t\mathbf{A})\mathbf{V} \tag{55}$$

$$= \mathbf{I}_N - (\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma} + \rho_t \mathbf{I}_N)^{-1}\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma}, \tag{56}$$

which is diagonal in $\mathbb{R}^{N\times N}$, with

$$(\tilde{\mathbf{S}}_t)_{\ell\ell} = \begin{cases} \dfrac{\rho_t}{s_\ell^2 + \rho_t}, & \ell \le M, \\ 1, & \ell > M. \end{cases} \tag{57}$$

Moreover,

$$\tilde{\mathbf{g}}_t = \mathbf{V}^\top \mathbf{g}_t = \mathbf{V}^\top \mathbf{M}_t \mathbf{y} \tag{58}$$

$$= (\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma} + \rho_t \mathbf{I}_N)^{-1}\boldsymbol{\Sigma}^\top\mathbf{U}^\top\mathbf{y} \tag{59}$$

$$= (\boldsymbol{\Sigma}^\top\boldsymbol{\Sigma} + \rho_t \mathbf{I}_N)^{-1}\boldsymbol{\Sigma}^\top\tilde{\mathbf{y}}. \tag{60}$$

Therefore,

$$(\tilde{\mathbf{g}}_t)_\ell = \begin{cases} \dfrac{s_\ell \tilde{y}_\ell}{s_\ell^2 + \rho_t}, & \ell \le M, \\ 0, & \ell > M. \end{cases} \tag{61}$$

For DDNM, $\mathbf{M}_t = \boldsymbol{\Psi}_t\mathbf{A}^\dagger$, where $\boldsymbol{\Psi}_t = \mathbf{V}\mathbf{D}_t\mathbf{V}^\top \in \mathbb{R}^{N\times N}$ and $\mathbf{D}_t = \mathrm{Diag}(d_{t,1}, \ldots, d_{t,N})$.[3] We have

$$\mathbf{M}_t\mathbf{A} = \boldsymbol{\Psi}_t\mathbf{A}^\dagger\mathbf{A} \tag{62}$$

$$= \mathbf{V}\mathbf{D}_t\mathbf{V}^\top\mathbf{V}\boldsymbol{\Sigma}^\dagger\mathbf{U}^\top\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top \tag{63}$$

$$= \mathbf{V}\mathbf{D}_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}\mathbf{V}^\top. \tag{64}$$

Then,

$$\tilde{\mathbf{S}}_t = \mathbf{V}^\top(\mathbf{I}_N - \mathbf{M}_t\mathbf{A})\mathbf{V} \tag{65}$$

$$= \mathbf{I}_N - \mathbf{D}_t\boldsymbol{\Sigma}^\dagger\boldsymbol{\Sigma}, \tag{66}$$

which is diagonal in $\mathbb{R}^{N\times N}$, with

$$(\tilde{\mathbf{S}}_t)_{\ell\ell} = \begin{cases} 1 - d_{t,\ell}, & \ell \le M, \\ 1, & \ell > M. \end{cases} \tag{67}$$

Moreover,

$$\tilde{\mathbf{g}}_t = \mathbf{V}^\top \mathbf{g}_t = \mathbf{V}^\top \mathbf{M}_t \mathbf{y} \tag{68}$$

$$= \mathbf{D}_t\boldsymbol{\Sigma}^\dagger\mathbf{U}^\top\mathbf{y} = \mathbf{D}_t\boldsymbol{\Sigma}^\dagger\tilde{\mathbf{y}}. \tag{69}$$

[3] The specific settings of $d_{t,j}$, $j \in \{1, \ldots, N\}$, are provided in Eq. (30) of Appendix A.

Therefore,

$$(\tilde{\mathbf{g}}_t)_\ell = \begin{cases} \dfrac{d_{t,\ell}\tilde{y}_\ell}{s_\ell}, & \ell \le M, \\ 0, & \ell > M. \end{cases} \tag{70}$$

For the generalized formulation in Eq. (21), the guidance strength modifies these coefficients as $\mathbf{S}_{\gamma_t,t} = \mathbf{I}_N - \gamma_t \mathbf{M}_t \mathbf{A}$ and $\mathbf{g}_{\gamma_t,t} = \gamma_t \mathbf{M}_t \mathbf{y}$. Hence, the SVD yields diagonal $\tilde{\mathbf{S}}_{\gamma_t,t} = \mathbf{V}^\top \mathbf{S}_{\gamma_t,t} \mathbf{V}$. For DiffPIR,

$$(\tilde{\mathbf{S}}_{\gamma_t,t})_{\ell\ell} = \begin{cases} \dfrac{\rho_t + (1-\gamma_t)s_\ell^2}{s_\ell^2 + \rho_t}, & \ell \le M, \\ 1, & \ell > M, \end{cases} \tag{71}$$

and

$$(\tilde{\mathbf{g}}_{\gamma_t,t})_\ell = \begin{cases} \dfrac{\gamma_t s_\ell \tilde{y}_\ell}{s_\ell^2 + \rho_t}, & \ell \le M, \\ 0, & \ell > M. \end{cases} \tag{72}$$

For DDNM,

$$(\tilde{\mathbf{S}}_{\gamma_t,t})_{\ell\ell} = \begin{cases} 1 - \gamma_t d_{t,\ell}, & \ell \le M, \\ 1, & \ell > M, \end{cases} \tag{73}$$

and

$$(\tilde{\mathbf{g}}_{\gamma_t,t})_\ell = \begin{cases} \dfrac{\gamma_t d_{t,\ell}\tilde{y}_\ell}{s_\ell}, & \ell \le M, \\ 0, & \ell > M. \end{cases} \tag{74}$$

Thus, introducing γ_t changes the coordinate-wise coefficients while preserving the diagonal spectral structure utilized in Proposition 3.

C.2 Proof of Proposition 2

Proof. Let $h(t) = f(t) + \frac{g^2(t)}{2\sigma_t^2}$ and $c(t) = -\frac{\alpha_t g^2(t)}{2\sigma_t^2}$. The ODE in Eq. (19) becomes

$$\mathrm{d}\tilde{\mathbf{x}}_t = \left(h(t)\tilde{\mathbf{x}}_t + c(t)(\tilde{\mathbf{S}}_t \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) + \tilde{\mathbf{g}}_t) \right) \mathrm{d}t. \tag{75}$$

Then, we have

$$\mathrm{d}\left(e^{-\int_s^t h(r)\mathrm{d}r} \tilde{\mathbf{x}}_t \right) = e^{-\int_s^t h(r)\mathrm{d}r} \left(\mathrm{d}\tilde{\mathbf{x}}_t - h(t)\tilde{\mathbf{x}}_t \mathrm{d}t \right) \tag{76}$$

$$= e^{-\int_s^t h(r)\mathrm{d}r} c(t)(\tilde{\mathbf{S}}_t \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) + \tilde{\mathbf{g}}_t)\mathrm{d}t. \tag{77}$$

Taking the integral of both sides gives

$$\tilde{\mathbf{x}}_t = e^{\int_s^t h(r)\mathrm{d}r} \tilde{\mathbf{x}}_s + \int_s^t e^{\int_r^t h(u)\mathrm{d}u} c(r)(\tilde{\mathbf{S}}_r \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_r)\mathrm{d}r. \tag{78}$$

Recall from Section 2.1 that $f(t) = \mathrm{d}\log(\alpha_t)/\mathrm{d}t$ and $g^2(t) = -2\sigma_t^2 \mathrm{d}\lambda_t/\mathrm{d}t$. Then, we obtain

$$h(t) = f(t) + \frac{g^2(t)}{2\sigma_t^2} = \frac{\mathrm{d}\log(\sigma_t)}{\mathrm{d}t}, \tag{79}$$

$$c(t) = -\frac{\alpha_t g^2(t)}{2\sigma_t^2} = \alpha_t \frac{\mathrm{d}\lambda_t}{\mathrm{d}t}. \tag{80}$$

Therefore,

$$e^{\int_s^t h(r)\mathrm{d}r} = \frac{\sigma_t}{\sigma_s}, \tag{81}$$

which directly gives the first term in the right-hand side of Eq. (20). Additionally, we have

$$\int_s^t e^{\int_r^t h(u)\mathrm{d}u} c(r)(\tilde{\mathbf{S}}_r \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_r)\mathrm{d}r \tag{82}$$

$$= \int_s^t \frac{\sigma_t}{\sigma_r} \alpha_r (\tilde{\mathbf{S}}_r \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_r)\mathrm{d}\lambda_r \tag{83}$$

$$= \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda} \left(\bar{\mathbf{S}}_\lambda \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \bar{\mathbf{g}}_\lambda \right) \mathrm{d}\lambda. \tag{84}$$

This corresponds to the second term in the right-hand side of Eq. (20) and completes the proof. □

C.3 Proof of Proposition 3

Proof. Let $h(t) = f(t) + \frac{\kappa g^2(t)}{2\sigma_t^2}$ and $c(t) = -\frac{\alpha_t \kappa g^2(t)}{2\sigma_t^2}$. The SDE in Eq. (21) after applying the SVD becomes

$$\mathrm{d}\tilde{\mathbf{x}}_t = \left(h(t)\tilde{\mathbf{x}}_t + c(t)(\tilde{\mathbf{S}}_{\gamma_t,t}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) + \tilde{\mathbf{g}}_{\gamma_t,t})\right)\mathrm{d}t + \tau g(t)\mathrm{d}\tilde{\mathbf{w}}_t. \tag{85}$$

Then, we have

$$\mathrm{d}\left(e^{-\int_s^t h(r)\mathrm{d}r}\tilde{\mathbf{x}}_t\right) = e^{-\int_s^t h(r)\mathrm{d}r}(\mathrm{d}\tilde{\mathbf{x}}_t - h(t)\tilde{\mathbf{x}}_t\mathrm{d}t) \tag{86}$$

$$= e^{-\int_s^t h(r)\mathrm{d}r}\left(c(t)(\tilde{\mathbf{S}}_{\gamma_t,t}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_t, t) + \tilde{\mathbf{g}}_{\gamma_t,t})\mathrm{d}t + \tau g(t)\mathrm{d}\tilde{\mathbf{w}}_t\right). \tag{87}$$

