FROM DECODER BACKWARD TO ENCODER FORWARD: ANTISYMMETRIC LATENT UPDATES FOR DIFFUSION INVERSE SOLVERS

Anonymous authors
Paper under double-blind review

ABSTRACT

High-dimensional and high-resolution inverse problems are difficult to solve with latent diffusion inverse solvers (DIS) because measurement consistency is imposed in pixel space, while the generative prior evolves in latent space. Thus, every sampling step must convert a pixel-space correction Δx into a latent update Δz. Most existing methods perform this conversion by backpropagating through the decoder, which dominates runtime and memory at high resolution; forward-only alternatives such as direct re-encoding avoid this cost but distort the latent prior enough to degrade reconstruction quality. We propose **Antisymmetric Latent Update** (ASLU), $\Delta z = \rho \cdot \frac{1}{2}[\mathcal{E}(x+\alpha\Delta x) - \mathcal{E}(x-\alpha\Delta x)]$: an antisymmetric difference of two encoder forward passes that vanishes at $\Delta x = 0$, preventing accumulated drift, and cancels second-order encoder-curvature bias while preserving the intended first-order Jacobian response. Because ASLU consumes only Δx—never a gradient of the decoder or the measurement operator—the same rule applies whatever DIS mechanism produces the correction. Controlled experiments show that curvature cancellation matters increasingly for larger corrections, while drop-in use in high-resolution image inverse problems and high-dimensional black-hole video reconstruction reduces peak memory by up to 64% and 76%, respectively. On an independent sparse-view CT benchmark against multiple latent DIS methods, ASLU maintains competitive reconstruction quality in a low-memory, short-runtime region not reached by the evaluated baselines.

1 INTRODUCTION

Many inverse problems that matter in practice are difficult because the unknown state is large, the measurements are non-standard, or both. Weather and climate reanalysis, turbulent-flow recovery, dynamic clinical MRI, and interferometric black-hole imaging all ask us to recover an unknown physical state from incomplete, indirect, or noisy observations at megapixel, volumetric, or video scale (Zhuang & Duraisamy, 2025; Steinbrenner et al., 2026; Zhang et al., 2025b; Akiyama et al., 2022; Rozet & Louppe, 2023; Jia et al., 2026).

Recent advances in generative AI, especially diffusion and flow-based models, make it possible to learn expressive data priors directly from examples. Diffusion inverse solvers (DIS) use these priors at inference time to produce samples that are both realistic under the learned data distribution and consistent with observed measurements (Chung et al., 2023a; Song et al., 2022; Daras et al., 2024a; Zheng et al., 2025). This is attractive for inverse problems because it replaces hand-designed regularizers with learned image or physical-field statistics, while still allowing the measurement model to be imposed only at test time. However, DIS methods typically require repeatedly translating measurement information into an update of the generative trajectory, so their practicality depends on how this translation is performed.

Latent diffusion and latent flow models (Rombach et al., 2022; Esser et al., 2024) improve scalability by moving the generative trajectory into a compressed VAE representation, and this latent formulation has become a promising route for high-resolution image and scientific-data generation (Kim et al., 2025; Zhang et al., 2025b; Song et al., 2024). The latent representation, however, creates a space mismatch: the prior evolves in latent space, whereas measurements and data-consistency

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.

corrections live in pixel or physical space. Existing latent DIS methods often bridge this mismatch by backpropagating a data-fidelity loss through the VAE decoder at every guidance step. For high-resolution images or high-dimensional scientific data, decoder-backward guidance becomes the dominant cost: in our measurements, it reaches 40.3 GB in super-resolution of 2K RGB image with FlowDPS (Kim et al., 2025), and 37.1 GiB in black-hole video reconstruction with STeP (Zhang et al., 2025b). To scale up, a more efficient method is needed.

This paper asks: when a DIS produces a pixel-space correction Δx, how should it be converted into a latent update Δz? Existing decoder-gradient-free alternatives trade one failure mode for another: direct re-encoding carries a zeroth-order drift that persists even when $\Delta x = 0$, while a one-sided encoder probe removes that drift but retains a second-order encoder-curvature bias that grows with the size of the correction. Our main finding is simple: **Antisymmetric Latent Update (ASLU)**, $\rho \cdot \frac{1}{2}[\mathcal{E}(x+\alpha\Delta x) - \mathcal{E}(x-\alpha\Delta x)]$, uses only ordinary forward evaluations of the encoder, bypassing the decoder's costly backward pass entirely, and is exactly zero whenever $\Delta x = 0$ so no drift compounds across sampling steps—while its centered, antisymmetric construction also cancels the second-order curvature bias as a further benefit, preserving the first-order Jacobian term the guidance is meant to deliver.

Contributions.

1. We formulate pixel-to-latent conversion as an update-design problem and show that existing approaches incur decoder-backward cost, zeroth-order cycle drift, or second-order curvature bias, and that the exact encoder JVP is not deployable at scale.
2. We propose ASLU, a forward-only encoder update that is zero at $\Delta x = 0$, cancels even-order terms, and reduces the leading approximation error from quadratic to cubic order, which in turn makes large single-shot corrections viable.
3. We validate ASLU through controlled correction-scale ablations, drop-in image and video integrations across two DIS mechanisms, and an independent sparse-view CT benchmark against latent-DIS baselines.

2 Background and Problem Setup

We study imaging inverse problems of the form

$$y = A(x^\star) + e, \tag{1}$$

where $x^\star \in \mathbb{R}^n$ is an unknown image, $A : \mathbb{R}^n \to \mathbb{R}^m$ is a known, ill-posed forward operator, and e denotes measurement noise.

Because this problem generally admits many solutions, we follow the common approach of sampling from the posterior $p(x \mid y) = p(x)p(y \mid x)/p(y)$ given by Bayes' rule. Classically, the likelihood $p(y \mid x)$ was made explicit by assuming, e.g., Gaussian measurement noise e, and the prior $p(x)$ was likewise made explicit through hand-designed regularizers such as total variation (TV), with a solution obtained by optimizing the resulting objective, e.g. with ADMM (Boyd et al., 2011). Generative models instead learn the data distribution directly, giving a samplable model of $p(x)$; Bora et al. (2017) were the first to use one for Eq. (1).

2.1 Diffusion inverse solvers

Generative models have since advanced rapidly, and diffusion models—which model priors such as natural images particularly well—have given rise to the large family of DIS methods, which combine a pretrained diffusion prior $p(x)$ with the known forward operator A to perform inference for the posterior $p(x \mid y)$ (Daras et al., 2024a; Chung et al., 2025; Zheng et al., 2025). DIS methods differ mainly in the mechanism that imposes measurement consistency on the diffusion sampling trajectory: gradient guidance (Chung et al., 2023a; Song et al., 2023; Rozet et al., 2024), projection (Jalal et al., 2021; Song et al., 2021; Chung & Ye, 2022; Kawar et al., 2021; 2022; Wang et al., 2023), variable splitting (Zhang et al., 2025a; Wu et al., 2024; Coeurdoux et al., 2024), sequential Monte Carlo (Cardoso et al., 2024; Dou & Song, 2024), and variational optimization (Mardani et al., 2024). Despite these differences, each mechanism produces, at every sampling step, a pixel-space correction that moves the current estimate toward measurement consistency; we denote this correction Δx.

Diffusion posterior sampling (DPS) (Chung et al., 2023a) is the representative gradient-guidance instance: for Gaussian measurement noise e, its correction takes the form

$$\Delta x \;=\; -\rho \nabla_x \|y - A(x)\|_2^2, \tag{2}$$

where ρ is a step-size hyperparameter.

2.2 Latent diffusion/flow inverse solvers

Learning $\nabla_{x_t} \log p(x_t)$ directly for high-dimensional data is difficult, which motivated training diffusion models in a lower-dimensional latent space instead. Let $\mathcal{E} : \mathbb{R}^n \to \mathbb{R}^k$ and $\mathcal{D} : \mathbb{R}^k \to \mathbb{R}^n$ denote the encoder and decoder of a pretrained VAE, and let z_t be the latent state at noise level t of a diffusion or flow-matching model trained in this latent space (Rombach et al., 2022).This has become the standard way to model high-resolution or high-dimensional data with diffusion or flow models, and is referred to as a latent diffusion/flow model.

DIS methods have since been extended to these latent models by applying an analogous correction to the model's clean-latent estimate $\hat{z}_{0|t}$, rather than to the noisy state z_t itself (Rout et al., 2024; Song et al., 2024; Kim et al., 2025; Zhang et al., 2025b). FlowDPS, for example, forms $\hat{z}_{0|t}$ from z_t via its flow-matching update rule and adds the term $-\rho_t \nabla_{\hat{z}_{0|t}} \|y - A(\mathcal{D}(\hat{z}_{0|t}))\|_2$ before continuing the trajectory (Kim et al., 2025). Write $z := \hat{z}_{0|t}$ for this clean-latent estimate, $x := \mathcal{D}(z)$ for its decoded pixel proposal, and $x^{\mathrm{dc}} := x + \Delta x$ for the measurement-improved pixel proposal of Eq. (2); the updated latent is then $z^{\mathrm{dc}} := z + \Delta z$. Existing work obtains z^{dc} in one of two ways: *decoder-backward guidance* or *direct re-encoding*.

