DIFFUSION WALTZ: MCMC POSTERIOR SAMPLING WITH DIFFUSION PRIORS IN THREE STEPS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained diffusion models are powerful priors for inverse problems, but posterior sampling under nonlinear, and possibly non-differentiable, forward models remains challenging. We introduce *diffusion waltz*, an MCMC method that uses SDEdit-style noising–denoising as a proposal, corrected via Metropolis–Hastings to sample exactly from the posterior. We show that this sampling mechanism strictly generalizes preconditioned Crank–Nicolson (pCN) to learned prior settings, and, like pCN, can sample from the posterior without ever evaluating the prior density. To improve mixing, we further introduce *conditional noising*, which injects observation information directly into the proposal while preserving exactness. Across a range of nonlinear inverse problems, with or without access to gradients, diffusion waltz matches or outperforms existing baselines in reconstruction accuracy and uncertainty calibration.

1 INTRODUCTION

Solving inverse problems is a central challenge across science and engineering, where the goal is to recover an unknown state x_0 from noisy, incomplete observations y. Such observations typically do not carry enough information to recover x_0 exactly, rendering the problem ill-posed. The Bayesian formulation of inverse problems resolves this by placing a prior $p_0(x_0)$ over the state and targeting the posterior (Stuart, 2010):

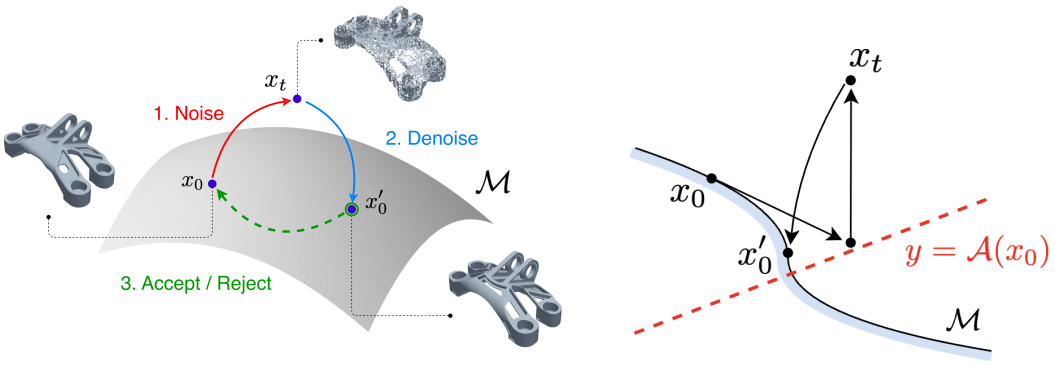
$$p(x_0 \mid y) \propto p(y \mid x_0)\, p_0(x_0). \tag{1}$$

A growing body of work seeks to use pretrained generative models as priors (Daras et al., 2024), but posterior sampling remains elusive when the prior is accessible only through simulation, with no tractable density. Methods such as classifier and classifier-free guidance (Dhariwal & Nichol, 2021; Ho & Salimans, 2022) sample approximately from (1), although, they require additional training and cannot adapt to new forward models. Inference-time "plug-and-play" guidance methods (Chung et al., 2023; Song et al., 2023; Rozet et al., 2024; Kim et al., 2025; Zhang et al., 2025) adapt to new problems without retraining, but typically rely on approximations to the likelihood score, leading to biased posterior samples. For applications in computer vision, where many of these methods were devised, it suffices to achieve strong *pointwise accuracy*; the Bayesian formulation exists merely to motivate method development, while actual evaluations are performed on pointwise metrics, such as PSNR, SSIM and LPIPS (Zhang et al., 2018)[1]. In contrast, scientific applications often demand accurate posterior estimation with well-calibrated uncertainties, requiring more careful consideration of these methods. In addition, many of the plug-and-play methods require $\mathcal{A}$ to be differentiable, which may not be reasonable to assume for many realistic applications.

To address this, we propose **Diffusion Waltz (DW)**, a three-step cycle to sample from (1) via Markov Chain Monte Carlo (MCMC), targeting the posterior exactly (in the asymptotic limit). Our proposal kernel $Q(x_0, x_0')$, inspired by SDEdit (Meng et al., 2022), is constructed by (i) noising x_0 to an appropriate level $t < 1$, then (ii) denoising back to $t = 0$ to yield a new sample x_0'. This is followed by (iii) a Metropolis-Hastings (MH) accept/reject step, where the acceptance probability crucially does not depend on the prior density $p_0(x_0)$ (see Figure 1a). We prove that in the special case when $p_0(x_0)$ is Gaussian, this approach recovers exactly the preconditioned Crank-Nicolson

[1]FID (Heusel et al., 2017), another commonly used metric, is distributional. However, it measures the similarity between a cloud of ground truths and their reconstructions; not actually evaluating posterior accuracy.

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.



(a) Diffusion waltz on 3D inverse design

(b) Conditional noising kernel

Figure 1: **(a) Noise, Denoise, Correct.** Diffusion waltz proceeds by iterating over three steps: (i) adding noise to the current sample x_0, (ii) denoising to produce an on-manifold mutation x_0' of x_0, (iii) a Metropolis-Hastings step to accept x_0', or if rejected, start over from x_0 (green dashed arrow). **(b)** We can inject information about y in the proposal by nudging x_0 towards the hyperplane $y = \mathcal{A}(x_0)$, pushing x_0 off-manifold, but subsequently projecting it back onto the manifold by noising/denoising.

(pCN) method (Cotter et al., 2013), making DW a *direct generalisation of pCN*. Moreover, we show a stability result, providing sufficient conditions when using an approximate denoiser retains this accuracy. To further improve mixing, we introduce **conditional noising**, a mechanism to inject information about y into the proposal, such that the modified proposal yields an exact MH correction without involving the prior density (see Figure 1b). We propose two variants: a gradient-based variant that exploits gradients of $\mathcal{A}$, and a gradient-free variant that does not require them. This makes DW applicable to a broad range of scientific and engineering problems that demand accurate posterior sampling.

2 Background

Bayesian inverse problems and MCMC. A Bayesian inverse problem (Stuart (2010)) seeks to infer a distribution of plausible states x_0 from noisy measurements y by computing the posterior (1). Except in very special cases, computing (1) is intractable. Thus, methods such as Markov Chain Monte Carlo (MCMC) (Hastings, 1970) are used to generate samples from the posterior to approximate it empirically. MCMC generates a Markov chain through a *proposal kernel* $Q(x_0, x_0')$, i.e., a rule to propose a new sample x_0' from x_0, and a *Metropolis-Hastings (MH) correction* step that accepts x_0' with probability $\alpha := \min\left\{1, \frac{p(y|x_0')Q(x_0,x_0')p_0(x_0)}{p(y|x_0)Q(x_0',x_0)p_0(x_0')}\right\}$, discarding it otherwise. However, in many applications the prior density $p_0(x_0)$ cannot be evaluated, rendering the MH step intractable. In this case, a common strategy is to design Q to be *reversible* with respect to p_0, i.e., $p_0(x_0)Q(x_0, x_0') = p_0(x_0')Q(x_0', x_0)$, so that α reduces to a tractable likelihood ratio. This underlies preconditioned Crank-Nicolson (pCN) Cotter et al. (2013), which designs Q to be reversible with respect to Gaussian p_0, enabling MCMC on function space with Gaussian measure priors.

Stochastic interpolants for generative modelling. Let p_0, p_1 be two probability distributions, representing the data and reference distributions, respectively, and α_t, β_t be differentiable, monotonic functions satisfying $\alpha_0 = \beta_1 = 1$ and $\alpha_1 = \beta_0 = 0$, e.g., $\alpha_t = 1 - t$ and $\beta_t = t$. A *stochastic interpolant* (Albergo et al., 2025) is defined as a stochastic process $x_t = \alpha_t x_0 + \beta_t x_1$, for $t \in [0, 1]$, where $x_0 \sim p_0$ and $x_1 \sim p_1$. Then, by (Albergo et al., 2025, Corollary 18), for any non-negative $\{\sigma_t\}_{t\in[0,1]}$, we have that the SDE

$$\mathrm{d}x_t' = \left(b(t, x_t') - \frac{\sigma_t^2}{2} s(t, x_t')\right) \mathrm{d}t + \sigma_t \mathrm{d}W_t, \tag{2}$$

solved backwards in time from $x_1' \sim p_1$ shares the same marginal law $p_t(x_t)$ as the stochastic interpolant, i.e., $\mathrm{Law}(x_t) = \mathrm{Law}(x_t')$ for all $t \in [0, 1]$. Thus, (2) can be used as a generative model

to sample data $x_0' \sim p_0$. Here, $b(t,x) := \mathbb{E}[\dot{x}_t | x_t = x]$ is the *drift* and $s(t,x) := \nabla \log p_t(x)$ is the *score function* of the interpolant; while these do not generally admit closed-form expressions, they can be estimated with neural networks using flow matching ($b_\theta \approx b$) (Lipman et al., 2023) and score matching ($s_\theta \approx s$) (Song et al., 2021), respectively (see Appendix A). In the standard case when the reference p_1 is Gaussian, the score s can be expressed explicitly in terms of b, and vice versa (see Appendix A.1), thus we only need to estimate one of these to be able to use (2).

3 METHOD

We now present our method *diffusion waltz* to sample from the posterior (1), when we have access to an interpolant-based generative model that transports a given reference p_1 to p_0. In general, we consider stochastic interpolants on a separable Hilbert space $\mathcal{H}$, to include learned priors over function spaces (see Yu et al. (2026) for a formalisation). For ease of presentation, we adopt a shared notational convention for measures and densities.

3.1 RANDOM WALK DIFFUSION WALTZ

We first describe the basic variant of diffusion waltz, which we refer to as *Random Walk Metropolis Hastings (RWMH) waltz*. This proceeds as follows: for a fixed $t \in (0,1)$ and an initial sample $x_0 \sim p_0$, we cycle through three steps (Algorithm 1):

1. Add noise to x_0 via the interpolant $x_t = \alpha_t x_0 + \beta_t x_1$, where $x_1 \sim p_1$.
2. Denoise x_t to generate a new sample $x_0' \sim p(x_0 \mid x_t)$.
3. Accept x_0' with probability $\alpha(x_0, x_0') := \min\left\{1, \frac{p(y|x_0')}{p(y|x_0)}\right\}$. If accepted, set $x_0 \leftarrow x_0'$.

The first two steps — perturbing x_0 via noising/denoising to produce a new sample x_0' that remains close to it on the data manifold — is a common strategy in image editing, known as SDEdit (Meng et al., 2022). Here we repurpose this idea to define a proposal kernel $Q(x_0, x_0')$ for MCMC, allowing random walk-like exploration of the data manifold. The third step is a Metropolis–Hastings (MH) correction, which guarantees that the resulting Markov chain converges to the posterior (1) as the number of iterations grows. Under some assumptions, we can show that this chain is geometrically ergodic and targets the posterior exactly (see Proposition 4 in Appendix B.2).

Diffusion waltz generalises pCN. Crucially, we find that the acceptance probability in step 3 of RWMH waltz depends *only* on the likelihood. Hence, we never have to evaluate the prior density p_0, which is either intractable, or non-existent (e.g. in the functional setting). Similar to pCN, this follows from a reversibility argument with respect to the prior, i.e., $p_0(x_0)Q(x_0, x_0') = p_0(x_0')Q(x_0', x_0)$, cancelling the prior terms in the acceptance ratio (see Appendix B.3.1). In fact, we may even go further and claim that RWMH waltz is not only similar in spirit but *directly generalises pCN*, as we show in the next result.

Theorem 1. *Let $\mathcal{C}$ be a positive-definite, self-adjoint, trace-class operator on a separable Hilbert space $\mathcal{H}$. Then, for any choice of α_t and β_t satisfying the usual conditions for stochastic interpolants, RWMH waltz for the mirror interpolant $p_0 = p_1 = \mathcal{N}(0, \mathcal{C})$ is identical to pCN.*

Proof. Computing the proposal kernel $Q(x_0, x_0') = \int p(x_0' \mid x_t) p(\mathrm{d}x_t \mid x_0)$ in closed form recovers exactly the pCN proposal. See Appendix B.1 for explicit computations. □

This result shows that we may understand RWMH waltz as a natural extension of pCN, where the fixed Gaussian is replaced by a data-driven prior. This is a key advantage when the true prior deviates strongly from Gaussianity, yet samples are available for learning.

Choice of the denoiser. Step 2 in diffusion waltz requires sampling from a denoising kernel $p(x_0 \mid x_t) \propto p(x_t \mid x_0)\, p_0(x_0)$. While the forward noising process $p(x_t \mid x_0)$ is straightforward to implement, denoising requires care: the SDE (2) only guarantees to match the marginal laws $p_t(x_t)$ of the stochastic interpolant, but not necessarily the conditional laws $p(x_s \mid x_t)$. Fortunately, there exists a choice of the diffusion coefficient σ_t that ensures correct sampling from $p(x_0 \mid x_t)$, as shown in (Potaptchik et al., 2026, Corollary C.1), paraphrased below.

Algorithm 1 Diffusion waltz (random walk)

1: Add noise $x_t \sim p(x_t \mid x_0)$ via interpolant
2: Denoise $x_0' \sim p(x_0 \mid x_t)$
3: Accept x_0' with probability

$$\alpha = \min\left\{1, \frac{p(y \mid x_0')}{p(y \mid x_0)}\right\}$$

(Set $x_0 \leftarrow x_0'$ if accepted, else keep x_0)

Algorithm 2 Diffusion waltz (conditional noising)

1: Propose $x_t \sim q(x_t \mid x_0, y)$ for a chosen kernel q
2: Denoise $x_0' \sim p(x_0 \mid x_t)$
3: Accept (x_t, x_0') with probability

$$\alpha = \min\left\{1, \frac{p(y \mid x_0')\, q(x_t \mid x_0', y)\, p(x_t \mid x_0)}{p(y \mid x_0)\, q(x_t \mid x_0, y)\, p(x_t \mid x_0')}\right\}$$

(Set $x_0 \leftarrow x_0'$ if accepted, else keep x_0)

Proposition 1 (Potaptchik et al. (2026)). *Choose σ_t in equation 2 such that*

$$\frac{\sigma_t^2}{2} = \frac{\beta_t(\alpha_t\dot{\beta}_t - \dot{\alpha}_t\beta_t)}{\alpha_t}, \quad t \in (0,1). \tag{3}$$

Then the terminal law of the reverse-time SDE (equation 2) integrated backwards from time t to 0 from initial condition x_t, is equivalent to the conditional law $p(x_0 \mid x_t)$ of the interpolant.

Thus, out of the infinite choice of SDEs that serve well as generative models, only one ensures correct conditional sampling $p(x_0 \mid x_t)$, required for diffusion waltz. In particular, this excludes the flow matching ODE, obtained by taking $\sigma_t \equiv 0$ in our denoising process. We may also employ *stochastic flow maps* during denoising (Potaptchik et al., 2026; Holderrieth et al., 2026; McCallum et al., 2026; Pan et al., 2026), which learns to directly sample from the posterior $p(x_0 \mid x_t)$ in one step, dramatically reducing the number of function evaluations.

Stability of RWMH waltz. In principle, RWMH waltz samples exactly from the posterior (1). However in practice, implementing this requires various approximations. These include (i) neural network approximations to the drift and/or score in the generative SDE (equation 2), leading to an approximate prior $p_\theta(x_0) \approx p_0(x_0)$; (ii) approximations to the denoising kernel $q_\theta(x_0 \mid x_t) \approx p(x_0 \mid x_t)$ due to numerical discretisation of the SDE (2), or the neural network used in stochastic flow maps; (iii) approximations to the likelihood $p(y \mid x)$, e.g., the forward model is given by a PDE solver, which we solve numerically, or use a neural network surrogate. The following result establishes the stability of RWMH waltz under approximations (i) and (ii), denoting by P^n_{true}, P^n_θ the laws of the true and approximate diffusion waltz Markov chains at step n, respectively.

Theorem 2 (Informal). *In the Euclidean setting $\mathcal{H} = \mathbb{R}^d$, define $V(x) := 1 + \|x\|^2$ and for $t \in (0,1)$, assume (i) bounds on the moments of $p_0(x_0)$, $p_\theta(x_0)$, $p(x_0 \mid x_t)$ and $q_\theta(x_0 \mid x_t)$, and (ii) the average acceptance rate stays above a threshold for x_0 far from the origin. Then, provided the denoiser error is sufficiently small, there exist positive constants $C, \kappa, \gamma < \infty$ and $\rho < 1$, such that*

$$\|P^n_{\text{true}} - P^n_\theta\|_V \le C\left(\rho^n \|p_0 - p_\theta\|_V + (1-\rho^n)\frac{\gamma\kappa}{1-\rho}\right), \tag{4}$$

for any $n \in \mathbb{N}$, where $\|\cdot\|_V$ is the V-weighted total variation norm. Moreover, γ is small if the denoiser error $\|p(\cdot \mid x_t) - q_\theta(\cdot \mid x_t)\|_{\text{tv}}$ is small; here $\|\cdot\|_{\text{tv}}$ is the standard total variation norm.

Proof. See Appendix B.2 for a more precise statement of the result and the full proof. □

Of the three error sources, the SDE discretisation error (ii) can, in principle, be corrected exactly by combining predictor–corrector-style sampling with further MH corrections (Lam et al., 2026). In practice, however, this is usually unnecessary to obtain satisfactory results. Regarding the likelihood error (iii), this is a standard setting considered in *approximate MCMC* and one can deduce V-norm error bounds to likelihood perturbations, e.g., from (Rudolf & Schweizer, 2018, Corollary 4.2).

3.2 Diffusion waltz with conditional noising

The RWMH diffusion waltz proposed above targets the posterior (1) exactly and is moreover stable to various approximations incurred during implementation. However, since its proposal $Q(x_0, x_0')$ only explores randomly around the current point x_0, it can be prone to low acceptance rates and, subsequently, slow mixing. To address this, we propose a novel extension that takes observation information into account during the proposal, which we refer to as *conditional noising waltz*. This guides

chains toward high likelihood regions during the proposal, akin to MALA (Roberts & Tweedie, 1996), leading to significantly faster mixing.

In its most general form, the noising step of RWMH waltz (step 1 in Algorithm 1) is replaced by $x_t \sim q(x_t \mid x_0, y)$ in conditional noising waltz, where q is a noising kernel that is designed to depend on the observation y; the denoising step remains unchanged. The resulting Metropolis-Hastings acceptance probability can then be computed as (see Appendix B.3.2 for the derivation)

$$\alpha = \min\left\{1, \frac{p(y \mid x_0')\, q(x_t \mid x_0', y)\, p(x_t \mid x_0)}{p(y \mid x_0)\, q(x_t \mid x_0, y)\, p(x_t \mid x_0')}\right\} \tag{5}$$

Remarkably, this acceptance probability still does not require evaluating the prior density p_0, making the method fully tractable as long as the noising kernel q can be evaluated. We summarise the three steps of conditional noising diffusion waltz in Algorithm 2.

Below, we provide two instances of the conditional noising kernel q: one that requires gradients of the forward model and one that is gradient-free. We assume that the likelihood takes the form $p(y \mid x) = \mathcal{N}(y \mid \mathcal{A}(x), \sigma_y^2 I)$, for some forward model $\mathcal{A} : \mathcal{H} \to \mathbb{R}^{d_y}$.

Gradient-based conditional noising. Suppose we have an update rule $x_{n+1} = x_n + g(x_n, y; h)$, for some step size $h > 0$ and descent direction $g(\cdot)$, that minimises the data misfit $\frac{1}{2}\|y - \mathcal{A}(x)\|^2 = -\log p(y \mid x)$. We consider a conditional noising kernel $x_t \sim q(x_t \mid x_0, y)$, given by

$$x_t = \alpha_t\big(x_0 + g(x_0, y; h)\big) + \beta_t x_1, \quad x_1 \sim p_1, \tag{6}$$

which applies a single step of said learning rule, followed by noising via the interpolant. Combined with denoising and MH correction, we may understand the full conditional waltz procedure as cycling through (i) minimising the data misfit by taking a single gradient update on x_0, which may initially push it off the data manifold, (ii) applying an SDEdit loop (noise/denoise) to project it back onto the data manifold, and (iii) an MH step to ensure exactness (see Figure 1b for an illustration).

For the descent direction g, a practical choice is Levenberg-Marquardt (LM) (Hanke, 1997):

$$g(x, y; h) = J_x^\top \left(J_x J_x^\top + h^{-1}\sigma_y^2 I\right)^{-1} (y - \mathcal{A}(x)), \tag{7}$$

where $J_x := D\mathcal{A}(x)$ is the Jacobian of the forward model. The cost of computing (7) is $\mathcal{O}(d_y^3 + d_x d_y^2)$, which is feasible for small enough observation dimension d_y. Compared to first-order gradient descent, the push to minimise data misfit using LM is sufficiently strong to yield notable gains in acceptance rate over RWMH waltz. We also note that, in contrast to common guidance methods such as diffusion posterior sampling (DPS) (Chung et al., 2023), conditional noising waltz applies gradients directly at the level of clean data x_0 instead of on partially noised data x_t, avoiding the need to estimate x_0 from x_t, for example via Tweedie's formula.