Taking the integral of both sides gives

$$e^{-\int_s^t h(r)\mathrm{d}r}\tilde{\mathbf{x}}_t = \int_s^t e^{-\int_s^r h(u)\mathrm{d}u}c(r)(\tilde{\mathbf{S}}_{\gamma_r,r}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_{\gamma_r,r})\mathrm{d}r$$
$$+ \int_s^t e^{-\int_s^r h(u)\mathrm{d}u}\tau g(r)\mathrm{d}\tilde{\mathbf{w}}_r + \mathbf{c}, \tag{88}$$

where $\mathbf{c} \in \mathbb{R}^N$ is a constant vector. Setting $t = s$ leads to $\mathbf{c} = \tilde{\mathbf{x}}_s$. Therefore, we derive

$$\tilde{\mathbf{x}}_t = e^{\int_s^t h(r)\mathrm{d}r}\tilde{\mathbf{x}}_s + \int_s^t e^{\int_r^t h(u)\mathrm{d}u}\tau g(r)\mathrm{d}\tilde{\mathbf{w}}_r$$
$$+ \int_s^t e^{\int_r^t h(u)\mathrm{d}u}c(r)(\tilde{\mathbf{S}}_{\gamma_r,r}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_{\gamma_r,r})\mathrm{d}r. \tag{89}$$

Recall from Section 2.1 that $f(t) = \mathrm{d}\log(\alpha_t)/\mathrm{d}t$ and $g^2(t) = -2\sigma_t^2\mathrm{d}\lambda_t/\mathrm{d}t$. Then, we obtain

$$h(t) = f(t) + \frac{\kappa g^2(t)}{2\sigma_t^2} = (1-\kappa)\frac{\mathrm{d}\log(\alpha_t)}{\mathrm{d}t} + \kappa\frac{\mathrm{d}\log(\sigma_t)}{\mathrm{d}t}, \tag{90}$$

$$c(t) = -\frac{\alpha_t \kappa g^2(t)}{2\sigma_t^2} = \kappa\alpha_t\frac{\mathrm{d}\lambda_t}{\mathrm{d}t}. \tag{91}$$

Therefore,

$$e^{\int_s^t h(r)\mathrm{d}r} = \left(\frac{\alpha_t}{\alpha_s}\right)^{1-\kappa}\left(\frac{\sigma_t}{\sigma_s}\right)^{\kappa}, \tag{92}$$

which directly gives the first term in the right-hand side of Eq. (22). Additionally, now we have that the third term in Eq. (89) is

$$\int_s^t e^{\int_r^t h(u)\mathrm{d}u}c(r)(\tilde{\mathbf{S}}_{\gamma_r,r}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_{\gamma_r,r})\mathrm{d}r$$

$$= \int_s^t \left(\frac{\alpha_t}{\alpha_r}\right)^{1-\kappa}\left(\frac{\sigma_t}{\sigma_r}\right)^{\kappa}\kappa\alpha_r(\tilde{\mathbf{S}}_{\gamma_r,r}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_{\gamma_r,r})\mathrm{d}\lambda_r \tag{93}$$

$$= \kappa\alpha_t^{1-\kappa}\sigma_t^{\kappa}\int_s^t e^{\kappa\lambda_r}(\tilde{\mathbf{S}}_{\gamma_r,r}\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\tilde{\mathbf{x}}_r, r) + \tilde{\mathbf{g}}_{\gamma_r,r})\mathrm{d}\lambda_r \tag{94}$$

$$= \kappa\alpha_t^{1-\kappa}\sigma_t^{\kappa}\int_{\lambda_s}^{\lambda_t} e^{\kappa\lambda}\left(\bar{\mathbf{S}}_{\gamma_{t(\lambda)},\lambda}\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \bar{\mathbf{g}}_{\gamma_{t(\lambda)},\lambda}\right)\mathrm{d}\lambda. \tag{95}$$

This corresponds to the second term in the right-hand side of Eq. (22). Moreover, for the second term of Eq. (89), if $\tau = 0$ (and thus $\kappa = 1 + \tau^2 = 1$), it is clear that this term is a zero vector. Therefore, we only focus on the case that $\tau > 0$ (and thus $\kappa > 1$). We have

$$\int_s^t e^{\int_r^t h(u)\mathrm{d}u}\tau g(r)\mathrm{d}\tilde{\mathbf{w}}_r = \int_s^t \left(\frac{\alpha_t}{\alpha_r}\right)^{1-\kappa}\left(\frac{\sigma_t}{\sigma_r}\right)^{\kappa}\tau g(r)\mathrm{d}\tilde{\mathbf{w}}_r \tag{96}$$

$$\sim \mathcal{N}\left(\mathbf{0}, \left(-\int_s^t \left(\frac{\alpha_t}{\alpha_r}\right)^{2(1-\kappa)}\left(\frac{\sigma_t}{\sigma_r}\right)^{2\kappa}\tau^2 g^2(r)\mathrm{d}r\right)\mathbf{I}_N\right). \tag{97}$$

Utilizing $g^2(r) = -2\sigma_r^2 \mathrm{d}\lambda_r/\mathrm{d}r$, we derive

$$
\begin{aligned}
&- \int_s^t \left(\frac{\alpha_t}{\alpha_r}\right)^{2(1-\kappa)} \left(\frac{\sigma_t}{\sigma_r}\right)^{2\kappa} \tau^2 g^2(r)\mathrm{d}r \\
&= 2\tau^2 \alpha_t^{2(1-\kappa)} \sigma_t^{2\kappa} \int_s^t \frac{\sigma_r^2}{\alpha_r^2} e^{2\kappa\lambda_r} \mathrm{d}r \qquad (98) \\
&= 2\tau^2 \alpha_t^{2(1-\kappa)} \sigma_t^{2\kappa} \int_{\lambda_s}^{\lambda_t} e^{(2\kappa-2)\lambda} \mathrm{d}\lambda \qquad (99) \\
&= 2\tau^2 \alpha_t^{2(1-\kappa)} \sigma_t^{2\kappa} \cdot \frac{e^{(2\kappa-2)\lambda_t} - e^{(2\kappa-2)\lambda_s}}{2\kappa - 2}. \qquad (100)
\end{aligned}
$$

Therefore, we obtain that $\int_s^t e^{\int_r^t h(u)\mathrm{d}u} \tau g(r) \mathrm{d}\tilde{\mathbf{w}}_r$ can be written as

$$
\sqrt{\frac{2}{2\kappa - 2}}\tau \cdot \alpha_t^{1-\kappa} \sigma_t^{\kappa} \sqrt{e^{2(\kappa-1)\lambda_t} - e^{2(\kappa-1)\lambda_s}} \cdot \mathbf{z}_s
$$

for $\mathbf{z}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$. Note that since $\kappa = 1 + \tau^2$, we have $\sqrt{\frac{2}{2\kappa-2}}\tau = 1$. This leads to the third term in the right-hand side of Eq. (22) and completes the proof. □

D FFT-BASED IMPLEMENTATION FOR CONVOLUTIONAL OPERATORS

We apply the proposed framework to linear inverse problems where the forward operator $\mathbf{A}$ represents a convolution, with motion deblurring as the example considered here. Unlike the general setting $\mathbf{A} \in \mathbb{R}^{M \times N}$, convolution under periodic boundary conditions preserves the signal dimension, giving $M = N$. The measurement model is $\mathbf{y} = \mathbf{h} * \mathbf{x}^* + \mathbf{n} = \mathbf{A}\mathbf{x}^* + \mathbf{n}$, where $\mathbf{h}$ is the convolution kernel and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is circulant. The discrete Fourier transform (DFT) diagonalizes $\mathbf{A}$, allowing the spectral operations required by our formulation to be implemented utilizing FFTs without explicitly constructing the forward matrix or its SVD.

D.1 DIAGONAL STRUCTURE IN THE FOURIER DOMAIN

Let $\mathbf{F} \in \mathbb{C}^{N \times N}$ denote the unitary DFT matrix, with $\mathbf{F}^{-1} = \mathbf{F}^{\mathrm{H}}$ and $\mathbf{F}^{\mathrm{H}}\mathbf{F} = \mathbf{I}_N$. The convolutional forward operator can be written as

$$
\mathbf{A} = \mathbf{F}^{-1} \mathbf{\Lambda}_{\hat{h}} \mathbf{F}, \qquad (101)
$$

where $\mathbf{\Lambda}_{\hat{h}} = \mathrm{Diag}(\hat{h}_1, \ldots, \hat{h}_N)$ contains the Fourier multipliers of the convolution kernel. We define $\hat{\mathbf{x}} = \mathbf{F}\mathbf{x}$ and $\hat{\mathbf{y}} = \mathbf{F}\mathbf{y}$. This representation provides the diagonal structure required for efficient computation of the linear terms without explicitly constructing the SVD of the forward operator.

As in Section 3.2, we first consider the deterministic formulation with $\gamma_t = 1$ and $\tau = 0$, and then extend the coefficients to the generalized formulation with guidance strength γ_t.

