SEQUENTIAL DATA ASSIMILATION USING ROLLING DIFFUSION MODELS WITH EDITABLE HISTORY

Anonymous authors
Paper under double-blind review

ABSTRACT

Data assimilation (DA) combines dynamical models with observations to estimate the hidden states of complex systems. Classical methods such as the ensemble Kalman filter struggle under highly non-Gaussian dynamics, while existing diffusion-based approaches are either fixed-window smoothers or bias-prone online filters. We introduce Rolling Diffusion Data Assimilation (RDA), which uses rolling diffusion models for forecasting, and assimilating data sequentially by guiding the denoising process with observations. Additionally, we propose a new rolling diffusion architecture with editable history blocks that can revise past frames as new observations arrive. For an idealized version with exact conditional denoising, our analysis identifies the limiting window posterior and bounds its discrepancy from the full filtering distribution under strong mixing. Across a range of nonlinear dynamical systems, from low-dimensional chaotic dynamics to high-dimensional geophysical settings, RDA matches or outperforms both classical and ML-based filtering methods, especially under sparse and nonlinear observations.

1 INTRODUCTION

Many systems across the sciences evolve through complex dynamics, yet their underlying sequential states $(\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots)$ can only be observed partially and imperfectly. Data assimilation (DA) combines model forecasts with observations to estimate these hidden states, making it a critical component in numerical weather prediction (Kalnay, 2002) and oceanography (Ghil & Malanotte-Rizzoli, 1991). Filtering, in particular, is the problem of assimilating observations $(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots)$ sequentially as they arrive, iterating between a forecast and an analysis step:

$$\text{(Forecast)} \qquad p(\boldsymbol{x}_{n+1} \mid \boldsymbol{y}_{1:n}) = \int p(\boldsymbol{x}_{n+1} \mid \boldsymbol{x}_n) p(\boldsymbol{x}_n \mid \boldsymbol{y}_{1:n}) \mathrm{d}\boldsymbol{x}_n$$

$$\text{(Analysis)} \qquad p(\boldsymbol{x}_{n+1} \mid \boldsymbol{y}_{1:n+1}) \propto p(\boldsymbol{y}_{n+1} \mid \boldsymbol{x}_{n+1}) p(\boldsymbol{x}_{n+1} \mid \boldsymbol{y}_{1:n})$$

For Gaussian forecast and observation models $p(\boldsymbol{x}_n \mid \boldsymbol{x}_{n-1}), p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$, the celebrated Kalman filter solves the filtering problem exactly (Kalman, 1960). For non-Gaussian systems, extensions of the Kalman filter exist that use explicit linearization or sigma-point methods (Julier & Uhlmann, 2004). Moreover, particle filters (PF) (Chopin & Papaspiliopoulos, 2020) provide an asymptotically exact solution under general assumptions. However, extended Kalman methods scale poorly to high dimensions due to computational intractability, and PF suffers from a well-known issue of weight collapse (Bengtsson et al., 2008). The ensemble Kalman filter (Evensen, 2003) and popular variants such as LETKF (Hunt et al., 2007) combines the benefits of particle and Kalman approaches, making it practical in high dimensions. Yet, its small ensemble size relative to the state dimension creates spurious correlations and underestimated variances, necessitating ad hoc inflation and localization, whose tuning remains challenging in highly non-Gaussian settings (Lei & Bickel, 2011).

To overcome these challenges, there has been increased interest in adopting diffusion-based generative modeling for DA. Score-based data assimilation (SDA) (Rozet & Louppe, 2023) uses a learned diffusion-based prior over a fixed window of frames, conditioning on observations via guidance (Chung et al., 2023; Rozet et al., 2024). While this has shown promise in large-scale applications (Andry et al., 2025; Manshausen et al., 2025), SDA is effectively a fixed-window smoothing method and does not adapt naturally to online filtering; it also incurs high memory and inference costs. Methods such as FlowDAS (Chen et al., 2025) instead propose DA methods native to generative models trained for forecasting, aligning with the recent success of diffusion-based weather

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

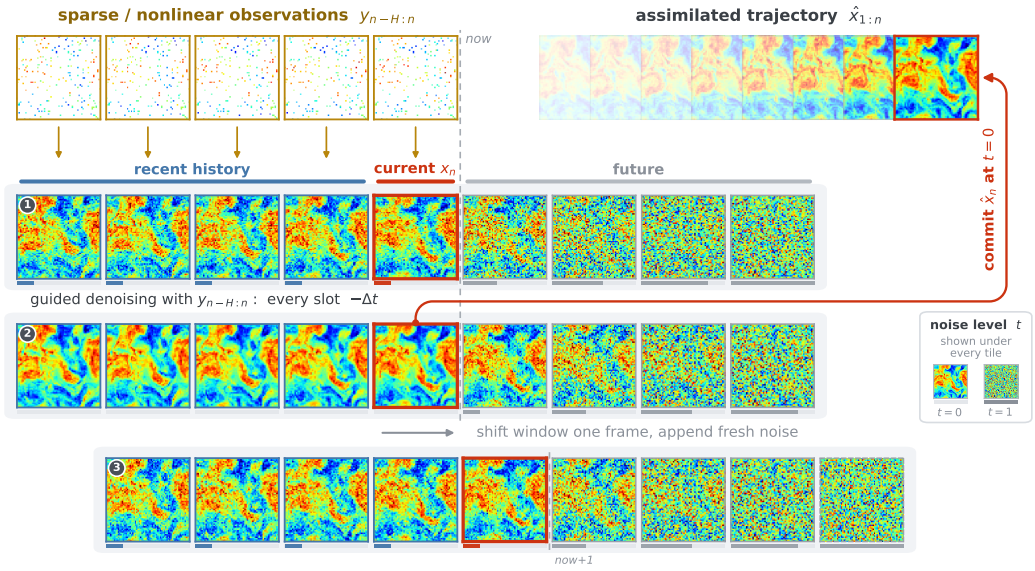


Figure 1: **Rolling diffusion data assimilation (RDA).** The bar under each frame indicates its noise level. **(1)** The rolling window consists of an editable recent history, the current frame $\boldsymbol{x}_n$, and future frames at increasing noise; observations $\boldsymbol{y}_{n-H:n}$ guide the window during denoising. **(2)** One cycle of guided denoising steps each frame down by Δt, bringing history and current frame to clean states; current estimate $\hat{\boldsymbol{x}}_n$ is committed to the assimilated trajectory (top right). **(3)** The window advances by one frame: a fresh-noise frame enters and the nearest future frame becomes the next current frame. Noise is added to the history frames, matching the noise level of the current frame.

models (Price et al., 2025; Kossaifi et al., 2026). However, FlowDAS learns to sample from the optimal proposal $p(\boldsymbol{x}_n \mid \boldsymbol{x}_{n-1}, \boldsymbol{y}_n)$ rather than the filtering distribution, leading to biased estimates, or require PF-style reweighing (Savary et al., 2026), risking weight collapse.

We address these issues by proposing a new method, **Rolling Diffusion Data Assimilation (RDA)**, based on rolling sequence diffusion models for generative forecasting (Ruhe et al., 2024; Rühling Cachay et al., 2025). Rolling diffusion generates forecasts by progressively denoising a window of frames with increasing noise levels toward the future. We perform assimilation by guiding this denoising process with observations, similar to SDA and FlowDAS; however, because the model operates on partially denoised windows rather than pure noise at every cycle, it requires fewer function evaluations (NFEs). Like FlowDAS, however, guiding rolling diffusion with only the current observation yields biased samples from the true filtering distribution. We address this by introducing **editable history blocks**, which allow past frames to be revised as new observations become available, yielding a more theoretically consistent approximation to the filtering distribution. We illustrate our full methodology in Figure 1.

2 Preliminaries

We review the background necessary for our work. For brevity, we use the shorthand $\boldsymbol{z}_{0:N}$ to denote a sequence of variables $(\boldsymbol{z}_0, \ldots, \boldsymbol{z}_N)$. We also use $n \in \{0, \ldots, N\}$ to index the (discrete) physical time and $t \in [0, 1]$ to denote the diffusion time.

2.1 Data Assimilation

Consider a discrete-time state-space model with latent states $\{\boldsymbol{x}_n\}_{n=0}^N$ and observations $\{\boldsymbol{y}_n\}_{n=1}^N$:

$$\boldsymbol{x}_n = \mathcal{F}(\boldsymbol{x}_{0:n-1}, \boldsymbol{\omega}_n), \qquad \boldsymbol{\omega}_n \sim p(\boldsymbol{\omega}), \qquad n = 1, \ldots, N, \tag{1}$$

$$\boldsymbol{y}_n = \mathcal{H}_n(\boldsymbol{x}_n) + \boldsymbol{\eta}_n, \qquad \boldsymbol{\eta}_n \sim p(\boldsymbol{\eta}), \qquad n = 1, \ldots, N, \tag{2}$$

with initial state $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$. Here, $\mathcal{F}$ denotes the dynamical evolution operator of the latent states and $\mathcal{H}_n$ is a possibly nonlinear observation operator that may vary with time n. The variables $\boldsymbol{\omega}_n$ and $\boldsymbol{\eta}_n$ represent the evolution operator's stochasticity and the observation noise, respectively, assumed mutually independent. We adopt the general form (1) to describe the state dynamics to accommodate rolling diffusion-based generative evolution operators $\mathcal{F}$, which we introduce in Section 3.1.

Given this setup, *data assimilation*, and in particular, *filtering*, seeks to combine the dynamical model (1) with observations (2) to recursively infer the posteriors $\pi_n := p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{1:n})$ on latent states conditioned on all observations available up to the current time n (Law et al., 2015). We also note the related problem of *smoothing*, which conditions the full latent trajectory on the entire observation sequence (i.e., $p(\boldsymbol{x}_{0:N} \mid \boldsymbol{y}_{1:N})$); in this work, we only consider the filtering problem.