Decoder-backward guidance. The most direct strategy differentiates the measurement loss through the decoder and applies the vector-Jacobian product $J_{\mathcal{D}}(z)^\top \Delta x$ in latent space via backpropagation (Kim et al., 2025; Zhang et al., 2025b). This strategy is not tied to one DIS mechanism: latent gradient guidance (Rout et al., 2024; Kim et al., 2025) and variable-splitting samplers whose inner MCMC uses likelihood gradients (Zhang et al., 2025b) both rely on it, because each needs the measurement loss differentiated with respect to the latent. This update is accurate when memory is available, but it retains the full decoder activation graph at every guidance step; the resulting memory footprint scales with physical resolution and temporal length, making this option the computational bottleneck for 2K images and long scientific videos.

Direct re-encoding. The simplest forward-only alternative replaces the current latent outright,

$$z^{\mathrm{dc}} = \mathcal{E}(x^{\mathrm{dc}}) = \mathcal{E}(x + \Delta x). \tag{3}$$

This is cheap, since it avoids backpropagating through the decoder, but Eq. (3) only crudely approximates the correction that decoder-backward guidance would apply to z: the encoder and decoder are not exact inverses of each other, so $\mathcal{E}(\mathcal{D}(z)) \neq z$ in general, and Eq. (3) implicitly assumes $\mathcal{E}(\mathcal{D}(z)) = z$. Chung et al. (2024) report that $\mathcal{E} \circ \mathcal{D} \circ \mathcal{E} \circ \mathcal{D} \circ \cdots \circ \mathcal{D}(z)$ diverges, and further Hong et al. (2024) showed $\mathcal{E}(\mathcal{D}(z)) \neq z$ in diverse VAEs for LDMs. Song et al. (2024) address this by using direct re-encoding early in sampling, when accuracy matters less, and switching to the more expensive but accurate decoder-backward guidance later, when it matters more—so the cost–accuracy trade-off remains unresolved. Li et al. (2024) instead purify the re-encoded latent by adding noise and running diffusion denoising on it; this suppresses re-encoding artifacts, but the denoiser can hallucinate content that the measurement does not support.

2.3 Practical limits of decoder-backward guidance

The cost above is not hypothetical. It already dominates in high-dimensional inverse problems, where the limiting factor is how to convert a pixel-space correction into a latent update without paying for a full decoder backward pass.

High-dimensional and spatiotemporal problems. Inverse problems over video-length sequences, rather than a single image, push this further: the dimension of the state to recover grows with the number of frames until posterior sampling becomes infeasible in a single pass. Existing work sidesteps this either by applying a 2D image prior frame by frame (Chung et al., 2023b;

Table 1: **Taxonomy of methods for lifting pixel-space updates Δx to latent-space updates Δz in latent DIS.** Rows distinguish methods that require derivative propagation through $\mathcal{D}$ or $\mathcal{E}$ (reverse-mode backward or forward-mode JVP) from those using only ordinary forward evaluations. Columns indicate whether the update Δz is zero at $\Delta x = 0$. One-sided differencing also reaches the bottom-right cell at a fixed cost, but retains a second-order curvature term. Our ASLU removes both zeroth-order drift and the second-order curvature term using only ordinary forward evaluations.

	$\Delta z \neq 0$ at $\Delta x = 0$	$\Delta z = 0$ at $\Delta x = 0$
w/ Derivative (heavy computation)	*(none)*	**Decoder-backward guidance:** $\Delta z = \rho\, J_{\mathcal{D}}(z)^{\top} \Delta x$ **Exact JVP:** $\Delta z = \rho \cdot J_{\mathcal{E}}(x) \Delta x$
Forward-only (efficient)	**Direct re-encoding:** $z^{\mathrm{dc}} = \mathcal{E}(x + \alpha \Delta x)$	**One-sided** (Chang et al., 2025): $\rho\big[\mathcal{E}(x+\alpha\Delta x) - \mathcal{E}(x)\big]$ – keeps the second-order curvature bias
		ASLU (Ours): $\Delta z = \rho \cdot \frac{1}{2}\big[\mathcal{E}(x+\alpha\Delta x) - \mathcal{E}(x-\alpha\Delta x)\big]$ + same cost, curvature bias removed too

Daras et al., 2024b; Kwon & Ye, 2025) or by relying on the less accurate 'direct re-encoding' described above (Spagnoletti et al., 2026). This is distinct from some medical-imaging settings, where high-dimensional or high-resolution priors are often limited more by training-data scarcity than by inference-time computation (Hu et al., 2024; 2025); our focus is instead on natural images and domains where data can be generated and priors can be trained through simulation, so posterior guidance becomes compute-limited rather than data-limited. Even when methods attempt joint high-dimensional guidance, they typically shrink the problem before sampling: STeP models only 64 frames at 256×256 yet already uses 37 GiB (Zhang et al., 2025b); weather models use temporal windowing to keep score computation tractable (Andry et al., 2025); and PDE inverse-problem methods often work on downsampled grids (Qu et al., 2024; Huang et al., 2024; Chen et al., 2026; Wang & Tartakovsky, 2026). Thus, the dimension one would like to model and the dimension one can afford to guide remain far apart.

3 METHOD

We analyze how pixel-space data-consistency information (Δx) should be transferred into latent space (Δz), reusing the notation z, x, Δx, x^{dc}, and $z^{\mathrm{dc}} = z + \Delta z$ from Section 2.2. Decoder-backward guidance has been effective across latent DIS frameworks (Kim et al., 2025; Zhang et al., 2025b), but Section 2.3 shows why its activation memory and runtime become increasingly burdensome as image resolution or temporal dimension grows. We therefore seek a practical replacement that uses only ordinary forward evaluations, without derivative propagation through either $\mathcal{D}$ or $\mathcal{E}$. Because the update should represent the pixel correction rather than alter the latent independently of it, we also require $\Delta z = 0$ whenever $\Delta x = 0$.

Table 1 organizes existing strategies by these two properties: whether they require derivative propagation and whether $\Delta x = 0 \Rightarrow \Delta z = 0$.

Direct re-encoding introduces a zeroth-order drift (Table 1, bottom-left). Direct re-encoding (Eq. (3)) is the cheapest forward-only option, requiring one encoder pass, but it can change the latent even when no pixel correction is requested. Expanding $\mathcal{E}(x+\Delta x)$ around x writes its deviation from z as a polynomial in Δx:

$$z^{\mathrm{dc}} - z = \underbrace{\mathcal{E}(x) - z}_{\text{zeroth order}} + J_{\mathcal{E}}(x)\Delta x + \tfrac{1}{2} H_{\mathcal{E}}(x)[\Delta x, \Delta x] + O(\|\Delta x\|^3). \tag{4}$$

Because $x = \mathcal{D}(z)$, the zeroth-order term is $\mathcal{E}(\mathcal{D}(z)) - z$, nonzero precisely because $\mathcal{E} \circ \mathcal{D} \neq \mathrm{Id}$: direct re-encoding carries this drift even at $\Delta x = 0$, and it compounds once repeated at every sampling step (Chung et al., 2024; Hong et al., 2024). The same expansion also retains a curvature term of order $O(\|\Delta x\|^2)$ that grows with the size of the correction—a second, independent problem that we return to below. Gradient-free decoder inversion (Hong et al., 2024) can reduce this re-encoding error through additional forward iterations, trading drift for higher computation (Appendix C.3).

Exact encoder JVP avoids finite-difference error, but is not a practical forward-only replacement (Table 1, top-right). Decoder-backward guidance computes the vector–Jacobian product (VJP) $J_{\mathcal{D}}(z)^\top \Delta x$ by backpropagating through the decoder. A natural proxy is to transport the pixel correction through the encoder tangent instead, using the Jacobian–vector product (JVP) $J_{\mathcal{E}}(x)\Delta x$. Although $J_{\mathcal{E}}(x)\Delta x \neq J_{\mathcal{D}}(z)^\top \Delta x$, both are linear responses to Δx. However, an exact JVP is not an ordinary encoder forward pass: it requires forward-mode automatic differentiation support for every encoder operation. In Appendix C.4, we show `torch.func.jvp` fails, and even after we fix it, it costs a lot of GPU memory and runtime. Exact JVP is therefore useful as a correctness oracle, but impractical as a deployment choice in the high-resolution and high-dimensional regimes of interest.

A one-sided encoder update removes the drift, but leaves a residual (Table 1, bottom-right). The key problem with direct re-encoding is that $\Delta z \neq 0$ even when $\Delta x = 0$. A related construction in Chang et al. (2025), although not developed for DIS, removes this bias by subtracting the unperturbed encoding $\mathcal{E}(x)$. For consistency with our method, we introduce scale parameters α and ρ and write the resulting update as $\Delta z = \rho\big[\mathcal{E}(x{+}\alpha\Delta x) - \mathcal{E}(x)\big]$. We refer to this construction as the *one-sided difference*. It therefore meets both practical requirements above using two encoder forward passes. Nonetheless, Taylor expansion of the one-sided difference around x reveals why it is not the right choice:

$$\mathcal{E}(x + \alpha\Delta x) - \mathcal{E}(x) = \alpha J_{\mathcal{E}}(x)\Delta x + \tfrac{1}{2}\alpha^2 H_{\mathcal{E}}(x)[\Delta x, \Delta x] + O(\alpha^3 \|\Delta x\|^3). \tag{5}$$

The first-order term is the desired response, whereas the second-order term is an unwanted curvature bias. Decoder-backward guidance, which we seek to replace, is linear in Δx and contains no corresponding second-order term. This motivates a new update that removes the second-order term.