Proposition 4 (Diagonal structure in the Fourier domain)**.** *For the deterministic formulation, define* $\hat{\mathbf{S}}_t := \mathbf{F}\mathbf{S}_t\mathbf{F}^{-1} \in \mathbb{C}^{N \times N}$ *and* $\hat{\mathbf{g}}_t := \mathbf{F}\mathbf{g}_t \in \mathbb{C}^N$*. For DiffPIR,* $\hat{\mathbf{S}}_t$ *is diagonal, with entries*

$$
(\hat{\mathbf{S}}_t)_{kk} = \frac{\rho_t}{|\hat{h}_k|^2 + \rho_t}, \qquad (102)
$$

and the transformed measurement term has entries

$$
(\hat{\mathbf{g}}_t)_k = \frac{\hat{h}_k^* \hat{y}_k}{|\hat{h}_k|^2 + \rho_t}, \qquad (103)
$$

where $\hat{h}_k^*$ *denotes the complex conjugate of* $\hat{h}_k$*.*

For the generalized formulation in Eq. (21)*, define* $\hat{\mathbf{S}}_{\gamma_t,t} := \mathbf{F}\mathbf{S}_{\gamma_t,t}\mathbf{F}^{-1}$ *and* $\hat{\mathbf{g}}_{\gamma_t,t} := \mathbf{F}\mathbf{g}_{\gamma_t,t}$*. Then* $\hat{\mathbf{S}}_{\gamma_t,t}$ *remains diagonal, with entries*

$$
(\hat{\mathbf{S}}_{\gamma_t,t})_{kk} = \frac{\rho_t + (1-\gamma_t)|\hat{h}_k|^2}{|\hat{h}_k|^2 + \rho_t}, \qquad (104)
$$

and

$$(\hat{\mathbf{g}}_{\gamma_t,t})_k = \frac{\gamma_t \hat{h}_k^* \hat{y}_k}{|\hat{h}_k|^2 + \rho_t}. \tag{105}$$

The result follows from the diagonal Fourier representation of the convolution operator: $\mathbf{A}^\top \mathbf{A}$ and $\mathbf{A}^\top \mathbf{y}$ correspond to $|\hat{h}_k|^2$ and $\hat{h}_k^* \hat{y}_k$, respectively, at each frequency. The linear terms can be computed coordinate-wise utilizing FFTs without explicitly forming the forward matrix or its SVD, while high-order approximation remains applied only to the nonlinear model prediction.

D.2 Integral representations in the Fourier domain

We next apply the integral representations in Propositions 2 and 3 to the Fourier domain. Define the Fourier-domain model prediction as

$$\hat{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_t, t) := \mathcal{F}\big(\mathbf{x}_{\boldsymbol{\theta}}(\mathcal{F}^{-1}(\hat{\mathbf{x}}_t), t)\big)\,.$$

With $\bar{\mathbf{x}}_\lambda = \hat{\mathbf{x}}_{t(\lambda)}$ and $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\cdot, \lambda) = \hat{\mathbf{x}}_{\boldsymbol{\theta}}(\cdot, t(\lambda))$, the diagonal coefficients in Proposition 4 allow the linear terms in the corresponding integral equations to be computed coordinate-wise.

For the deterministic formulation with $\gamma_t = 1$ and $\tau = 0$, Proposition 2 becomes

$$\hat{\mathbf{x}}_t = \frac{\sigma_t}{\sigma_s}\hat{\mathbf{x}}_s + \sigma_t \int_{\lambda_s}^{\lambda_t} e^{\lambda} \left(\hat{\mathbf{S}}_\lambda \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \hat{\mathbf{g}}_\lambda\right) \mathrm{d}\lambda, \tag{106}$$

where $\hat{\mathbf{S}}_\lambda := \hat{\mathbf{S}}_{t(\lambda)}$ and $\hat{\mathbf{g}}_\lambda := \hat{\mathbf{g}}_{t(\lambda)}$. Since $\hat{\mathbf{S}}_\lambda$ is diagonal, its multiplication with the model prediction is performed coordinate-wise.

For the generalized formulation with guidance strength and stochasticity, Proposition 3 becomes

$$\begin{aligned}\hat{\mathbf{x}}_t = &\left(\frac{\alpha_t}{\alpha_s}\right)^{1-\kappa} \left(\frac{\sigma_t}{\sigma_s}\right)^{\kappa} \hat{\mathbf{x}}_s \\ &+ \kappa \alpha_t^{1-\kappa} \sigma_t^{\kappa} \int_{\lambda_s}^{\lambda_t} e^{\kappa\lambda} \left(\hat{\mathbf{S}}_{\gamma_{t(\lambda)},\lambda} \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\bar{\mathbf{x}}_\lambda, \lambda) + \hat{\mathbf{g}}_{\gamma_{t(\lambda)},\lambda}\right) \mathrm{d}\lambda \\ &+ \alpha_t^{1-\kappa} \sigma_t^{\kappa} \sqrt{e^{2(\kappa-1)\lambda_t} - e^{2(\kappa-1)\lambda_s}}\, \hat{\mathbf{z}}_s,\end{aligned} \tag{107}$$

where $\kappa = 1 + \tau^2$, $\hat{\mathbf{S}}_{\gamma_{t(\lambda)},\lambda} := \hat{\mathbf{S}}_{\gamma_{t(\lambda)},t(\lambda)}$, $\hat{\mathbf{g}}_{\gamma_{t(\lambda)},\lambda} := \hat{\mathbf{g}}_{\gamma_{t(\lambda)},t(\lambda)}$, and $\hat{\mathbf{z}}_s = \mathcal{F}(\mathbf{z}_s)$ for $\mathbf{z}_s \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$.

Equation (107) gives the spectral integral representation corresponding to Proposition 3 in the Fourier domain. The first term gives the closed-form transition from $\hat{\mathbf{x}}_s$, the diagonal Fourier coefficients in the second term allow efficient coordinate-wise computation of the linear terms, and the third term introduces the stochastic noise. The numerical approximations in Section 3.4 are therefore applied only to the nonlinear model prediction.

D.3 FFT-based sampling

The complete FFT-based implementation is summarized in Algorithm 2. The SVD-based spectral operations in Algorithm 1 are replaced by FFTs, while the DPM-1, DPM-2, and UniPC-2 discretizations remain unchanged.

Algorithm 2 replaces the SVD-based spectral operations with FFTs. The forward and adjoint operations become element-wise multiplications by $\hat{\mathbf{h}}$ and its complex conjugate, respectively, and the linear coefficients are computed coordinate-wise utilizing Eqs. (104) and (105). The remaining sampling procedure follows the DPM-1, DPM-2, and UniPC-2 discretizations in Section 3.4.

For the stochastic term, we sample $\mathbf{z}_{t_i} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ in the real spatial domain and then apply the FFT to obtain $\hat{\mathbf{z}}_{t_i}$. This preserves the conjugate symmetry of the Fourier coefficients, so that the corresponding inverse Fourier transform remains real-valued up to numerical precision.

Algorithm 2 High-order sampling with FFT-based spectral operations

Require: Convolution kernel $\mathbf{h} \in \mathbb{R}^N$, measurements $\mathbf{y} \in \mathbb{R}^N$, guidance schedule $\{\gamma_{t_i}\}_{i=0}^I$, stochasticity $\tau \geq 0$, time steps $\{t_i\}_{i=0}^I$, noise schedule $\{\alpha_{t_i}, \sigma_{t_i}\}_{i=0}^I$, model prediction network $\mathbf{x}_{\boldsymbol{\theta}}$, numerical solver $\mathcal{T} \in \{\text{DPM-1}, \text{DPM-2}, \text{UniPC-2}\}$

Compute $\hat{\mathbf{h}} = \mathcal{F}(\mathbf{h})$, $\hat{\mathbf{y}} = \mathcal{F}(\mathbf{y})$, $\kappa = 1 + \tau^2$, and $\{\lambda_{t_i}\}_{i=0}^I$ from $\{\alpha_{t_i}, \sigma_{t_i}\}_{i=0}^I$
Compute $\{\hat{\mathbf{S}}_{\gamma_{t_i}, t_i}, \hat{\mathbf{g}}_{\gamma_{t_i}, t_i}\}_{i=0}^I$ from $\hat{\mathbf{h}}$, $\hat{\mathbf{y}}$, and $\{\gamma_{t_i}\}_{i=0}^I$ using Eqs. (104) and (105)
Define $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}, t) = \mathcal{F}\big(\mathbf{x}_{\boldsymbol{\theta}}(\mathcal{F}^{-1}(\hat{\mathbf{x}}), t)\big)$
Sample $\mathbf{x}_{t_I} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ and set $\hat{\mathbf{x}}_{t_I} = \mathcal{F}(\mathbf{x}_{t_I})$; cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_I}, t_I)$ if $\mathcal{T} = \text{UniPC-2}$
for $i = I, I-1, \ldots, 1$ **do**
    Sample $\mathbf{z}_{t_i} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_N)$ and set $\hat{\mathbf{z}}_{t_i} = \mathcal{F}(\mathbf{z}_{t_i})$
    if $\mathcal{T} = \text{DPM-1}$ **then**
        Evaluate $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_i}, t_i)$; update $\hat{\mathbf{x}}_{t_{i-1}}$ using Eqs. (107) and (23)
    else if $\mathcal{T} = \text{DPM-2}$ **then**
        Evaluate and cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{t_i}, t_i)$; update $\hat{\mathbf{x}}_{t_{i-1}}$ using Eqs. (107) and (24), or Eq. (23) for $i = I$
    else
        Predict $\hat{\mathbf{x}}^{\text{pred}}_{t_{i-1}}$ using Eqs. (107) and (23) with the cached model prediction
        Evaluate and cache $\bar{\mathbf{x}}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}^{\text{pred}}_{t_{i-1}}, t_{i-1})$; correct using Eqs. (107), (24), and (25)
    end if
end for
return $\mathbf{x}_{t_0} = \mathcal{F}^{-1}(\hat{\mathbf{x}}_{t_0})$

D.4 Proof of Proposition 4

Proof. For the deterministic formulation,

$$\mathbf{M}_t = (\mathbf{A}^\top \mathbf{A} + \rho_t \mathbf{I}_N)^{-1} \mathbf{A}^\top.$$

Utilizing the Fourier decomposition $\mathbf{A} = \mathbf{F}^{-1} \mathbf{\Lambda}_{\hat{h}} \mathbf{F}$ and the unitarity of $\mathbf{F}$, we have

$$\mathbf{A}^\top \mathbf{A} = \mathbf{F}^{-1} \mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} \mathbf{F}.$$

Therefore,

$$\mathbf{F} \mathbf{M}_t \mathbf{A} \mathbf{F}^{-1} = (\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} + \rho_t \mathbf{I}_N)^{-1} \mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}}. \tag{108}$$

Since $\mathbf{S}_t = \mathbf{I}_N - \mathbf{M}_t \mathbf{A}$, its Fourier representation is

$$\hat{\mathbf{S}}_t = \mathbf{F} \mathbf{S}_t \mathbf{F}^{-1} \tag{109}$$

$$= \mathbf{I}_N - (\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} + \rho_t \mathbf{I}_N)^{-1} \mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}}. \tag{110}$$

Because $\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} = \text{Diag}(|\hat{h}_1|^2, \ldots, |\hat{h}_N|^2)$, $\hat{\mathbf{S}}_t$ is diagonal, with entries

$$(\hat{\mathbf{S}}_t)_{kk} = \frac{\rho_t}{|\hat{h}_k|^2 + \rho_t}, \tag{111}$$

which gives Eq. (102).