2.2 Diffusion models and conditional sampling via guidance

Let $\boldsymbol{x}(0) \sim p(\boldsymbol{x}(0))$ be a sample from the data distribution. Continuous-time diffusion models (Song et al., 2021) progressively add noise to this clean sample by solving the linear SDE $\mathrm{d}\boldsymbol{x}(t) = f(t)\boldsymbol{x}(t)\,\mathrm{d}t + g(t)\,\mathrm{d}\boldsymbol{w}(t)$ forward ($t = 0 \to 1$), initialised at $\boldsymbol{x}(0)$, where $\boldsymbol{w}(t)$ is a standard Wiener process. Without loss of generality, we choose schedules $f(t), g(t)$ such that $\mathrm{Law}(\boldsymbol{x}(1)) \approx \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \boldsymbol{I})$ for some $\sigma_{\max} > 0$ (see Appendix A for details). Then, the SDE

$$\mathrm{d}\boldsymbol{x}(t) = \big[f(t)\boldsymbol{x}(t) - g(t)^2 \nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{x}(t)\big)\big]\,\mathrm{d}t + g(t)\,\mathrm{d}\overline{\boldsymbol{w}}(t), \tag{3}$$

solved backwards in time ($t = 1 \to 0$) and initialized randomly at $\boldsymbol{x}(1) \sim \mathcal{N}(\mathbf{0}, \sigma_{\max}^2 \boldsymbol{I})$, where $\overline{\boldsymbol{w}}(t)$ denotes the reverse-time Wiener process, generates samples from the data distribution $p(\boldsymbol{x}(0))$ Song et al. (2021). In practice, the score function $\nabla_{\boldsymbol{x}(t)} \log p(\boldsymbol{x}(t))$ appearing in equation 3 is intractable; thus it is approximated by a neural network $\boldsymbol{s}_\theta(\boldsymbol{x}(t), t)$ via *score matching* (see Appendix A). While the reverse-time SDE (3) samples from $p(\boldsymbol{x}(0))$ unconditionally, we are often interested in sampling from the conditional distribution $p(\boldsymbol{x}(0) \mid \boldsymbol{y})$, where $\boldsymbol{y} \sim p(\boldsymbol{y} \mid \boldsymbol{x}(0))$ is some observation of $\boldsymbol{x}(0)$. In principle, this can be achieved by replacing the score with the *conditional score*:

$$\nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{x}(t) \mid \boldsymbol{y}\big) = \nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{x}(t)\big) + \nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{y} \mid \boldsymbol{x}(t)\big). \tag{4}$$

The first term can be approximated by the score network $\boldsymbol{s}_\theta(\boldsymbol{x}(t), t)$, however, the second term (referred to as the *guidance term*) is still intractable as we usually only have access to $p(\boldsymbol{y} \mid \boldsymbol{x}(0))$, but not $p(\boldsymbol{y} \mid \boldsymbol{x}(t))$, i.e., the likelihood of $\boldsymbol{y}$ given noised states. Various methods exist to approximate this term, such as DPS Chung et al. (2023) and MMPS Rozet et al. (2024) (see Appendix B). Decoupling the likelihood score estimation from the prior score allows one to reuse the same score network $\boldsymbol{s}_\theta$ to sample from posteriors corresponding to different observations without retraining.

3 Method

We develop a data assimilation method in which the state dynamics (1) is given by a *rolling diffusion model* (Ruhe et al., 2024; Rühling Cachay et al., 2025). Rolling diffusion extends diffusion models to sequential generation, thus can be used naturally as a forecasting model for physical systems (e.g. weather). We first discuss our dynamical system given by rolling diffusion, followed by our methodology to assimilate observations sequentially with this model, and finally propose an extension with additional "history" blocks.

3.1 Rolling diffusion state dynamics

For ease of presentation, we describe the variance-exploding (VE) variant of rolling diffusion; other schedules follow similarly. Our experiments (Section 5) use a variance-preserving (VP) schedule with ϵ-prediction for Lorenz–63 and SQG (Appendix A) and a rectified schedule with clean-state prediction for ERA5 (Appendix E.4).

Rolling diffusion. Rolling diffusion (Ruhe et al., 2024; Rühling Cachay et al., 2025) extends diffusion models to sequential generation by associating noise levels with forecasting uncertainty: frames further into the future have more noise added than frames near the present. Concretely, for fixed $W > 0$, we consider a rolling window $\boldsymbol{X}_n := (\boldsymbol{x}_n, \ldots, \boldsymbol{x}_{n+W})$ of clean frames for $n \in \{0, 1, \ldots\}$.

Introducing an increasing sequence of noise levels $\sigma_0(t) < \cdots < \sigma_W(t)$ for $t \in [0,1]$, where each $\sigma_w(t)$ is monotonically increasing in $t \in [0,1]$, such that it satisfies

$$(i)\ \sigma_0(0) = 0, \quad (ii)\ \sigma_W(1) = \sigma_{\max}, \quad \text{and} \quad (iii)\ \sigma_{w+1}(0) = \sigma_w(1), \quad w = 0, \dots W-1, \tag{5}$$

we perturb each frame in $\boldsymbol{X}_n$ as $\boldsymbol{x}_{n+w}^{\sigma_w}(t) \sim \mathcal{N}(\boldsymbol{x}_{n+w}, \sigma_w^2(t)\boldsymbol{I})$, resulting in progressively noised windows $\tilde{\boldsymbol{X}}_n(t) := \big(\boldsymbol{x}_n^{\sigma_0}(t), \dots, \boldsymbol{x}_{n+W}^{\sigma_W}(t)\big)$ for $t \in [0,1]$. Rolling diffusion learns to generate samples $\tilde{\boldsymbol{X}}_n(0) \sim p\big(\tilde{\boldsymbol{X}}_n(0) \mid \tilde{\boldsymbol{X}}_n(1)\big)$ by solving a reverse-time SDE (3) with a learned score network $\boldsymbol{s}_\theta(\tilde{\boldsymbol{X}}_n(t), t)$, initialized at $\tilde{\boldsymbol{X}}_n(1)$ (see Appendix A for details). We note that in contrast to standard diffusion models, the "denoised" samples $\tilde{\boldsymbol{X}}_n(0)$ in rolling diffusion are still contaminated by noise; only the first frame is clean by property (i) in (5). Likewise, only the last frame in the starting sample $\tilde{\boldsymbol{X}}_n(1)$ is pure noise. Thus, rolling diffusion generalizes diffusion models by learning to denoise a staircase-noised window, advancing each frame by exactly one noise level per pass. Diffusion Forcing (Chen et al., 2024; Song et al., 2025) generalizes this further by learning to denoise arbitrary per-frame noise levels.

Roll-out state dynamics. An attractive property of the staircase noising pattern is that it allows for indefinite roll out: after denoising, we commit the leading clean frame $\boldsymbol{x}_n = \boldsymbol{x}_n^{\sigma_0}(0)$ as output, the rest are shifted forward, and a fresh frame $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma_{\max}^2 \boldsymbol{I})$ is appended, giving the next input

$$\tilde{\boldsymbol{X}}_{n+1}(1) := \big(\boldsymbol{x}_{n+1}^{\sigma_0}(1), \dots, \boldsymbol{x}_{(n+1)+W}^{\sigma_W}(1)\big) \leftarrow \big(\boldsymbol{x}_{n+1}^{\sigma_1}(0), \dots, \boldsymbol{x}_{n+W}^{\sigma_W}(0), \boldsymbol{\epsilon}\big). \tag{6}$$

Condition (iii) in (5) ensures that each shifted frame enters at the noise level its new position expects. In our context, the stochastic dynamics $\mathcal{F}$ in equation 1 is described by the sequence of committed frames through this rolling mechanism, with stochasticity present due to the inherent randomness of the reverse SDE during denoising. To initialize our dynamics, we sample an initial state $\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0)$, copy it across $W+1$ frames, and add noise accordingly to yield $\tilde{\boldsymbol{X}}_0(1) := \big(\boldsymbol{x}_0^{\sigma_0}(1), \dots, \boldsymbol{x}_0^{\sigma_W}(1)\big)$.

Assimilation method. Denoting the learned rolling diffusion score by $\boldsymbol{s}_\theta(\tilde{\boldsymbol{X}}_n(t), t)$, with the corresponding generative model denoted by $p_\theta(\tilde{\boldsymbol{X}}_n(t) \mid \tilde{\boldsymbol{X}}_n(1))$, we propose to assimilate observations $\boldsymbol{y}_1, \boldsymbol{y}_2, \dots$ sequentially, by sampling

$$\tilde{\boldsymbol{X}}_n(0) \sim p_\theta\big(\tilde{\boldsymbol{X}}_n(0) \mid \tilde{\boldsymbol{X}}_n(1), \boldsymbol{y}_n\big) \propto p(\boldsymbol{y}_n \mid \tilde{\boldsymbol{X}}_n(0))\, p_\theta(\tilde{\boldsymbol{X}}_n(0) \mid \tilde{\boldsymbol{X}}_n(1)), \tag{7}$$

rolling-out the next input $\tilde{\boldsymbol{X}}_n(0) \to \tilde{\boldsymbol{X}}_{n+1}(1)$ and repeating (7) with $n \leftarrow n+1$. We note that since $\boldsymbol{y}_n$ only depends on the clean leading frame $\boldsymbol{x}_n = \boldsymbol{x}_n^{\sigma_0}(0)$ of the window $\tilde{\boldsymbol{X}}_n(0)$ and the rest of the frames are conditionally independent given this clean frame, the likelihood effectively reads $p(\boldsymbol{y}_n \mid \tilde{\boldsymbol{X}}_n(0)) = p(\boldsymbol{y}_n \mid \boldsymbol{x}_n)$. In practice, to sample from the posterior (7), we employ inference-time guidance methods, such as DPS and MMPS (see Section 2.2). In addition, to represent the full state distribution rather than a single point estimate, we run J independent copies of this procedure in parallel, each with its own initial guess $\boldsymbol{x}_0^{(j)} \sim p(\boldsymbol{x}_0)$ for $j = 1, \dots, J$, giving an ensemble of particles $\{\boldsymbol{x}_n^{(1)}, \dots, \boldsymbol{x}_n^{(J)}\}$ at each step n.