Ensemble Kalman conditional noising. When gradients of $\mathcal{A}$ are not available, we can approximate the LM update (7) using statistical linearisation, defining our *ensemble Kalman conditional noising (EKCN)* kernel. Precisely, we consider running J chains $\{x_0^{(j)}\}_{j=1}^J$ in parallel, each initialised from an unconditional sample of p_0, and updated according to the conditional noising rule

$$x_t^{(j)} = \alpha_t\left(x_0^{(j)} + K(h)\big(y - \mathcal{A}(x_0^{(j)})\big)\right) + \beta_t x_1^{(j)}, \quad x_1^{(j)} \overset{\text{i.i.d.}}{\sim} p_1, \tag{8}$$

for $j = 1, \ldots, J$. Here, $K(h) := C^{xy}(C^{yy} + h^{-1}\sigma_y^2 I)^{-1}$ is the *Kalman gain*, where C^{xy} is the empirical cross-covariance between variables x_0 and $\mathcal{A}(x_0)$, and C^{yy} is the covariance of $\mathcal{A}(x_0)$ (Sanz-Alonso et al., 2023). We may also invoke the push-through identity to reduce the cost of computing $K(h)v$ for $v \in \mathbb{R}^{d_y}$ from $\mathcal{O}(d_y^3 + d_x d_y^2)$ to $\mathcal{O}(J^3 + J^2 d_y + J d_x)$, making the computation tractable even when the observation dimension is large, provided $J \ll d_y$ (see Appendix C.1.1). Strictly speaking, since $K(h)$ is computed using the full ensemble, the chains become dependent on one another, so we no longer preserve exact reversibility. This makes conditional noising waltz with this kernel slightly biased. Empirically, however, we find that it yields strong results, significantly accelerating mixing compared to RWMH waltz and ultimately achieving better performance despite RWMH waltz's exactness guarantees.

Remark 1 (Conditional denoising)**.** *While we have only considered adding observation information during noising in conditional noising waltz, we could have alternatively added it during denoising (step 2), leading to what we call "conditional denoising" waltz: (i) noise $x_t \sim p(x_t \mid x_0)$, (ii) denoise conditionally $x_0' \sim p(x_0' \mid x_t, y)$, and (iii) accept x_0' unconditionally. Unlike RWMH or conditional noising waltz, this proposal is itself reversible with respect to the posterior $p(x_0 \mid y)$, so acceptance is automatic and no MH correction is required. Instances of this approach have appeared in the literature, e.g. DAPS (Zhang et al., 2025) and RePS (Ahmed & Makin, 2025), although not framed from an MCMC perspective. A key challenge with this approach, however, is precisely how to propose a sample from the conditional denoising kernel $p(x_0' \mid x_t, y)$; DAPS uses an unadjusted Langevin algorithm targeting this kernel, and RePS uses DPS-style denoising (using MAP evaluation instead of a Tweedie estimate), both of which introduce irreducible bias.*

Remark 2 (Adaptive MCMC)**.** *Following Andrieu & Thoms (2008), we find that adaptively tuning the hyperparameters instead of fixing them improves the performance of MCMC, especially for conditional noising waltz. We defer the details of our adaptive tuning strategy to Appendix D.*

4 Related Works

We note that the RWMH waltz proposal (Algorithm 1) was arrived at independently in two concurrent works (Kang et al., 2026; Phunyaphibarn & Sung, 2026). Kang et al. (2026) studies the ergodic properties of the Markov chain generated by the SDEdit-style proposal $Q(x_0, x_0')$; while the possibility of applying this proposal to MCMC is mentioned, it is not the core focus of the work. Phunyaphibarn & Sung (2026) is closer in spirit to ours, applying RWMH waltz to achieve reward alignment with discrete diffusion models. However, neither work explores mechanisms for injecting observation information into the proposal, establishes the concrete connection to pCN, nor provides a stability analysis justifying the use of an approximate denoising kernel. Our conditional noising mechanism makes the algorithm competitive in a wide range of inverse problems in science and engineering; we demonstrate this by benchmarking against competing methods in Section 5.

Various existing works have also drawn from MCMC to propose diffusion posterior samplers with asymptotic guarantees: TDS (Wu et al., 2023), MCGDiff (Cardoso et al., 2024) and FK Steering (Singhal et al., 2025) use sequential Monte Carlo (SMC) during the denoising process, which inherit the curse-of-dimensionality of SMC, leading to collapsed samples. Methods based on split-Gibbs sampling (Coeurdoux et al., 2024; Wu et al., 2024; Zheng et al., 2025c) overcome this issue, although the use of Langevin sampler in the inner loop may lead to numerical instability (Zheng et al., 2025b), especially as the forward model is evaluated on noised samples. ESS-Flow (Kalaivanan et al., 2025) on the other hand, exploit Gaussianity to sample from the source-space posterior. However, it is non-trivial to exploit gradients with this approach, in addition to requiring many forward model during shrinking leading to higher cost. Diffusion waltz serves as an alternative, complementary approach where these methods struggle or do not apply, as we show in our experiments in the next section.

5 Experiments

Posterior accuracy. To test RWMH diffusion waltz (DW)'s ability to sample from the posterior accurately, we consider the checkerboard inverse problem in Mammadov et al. (2026); a 4×4 checkerboard pattern serves as the prior and the likelihood is given by observing the x-coordinate with additive Gaussian noise. We consider a ground truth posterior via rejection sampling and evaluate posterior sampling performance using maximum mean discrepancies (MMD) (Gretton et al., 2012) against it. To isolate the performance of the method itself, we use a "true" drift $b(t, x) := \mathbb{E}[\dot{x}_t | x_t = x]$, computed via Monte-Carlo (see Appendix E.2.1 for details). For the denoising step, we consider (a) an SDE denoiser in Proposition 1, (b) an ODE denoiser, and (c) a single-step denoiser learned via meta flow map (MFM, Potaptchik et al. (2026)). We additionally include MMPS (Rozet et al., 2024) as a reference baseline.

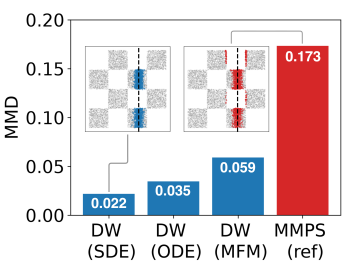
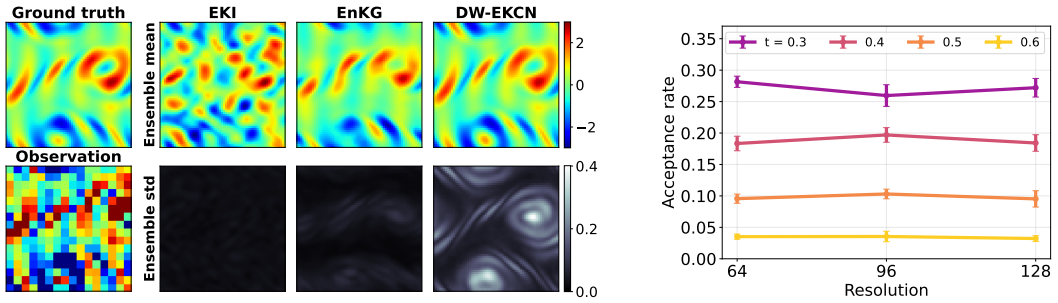


Figure 2: MMD comparisons of DW and MMPS on the checkerboard inverse problem.



(a) Comparisons on NS (high nonlinearity/low noise).

(b) Resolution robustness.

Figure 3: **(a)** Ensemble mean and standard deviation for EKI, EnKG and DW-EKCN. DW-EKCN produces on-manifold reconstructions while maintaining meaningful ensemble spread. **(b)** RWMH waltz acceptance rates remain stable across resolutions for each fixed noising time t.

Figure 2 shows MMD against the ground truth posterior. All DW variants outperform MMPS by a large margin: visually, we see that a small but non-negligible fraction of MMPS samples fall in cells where the true posterior has vanishing mass, which we do not observe for DW. Among the DW variants, the SDE denoiser slightly outperforms the ODE denoiser, consistent with Proposition 1. We also observed that while the ODE denoiser has a higher acceptance rate ($\sim 33\%$ vs. 25% for SDE), it has slightly lower effective sample size (3.89% vs. 4.39%). The MFM denoiser yields slightly worse MMD than both, but runs significantly faster ($\sim$ 70s vs. 1200s for SDE, for a chain of $100,000$ samples on a single A100 GPU); our SDE denoiser uses an accurate drift, leading to a tighter error bound in Theorem 2, explaining its better performance. We also test conditional noising waltz's accuracy on a Gaussian process regression problem in Appendix E.2.2.

Resolution robustness. In Theorem 1, we showed that RWMH waltz naturally extends pCN to learned prior settings. A key property of pCN is *resolution robustness*, which allows it to maintain similar acceptance rates even as finer discretisations of functions in $\mathcal{H}$ are used (Cotter et al., 2013). To test whether RWMH waltz retains this property, we consider the Navier-Stokes initial condition recovery problem, where an initial vorticity field ω_0 is inferred from a noisy, coarse representation y of the vorticity field ω_T at time T (we set $T = 1$ here). For the prior, we use functional flow matching (FFM) (Kerrigan et al., 2023)) trained on snapshots of the Navier-Stokes vorticity fields from Li et al. (2022) at resolution 64×64, and fine-tuned on the same data at resolutions 96×96 and 128×128 (see Appendix E.3 for experimental details). We compute running acceptance rates for chains (5000 steps with a 1000-step burn-in) at each resolutions of ω_0 using the same observation y throughout. Figure 3b shows acceptance rates for RWMH waltz chains at noising times $t = 0.3, 0.4, 0.5, 0.6$ (error bars indicate 1 std over ten initialisations). As expected, acceptance rates are similar across the three resolutions for each t, and increase as t decreases, i.e., as the noising step becomes smaller.

Baseline comparisons on Navier-Stokes. We compare diffusion waltz against various baselines on the Navier-Stokes initial condition recovery problem, across regimes varying in the level of non-linearity (controlled by T) and noise intensity (controlled by σ_y) (see Appendix E.3). Since we use a non-differentiable numerical solver to simulate Navier-Stokes, we restrict our comparisons to derivative-free approaches: EKI (Iglesias et al., 2013), EnKG (Zheng et al., 2025a), FK Steering (Singhal et al., 2025), ESS-Flow (Kalaivanan et al., 2025), Blade (Zheng et al., 2025c) and MFM-GF (Potaptchik et al., 2026) (details are provided in Appendix E.3). For diffusion waltz, we consider RWMH waltz (DW-RWMH) and EKCN waltz (DW-EKCN), which are both derivative-free.

Table 1 reports the relative ℓ_2 score (Rel-ℓ_2), continuous ranked probability score (CRPS), and spread-skill ratio (SSR), giving a mix of reconstruction and probabilistic metrics (see Appendix E.1 for details). DW-EKCN performs strongly across all metrics, significantly improving over EKI and EnKG, which also draw on ensemble Kalman methodology: EKI, being essentially a maximum likelihood estimator, fails to produce on-manifold samples, leading to poor Rel-ℓ_2. EnKG resolves the off-manifold issue but produces low uncertainty, resulting in very low SSR. DW-EKCN resolves both issues, producing on-manifold samples while maintaining diversity (Figure 3a). Among asymptotically exact methods, FK steering (based on SMC) suffers from weight collapse, giving poor results across all metrics; MFM-GF, on the other hand, shows underconfident predictions, as indicated

Table 1: Baseline comparisons on Navier-Stokes. We display the mean Rel-ℓ_2, CRPS and SSR across five unseen test examples. Best results in **bold**, second best underlined. Results for DW are highlighted.

	Low nonlinearity ($T = 1$)						High nonlinearity ($T = 5$)					
	Low noise			High noise			Low noise			High noise		
Methods	Rel $\ell_2 \downarrow$	CRPS$\downarrow$	SSR$\rightarrow 1$	Rel $\ell_2 \downarrow$	CRPS$\downarrow$	SSR$\rightarrow 1$	Rel $\ell_2 \downarrow$	CRPS$\downarrow$	SSR$\rightarrow 1$	Rel $\ell_2 \downarrow$	CRPS$\downarrow$	SSR$\rightarrow 1$
EKI	0.517	–	–	0.832	–	–	1.059	–	–	0.865	–	–
EnKG	0.129	0.086	0.076	0.617	0.414	0.071	0.418	0.279	0.098	0.812	0.523	0.084
FK Steering	0.787	0.538	0.012	0.782	0.500	0.101	1.383	1.000	0.016	0.992	0.511	1.084
ESS-Flow	0.178	0.080	1.405	0.248	0.118	1.179	0.506	0.242	**0.987**	0.491	0.233	**0.987**
Blade	**0.069**	**0.037**	1.797	0.296	0.139	1.154	0.421	0.239	2.312	0.442	0.231	1.326
MFM-GF	0.406	0.191	**1.395**	0.425	0.202	1.406	0.549	0.262	1.245	0.547	0.261	1.253
DW-RWMH	0.451	0.177	1.703	0.550	0.229	1.476	0.810	0.398	1.163	0.815	0.396	1.155
DW-EKCN	0.086	0.045	1.396	**0.220**	**0.101**	**1.081**	**0.247**	**0.152**	1.054	**0.435**	**0.207**	0.945

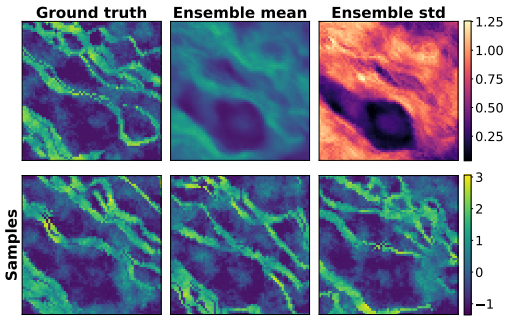


Figure 4: Results of DW-AdCN on Darcy flow, showing accurate, on-manifold reconstructions.

Methods	Rel $\ell_2 \downarrow$	CRPS$\downarrow$	SSR$\rightarrow 1$
Traditional			
LM	1.456 (0.045)	–	–
EKI*	1.397 (0.126)	–	–
pCN-Langevin	0.902 (0.057)	0.612 (0.041)	0.228 (0.041)
Plug-and-play			
DPS	0.832 (0.026)	0.435 (0.013)	0.916 (0.041)
DAPS	1.007 (0.008)	0.573 (0.018)	0.831 (0.018)
PnP-DM	1.017 (0.004)	0.620 (0.019)	0.625 (0.014)
Blade*	1.003 (0.004)	0.575 (0.013)	1.107 (0.021)
TDS	1.630 (0.411)	1.129 (0.246)	0.000 (0.000)
Diffusion waltz			
DW-EKCN*	0.800 (0.082)	0.422 (0.051)	0.723 (0.050)
DW-AdCN	**0.790 (0.022)**	**0.409 (0.025)**	**0.977 (0.046)**

Table 2: Baseline comparisons on Darcy flow inversion. * indicates derivative-free methods.

by its high SSR. Blade and ESS-Flow are more competitive, with Blade outperforming DW-EKCN on Rel-ℓ_2 and CRPS in the low noise / low nonlinearity regime, and ESS-Flow achieving slightly better SSR in the high nonlinearity regime. However, both methods (especially ESS-Flow) suffer from slow inference due to the many forward map passes required in their inner loops (see Table 6). DW-RWMH, while being asymptotically exact, also performs poorly due to the slow convergence of the chains (see Figure 11 for the trace plots). This highlights the importance of conditional noising in this problem to accelerate mixing.

Darcy flow inversion. We now consider the inverse Darcy flow problem, where the goal is to infer the permeability field of subsurface flow from noisy point observations of the corresponding pressure field (details in Appendix E.4). The standard Darcy flow benchmark used in, e.g., Li et al. (2021); Yao et al. (2026) uses Gaussian process-derived permeability fields, which is solvable using classical methods such as pCN. Thus, we opt for a more complex dataset in Seabra et al. (2026), whose permeability fields follow a highly non-Gaussian distribution (see Figure 13 for example fields). We assume access to gradients through the adjoint model, and consider a mix of derivative-based and derivative-free baselines, including classical methods (Levenberg-Marquardt (LM), ensemble Kalman inversion (EKI), pCN-Langevin), and recent plug-and-play diffusion samplers for inverse problems (DPS, DAPS, PnP-DM, Blade, and TDS). For diffusion waltz, in addition to EKCN, we consider gradient-based conditional noising using the adjoint model (DW-AdCN).

Table 2 shows our results: among traditional methods, LM and EKI, being pure optimisers, produce poor results, indicating the need for a prior for accurate reconstruction. pCN-Langevin, which uses a Gaussian process prior, reconstructs better, but its uncertainty is poorly calibrated (low SSR) due to prior mismatch. Among the plug-and-play baselines, methods relying on Langevin samplers (DAPS, PnP-DM, Blade) struggle, consistent with observations made in Zheng et al. (2025b), which reports similar instability for DAPS and PnP-DM on a similar waveform inversion problem. PnP-DM and Blade further require evaluating the forward model on noisy versions of clean data, further promoting instability. TDS (Wu et al., 2023), based on SMC, performs poorly due to particle collapse. DPS, while the most competitive of the baselines, is outperformed by DW: EKCN waltz tends to produce slightly biased, overconfident results, likely due to the inexactness of the algorithm. DW-AdCN does not share this limitation and performs strongest across all metrics.

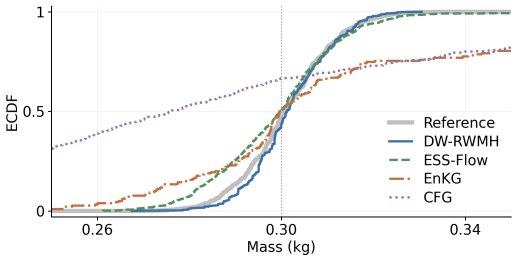


Figure 5: Empirical CDF of posterior mass samples against importance weighted reference.

Method	Bias / 10^{-3} ($\to 0$)	$\hat{\sigma}_y/\sigma_y$ ($\to 1$)	ESS/1k ↑	MMD ↓
Guidance				
CFG	-0.80 (4.8)	10.33 (0.80)	–	0.656
EnKG	+19.2 (4.8)	6.18 (0.71)	–	0.313
MCMC				
ESS-Flow	-1.4 (1.1)	1.71 (0.25)	**35**	0.171
DW-RWMH	**+0.8 (0.8)**	**0.91 (0.005)**	32	**< 0.081**

Table 3: Comparisons on 3D inverse design.

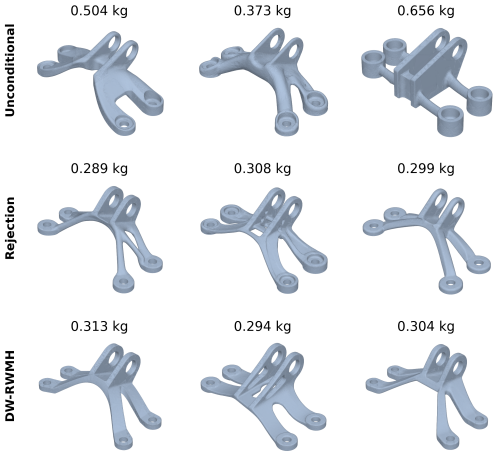


Figure 6: Random prior & posterior samples.

3D inverse design. In our final experiment, we consider an application to *3D inverse design*, where the goal is to generate 3D geometries with target physical properties. For our prior we use TRELLIS.2 (Xiang et al., 2026) fine-tuned on the SimJEB dataset (Whalen et al., 2021), consisting of diverse jet engine bracket designs. We consider the task of generating brackets from the "beam" category with target mass $y = 0.3\,\text{kg}$ and tolerance $\sigma_y = 0.01\,\text{kg}$, formulated as a Bayesian inverse problem (1), where the forward operator $\mathcal{A}$ evaluates the mass of a geometry x. As output meshes are non-manifold, evaluating $\mathcal{A}$ requires non-differentiable postprocessing. A further challenge is that TRELLIS.2 generates geometry in two-stages, first sampling a latent variable z, before generating a mesh $x \sim p(x \mid z)$, a common paradigm in frontier 3D generative models that is not well supported by existing plug-and-play methods. RWMH waltz, on the other hand, extends naturally to two-stage generation while preserving exactness, as we show in Appendix E.5.

Results are shown in Table 3. Reference statistics for bias and calibration are obtained via importance weighting of a large sample from the prior, with all benchmarks given a similar budget of approximately 5k forward model evaluations. For the MCMC methods, we discard burn-in and compute autocorrelation based on mass; ESS/1k is defined as the resulting independent samples per 1k evaluations of $\mathcal{A}$. MMD is similarly computed on mass distributions, using rejection sampling as the ground truth. Though commonly used for inverse design, CFG doesn't target the posterior so performs poorly on distributional metrics. EnKG and ESS-Flow necessitate modifications for two step sampling, described in Appendix E.5. The former is challenging to tune, often suffering from mode collapse or under-confidence. ESS-Flow performs stronger, but noise introduced by the two stage modifications ultimately impact performance. RWMH waltz requires no such approximations, and despite its simplicity scores best across all accuracy metrics.

6 Conclusion

Diffusion waltz provides a simple, flexible framework for accurate posterior sampling using diffusion priors, applicable for any tasks involving differentiable or non-differentiable forward models. Our new conditional noising mechanism allows injection of observation information during proposal, substantially accelerating convergence. Limitations of our method include: (1) Running a long chain is expensive, requiring many NFEs and forward model passes. However, conditional noising accelerates convergence enough to often outpace other asymptotically exact methods (PnP-DM, Blade, ESS-Flow), which rely on expensive inner loops. (2) Low acceptance rates when likelihood is peaky. While adaptive MCMC helps, a more tailored adaptive tuning strategy may be needed. We believe that further work exploring better tuning strategies, such as promoting t to a random variable, similar to Kim & Ye (2022), or developing better heuristics for adaptive tuning are promising directions. In addition, we may borrow ideas from multilevel MCMC Lykkegaard et al. (2023) to accelerate sampling and improve efficiency further.