Similarly,

$$\hat{\mathbf{g}}_t = \mathbf{F} \mathbf{g}_t \tag{112}$$

$$= \mathbf{F} \mathbf{M}_t \mathbf{y} \tag{113}$$

$$= (\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} + \rho_t \mathbf{I}_N)^{-1} \mathbf{\Lambda}_{\hat{h}}^* \hat{\mathbf{y}}. \tag{114}$$

Hence,

$$(\hat{\mathbf{g}}_t)_k = \frac{\hat{h}_k^* \hat{y}_k}{|\hat{h}_k|^2 + \rho_t}, \tag{115}$$

which gives Eq. (103).

For the generalized formulation in Eq. (21), the guidance strength modifies these coefficients as $\mathbf{S}_{\gamma_t, t} = \mathbf{I}_N - \gamma_t \mathbf{M}_t \mathbf{A}$ and $\mathbf{g}_{\gamma_t, t} = \gamma_t \mathbf{M}_t \mathbf{y}$. Hence, the Fourier decomposition yields diagonal $\hat{\mathbf{S}}_{\gamma_t, t} = \mathbf{F} \mathbf{S}_{\gamma_t, t} \mathbf{F}^{-1}$. Specifically,

$$\hat{\mathbf{S}}_{\gamma_t, t} = \mathbf{I}_N - \gamma_t (\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} + \rho_t \mathbf{I}_N)^{-1} \mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}}, \tag{116}$$

which gives Eq. (104). Similarly,

$$\hat{\mathbf{g}}_{\gamma_t,t} = \mathbf{F}\mathbf{g}_{\gamma_t,t} \tag{117}$$

$$= \gamma_t(\mathbf{\Lambda}_{\hat{h}}^* \mathbf{\Lambda}_{\hat{h}} + \rho_t \mathbf{I}_N)^{-1} \mathbf{\Lambda}_{\hat{h}}^* \hat{\mathbf{y}}, \tag{118}$$

which gives Eq. (105). □

E Error analysis of Algorithm 1

In this section, we provide an error analysis for DPM-k ($k = 1, 2$) and UniPC-2, which are used in the main experiments and applied to the exact solution in Eq. (20). For $i \in \{1, \dots, I\}$, we let $s = t_i$ and $t = t_{i-1}$ for brevity, and denote the step size by

$$h := \lambda_t - \lambda_s > 0.$$

For notational convenience, we introduce the integrand

$$\boldsymbol{\phi}(\mathbf{x}, \lambda) := e^{\lambda}\Big(\bar{\mathbf{S}}_\lambda\, \bar{\mathbf{x}}_{\boldsymbol{\theta}}(\mathbf{x}, \lambda) + \bar{\mathbf{g}}_\lambda\Big).$$

With this notation, the exact solution in Eq. (20) can be written as

$$\tilde{\mathbf{x}}_t = \frac{\sigma_t}{\sigma_s}\tilde{\mathbf{x}}_s + \sigma_t \int_{\lambda_s}^{\lambda_t} \boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda, \lambda)\, \mathrm{d}\lambda, \tag{119}$$

where $\{\bar{\mathbf{x}}_\lambda\}_{\lambda\in[\lambda_s,\lambda_t]}$ denotes the exact trajectory in the λ-parameterization. We utilize $\mathcal{I}_{t,s}$ to denote the integral $\int_{\lambda_s}^{\lambda_t} \boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda, \lambda)\, \mathrm{d}\lambda$. Let $\tilde{\mathbf{x}}_t^h$ be the numerical update produced by DPM-k ($k = 1, 2$) or UniPC-2, defined by

$$\tilde{\mathbf{x}}_t^h = \frac{\sigma_t}{\sigma_s}\tilde{\mathbf{x}}_s^h + \sigma_t\, \widehat{\mathcal{I}}_{t,s}^{(k)}, \tag{120}$$

where $\widehat{\mathcal{I}}_{t,s}^{(k)}$ is obtained by substituting the corresponding order-k numerical approximations (see Eqs. (23), (24), and (25)) into the integral and performing analytic integration.

We assume the following regularity and stability conditions hold uniformly on the full interval $[\lambda_T, \lambda_0]$.

Assumption 1 (Regularity of the integrand $\boldsymbol{\phi}$)**.**

(A1) ***Bounded λ-derivatives.*** *There exist constants $M_\lambda, M_{\lambda\lambda}, M_{\lambda x} > 0$ such that for all $\lambda \in [\lambda_T, \lambda_0]$ and all $\mathbf{x} \in \mathbb{R}^N$,*

$$\|\partial_\lambda \boldsymbol{\phi}(\mathbf{x}, \lambda)\| \le M_\lambda, \qquad \|\partial_{\lambda\lambda} \boldsymbol{\phi}(\mathbf{x}, \lambda)\| \le M_{\lambda\lambda},$$
$$\|\partial_\lambda \nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{x}, \lambda)\| \le M_{\lambda x}.$$

(A2) ***Uniform $C^{1,1}$ regularity in state.*** *For all $\lambda \in [\lambda_T, \lambda_0]$, the map $\mathbf{x} \mapsto \boldsymbol{\phi}(\mathbf{x}, \lambda)$ is differentiable and its gradient is globally Lipschitz with constant $L_1 > 0$, i.e.,*

$$\|\nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{x}_1, \lambda) - \nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{x}_2, \lambda)\| \le L_1 \|\mathbf{x}_1 - \mathbf{x}_2\|, \qquad \forall \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^N.$$

Moreover, $\nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{0}, \lambda)$ is uniformly bounded in λ:

$$\sup_{\lambda\in[\lambda_T,\lambda_0]} \|\nabla_{\mathbf{x}} \boldsymbol{\phi}(\mathbf{0}, \lambda)\| \le G_0.$$

As a consequence, $\nabla_{\mathbf{x}}^2 \boldsymbol{\phi}(\mathbf{x}, \lambda)$ exists for a.e. $\mathbf{x}$ and satisfies $\|\nabla_{\mathbf{x}}^2 \boldsymbol{\phi}(\mathbf{x}, \lambda)\| \le L_1$ a.e., uniformly in λ.

Lemma 1 (Local truncation error)**.** *Consider the ODE in Eq.* (18)*, such that the exact trajectory $\{\bar{\mathbf{x}}_\lambda\}_{\lambda\in[\lambda_s,\lambda_t]}$ is twice differentiable. Under Assumption 1, there exists a constant $C_k > 0$, independent of h, such that*

$$\big\|\mathcal{I}_{t,s} - \widehat{\mathcal{I}}_{t,s}^{(k)}\big\| \le C_k\, h^{k+1}, \qquad k = 1, 2.$$

Moreover, the corresponding one-step local truncation error satisfies

$$\big\|\tilde{\mathbf{x}}_t - \tilde{\mathbf{x}}_t^h\big\| \le \frac{\sigma_t}{\sigma_s}\big\|\tilde{\mathbf{x}}_s - \tilde{\mathbf{x}}_s^h\big\| + \sigma_t C_k\, h^{k+1}.$$

Proof. Let

$$\boldsymbol{\psi}(\lambda) := \boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda, \lambda).$$

Then, we have $\mathcal{I}_{t,s} := \int_{\lambda_s}^{\lambda_t} \boldsymbol{\psi}(\lambda)\,\mathrm{d}\lambda$. The exact solution $\bar{\mathbf{x}}_\lambda$ satisfies a deterministic ODE obtained from Eq. (18). Under Assumption 1, the vector field and its gradient are globally Lipschitz with bounded coefficients on a compact interval, hence $\bar{\mathbf{x}}_\lambda \in C^2$ and there exist finite constants

$$R := \sup_{\lambda\in[\lambda_s,\lambda_t]} \|\bar{\mathbf{x}}_\lambda\|,$$

$$V_1 := \sup_{\lambda\in[\lambda_s,\lambda_t]} \Big\|\frac{\mathrm{d}}{\mathrm{d}\lambda}\bar{\mathbf{x}}_\lambda\Big\|,$$

$$V_2 := \sup_{\lambda\in[\lambda_s,\lambda_t]} \Big\|\frac{\mathrm{d}^2}{\mathrm{d}\lambda^2}\bar{\mathbf{x}}_\lambda\Big\|.$$