ASLU removes the second-order bias at no extra cost (Table 1, bottom-right). We replace the one-sided difference with a centered difference, which requires the same two encoder forward passes:

$$\Delta z = \rho \cdot \tfrac{1}{2}\big[\mathcal{E}(x + \alpha\Delta x) - \mathcal{E}(x - \alpha\Delta x)\big]. \tag{6}$$

As a function of Δx, this update is antisymmetric (odd): $\Delta z(-\Delta x) = -\Delta z(\Delta x)$. We therefore call it the *Antisymmetric Latent Update* (ASLU). Antisymmetry cancels all even-order terms, including the second-order curvature bias in eq. (5). For smooth $\mathcal{E}$,

$$\Delta z = \rho\alpha\, J_{\mathcal{E}}(x)\, \Delta x + O(\rho\alpha^3 \|\Delta x\|^3). \tag{7}$$

Thus, ASLU preserves the desired first-order response while eliminating the $O(\alpha^2 \|\Delta x\|^2)$ bias for any α and ρ. Like the one-sided construction, it uses only ordinary encoder forward passes and is exactly zero when $\Delta x = 0$, but removes the second-order bias at no additional cost.

Because ASLU acts only on Δx, it is agnostic to the DIS mechanism that produces the correction. In section 4, we evaluate it in three settings: gradient guidance in FlowDPS on high-resolution images, variable splitting with gradient-based HMC in STeP on high-dimensional video, and PSLD measurement guidance on sparse-view CT under the DM4CT benchmark.

4 Experiments

We evaluate ASLU at three levels: a controlled comparison of latent-update rules, drop-in replacements in FlowDPS and STeP, and comparison with latent DIS methods on the independently developed DM4CT benchmark. These respectively test the design choice, scaling across high resolution and high dimension, and ASLU's position in the broader quality–compute landscape.

4.1 Controlled validation of the latent-update design

We first compare the latent-update strategies introduced in Section 3, using FlowDPS's FFHQ Gaussian-deblurring setting at 1024^2 resolution. FlowDPS applies $K{=}3$ incremental corrections based on $-\nabla_x \|y - Ax\|_2$ at each diffusion timestep, producing a relatively small accumulated Δx through an inner loop. We additionally consider a one-shot analytic-backprojection correction, $\Delta x = A^\top(y - Ax)$, which removes this inner loop and moves directly in the least-squares descent

direction, but produces a substantially larger correction. This gives a controlled small-versus-large-$\|\Delta x\|$ comparison: since (α, ρ) is fixed per pipeline while $\|\Delta x\|$ varies by orders of magnitude along the trajectory, the one-sided update's $O(\|\Delta x\|^2)$ curvature bias cannot be tuned away, and we expect ASLU's advantage to grow in the large-correction regime.

Table 2: **Controlled comparison of pixel-to-latent update strategies.** Direct re-encoding trades quality for speed, while gradient-free decoder inversion is iterative and slow. Under matched Δx, ASLU improves over the one-sided update most strongly for a large correction, consistent with the latter's $O(\|\Delta x\|^2)$ curvature bias. FFHQ Gaussian deblurring (σ_k=3, kernel 61×61, 1024^2, 100 images); time and peak memory are per image on one A6000. ↑: higher is better; ↓: lower is better.

Method	PSNR↑	SSIM↑	FID↓	LPIPS↓	Time (s/img)↓	Mem (GB)↓
Decoder-backward (FlowDPS)	26.81	0.632	68.1	0.154	52.9	17.1
Direct re-encoding	24.92	0.598	134.4	0.180	**16.7**	**8.3**
Gradient-free decoder inversion	26.06	0.672	109.0	0.169	112.6	**8.3**
Latent update, $\Delta x = A^\top(y - Ax)$ (analytic backprojection; large $\|\Delta x\|$)						
One-sided (Eq. 5)	25.60	0.541	73.2	0.162	20.0	**8.3**
ASLU	27.05	0.641	**62.5**	**0.135**	20.1	**8.3**
Latent update, Δx via K=3 steps of $-\nabla_x\|y - Ax\|_2$ (FlowDPS rule; small $\|\Delta x\|$)						
One-sided (Eq. 5)	27.37	0.743	103.9	0.162	21.1	**8.3**
ASLU	**27.60**	**0.757**	110.1	0.160	21.1	**8.3**

Table 2 shows that direct re-encoding loses substantial quality, while ASLU reduces runtime of decoder-backward FlowDPS by 62% while matching (or exceeding) quality metrics. With matched large Δx, ASLU improves over one-sided differencing by 1.45 dB and leads on every perceptual metric; with the smaller FlowDPS correction, the gap narrows to 0.23 dB, consistent with the predicted quadratic scaling of the removed bias. This trend supports the Taylor analyses in Eqs. (5) and (7): the one-sided curvature error grows quadratically with the correction magnitude, whereas ASLU cancels that term. Appendix B.2 provides a qualitative comparison of the same latent-update strategies on a representative example.

4.2 Drop-in scaling of established inverse solvers

We next replace only the decoder-backward conversion inside FlowDPS (Kim et al., 2025) and STeP (Zhang et al., 2025b), choosing α and ρ once per pipeline as detailed in the appendix. FlowDPS stresses spatial resolution, whereas STeP stresses temporal dimension and uses a different inference mechanism.

4.2.1 High-resolution image inverse problems: FlowDPS

FlowDPS's flow trajectory update includes the term $-\rho_t \nabla_{\hat{z}_{0|t}} \|y - A(\mathcal{D}(\hat{z}_{0|t}))\|_2$; we replace this decoder-backward data-consistency step with ASLU,

$$\Delta x = -\nabla_{\hat{x}_{0|t}} \|y - A(\hat{x}_{0|t})\|_2, \qquad \Delta z = \rho \cdot \tfrac{1}{2}\big[\mathcal{E}(x{+}\alpha\Delta x) - \mathcal{E}(x{-}\alpha\Delta x)\big], \tag{8}$$

applied at each of K=3 inner steps (i.e., three updates $z \leftarrow z + \Delta z$ per diffusion timestep t). Table 3a shows that ASLU reduces GPU memory usage consistently and substantially: by about 51% at 1024^2 and 64% at 2K, alongside runtime reductions of 10–21% from re-decoding at every inner step instead of backpropagating through the decoder. Despite these efficiency gains, reconstruction quality remains comparable: across the four tasks in Table 3a, individual metrics move in both directions, but ASLU matches or improves most entries without systematic degradation. The reconstructions are also visually comparable in Figure 1(a); Appendix B.1 provides additional examples for Gaussian deblurring and super-resolution.

4.2.2 High-dimensional video inverse problems: STeP

STeP (Zhang et al., 2025b) combines a spatiotemporal latent diffusion prior with Decoupled Annealing Posterior Sampling (DAPS) (Zhang et al., 2025a), whose data-consistency step uses Hamiltonian Monte Carlo (HMC). At each HMC leapfrog step, STeP decodes the current latent video z

Table 3: **ASLU substantially reduces memory in both FlowDPS and STeP while maintaining comparable reconstruction quality.** ↑: higher is better; ↓: lower is better.

(a) FlowDPS image inverse problems. FFHQ at 1024^2 (100 images) and DIV2K at near-native 2K resolution (100 images); time and peak memory are per image on a single A6000.

Domain	Method	Task	PSNR↑	SSIM↑	FID↓	LPIPS/$\bar{\chi}^2$ ↓	Time↓	Mem↓
FFHQ 1024^2	FlowDPS	Gauss σ_k=3	26.81	0.632	68.1	0.154	52.9 s	17.1 GB
	ASLU (Ours)		**27.07**	**0.638**	**67.7**	**0.141**	**46.7 s**	**8.4 GB**
	FlowDPS	Motion 61	28.56	0.640	52.2	0.088	52.9 s	17.1 GB
	ASLU (Ours)		**29.93**	**0.703**	**43.4**	**0.066**	**47.6 s**	**8.4 GB**
DIV2K ∼2K	FlowDPS	SR ×4	**24.46**	0.541	64.6	0.076	177.4 s	40.3 GB
	ASLU (Ours)		24.04	0.541	**48.6**	**0.073**	**140.1 s**	**14.7 GB**
	FlowDPS	SR ×8	**23.30**	0.502	70.8	**0.080**	178.9 s	39.7 GB
	ASLU (Ours)		23.09	**0.532**	**64.4**	0.101	**143.1 s**	**14.5 GB**

(b) STeP black-hole video reconstruction on the EHT-2017 array. PSNR and SSIM are mean (std) over 20 public GRMHD test videos. $\bar{\chi}^2$ is reported for two representative videos (s0/s1); peak memory and time are per HMC leapfrog step on a single A6000.