3.2 Extending rolling diffusion with editable history frames

The data assimilation procedure introduced above conditions on the current observation $\boldsymbol{y}_n$ at each assimilation step. However in practice, we find that conditioning on $\boldsymbol{y}_n$ alone leads to unsatisfactory results. This can be attributed to the following theoretical gap had we only conditioned on $\boldsymbol{y}_n$ at each assimilation step.

Theoretical gap. Marginalizing out the future frames, equation 7 can be interpreted as sampling $p(\boldsymbol{x}_n \mid \boldsymbol{x}_{0:n-1}, \boldsymbol{y}_n) \propto p(\boldsymbol{y}_n \mid \boldsymbol{x}_n) p(\boldsymbol{x}_n \mid \boldsymbol{x}_{0:n-1})$, where $p(\boldsymbol{x}_n \mid \boldsymbol{x}_{0:n-1}) = \mathrm{Law}(\mathcal{F}(\boldsymbol{x}_{0:n-1}, \cdot))$ is the transition probability of the state dynamics (1). However, the true filtering distribution reads

$$p(\boldsymbol{x}_n \mid \boldsymbol{y}_{1:n}) = \int p(\boldsymbol{x}_n \mid \boldsymbol{x}_{0:n-1}, \boldsymbol{y}_n) p(\boldsymbol{x}_{0:n-1} \mid \boldsymbol{y}_{1:n}) \mathrm{d}\boldsymbol{x}_{0:n-1}. \tag{8}$$

Hence, for an ensemble $\boldsymbol{x}_n^{(j)} \sim p\big(\boldsymbol{x}_n \mid \boldsymbol{x}_{0:n-1}^{(j)}, \boldsymbol{y}_n\big), j = 1, \dots, J$ to represent (8) correctly, the previously committed frames $\boldsymbol{x}_{0:n-1}^{(j)}$ must be samples of the *smoothing distribution* $p(\boldsymbol{x}_{0:n-1} \mid \boldsymbol{y}_{1:n})$,

which require adjustments in light of observations that arrived after they were originally filtered. The procedure of the previous section does not take these adjustments into account, which lead to deviations from the true filtering distribution.

Rolling diffusion with history frames. Motivated by this, we extend the rolling diffusion model with additional *history frames*, which are edited at each assimilation step. For a current frame $\boldsymbol{x}_n$ and history length $H \in \{1, \dots, n\}$, we consider the augmented window

$$\boldsymbol{X}'_n = \big(\underbrace{\boldsymbol{x}_{n-H}, \dots, \boldsymbol{x}_{n-1}}_{\text{recent history}}, \underbrace{\boldsymbol{x}_n}_{\text{current frame}}, \underbrace{\boldsymbol{x}_{n+1}, \dots, \boldsymbol{x}_{n+W}}_{\text{future frames}} \big). \tag{9}$$

Taking the noise levels $\sigma_0(t) < \cdots < \sigma_W(t)$ introduced in Section 3.1, we produce the noised windows $\tilde{\boldsymbol{X}}'_n(t) = \big(\boldsymbol{x}^{\sigma_0}_{n-H:n}(t), \boldsymbol{x}^{\sigma_1}_{n+1}(t), \dots, \boldsymbol{x}^{\sigma_W}_{n+W}(t)\big)$, where we recall the notation for perturbed frames: $\boldsymbol{x}^{\sigma_w}_{n+w}(t) \sim \mathcal{N}(\boldsymbol{x}_{n+w}, \sigma_w^2(t)\boldsymbol{I})$ and the shorthand $\boldsymbol{x}^{\sigma_0}_{n-H:n}(t)$ indicates that each frame in $\boldsymbol{x}_{n-H:n}$ is noised by the same level $\sigma_0(t)$ independently, i.e., the history frames are noised by the same amount as the current frame. The rolling diffusion in this construction then learns to generate $\tilde{\boldsymbol{X}}'_n(0) \sim p\big(\tilde{\boldsymbol{X}}'_n(0) \mid \tilde{\boldsymbol{X}}'_n(1)\big)$ as before, except that it now re-generates history. To roll out, only the current frame $\boldsymbol{x}_n = \boldsymbol{x}^{\sigma_0}_n(0)$ is committed, the window advances by one, and a fresh noise frame is appended at the end. The history block $\boldsymbol{x}_{n-H:n-1}$ is re-noised to level $\sigma_0(1)$ at the start of each iteration rather than being dropped, to enable editing during assimilation.

Assimilation method. We now propose a slight extension of the assimilation method presented earlier, where we condition on a window of past and present observations $\boldsymbol{y}_{n-H:n}$, sampling

$$\tilde{\boldsymbol{X}}'_n(0) \sim p_\theta\big(\tilde{\boldsymbol{X}}'_n(0) \mid \tilde{\boldsymbol{X}}'_n(1), \boldsymbol{y}_{n-H:n}\big) \propto \prod_{h=0}^{H} p(\boldsymbol{y}_{n-h} \mid \boldsymbol{x}_{n-h})\, p_\theta(\tilde{\boldsymbol{X}}'_n(0) \mid \tilde{\boldsymbol{X}}'_n(1)), \tag{10}$$

again realized in practice by guidance (e.g., DPS, MMPS). We may additionally iterate over (i) re-noising $\tilde{\boldsymbol{X}}'_n(0) \mapsto \tilde{\boldsymbol{X}}'_n(1)$ and (ii) denoising conditionally $\tilde{\boldsymbol{X}}'_n(1) \mapsto \tilde{\boldsymbol{X}}'_n(0)$ via equation 10 K times. The following result gives an asymptotic filtering interpretation under idealized assumptions.

Theorem 1. *Under the assumptions of Theorem 3 (appendix), let $\widehat{\boldsymbol{x}}^{[K]}_n$ denote the current-frame sample committed after K re-noising and denoising passes at time $n \in \{n_0, \dots, N\}$, where $n_0 > H$. Then, there exists $\rho_N \in (0, 1)$, depending only on the physical model and the observation record $\boldsymbol{y}_{1:N}$, but not on K, n, H, or W, such that*

$$\lim_{K \to \infty} \Big\| \mathrm{Law}(\widehat{\boldsymbol{x}}^{[K]}_n) - \pi_n \Big\|_{\mathrm{TV}} \le 2\rho_N^H, \tag{11}$$

where π_n denotes the true filtering distribution at time n and $\|\cdot\|_{\mathrm{TV}}$ denotes the total variation distance of probability measures. The limiting error bound (for $K \to \infty$) therefore decreases exponentially with the history length H.

Proof. The precise statement and proof are given in Appendix D. □

The limiting current-frame distribution is $p(\boldsymbol{x}_n \mid \boldsymbol{y}_{n-H:n})$, so the bound concerns the error from truncating the observation history. Appendix D.6 discusses the relation between this limit and the practical implementation. We use $K = 1$ with tuned guidance in the main experiments and examine larger K in Figure 2c.

We summarize our full data assimilation procedure using rolling diffusion with editable history blocks in Algorithm 1 (Appendix), referred to as **Rolling Diffusion Data Assimilation (RDA)**. Crucially, history editing is only used to produce an estimate of the current frame $\boldsymbol{x}_n$ and *does not overwrite the previously committed frames* $\boldsymbol{x}_{n-H:n-1}$. This is so that the filtered $\boldsymbol{x}_m$ for $m < n$ remains computed using only observations available up to time m.

We may also implement the RDA procedure using the *ForcingDAS* framework (Jia et al., 2026), which we refer to as **ForcingDAS-Rolling**. Since Diffusion Forcing supports arbitrary inference schedules without retraining (Chen et al., 2024), we may use the RDA noise schedule in ForcingDAS in place of the autoregressive filtering schedule that was originally proposed. ForcingDAS-Rolling therefore retains ForcingDAS' training procedure, changing only its inference schedule.

4 Related Works

Machine learning methods have been used in data assimilation, both to learn the analysis map (McCabe & Brown, 2021; Chattopadhyay et al., 2023; Bach et al., 2026b;a), and to construct generative priors. Following the latter approach, Score-based Data Assimilation (SDA) (Rozet & Louppe, 2023; Andry et al., 2025) uses diffusion guidance over fixed-length trajectories to sample from the *smoothing distribution* $p(\boldsymbol{x}_{0:N} \mid \boldsymbol{y}_{1:N})$. While this has been applied to large-scale problems (Manshausen et al., 2025), adapting it to online filtering remains challenging. The Ensemble Score Filter (EnSF) and related approaches (Bao et al., 2024b;a; Si & Chen, 2025; Transue et al., 2026) instead constructs a score approximation directly from the forecast ensemble, providing a training-free filtering method. However unlike SDA, the training-free prior in EnSF is effectively a mixture of point mass around the forecasts, making it prone to particle collapse. FlowDAS learns a conditional diffusion-based forecast model and performs guided sampling at each assimilation time (Chen et al., 2025); however, it suffers from the same theoretical gap that we highlighted in Section 3.2. DiffDA (Huang et al., 2024) proposes a similar coupling of the forecast and analysis steps via a GraphCast (Lam et al., 2023)-based forecast model. ForcingDAS extends FlowDAS by using frame-independent noise schedules (Jia et al., 2026; Chen et al., 2024) allowing one to perform autoregressive (AR) filtering or fixed-lag and full-sequence smoothing. However, the theoretical gap persists in their AR filtering method. FA-APF (Savary et al., 2026) addresses this by correcting the error through particle reweighting, although this makes it prone to particle collapse. DAISI (Andrae et al., 2026) on the other hand overcomes the gap by decoupling the learned prior from the forecast model, combining them via an inverse sampling procedure; however, along the way, they introduce a separate irreducible bias.