AI USE STATEMENT

In this work, we used generative AI tools to implement methods, assist with the mathematical proofs and generate feedbacks on the methodology/experiments. We have not used generative AI tools to propose hypotheses and develop the core conceptual ideas. Use of generative AI to generate synthetic datasets, provide translations and reformatting datasets is not applicable in this work. Additionally, we used generative AI tools to help create the figures, assist with the writing and help identify the relevant literature. We have reviewed all AI-assisted work: LLM-generated code was verified and tested extensively by the authors, and AI-assisted mathematical claims were carefully checked by proof-reading and consulting the original sources. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Bilal Ahmed and Joseph G Makin. Solving diffusion inverse problems with restart posterior sampling. *arXiv preprint arXiv:2511.20705*, 2025.

Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

Christophe Andrieu and Johannes Thoms. A tutorial on adaptive MCMC. *Statistics and computing*, 18(4):343–373, 2008.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015.

Gabriel Cardoso, Sylvain Le Corff, Eric Moulines, et al. Monte carlo guided denoising diffusion models for Bayesian linear inverse problems. In *International Conference on Learning Representations*, volume 2024, pp. 44001–44037, 2024.

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

Florentin Coeurdoux, Nicolas Dobigeon, and Pierre Chainais. Plug-and-play split Gibbs sampler: embedding deep generative priors in Bayesian inference. *IEEE Transactions on Image Processing*, 33:3496–3507, 2024.

Simon L Cotter, Gareth O Roberts, Andrew M Stuart, and David White. MCMC methods for functions: modifying old algorithms to make them faster. *Statistical Science*, pp. 424–446, 2013.

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.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Geir Evensen. The ensemble Kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics*, 53(4):343–367, 2003.

Jonathan Goodman and Jonathan Weare. Ensemble samplers with affine invariance. *Communications in applied mathematics and computational science*, 5(1):65–80, 2010.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Martin Hanke. A regularizing Levenberg-Marquardt scheme, with applications to inverse groundwater filtration problems. *Inverse problems*, 13(1):79–95, 1997.

W Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 1970.

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. *Advances in neural information processing systems*, 30, 2017.

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

Peter Holderrieth, Douglas Chen, Luca Eyring, Ishin Shah, Giri Anantharaman, Yutong He, Zeynep Akata, Tommi Jaakkola, Nicholas Matthew Boffi, and Max Simchowitz. Diamond maps: Efficient reward alignment via stochastic flow maps. *arXiv preprint arXiv:2602.05993*, 2026.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Marco A Iglesias, Kody JH Law, and Andrew M Stuart. Ensemble Kalman methods for inverse problems. *Inverse Problems*, 29(4):045001, 2013.

Adhithyan Kalaivanan, Zheng Zhao, Jens Sjölund, and Fredrik Lindsten. ESS-Flow: Training-free guidance of flow-based models as inference in source space. *arXiv preprint arXiv:2510.05849*, 2025.

Hyunmo Kang, Noam Itzhak Levi, Corinna Elena Wegner, Daniel J Korchinski, and Matthieu Wyart. Sampling data with chains of forward-backward diffusion steps. *arXiv preprint arXiv:2605.27006*, 2026.

Gavin Kerrigan, Giosue Migliorini, and Padhraic Smyth. Functional flow matching. *arXiv preprint arXiv:2305.17209*, 2023.

Beomsu Kim and Jong Chul Ye. Denoising MCMC for accelerating diffusion-based generative models. *arXiv preprint arXiv:2209.14593*, 2022.

Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. FlowDPS: Flow-driven posterior sampling for inverse problems. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12328–12337. IEEE, 2025.

Kevin H Lam, Tyler Farghly, Christopher Williams, Jun Yang, Yee Whye Teh, and Arnaud Doucet. Metropolis-adjusted diffusion models. *arXiv preprint arXiv:2605.09654*, 2026.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.

Zongyi Li, Miguel Liu-Schiaffini, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35:16768–16781, 2022.

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

Mikkel Bue Lykkegaard, Tim J Dodwell, Colin Fox, Grigorios Mingas, and Robert Scheichl. Multilevel delayed acceptance MCMC. *SIAM/ASA journal on uncertainty quantification*, 11(1):1–30, 2023.

Abbas Mammadov, So Takao, Bohan Chen, Ricardo Baptista, Morteza Mardani, Yee Whye Teh, and Julius Berner. Variational flow maps: Make some noise for one-step conditional generation. *International Conference on Machine Learning*, 2026.

Sam McCallum, Zander W Blasingame, Timothy Herschell, Niklas Rindtorff, Alexander Tong, and James Foster. Strong stochastic flow maps. *arXiv preprint arXiv:2606.01086*, 2026.

Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

Kerrie L Mengersen and Richard L Tweedie. Rates of convergence of the Hastings and Metropolis algorithms. *The annals of Statistics*, 24(1):101–121, 1996.

Zhengkai Pan, Peter Potaptchik, Wenxi Yao, Michael S Albergo, and Jakiw Pidstrigach. Itô maps for any-step SDEs. *arXiv preprint arXiv:2606.11156*, 2026.

Prin Phunyaphibarn and Minhyuk Sung. Reward-guided discrete diffusion via clean-sample Markov chain for molecule and biological sequence design. *arXiv preprint arXiv:2602.09424*, 2026.

Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael S Albergo, and Yee Whye Teh. Meta flow maps enable scalable reward alignment. *arXiv preprint arXiv:2601.14430*, 2026.

Gareth O Roberts and Richard L Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 1996.

François Rozet, Gérôme Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. *Advances in Neural Information Processing Systems*, 37:87647–87682, 2024.

Daniel Rudolf and Nikolaus Schweizer. Perturbation theory for Markov chains via Wasserstein distance. *Bernoulli*, pp. 2610–2639, 2018.

Daniel Sanz-Alonso, Andrew Stuart, and Armeen Taeb. *Inverse problems and data assimilation*, volume 107. Cambridge University Press, 2023.

Gabriel Serrão Seabra, Nikolaj T Mücke, Vinicius Luiz Santos Silva, Alexandre A Emerick, Denis Voskov, and Femke C Vossepoel. Integrating score-based diffusion models with machine learning-enhanced localization for advanced data assimilation in geological carbon storage. *Computational Geosciences*, 30(4):68, 2026.

Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. *arXiv preprint arXiv:2501.06848*, 2025.

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

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

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

Eamon Whalen, Azariah Beyene, and Caitlin Mueller. SimJEB: simulated jet engine bracket dataset. In *Computer Graphics Forum*, volume 40, pp. 9–17. Wiley Online Library, 2021.

Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. *Advances in Neural Information Processing Systems*, 36:31372–31403, 2023.

Zihui Wu, Yu Sun, Yifan Chen, Bingliang Zhang, Yisong Yue, and Katherine L Bouman. Principled probabilistic imaging using diffusion models as plug-and-play priors. *Advances in Neural Information Processing Systems*, 37:118389–118427, 2024.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. Native and compact structured latents for 3D generation. In *Conference on Computer Vision and Pattern Recognition*, 2026.

Jiachen Yao, Abbas Mammadov, Julius Berner, Gavin Kerrigan, Jong Chul Ye, Kamyar Azizzadenesheli, and Animashree Anandkumar. Guided diffusion sampling on function spaces with applications to PDEs. *Advances in Neural Information Processing Systems*, 38:127057–127094, 2026.

James Boran Yu, RuiKang OuYang, Julien Horwood, and José Miguel Hernández-Lobato. Stochastic interpolants in Hilbert spaces. *arXiv preprint arXiv:2602.01988*, 2026.

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

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

Hongkai Zheng, Wenda Chu, Austin Wang, Nikola Kovachki, Ricardo Baptista, and Yisong Yue. Ensemble Kalman diffusion guidance: A derivative-free method for inverse problems. *Transactions on Machine Learning Research*, 2025a.

Hongkai Zheng, Wenda Chu, Bingliang Zhang, Zihui Wu, Austin Wang, Berthy Feng, Caifeng Zou, Yu Sun, Nikola Kovachki, Zachary Ross, et al. InverseBench: Benchmarking plug-and-play diffusion priors for inverse problems in physical sciences. In *International Conference on Learning Representations*, volume 2025, pp. 90912–90940, 2025b.

Hongkai Zheng, Austin Wang, Zihui Wu, Zhengyu Huang, Ricardo Baptista, and Yisong Yue. Blade: A derivative-free Bayesian inversion method using diffusion priors. *arXiv preprint arXiv:2510.10968*, 2025c.

A Background on stochastic interpolants

Given measures p_0 and p_1, a stochastic interpolant is defined by the stochastic process

$$x_t = \alpha_t x_0 + \beta_t x_1, \tag{9}$$

where $x_0 \sim p_0$, $x_1 \sim p_1$, and α_t, β_t for $t \in [0,1]$ are arbitrary monotonic C^2 functions satisfying the endpoint conditions $\alpha_0 = \beta_1 = 1$ and $\alpha_1 = \beta_0 = 0$. By a slight extension of (Albergo et al., 2025, Corollary 18), for any positive definite $\{C_t\}_{t\in[0,1]}$, we have that the SDE

$$\mathrm{d}x_t = \left(b(t,x_t) - \frac{1}{2}C_t s(t,x_t)\right)\mathrm{d}t + \sqrt{C_t}\mathrm{d}W_t, \tag{10}$$

solved backwards in time from $x_1 \sim p_1$ shares the same marginal law $p(x_t)$ as the forward interpolant (9). Here, $b(t,x_t) := \mathbb{E}[\dot{x}_t | x_t]$ is the marginal velocity (as used in flow matching) and $s(t,x_t) := \nabla_{x_t} \log p(x_t)$ is the score function. While both $b(t,x_t)$ and $s(t,x_t)$ are intractable, these can be approximated by neural networks $b_\theta(t,x_t)$, $s_\theta(t,x_t)$, respectively, by minimizing the losses

$$\mathcal{L}_b(\theta) = \mathbb{E}_{x_0\sim p_0, x_1\sim p_1, t\sim\mathcal{U}([0,1])}\Big[\|b_\theta(t,x_t) - (\dot{\alpha}_t x_0 + \dot{\beta}_t x_1)\|^2\Big], \tag{11}$$

$$\mathcal{L}_s(\theta) = \mathbb{E}_{x_0\sim p_0, x_1\sim p_1, t\sim\mathcal{U}([0,1])}\Big[\|s_\theta(t,x_t) - \nabla_{x_t}\log p(x_t \mid x_0)\|^2\Big]. \tag{12}$$

A.1 Three-sides of the same coin

In the special case when the reference is given by a Gaussian $p_1 = \mathcal{N}(0,C)$[2], we can show that the drift $b(t,x_t)$, the score $s(t,x_t)$ and the clean sample estimate $\mathbb{E}[x_0 \mid x_t]$ are all related linearly. First, we can easily check the relation between $s(t,x_t)$ and $\mathbb{E}[x_0 \mid x_t]$, noting that $p(x_t \mid x_0) = \mathcal{N}(x_t \mid \alpha_t x_0, \beta_t^2 C)$:

$$s(t,x_t) := \nabla_{x_t}\log p(x_t) \tag{13}$$

$$= \int \nabla_{x_t}\log p(x_t \mid x_0) p(x_0 \mid x_t)\mathrm{d}x_0 \tag{14}$$

$$= -\beta_t^{-2} C^{-1}\big(x_t - \alpha_t \mathbb{E}[x_0 \mid x_t]\big). \tag{15}$$

[2]We consider a general covariance C here to adapt to the functional flow matching (FFM) architecture Kerrigan et al. (2023) used in our Navier-Stokes experiments. FFM uses a Matérn GP covariance instead of white noise.

Re-arranging this expression gives us Tweedie's formula:

$$\mathbb{E}[x_0 \mid x_t] = \frac{x_t + \beta_t^2 C s(t, x_t)}{\alpha_t} \tag{16}$$

On the other hand, using the fact that $x_t = \alpha_t x_0 + \beta_t x_1$ implies $x_t = \alpha_t \mathbb{E}[x_0 \mid x_t] + \beta_t \mathbb{E}[x_1 \mid x_t]$ and $b(t, x_t) := \mathbb{E}[\dot{x}_t \mid x_t] = \dot{\alpha}_t \mathbb{E}[x_0 \mid x_t] + \dot{\beta}_t \mathbb{E}[x_1 \mid x_t]$, a simple algebra yields the relation between the drift $b(t, x_t)$ and the Tweedie estimate $\mathbb{E}[x_0 \mid x_t]$:

$$\mathbb{E}[x_0 \mid x_t] = \frac{\beta_t b(t, x_t) - \dot{\beta}_t x_t}{\dot{\alpha}_t \beta_t - \alpha_t \dot{\beta}_t}. \tag{17}$$

Finally, combining equation 16 and equation 17 gives us the relation between the drift $b(t, x)$ and the score $s(t, x_t)$:

$$C s(t, x_t) = \frac{\alpha_t b(t, x_t) - \dot{\alpha}_t x_t}{\beta_t (\dot{\alpha}_t \beta_t - \alpha_t \dot{\beta}_t)}. \tag{18}$$

A.2 Choice of SDE to sample from $p(x_0 \mid x_t)$

Given particular choices of C_t, we can show that the SDE equation 10 not only samples the interpolant marginals $p(x_t)$ *but also the conditional* $p(x_0 \mid x_t)$, which is what we need in the denoising step of diffusion waltz. This follows from the following result.

Proposition 2. *Let $p_1 = \mathcal{N}(0, C)$ for some positive definite C. Then, setting $C_t := \sigma_t^2 C$ in equation 10 with $\{\sigma_t\}_{t \in (0,1)}$ satisfying*

$$\frac{\sigma_t^2}{2} = \frac{\beta_t(\alpha_t \dot{\beta}_t - \dot{\alpha}_t \beta_t)}{\alpha_t}, \quad t \in (0, 1), \tag{19}$$

the terminal law of the reverse-time SDE (equation 10) integrated backwards from time t to 0 from initial condition x_t, is equivalent to the conditional law $p(x_0 \mid x_t)$ of the interpolant.

Proof. The proof is a slight adaptation of (Potaptchik et al., 2026, Proposition C.1). □

Using this result, we can further simplify the expression for our denoising SDE equation 10. First, by the relation equation 18, the drift of equation 10 reads

$$b(t, x_t) - \frac{\sigma_t^2}{2} C s(t, x_t) = b(t, x_t) - \frac{\sigma_t^2}{2} \frac{\alpha_t b(t, x_t) - \dot{\alpha}_t x_t}{\beta_t (\dot{\alpha}_t \beta_t - \alpha_t \dot{\beta}_t)} \tag{20}$$

$$\overset{(19)}{=} b(t, x_t) + \left(b(t, x_t) - \frac{\dot{\alpha}_t}{\alpha_t} x_t \right) \tag{21}$$

$$= 2b(t, x_t) - \frac{\dot{\alpha}_t}{\alpha_t} x_t \tag{22}$$

or equivalently,

$$b(t, x_t) - \frac{\sigma_t^2}{2} C s(t, x_t) = \frac{\dot{\alpha}_t}{\alpha_t} x_t - \sigma_t^2 C s(t, x_t). \tag{23}$$

Thus, the denoising SDE equation 10 simplifies to

$$\mathrm{d}x_t = \left(2b(t, x_t) - \frac{\dot{\alpha}_t}{\alpha_t} x_t \right) \mathrm{d}t + \sigma_t \mathrm{d}W_t^C, \tag{24}$$

or equivalently, using the score instead, we have

$$\mathrm{d}x_t = \left(\frac{\dot{\alpha}_t}{\alpha_t} x_t - \sigma_t^2 C s(t, x_t) \right) \mathrm{d}t + \sigma_t \mathrm{d}W_t^C, \tag{25}$$

where $W_t^C := \sqrt{C} W_t$ is the C-Brownian motion.

B PROOFS OF THE MAIN RESULTS

B.1 DIFFUSION WALTZ GENERALISES PRECONDITIONED CRANK-NICOLSON (PCN)

Lemma 1 (Stuart (2010), Theorem 6.20)**.** *For a separable Hilbert space $\mathcal{H}$ with inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, let $(x_1, x_2) \in \mathcal{H} \oplus \mathcal{H}$ be a Gaussian random variable with mean $m := (m_1, m_2) \in \mathcal{H} \oplus \mathcal{H}$ and positive definite covariance operator $\mathcal{C} \in \mathcal{L}(\mathcal{H} \oplus \mathcal{H}, \mathcal{H} \oplus \mathcal{H})$. Define the cross-covariance operators $\mathcal{C}_{ij} \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ for $i, j \in \{1, 2\}$ by $\mathcal{C}_{ij} := \mathbb{E}\big[(x_i - m_i) \otimes (x_j - m_j)\big]$, or equivalently,*

$$\langle f, \mathcal{C}_{ij} g \rangle_{\mathcal{H}} = \mathbb{E}\big[\, \langle f, (x_i - m_i) \rangle_{\mathcal{H}} \, \langle g, (x_j - m_j) \rangle_{\mathcal{H}} \,\big], \quad \forall f, g \in \mathcal{H}. \tag{26}$$

Then, the conditional distribution of x_1 given x_2 is Gaussian with mean

$$m' = m_1 + \mathcal{C}_{12} \mathcal{C}_{22}^{-1} (x_2 - m_2) \in \mathcal{H} \tag{27}$$

and covariance

$$\mathcal{C}' = \mathcal{C}_{11} - \mathcal{C}_{12} \mathcal{C}_{22}^{-1} \mathcal{C}_{21} \in \mathcal{L}(\mathcal{H}, \mathcal{H}). \tag{28}$$

Theorem 1. *Let $\mathcal{C}$ be a positive-definite, self-adjoint, trace-class operator on a separable Hilbert space $\mathcal{H}$. Then, for any choice of α_t and β_t satisfying the usual conditions for stochastic interpolants, RWMH waltz for the mirror interpolant $p_0 = p_1 = \mathcal{N}(0, \mathcal{C})$ is identical to pCN.*

Proof. Given $x_0 \in \mathcal{H}$ and some $\gamma \in (0, 1)$, we recall that the pCN proposal with respect to a Gaussian measure $\mathcal{N}(0, \mathcal{C})$ is given by (Cotter et al., 2013):

$$x_0' \mid x_0 \sim \mathcal{N}\Big(\sqrt{1 - \gamma^2}\, x_0,\ \gamma^2 \mathcal{C}\Big). \tag{29}$$

We claim that the diffusion waltz proposal with $p_0 = p_1 = \mathcal{N}(0, \mathcal{C})$ reduces to this form, with a one-to-one relation between the noising time $t \in (0, 1)$ and the pCN proposal step size γ.

To this end, for a Gaussian random variable $x_t \in \mathcal{H}$ given by $x_t = \alpha_t x_0' + \beta_t \varepsilon'$ for some $x_0' \sim \mathcal{N}(0, \mathcal{C})$ and $\varepsilon' \sim \mathcal{N}(0, \mathcal{C})$ with $x_0' \perp \varepsilon$, we can deduce that

$$x_0' \sim \mathcal{N}(0, \mathcal{C}), \quad x_t \sim \mathcal{N}\big(0,\ (\alpha_t^2 + \beta_t^2)\mathcal{C}\big), \tag{30}$$

and the cross-covariances between them are given by

$$\mathbb{E}[x_0' \otimes x_t] = \mathbb{E}[x_t \otimes x_0'] = \alpha_t \mathcal{C}. \tag{31}$$

Then, by Lemma 1, we can compute

$$x_0' \mid x_t \sim \mathcal{N}\big(m_{0|t}', \mathcal{C}_{0|t}'\big), \tag{32}$$

where

$$m_{0|t}' := \frac{\alpha_t}{\alpha_t^2 + \beta_t^2} x_t, \qquad \mathcal{C}_{0|t}' := \frac{\beta_t^2}{\alpha_t^2 + \beta_t^2} \mathcal{C}. \tag{33}$$

Thus, we can equivalently express the random variable x_0' as

$$x_0' = \frac{\alpha_t}{\alpha_t^2 + \beta_t^2} x_t + \sqrt{\frac{\beta_t^2}{\alpha_t^2 + \beta_t^2}}\, \eta, \tag{34}$$

for some $\eta \sim \mathcal{N}(0, \mathcal{C})$, which gives us the explicit form for the second step of diffusion waltz, under this special case. As a sanity check, we can verify that marginalizing x_0', given by equation 34, over $x_t \sim \mathcal{N}\big(0,\ (\alpha_t^2 + \beta_t^2)\mathcal{C}\big)$ gives us $x_0' \sim \mathcal{N}(0, \mathcal{C})$ as expected. Now, combining with the first step of diffusion waltz, written as

$$x_t = \alpha_t x_0 + \beta_t \varepsilon, \tag{35}$$

for $x_0 \sim \mathcal{N}(0, \mathcal{C})$ and $\varepsilon \sim \mathcal{N}(0, \mathcal{C})$ with $x_0 \perp \varepsilon$, $x_0 \perp \eta$ and $\varepsilon \perp \eta$, we can deduce that

$$\mathbb{E}[x_0 \otimes x_0'] = \mathbb{E}[x_0' \otimes x_0] = \frac{\alpha_t^2}{\alpha_t^2 + \beta_t^2} \mathcal{C}. \tag{36}$$

Applying Lemma 1 again, this time to the joint distribution between x_0 and x_0', with the intermediate variable x_t marginalized out, we find

$$x_0' \mid x_0 \sim \mathcal{N}\big(m_{0|0}', \mathcal{C}_{0|0}'\big), \tag{37}$$

where

$$m_{0|0}' := \frac{\alpha_t^2}{\alpha_t^2 + \beta_t^2} x_0, \qquad \mathcal{C}_{0|0}' := \frac{\beta_t^2(2\alpha_t^2 + \beta_t^2)}{(\alpha_t^2 + \beta_t^2)^2}\mathcal{C}. \tag{38}$$

Now, setting

$$\gamma := \sqrt{\frac{\beta_t^2(2\alpha_t^2 + \beta_t^2)}{(\alpha_t^2 + \beta_t^2)^2}}, \tag{39}$$

we have

$$1 - \gamma^2 = 1 - \frac{\beta_t^2(2\alpha_t^2 + \beta_t^2)}{(\alpha_t^2 + \beta_t^2)^2} \tag{40}$$

$$= \frac{(\alpha_t^4 + \cancel{2\alpha_t^2\beta_t^2} + \beta_t^4) - (\cancel{2\alpha_t^2\beta_t^2} + \beta_t^4)}{(\alpha_t^2 + \beta_t^2)^2} \tag{41}$$

$$= \frac{\alpha_t^4}{(\alpha_t^2 + \beta_t^2)^2}. \tag{42}$$

This implies that $m_{0|0}' = \sqrt{1-\gamma^2}x_0$ and $\mathcal{C}_{0|0}' = \gamma^2\,\mathcal{C}$, and comparing with equation 29, we conclude that the diffusion waltz proposal (equation 37), is equivalent to pCN. □

B.2 Stability of RWMH waltz

In this section, we prove the stability of RWMH diffusion waltz to various approximations that are invoked in practice (neural network approximations, SDE discretisation, etc...). For simplicity, we only consider chains on finite-dimensional Euclidean space $\mathbb{R}^d$ in this section.