Method	PSNR↑	SSIM↑	$\bar{\chi}^2_{s0}$↓	$\bar{\chi}^2_{s1}$↓	Peak Mem↓	Time/step↓	Grad?
STeP (decoder ∇)	23.56 (4.61)	**0.710** (0.115)	1.15	2.69	37.1 GiB	53 s	✓
STeP + ASLU (Ours)	**24.00** (4.05)	0.683 (0.132)	**1.14**	**2.02**	**8.8 GiB**	**45.5 s**	×
Δ	+0.44	−0.03	≈ 0	−0.67	−76%	−14%	

into $x = \mathcal{D}(z)$ and computes the pixel-space correction $\Delta x = -\nabla_x \mathcal{L}(y, A(x))$, where A maps the decoded video to the measured interferometric quantities. The original STeP then converts this correction into the latent update $\Delta z_{\mathrm{STeP}} = J_{\mathcal{D}}(z)^\top \Delta x$ using VJP, which requires backpropagation through the decoder at every leapfrog step.

We leave the DAPS sampler, HMC updates, measurement loss, and forward operator unchanged, and replace only this pixel-to-latent conversion. Let $\widehat{\Delta x} = \Delta x / \|\Delta x\|_\infty$. Our implementation uses

$$\Delta z_{\mathrm{ASLU}} = \frac{\mathcal{E}(x + \delta\widehat{\Delta x}) - \mathcal{E}(x - \delta\widehat{\Delta x})}{2\delta} \|\Delta x\|_\infty \frac{n_x}{n_z}, \tag{9}$$

where $\delta = 0.01\,\mathrm{std}(x)$ and $n_x/n_z = 64$ is the VAE's spatial compression ratio. This is the ASLU update of Eq. (6) with STeP-specific normalization and scaling (Appendix A.2). Computing Δx differentiates through A only; obtaining Δz_{ASLU} then requires one decoder forward pass and two encoder forward passes, with no decoder activation graph stored. For the 64-frame videos used by STeP, this reduces peak gradient memory from 37.1 to 8.8 GiB (76%) and HMC-step time from 53 to 45.5 s (14%), while preserving image quality and the data-fit statistic $\bar{\chi}^2$ (1 is ideal); see Table 3b and Figure 1(b).

4.3 Positioning ASLU on an independent latent-DIS benchmark

The preceding paired integrations isolate ASLU inside fixed solvers, but do not compare against the broader latent-DIS landscape. We therefore swap ASLU into PSLD's decoder-backward measurement term on DM4CT's sparse-view CT benchmark and compare with its latent-DIS baselines under identical conditions (full setup in Appendix A.3).

Figure 2 reports DM4CT's latent DIS methods—PSLD, DiffStateGrad, and ReSample—averaged across configurations i–v, and reproduces the baselines' PSNR ordering in DM4CT's own results (absolute runtimes differ because we use different hardware, but the ranking matches). PSLD+ASLU attains slightly higher average PSNR and SSIM than PSLD while requiring substantially less memory and time. Because ASLU's hyperparameters were tuned by ourselves, however, we do not interpret this small quality difference as evidence that replacing decoder-backward guidance with ASLU intrinsically improves reconstruction quality; the supported conclusion is *comparable* quality with substantially better efficiency. Figure 1(c) provides the corresponding single-

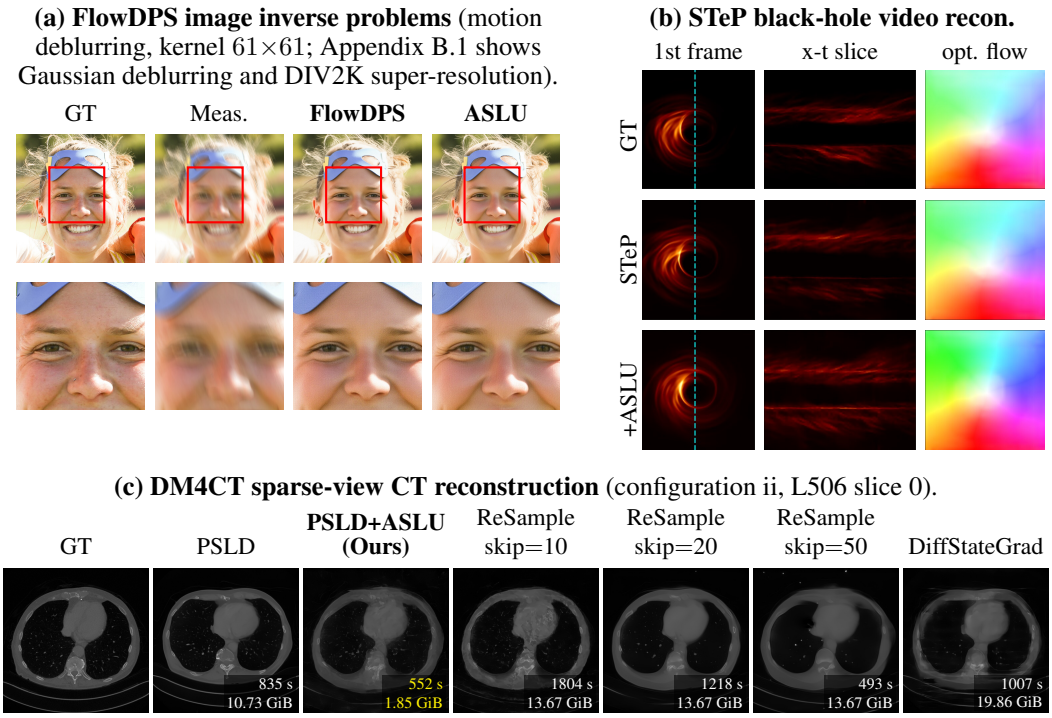


Figure 1: **Across image, video, and CT inverse problems, ASLU remains visually competitive with or improves upon decoder-backward guidance without any decoder backward pass.** In **(b)** (EHT-2017, video 7; visualization convention of STeP), columns show the first frame, x-t slice, and averaged optical flow; rows compare ground truth, STeP baseline (37.1 GiB), and STeP+ASLU (8.8 GiB). In **(c)**, all images show the actual reconstructions of the same configuration-ii CT input; overlays report runtime and peak memory for each reconstruction, with PSLD+ASLU's efficiency highlighted in yellow. Aggregate results across configurations i–v and controlled efficiency measurements are reported in Figure 2 and Appendix A.3.

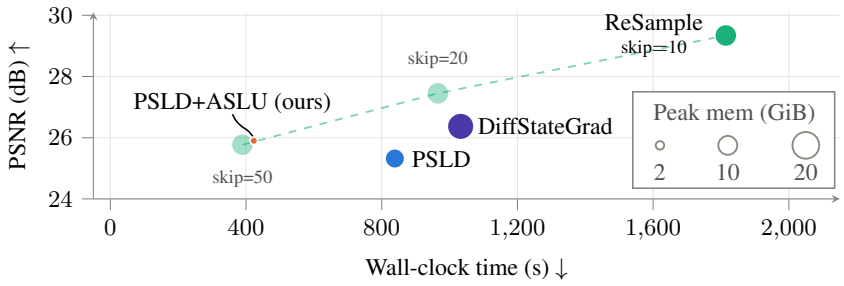


Figure 2: **PSLD+ASLU lies on the PSNR–runtime Pareto frontier of the DM4CT benchmark while reducing peak memory by at least 85% relative to every evaluated baseline, enabled by replacing PSLD's decoder backward pass with encoder forward passes.** Results are averaged across configurations i–v of the DM4CT (Shi et al., 2026) `lodochallenge` task on L506, with 20 slices per configuration. Full numbers, including SSIM, are in Appendix A.3. Point *area* (not radius) is proportional to peak GPU memory; time and memory follow the controlled measurements described in Appendix A.3. The ReSample points trace its skip-step-size runtime–quality tradeoff.

slice visual comparison across the actual outputs of PSLD, PSLD+ASLU, three ReSample skip-step settings, and DiffStateGrad. This independent benchmark thus places PSLD+ASLU on the PSNR–runtime Pareto frontier while occupying a peak-memory regime unattained by the evaluated baselines.

Table 4: **ASLU is substantially better aligned with the exact decoder-backward update; direct re-encoding and gradient-free decoder inversion are nearly orthogonal to it.** Cosine similarity and relative magnitude of each method's Δz against $J_{\mathcal{D}}(z_0)^\top \Delta x$, for one shared Δx (4-image average, FFHQ Gaussian deblurring). Relative magnitude is reported as a percentage of the decoder-backward norm.

Method	Cosine sim. vs. $J_{\mathcal{D}}(z_0)^\top \Delta x$	Relative magnitude (%)
Direct re-encoding	0.06	162
Gradient-free decoder inversion	0.08	150
ASLU (ours)	**0.62**	19

5 Discussion

How close is ASLU to decoder-backward guidance? To assess this experimentally, we use the FFHQ Gaussian-deblurring setting of Section 4.2.1 and apply each latent-update rule to the same FlowDPS pixel-space correction Δx. We compare the ASLU update $\Delta z_{\mathrm{ASLU}} = \frac{\rho}{2}[\mathcal{E}(x_0+\alpha\Delta x) - \mathcal{E}(x_0-\alpha\Delta x)]$ with the exact decoder-backward VJP $\Delta z_{\mathrm{DB}} = J_{\mathcal{D}}(z_0)^\top \Delta x$. Table 4 shows that direct re-encoding and gradient-free decoder inversion (Hong et al., 2024) are nearly orthogonal to the decoder-backward update (cosine similarity 0.06–0.08): their latent displacements are governed more by the encoder's projection of off-manifold images than by the local gradient geometry of decoder backpropagation. ASLU is substantially better aligned (cosine similarity 0.62), although only partially because the two updates use different operators: ASLU's leading term is $J_{\mathcal{E}}(x_0)\Delta x$, whereas decoder-backward guidance uses $J_{\mathcal{D}}(z_0)^\top \Delta x$. ASLU should therefore be understood not as a cheap replica of decoder-backward guidance, but as an encoder-native way to lift the same pixel-space correction. ASLU's norm is also 81% smaller than that of the decoder-backward update, so their step-size hyperparameters should not be expected to transfer directly.