By Assumption 1(A2), $\nabla^2_{\mathbf{x}}\boldsymbol{\phi}(\mathbf{x},\lambda)$ exists for a.e. $\mathbf{x}$ and satisfies $\|\nabla^2_{\mathbf{x}}\boldsymbol{\phi}(\mathbf{x},\lambda)\| \le L_1$ a.e. Moreover,

$$\|\nabla_{\mathbf{x}}\boldsymbol{\phi}(\mathbf{x},\lambda)\| \le \|\nabla_{\mathbf{x}}\boldsymbol{\phi}(\mathbf{0},\lambda)\| + L_1\|\mathbf{x}\| \le G_0 + L_1 R \quad \text{on } [\lambda_s,\lambda_t],$$

where we assume that $\|\mathbf{x}\|$ is upper bounded by $R > 0$. Applying the chain rule yields

$$\boldsymbol{\psi}'(\lambda) = \partial_\lambda\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda) + \nabla_{\mathbf{x}}\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda)\,\bar{\mathbf{x}}'_\lambda,$$

and

$$\begin{aligned}\boldsymbol{\psi}''(\lambda) = {}& \partial_{\lambda\lambda}\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda) + 2\,\partial_\lambda\nabla_{\mathbf{x}}\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda)\,\bar{\mathbf{x}}'_\lambda \\ & + \nabla^2_{\mathbf{x}}\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda)\big[\bar{\mathbf{x}}'_\lambda,\bar{\mathbf{x}}'_\lambda\big] + \nabla_{\mathbf{x}}\boldsymbol{\phi}(\bar{\mathbf{x}}_\lambda,\lambda)\,\bar{\mathbf{x}}''_\lambda.\end{aligned}$$

Utilizing Assumption 1(A1)–(A2), we obtain the uniform bounds

$$\sup_{\lambda\in[\lambda_s,\lambda_t]} \|\boldsymbol{\psi}'(\lambda)\| \le M_\lambda + (G_0 + L_1 R)V_1 =: C_\psi,$$

and

$$\begin{aligned}\sup_{\lambda\in[\lambda_s,\lambda_t]} \|\boldsymbol{\psi}''(\lambda)\| \le {}& M_{\lambda\lambda} + 2M_{\lambda x}V_1 + L_1V_1^2 \\ & + (G_0 + L_1R)V_2 =: C^{(2)}_\psi.\end{aligned}$$

Case $k = 1$ Utilizing the constant approximation $\boldsymbol{\psi}(\lambda) \approx \boldsymbol{\psi}(\lambda_s)$,

$$\widehat{\mathcal{I}}^{(1)}_{t,s} = h\,\boldsymbol{\psi}(\lambda_s),$$

and hence

$$\mathcal{I}_{t,s} - \widehat{\mathcal{I}}^{(1)}_{t,s} = \int_{\lambda_s}^{\lambda_t}\int_{\lambda_s}^{\lambda} \boldsymbol{\psi}'(\tau)\,\mathrm{d}\tau\,\mathrm{d}\lambda.$$

Therefore,

$$\big\|\mathcal{I}_{t,s} - \widehat{\mathcal{I}}^{(1)}_{t,s}\big\| \le \sup_\tau \|\boldsymbol{\psi}'(\tau)\| \int_{\lambda_s}^{\lambda_t} (\lambda - \lambda_s)\,\mathrm{d}\lambda = \frac{C_\psi}{2}\,h^2.$$

Case $k = 2$ For DPM-2 or UniPC-2, let $\mathbf{p}_1(\lambda)$ denote the degree-1 approximation of $\boldsymbol{\psi}(\lambda)$ on $[\lambda_s,\lambda_t]$ constructed by extrapolation or predictor–corrector interpolation. Then

$$\widehat{\mathcal{I}}^{(2)}_{t,s} = \int_{\lambda_s}^{\lambda_t} \mathbf{p}_1(\lambda)\,\mathrm{d}\lambda.$$

The interpolation remainder for a C^2 function gives

$$\|\boldsymbol{\psi}(\lambda) - \mathbf{p}_1(\lambda)\| \le \frac{1}{2}\Big(\sup_{\xi\in[\lambda_s,\lambda_t]} \|\boldsymbol{\psi}''(\xi)\|\Big)(\lambda - \lambda_s)(\lambda_t - \lambda_s),$$

and thus

$$\big\|\mathcal{I}_{t,s} - \widehat{\mathcal{I}}^{(2)}_{t,s}\big\| \le \frac{h^3}{12} \sup_{\xi} \|\boldsymbol{\psi}''(\xi)\| \le \frac{C^{(2)}_{\psi}}{12}\, h^3.$$

The error orders also hold when the approximation is applied only to the model prediction.

Finally, subtracting Eq. (120) from Eq. (119) yields

$$\tilde{\mathbf{x}}_t - \tilde{\mathbf{x}}^h_t = \frac{\sigma_t}{\sigma_s}(\tilde{\mathbf{x}}_s - \tilde{\mathbf{x}}^h_s) + \sigma_t\big(\mathcal{I}_{t,s} - \widehat{\mathcal{I}}^{(k)}_{t,s}\big),$$

from which the stated bound follows. □

Theorem 1 (Global error)**.** *Consider the ODE in Eq.* (18)*. Let $\{t_i\}_{i=0}^I$ be the sequence of timesteps $T = t_I > t_{I-1} > \cdots > t_0$ with corresponding half log-SNRs $\lambda_{t_I} < \lambda_{t_{I-1}} < \cdots < \lambda_{t_0}$, and define the maximal step size*

$$h := \max_{1\le i\le I} \big(\lambda_{t_{i-1}} - \lambda_{t_i}\big).$$

Let $\{\tilde{\mathbf{x}}_{t_i}\}_{i=0}^I$ be the exact solutions given by Eq. (119) *and $\{\tilde{\mathbf{x}}^h_{t_i}\}_{i=0}^I$ be the numerical iterates produced by DPM-k ($k \in \{1, 2\}$) or UniPC-2, utilizing the following initial condition $\tilde{\mathbf{x}}^h_{t_I} = \tilde{\mathbf{x}}_{t_I}$. Suppose that Assumption 1 holds, then there exists a constant $C > 0$, independent of h, such that*

$$\max_{0\le i\le I} \|\tilde{\mathbf{x}}_{t_i} - \tilde{\mathbf{x}}^h_{t_i}\| \le C\, h^k, \qquad k = 1, 2.$$

Proof. It is easy to see that there exists a constant $\sigma_* \ge 0$ independent of h such that

$$|\sigma_{t_i}| \le \sigma_*, \qquad i = 0, \dots, I,$$

and moreover the multiplicative stability factor satisfies

$$\prod_{i=1}^{I} \frac{\sigma_{t_{i-1}}}{\sigma_{t_i}} = \frac{\sigma_{t_0}}{\sigma_{t_I}} \le C_\sigma < \infty.$$

Define the grid error $e_i := \|\tilde{\mathbf{x}}_{t_i} - \tilde{\mathbf{x}}^h_{t_i}\|$. By Lemma 1, for each step (t_i, t_{i-1}) we have

$$e_{i-1} \le \frac{\sigma_{t_{i-1}}}{\sigma_{t_i}}\, e_i + \sigma_{t_{i-1}} C_k\, h_i^{k+1}, \qquad h_i := \lambda_{t_{i-1}} - \lambda_{t_i}.$$

Iterating this inequality from $i = I$ down to 1 and utilizing $e_I = 0$ yields

$$e_0 \le \sum_{j=1}^{I} \left(\prod_{\ell=1}^{j-1} \frac{\sigma_{t_{\ell-1}}}{\sigma_{t_\ell}} \right) \sigma_{t_{j-1}} C_k\, h_j^{k+1}.$$

Utilizing the uniform bound $|\sigma_{t_{j-1}}| \le \sigma_*$ and

$$\prod_{\ell=1}^{j-1} \frac{\sigma_{t_{\ell-1}}}{\sigma_{t_\ell}} = \frac{\sigma_{t_0}}{\sigma_{t_{j-1}}} \le C_\sigma,$$

we obtain

$$e_0 \le C_\sigma\, \sigma_*\, C_k \sum_{j=1}^{I} h_j^{k+1}.$$

Since $h_j \le h$ and $\sum_{j=1}^I h_j = \lambda_{t_0} - \lambda_{t_I}$, we have

$$\sum_{j=1}^{I} h_j^{k+1} \le h^k \sum_{j=1}^{I} h_j = h^k(\lambda_{t_0} - \lambda_{t_I}),$$

and therefore

$$e_0 \le C_\sigma\, \sigma_*\, C_k(\lambda_{t_0} - \lambda_{t_I})\, h^k.$$

This argument applies to any e_i (by truncating the product/sum), hence $\max_{0\le i\le I} e_i \le Ch^k$ for a constant C independent of h. This completes the proof. □

F Additional Experimental Results and Analysis

F.1 Additional results

We provide additional quantitative and qualitative results that complement the main CelebA comparisons in Section 4.2. Results on ImageNet inpainting and compressed sensing at NFE $= 5$ are reported in Table 6, with qualitative comparisons in Fig. 2. Motion deblurring results on CelebA and ImageNet at NFE $= 5$ are reported in Table 7, with qualitative results on CelebA and ImageNet in Fig. 3. We further evaluate NFE $= 10$ on CelebA and ImageNet for inpainting and compressed sensing in Tables 8 and 9, respectively, and report the corresponding motion deblurring results in Table 10. These results further demonstrate the effectiveness of the proposed solvers across datasets, inverse tasks, and sampling budgets.