Denote by $p_0(x_0)$ our true data measure and $p_\theta(x_0)$ the neural network approximation defined via a generative model. Given a forward noising kernel $p(x_t \mid x_0) = \mathcal{N}(x_t \mid \alpha_t x_0, \beta_t^2 I)$, we have:

- A true denoiser $p(x_0 \mid x_t) \propto p(x_t \mid x_0)p_0(x_0)$, and
- An approximate denoiser $q_\theta(x_0 \mid x_t) \approx p_\theta(x_0 \mid x_t) \propto p(x_t \mid x_0)p_\theta(x_0)$,

where the approximation in the latter may be due to discretisation of the backward SDE, or approximation by a stochastic flow map (Holderrieth et al., 2026; Potaptchik et al., 2026). Our true and approximate proposal kernels are given by

$$Q(x_0, x_0') := \int_{\mathbb{R}^d} p(x_0' \mid x_t)p(x_t \mid x_0)\mathrm{d}x_t, \quad \text{and} \tag{43}$$

$$Q_\theta(x_0, x_0') := \int_{\mathbb{R}^d} q_\theta(x_0' \mid x_t)p(x_t \mid x_0)\mathrm{d}x_t, \tag{44}$$

respectively, and the corresponding Metropolis-Hastings kernel reads

$$K(x_0, x_0') := \alpha(x_0, x_0')Q(x_0, x_0') + \left(1 - \int_{\mathbb{R}^d} \alpha(x_0, z)Q(x_0, z)\mathrm{d}z\right)\delta(x_0 - x_0') \tag{45}$$

$$K_\theta(x_0, x_0') := \alpha(x_0, x_0')Q_\theta(x_0, x_0') + \left(1 - \int_{\mathbb{R}^d} \alpha(x_0, z)Q_\theta(x_0, z)\mathrm{d}z\right)\delta(x_0 - x_0'), \tag{46}$$

where $\alpha(x_0, x_0')$ is the acceptance probability. Finally, define the Markov operators

$$\mathcal{P}\nu := \int_{\mathbb{R}^d} K(x, \cdot)\nu(\mathrm{d}x), \quad \mathcal{P}_\theta\nu := \int_{\mathbb{R}^d} K_\theta(x, \cdot)\nu(\mathrm{d}x) \tag{47}$$

and for any $n \in \mathbb{N}$,

$$P^n_{\text{true}} := \underbrace{(\mathcal{P} \circ \cdots \circ \mathcal{P})}_{n \text{ times}} p_0, \quad P^n_\theta := \underbrace{(\mathcal{P}_\theta \circ \cdots \circ \mathcal{P}_\theta)}_{n \text{ times}} p_\theta, \tag{48}$$

are the marginal laws of the MCMC chain at step n using the exact and approximate RWMH diffusion waltz, respectively. We are interested in investigating how much P^n_θ deviates from P^n under the approximations $p_\theta(x_0) \approx p_0(x_0)$ and $q_\theta(x_0 \mid x_t) \approx p(x_0 \mid x_t)$.

To make this precise, we introduce some notions of distance on the space of probability densities.

Definition 1 (Total variation distance)**.** *Given a measurable space $(\Omega, \mathcal{F})$ and two probability densities p, q, we define the total variation (TV) distance by*

$$\|p - q\|_{\text{tv}} := \frac{1}{2} \int_\Omega \big|p(x) - q(x)\big| \mathrm{d}x. \tag{49}$$

Definition 2 (V-weighted TV norm)**.** *Given a measurable space $(\Omega, \mathcal{F})$ and a measurable function $V : \Omega \to [1, \infty]$, we define the V-norm by*

$$\|p - q\|_V := \sup_{|f| \le V} \left\{ \left| \int_\Omega f(x)(p(x) - q(x)) \mathrm{d}x \right| \right\}. \tag{50}$$

We prove the following stability result:

Theorem (Rigorous restatement of Theorem 2)**.** *Let $V(x) := 1 + \|x\|^2$. For $t \in (0, 1)$, assume*

(A1) (Assumptions on prior and likelihood) $p_0(x_0), p_\theta(x_0)$ are continuous, has full-support on $\mathbb{R}^d$ with strictly positive densities and bounded moments. $p(y \mid x_0)$ is positive everywhere and continuous in x_0.

(A2) (Linear growth of the mean) There exists $A_t, B_t \ge 0$ satisfying $A_t \alpha_t < 1$ such that

$$\big\|\mathbb{E}[x_0 \mid x_t]\big\| \le A_t \|x_t\| + B_t, \tag{51}$$

where $\mathbb{E}[x_0 \mid x_t]$ is the mean of the true denoising kernel $p(x_0 \mid x_t)$.

(A3) (Covariance bound) There exists $C_t > 0$ such that

$$\mathrm{Cov}[x_0 \mid x_t] \preceq C_t I, \tag{52}$$

where $\mathrm{Cov}[x_0 \mid x_t]$ is the covariance of the true denoising kernel $p(x_0 \mid x_t)$.

(A4) (Fourth-moment bounds) We have

$$\sup_{x_0 \in \mathbb{R}^d} \big\{ \mathbb{E}_\bullet \big[V(x_0')^2 \mid x_0 \big] / V(x_0)^2 \big\} < \infty, \quad \text{for} \quad \bullet \in \{Q, Q_\theta\}, \tag{53}$$

where $\mathbb{E}_Q[x_0' \mid x_0]$ and $\mathbb{E}_{Q_\theta}[x_0' \mid x_0]$ denote the conditional expectation with respect to the true and approximate proposals $Q(x_0, x_0'), Q_\theta(x_0, x_0')$, respectively.

(A5) (Control on the rejection rate) Defining the average acceptance rate $\bar{\alpha}(x_0) := \mathbb{E}_Q[\alpha(x_0, \cdot)]$ of the true proposal $Q(x_0, \cdot)$, there exists $R^* > 0$ and $0 \le r < 1 - (A_t \alpha_t)^2$ such that $1 - \bar{\alpha}(x_0) \le r$ whenever $|x_0| \ge R^*$. That is, the average rejection rate starting at any point x_0 beyond a ball of radius R^* is at most r.

Under these assumptions, there exist $\gamma_0 > 0$ and positive constants $C, \kappa, \gamma < \infty$ and $\rho < 1$, such that, whenever $\gamma < \gamma_0$, the bound

$$\|P^n_{\text{true}} - P^n_\theta\|_V \le C \left(\rho^n \|p_0(\cdot) - p_\theta(\cdot)\|_V + (1 - \rho^n) \frac{\gamma \kappa}{1 - \rho} \right), \tag{54}$$

holds for any $n \in \mathbb{N}$, where $\gamma > 0$ is further controlled by

$$\gamma \lesssim \sup_{x_0 \in \mathbb{R}^d} \left\{ \sqrt{E(x_0)} + E(x_0) \right\}, \tag{55}$$

where

$$E(x_0) := \int_{\mathbb{R}^d} p(x_t \mid x_0) \big\| p(\cdot \mid x_t) - q_\theta(\cdot \mid x_t) \big\|_{\text{tv}} \mathrm{d}x_t \tag{56}$$

To prove this statement, we use the following results

Definition 3. *A Markov chain $\{x^{(n)}\}_{n\in\mathbb{N}}$ is said to be geometrically ergodic if there exists a function $V:\mathbb{R}^d\to[1,\infty]$, constants $C>0$, $0<\rho<1$ and a measure π such that*

$$\|\mathrm{Law}(x^{(n)})-\pi\|_V \le CV(x)\rho^n. \tag{57}$$

The result below is a well-known characterisation of geometric ergodicity.

Lemma 2 (Theorem 1.4 in Mengersen & Tweedie (1996))**.** *An irreducible, aperiodic Markov chain $\{x^{(n)}\}_{n\in\mathbb{N}}$ with Markov operator $\mathcal{P}$ is geometrically ergodic if and only if it satisfies the "geometric drift condition": there exists a function $V:\mathbb{R}^d\to[1,\infty]$, a small set S, and positive constants $\lambda_S<1, b_S<\infty$ such that*

$$(\mathcal{P}V)(x)\le\lambda_S V(x)+b_S\mathbb{1}_S(x), \tag{58}$$

holds for all $x\in\mathbb{R}^d$, where $(\mathcal{P}V)(x):=\int_{\mathbb{R}^d}V(x')\mathcal{P}\delta_x(\mathrm{d}x')$.

We refer the readers to Mengersen & Tweedie (1996) for the proof and precise definition of irreduciblity, aperiodicity and small sets. The result below shows that the RWMH waltz Markov chains are indeed irreducible and aperiodic, and moreover contains all compact sets as small sets.

Proposition 3. *Under assumption (A1), the Markov chain induced by the exact RWMH diffusion waltz is irreducible and aperiodic. Moreover, every compact set S with positive Lebesgue measure is a small set of the chain.*

Proof. By Bayes' rule, we can rewrite the RWMH waltz proposal kernel as

$$Q(x_0,x_0')=\int_{\mathbb{R}^d}p(x_0'\mid x_t)p(x_t\mid x_0)\mathrm{d}x_t=p_0(x_0')\int_{\mathbb{R}^d}\frac{p(x_t\mid x_0')p(x_t\mid x_0)}{p_t(x_t)}\mathrm{d}x_t. \tag{59}$$

By our assumption that $p_0(x_0)$ has full support and since both $p(x_t\mid x_0)$ and $p(x_t\mid x_0')$ are Gaussians, we have that $Q(x_0,x_0')$ is positive for all $x_0,x_0'\in\mathbb{R}^d$. Moreover, we can use the continuity of $p_0, p(\cdot\mid x_0)$ and the dominated convergence theorem using a standard Gaussian tail bound on the integrand to show continuity of Q in x_0,x_0'. Then by Lemmas 1.1 and 1.2 in Mengersen & Tweedie (1996), the RWMH waltz Markov chain is irreducible, aperiodic, and contains all compact sets with positive Lebesgue measure as small sets. □

Next, we establish that under our assumptions in the main theorem, the RWMH waltz Markov chain is geometrically ergodic. First, we prove the following technical lemma.

Lemma 3. *Assume (A1) and (A5). For $V(x):=1+\|x\|^2$, let $\mathbb{E}[V(x_0')\mid x_0]\le\delta V(x_0)+C$ for some positive constants $\delta<1$ and $C>0$ with $\delta+r<1$, where r is the constant appearing in (A5) and the conditional expectation is defined with respect to the transition kernel $Q(x_0,x_0')$ of the RWMH waltz Markov chain. Then, said Markov chain is geometrically ergodic.*

Proof. By equation 45 and equation 47, we have

$$\begin{aligned}(\mathcal{P}V)(x_0)&:=\int_{\mathbb{R}^d}V(x_0')\mathcal{P}\delta_{x_0}(\mathrm{d}x_0')=\int_{\mathbb{R}^d}V(x_0')K(x_0,x_0')\mathrm{d}x_0' &(60)\\ &=\int_{\mathbb{R}^d}V(x_0')\Big(\alpha(x_0,x_0')Q(x_0,x_0')+\big(1-\bar\alpha(x_0)\big)\delta(x_0-x_0')\Big)\mathrm{d}x_0' &(61)\\ &\le\mathbb{E}[V(x_0')\mid x_0]+V(x_0)\big(1-\bar\alpha(x_0)\big) &(62)\\ &\le\big(\delta+(1-\bar\alpha(x_0))\big)V(x_0)+C, &(63)\end{aligned}$$

where we used the fact that $\alpha(x_0,x_0')\le 1$ for all x_0,x_0' in the third line, since it is a probability. Now, for arbitrary $\eta>0$ such that $\delta+r+\eta<1$, choose $R\ge R^*$ such that $C\le\eta(1+R^2)$, where the constants r, R^* come from Assumption (A5). Then, given a compact set $S:=\{x_0\in\mathbb{R}^d:\|x_0\|\le R\}$, we have that $C\le\eta V(x_0)$ on $x_0\in\mathbb{R}^d\backslash S$ and therefore by Assumption (A5), we have

$$(\mathcal{P}V)(x_0)\le\begin{cases}(\delta+1)(1+R^2)+C, & \text{on}\quad x_0\in S,\\ (\delta+r+\eta)V(x_0), & \text{on}\quad x_0\in\mathbb{R}^d\backslash S.\end{cases} \tag{64}$$

Thus, setting $\lambda_S := \delta + \eta + r < 1$ and $b_S := (\delta + 1)(1 + R^2) + C < \infty$, we see that $\mathcal{P}$ satisfies the geometric drift condition (58) and moreover, by Proposition 3, the chain is irreducible, aperiodic and contains S as a small set. Hence, by Lemma 2, the chain is geometrically ergodic. □

The following result establishes the geometric ergodicity of RWMH waltz under our assumptions.

Proposition 4 (Geometric ergodicity of RWMH waltz)**.** *Under assumptions (A1)–(A5), the RWMH waltz Markov chain is geometrically ergodic with* $V(x) = 1 + \|x\|^2$.

Proof. By the law of total variance and using that $x_t \mid x_0 \sim \mathcal{N}\big(\alpha_t x_0, \beta_t^2 I\big)$, we have

$$\mathbb{E}\big[\|x_t\|^2 \mid x_0\big] = \big\|\mathbb{E}[x_t \mid x_0]\big\|^2 + \mathrm{tr}\big(\mathrm{Cov}[x_t \mid x_0]\big) \tag{65}$$

$$= \alpha_t^2 \|x_0\|^2 + d\beta_t^2. \tag{66}$$

Thus, from $V(x) = 1 + \|x\|^2$, we have

$$\mathbb{E}\big[V(x_t) \mid x_0\big] = \alpha_t^2 V(x_0) + (1 - \alpha_t^2) + d\beta_t^2. \tag{67}$$

Note that $1 - \alpha_t^2 > 0$ since $\alpha_t \in (0, 1)$. Similarly, we have

$$\mathbb{E}\big[\|x_0'\|^2 \mid x_t\big] = \big\|\mathbb{E}[x_0' \mid x_t]\big\|^2 + \mathrm{tr}\big(\mathrm{Cov}[x_0' \mid x_t]\big) \tag{68}$$

$$\overset{(A2,3)}{\leq} \big(A_t \|x_t\| + B_t\big)^2 + dC_t \tag{69}$$

$$= A_t^2 \|x_t\|^2 + 2A_t B_t \|x_t\| + B_t^2 + dC_t. \tag{70}$$

Using the Peter-Paul inequality $ab \leq \frac{\varepsilon a^2}{2} + \frac{b^2}{2\varepsilon}$ for arbitrary $\varepsilon > 0$, we have

$$(70) \leq A_t^2 \|x_t\|^2 + \left(\varepsilon A_t^2 \|x_t\|^2 + \frac{B_t^2}{\varepsilon}\right) + B_t^2 + dC_t \tag{71}$$

$$\leq (1 + \varepsilon) A_t^2 \|x_t\|^2 + \big(dC_t + (1 + \varepsilon^{-1}) B_t^2\big). \tag{72}$$

Therefore we get

$$\mathbb{E}\big[V(x_0') \mid x_t\big] \leq (1 + \varepsilon) A_t^2 V(x_t) + C_t', \tag{73}$$

where $C_t' := dC_t + (1 + \varepsilon^{-1}) B_t^2 + (1 - (1 + \varepsilon) A_t^2)$, which is positive for small enough ε. Now, by the law of conditional expectation, we find that

$$\mathbb{E}\big[V(x_0') \mid x_0\big] = \mathbb{E}\big[\mathbb{E}\big[V(x_0') \mid x_t\big] \mid x_0\big] \tag{74}$$

$$\overset{(73)}{\leq} (1 + \varepsilon) A_t^2 \mathbb{E}\big[V(x_t) \mid x_0\big] + C_t' \tag{75}$$

$$\overset{(67)}{=} \underbrace{(1 + \varepsilon)(A_t \alpha_t)^2}_{=:\delta} V(x_0) + C_t'', \tag{76}$$

where $C_t'' := C_t' + (1 + \varepsilon) A_t^2 \big((1 - \alpha_t^2) + d\beta_t^2\big) > 0$. By assumption (A2), we have $A_t \alpha_t < 1$ so we can set ε arbitrarily small such that $\delta < 1$. Geometric ergodicity of the chain then follows from Lemma 3. □

The following result, paraphrased from (Rudolf & Schweizer, 2018, Corollary 3.4), establishes the stability of a geometrically ergodic Markov chain with respect to perturbations.

Lemma 4 (Corollary 3.4 in Rudolf & Schweizer (2018))**.** *Let* $\{x^{(n)}\}$ *be a Metropolis-Hastings Markov chain with Markov operator* $\mathcal{P}\nu := \int_{\mathbb{R}^d} K(x, \cdot)\nu(\mathrm{d}x)$ *that is geometric ergodic with constants* $\rho < 1$, $C > 0$ *and measurable function* $V : \mathbb{R}^d \to [1, \infty]$. *For a given measure* p_0 *define* $P^n := \mathcal{P}^{(n)} p_0$. *Likewise, for another measure* $\tilde{p}_0$, *possibly different from* p_0, *and a "perturbed" operator* $\widetilde{\mathcal{P}}\nu := \int_{\mathbb{R}^d} \widetilde{K}(x, \cdot)\nu(\mathrm{d}x)$, *define* $\widetilde{P}^n := \widetilde{\mathcal{P}}^{(n)} \tilde{p}_0$. *Then, letting*

$$\gamma := \sup_{x \in \mathbb{R}^d} \frac{\|K(x, \cdot) - \widetilde{K}(x, \cdot)\|_V}{V(x)} \quad \textit{and} \quad \kappa := \max\Big\{\tilde{p}_0(V), \frac{b_S}{1 - \lambda_S - \gamma}\Big\}, \tag{77}$$

where $\tilde{p}_0(V) := \int_{\mathbb{R}^d} V(x) \tilde{p}_0(x) \mathrm{d}x$ *and* b_S, λ_S *are coefficeints of the geometric drift condition (58), and assuming that* $\lambda_S + \gamma < 1$, *we have the perturbation bound*

$$\|P^n - \widetilde{P}^n\|_V \leq C \left(\rho^n \|p_0(\cdot) - \tilde{p}_0(\cdot)\|_V + (1 - \rho^n) \frac{\gamma \kappa}{1 - \rho}\right). \tag{78}$$

We are now ready to prove our main theorem.