5 Experiments

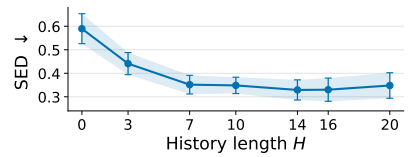
We evaluated rolling diffusion DA on three datasets: the Lorenz '63 (L63) system with squared observation operator Lorenz (1963), a Surface Quasi-Geostrophic system (SQG) Tulloch & Smith (2009) and ERA5 (Hersbach et al., 2020).

5.1 Lorenz–63 squared observations

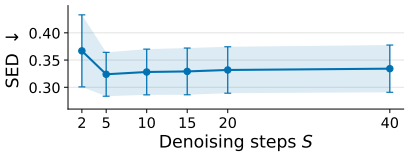
We evaluate RDA on the Lorenz–63 (L63) system with nonlinear squared observations, $y_n = x_{n,1}^2 + \varepsilon_n$, where $\sigma_{\text{obs}} = 17$ in physical units. Since this operator cannot distinguish the sign of $x_{n,1}$, the resulting filtering distributions can be multimodal when the initial distribution is diffuse. We consider observations at every time step (stride 1) and every six time steps (stride 6). We approximate the reference filtering distribution using a bootstrap particle filter (Gordon et al., 1993) with 10^5 particles and evaluate SED and ES (Appendix E.1). All methods use $J = 512$ ensemble members and are evaluated on nine held-out trajectories of length 1000, discarding the first 100 steps. Additional experimental and baseline initialization details are provided in Appendix E.2.

Table 1: Lorenz–63 squared-observation results. All methods use $J = 512$ ensemble members. Metrics are averaged over nine held-out trajectories after discarding the first 100 time steps.

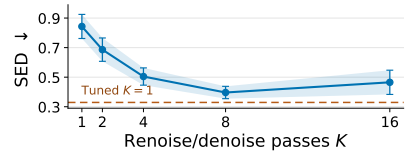
	High frequency, stride $= 1$		Low frequency, stride $= 6$	
Method	SED	ES	SED	ES
EnKF	1.70 ± 0.31	9.19 ± 4.46	1.58 ± 0.21	8.56 ± 2.96
ESRF	2.07 ± 0.23	9.41 ± 3.60	1.87 ± 0.18	9.50 ± 2.83
EnSF	2.75 ± 0.06	11.02 ± 1.13	2.37 ± 0.09	9.61 ± 0.88
DAISI	0.58 ± 0.03	6.21 ± 0.13	$\underline{0.48 \pm 0.07}$	6.56 ± 0.19
FlowDAS	0.81 ± 0.20	5.77 ± 1.16	1.55 ± 0.17	8.50 ± 1.02
ForcingDAS-AR	3.08 ± 0.11	11.35 ± 1.14	3.29 ± 0.13	11.98 ± 1.11
SDA (filtering)	$\underline{0.32 \pm 0.03}$	$\underline{5.74 \pm 0.19}$	1.84 ± 0.09	8.07 ± 0.09
ForcingDAS-Rolling	0.45 ± 0.04	6.20 ± 0.59	0.58 ± 0.05	$\underline{6.53 \pm 0.62}$
RDA	$\mathbf{0.29 \pm 0.06}$	$\mathbf{5.69 \pm 0.20}$	$\mathbf{0.33 \pm 0.04}$	$\mathbf{6.19 \pm 0.27}$



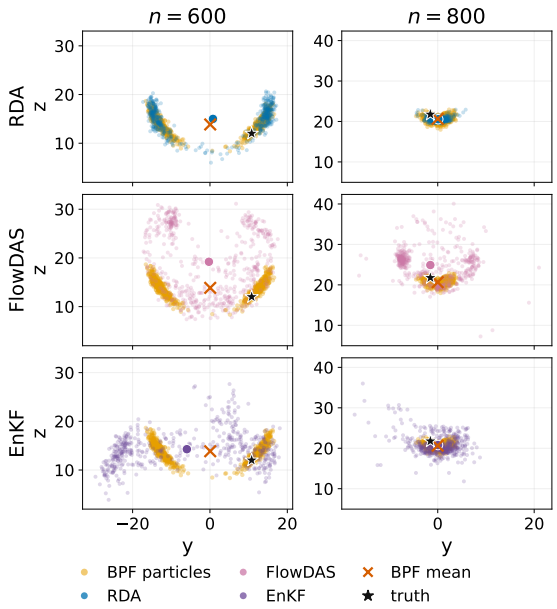
(a) History-length ablation.



(b) Denoising-step ablation.



(c) Renoise/denoise refinement.



(d) Posterior comparison against the BPF reference.

Figure 2: Lorenz–63 results with low-frequency squared observations (stride $= 6$). **(a)** Effect of history length. **(b)** Effect of denoising steps with $H = 14$. **(c)** With $H = 14$ and unit guidance strength, mean SED decreases as K increases from 1 to 8, but increases at $K = 16$. The dashed line indicates the tuned $K = 1$ result used in the main comparison. Ablation curves report mean SED with standard-deviation error bars and shading over nine held-out trajectories. **(d)** Posterior ensembles for RDA ($H = 14$), FlowDAS, and EnKF on sample 4 at $n = 600, 800$, compared with BPF particles in the (y, z) projection. Orange crosses mark BPF means; black stars mark the true states.

Table 1 summarizes the filtering results under both observation frequencies. RDA achieves the lowest mean SED and ES in both observation settings, combining accurate state estimation with close agreement to the reference posterior. ForcingDAS-Rolling also consistently outperforms ForcingDAS-AR, supporting the effectiveness of our rolling filtering construction across different diffusion priors. Figure 2d visualizes representative filtering posteriors for the low-frequency squared-observation case. Each row compares one method against the BPF reference posterior in the (y, z) projection at two time points. RDA tracks the BPF support more closely than FlowDAS and EnKF, consistent with the lower SED in Table 1.

Effect of history length. Our filtering interpretation motivates retaining recent observations: Lemma 1 (Appendix D.4) bounds the error from truncating the observation history by a quantity that decreases with H under strong mixing. Consistent with this motivation, increasing history substantially improves posterior approximation, with diminishing gains as the history length increases.

Number of denoising steps We examine whether rolling filtering requires a large number of denoising steps by varying it for evaluation. SED and ES improve when increasing the number of denoising steps from 2 to 5. Metrics change little between 5 and 40 denoising steps, whereas evaluation time increases substantially. Thus, a small number of denoising steps is sufficient for accurate filtering.

Number of renoise/denoise cycles. With MMPS guidance strength fixed to one, increasing K from 1 to 8 reduces mean SED and ES, while both increase at $K = 16$. The tuned $K = 1$ configuration achieves lower mean SED and ES than all unit-guidance configurations in this ablation, without additional passes.

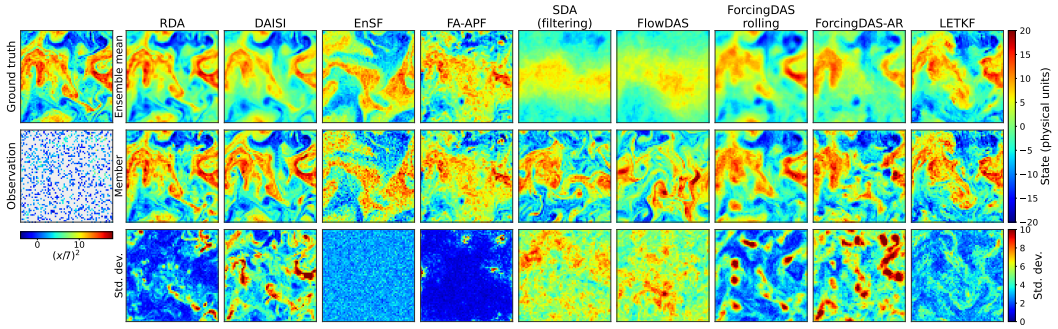


Figure 3: Qualitative comparison for the multimodal SQG experiment on a held-out trajectory at the final assimilation time.

5.2 Surface Quasi-Geostrophic (SQG) System

Next, we evaluate DA on a two-layer surface quasi-geostrophic (SQG) system with two channels on a 64×64 grid over 100 time steps with 3-hour intervals. We consider three observation settings used in Andrae et al. (2026): The *sparse* setting observes 5% of spatial locations using $h(x) = x$ with $\sigma_{\text{obs}} = 1$. The *multimodal* setting observes 25% using $h(x) = (x/7)^2$ with $\sigma_{\text{obs}} = 1$, while the *saturating* setting observes 25% using $h(x) = \arctan(x)$ with $\sigma_{\text{obs}} = 0.01$. In each setting, σ_{obs} is the standard deviation of Gaussian noise added after applying h. All methods uses 20 ensemble members and the same observation masks and noisy observations. We evaluate 10 held-out trajectories of 100 assimilation steps each, computing metrics over the final 20 steps. We report root mean squared error (RMSE) and continuous ranked probability score (CRPS) in physical units. We refer the readers to Appendix E.3 for exact details on DA setup, baselines, initialization and hyperparameter tuning.

Table 2: CRPS on ten held-out SQG trajectories, averaged over the final 20 assimilation steps. All methods use 20 ensemble members and assimilate the same saved observations. Metrics are reported in physical units. The best result for each setting is shown in bold and the second best is underlined.