Additional hyperparameter and sensitivity. Decoder-backward guidance typically exposes a single step size, as in Eq. (2), whereas ASLU separates the pixel-space probe scale α from the latent-update scale ρ. Because guidance schedules already matter substantially in DIS (Bang et al., 2026), this additional degree of freedom requires joint tuning, although one pair per system sufficed in our experiments (Appendix C.2).

Limitations. ASLU removes a computational barrier, not a data barrier: it is most useful when a suitable latent prior already exists but decoder backpropagation limits inference. Where representative training data are scarce, including many medical-imaging settings, learning such a prior remains the primary challenge (Hu et al., 2024; 2025). The system-specific joint tuning of (α, ρ) described above is a second limitation; decoder-backward step sizes do not transfer automatically to ASLU.

6 Conclusion

We studied how a pixel-space measurement correction Δx should be converted into a latent update for diffusion inverse solvers, and proposed **Antisymmetric Latent Update (ASLU)**: an update built from ordinary encoder forward passes alone that is exactly zero when $\Delta x = 0$, so no drift compounds across sampling steps, and whose centered construction also cancels the second-order curvature bias, preserving the first-order term the guidance is meant to deliver. We tested this construction at three complementary levels: controlled comparisons confirmed that its advantage over one-sided differencing grows with the correction magnitude; drop-in integrations into FlowDPS and STeP demonstrated substantial memory savings across high-resolution images, high-dimensional video, and two distinct DIS mechanisms; and an independently developed DM4CT evaluation placed PSLD+ASLU on the PSNR–runtime Pareto frontier while substantially reducing peak memory. Future work will explore whether ASLU can serve as an efficient pixel-to-latent update beyond latent DIS, in broader settings where pixel-space corrections must be lifted into latent representations.

AI USE STATEMENT

In this work, we used generative AI tools (Anthropic's Claude via Claude Code and OpenAI Codex) throughout the experimental and writing workflow. Required disclosures: AI tools were used to (i) implement methods and experiment code, including our ASLU update rule and its integration into FlowDPS (image inverse problems), STeP (spatiotemporal video inverse problems), and our independent CT reconstruction benchmark on DM4CT; (ii) design and run experimental methodology, including the hyperparameter robustness sweeps and latent-update ablations reported in the appendix; (iii) interpret experimental results and cross-check them against baseline papers' own published numbers; and (iv) propose and refine hypotheses during debugging and experiment design. We also used AI tools for recommended-disclosure tasks, including drafting and editing portions of the manuscript text and creating/editing software code more generally. All AI-assisted code, experimental results, and text were reviewed by the authors and verified via smoke tests before being used to produce reported numbers. The core theoretical framework of this paper (ASLU) and its derivation are the authors' own; AI tools were used to implement, stress-test, and validate this framework experimentally, not to originate its mathematical claims. We take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Eirikur Agustsson and Radu Timofte. NTIRE 2017 challenge on single image super-resolution: Dataset and study. *CVPR*, 2017. 14

Kazunori Akiyama, Antxon Alberdi, Walter Alef, Juan Carlos Algaba, Richard Anantua, Keiichi Asada, Rebecca Azulay, Uwe Bach, Anne-Kathrin Baczko, David Ball, et al. First Sagittarius A* event horizon telescope results. iii. imaging of the galactic center supermassive black hole. *The Astrophysical Journal Letters*, 930(2):L14, 2022. 1

Gérôme Andry, Sacha Lewin, François Rozet, Omer Rochman, Victor Mangeleer, Matthias Pirlet, Elise Faulx, Marilaure Grégoire, and Gilles Louppe. Appa: Bending weather dynamics with latent diffusion models for global data assimilation. *arXiv preprint arXiv:2504.18720*, 2025. 4

Junseo Bang, Dong Ju Mun, Hoigi Seo, Seongmin Hong, and Se Young Chun. Triadic dynamics aware diffusion posterior sampling for inverse problems: Optimizing guidance and stochasticity schedules. *ICML*, 2026. 9

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. *ICML*, 2017. 2

Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. 2

Gabriel Cardoso, Yazid Janati El Idrissi, Sylvain Le Corff, and Eric Moulines. Monte Carlo guided diffusion for Bayesian linear inverse problems. *ICLR*, 2024. 2

Pascal Chang, Sergio Sancho, Junxuan Tang, Markus Gross, and Vinicius C. Azevedo. LookingGlass: Generative anamorphoses via Laplacian pyramid warping. *CVPR*, 2025. 4, 5, 15

Siyi Chen, Yixuan Jia, Qing Qu, He Sun, and Jeffrey Fessler. FlowDAS: A stochastic interpolant-based framework for data assimilation. *NeurIPS*, 2026. 4

Hyungjin Chung and Jong Chul Ye. Score-based diffusion models for accelerated MRI. *Medical Image Analysis*, pp. 102479, 2022. 2

Hyungjin Chung, Jeongsol Kim, Michael T. McCann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *ICLR*, 2023a. 1, 2, 3

Hyungjin Chung, Dohoon Ryu, Michael T McCann, Marc L Klasky, and Jong Chul Ye. Solving 3D inverse problems using pre-trained 2d diffusion models. *CVPR*, 2023b. 3

Hyungjin Chung, Jong Chul Ye, Peyman Milanfar, and Mauricio Delbracio. Prompt-tuning latent diffusion models for inverse problems. *ICLR*, 2024. 3, 4

Hyungjin Chung, Jeongsol Kim, and Jong Chul Ye. Diffusion models for inverse problems. *arXiv preprint arXiv:2508.01975*, 2025. 2

Florentin Coeurdoux, Nicolas Dobigeon, and Pierre Chainais. Plug-and-play split Gibbs sampler: Embedding deep generative priors in Bayesian inference. *IEEE TIP*, 2024. 2

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*, 2024a. 1, 2

Giannis Daras, Weili Nie, Karsten Kreis, Alexandros G Dimakis, Morteza Mardani, Nikola B Kovachki, and Arash Vahdat. Warped Diffusion: Solving video inverse problems with image diffusion models. *NeurIPS*, 2024b. 4

Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. *ICLR*, 2024. 2

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. *ICLR*, 2024. 1

Seongmin Hong, Suh Yoon Jeon, Kyeonghyun Lee, Ernest Ryu, and Se Young Chun. Gradient-free decoder inversion in latent diffusion models. *NeurIPS*, 2024. 3, 4, 9, 18

Jason Hu, Bowen Song, Xiaojian Xu, Liyue Shen, and Jeffrey A Fessler. Learning image priors through patch-based diffusion models for solving inverse problems. *NeurIPS*, 2024. 4, 9

Jason Hu, Bowen Song, Jeffrey A Fessler, and Liyue Shen. Test-time adaptation improves inverse problem solving with patch-based diffusion models. *IEEE Transactions on Computational Imaging*, 2025. 4, 9

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. DiffusionPDE: Generative PDE-solving under partial observation. *NeurIPS*, 2024. 4

Ajil Jalal, Marius Arvinte, Giannis Daras, Eric Price, Alexandros G. Dimakis, and Jonathan Tamir. Robust compressed sensing MRI with deep generative priors. *NeurIPS*, 2021. 2

Yixuan Jia, Siyi Chen, Yida Pan, Xiao Li, Lianghe Shi, Chanyong Jung, Haijie Yuan, Ismail Alkhouri, Yue Cynthia Wu, Saiprasad Ravishankar, Jeffrey A Fessler, and Qing Qu. ForcingDAS: Unified and robust data assimilation via diffusion forcing. In *ICML 2026 Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026. 1

Bahjat Kawar, Gregory Vaksman, and Michael Elad. SNIPS: Solving noisy inverse problems stochastically. *NeurIPS*, 2021. 2

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *NeurIPS*, 2022. 2

Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. *ICCV*, 2025. 1, 2, 3, 4, 6

Taesung Kwon and Jong Chul Ye. VISION-XL: High definition video inverse problem solver using latent image diffusion models. *ICCV*, 2025. 4

Xiang Li, Soo Min Kwon, Ismail R Alkhouri, Saiprasad Ravishanka, and Qing Qu. Decoupled data consistency with diffusion purification for image restoration. *arXiv preprint arXiv:2403.06054*, 2024. 3

Morteza Mardani, Jiaming Song, Jan Kautz, and Arash Vahdat. A variational perspective on solving inverse problems with diffusion models. *ICLR*, 2024. 2