Table 6: Quantitative comparison on ImageNet at NFE = 5 for inpainting with a 70% random mask and compressed sensing. We report PSNR, SSIM, LPIPS and FID. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDRM	17.07±1.97	0.349±0.120	0.633±0.078	203.76	18.45±2.65	0.488±0.115	0.500±0.055	150.95
DDNM	17.02±2.07	0.341±0.111	0.641±0.081	216.38	18.61±2.85	0.492±0.119	0.496±0.058	148.61
DiffPIR	17.32±2.02	0.331±0.072	0.619±0.078	236.33	19.02±2.55	0.489±0.046	0.492±0.058	154.69
DDS	15.03±2.18	0.287±0.077	0.686±0.083	258.47	17.34±2.29	0.444±0.061	0.553±0.056	232.94
SSD	19.76±2.02	0.410±0.068	0.553±0.063	180.95	19.10±2.79	0.490±0.062	0.476±0.069	147.60
DDNM-Solver (DPM-1)	21.67±3.12	0.536±0.138	0.440±0.083	110.28	18.34±3.17	0.502±0.120	0.473±0.064	115.90
DDNM-Solver (DPM-2)	20.63±3.44	0.483±0.143	0.494±0.096	131.93	17.61±2.59	0.476±0.108	0.493±0.063	131.81
DDNM-Solver (UniPC-2)	**22.63**±3.16	**0.584**±0.139	**0.431**±0.085	**100.19**	**19.29**±3.16	**0.521**±0.118	**0.451**±0.064	**105.91**
DiffPIR-Solver (DPM-1)	21.43±3.02	0.568±0.141	0.448±0.085	110.50	18.29±3.16	0.498±0.117	0.475±0.068	112.91
DiffPIR-Solver (DPM-2)	21.61±3.12	0.544±0.143	0.457±0.088	116.43	18.22±2.94	0.481±0.115	0.489±0.063	128.97
DiffPIR-Solver (UniPC-2)	22.55±3.04	0.579±0.142	0.437±0.086	105.45	19.27±3.16	0.518±0.122	0.455±0.065	105.95

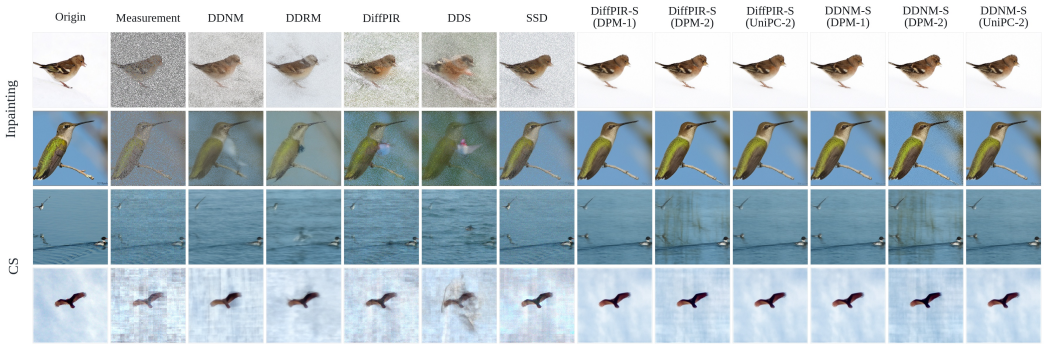

Figure 2: Qualitative results for inpainting with 70% random mask and compressed sensing on ImageNet utilizing NFE = 5. All measurements include additive Gaussian noise with $\sigma_y = 0.05$.

Table 7: Quantitative comparison for motion deblurring on CelebA and ImageNet at NFE = 5. We report PSNR, SSIM, LPIPS and FID. Methods without motion deblurring results on either dataset are omitted. We mark **bold** the best scores and underline the second best.

	CelebA				ImageNet			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DiffPIR	26.21±0.72	0.557±0.034	0.431±0.041	109.17	22.17±1.71	0.415±0.084	0.506±0.057	136.13
DDS	22.50±1.45	0.638±0.087	0.377±0.054	83.67	20.37±2.39	0.431±0.150	0.545±0.067	200.77
DiffPIR-Solver (DPM-1)	27.75±1.60	0.775±0.042	0.263±0.027	75.32	22.68±2.80	0.536±0.140	0.442±0.071	108.61
DiffPIR-Solver (DPM-2)	27.81±1.58	0.773±0.041	0.264±0.028	75.11	22.70±2.62	0.504±0.143	0.462±0.082	109.52
DiffPIR-Solver (UniPC-2)	**28.20**±1.59	**0.783**±0.042	**0.262**±0.028	**75.04**	**23.62**±2.78	**0.561**±0.141	**0.434**±0.072	**107.27**

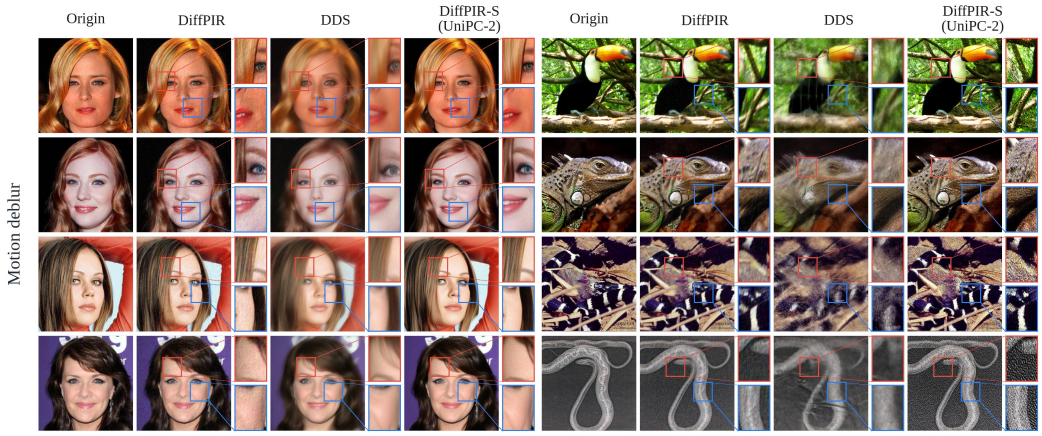

Figure 3: Qualitative results for motion deblurring on CelebA and ImageNet at NFE $= 5$.

Table 8: Quantitative comparison on CelebA at NFE = 10 for inpainting with a 70% random mask and compressed sensing. We report PSNR, SSIM, LPIPS and FID. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDRM	20.44±1.44	0.621±0.070	0.409±0.057	86.18	20.94±3.10	0.707±0.079	0.296±0.054	60.30
DDNM	20.30±1.29	0.617±0.065	0.414±0.050	87.94	20.87±2.73	0.710±0.065	0.295±0.050	58.94
DiffPIR	20.13±1.25	0.512±0.049	0.496±0.044	102.69	21.36±2.89	0.651±0.046	0.341±0.043	72.79
DDS	17.94±1.62	0.490±0.070	0.541±0.059	119.75	18.16±2.39	0.578±0.067	0.410±0.057	124.99
SSD	24.55±1.08	0.571±0.043	0.387±0.034	66.41	19.40±2.50	0.517±0.052	0.414±0.048	118.74
DDNM-Solver (DPM-1)	28.12±1.67	0.814±0.038	0.223±0.026	54.20	22.86±3.10	0.750±0.067	0.266±0.047	52.66
DDNM-Solver (DPM-2)	28.44±1.70	0.816±0.039	**0.221**±0.026	**52.79**	**23.38**±3.26	**0.757**±0.069	**0.258**±0.047	53.22
DDNM-Solver (UniPC-2)	**28.76**±1.63	**0.821**±0.038	0.222±0.026	53.96	23.12±3.16	0.755±0.067	0.262±0.046	**52.51**
DiffPIR-Solver (DPM-1)	27.93±1.57	0.808±0.042	0.232±0.028	56.84	22.66±3.03	0.744±0.068	0.272±0.047	56.01
DiffPIR-Solver (DPM-2)	27.89±1.65	0.807±0.043	0.229±0.027	55.63	23.14±3.18	0.749±0.071	0.265±0.047	56.25
DiffPIR-Solver (UniPC-2)	28.16±1.58	0.811±0.042	0.231±0.027	56.80	22.91±3.08	0.749±0.068	0.268±0.047	55.70

Table 9: Quantitative comparison on ImageNet at NFE = 10 for inpainting with a 70% random mask and compressed sensing. We report PSNR, SSIM, LPIPS and FID. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DDRM	18.72±2.09	0.393±0.099	0.588±0.070	196.38	19.70±3.20	0.566±0.103	0.430±0.060	112.64
DDNM	18.68±2.15	0.397±0.103	0.587±0.072	198.21	19.67±3.19	0.563±0.103	0.436±0.061	117.50
DiffPIR	18.41±1.99	0.374±0.065	0.589±0.068	204.92	20.12±2.93	0.564±0.096	0.422±0.052	97.67
DDS	16.83±2.11	0.316±0.077	0.630±0.075	224.29	17.86±2.54	0.465±0.056	0.510±0.064	189.67
SSD	20.35±3.25	0.423±0.101	0.504±0.088	130.68	19.38±2.78	0.488±0.050	0.454±0.064	114.55
DDNM-Solver (DPM-1)	22.23±3.29	0.588±0.126	0.412±0.075	89.21	19.59±3.57	0.539±0.100	0.421±0.063	100.74
DDNM-Solver (DPM-2)	22.02±3.61	0.578±0.141	0.413±0.079	89.29	19.62±3.78	0.540±0.107	0.424±0.069	100.90
DDNM-Solver (UniPC-2)	**23.12**±3.35	**0.615**±0.128	**0.401**±0.076	**88.13**	**20.15**±3.63	**0.567**±0.102	**0.411**±0.065	97.27
DiffPIR-Solver (DPM-1)	22.08±3.20	0.572±0.135	0.416±0.080	94.78	19.42±3.67	0.531±0.117	0.425±0.065	97.42
DiffPIR-Solver (DPM-2)	22.01±3.52	0.579±0.147	0.410±0.084	89.40	19.45±3.71	0.533±0.116	0.424±0.069	97.83
DiffPIR-Solver (UniPC-2)	23.00±3.26	0.610±0.137	0.404±0.081	91.33	20.01±3.57	0.561±0.114	0.413±0.066	**94.60**

Table 10: Quantitative comparison for motion deblurring on CelebA and ImageNet at NFE = 10. We report PSNR, SSIM, LPIPS and FID. Methods without motion deblurring results on either dataset are omitted. We mark **bold** the best scores and underline the second best.