Setting	RDA (ours)	LETKF	SDA (filtering)	EnSF	FA-APF	DAISI	FlowDAS	ForcingDAS AR	ForcingDAS Rolling
`Sparse`	**1.60 ± 0.18**	2.18 ± 0.31	1.93 ± 0.17	4.45 ± 0.52	4.29 ± 0.31	$\underline{1.64 \pm 0.17}$	3.35 ± 0.37	2.40 ± 0.22	2.29 ± 0.30
`Multimodal`	**1.61 ± 0.20**	3.07 ± 0.70	3.56 ± 0.30	5.82 ± 0.83	5.13 ± 0.48	$\underline{1.69 \pm 0.21}$	4.47 ± 0.36	2.48 ± 0.19	2.23 ± 0.31
`Saturating`	**1.02 ± 0.14**	5.23 ± 0.21	3.72 ± 0.26	$\underline{1.35 \pm 0.17}$	3.46 ± 0.46	1.57 ± 0.14	4.19 ± 0.54	2.11 ± 0.14	1.95 ± 0.18

Table 2 reports CRPS across the three SQG observation settings. RDA achieves the lowest CRPS in all three settings. Its advantage is most pronounced under saturating observations, while it remains the strongest and achieves lower CRPS than the best-performing baseline DAISI under sparse and multimodal observations. Figure 3 provides a qualitative comparison of ensemble means, individual samples, and ensemble standard deviations under multimodal observations. Together, these results demonstrate RDA's effectiveness across distinct observation challenges, including spatial sparsity, sign ambiguity, and saturation. ForcingDAS-Rolling improves upon ForcingDAS-AR in all three settings, supporting the benefit of RDA's DA framework. RDA combines this accuracy with substantial computational savings, achieving approximately 3.9–5.6$\times$ speedups over DAISI and 37–53$\times$ speedups over FlowDAS across the three observation settings (Table 26).

5.3 ERA5

We evaluate whether RDA can enable online filtering at atmospheric scale while retaining much of the reconstruction accuracy of APPA's offline smoothing. We use six-hourly fields on a 121×240 latitude–longitude grid with 71 channels. Assimilation uses synthetic noisy observations with spatial coverage based on weather station locations and satellite scans. Following APPA (Andry et al., 2025), we perform diffusion-based assimilation in an autoencoder's latent space. We evaluate ten-member ensembles on four held-out trajectories of 96 snapshots from 2021. Further details are

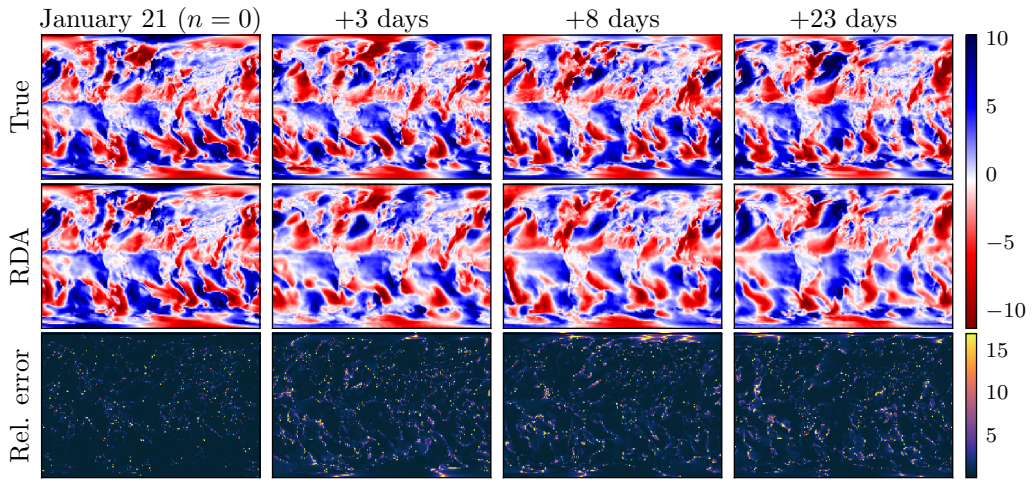


Figure 4: RDA snapshots of 10 m V-wind [m/s] and absolute relative error.

Table 3: End-of-trajectory ERA5 results with ten ensemble members on four held-out 96-snapshot trajectories. We report mean ± standard deviation across trajectories. APPA performs smoothing and RDA performs filtering.

	10 m U-wind [m/s]		10 m V-wind [m/s]		2 m temperature [K]		850 hPa temperature [K]	
Method	RMSE	CRPS	RMSE	CRPS	RMSE	CRPS	RMSE	CRPS
Climatology	4.05 ± 0.06	1.99 ± 0.05	4.05 ± 0.32	2.00 ± 0.17	3.31 ± 0.37	1.47 ± 0.14	3.97 ± 0.43	2.00 ± 0.19
APPA forecast	4.13 ± 0.15	2.02 ± 0.05	4.07 ± 0.25	1.99 ± 0.11	3.40 ± 0.57	1.49 ± 0.19	3.94 ± 0.56	1.99 ± 0.25
RDA forecast	4.53 ± 0.33	2.27 ± 0.16	4.17 ± 0.20	2.06 ± 0.08	4.07 ± 0.36	1.85 ± 0.19	4.49 ± 0.31	2.32 ± 0.21
APPA	2.18 ± 0.15	0.98 ± 0.07	2.12 ± 0.16	0.97 ± 0.05	1.68 ± 0.01	0.78 ± 0.03	1.94 ± 0.05	0.97 ± 0.04
RDA	2.34 ± 0.27	1.05 ± 0.10	2.11 ± 0.27	0.96 ± 0.09	1.79 ± 0.33	0.82 ± 0.15	2.20 ± 0.26	1.07 ± 0.10

provided in Appendix E.4. APPA provides a strong smoothing reference, reconstructing trajectories using observations both before and after each estimated state. Adapting this trajectory-level procedure to online filtering has challenges. A straightforward adaptation repeatedly runs full-trajectory denoising whenever new observations are collected, but this incurs quadratic computational complexity with respect to trajectory length. In contrast, RDA carries forward a partially denoised window and incrementally assimilates observations, producing the current analysis without waiting for future observations or restarting full-window generation from noise. Table 3 reports end-of-trajectory reconstruction accuracy. RDA reduces CRPS approximately 50% relative to unguided RDA forecasting. In addition, RDA's performance is close to APPA's offline smoothing results. RDA's mean CRPS is only about 5–10% higher for 10 m U-wind and the temperatures, and its slightly lower for 10 m V-wind. These results demonstrate that RDA enables online filtering at atmospheric scale, delivering analyses sequentially while achieving final-state accuracy close to that of an offline smoother.

6 Conclusion

We introduced Rolling Diffusion Data Assimilation (RDA), which guides rolling diffusion forecasting models with observations at each step, requiring fewer function evaluations compared to SDA while supporting indefinite roll-out. Editable history blocks allow recent states to be revised as new observations arrive. Under the strong mixing assumption, our idealized analysis gives an exponentially decreasing bound in H on the observation-truncation error. Experiments on L63, SQG, and ERA5 demonstrate that RDA matches or outperforms classical and ML-based filters. Current limitations include (1) the need to train a dedicated forecaster rather than reuse an external model, e.g., GenCast; (2) difficulty of tuning the history/window lengths as these are fixed during training. ForcingDAS-Rolling partly resolves the latter at the cost of lower accuracy. Addressing these is a promising direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, assisting with the writing of mathematical proofs, and providing feedback on the methodology and experiments.

We have not used generative AI tools for proposing hypotheses or developing the core conceptual ideas, and the use of generative AI tools for generating synthetic datasets, providing translations, or reformatting datasets is not applicable to this work.

Additionally, we used generative AI tools for assisting with figure creation, supporting the writing process, and helping to identify 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 and arguments were carefully checked through proofreading and consultation of 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

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

Martin Andrae, Erik Wikingsson, So Takao, Tomas Landelius, and Fredrik Lindsten. DAISI: Data assimilation with inverse sampling using stochastic interpolants. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2512.00252.

Gérôme Andry, Sacha Lewin, François Rozet, Omer Rochman, Victor Mangeleer, Matthias Pirlet, Elise Faulx, Marilaure Grégoire, and Gilles Louppe. Appa: Bending weather dynamics with latent diffusion models for global data assimilation. In *Machine Learning and the Physical Sciences Workshop at NeurIPS*, 2025. URL https://arxiv.org/abs/2504.18720.

Eviatar Bach, Ricardo Baptista, Jochen Bröcker, Bohan Chen, and Andrew Stuart. Learning probabilistic filters with strictly proper scoring rules. *arXiv preprint arXiv:2606.26497*, 2026a. URL https://arxiv.org/abs/2606.26497.

Eviatar Bach, Ricardo Baptista, Edoardo Calvello, Bohan Chen, and Andrew Stuart. Learning enhanced ensemble filters. *Journal of Computational Physics*, 547:114550, 2026b. doi: 10.1016/j.jcp.2025.114550.

Feng Bao, Zezhong Zhang, and Guannan Zhang. An ensemble score filter for tracking high-dimensional nonlinear dynamical systems. *Computer Methods in Applied Mechanics and Engineering*, 432:117447, 2024a. doi: 10.1016/j.cma.2024.117447.

Feng Bao, Zezhong Zhang, and Guannan Zhang. A score-based filter for nonlinear data assimilation. *Journal of Computational Physics*, 514, 2024b. doi: 10.1016/j.jcp.2024.113207. URL https://doi.org/10.1016/j.jcp.2024.113207.

Thomas Bengtsson, Peter Bickel, and Bo Li. Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In *Probability and Statistics: Essays in Honor of David A. Freedman*, volume 2 of *IMS Collections*, pp. 316–334. Institute of Mathematical Statistics, 2008. doi: 10.1214/193940307000000518.

Olivier Cappé, Eric Moulines, and Tobias Rydén. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer, New York, 2005. doi: 10.1007/0-387-28982-8.

Ashesh Chattopadhyay, Ebrahim Nabizadeh, Eviatar Bach, and Pedram Hassanzadeh. Deep learning-enhanced ensemble-based data assimilation for high-dimensional nonlinear dynamical systems. *Journal of Computational Physics*, 477:111918, 2023. doi: 10.1016/j.jcp.2023.111918.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion Forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems*, volume 37, pp. 24081–24125, 2024. doi: 10.52202/079017-0759.

Siyi Chen, Yixuan Jia, Qing Qu, He Sun, and Jeffrey A. Fessler. FlowDAS: A stochastic interpolant-based framework for data assimilation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 114163–114204, 2025. doi: 10.52202/085713-3445.