Cynthia H McCollough, Adam C Bartley, Rickey E Carter, Baiyu Chen, Taly A Drees, Phillip Edwards, David R Holmes III, Alice E Huang, Farhana Khan, Shuai Leng, Kyle L McMillan, Gregory J Michalak, Kristina M Nunez, Lifeng Yu, and Joel G Fletcher. Low-dose CT for the detection and classification of metastatic liver lesions: Results of the 2016 Low Dose CT grand challenge. *Medical Physics*, 44(10):e339–e352, 2017. 14

Yongquan Qu, Juan Nathaniel, Shuolin Li, and Pierre Gentine. Deep generative data assimilation in multimodal setting. *CVPR*, 2024. 4

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CVPR*, 2022. 1, 3

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. *NeurIPS*, 2024. 3

François Rozet and Gilles Louppe. Score-based data assimilation. *NeurIPS*, 2023. 1

François Rozet, Gaétan Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. *NeurIPS*, 2024. 2

Jiayang Shi, Daniël M Pelt, and K Joost Batenburg. DM4CT: Benchmarking diffusion models for computed tomography reconstruction. *ICLR*, 2026. 8

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. *ICLR*, 2024. 1, 3, 15

Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. *ICLR*, 2023. 2

Yang Song, Jascha Sohl-Dickstein, Durk P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021. 2

Yang Song, Liyue Shen, Lei Xing, and Stefano Ermon. Solving inverse problems in medical imaging with score-based generative models. *ICLR*, 2022. 1

Alessio Spagnoletti, Andres Almansa, and Marcelo Pereyra. LVTINO: LAtent video consistency INverse solver for high definition video restoration. *ICLR*, 2026. 4

Fabian Steinbrenner, Baris Turan, Hao Teng, and Heng Xiao. Turbulence generation and data assimilation in wall-bounded flows with a latent diffusion model. *arXiv preprint arXiv:2603.02143*, 2026. 1

Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. *ICLR*, 2023. 2

Yuanzhe Wang and Alexandre M Tartakovsky. Latent diffusion posterior sampling with surrogate likelihood guidance for PDE inverse problems. *arXiv preprint arXiv:2606.26592*, 2026. 4

Zihui Wu, Yu Sun, Yifan Chen, Bingliang Zhang, Yisong Yue, and Katherine Bouman. Principled probabilistic imaging using diffusion models as plug-and-play priors. *NeurIPS*, 2024. 2

Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. *CVPR*, 2025a. 2, 6

Bingliang Zhang, Zihui Wu, Berthy T. Feng, Yang Song, Yisong Yue, and Katherine L. Bouman. STeP: A framework for solving scientific video inverse problems with spatiotemporal diffusion priors. *arXiv preprint arXiv:2504.07549*, 2025b. 1, 2, 3, 4, 6

Hongkai Zheng, Wenda Chu, Bingliang Zhang, Zihui Wu, Austin Wang, Berthy Feng, Caifeng Zou, Yu Sun, Nikola Borislavov Kovachki, Zachary E Ross, Katherine Bouman, and Yisong Yue. InverseBench: Benchmarking plug-and-play diffusion priors for inverse problems in physical sciences. *ICLR*, 2025. 1, 2

Yilin Zhuang and Karthik Duraisamy. LaDCast: A latent diffusion model for medium-range ensemble weather forecasting. *arXiv preprint arXiv:2506.09193*, 2025. 1

Rayhan Zirvi, Bahareh Tolooshams, and Anima Anandkumar. Diffusion state-guided projected gradient for inverse problems. *ICLR*, 2025. 15

A Experimental Details

A.1 FlowDPS image inverse problems

We evaluate on two tasks:

- **Gaussian deblurring**: separable Gaussian kernel, σ_k=3.0, kernel size 61×61, noise σ_n=0.03.
- **Motion deblurring**: random motion blur kernel of size 61×61, intensity 0.5, noise σ_n=0.03.
- **Super-resolution** $\times 4$ **and** $\times 8$: average-pool downsampling, noise σ_n=0.03.

For deblurring we use 100 validation images from FFHQ at their native 1024×1024 resolution. For super-resolution we use 100 validation images from DIV2K (Agustsson & Timofte, 2017) at near-native 2K resolution: each image is centre-cropped to 2016×1344 (SR$\times$4) or 1984×1344 (SR$\times$8), preserving the original landscape aspect ratio. We report PSNR (↑) and SSIM (↑) for pixel-level fidelity, FID (↓) for perceptual quality, LPIPS (↓) for perceptual similarity, per-image runtime (↓), and peak GPU memory (↓). All methods use 28 NFE and CFG scale 2.0.

Both methods apply data consistency at every sampling step using $K = 3$ gradient iterations, re-decoding $x = \mathcal{D}(\hat{z}_{0|t})$ at each of the K steps. The critical difference is *where* the gradient and the latent update occur: **FlowDPS** computes $\nabla_{\hat{z}_{0|t}} \mathcal{L}$ via backpropagation through the VAE decoder (retaining its full activation graph), scaled by a task-tuned step size β_t. **ASLU (Ours)** computes the same unsquared-residual-norm gradient $\nabla_x \mathcal{L} = \nabla_x \|y - Ax\|_2$ through A only, no decoder involved, scales it by the pixel-space factor α to form $\Delta x = -\alpha \nabla_x \mathcal{L}$, and converts it to a latent update via the antisymmetric encoder difference $\frac{1}{2}[\mathcal{E}(x+\Delta x) - \mathcal{E}(x-\Delta x)]$ scaled by the latent-space factor ρ, at every one of the K steps. No decoder backpropagation occurs at any point. We set α=4, ρ=50 throughout, chosen from the hyperparameter sweep in Appendix C.2 rather than tuned per task; ρ is substantially larger than FlowDPS's own β_t=15 because the raw autograd direction is computed without the decoder Jacobian attenuation that implicitly scales FlowDPS's latent update. The controlled comparison of Table 2 instead lifts the accumulated Δx once per timestep with untuned (α, ρ) to isolate the update formula, hence its shorter runtime ($\sim$20 s vs. 46.7 s).

A.2 STeP scientific video inverse problems

For the black hole reconstruction task (EHT-2017 array), the latent video has shape $(T, C, H, W) = (64, 1, 32, 32)$ spanning 64 temporal frames, each decoded to a 256×256 image. The ASLU replacement described in Section 4.2.2 uses the approximate local relation $J_{\mathcal{E}} \approx J_{\mathcal{D}}^{\top}$, motivated by an encoder–decoder cycle that is nearly distance-preserving.

We evaluate on STeP's standard EHT-2017 black hole protocol: 25 DAPS outer steps with 53 HMC inner steps. For BH reconstruction, pixel-space metrics (PSNR, SSIM) are not the primary evaluation criterion because the problem is extremely ill-posed: many structurally different images can produce identical interferometric closures. STeP's own online evaluator (`BlackholeEvaluator`) therefore measures only the *data fit* $\bar{\chi}^2$, defined as the unified average chi-squared statistic over closure phases and log-closure amplitudes ($\bar{\chi}^2 = 1$ is a perfect fit). We report $\bar{\chi}^2$ as the quality metric alongside **efficiency** (peak gradient memory and per-HMC-step wall time).

A.3 DM4CT sparse-view CT reconstruction

We evaluate on configurations i–v of DM4CT's `lodochallenge` task, using the held-out L506 volume from the AAPM/Mayo Clinic Low Dose CT Grand Challenge (McCollough et al., 2017). For each configuration, we use DM4CT's own measurement setup, ASTRA-based forward operator, and pretrained latent-diffusion checkpoint (a VQ-VAE unrelated to the FlowDPS/STeP VAEs used elsewhere in this paper). PSLD is DM4CT's decoder-backward baseline for this pipeline, using its published setup for each configuration; PSLD+ASLU replaces only PSLD's decoder-backward measurement term $\nabla_{\hat{z}_0} \|y - A(\mathcal{D}(\hat{z}_0))\|$ with the ASLU update of Eq. (6), leaving PSLD's gluing term and every other design choice untouched. As in Appendix A's other two pipelines, the entire PSLD+ASLU sampling loop runs under `torch.no_grad()`, and every decoded proposal is

checked by an explicit runtime assertion rather than relying on how the update happens to be written. L506 has 526 axial slices; for each configuration, we evaluate 20 slices sampled evenly across the volume (indices $0, 28, 55, \ldots, 525$), for 100 reconstructions per method in total. PSNR/SSIM use DM4CT's fixed `data_range=2` convention and are averaged across configurations i–v. Figure 1(c) reports per-slice values for one configuration-ii example, while Table A1 gives the aggregate quality and controlled efficiency measurements behind Figure 2. Time and memory are measured separately under controlled, uncontended conditions, with every method run alone on an otherwise-idle GPU under the same protocol.

Table A1: **Full numbers behind Figure 2.** PSNR/SSIM are averaged across configurations i–v, using 20 slices per configuration. Time and peak memory use the controlled, uncontended measurements described above.