Proof of Theorem 2. First we note that the Markov chain generated by RWMH waltz under assumptions (A1)–(A5) is geometric ergodic with $V(x) = 1 + \|x\|^2$, by Proposition 4. Thus, by Lemma 4, we have that the perturbation bound (54) holds. Next, we relate γ to approximation errors in the denoiser with respect to the total variation distance. By (45)–(46), we have

$$\|K(x_0,\cdot) - K_\theta(x_0,\cdot)\|_V \tag{79}$$

$$\leq \Big\|\alpha(x_0,\cdot)\big(Q(x_0,\cdot) - Q_\theta(x_0,\cdot)\big)\Big\|_V + V(x_0)\left|\int_{\mathbb{R}^d} \alpha(x_0,z)\Big(Q(x_0,z) - Q_\theta(x_0,z)\Big)\mathrm{d}z\right|. \tag{80}$$

For the first term, using the definition of the V-norm (Definition 2) and using the fact that $\alpha(x_0,\cdot) < 1$, we have

$$\Big\|\alpha(x_0,\cdot)\big(Q(x_0,\cdot) - Q_\theta(x_0,\cdot)\big)\Big\|_V \leq \left|\int_{\mathbb{R}^d} V(z)\big|Q(x_0,z) - Q_\theta(x_0,z)\big|\mathrm{d}z\right| \tag{81}$$

$$\leq \left(\int_{\mathbb{R}^d} V(z)^2\big|Q(x_0,z) - Q_\theta(x_0,z)\big|\mathrm{d}z\right)^{\frac{1}{2}} \left(\int_{\mathbb{R}^d} \big|Q(x_0,z) - Q_\theta(x_0,z)\big|\mathrm{d}z\right)^{\frac{1}{2}} \tag{82}$$

$$\leq \underbrace{\sqrt{\mathbb{E}_Q[V(\cdot)^2 \mid x_0] + \mathbb{E}_{Q_\theta}[V(\cdot)^2 \mid x_0]}}_{=:C(x_0)} \Big(2\big\|Q(x_0,\cdot) - Q_\theta(x_0,\cdot)\big\|_{\mathrm{tv}}\Big)^{\frac{1}{2}}, \tag{83}$$

where we used Hölder's inequality in the second line. By assumption (A4), we have that $C(x_0)/V(x_0)$ is uniformly bounded. Similarly, we can bound the second term by

$$\left|\int_{\mathbb{R}^d} \alpha(x_0,z)\Big(Q(x_0,z) - Q_\theta(x_0,z)\Big)\mathrm{d}z\right| \leq \int_{\mathbb{R}^d} \Big|Q(x_0,z) - Q_\theta(x_0,z)\Big|\mathrm{d}z \tag{84}$$

$$= 2\big\|Q(x_0,\cdot) - Q_\theta(x_0,\cdot)\big\|_{\mathrm{tv}}. \tag{85}$$

Now, by the expressions (43)–(44) of the proposals Q and Q_θ, we have

$$\big\|Q(x_0,\cdot) - Q_\theta(x_0,\cdot)\big\|_{\mathrm{tv}} = \frac{1}{2}\int_{\mathbb{R}^d} \big|Q(x_0,x_0') - Q_\theta(x_0,x_0')\big|\mathrm{d}x_0' \tag{86}$$

$$= \frac{1}{2}\int_{\mathbb{R}^d} \left|\int_{\mathbb{R}^d} p(x_t \mid x_0)\Big(p(x_0' \mid x_t) - q_\theta(x_0' \mid x_t)\Big)\mathrm{d}x_t\right| \mathrm{d}x_0' \tag{87}$$

$$\leq \int_{\mathbb{R}^d} p(x_t \mid x_0)\left(\frac{1}{2}\int_{\mathbb{R}^d} \big|p(x_0' \mid x_t) - q_\theta(x_0' \mid x_t)\big|\,\mathrm{d}x_0'\right)\mathrm{d}x_t \tag{88}$$

$$= \int_{\mathbb{R}^d} p(x_t \mid x_0)\big\|p(\cdot \mid x_t) - q_\theta(\cdot \mid x_t)\big\|_{\mathrm{tv}}\mathrm{d}x_t \tag{89}$$

$$=: E(x_0). \tag{90}$$

Putting this together, we establish the bound

$$\gamma = \sup_{x_0 \in \mathbb{R}^d}\left\{\frac{\|K(x_0,\cdot) - K_\theta(x_0,\cdot)\|_V}{V(x_0)}\right\} \lesssim \sup_{x_0 \in \mathbb{R}^d}\left\{\sqrt{E(x_0)} + E(x_0)\right\}. \tag{91}$$

Now set $\gamma_0 := 1 - \lambda_S$. By assumption (A1), we have $p_\theta(V) := \int_{\mathbb{R}^d} V(x)p_\theta(x)\mathrm{d}x < \infty$, and since $\gamma < \gamma_0$, we have that $\lambda_S + \gamma < 1$, which implies $\kappa := \max\left\{p_\theta(V), \frac{b_S}{1-\lambda_S-\gamma}\right\} < \infty$. This completes the proof. □

B.3 Acceptance probability computations

Given a proposal kernel $Q(x_0, x_0')$, the Metropolis-Hastings accept/reject probability to sample from the posterior $p(x_0' \mid y)$ is given by

$$\alpha = \min\left\{1, \frac{p(x_0' \mid y)Q(x_0', x_0)}{p(x_0 \mid y)Q(x_0, x_0')}\right\}. \tag{92}$$

In this appendix, we derive the Metropolis/Hastings acceptance probabilities for diffusion waltz as it appears in Algorithms 1 and 2.

B.3.1 RWMH DIFFUSION WALTZ

We first show that for RWMH diffusion waltz, as described in Algorithm 1, this acceptance probability reduces to a simple likelihood ratio, avoiding the need to evaluate prior densities. To show this, we use the following result.

Lemma 5. *The RWMH diffusion waltz proposal kernel $Q(x_0, x_0')$ is reversible with respect to the prior $p_0(x_0)$, i.e.,*

$$p_0(x_0)Q(x_0, x_0') = p_0(x_0')Q(x_0', x_0). \tag{93}$$

Proof. The RWMH diffusion waltz proposal kernel (steps 1&2 of Algorithm 1) can be expressed as

$$Q(x_0, x_0') := \int p(x_0' \mid x_t)p(x_t \mid x_0)\mathrm{d}x_t. \tag{94}$$

Thus, we have

$$\begin{aligned}
p_0(x_0)Q(x_0, x_0') &= \int p(x_0' \mid x_t)p(x_t \mid x_0)p(x_0)\mathrm{d}x_t && (95)\\
&= \int \frac{p(x_t \mid x_0')p(x_0')}{p_t(x_t)}p(x_t \mid x_0)p_0(x_0)\mathrm{d}x_t && (96)\\
&= \int p(x_t \mid x_0')p_0(x_0')\frac{p(x_t \mid x_0)p_0(x_0)}{p_t(x_t)}\mathrm{d}x_t && (97)\\
&= \int p(x_0 \mid x_t)p(x_t \mid x_0')p_0(x_0')\mathrm{d}x_t && (98)\\
&= p_0(x_0')Q(x_0', x_0), && (99)
\end{aligned}$$

which establishes the reversibility of the diffusion waltz proposal kernel with respect to p_0. □

From this, we can simplify the ratio in equation 92 as

$$\frac{p(x_0' \mid y)Q(x_0', x_0)}{p(x_0 \mid y)Q(x_0, x_0')} = \frac{p(y \mid x_0')p_0(x_0')Q(x_0', x_0)}{p(y \mid x_0)p_0(x_0)Q(x_0, x_0')} = \frac{p(y \mid x_0')}{p(y \mid x_0)}, \tag{100}$$

proving our claim, where the first equality follows from Bayes' rule and the second follows from the reversibility property (93).

B.3.2 CONDITIONAL NOISING WALTZ

The conditional waltz kernel is given in full by:

$$K(x_0, x_0') = P_{\text{accept}}(x_0, x_0') + P_{\text{reject}}(x_0, x_0')\delta(x_0 - x_0'). \tag{101}$$

where

$$P_{\text{accept}}(x_0, x_0') := \int p(x_0' \mid x_t)q(x_t \mid x_0, y)\min\left\{1, \frac{p(y \mid x_0')\,q(x_t \mid x_0', y)\,p(x_t \mid x_0)}{p(y \mid x_0)\,q(x_t \mid x_0, y)\,p(x_t \mid x_0')}\right\}\mathrm{d}x_t, \tag{102}$$

$$P_{\text{reject}}(x_0, x_0') := 1 - P_{\text{accept}}(x_0, x_0'). \tag{103}$$

We show that this kernel is reversible with respect to the posterior $p_\theta(x_0 \mid y)$, that is,

$$p(x_0 \mid y)K(x_0, x_0') = p(x_0' \mid y)K(x_0', x_0). \tag{104}$$

To show this, we use the following lemma

Lemma 6. *The following identity holds for any (x_0, x_0', x_t):*

$$\begin{aligned}
&\left(p(x_0' \mid x_t)q(x_t \mid x_0, y)\min\left\{1, \frac{p(y \mid x_0')\,q(x_t \mid x_0', y)\,p(x_t \mid x_0)}{p(y \mid x_0)\,q(x_t \mid x_0, y)\,p(x_t \mid x_0')}\right\}\right)p(x_0 \mid y)\\
&\quad= \left(p(x_0 \mid x_t)q(x_t \mid x_0', y)\min\left\{1, \frac{p(y \mid x_0)\,q(x_t \mid x_0, y)\,p(x_t \mid x_0')}{p(y \mid x_0')\,q(x_t \mid x_0', y)\,p(x_t \mid x_0)}\right\}\right)p(x_0' \mid y).
\end{aligned} \tag{105}$$

Proof. For simplicity, let

$$r := \frac{p(y \mid x_0')\, q(x_t \mid x_0', y)\, p(x_t \mid x_0)}{p(y \mid x_0)\, q(x_t \mid x_0, y)\, p(x_t \mid x_0')}. \tag{106}$$

When $r < 1$, the left hand side and right hand side of equation 105 reads, respectively,

$$\text{LHS} = p(x_0' \mid x_t) q(x_t \mid x_0, y) p(x_0 \mid y)\, r \tag{107}$$

$$\text{RHS} = p(x_0 \mid x_t) q(x_t \mid x_0', y) p(x_0' \mid y). \tag{108}$$

Thus, for equality (105) to hold, we need

$$r = \frac{p(x_0 \mid x_t) q(x_t \mid x_0', y) p(x_0' \mid y)}{p(x_0' \mid x_t) q(x_t \mid x_0, y) p(x_0 \mid y)}. \tag{109}$$

Now, we have

$$p(x_0 \mid x_t) = \frac{p(x_t \mid x_0) p_0(x_0)}{p_t(x_t)}, \qquad p(x_0' \mid y) = \frac{p(y \mid x_0') p_0(x_0')}{p(y)}, \tag{110}$$

by Bayes' rule and substituting this into equation 109, we get

$$(109) = \frac{p(x_t \mid x_0) p_0(x_0) \cancel{p_t(x_t)}}{p(x_t \mid x_0') p_0(x_0') \cancel{p_t(x_t)}} \times \frac{q(x_t \mid x_0', y)}{q(x_t \mid x_0, y)} \times \frac{p(y \mid x_0') p_0(x_0') \cancel{p(y)}}{p(y \mid x_0) p_0(x_0) \cancel{p(y)}} \tag{111}$$

$$= \frac{p(x_t \mid x_0) q(x_t \mid x_0', y) p(y \mid x_0') \cancel{p_0(x_0) p_0(x_0')}}{p(x_t \mid x_0') q(x_t \mid x_0, y) p(y \mid x_0) \cancel{p_0(x_0') p_0(x_0)}}, \tag{112}$$

which agrees with the expression in equation 106, and therefore the equality (105) holds. By symmetry, (105) also holds in the case $r \geq 1$. □

By Lemma 6, since identity (105) holds pointwise in x_t, it also holds when we integrate this out, which gives us

$$p(x_0 \mid y) P_{\text{accept}}(x_0, x_0') = p(x_0' \mid y) P_{\text{accept}}(x_0', x_0). \tag{113}$$

Trivially, we also have

$$\delta(x_0 - x_0') p_\theta(x_0 \mid y) P_{\text{reject}}(x_0, x_0') = \delta(x_0' - x_0) p_\theta(x_0' \mid y) P_{\text{reject}}(x_0', x_0), \tag{114}$$

thus combining this, we have that the conditional waltz kernel $K(x_0, \cdot)$ is reversible with respect to $p(x_0 \mid y)$, i.e., equation 104 holds.

Remark 3. *As a sanity check, the RWMH waltz acceptance probability (100) follows from the conditional noising waltz acceptance probability (106), by setting* $q(x_t \mid x_0, y) = p(x_t \mid x_0)$.

C Statistical linearisation and ensemble Kalman update rules

Here, we provide details on the statistical linearisation procedure to derive the ensemble Kalman conditional noising (EKCN) kernel. The essence of statistical linearisation is to seek a linear approximation to a nonlinear function $\mathcal{A}(x) \approx Ax + b$ that is a best fit in terms of least squares (Zheng et al., 2025a), i.e., given an ensemble $\{x^{(j)}\}_{j=1}^J$

$$(A, b) = \arg\min_{A,b} \left\{ \frac{1}{J} \sum_{j=1}^J \left\| \mathcal{A}(x^{(j)}) - \left(Ax^{(j)} + b\right) \right\|^2 \right\} \tag{115}$$

This gives us

$$A := (C^{xy})^\top (C^{xx})^{-1}, \qquad b := \overline{\mathcal{A}(x)} - A\bar{x}, \tag{116}$$

where

$$\bar{x} := \frac{1}{J} \sum_{j=1}^J x^{(j)}, \qquad \overline{\mathcal{A}(x)} := \frac{1}{J} \sum_{j=1}^J \mathcal{A}(x^{(j)}) \tag{117}$$

$$C^{xx} := \frac{1}{J} \sum_{i,j=1}^J \left(x^{(i)} - \bar{x}\right)\left(x^{(j)} - \bar{x}\right)^\top \qquad C^{xy} := \frac{1}{J} \sum_{i,j=1}^J \left(x^{(i)} - \bar{x}\right)\left(\mathcal{A}(x^{(j)}) - \overline{\mathcal{A}(x)}\right)^\top. \tag{118}$$

A consequence of this is that we can approximate the Jacobian of $\mathcal{A}$ as $D\mathcal{A}(x) \approx (C^{xy})^\top (C^{xx})^{-1}$, which can be computed without explicitly taking derivatives. Thus, we can use this to solve optimisation problems in a derivative-free manner. For example, if we consider the objective

$$U(x) = \frac{1}{2}\|y - \mathcal{A}(x)\|_\Gamma^2, \tag{119}$$

where $\|x\|_A := x^\top A^{-1} x$, its gradient can be approximated as

$$\nabla_x U(x) = D^\top \mathcal{A}(x) \Gamma^{-1}\big(\mathcal{A}(x) - y\big) \approx (C^{xx})^{-1} C^{xy} \Gamma^{-1}\big(\mathcal{A}(x) - y\big), \tag{120}$$

and we can consider gradient descent with this approximation. We can also derive analogues of more sophisticated solvers such as Levenberg-Marquardt, as we show next.

C.1 Ensemble Kalman update rule and the Levenberg-Marquardt solver

We closely follow the presentation in (Sanz-Alonso et al., 2023, Chapter 13) for the derivations of the ensemble Kalman variant of the Levenberg-Marquardt (LM) solver. The derivation largely follow from the following simple lemma:

Lemma 7 (Sanz-Alonso et al. (2023), Lemma 13.3)**.** *The following identity holds*

$$\frac{1}{2}\|y - Ax\|_\Gamma^2 + \frac{1}{2}\|x - m\|_C^2 = \frac{1}{2}\|x - \mu\|_\Sigma^2 + \beta, \tag{121}$$

where β is a constant independent of x, and

$$\mu := m + K(y - Am), \tag{122}$$

$$\Sigma := (I - KA)C, \tag{123}$$

where K is the **Kalman gain matrix***, given by*

$$K = CA^\top (ACA^\top + \Gamma)^{-1}. \tag{124}$$

Equations (122)–(123) define the **Kalman update rule**, which give us closed form expressions for the solution to the regularised least squares problem, given by the LHS of equation 121, and the associated curvature at that minima.

The Levenberg-Marquardt algorithm seeks to minimise the data misfit loss

$$\mathcal{J}(x) = \frac{1}{2}\|y - \mathcal{A}(x)\|_\Gamma^2, \tag{125}$$

and for a learning rate $h > 0$, proceeds by iterating

$$x_{n+1} = x_n + g_n(h), \tag{126}$$

where $g_n(h)$ is a minima of the least squares problem

$$g_n(h) = \arg\min_v \left\{ \frac{1}{2}\big\|y - \mathcal{A}(x_n) - D\mathcal{A}(x_n)v\big\|_\Gamma^2 + \frac{1}{2h}\big\|xv\big\|_C^2 \right\}, \tag{127}$$

for some PSD matrix C. This may be interpreted as minimising $\frac{1}{2}\big|y - \mathcal{A}(x_n) - D\mathcal{A}(x_n)v\big|_\Gamma^2$, while restricting the search space of v to a ball $\{v : \|v\|_C \le \delta\}$ for some δ associated to h, i.e., minimise within a *trust region*.

Invoking Lemma 7, we get

$$v_n(h) = K(h)(y - \mathcal{A}(x_n)), \tag{128}$$

where the Kalman gain is given by

$$K(h) := hCD^\top \mathcal{A}(x_n) \left(hD\mathcal{A}(x_n)\, C\, D^\top \mathcal{A}(x_n) + \Gamma\right)^{-1}. \tag{129}$$

Now recall that by statistical linearisation, given a cloud of particles $\{x^{(j)}\}_{j=1}^J$, we have that

$$\mathcal{A}(x) \approx (C^{xy})^\top (C^{xx})^{-1}(x - \bar{x}) + \overline{\mathcal{A}(x)}, \tag{130}$$

$$D\mathcal{A}(x) \approx (C^{xy})^\top (C^{xx})^{-1}. \tag{131}$$

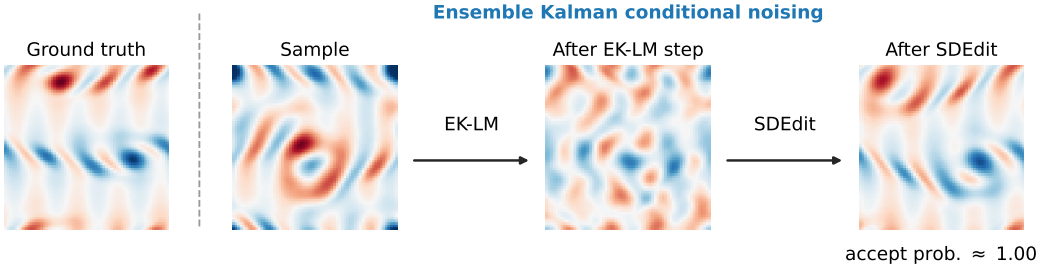


Figure 7: Illustration of the ensemble Kalman conditional noising diffusion waltz (DW-EKCN). This applies a single step of ensemble Kalman Levenberg-Marquardt (EK-LM) update to minimise data misfit, followed by applying SDEdit to project it back onto the data manifold.

Since C in equation 127 is arbitrary, we set $C = C^{xx}$. This implies

$$C^{xy} \overset{(118)}{=} CD^\top \mathcal{A}(x), \tag{132}$$

and

$$C^{yy} := \frac{1}{J}\sum_{i,j=1}^{J} \left(\mathcal{A}(x^{(i)}) - \overline{\mathcal{A}(x)}\right)\left(\mathcal{A}(x^{(j)}) - \overline{\mathcal{A}(x)}\right)^\top \tag{133}$$

$$\overset{(130)}{=} \frac{1}{J}\sum_{i,j=1}^{J} (C^{xy})^\top (C^{xx})^{-1}(x^{(i)} - \bar{x})(x^{(j)} - \bar{x})^\top (C^{xx})^{-1} C^{xy} \tag{134}$$

$$\overset{(118)}{=} (C^{xy})^\top (C^{xx})^{-1} C_{xx} (C^{xx})^{-1} C^{xy} \tag{135}$$

$$\overset{(131)}{\approx} D\mathcal{A}(x) C D^\top \mathcal{A}(x). \tag{136}$$

Thus, we can approximate the Kalman gain (129) in a gradient-free manner using ensembles as

$$K(h) \approx hC^{xy}\left(hC^{yy} + \Gamma\right)^{-1} = C^{xy}\left(C^{yy} + h^{-1}\Gamma\right)^{-1}. \tag{137}$$

In practice, one iterates equation 126 for an ensemble of particles $\{x_n^{(j)}\}_{j=1}^J$ in parallel and at each time step n, the Kalman gain (137) is computed using the ensemble $\{x_n^{(j)}\}_{j=1}^J$.

C.1.1 Details on the EKCN kernel

We use the ensemble Kalman variant of the LM update rule to define our ensemble Kalman conditional noising (EKCN) kernel $q(x_t \mid x_0, y)$. We split this into two steps:

1. For fixed $h > 0$, apply a single ensemble Kalman LM update to x_0:
$$\tilde{x}_0 = x_0 + K(h)(y - \mathcal{A}(x_0)), \tag{138}$$
where $K(h)$ is computed using equation 137.
2. Add noise via the interpolant
$$x_t = \alpha_t \tilde{x}_0 + \beta_t x_1, \quad x_1 \sim p_1. \tag{139}$$

Combining this with the denoising step in diffusion waltz

3. Denoise $x_0' \sim p(x_0 \mid x_t)$,

the full conditional noising waltz proposal kernel $Q(x_0, x_0')$ may be understood as first applying an ensemble Kalman update to minimise the data misfit (125) (step 1); however, since without prior regularisation, this is likely to push the particle off-manifold, this is corrected by applying SDEdit Meng et al. (2022) (steps 2-3), which keeps it on-manifold (see Figure 7 for an illustration).

In practice, we run J MCMC chains in parallel and the Kalman gain $K(h)$ at each step is computed using the ensemble. However, when the observation dimension is high, the matrix inverse in equation 137 is expensive to compute. Following a standard trick used in ensemble Kalman filters Evensen (2003), we use the pushthrough identity identity $V(V^TV + \lambda I)^{-1} = (VV^T + \lambda I)^{-1}V$ to reduce this complexity. Letting $\Gamma = \sigma_y^2 I$, $C^{xy} = U^\top V$ and $C^{yy} = V^\top V$, where

$$U_j = \frac{1}{\sqrt{J}}(x_0^{(j)} - \bar{x}_0), \qquad V_j = \frac{1}{\sqrt{J}}(\mathcal{A}(x_0^{(j)}) - \overline{\mathcal{A}(x_0)}), \tag{140}$$

we have

$$K(h) \overset{(137)}{\approx} U^\top V \left(V^\top V + (\sigma_y^2/h)I\right)^{-1} \tag{141}$$

$$= U^\top \left(VV^\top + (\sigma_y^2/h)I\right)^{-1} V. \tag{142}$$

Noting that the matrix inverse in equation 142 now has size $J \times J$, and since the ensemble size J is a hyperparameter that one can choose freely, the computation of equation 142 can be made tractable.