Nicolas Chopin and Omiros Papaspiliopoulos. *An Introduction to Sequential Monte Carlo*. Springer Series in Statistics. Springer, Cham, 2020. doi: 10.1007/978-3-030-47845-2.

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. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. *Journal of Geophysical Research: Oceans*, 99(C5): 10143–10162, 1994.

Geir Evensen. The ensemble Kalman filter: Theoretical formulation and practical implementation. *Ocean Dynamics*, 53:343–367, 2003. doi: 10.1007/s10236-003-0036-9.

Christopher AT Ferro. Fair scores for ensemble forecasts. *Quarterly Journal of the Royal Meteorological Society*, 140(683):1917–1923, 2014.

Michael Ghil and Paola Malanotte-Rizzoli. Data assimilation in meteorology and oceanography. In *Advances in Geophysics*, volume 33, pp. 141–266. Academic Press, 1991. doi: 10.1016/S0065-2687(08)60442-2.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE proceedings F (radar and signal processing)*, volume 140, pp. 107–113. IET, 1993.

Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, et al. The era5 global reanalysis. *Quarterly journal of the royal meteorological society*, 146(730):1999–2049, 2020. doi: 10.1002/qj.3803.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Langwen Huang, Lukas Gianinazzi, Yuejiang Yu, Peter Dominik Dueben, and Torsten Hoefler. DiffDA: A diffusion model for weather-scale data assimilation. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 19798–19815. PMLR, 2024. URL https://proceedings.mlr.press/v235/huang24h.html.

Brian R. Hunt, Eric J. Kostelich, and Istvan Szunyogh. Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Physica D: Nonlinear Phenomena*, 230(1-2): 112–126, 2007. doi: 10.1016/j.physd.2006.11.008.

Yixuan Jia, Siyi Chen, Yida Pan, Xiao Li, Lianghe Shi, Chanyong Jung, Haijie Yuan, Ismail Alkhouri, Yue Cynthia Wu, Saiprasad Ravishankar, Jeffrey A. Fessler, and Qing Qu. ForcingDAS: Unified and robust data assimilation via Diffusion Forcing. *arXiv preprint arXiv:2605.14285*, 2026. URL https://arxiv.org/abs/2605.14285.

Simon J. Julier and Jeffrey K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, 2004. doi: 10.1109/JPROC.2003.823141.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. doi: 10.1115/1.3662552.

Eugenia Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, Cambridge, 2002. doi: 10.1017/CBO9780511802270.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

Jean Kossaifi, Nikola Kovachki, Morteza Mardani, Daniel Leibovici, Suman Ravuri, Ira Shokar, Edoardo Calvello, Mohammad Shoaib Abbas, Peter Harrington, Ashay Subramaniam, Noah Brenowitz, Boris Bonev, Wonmin Byeon, Karsten Kreis, Dale Durran, Arash Vahdat, Mike Pritchard, and Jan Kautz. Demystifying data-driven probabilistic medium-range weather forecasting. *arXiv preprint arXiv:2601.18111*, 2026. URL https://arxiv.org/abs/2601.18111.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Oriol Vinyals, Jacklynn Stott, Alexander Pritzel, Shakir Mohamed, and Peter Battaglia. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023. doi: 10.1126/science.adi2336.

Kody Law, Andrew Stuart, and Konstantinos Zygalakis. *Data Assimilation: A Mathematical Introduction*, volume 62 of *Texts in Applied Mathematics*. Springer, Cham, 2015. doi: 10.1007/978-3-319-20325-6.

Jing Lei and Peter Bickel. A moment matching ensemble filter for nonlinear non-Gaussian data assimilation. *Monthly Weather Review*, 139(12):3964–3973, 2011. doi: 10.1175/2011MWR3553.1.

Siming Liang, Hoang Tran, Feng Bao, Hristo G. Chipilski, Peter Jan van Leeuwen, and Guannan Zhang. Ensemble score filter with image inpainting for data assimilation in tracking surface quasi-geostrophic dynamics with partial observations. *arXiv preprint arXiv:2501.12419*, 2025. URL https://arxiv.org/abs/2501.12419.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141, 1963. doi: 10.1175/1520-0469(1963)020⟨0130:DNF⟩2.0.CO;2.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Peter Manshausen, Yair Cohen, Peter Harrington, Jaideep Pathak, Mike Pritchard, Piyush Garg, Morteza Mardani, Karthik Kashinath, Simon Byrne, and Noah Brenowitz. Generative data assimilation of sparse weather station observations at kilometer scales. *Journal of Advances in Modeling Earth Systems*, 17(10):e2024MS004505, 2025. doi: 10.1029/2024MS004505.

Michael McCabe and Jed Brown. Learning to assimilate in chaotic dynamical systems. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 12237–12250. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/65cc2c8205a05d7379fa3a6386f710e1-Paper.pdf.

NOAA National Centers for Environmental Information. Global surface summary of the day - GSOD, 1999. URL https://www.ncei.noaa.gov/data/global-summary-of-the-day/access/2025/.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Ilan Price, Alvaro Sanchez-Gonzalez, Ferran Alet, Tom R. Andersson, Andrew El-Kadi, Dominic Masters, Timo Ewalds, Jacklynn Stott, Shakir Mohamed, Peter Battaglia, Remi Lam, and Matthew Willson. Probabilistic weather forecasting with machine learning. *Nature*, 637:84–90, 2025. doi: 10.1038/s41586-024-08252-9. URL https://www.nature.com/articles/s41586-024-08252-9.

Gareth O. Roberts and Jeffrey S. Rosenthal. General state space Markov chains and MCMC algorithms. *Probability Surveys*, 1:20–71, 2004. doi: 10.1214/154957804100000024.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

François Rozet, Ruben Ohana, Michael McCabe, Gilles Louppe, Francois Lanusse, and Shirley Ho. Lost in latent space: An empirical study of latent diffusion models for physics emulation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 134612–134657. Curran Associates, Inc., 2025. doi: 10.52202/085713-4492. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/c3b88408bb209dca87dad596ecf88bba-Paper-Conference.pdf.

François Rozet and Gilles Louppe. Score-based data assimilation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 40521–40541, 2023. doi: 10.52202/075280-1763.

François Rozet, Gérôme Andry, François Lanusse, and Gilles Louppe. Learning diffusion priors from observations by expectation maximization. In *Advances in Neural Information Processing Systems*, volume 37, pp. 87647–87682, 2024. doi: 10.52202/079017-2783.

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42818–42835. PMLR, 2024. URL https://proceedings.mlr.press/v235/ruhe24a.html.

Salva Rühling Cachay, Miika Aittala, Karsten Kreis, Noah Brenowitz, Arash Vahdat, Morteza Mardani, and Rose Yu. Elucidated rolling diffusion models for probabilistic forecasting of complex dynamics. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 1821–1860, 2025. doi: 10.52202/085713-0068. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/02cf868040aa0f0a1e1121cf255fdcfb-Paper-Conference.pdf.

Thomas Savary, François Rozet, and Gilles Louppe. Training-free Bayesian filtering with generative emulators. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ibcZNZwKfZ.

Phillip Si and Peng Chen. Latent-ensf: A latent ensemble score filter for high-dimensional data assimilation with sparse observation data. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 94879–94895, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ecc276bd799a37e9b66508ed8f609d3d-Paper-Conference.pdf.

Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pp. 369–386. SPIE, 2019.

Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 56242–56280. PMLR, 2025. URL https://proceedings.mlr.press/v267/song25b.html.

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. URL https://openreview.net/forum?id=PxTIG12RRHS.

Michael K Tippett, Jeffrey L Anderson, Craig H Bishop, Thomas M Hamill, and Jeffrey S Whitaker. Ensemble square root filters. *Monthly weather review*, 131(7):1485–1490, 2003.

Taos Transue, Bohan Chen, So Takao, and Bao Wang. Flow matching for efficient and scalable data assimilation. *SIAM/ASA Journal on Uncertainty Quantification*, 2026. URL https://arxiv.org/abs/2508.13313.

Ross Tulloch and K. Shafer Smith. Quasigeostrophic turbulence with explicit surface dynamics: Application to the atmospheric energy spectrum. *Journal of the Atmospheric Sciences*, 66(2): 450–467, 2009. doi: 10.1175/2008JAS2653.1.

Nikhil Vyas, Depen Morwani, Rosie Zhao, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. Soap: Improving and stabilizing shampoo using adam for language modeling. In *International Conference on Learning Representations*, volume 2025, pp. 93423–93444, 2025.

Xuguang Wang, Hristo G. Chipilski, Craig H. Bishop, Elizabeth Satterfield, Nancy Baker, and Jeffrey S. Whitaker. A multiscale local gain form ensemble transform Kalman filter (MLGETKF). *Monthly Weather Review*, 149(3):605–622, 2021. doi: 10.1175/MWR-D-20-0290.1.

A Details on variance-preserving diffusion models

The main body introduces diffusion models using the variance-exploding (VE) formulation. The rolling diffusion priors used in our experiments instead employ a variance-preserving (VP) parameterization. This choice changes the perturbation kernel and reverse update, but not the proposed data assimilation construction.

Let $\boldsymbol{x}_{\text{clean}} \in \mathbb{R}^D$ denote a clean sample. At diffusion time $t \in [0, 1]$, the VP perturbation process is defined by

$$\boldsymbol{x}(t) = \alpha(t)\boldsymbol{x}_{\text{clean}} + \sigma(t)\boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_D), \tag{12}$$

or, equivalently,

$$p\big(\boldsymbol{x}(t) \mid \boldsymbol{x}_{\text{clean}}\big) = \mathcal{N}\left(\boldsymbol{x}(t) \mid \alpha(t)\boldsymbol{x}_{\text{clean}}, \sigma(t)^2\boldsymbol{I}_D\right). \tag{13}$$

Our implementation uses

$$\alpha(t) = \cos^2\left(t \arccos\sqrt{\eta}\right), \qquad \sigma(t)^2 = 1 - \alpha(t)^2 + \eta^2, \qquad \eta = 10^{-3}. \tag{14}$$

Here $\alpha(t)$ denotes the direct signal coefficient multiplying the clean sample, rather than the cumulative variance coefficient $\bar{\alpha}(t)$ commonly used in DDPM (Ho et al., 2020) notation.