	CelebA				ImageNet			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DiffPIR	28.43±1.38	0.720±0.029	0.278±0.034	66.88	22.65±2.30	0.455±0.095	0.463±0.054	98.91
DDS	25.62±1.75	0.748±0.059	0.257±0.034	55.61	21.83±2.85	0.480±0.147	0.465±0.062	131.94
DiffPIR-Solver (DPM-1)	28.78±1.71	0.805±0.043	0.229±0.026	51.59	23.02±2.84	0.514±0.159	0.448±0.098	94.54
DiffPIR-Solver (DPM-2)	28.25±1.84	0.800±0.046	**0.228**±0.026	51.54	22.17±2.82	0.499±0.165	0.457±0.119	95.11
DiffPIR-Solver (UniPC-2)	**28.94**±1.74	**0.809**±0.044	**0.228**±0.026	**51.39**	**23.34**±2.94	**0.518**±0.167	**0.438**±0.106	**92.95**

F.2 Comparison with additional accelerated baselines

We provide additional comparisons with accelerated inverse solvers that are not included in the main experiments. We evaluate C-ΠGDM, CCDF, and LLE-based variants of DDRM, DDNM, and DiffPIR on CelebA inpainting and compressed sensing at NFE $= 5$ and 10, with quantitative results reported in Tables 11 and 12 and qualitative comparisons shown in Figs. 4 and 5.

Table 11: Quantitative results with additional DM-based baselines and our accelerated solvers on CelebA inpainting and compressed sensing at NFE $= 5$. We report PSNR, SSIM, LPIPS and FID. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
C-ΠGDM Pandey et al. (2024)	17.22±1.10	0.286±0.036	0.653±0.033	160.22	18.27±2.01	0.435±0.041	0.519±0.037	182.53
CCDF Chung et al. (2022b)	13.43±1.64	0.234±0.048	0.733±0.041	300.52	–	–	–	–
DDRM (LLE) Zhang et al. (2025b)	19.61±0.99	0.468±0.045	0.548±0.051	105.13	19.14±2.19	0.639±0.064	0.368±0.045	76.46
DDNM (LLE) Zhang et al. (2025b)	19.65±0.94	0.470±0.043	0.545±0.050	102.27	19.32±2.25	0.635±0.053	0.372±0.054	74.93
DiffPIR (LLE) Zhang et al. (2025b)	19.89±1.01	0.420±0.033	0.560±0.038	116.55	20.00±2.38	0.510±0.047	0.461±0.045	120.91
DDNM-Solver (UniPC-2)	**27.67**±1.49	**0.797**±0.044	**0.254**±0.029	**68.34**	**20.91**±2.67	**0.689**±0.070	**0.324**±0.050	**71.42**
DiffPIR-Solver (UniPC-2)	27.13±1.45	0.788±0.047	0.262±0.031	69.51	20.88±2.66	0.687±0.070	**0.324**±0.050	72.42

Table 12: Quantitative results with additional DM-based baselines and our accelerated solvers on CelebA inpainting and compressed sensing at NFE $= 10$. We report PSNR, SSIM, LPIPS and FID. We mark **bold** the best scores and underline the second best.

	Inpainting (random)				Compressed sensing			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
C-ΠGDM Pandey et al. (2024)	18.76±1.27	0.391±0.051	0.586±0.039	140.52	18.72±2.15	0.498±0.045	0.462±0.041	164.21
CCDF Chung et al. (2022b)	13.43±1.65	0.223±0.043	0.734±0.039	303.99	–	–	–	–
DDRM (LLE) Zhang et al. (2025b)	20.93±0.94	0.595±0.079	0.384±0.049	79.38	20.86±3.02	0.705±0.058	0.295±0.052	61.07
DDNM (LLE) Zhang et al. (2025b)	21.20±0.95	0.604±0.069	0.388±0.047	80.64	20.49±1.80	0.711±0.059	0.299±0.046	59.68
DiffPIR (LLE) Zhang et al. (2025b)	22.78±1.13	0.596±0.033	0.407±0.037	85.02	21.26±2.65	0.658±0.043	0.340±0.042	73.19
DDNM-Solver (UniPC-2)	**28.76**±1.63	**0.821**±0.038	**0.222**±0.026	**53.96**	**23.12**±3.16	**0.755**±0.067	**0.262**±0.046	**52.51**
DiffPIR-Solver (UniPC-2)	28.16±1.58	0.811±0.042	0.231±0.027	56.80	22.91±3.08	0.749±0.068	0.268±0.047	55.70

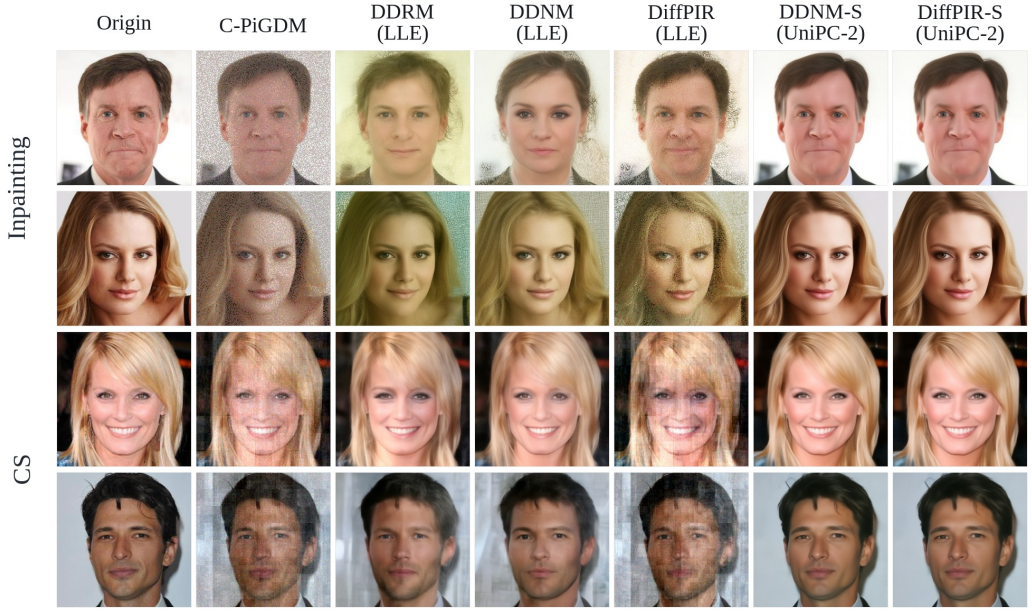

Figure 4: Qualitative comparison of C-ΠGDM, LLE-based DDRM, DDNM, and DiffPIR, SSD, and our UniPC-2 solvers on CelebA at NFE $= 5$. The upper two rows show inpainting with a 70% random mask; the lower two rows show compressed sensing with a 50% measurement ratio.



Figure 5: Qualitative comparison of C-ΠGDM, LLE-based DDRM, DDNM, and DiffPIR, SSD, and our UniPC-2 solvers on CelebA at NFE $= 10$. The upper two rows show inpainting with a 70% random mask; the lower two rows show compressed sensing with a 50% measurement ratio.

F.3 Ablation Study

We investigate the interaction between the stochasticity parameter τ and the guidance schedule γ_t. We utilize a linear guidance schedule $\gamma_t = \gamma_0 + (\gamma_1 - \gamma_0)t$ with $\gamma_1 = 1$, and vary γ_0 and τ for DDNM-Solver and DiffPIR-Solver with UniPC-2 at NFE $= 5$. Figures 6 and 7 show qualitative results for CelebA inpainting and compressed sensing, respectively. Tables 13 and 14 report the corresponding quantitative results for inpainting, while Tables 15 and 16 report those for compressed sensing. Random inpainting benefits from stronger initial guidance, whereas compressed sensing generally favors moderate initial guidance. Under suitable guidance strengths, increasing τ generally improves perceptual quality in the evaluated few-step settings.

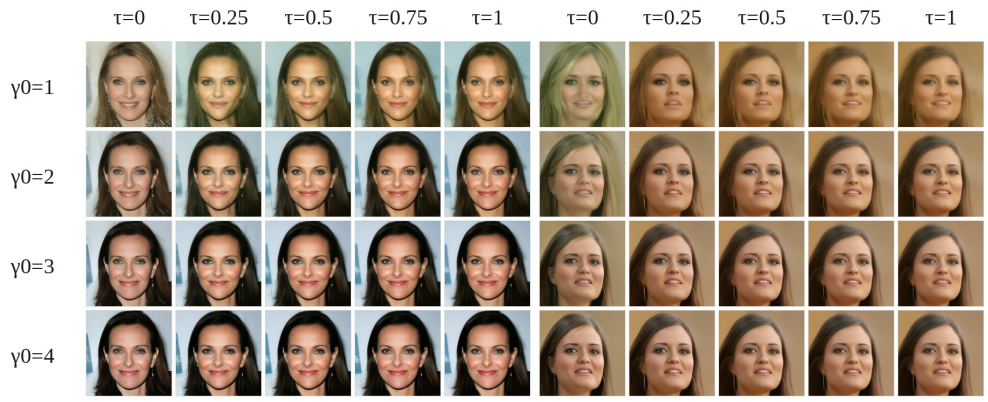

Figure 6: Qualitative joint ablation of stochasticity τ and initial guidance strength γ_0 for CelebA inpainting at NFE = 5 with DiffPIR-Solver (UniPC-2). Columns vary τ from 0 to 1, and rows vary γ_0 while the guidance endpoint is fixed as $\gamma_1 = 1$.