We now make a remark that in the computation of the acceptance probability

$$\alpha = \min\left\{1, \frac{p(y \mid x_0')\, q(x_t \mid x_0', y)\, p(x_t \mid x_0)}{p(y \mid x_0)\, q(x_t \mid x_0, y)\, p(x_t \mid x_0')}\right\}, \tag{143}$$

we *use the same Kalman gain $K(h)$ in the computation of $q(x_t \mid x_0', y)$ and $q(x_t \mid x_0, y)$* to ensure that the same kernel q is used in both the numerator and denominator. More precisely, given an ensemble $\{x_0^{(j)}\}_{j=1}^J$, we first compute $K(h)$ using this ensemble, cache it, and compute

$$q(x_t^{(j)} \mid x_0^{(j)}, y) = \mathcal{N}\Big(x_t^{(j)} \mid \alpha_t\big(x_0^{(j)} + K(h)\big(y - \mathcal{A}(x_0^{(j)})\big)\big), \beta_t^2 C\Big) \tag{144}$$

$$q(x_t^{(j)} \mid x_0^{'(j)}, y) = \mathcal{N}\Big(x_t^{(j)} \mid \alpha_t\big(x_0^{'(j)} + K(h)\big(y - \mathcal{A}(x_0^{'(j)})\big)\big), \beta_t^2 C\Big), \tag{145}$$

for *the same* $K(h)$, i.e., we do not recompute it in (145) using the updated samples $\{x_0^{'(j)}\}_{j=1}^J$.

Remark 4. *The interacting particle-style update in equations (144)–(145) subtly breaks exactness of the method due to the dependence of the proposal on the law of the particles. Methods exist to overcome this, e.g., splitting particles into groups and updating one group using information from the other Goodman & Weare (2010); however, in practice, we found that simply using all the ensemble members yield faster convergence and better empirical performance.*

D Adaptive hyperparameter tuning

We consider Robbins-Monro method to adaptively tune the hyperparameters $t \in (0, 1)$ (the noising time used in waltz) and $h > 0$ (the Levenberg-Marquardt step size) (Andrieu & Thoms, 2008). This reads

$$\operatorname{logit}(t_{n+1}) = \texttt{clip}\big(\operatorname{logit}(t_n) + \eta_n(\alpha_n - \alpha^*)\big), \tag{146}$$

$$\log(h_{n+1}) = \texttt{clip}\big(\log(h_n) + \eta_n(\alpha_n - \alpha^*)\big), \tag{147}$$

where α^* is a target acceptance rate, α_n is the instantaneous acceptance rate, and $\eta_n > 0$ is a (possibly decaying) step-size, all specified by the users. Clipping is applied e.g. to prevent t from being driven down to 0.

In general, we find that for conditional noising waltz, adaptively tuning the hyperparameters starting from high t, h improves the performance. Intuitively, this is because the LM update in equation 138 makes large jumps early in the iteration, requiring a larger noising time t to project the proposal $\tilde{x}_0$ back onto the data manifold. However, as the chains converge, the updates become smaller, requiring smaller t. Robbins-Monro automatically drives t down to smaller values as MCMC progresses, as acceptance rates decrease when the chains have converged to high likelihood regions and smaller "edits" become necessary for higher acceptance.

For the choice of learning rate η_n, we use either a constant learning rate with hard-thresholding (t/h pin at their lower bounds after relaxation) or a decaying learning rate, e.g., $\eta_n = n^{-2/3}$ to ensure

reversibility. In our experiments with EKCN waltz, we find that tuning all chains simultaneously with a constant learning rate+thresholding, and using $\alpha_n =$ *the fraction of chains (out of J) accepted at step n*, to work slightly better than adapting the hyperparameters of each chain separately. For gradient-based conditional noising waltz, we default to a decaying learning rate with independent hyperparameters, using $\alpha_n \in \{0, 1\}$ depending on whether the chain at step n was accepted (1) or rejected (0).

E EXPERIMENTAL DETAILS

In this appendix, we provide further details of our experiments in Section 5.

E.1 METRICS

To evaluate the performance of the posterior sampling methods, we draw J samples $\{x_{0,j}\}_{j=1}^{J}$ from the approximate posterior $p(x_0 \mid y)$ produced by each method and compare them against the ground truth $x_0^\dagger$. We report the following metrics, each of which captures a different aspect of sample quality: accuracy of the point estimate, distributional fit, and calibration of the uncertainty.

Relative ℓ_2 (Rel-ℓ_2). The Rel-ℓ_2 error measures how close the ensemble mean $\bar{x}_0$ is to the ground truth, normalized by the norm of the ground truth itself. It is a standard measure of reconstruction accuracy and is agnostic to the spread of the posterior samples.

$$\texttt{RelL2} = \frac{\|\bar{x}_0 - x_0^\dagger\|_{L^2}}{\|x_0^\dagger\|_{L^2}}, \quad \text{where} \quad \bar{x}_0 = \frac{1}{J}\sum_{j=1}^{J} x_{0,j} \tag{148}$$

Lower values indicate that the posterior mean is closer to the true signal.

Continuous ranked probability score (CRPS). Unlike Rel-ℓ_2, CRPS is a proper scoring rule that evaluates the full ensemble rather than just its mean. It rewards both accuracy (samples close to $x_0^\dagger$) and sharpness (low spread among samples), while penalizing ensembles that are either biased or overly diffuse.

$$\texttt{CRPS} = \frac{1}{J}\sum_{j=1}^{J} \|x_0^{(j)} - x_0^\dagger\|_{L^1} - \frac{1}{2J(J-1)}\sum_{i,j=1}^{J} \|x_0^{(i)} - x_0^{(j)}\|_{L^1}. \tag{149}$$

Lower CRPS values indicate a better-calibrated and more accurate ensemble.

Spread skill ratio (SSR). The SSR assesses whether the spread of the posterior ensemble is consistent with its actual error, i.e., whether the uncertainty estimates are well calibrated. The numerator captures the average ensemble variance (spread), while the denominator captures the squared error of the ensemble mean (skill).

$$\texttt{SSR} = \sqrt{\left(\frac{J+1}{J}\right)\frac{\frac{1}{J}\sum_{j=1}^{J}\|x_0^{(j)} - \bar{x}_0\|_{L^2}^2}{\|\frac{1}{J}\sum_{j=1}^{J} x_0^{(j)} - x_0^\dagger\|_{L^2}^2}}, \tag{150}$$

An SSR close to 1 indicates that the ensemble spread matches the actual estimation error (i.e., the posterior is well calibrated). Values of $\texttt{SSR} < 1$ indicate an overconfident (underdispersed) posterior, while $\texttt{SSR} > 1$ indicates an overdispersed posterior.

E.2 POSTERIOR ACCURACY EVALUATION

In this section, we consider two experiments to verify the diffusion waltz's ability to recover posteriors accurately. The first example is a low-dimensional checkerboard

E.2.1 CHECKERBOARD INVERSE PROBLEM

We consider the same setting used in Mammadov et al. (2026) for the checkerboard inverse problem. For the flow matching velocity, we use a "closed form" version obtained by importance sampling:

$$b(t, x_t) := \iint_{\mathbb{R}^2} \left(\dot{\alpha}_t x_0 + \dot{\beta}_t x_1\right) p(x_0, x_1 \mid x_t) \mathrm{d}x_0 \mathrm{d}x_1 \tag{151}$$

$$= \frac{1}{Z} \iint_{\mathbb{R}^2} \left(\dot{\alpha}_t x_0 + \dot{\beta}_t x_1\right) p(x_t \mid x_0, x_1) p_0(x_0) p_1(x_1) \mathrm{d}x_0 \mathrm{d}x_1 \tag{152}$$

$$= \frac{1}{Z} \iint_{\mathbb{R}^2} \left(\dot{\alpha}_t x_0 + \dot{\beta}_t x_1\right) \delta(x_t - (\alpha_t x_0 + \beta_t x_1)) p_0(x_0) p_1(x_1) \mathrm{d}x_0 \mathrm{d}x_1 \tag{153}$$

$$= \frac{1}{Z\beta_t^2} \iint_{\mathbb{R}^2} \left(\dot{\alpha}_t x_0 + \dot{\beta}_t x_1\right) \delta\left(x_1 - \frac{x_t - \alpha_t x_0}{\beta_t}\right) p_0(x_0) p_1(x_1) \mathrm{d}x_0 \mathrm{d}x_1 \tag{154}$$

$$= \frac{1}{Z\beta_t^2} \int_{\mathbb{R}^2} \left(\dot{\alpha}_t x_0 + \frac{\dot{\beta}_t (x_t - \alpha_t x_0)}{\beta_t}\right) p_1\left(\frac{x_t - \alpha_t x_0}{\beta_t}\right) p_0(x_0) \mathrm{d}x_0 \tag{155}$$

$$\approx \frac{1}{Z\beta_t^2} \sum_{n=1}^{N} \left(\dot{\alpha}_t x_0^{(n)} + \frac{\dot{\beta}_t \left(x_t - \alpha_t x_0^{(n)}\right)}{\beta_t}\right) p_1\left(\frac{x_t - \alpha_t x_0^{(n)}}{\beta_t}\right), \; x_0^{(n)} \overset{\text{i.i.d.}}{\sim} p_0(x_0), \tag{156}$$

where we use a Gaussian reference, i.e., $p_1(x) = \mathcal{N}(x \mid 0, I)$, and the normalising constant reads

$$Z = \iint_{\mathbb{R}^2} p(x_t \mid x_0, x_1) p_0(x_0) p_1(x_1) \mathrm{d}x_0 \mathrm{d}x_1 \tag{157}$$

$$= \frac{1}{\beta_t^2} \int_{\mathbb{R}^2} p_1\left(\frac{x_t - \alpha_t x_0}{\beta_t}\right) p_0(x_0) \mathrm{d}x_0 \tag{158}$$

$$\approx \frac{1}{\beta_t^2} \sum_{m=1}^{N} p_1\left(\frac{x_t - \alpha_t x_0^{(m)}}{\beta_t}\right), \; x_0^{(m)} \overset{\text{i.i.d.}}{\sim} p_0(x_0). \tag{159}$$

Putting this together, we have

$$b(t, x_t) \approx \frac{\sum_{n=1}^{N} \left(\dot{\alpha}_t x_0^{(n)} + \dot{\beta}_t \, x_1^{(n)}(x_t)\right) p_1\left(x_1^{(n)}(x_t)\right)}{\sum_{m=1}^{N} p_1\left(x_1^{(m)}(x_t)\right)}, \qquad x_1^{(n)}(x_t) := \frac{x_t - \alpha_t x_0^{(n)}}{\beta_t}. \tag{160}$$

For the MFM denoiser, we use self-distillation with the mean flow loss (see (Potaptchik et al., 2026) for details).

For the baseline, we consider MMPS (Rozet et al., 2024) on this problem, which improves over DPS (Chung et al., 2023) by taking second order moment into account when approximating the denoiser, i.e., they consider $p(x_0 \mid x_t) \approx \mathcal{N}\big(x_0 \mid \mathbb{E}[x_0 \mid x_t], \mathbb{V}[x_0 \mid x_t]\big)$, where $\mathbb{V}[x_0 \mid x_t]$ is the covariance of $p(x_0 \mid x_t)$, which also admits a closed-form expression in terms of the score/drift. An advantage of MMPS over DPS is that guidance strength is purely controlled by the amount of observation noise, thus no ad-hoc tuning of guidance strength is necessary, making it suitable for this setting.

E.2.2 GAUSSIAN PROCESS REGRESSION

In the second experiment to verify posterior accuracy, we test conditional noising waltz's ability to recover the posterior in a standard Gaussian process (GP) regression problem. We use a Matérn-5/2 Gaussian prior with lengthscale = 0.1 on a 64×64 grid and observe 64 random point locations with observation noise $\sigma_y = 0.1$. We consider (i) gradient-based conditional noising waltz using the Levenberg-Marquardt update rule (DW-LM), and (ii) EKCN diffusion waltz (DW-EKCN). As our guidance-based baseline, we again consider MMPS, in addition to a reference obtained by resampling the true GP posterior, serving as a "best achievable" result. For the conditional noising waltz, we consider adaptive tuning of hyperparameters (Appendix D) intialised at $t = 0.7$ with decaying

step-size (with $\eta_n = n^{-0.6}$) and t clipped to between 0.05 and 0.7. For the Levenberg-Marquardt learning rate, we initialised at $h = 1.0$ for DW-LM and $h = 0.1$ for DW-EKCN, which lead to better acceptance rates throughout. To isolate the method, we considered an exact denoiser, obtained by considering the mirror interpolant $p_0 = p_1 = \mathcal{N}(0, \mathcal{C})$ (our GP prior) and using equation 32.

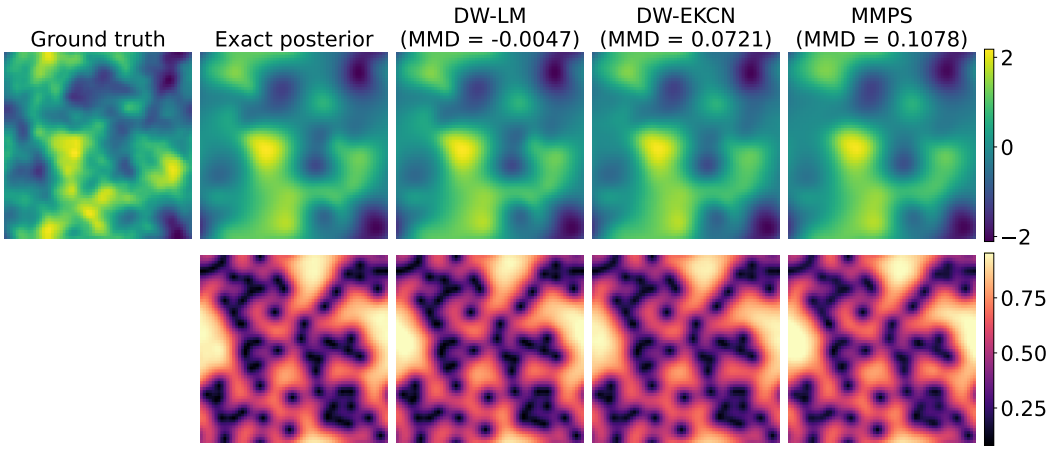


Figure 8: Visualisation of results on Gaussian process regression achieved (a) via exact Gaussian computation, (b) using conditional noising waltz, and (c) MMPS.

We plot the results in Figure 8 and display the corresponding metrics in Table 4, where we ran 1000 chains in parallel for a total of 2000 MCMC steps and used the 1000 samples at the final step to evaluate results — in general, we find that due to low acceptance rates once chains converge, it is more effective to run multiple chains in parallel (especially with massive parallelisation enabled by modern GPUs) and use the final ensemble for evaluation rather than running a single long chain and compute statistics from the chain. For MMPS, we also used 1000 independent samples for evaluation. The metrics used were: relative mean and covariance errors posterior (the latter is computed using the Frobenius norm), MMD (Gretton et al., 2012) and the sliced Wasserstein distance (Bonneel et al., 2015); all computed relative to the exact GP posterior.

While visual results for all methods look similar in Figure 8, Table 4 shows that both conditional noising waltz methods outperform MMPS on all metrics. In particular, DM-LM achieve near-optimal performance, reaching the accuracy of the GP posterior resampling baseline. On the other hand, DW-EKCN does not reach this level, following from the fact that it breaks exactness due to the dependence on the law of the ensembles during proposal. However, despite DW-EKCN's inexactness, the performance is still strong in comparison to MMPS's results.

Method	Rel. Mean Err.	Rel. Cov. Err.	MMD	Sliced W_2
DW-LM	**0.0205**	0.1956	**-0.0047***	**0.0422**
DW-EKCN	0.0855	**0.1923**	0.0721	0.0896
MMPS	0.1352	0.2169	0.1078	0.1202
Ref. (exact samples)	0.0214	0.1957	-0.0101*	0.0438

Table 4: Quantitative results on the Gaussian regression problem. Covariance error is computed using the Frobenius norm. * displayed MMD is negative as it uses an unbiased estimate.

Next, we use the exact drift:

$$b(t,x) = \mathbb{E}[\dot{x}_t \mid x_t = x] \tag{161}$$
$$= \dot{\alpha}_t \mathbb{E}[x_0 \mid x_t = x] + \dot{\beta}_t \mathbb{E}[x_1 \mid x_t = x] \tag{162}$$
$$\overset{(32)}{=} \dot{\alpha}_t \left(\frac{\alpha_t x}{\alpha_t^2 + \beta_t^2} \right) + \dot{\beta}_t \left(\frac{\beta_t x}{\alpha_t^2 + \beta_t^2} \right) \tag{163}$$
$$= \frac{\dot{\alpha}_t \alpha_t + \dot{\beta}_t \beta_t}{\alpha_t^2 + \beta_t^2} x, \tag{164}$$

to ablate over the number of SDE steps, while isolating the error incurred by the SDE discretisation. In Figure 9, we display the MMD and the sliced Wasserstein distance of samples obtained from DW-LM with the SDE denoiser, computed against the exact GP posterior. As a reference, we also plot the MMD and sliced Wasserstein distance of DW-LM using the exact denoiser (displayed via dashed lines). As expected, we see that the results on both metrics converge to the exact denoiser's results as we increase the number of SDE step sizes. Near-exact performance is already achieved at around $10 \sim 25$ SDE steps, highlighting the stability of DW-LM to discretisation errors.

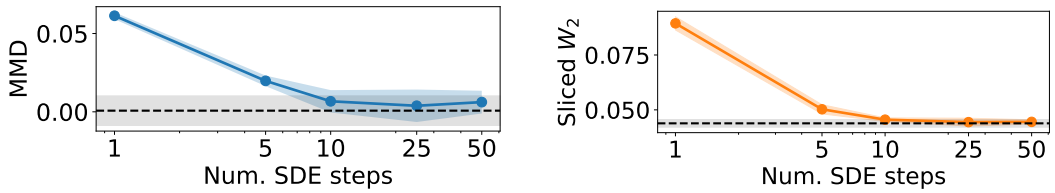


Figure 9: Ablation of DW-LM's performance on the number of SDE steps when using a SDE denoiser. We compute the MMD (left) and sliced Wasserstein distance (right) of samples obtained from DW-LM against the ground truth posterior obtained by exact Gaussian computation. We compute results on ten different seeds and display the mean and standard deviation. As a reference, we also display the results using an exact denoiser, displayed in gray.

E.3 Navier-Stokes initial condition recovery

We consider the Navier-Stokes (NS) equation on a 2D torus (vorticity form)

$$\begin{aligned} \partial_t \omega(\boldsymbol{x},t) + \boldsymbol{u}(\boldsymbol{x},t) \cdot \nabla \omega(\boldsymbol{x},t) &= \nu \Delta \omega(\boldsymbol{x},t) + f(\boldsymbol{x}), && \boldsymbol{x} \in (0,2\pi)^2, t \in (0,T] \\ \nabla \cdot \boldsymbol{u}(\boldsymbol{x},t) &= 0, && \boldsymbol{x} \in (0,2\pi)^2, t \in [0,T] \\ \omega(\boldsymbol{x},0) &= \omega_0(\boldsymbol{x}), && \boldsymbol{x} \in (0,2\pi)^2. \end{aligned} \tag{165}$$

Here, $\boldsymbol{u}(\boldsymbol{x},t) := (u(\boldsymbol{x},t), v(\boldsymbol{x},t))$ is the velocity field, $\omega(\boldsymbol{x},t) := \frac{\partial v}{\partial x}(\boldsymbol{x},t) - \frac{\partial u}{\partial y}(\boldsymbol{x},t)$ is the vorticity field, ν is the Reynolds' number and $f(\boldsymbol{x})$ is an external forcing forcing.

Inverse problem set up. The observations for our inverse problem are obtained as follows: First, we solve the NS system for T time units using an in-house pseudo-spectral NS solver with finite-difference time-stepping to obtain $\mathcal{F}_T : \omega_0 \mapsto \omega_T$. Then, we take a spatially strided downsampling $\mathcal{P}_k$ ($\times k$ downsampling), with additive isotropic Gaussian noise scaled relative to the dataset standard deviation σ_0. Thus the observation model reads

$$y = \mathcal{P}_k \circ \mathcal{F}_T(\omega_0) + \sigma_y \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I). \tag{166}$$

In our experiments, we fix $k = 4$ and consider various choices of T and σ_y controlling the degree of nonlinearity and intensity of noise (see Table 5)

Our data comes from Navier-Stokes vorticity trajectories (200 trajectories of 501 timesteps at 64×64 resolution with Kolmogorov forcing $f(x,y) = -4\cos(4y)$ and Reynolds' number $\nu = 40$) in (Li et al., 2022). To avoid highly correlated samples, we thin the trajectories by subsampling every 5 timesteps, giving us $\sim 20,000$ snapshots, split by trajectory into 180 train / 20 validation trajectories.

Table 5: Navier-Stokes experimental regimes

	Low	High
Nonlinearity	$T = 1$	$T = 5$
Noise	$\sigma_y/\sigma_0 = 0.1$	$\sigma_y/\sigma_0 = 1.0$

Generative prior. We use functional flow matching Kerrigan et al. (2023), where we parameterise the flow matching marginal velocity $b_\theta(t, \omega_t)$ by a Fourier Neural Operator (FNO) (Li et al., 2021). The input field ω_t is concatenated with an explicit 2D positional grid $\boldsymbol{x}$ and a constant channel broadcasting the flow-matching time t, then lifted to a hidden width of 64 channels via a pointwise MLP. This is followed by 4 Fourier layers, each retaining the lowest 32 Fourier modes per spatial axis, with GELU activations. The final hidden representation is projected back to the physical output channel via a second pointwise MLP.