Method	PSNR↑	SSIM↑	Time (s)↓	Peak mem↓
PSLD (DM4CT baseline)	25.32	0.737	838.6	10.73 GiB
PSLD+ASLU (ours)	25.90	0.739	**422.8**	**1.59 GiB**
ReSample (Song et al., 2024) (skip=10)	**29.34**	**0.789**	1814.1	13.80 GiB
ReSample (skip=20)	27.45	0.779	965.3	13.80 GiB
ReSample (skip=50)	25.77	0.755	389.4	13.80 GiB
DiffStateGrad (Zirvi et al., 2025)	26.37	0.749	1032.6	19.99 GiB

$\alpha{=}\rho{=}0.6$ was chosen from a sweep over $\{0.2, \ldots, 0.9\}$ on a small validation subset; larger values caused occasional divergence, illustrating that ASLU's local correctness properties (Section 3) do not guarantee stability at every step size. Because this tuning was performed specifically for PSLD+ASLU, its small average PSNR/SSIM lead over PSLD in Table A1 should not be interpreted as evidence that ASLU intrinsically improves reconstruction quality; we treat the two methods as providing comparable quality for the efficiency comparison. ReSample and DiffStateGrad use DM4CT's grid-searched hyperparameters for each configuration, rather than the illustrative values in its usage-example script. ReSample's `skip_step` controls how often its time-travel and hard-data-consistency operations are triggered; we evaluate `skip_step` $\in \{10, 20, 50\}$ under the same configurations-i–v protocol to expose its runtime–quality tradeoff.

B Additional Experimental Results

B.1 Additional FlowDPS qualitative examples

Figure 1(a) in the main text shows motion deblurring; Figure A1 below shows Gaussian deblurring and DIV2K super-resolution from Table 3a in the same style.

B.2 Qualitative comparison of latent-update strategies

Figure A2 complements the controlled comparison in Section 4.1 using its median case by PSNR gap between one-sided differencing and ASLU.

C Additional Latent-Update Ablations

Section 4.1 validates the latent-update design, while Section 5 asks how ASLU relates to decoder-backward guidance. This appendix provides the supporting analyses. Throughout, we write the one-sided construction of Eq. (5) as $d_{\mathrm{fwd}} = \rho[\mathcal{E}(x{+}\alpha\Delta x) - \mathcal{E}(x)]$—the same one-step difference used by LookingGlass (Chang et al., 2025) and the $K{=}1, \rho{=}1$ special case of gradient-free decoder inversion (Eq. (A1)). We report direction and magnitude, hyperparameter robustness, details of gradient-free decoder inversion, and exact-JVP diagnostics.

C.1 Direction and magnitude of Δz

For a fixed, realistic decoder-manifold point $x_0 = \mathcal{D}(z_0)$ (obtained by perturbing a clean image's latent and decoding), we construct one shared pixel-space correction Δx using FlowDPS's own up-

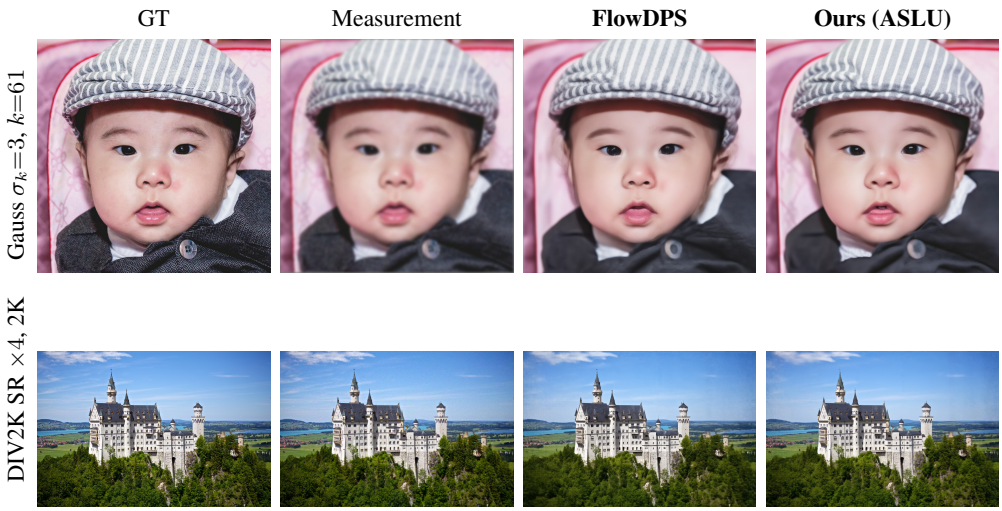


Figure A1: **ASLU reconstructs Gaussian-deblurred and super-resolved images visually on par with FlowDPS.** Rows: Gaussian deblurring (σ_k=3, kernel 61×61); DIV2K 4× super-resolution at near-native 2K. Cols: ground truth; measurement; FlowDPS; ASLU (ours). The DIV2K example is the median case by PSNR gap between FlowDPS and ASLU.

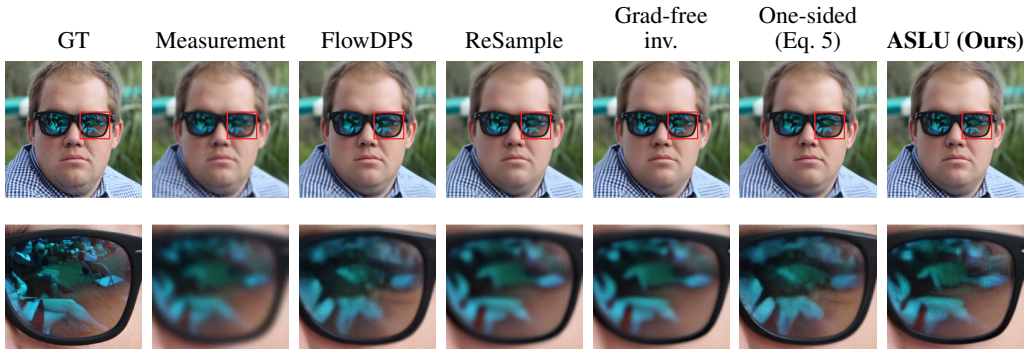


Figure A2: **ASLU matches FlowDPS's visual quality without decoder backpropagation, while Eq. (5) and direct re-encoding show visible ringing artifacts.** Qualitative comparison of the five latent-update strategies of Table 2 on FFHQ Gaussian deblurring (σ_k=3, kernel 61×61, 1024^2). **Top row:** full-image thumbnails; the red box marks the zoomed region below. **Bottom row:** zoomed crops of the sunglasses lens, which contains fine high-frequency reflection detail (best viewed on screen). The example image was selected as the median case by PSNR gap between Eq. (5) and ASLU across the 100-image benchmark, i.e. a typical rather than a best-case or worst-case example. Direct re-encoding and Eq. (5) both show visible checkerboard ringing in the lens reflection, consistent with their second-order curvature bias and worse FID/LPIPS in Table 2. Gradient-free decoder inversion and FlowDPS produce smoother reconstructions, but ASLU reduces runtime by 82% relative to gradient-free decoder inversion. ASLU produces a comparably clean result to FlowDPS without any decoder backpropagation.

date rule—autograd of the *unsquared* residual norm $\|y - Ax\|_2$ through A only, K=3 steps, step size 15, matching its actual implementation rather than the squared-norm convention used elsewhere in this paper. We then apply each method's Δz formula—direct re-encoding, gradient-free decoder inversion (Eq. (A1)), the one-sided construction (Eq. (5)), and ASLU (Eq. (6))—to this same Δx and compare every result against the exact decoder-backward update $\Delta z_{\text{FlowDPS}} = J_{\mathcal{D}}(z_0)^\top \Delta x$, computed as a vector–Jacobian product. Table 4 reports the result for direct re-encoding, gradient-free decoder inversion, and ASLU; Eq. (5) scores cosine similarity 0.60 and relative magnitude 0.20, as expected: this single-step test probes the first-order term the two constructions share (Eqs. (5), (7)).

Direct re-encoding and gradient-free decoder inversion re-encode the fully updated pixel image $x_0 + \Delta x$ from scratch, so their latent displacement is dominated by where the VAE's encoder happens to project an off-manifold point, not by the local gradient geometry the decoder-backward update follows. This explains their poor FID and LPIPS in Table 2 directly: they move the latent a large distance in a direction the decoder does not expect. Eq. (5) and ASLU, by contrast, are both moderately aligned with the decoder-backward direction and about 80% smaller in norm—the encoder-based update is intrinsically more conservative than decoder-backward guidance, since it applies the encoder's own local Jacobian $J_{\mathcal{E}}(x_0)$ rather than the decoder's transpose Jacobian $J_{\mathcal{D}}(z_0)^\top$, and these two operators coincide only under an exact encoder–decoder isometry that real VAEs satisfy approximately at best. Critically, Eq. (5) and ASLU are *not* distinguishable by this single-step comparison: their cosine similarity to the FlowDPS direction differs by only 0.02, and ASLU's curvature correction is a higher-order effect that a single linearized step is too coarse to reveal.

C.2 Hyperparameter robustness across three tasks

The single-step alignment comparison of Table 4 cannot separate a genuine curvature effect from an artifact of a single, hand-picked (α, ρ). We therefore sweep α (6 values) and ρ (5 values) for both Eq. (5) and ASLU, on one held-out image each from Gaussian deblurring, motion deblurring, and SR$\times$4, scoring every combination by LPIPS. We exclude the most extreme value on each axis ($\alpha = 0.25$ and $\rho = 800$), where both methods collapse together and the comparison carries no signal, leaving a 5×4 grid per task. Figure A3 shows the trimmed grids: ASLU's advantage is real but visually subtle, consistent with these tasks' small, iterative Δx. Head-to-head at matched (α, ρ) on these trimmed grids, ASLU beats Eq. (5) on LPIPS in 67% of cells (mean advantage 0.007), and is never the worse method's best point on any of the three tasks. Neither method's *worst*-case cell is meaningfully better than the other's, so a poorly chosen (α, ρ) degrades both similarly here; Table 2 examines a sharper, one-shot-Δx regime (analytic backprojection) directly, where the gap is substantially larger.