The small constant η avoids singular behavior at the endpoints. In particular, $\alpha(0) = 1$ and $\sigma(0) = \eta$, so $\boldsymbol{x}(t = 0) = \boldsymbol{x}_{\text{clean}} + \eta\boldsymbol{\epsilon}$ is approximately clean, while $\alpha(1) = \eta$ and $\sigma(1) = 1$, so the terminal distribution is approximately standard Gaussian.

The conditional score of the perturbation kernel is:

$$\nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{x}(t) \mid \boldsymbol{x}_{\text{clean}}\big) = -\frac{\boldsymbol{x}(t) - \alpha(t)\boldsymbol{x}_{\text{clean}}}{\sigma(t)^2} = -\frac{\boldsymbol{\epsilon}}{\sigma(t)}. \tag{15}$$

We used the noise-prediction parameterization $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}(t), t)$. Its score-equivalent parameterization is

$$\boldsymbol{s}_\theta\big(\boldsymbol{x}(t), t\big) = -\frac{\boldsymbol{\epsilon}_\theta\big(\boldsymbol{x}(t), t\big)}{\sigma(t)}, \tag{16}$$

where $\boldsymbol{s}_\theta$ approximates the score of the noisy marginal distribution $p_t(\boldsymbol{x}(t))$.

During startup, negative-time state slots are initialized as padding copies of the initial-state sample. In the practical algorithm, $y_{n-H:n}$ denotes only available observations with indices between $\max(1, n - H)$ and n. Unavailable observations contribute no likelihood factor.

Algorithm 1 Rolling Diffusion Data Assimilation (RDA) with editable history

Require: Score network s_θ; reverse-SDE coefficients $f(t), g(t)$; noise schedule $\sigma_0(t) < \cdots < \sigma_W(t)$; window/history sizes W, H; denoising steps S; renoise/denoise steps K; ensemble size J; observations $(y_1, y_2, \ldots)$; guidance strength λ; initial ensemble $\{x_0^{(j)}\}_{j=1}^J$

for $j = 1, \ldots, J$ **in parallel do** ▷ initialize window
  Build window $\widetilde{X}_0'^{(j)}(1)$ by staircase-noising $H{+}1{+}W$ copies of $x_0^{(j)}$, i.e., $\widetilde{X}_0'^{(j)}(1) := (x_{-H:0}^{(j),\sigma_0}(1), x_1^{(j),\sigma_1}(1), \ldots, x_W^{(j),\sigma_W}(1))$, where $x_w^{(j),\sigma_w}(1) \overset{\text{i.i.d.}}{\sim} \mathcal{N}(x_0^{(j)}, \sigma_w^2(1)I)$
end for
for $n = 0, 1, 2, \ldots$ **do** ▷ main assimilation loop
  for $j = 1, \ldots, J$ **in parallel do**
    for $k = 1, \ldots, K$ **do**
      $\widetilde{X}_n'^{(j)}(0) \leftarrow \textsc{Denoise}\big(\widetilde{X}_n'^{(j)}(1), y_{n-H:n}, f(t), g(t), s_\theta, S, \lambda\big)$
      $\widetilde{X}_n'^{(j)}(1) \leftarrow \textsc{Renoise}\big(\widetilde{X}_n'^{(j)}(0), \sigma_{0:W}(0), \sigma_{0:W}(1)\big)$
    end for
    $\big(\widehat{x}_n^{(j)},\ \widetilde{X}_{n+1}'^{(j)}(1)\big) \leftarrow \textsc{Rollout}\big(\widetilde{X}_n'^{(j)}(0)\big)$
  end for
  output ensemble $\{\widehat{x}_n^{(j)}\}_{j=1}^J$
end for

Algorithm 2 Re-noising and de-noising window

procedure $\textsc{Renoise}(\widetilde{X}_n'(0), \sigma_{0:W}(0), \sigma_{0:W}(1))$
  $v_0 \leftarrow \sigma_0^2(1)$ ▷ history and current frames
  $v_w \leftarrow \sigma_w^2(1) - \sigma_w^2(0), \quad w = 1, \ldots, W$ ▷ future frames: add only the missing variance
  Sample $\varepsilon \sim \mathcal{N}\Big(0, \mathrm{diag}\big(\underbrace{v_0 I, \ldots, v_0 I}_{H+1 \text{ blocks}}, v_1 I, \ldots, v_W I\big)\Big)$
  $\widetilde{X}_n'(1) \leftarrow \widetilde{X}_n'(0) + \varepsilon$
  return $\widetilde{X}_n'(1)$
end procedure

procedure $\textsc{Denoise}(\widetilde{X}_n'(1), y_{n-H:n}, f(t), g(t), s_\theta, S, \lambda)$
  (Use the equal partition $1 = t_0 > \cdots > t_S = t_{\min} \approx 0$*, with* $\Delta t = (1 - t_{\min})/S$*.)*
  for $s = 0, \ldots, S-1$ **do**
    $g_n \approx \nabla_{\widetilde{X}_n'(t_s)} \log p\big(y_{n-H:n} \mid \widetilde{X}_n'(t_s)\big)$ ▷ compute guidance term (DPS / MMPS)
    $\tilde{s}_n \leftarrow s_\theta\big(\widetilde{X}_n'(t_s), t_s\big) + \lambda g_n$ ▷ compute guided score
    Sample $z \sim \mathcal{N}(0, I)$
    $\widetilde{X}_n'(t_{s+1}) \leftarrow \widetilde{X}_n'(t_s) - \big[f(t_s)\widetilde{X}_n'(t_s) - g(t_s)^2 \tilde{s}_n\big]\Delta t + g(t_s)\sqrt{\Delta t}\, z$ ▷ Guided reverse-diffusion update
  end for
  return $\widetilde{X}_n'(0)$
end procedure

Algorithm 3 ROLLOUT: commit the current frame and advance the window

procedure ROLLOUT($\widetilde{X}'_n(0) = (h_1, \dots, h_H,\ c,\ f_1, \dots, f_W)$)
(h_i: edited history c: current clean frame f_w: partially denoised future frames)
  $\widehat{x}_n \leftarrow c$ ▷ commit current frame
  $(h'_1, \dots, h'_H) \leftarrow (h_2, \dots, h_H, c)$ ▷ drop oldest; committed frame enters history
  $h'_i \leftarrow h'_i + \sigma_0(1)\,\varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, I), \quad i = 1, \dots, H$ ▷ re-noise history
  $\varepsilon \sim \mathcal{N}(0, \sigma^2_{\max} I)$ ▷ sample fresh future frame
  $\widetilde{X}'_{n+1}(1) \leftarrow (h'_1, \dots, h'_H,\ f_1,\ f_2, \dots, f_W,\ \varepsilon)$ ▷ f_1 becomes current
  return $\widehat{x}_n,\ \widetilde{X}'_{n+1}(1)$
end procedure

B SOLVING INVERSE PROBLEMS VIA GUIDANCE

An unconditional diffusion model generates samples from $p(\boldsymbol{x}_{\text{clean}})$. Given an observation $\boldsymbol{y}$, inverse-problem guidance instead seeks samples from $p(\boldsymbol{x}_{\text{clean}} \mid \boldsymbol{y})$. By Bayes' rule, the posterior score at diffusion time t decomposes as

$$\nabla_{\boldsymbol{x}(t)} \log p_t\big(\boldsymbol{x}(t) \mid \boldsymbol{y}\big) = \nabla_{\boldsymbol{x}(t)} \log p_t\big(\boldsymbol{x}(t)\big) + \nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{y} \mid \boldsymbol{x}(t)\big). \tag{17}$$

The first term is supplied by the unconditional diffusion model. The remaining task is to approximate the likelihood score $\nabla_{\boldsymbol{x}(t)} \log p(\boldsymbol{y} \mid \boldsymbol{x}(t))$. Suppose that the observation model is $p(\boldsymbol{y} \mid \boldsymbol{x}_{\text{clean}}) = \mathcal{N}\left(\boldsymbol{y} \mid \mathcal{H}(\boldsymbol{x}_{\text{clean}}), \boldsymbol{\Sigma}_y\right)$, where $\mathcal{H}$ is a differentiable observation operator. Since this likelihood is defined on the clean state, its value at diffusion time t requires marginalizing over the unknown clean state: $p(\boldsymbol{y} \mid \boldsymbol{x}(t)) = \int p(\boldsymbol{y} \mid \boldsymbol{x}_{\text{clean}}) p(\boldsymbol{x}_{\text{clean}} \mid \boldsymbol{x}(t))\, \mathrm{d}\boldsymbol{x}_{\text{clean}}$. This integral is generally intractable.

B.1 DIFFUSION POSTERIOR SAMPLING (DPS)

Diffusion posterior sampling (DPS) approximates $p(\boldsymbol{x}_{\text{clean}} \mid \boldsymbol{x}(t))$ by a point mass at its conditional mean. Using Tweedie's formula, this clean estimate is

$$\widehat{\boldsymbol{x}}(\boldsymbol{x}(t)) := \mathbb{E}[\boldsymbol{x}_{\text{clean}} \mid \boldsymbol{x}(t)] \approx \frac{\boldsymbol{x}(t) + \sigma(t)^2 \boldsymbol{s}_\theta(\boldsymbol{x}(t), t)}{\alpha(t)} = \frac{\boldsymbol{x}(t) - \sigma(t) \boldsymbol{\epsilon}_\theta(\boldsymbol{x}(t), t)}{\alpha(t)}. \tag{18}$$

DPS therefore uses the approximation

$$p(\boldsymbol{y} \mid \boldsymbol{x}(t)) \approx \mathcal{N}\left(\boldsymbol{y} \mid \mathcal{H}\left(\widehat{\boldsymbol{x}}(\boldsymbol{x}(t))\right), \boldsymbol{\Sigma}_y\right), \tag{19}$$

whose gradient with respect to $\boldsymbol{x}(t)$ provides the likelihood guidance Chung et al. (2023).