We use a linear interpolant $\alpha_t = 1 - t$, $\beta_t = t$ and a Matérn Gaussian-1/2 process (GP) reference measure p_1 with lengthscale 0.1. In practice, we sample from the GP using the Karhunen-Loève expansion

$$\omega_1(\boldsymbol{x}) = \sum_{k \in \mathbb{Z}^2} \sqrt{\lambda_k}\, \xi_k\, e_k(\boldsymbol{x}), \qquad \xi_k \sim \mathcal{N}(0, 1) \text{ i.i.d.}, \tag{167}$$

where e_k are the 2D Fourier modes on the two-torus (i.e., $\sin(2\pi k\boldsymbol{x}), \cos(2\pi k\boldsymbol{x})$) and λ_k is the spectral density of the kernel:

$$\lambda_k = \sigma^2\, \frac{2\pi(2\nu)^{\nu+1}}{\ell^{2\nu}} \left(\frac{2\nu}{\ell^2} + |k|^2\right)^{-(\nu+1)}, \qquad k \in \mathbb{Z}^2. \tag{168}$$

We use the denoising SDE equation 24, which in the linear interpolant setting, reads

$$\mathrm{d}\omega_t(\boldsymbol{x}) = \left(-\frac{\omega_t(\boldsymbol{x})}{1-t} + 2b_\theta(t, \omega_t(\boldsymbol{x}))\right) \mathrm{d}t + \sqrt{\frac{2t}{1-t}} \mathrm{d}W_t^C, \tag{169}$$

solved backwards in time, where C is the Matérn covariance operator and $\{W_t^C\}_{t\in[0,1]}$ is the C-Brownian motion. By Proposition 1, this samples from $p_\theta(\omega_0 \mid \omega_t)$. In practice, we discretise the field ω_t on a grid of size $N \times N$, whose discretisation we denote by $\boldsymbol{\omega}_t$. We solve equation 169 using Euler-Maruyama:

$$\boldsymbol{\omega}_{n+1} = \boldsymbol{\omega}_n - \left(\frac{\boldsymbol{\omega}_n}{1-t_n} - 2\boldsymbol{b}_\theta(t, \boldsymbol{\omega}_n)\right) \Delta t + \sqrt{\frac{2t_n \Delta t}{1-t_n}} \boldsymbol{\xi}_n^C, \tag{170}$$

where $\boldsymbol{\xi}_n^C$ are i.i.d. draws from the GP using equation 167.

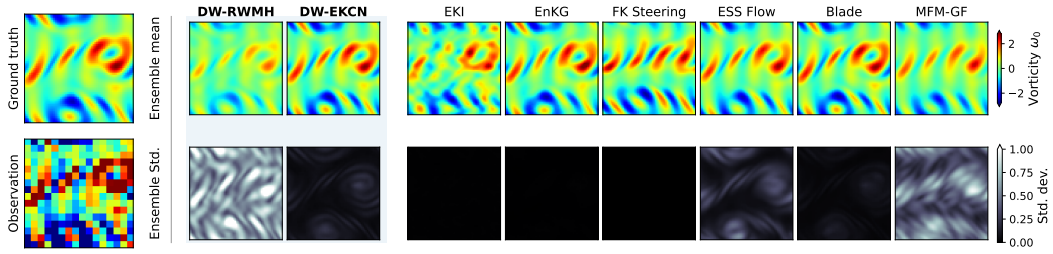
Baseline and method details. The hyperparameters of the baselines were tuned on five validation examples using grid search to minimise CRPS (except Blade, which used Bayesian optimisation due to the large number of sensitive hyperparameters) and tested on five examples, unseen during training of the flow matching model or validation. For slower methods such as Blade and ESS-Flow, we used an accurate FNO surrogate models in place of the numerical model to tune the hyperaparameters. Diffusion waltz was hand-tuned on this validation example, also using the surrogate model. See Table 7 for the final hyperparameters used in the experiments and Table 6 for the timings.

$T = 1$		$T = 5$	
Method	**Time (s)**	**Method**	**Time (s)**
EKI	36.6	EKI	123.0
EnKG	57.6	EnKG	192.4
FK Steering (no-grad)	22.3	FK Steering (no-grad)	91.3
ESS-Flow	3395.4	ESS-Flow	10886.4
Blade	939.2	Blade	3110.3
MFM-GF	113.0	MFM-GF	410.7
DW-RWMH	470.6	DW-RWMH	821.6
DW-EKCN	435.3	DW-EKCN	1857.9

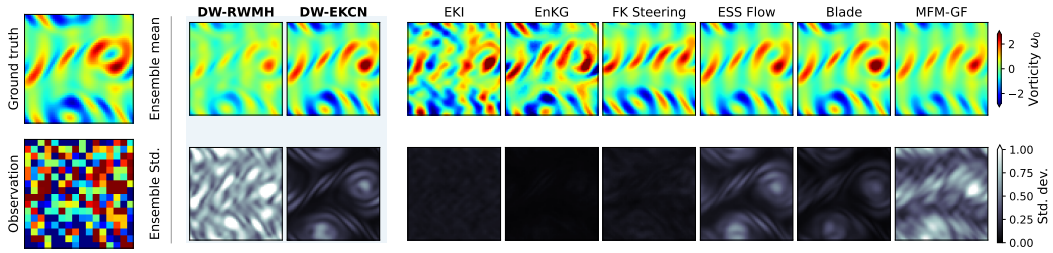
Table 6: Mean wall-clock time (100 posterior samples, using the real numerical NS solver) on the Navier-Stokes experiment, for $T = 1$ and $T = 5$.

	Low nonlinearity		**High nonlinearity**	
Methods	**Low noise**	**High noise**	**Low noise**	**High noise**
EKI	Num. steps = 50 h = 1.0	Num. steps = 50 h = 1.0	Num. steps = 50 h = 1.0	Num. steps = 50 h = 1.0
EnKG	Num. steps = 100 ζ = 1.0	Num. steps = 100 ζ = 1.0	Num. steps = 100 ζ = 1.0	Num. steps = 100 ζ = 1.0
FK Steering	Num. steps = 50 ζ = 0.03 Strategy = `SUM`	Num. steps = 50 ζ = 0.03 Strategy = `SUM`	Num. steps = 50 ζ = 0.03 Strategy = `DIFFERENCE`	Num. steps = 50 ζ = 0.03 Strategy = `DIFFERENCE`
ESS-Flow	Num. steps = 250 ODE steps = 25	Num. steps = 250 ODE steps = 25	Num. steps = 250 ODE steps = 25	Num. steps = 250 ODE steps = 25
Blade	Num. annealing = 50 Num. Langevin = 50 SDE steps = 25 γ = 5.6 τ = 0.05 ρ schedule = `linear` ρ_{max} = 1.6 ρ_{min} = 0.002	Num. annealing = 50 Num. Langevin = 50 SDE steps = 25 γ = 7.6 τ = 0.2 ρ schedule = `linear` ρ_{max} = 9.3 ρ_{min} = 0.002	Num. annealing = 50 Num. Langevin = 50 SDE steps = 25 γ = 27 τ = 0.05 ρ schedule = `linear` ρ_{max} = 2.5 ρ_{min} = 0.002	Num. annealing = 50 Num. Langevin = 50 SDE steps = 25 γ = 4.9 τ = 0.4 ρ schedule = `linear` ρ_{max} = 5.7 ρ_{min} = 0.002
MFM-GF	MC samples = 10 Num. steps = 100 Guidance strength = 2.0	MC samples = 10 Num. steps = 100 Guidance strength = 2.0	MC samples = 10 Num. steps = 100 Guidance strength = 2.0	MC samples = 10 Num. steps = 100 Guidance strength = 2.0
DW-RWMH	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 t_{bounds} = [0.3, 0.7]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 t_{bounds} = [0.3, 0.7]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 t_{bounds} = [0.3, 0.7]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 t_{bounds} = [0.3, 0.7]
DW-EKCN	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 h_{init} = 1.0 t_{bounds} = [0.3, 0.7] h_{bounds} = [0.01, 1.0]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 h_{init} = 1.0 t_{bounds} = [0.3, 0.7] h_{bounds} = [0.01, 1.0]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 h_{init} = 1.0 t_{bounds} = [0.3, 0.7] h_{bounds} = [0.01, 1.0]	MCMC steps = 250 Adaptive MCMC = `True` α^* = 0.3 η_n = 0.3 t_{init} = 0.7 h_{init} = 1.0 t_{bounds} = [0.3, 0.7] h_{bounds} = [0.01, 1.0]

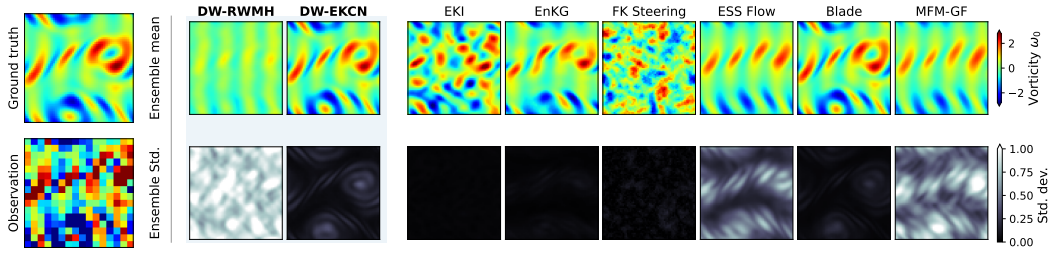
Table 7: Hyperparameters used for each method on the Navier-Stokes experiment.



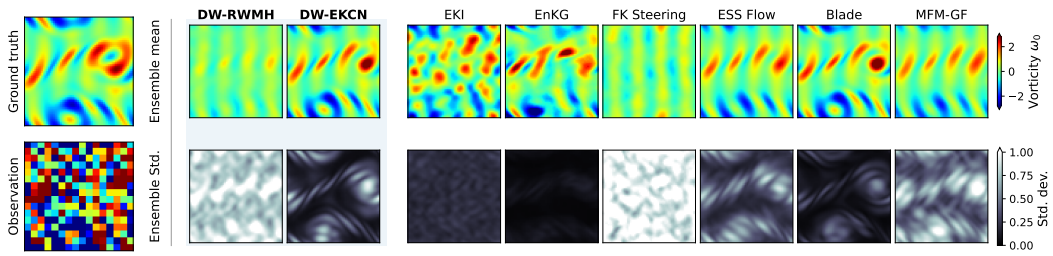
(a) Low nonlinearity ($T = 1$) / Low noise ($\sigma_y/\sigma_0 = 0.1$) regime.



(b) Low nonlinearity ($T = 1$) / High noise ($\sigma_y/\sigma_0 = 1.0$) regime.

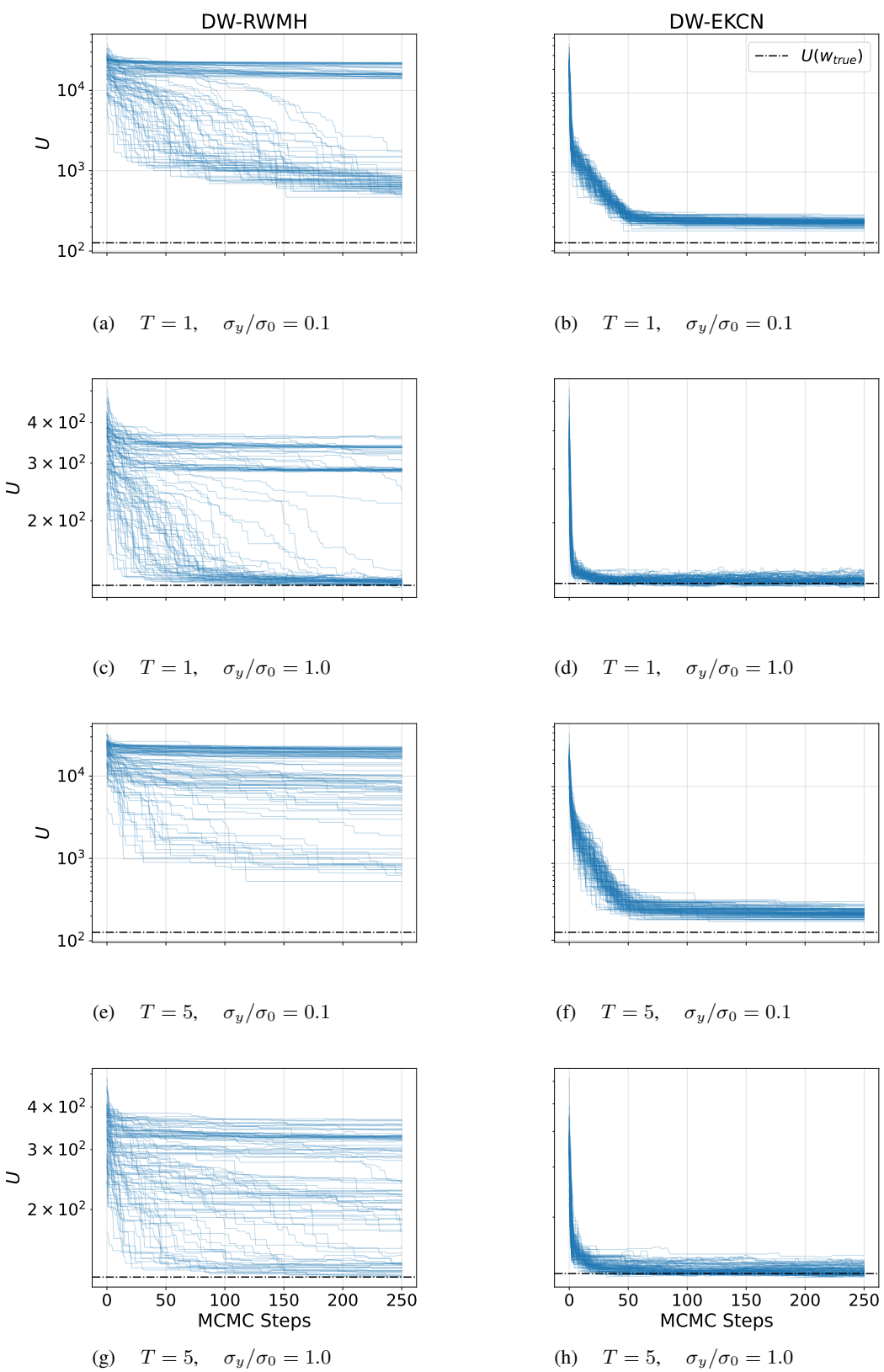


(c) High nonlinearity ($T = 5$) / Low noise ($\sigma_y/\sigma_0 = 0.1$) regime.



(d) High nonlinearity ($T = 5$) / High noise ($\sigma_y/\sigma_0 = 1.0$) regime.

Figure 10: Comparison across nonlinearity and noise regimes for the Navier–Stokes experiments. EKI, which is a classical method, produce off-manifold results, being an optimizer. EnKG addresses this issue but retains low spread. FK steering, which is based on SMC suffers from particle collapse in most regions and fails to learn anything in the high nonlinearity/high noise regime. On the other hand, MFM-GF yields underconfident results due to particle collapse in likelihood score estimation, yielding biased guidance. Blade and ESS flow yield competitive results. However, inference is slow especially for ESS flow. Comparing the two DW methods, DW-RWMH yields poor results, especially in the highly nonlinear regime due to poor mixing. DW-EKCN converges quickly and reconstructs ω_0 accurately with well-calibrated uncertainty.



(a) $T = 1, \quad \sigma_y/\sigma_0 = 0.1$

(b) $T = 1, \quad \sigma_y/\sigma_0 = 0.1$

(c) $T = 1, \quad \sigma_y/\sigma_0 = 1.0$

(d) $T = 1, \quad \sigma_y/\sigma_0 = 1.0$

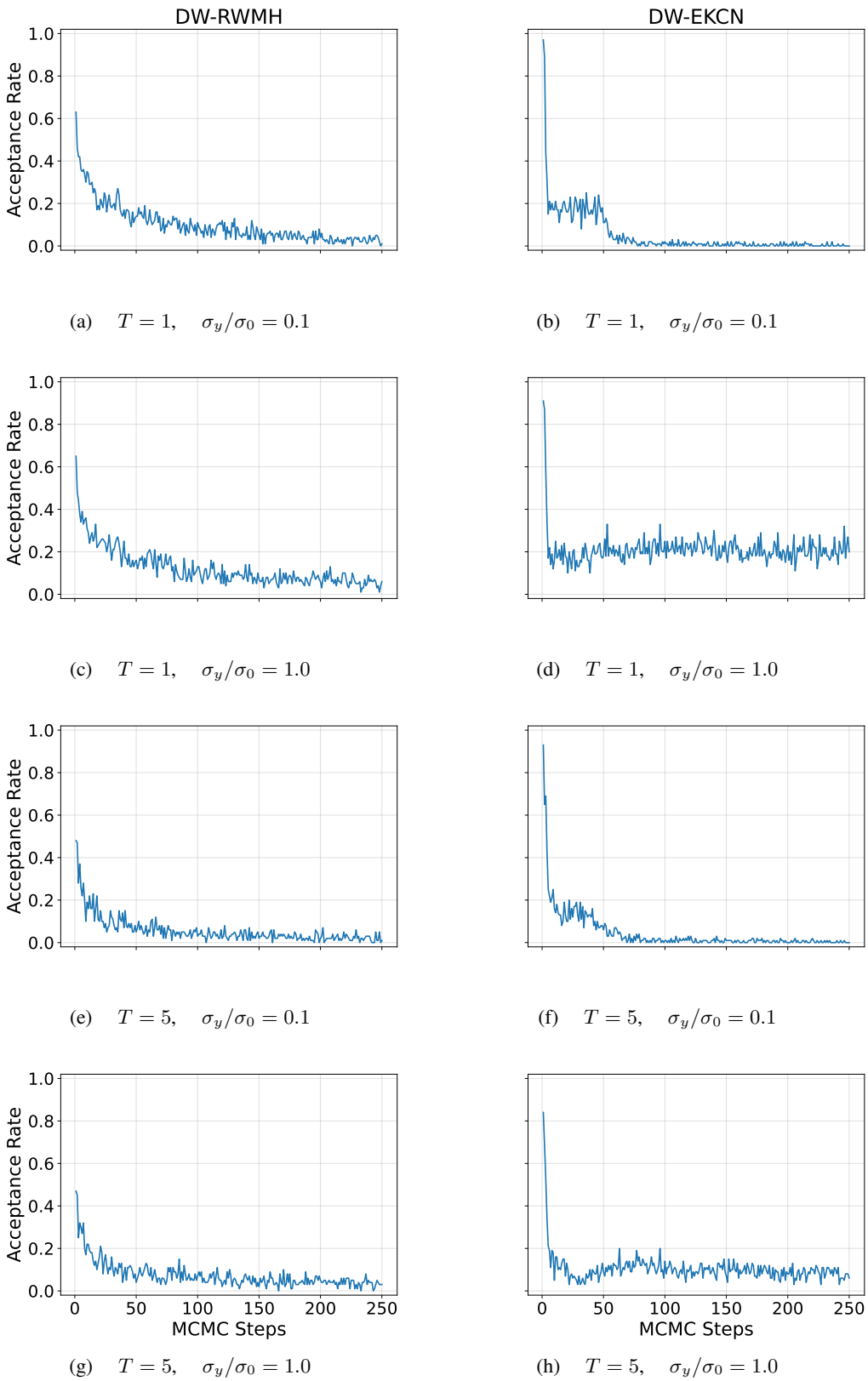
(e) $T = 5, \quad \sigma_y/\sigma_0 = 0.1$

(f) $T = 5, \quad \sigma_y/\sigma_0 = 0.1$

(g) $T = 5, \quad \sigma_y/\sigma_0 = 1.0$

(h) $T = 5, \quad \sigma_y/\sigma_0 = 1.0$

Figure 11: MCMC traces for the potential $U(x_0) := -\log p(y \mid x_0)$. We compare DW-RWMH (left) and DW-EKCN (right), across all four regimes. Many chains in DW-RWMH get stuck, unable to find a path to high likelihood/low potential regions, leading to slow convergence. Conditional noising in DW-EKCN alleviates this problem, leading to rapid convergence within the first 50 steps.



(a) $T = 1, \quad \sigma_y/\sigma_0 = 0.1$ (b) $T = 1, \quad \sigma_y/\sigma_0 = 0.1$

(c) $T = 1, \quad \sigma_y/\sigma_0 = 1.0$ (d) $T = 1, \quad \sigma_y/\sigma_0 = 1.0$

(e) $T = 5, \quad \sigma_y/\sigma_0 = 0.1$ (f) $T = 5, \quad \sigma_y/\sigma_0 = 0.1$

(g) $T = 5, \quad \sigma_y/\sigma_0 = 1.0$ (h) $T = 5, \quad \sigma_y/\sigma_0 = 1.0$

Figure 12: Instantaneous acceptance rates for DW-RWMH (left) and DW-EKCN (right), across all four regimes. DW-RWMH shows steady decay with MCMC steps. In the low noise regime, DW-EKCN has high acceptance rates initially, fluctuates around 20% while the chains converge, then fluctuations between $0 \sim 5\%$. In the high noise regime, acceptance rate is high initially, then immediately relaxes to $10 \sim 20\%$.

E.4 DARCY FLOW INVERSION.