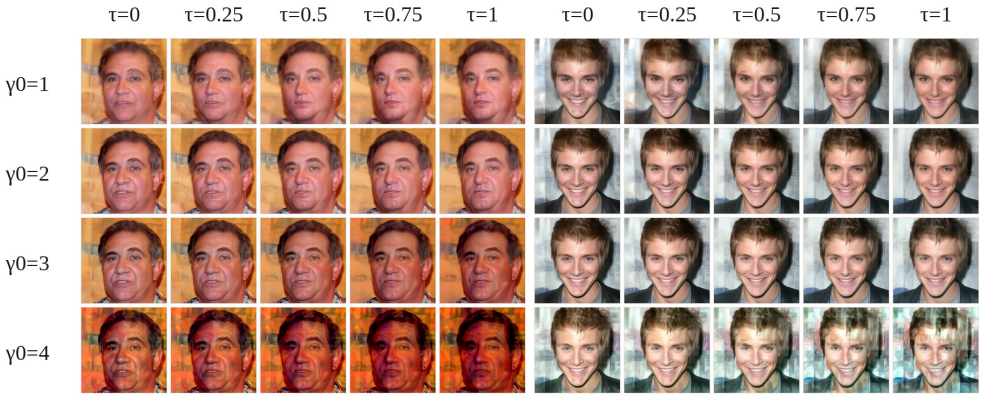

Figure 7: Qualitative joint ablation of stochasticity τ and initial guidance strength γ_0 for CelebA compressed sensing at NFE = 5 with DiffPIR-Solver (UniPC-2). Columns vary τ from 0 to 1, and rows vary γ_0 while the guidance endpoint is fixed as $\gamma_1 = 1$.

Table 13: Joint ablation of stochasticity τ and the initial guidance strength γ_0 utilizing DDNM-Solver (UniPC-2) on CelebA inpainting with a 70% random mask at NFE = 5. The guidance strength is linearly scheduled to the fixed endpoint $\gamma_1 = 1$. We mark **bold** the best scores and underline the second best.

	$\gamma_0 = 1$				$\gamma_0 = 2$				$\gamma_0 = 3$				$\gamma_0 = 4$			
τ	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
0	17.45	0.480	0.573	107.53	21.90	0.616	0.411	82.29	25.22	0.703	0.325	73.61	26.86	0.742	0.291	69.38
0.25	17.44	0.491	0.572	108.61	22.12	0.632	0.404	82.72	25.43	0.717	0.318	75.27	27.02	0.754	0.285	71.12
0.5	17.91	0.531	0.546	99.83	22.69	0.669	0.377	81.44	25.82	0.745	0.302	74.13	27.26	0.775	0.274	70.89
0.75	18.40	0.571	0.515	95.21	23.23	0.698	0.351	78.91	26.11	0.764	0.287	72.17	**27.40**	0.790	0.264	69.28
1	**18.58**	**0.592**	**0.491**	**90.39**	**23.43**	**0.707**	**0.333**	**76.82**	**26.13**	**0.767**	**0.279**	**71.18**	27.30	**0.791**	**0.259**	**69.02**

Table 14: Joint ablation of stochasticity τ and the initial guidance strength γ_0 utilizing DiffPIR-Solver (UniPC-2) on CelebA inpainting with a 70% random mask at NFE = 5. The guidance strength is linearly scheduled to the fixed endpoint $\gamma_1 = 1$. We mark **bold** the best scores and underline the second best.

	$\gamma_0 = 1$				$\gamma_0 = 2$				$\gamma_0 = 3$				$\gamma_0 = 4$			
τ	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
0	16.82	0.500	0.562	97.14	21.32	0.627	0.395	81.74	24.77	0.711	0.312	74.45	26.56	0.752	0.280	69.75
0.25	16.82	0.510	0.564	102.07	21.53	0.641	0.389	82.19	24.97	0.723	0.305	75.04	26.69	0.762	0.275	70.22
0.5	17.32	0.541	0.539	96.81	22.11	0.668	0.367	79.16	25.35	0.743	0.293	73.41	26.89	0.777	0.267	70.35
0.75	17.90	0.569	0.516	93.07	22.72	0.689	0.349	77.32	25.68	0.756	0.286	71.82	27.03	0.785	0.264	**68.96**
1	**18.37**	**0.588**	**0.500**	**91.36**	**23.19**	**0.701**	**0.340**	**77.17**	**25.92**	**0.763**	**0.283**	**71.75**	**27.13**	**0.788**	**0.262**	69.51

Table 15: Joint ablation of stochasticity τ and the initial guidance strength γ_0 utilizing DDNM-Solver (UniPC-2) on CelebA compressed sensing at NFE = 5. The guidance strength is linearly scheduled to the fixed endpoint $\gamma_1 = 1$. We mark **bold** the best scores and underline the second best.

	$\gamma_0 = 1$				$\gamma_0 = 2$				$\gamma_0 = 3$				$\gamma_0 = 4$			
τ	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
0	18.26	0.571	0.419	100.00	19.87	0.626	0.368	81.86	**19.27**	0.614	**0.420**	97.38	**16.16**	**0.513**	**0.597**	**214.29**
0.25	18.49	0.585	0.407	91.54	20.08	0.639	0.359	78.64	19.20	0.621	0.427	94.65	15.80	0.511	0.612	218.43
0.5	18.90	0.611	0.387	83.94	20.43	0.663	0.343	74.12	18.62	**0.626**	0.459	**94.53**	14.74	0.495	0.655	237.69
0.75	19.33	0.636	0.366	**77.13**	20.78	0.685	0.329	**70.40**	17.49	0.616	0.511	102.55	13.31	0.461	0.710	267.65
1	**19.58**	**0.641**	**0.355**	77.67	**20.91**	**0.689**	**0.324**	71.42	15.98	0.582	0.575	108.80	11.58	0.398	0.766	291.28

Table 16: Joint ablation of stochasticity τ and the initial guidance strength γ_0 utilizing DiffPIR-Solver (UniPC-2) on CelebA compressed sensing at NFE = 5. The guidance strength is linearly scheduled to the fixed endpoint $\gamma_1 = 1$. We mark **bold** the best scores and underline the second best.

	$\gamma_0 = 1$				$\gamma_0 = 2$				$\gamma_0 = 3$				$\gamma_0 = 4$			
τ	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
0	18.00	0.574	0.405	89.71	19.69	0.636	0.356	79.63	**19.10**	0.626	**0.416**	89.45	**15.52**	**0.509**	**0.611**	**196.77**
0.25	18.23	0.585	0.396	83.57	19.89	0.646	0.348	75.83	19.02	**0.630**	0.425	**86.62**	15.19	0.505	0.625	199.39
0.5	18.65	0.605	0.381	83.34	20.23	0.663	0.337	74.90	18.52	0.629	0.456	86.74	14.30	0.485	0.662	218.82
0.75	19.12	0.624	0.368	79.94	20.60	0.677	0.329	73.19	17.67	0.616	0.500	94.38	13.28	0.456	0.705	245.60
1	**19.49**	**0.638**	**0.359**	**78.35**	**20.88**	**0.687**	**0.324**	**72.42**	16.93	0.599	0.538	101.90	12.51	0.431	0.736	267.80

F.4 RUNTIME EFFICIENCY

We evaluate the inference cost over 100 images on a single NVIDIA RTX 4090 GPU. Inpainting uses coordinate selection and permutation, compressed sensing uses the Fast Walsh–Hadamard Transform with sampled coordinates, and motion deblurring uses the FFT-based implementation described in Appendix D. The operator-dependent representations are determined once for each task configuration and reused throughout sampling.

Both DiffPIR and DDS use iterative computations for data consistency in their original implementations, whereas our method computes the corresponding operator-dependent terms directly. To separate the cost of these iterative computations from that of the overall sampling procedure, we additionally evaluate DiffPIR-CF and DDS-CF as implementation controls. These variants replace only the iterative data-consistency computations with closed-form solutions for the structured forward operators considered here, while retaining the remaining sampling procedures. For motion deblurring, where the forward operator is not explicitly constructed, the closed-form computations are implemented in the Fourier domain using FFTs. The iterative computation in DiffPIR can introduce a small approximation error relative to its closed-form counterpart, leading to slightly different sampling results, whereas the CG solution in DDS closely matches its closed-form counterpart and yields nearly identical reconstructions.

Table 17 reports the average runtime per image per reverse timestep at NFE $= 10$. To verify that the closed-form implementations provide comparable data-consistency corrections, Table 18 reports reconstruction quality on CelebA at 10 steps. The iterative and closed-form implementations yield comparable reconstruction quality across the three inverse tasks.

Table 17: Average runtime per image per reverse timestep on CelebA and ImageNet at NFE $= 10$, reported in seconds.

	CelebA			ImageNet		
Method	Inpainting	CS	Motion	Inpainting	CS	Motion
DDRM	0.0220	0.0279	–	0.0889	0.0990	–
DDNM	0.0237	0.0332	–	0.0901	0.1037	–
DiffPIR	0.1079	0.8356	0.0811	0.1760	0.9926	0.1491
DiffPIR-CF	0.0254	0.0295	0.0245	0.0923	0.1030	0.0923
DDS	0.0237	0.0409	0.0232	0.0918	0.1087	0.0906
DDS-CF	0.0219	0.0251	0.0216	0.0890	0.0937	0.0876
DDNM-Solver (UniPC-2)	0.0232	0.0269	–	0.0903	0.0954	–
DiffPIR-Solver (UniPC-2)	0.0229	0.0271	0.0222	0.0928	0.0955	0.0903

Table 18: Quantitative results with the iterative and closed-form data-consistency implementations on CelebA at NFE $= 10$.

	Inpainting (random)				Compressed sensing				Motion deblurring			
Method	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓	PSNR↑	SSIM↑	LPIPS↓	FID↓
DiffPIR	20.13	0.512	0.496	102.69	21.36	0.651	0.341	72.79	28.43	0.720	0.278	66.88
DiffPIR-CF	19.97	0.506	0.501	103.21	21.69	0.657	0.336	70.24	28.47	0.723	0.274	66.00
DDS	17.94	0.490	0.541	119.75	18.16	0.578	0.410	124.99	25.62	0.748	0.257	55.61
DDS-CF	17.94	0.490	0.541	119.73	18.16	0.578	0.410	124.98	25.62	0.748	0.257	55.63