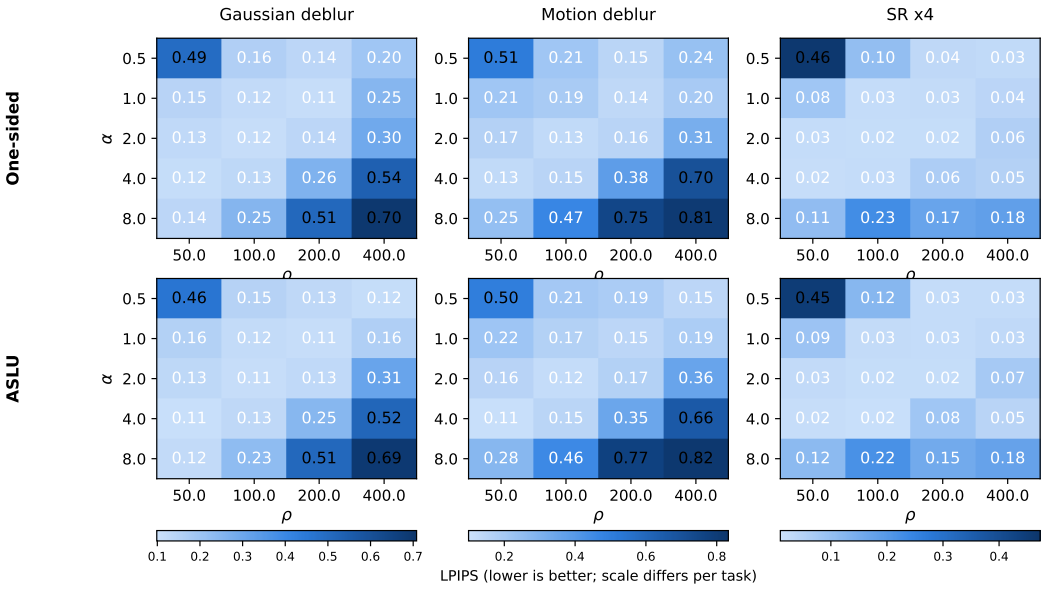


Figure A3: **ASLU's advantage over the one-sided update of Eq. (5) is real but visually subtle on these three small, iterative-Δx tasks.** LPIPS, one held-out image per task: Gaussian deblurring, motion deblurring, and SR$\times$4, 5×4 (α, ρ) grid each ($\alpha = 0.25$ and $\rho = 800$ dropped as uninformative: both methods collapse together there); each task uses its own LPIPS colour scale, since SR$\times$4's values sit far below Gaussian/motion deblurring's. Head-to-head at matched (α, ρ): ASLU wins 67% of the 60 cells (mean advantage 0.007)—consistent with the curvature bias being small when Δx is small; Table 2 shows the same bias becoming a 1.45 dB effect once Δx is large (analytic backprojection).

C.3 GRADIENT-FREE DECODER INVERSION IN DETAIL

Section 3 introduces gradient-free decoder inversion (Hong et al., 2024) as the forward-only strategy that removes direct re-encoding's zeroth-order error without our antisymmetric construction. It searches for a latent whose decoded-and-reencoded image matches the pixel proposal,

$$z^{k+1} = z^k - \rho\big(\mathcal{E}(\mathcal{D}(z^k)) - \mathcal{E}(x + \alpha\Delta x)\big). \tag{A1}$$

Setting $\alpha{=}\rho{=}1$ and $K{=}1$ gives the one-sided update $\rho[\mathcal{E}(x + \alpha\Delta x) - \mathcal{E}(x)]$ of Section 3 exactly, since $\mathcal{E}(\mathcal{D}(z^0)) = \mathcal{E}(x)$; running the recursion to convergence instead removes the zeroth-order error entirely, at the cost of several inner iterations for every posterior-sampling step. Moreover, when Δx contains off-manifold measurement corrections, i.e., components outside $\mathrm{range}(J_{\mathcal{D}})$, decoder-manifold inversion can amplify rather than remove high-frequency artifacts. Table 2 verifies both costs empirically: Eq. (5)/ASLU reduces runtime by 82% relative to gradient-free decoder inversion and achieves substantially better FID and LPIPS.

C.4 EXACT ENCODER TANGENT LIFTS VERSUS DEPLOYABLE FORWARD-ONLY LIFTS

Eq. (6)'s central difference converges to the exact directional derivative $\rho\alpha\, J_{\mathcal{E}}(x)\Delta x$ as $\alpha \to 0$, and forward-mode automatic differentiation can compute this same Jacobian–vector product (JVP) exactly, e.g. via `torch.func.jvp`. As Section 3 explains, this JVP is the exact encoder-tangent lift $J_{\mathcal{E}}(x)\Delta x$ that ASLU's leading term approximates—not an exact version of decoder-backward guidance's $J_{\mathcal{D}}(z)^\top \Delta x$, and not a way to remove the latent step scale ρ, which exact JVP still needs just as ASLU does. Using it in place of ASLU would remove only the finite-difference radius α; this appendix asks whether that trade is actually deployable. We test it on the spatiotemporal-video VAE encoder used in our STeP experiments (a `diffusers.AutoencoderKL`), on a single L40S (44.5 GB).

JVP is exact where it runs. On one real video frame and one random probe direction Δx (matching the $\hat{v}$ convention of our implementation), we compare the exact JVP against the central difference $[\mathcal{E}(x{+}h\Delta x) - \mathcal{E}(x{-}h\Delta x)]/2h$ at several probe radii h (Table A2). The relative error traces a textbook finite-difference curve: it falls as h shrinks from 10^{-1}, bottoms out near $h \approx \alpha_{\text{default}} = 10^{-2}$ at $\sim$2% (float32 rounding, not a discrepancy between the two derivatives), then rises again for $h \lesssim 10^{-3}$ as cancellation error in the subtraction dominates. This confirms the central difference is estimating the correct quantity.

Table A2: **The finite-difference lift converges to the exact JVP as $h \to$ its optimal float32 radius, then degrades on both sides—confirming both are estimating the same derivative.** Relative error $\|\text{JVP} - \text{FD}(h)\|/\|\text{JVP}\|$ on a real video frame from our STeP test set, 4-frame batch.

h	10^{-3}	3×10^{-3}	10^{-2}	3×10^{-2}	10^{-1}
Rel. error	0.158	0.052	**0.017**	0.058	0.401

JVP does not run at all with PyTorch's default attention kernel. `torch.func.jvp` raises `NotImplementedError` inside the encoder's mid-block self-attention: PyTorch dispatches `scaled_dot_product_attention` to `_scaled_dot_product_efficient_attention` by default, which has no forward-mode-AD rule. Forcing PyTorch's exact, autodiff-complete `MATH` attention backend (same operator, a different kernel) fixes this without changing the result—Table A2 above uses this backend—but ASLU needs no such workaround.

Even fixed, JVP does not fit our production batch size. Table A3 compares peak memory and time for one exact JVP call (`MATH` backend) against ASLU's two ordinary sequential forward passes, same input. Across the tested batch sizes, ASLU reduces peak memory by 77–78% and runtime by 64–82% relative to exact JVP. Memory scales close to linearly with batch, letting us extrapolate to our pipeline's actual batch (64 video frames): ASLU needs an estimated 17 GB, comfortably inside our 44.5 GB GPU, while JVP needs an estimated 76 GB. We confirmed this directly rather than relying on the extrapolation alone: running the full sampling pipeline with the encoder lift replaced

by an exact JVP call, at the real 64-frame batch, out-of-memory before even reaching the attention layer above—a compounding, not alternative, failure mode. Replacing it back with ASLU's central difference completes normally.

Table A3: **Relative to exact JVP, ASLU reduces peak memory by 77–78% and runtime by 64–82% at the tested batch sizes; the memory gap is large enough that JVP fails to fit our production batch on a single GPU where ASLU does.** Peak CUDA memory and time for one lift call (10 trials, 3 warm-up), same input; L40S.

Frames	Peak memory (MiB)		Time (ms/call)	
	JVP	ASLU	JVP	ASLU
4	4924	1109	54.4	9.7
16	19487	4231	149.8	53.4
64 (extrap.; confirmed OOM)	∼76,000	∼17,000	—	—

Exact JVP is therefore best understood as the oracle this construction is built to approach, not a competing deployment choice: Table A2 confirms ASLU's central difference converges to the same first-order term forward-mode autodiff computes exactly, which is the correctness property ASLU is designed to inherit. But the realization is a separate problem from the derivative itself—one common attention kernel has no forward-AD rule at all, and even routed around that, exact JVP's memory cost rules it out at exactly the batch sizes ASLU is designed for. ASLU obtains the same first-order limit through two ordinary forward passes precisely because that realization problem, not the derivative, is what needed solving.