Darcy flow is described by the system

$$-\nabla \cdot \left(e^{\kappa(x)} \nabla u(x)\right) = f(x), \quad x \in \Omega \tag{171}$$

$$u(x) = g(x), \quad x \in \partial\Omega. \tag{172}$$

where $\kappa(x)$ is the log-permeability field and $u(x)$ is the pressure field. Following the setting in Li et al. (2021), we set $f(x) \equiv 1$ and $g(x) \equiv 0$ on a unit square $\Omega = [0,1] \times [0,1]$. Since Li et al. (2021) uses a Gaussian-derived permeability field, which makes the inverse problem easy to solve using traditional methods, we use a more complex permeability dataset from Seabra et al. (2026), consisting of realistic channelised fields (see Figure 13 for samples of log-permeability fields from this dataset, alongside corresponding pressure fields). All samples have resolution 64×64.

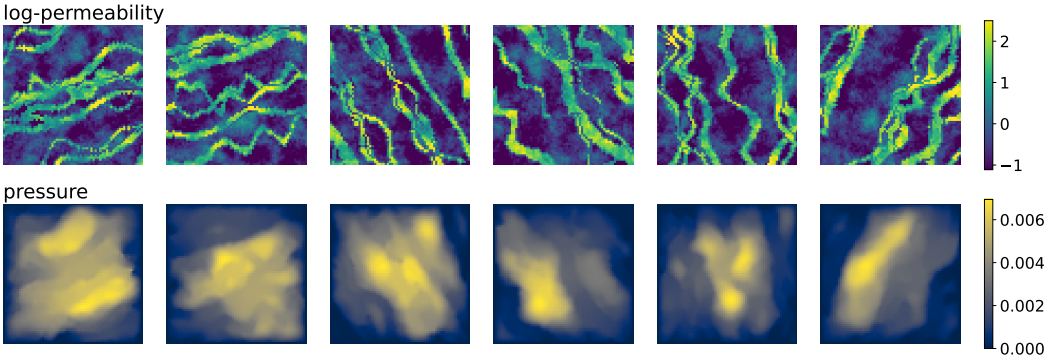


Figure 13: Examples of the channelised log-permeability fields from Seabra et al. (2026) (top row) and the corresponding pressure fields (bottom row) obtained by solving (171)–(172).

Inverse problem set up. For y, we use random point observations of the pressure field, that is, we sample $M = 256$ random points $\{x_m\}_{m=1}^M$ from our square domain Ω and take $y \in \mathbb{R}^M$, where

$$y_m = u(x_m) + \varepsilon_m, \quad \varepsilon_m \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_y^2 I), \quad m = 1, \ldots, M. \tag{173}$$

For the observation noise, we take $\sigma_y = \texttt{1e-3}$.

Adjoint model and gradients. Since the LHS of equation 171 is given by a self-adjoint operator, we can deduce the adjoint model easily: For any loss functional $J[u]$, the adjoint model of equation 171 reads

$$-\nabla \cdot \left(e^{\kappa(x)} \nabla A(x)\right) = \frac{\delta J}{\delta u}(x), \quad x \in \Omega, \qquad A(x) = 0, \quad x \in \partial\Omega. \tag{174}$$

In particular, choosing J to be the data misfit $J[u] = \frac{1}{2\sigma_y^2} \sum_{m=1}^M \|y_m - u(x_m)\|^2$, we get

$$-\nabla \cdot \left(e^{\kappa(x)} \nabla A(x)\right) = \sum_{m=1}^{M} \frac{u(x_n) - y_n}{\sigma_y^2} \, \delta(x - x_n), \quad x \in \Omega, \tag{175}$$

$$A(x) = 0, \quad x \in \partial\Omega. \tag{176}$$

The gradient $\delta J / \delta \kappa = (\delta J / \delta u)(\delta u / \delta \kappa)$ can be computed as:

$$\frac{\delta J}{\delta \kappa} = -e^{\kappa(x)} \nabla u(x) \cdot \nabla A(x). \tag{177}$$

We use central difference scheme on a regular 64×64 grid to solve the forward model (171) and the adjoint (175) numerically.

Generative prior. For the generative model, we use a U-Net backbone with channels $(32, 64, 128, 256)$ and self-attention layers in and around the bottleneck, to model the flow matching velocity $b_\theta(t, x)$. This is trained on the log-permeability data from Seabra et al. (2026) using the conditional flow matching loss (11), with linear interpolant coefficients $\alpha_t = 1 - t$, $\beta_t = t$.

Baseline and method details. The hyperparameters of the baselines were tuned on a single validation example using Bayesian optimisation to minimise CRPS and tested on five examples unseen during training of the flow matching model or validation. Diffusion waltz was hand-tuned on this validation example using short runs of 20 MCMC steps. The final hyperparameters of the baselines are displayed in Table 8 and for diffusion waltz in Table 10. For each method (excluding LM, which is an optimiser), we use 100 samples to compute the metrics. For diffusion waltz, similar to PnP-DM and Blade, we run 100 chains in parallel and compute the metrics using samples at the final step. The timings of the methods are shown in Table 9. In contrast to Navier-Stokes, Blade (and related methods DAPS and PnP-DM) runs much faster than DW. This is due to the hyperparameters learned during the validation step yielding low annealing and Langevin steps, giving conservative, prior reverting results. DW-AdCN is slow here as our Darcy solver can only be solved serially and we used a single GPU to run all 100 chains. However, DW-AdCN is embarassingly parallel and thus, we can reduce this time by parallelising across multiple GPUs.

Methods/Parameters	**Value**	**Methods/Parameters**	**Value**
DPS		**DAPS**	
Num. steps	300	Annealing steps	15
Guidance strength ζ	24,000	Langevin steps	12
		ODE steps	25
		Learning rate	`1e-4`
		Noise level τ	0.25
LM		**EKI**	
Max steps	250	Max steps	250
Early stopping	`True`	Early stopping	`True`
Tolerance	`1e-3`	Tolerance	`1e-3`
PnP-DM		**Blade**	
Annealing steps	10	Annealing steps	10
Langevin steps	10	Langevin steps	13
ODE steps	25	ODE steps	25
Learning rate	0.006	Guidance scale γ	165
Noise level τ	0.02	Noise level τ	0.06
pCN-Langevin		**FK Steering (gradient) / TDS**	
Matérn Smoothness ν	0.5	Num. steps	130
Matérn Lengthscale ℓ	0.02	Guidance scale ζ	0.005
Initialise from MAP	`True`	Strategy	`MAX`
Proposal step	`1e-3`		

Table 8: Hyperparameters used for each baseline method in the Darcy flow experiments. We use 100 samples for all methods (excluding LM, which is an optimisation method).

Method	**Mean time (s)**
Traditional	
LM	16.6
EKI*	1720.6
pCN-Langevin	218.5
Plug-and-play	
DPS	4135.8
DAPS	2467.7
PnP-DM	1408.4
Blade*	407.8
TDS	1686.8
Diffusion waltz (ours)	
DW-EKCN*	4215.5
DW-AdCN	8347.4

Table 9: Mean wall-clock time to generate 100 posterior samples (except LM). * indicates methods that are derivative-free.

DW-EKCN		DW-AdCN	
Parameter	**Value**	**Parameter**	**Value**
MCMC steps	250	MCMC steps	250
Num chains	100	Num chains	100
t_{init}	0.7	t_{init}	0.8
h_{init}	1.0	h_{init}	1.0
α^*	0.3	α^*	0.3
η_n	0.3	η_n	$n^{-0.6}$
t_{bounds}	[0.05, 0.7]	t_{bounds}	[0.05, 0.7]
h_{bounds}	[0.01, 1.0]	h_{bounds}	[0.01, 1.0]

Table 10: Hyperparameters used for DW-EKCN and DW-AdCN on the Darcy flow experiment.

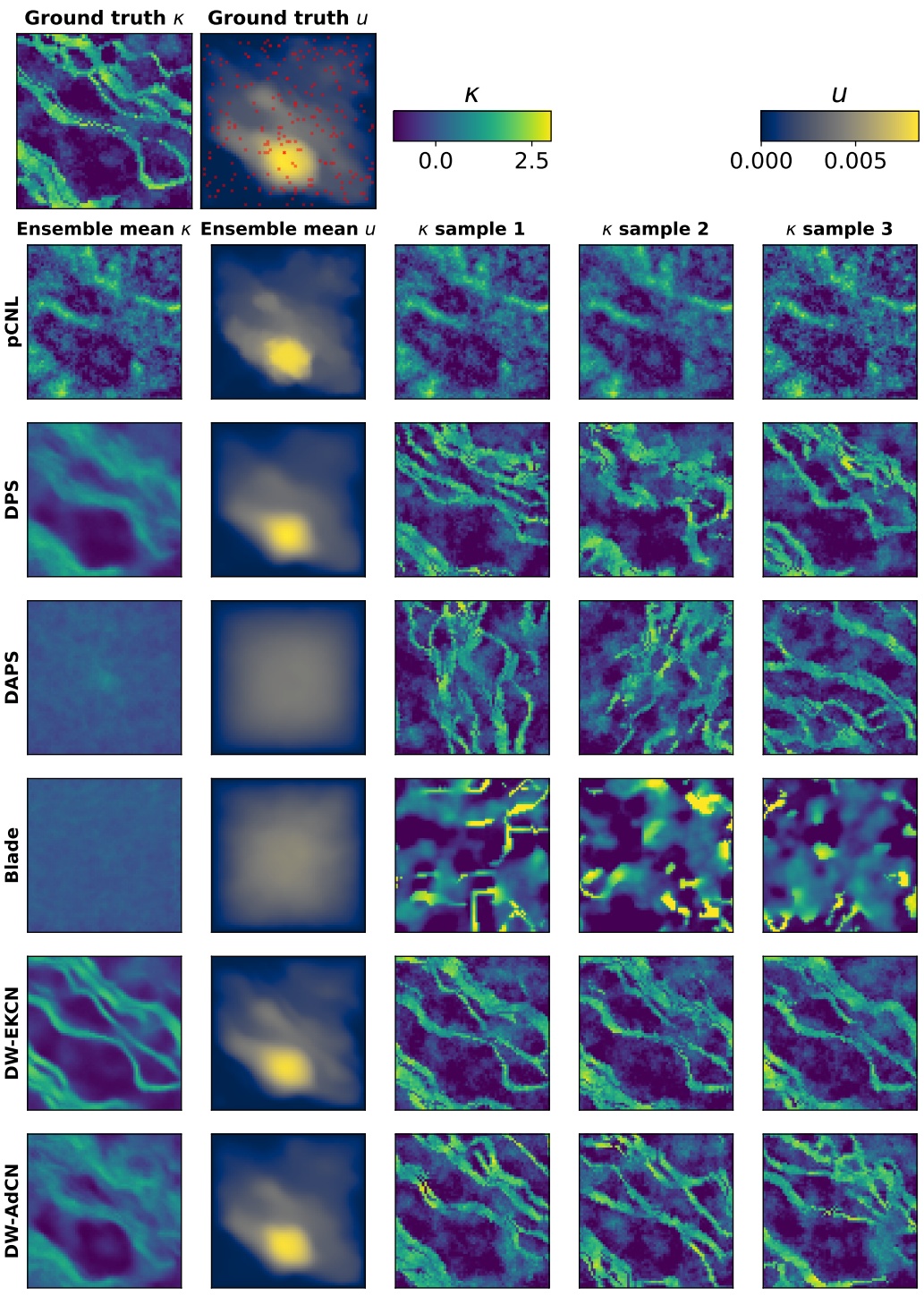


Figure 14: Result visualisations of diffusion waltz and baseline methods on the Darcy flow inversion problem. Traditional method such as pCN-Langevin, which use a Gaussian process prior, struggle due to prior mismatch; methods relying on Langevin sampling (DAPS, Blade) are unstable on this task, leading to conservative hyperparameters during tuning and therefore underconfident predictions. A well-tuned DPS is competitive here, however it is outperformed by the two diffusion waltz variants, with the adjoint variant producing better calibrated uncertainties.

E.5 3D INVERSE DESIGN.

In this experiment, we consider a 3D inverse design task of generating jet engine bracket designs with a target physical property. For the generative backbone, we use TRELLIS.2 (Xiang et al., 2026), an open-source, open-weight frontier model for 3D generative tasks. We fine-tune TRELLIS.2 with LoRA (Hu et al., 2022) on the SimJEB dataset (Whalen et al., 2021), consisting of diverse jet engine bracket designs. We fine tune conditionally with independent CFG dropout against mass and "category" labels from the original dataset.

A challenge with TRELLIS.2 is that geometry generation operates in multiple stages:

- In the first stage, 3D latents z_0 are generated using a flow matching prior $p_\theta(z_0)$ that get decoded to a *sparse structure (SS)*: a 3D voxel grid where only voxels intersecting the surface are active, determining the structure of the geometry to be generated. As only active voxels are stored, these discrete encodings vary in shape depending on occupancy.
- In the second stage, a second flow matching model is used to generate the *structured latents (SLat)* $x_0 \sim p_\theta(x_0 \mid z_0)$, which is a set of vectors that live on the active voxels determined by the SS, generated in the first stage.
- Finally, the SLats are decoded to an *O-voxel representation*, which can be be converted to a mesh with various attributes of interest, such as colour, texture and opacity (here, we are only interested in generating the mesh itself).

Without loss of generality, we may identify the generated mesh with its corresponding SLat x_0, as the process of decoding to a mesh is deterministic. Thus, we slightly abuse notation and use x_0 to denote both the SLat and the generated mesh.

Mesh post-processing. TRELLIS.2 is able to generate meshes with fine-details and internal structures. However, to achieve this flexibility, guarantees on manifoldness were deliberately relaxed, preventing immediate computation of basic geometric quantities such as volume (and consequently, mass), or use in simulation. We therefore postprocess raw TRELLIS.2 outputs by converting to an unsigned distance field and extracting a manifold surface via dual contouring, a common approach in data preprocessing for geometry model training.

In general, our post-processing method is stochastic, thus we may think about this step as generating $\tilde{x}_0 \sim p(\tilde{x}_0 \mid x_0)$, where $\tilde{x}_0$ is the post-processed mesh and x_0 is the SLat/original mesh. However, to simplify discussions, we will still use x_0 to denote our post-processed mesh. Thus, in our following discussions, $x_0 \sim p_\theta(x_0 \mid z_0)$, in reality means sampling $\tilde{x}_0 \sim \int p(\tilde{x}_0 \mid x_0) p_\theta(x_0 \mid z_0) \mathrm{d}x_0$.

Inverse problem set-up We formulate our inverse design problem as an inverse problem $p(x_0 \mid y) \propto \mathcal{N}(y \mid \mathcal{A}(x_0), \sigma_y^2 I) p_0(x_0)$, where $y = 0.3$kg is our target mass, $\sigma_y = 0.01$kg is the tolerance, and $\mathcal{A}(x_0)$ computes the mass of our mesh x_0. $p_0(x_0)$ is the distribution of the jet engine bracket designs from the "beam" category, which we approximate using our aforementioned LoRA fine-tuned TRELLIS.2 model conditioned on this class without CFG rescaling. We note that diffusion waltz does not apply directly to this setting as x_0 is generated in two-stages using TRELLIS.2. However, there is a straightforward modification that adapts to this setting, which we describe next.

RWMH waltz for TRELLIS.2 We now propose a slight extension of RWMH waltz that makes sampling from $p(x_0 \mid y)$ possible with TRELLIS.2's two-stage design. The crux is to construct an extended proposal $Q\big((x_0, z_0), (x_0', z_0')\big)$ such that it is reversible with respect to the joint prior $p_\theta(x_0, z_0) = p_\theta(x_0 \mid z_0) p_\theta(z_0)$. Thus, in this joint space, the MH acceptance ratio simplifies again, to a humble likelihood ratio

$$\frac{p(x_0', z_0' \mid y) Q\big((x_0', z_0'), (x_0, z_0)\big)}{p(x_0, z_0 \mid y) Q\big((x_0, z_0), (x_0', z_0')\big)} = \frac{p(y \mid x_0') p_\theta(x_0', z_0') \cancel{Q\big((x_0', z_0'), (x_0, z_0)\big)}}{p(y \mid x_0) p_\theta(x_0, z_0) \cancel{Q\big((x_0, z_0), (x_0', z_0')\big)}} = \frac{p(y \mid x_0')}{p(y \mid x_0)}. \tag{178}$$

The extended proposal $Q\big((x_0, z_0), (x_0', z_0')\big)$ that we use is constructed as follows (we note below that the prior $p_\theta(z_0)$ determining the SS is a flow matching model with linear interpolant schedules $\alpha_t = 1 - t$, $\beta_t = t$):

1. Add noise $z_t \sim (1-t)z_0 + tz_1$, where $z_1 \sim \mathcal{N}(0, I)$.
2. Denoise $z_0' \sim p_\theta(z_0' \mid z_t)$.
3. Sample $x_0' \sim p_\theta(x_0' \mid z_0')$.

We can easily demonstrate that this kernel is reversible with respect to our joint prior $p_\theta(x_0', z_0')$: denoting by $q(z_0, z_0')$ the SDEdit-style transition kernel in steps 1-2, we have

$$p_\theta(x_0, z_0)Q\big((x_0, z_0), (x_0', z_0')\big) = \Big(p_\theta(x_0 \mid z_0)p_\theta(z_0)\Big)\Big(p_\theta(x_0' \mid z_0')q(z_0, z_0')\Big) \tag{179}$$

$$= \Big(q(z_0, z_0')p_\theta(z_0)\Big)\Big(p_\theta(x_0' \mid z_0')p_\theta(x_0 \mid z_0)\Big) \tag{180}$$

$$= \Big(q(z_0', z_0)p_\theta(z_0')\Big)\Big(p_\theta(x_0' \mid z_0')p_\theta(x_0 \mid z_0)\Big) \tag{181}$$

$$= \Big(p_\theta(x_0' \mid z_0')p_\theta(z_0')\Big)\Big(p_\theta(x_0 \mid z_0)q(z_0', z_0)\Big) \tag{182}$$

$$= p_\theta(x_0', z_0')Q\big((x_0', z_0'), (x_0, z_0)\big), \tag{183}$$

where we used the reversibility of the SDEdit kernel $q(z_0, z_0')$ with respect to the prior $p_\theta(z_0)$ in the third line. Thus, reversibility holds and we may use this proposal alongside the MH acceptance probability $\min\{1, p(y \mid x_0')/p(y \mid x_0)\}$ to sample exactly from $p(x_0 \mid y)$.

Baselines. As this is a non-differentiable objective, we benchmark against EnKG Zheng et al. (2025a) and ESS-Flow Kalaivanan et al. (2025), as well as CFG (Ho & Salimans, 2022), which is commonly used in inverse design problems. Both EnKG and ESS-Flow update a state over the SS latent z_0, assuming a deterministic map to the target y. However, our forward model $\mathcal{A}(x_0)$ is defined on the second stage latent x_0, which is sampled from the first via $x_0 \sim p(x_0 \mid z_0)$. We therefore compute $\hat{x}_0 = \mathbb{E}[x_0 \mid z_0]$ using Tweedie's formula to obtain the expected mass for a given z_0, motivated by the fact that volume is dominated by the frequency structure captured by z_0. This gives reasonable mass values but low quality geometry, so we resample x_0 for the final states z_0 to obtain high quality samples, at the cost of introducing noise.

Method/Parameter	**Value**
Rejection Sampling	
Proposals	10,000
CFG	
Samples	500
Guidance scale w	2.5
EnKG	
Ensemble size	21
Num samples	10
Update t_{min}, t_{max}	0.05, 0.8
ζ	2
ESS-Flow	
MCMC steps	200
Num chains	4
Max shrinks	12
Burn-in	25
DW-RWMH	
MCMC steps	1000
Num chains	5
Burn-in	100
Noise step t	0.875

Table 11: Hyperparameters used in the 3D inverse design experiment.

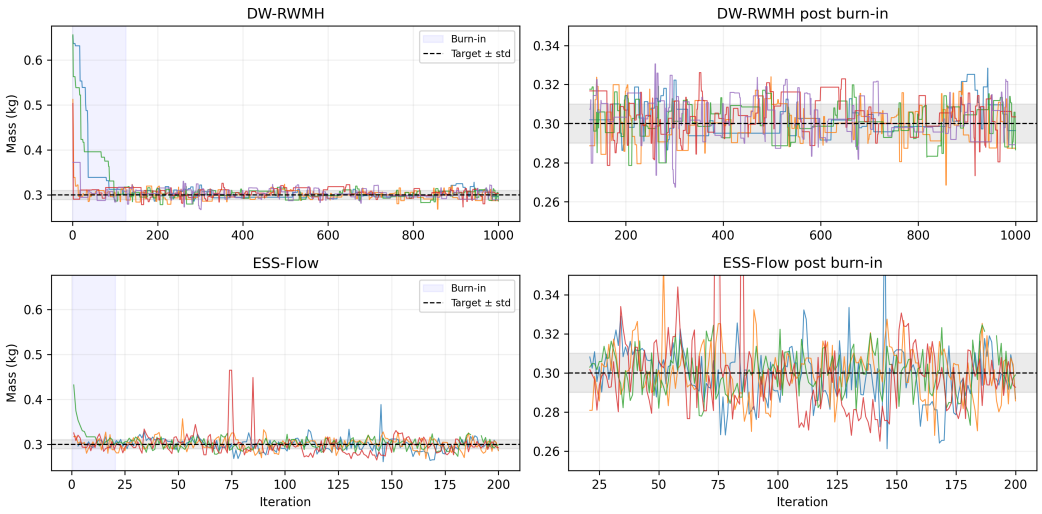


Figure 15: Trace plots for 3D inverse design.