B.2 MOMENT-MATCHING POSTERIOR SAMPLING (MMPS)

MMPS extends DPS by approximating the clean-state conditional as

$$p\big(\boldsymbol{x}_{\text{clean}} \mid \boldsymbol{x}(t)\big) \approx \mathcal{N}\big(\boldsymbol{x}_{\text{clean}} \mid \widehat{\boldsymbol{x}}_t, \widehat{\boldsymbol{\Sigma}}_t\big), \qquad \widehat{\boldsymbol{\Sigma}}_t = \frac{\sigma(t)^2}{\alpha(t)} \nabla^\top_{\boldsymbol{x}(t)} \widehat{\boldsymbol{x}}_t, \tag{20}$$

where $\widehat{\boldsymbol{x}}_t = \widehat{\boldsymbol{x}}(\boldsymbol{x}(t))$ is the Tweedie estimate defined above Rozet et al. (2024). The likelihood score is then approximated by

$$\nabla_{\boldsymbol{x}(t)} \log p\big(\boldsymbol{y} \mid \boldsymbol{x}(t)\big) \approx \left(\nabla^\top_{\boldsymbol{x}(t)} \widehat{\boldsymbol{x}}_t\right) \boldsymbol{H}_t^\top \left(\boldsymbol{\Sigma}_y + \boldsymbol{H}_t \widehat{\boldsymbol{\Sigma}}_t \boldsymbol{H}_t^\top\right)^{-1} \left(\boldsymbol{y} - \mathcal{H}(\widehat{\boldsymbol{x}}_t)\right), \tag{21}$$

where $\boldsymbol{H}_t = \nabla_{\boldsymbol{x}} \mathcal{H}(\boldsymbol{x})|_{\boldsymbol{x} = \widehat{\boldsymbol{x}}_t}$. The expression is exact under the Gaussian approximation when $\mathcal{H}$ is linear.

Guidance stabilization. Where enabled, we stabilize the MMPS guidance correction in the parameterization used by each sampler. Let $\boldsymbol{u}_t$ denote that correction, before multiplication by the guidance strength. We first apply an isotropic Fourier low-pass filter $\mathcal{L}_{\rho(t)}$ over the spatial dimensions, independently for each state channel and frame. The cutoff $\rho(t)$ may be fixed or vary over

denoising steps. The cutoff is a fraction of the maximum spatial radial frequency: $\mathcal{L}_\rho$ retains Fourier modes with radial frequency at most ρ times that maximum. In Table 18, "first $\rightarrow$ last" gives the cutoff at the first and last denoising steps; equal values indicate a constant cutoff. We then cap the RMS of the filtered correction separately for each ensemble member:

$$\widetilde{\boldsymbol{u}}_t = \mathcal{L}_{\rho(t)}(\boldsymbol{u}_t) \min\left\{1,\ \frac{c}{\mathrm{RMS}\big(\mathcal{L}_{\rho(t)}(\boldsymbol{u}_t)\big)}\right\}. \tag{22}$$

The sampler applies its guidance strength to $\widetilde{\boldsymbol{u}}_t$. Neither operation changes the observations. Filtering and clipping can each be disabled; their thresholds are method-specific because the samplers represent $\boldsymbol{u}_t$ in different units.

B.3 Monte Carlo Guidance

Monte Carlo guidance instead approximats the marginal likelihood using samples $\boldsymbol{x}_{\mathrm{clean}}^{(m)} \sim p(\boldsymbol{x}_{\mathrm{clean}} \mid \boldsymbol{x}(t))$:

$$p\big(\boldsymbol{y} \mid \boldsymbol{x}(t)\big) \approx \frac{1}{M} \sum_{m=1}^{M} p\big(\boldsymbol{y} \mid \boldsymbol{x}_{\mathrm{clean}}^{(m)}\big). \tag{23}$$

The likelihood guidance is obtained by differentiating the logarithm of this Monte Carlo estimate with respect to $\boldsymbol{x}(t)$. The approximation becomes exact as $M \rightarrow \infty$ under exact conditional sampling.

C Details on rolling diffusion

As mentioned, the main text presents rolling diffusion using a variance-exploding process. Here we describe the variance-preserving formulation in the case of rolling diffusion. Let $\boldsymbol{X}_n = (\boldsymbol{x}_n, \ldots, \boldsymbol{x}_{n+W})$ denote a window of $W+1$ consecutive clean states, consisting of one current frame and W future frames. Rolling diffusion associates each frame with a local diffusion time $\boldsymbol{\tau}(t) = (\tau_0(t), \ldots, \tau_W(t))$, where t denotes the global progress of a denoising pass.

For the standard rolling construction, the frame-specific VP coefficients are obtained by evaluating the scalar VP schedule at progressively increasing times:

$$\alpha_w^{\mathrm{roll}}(t) = \alpha\left(\frac{w+t}{W+1}\right), \qquad \sigma_w^{\mathrm{roll}}(t) = \sigma\left(\frac{w+t}{W+1}\right), \qquad w = 0, \ldots, W. \tag{24}$$

These coefficients satisfy

$$\begin{aligned} &\alpha_0^{\mathrm{roll}}(0) = \alpha(0), && \sigma_0^{\mathrm{roll}}(0) = \sigma(0), \\ &\alpha_W^{\mathrm{roll}}(1) = \alpha(1), && \sigma_W^{\mathrm{roll}}(1) = \sigma(1), \\ &\alpha_{w+1}^{\mathrm{roll}}(0) = \alpha_w^{\mathrm{roll}}(1), && \sigma_{w+1}^{\mathrm{roll}}(0) = \sigma_w^{\mathrm{roll}}(1), \qquad w = 0, \ldots, W-1. \end{aligned} \tag{25}$$

Earlier frames are assigned smaller local times and therefore retain more signal, while later frames are assigned larger local times and remain closer to Gaussian. This is the VP counterpart of the staircase conditions in Equations 5. The denoising, committing, shifting, and appending operations therefore remain the same as those described in Section 3.1. Although the main text expresses reverse diffusion using the score $\boldsymbol{s}_\theta$, our implementation uses the equivalent noise-prediction parameterization $\boldsymbol{\epsilon}_\theta$ introduced in Appendix A.

Training the rolling prior. For the editable-history model, we train on augmented clean windows $\boldsymbol{X}'_n = (\boldsymbol{x}_{n-H}, \ldots, \boldsymbol{x}_{n+W})$ of length $H+W+1$. The history and current frames share the VP coefficients $\alpha_0^{\mathrm{roll}}(t)$ and $\sigma_0^{\mathrm{roll}}(t)$, while future frame $\boldsymbol{x}_{n+w}$ uses $\alpha_w^{\mathrm{roll}}(t)$ and $\sigma_w^{\mathrm{roll}}(t)$ for $w = 1, \ldots, W$. Independent Gaussian noise is added to every frame.

The network jointly predicts the noise across the complete window and is trained with

$$\mathcal{L}_{\mathrm{roll}}(\theta) = \mathbb{E}\left[\frac{1}{(H+W+1)d_x} \sum_{r=0}^{H+W} \left\| \boldsymbol{\epsilon}_\theta\big(\widetilde{\boldsymbol{X}}'_n(t), t\big)_r - \boldsymbol{\epsilon}_r \right\|_2^2\right]. \tag{26}$$

Here, r indexes positions in the augmented window, and the expectation is over clean training windows, sampled diffusion times, and Gaussian perturbations. Dependence on the frame-specific noise levels is suppressed in the network notation. Training uses no observations; these are introduced only during guided sampling.

D FILTERING INTERPRETATION

We first show that repeated re-noising and denoising form a Markov chain that converges to the window posterior given recent observations. Lemma 1 then bounds the error caused by omitting older observations under strong mixing. Finally, we propagate the fixed-window convergence across assimilation times and combine these results to prove Theorem 3.

D.1 PROBABILITY MEASURES AND MARKOV OPERATORS

For the analysis, we use a noised trajectory marginal of a fixed state–observation model as the window prior and assume exact conditional denoising. Throughout this appendix, observations are treated as fixed data, and the laws of algorithmic outputs refer only to the sampler's randomness.

Let d_x denote the dimension of one frame and $d_{\mathrm{w}} := (H + W + 1)d_x$ the window dimension. Window coordinates are indexed by physical times $n - H, \dots, n + W$. All spaces carry their Borel sigma-algebras. We write $\mathcal{P}(\mathbb{R}^d)$ for Borel probability measures and $\delta_{\boldsymbol{u}}$ for the Dirac measure at $\boldsymbol{u}$.

Our total variation convention is $\|\mu - \nu\|_{\mathrm{TV}} = 2 \sup_A |\mu(A) - \nu(A)|$. Equivalently,

$$\|\lambda - \lambda'\|_{\mathrm{TV}} = \sup_{\substack{f \text{ measurable} \\ \|f\|_\infty \le 1}} \left| \int f(\boldsymbol{u})\lambda(\mathrm{d}\boldsymbol{u}) - \int f(\boldsymbol{u})\lambda'(\mathrm{d}\boldsymbol{u}) \right|. \tag{27}$$

This distance is at most 2 for probability measures.

A Markov operator $\mathcal{T}$ acts through its transition kernel:

$$\mathcal{T}(\lambda)(A) = \int \mathcal{T}(\delta_{\boldsymbol{u}})(A)\lambda(\mathrm{d}\boldsymbol{u}). \tag{28}$$

Since integrating a function bounded by 1 against a probability kernel preserves this bound, equation 27 implies

$$\|\mathcal{T}(\lambda) - \mathcal{T}(\lambda')\|_{\mathrm{TV}} \le \|\lambda - \lambda'\|_{\mathrm{TV}}. \tag{29}$$

For $\lambda \in \mathcal{P}(\mathbb{R}^{d_{\mathrm{w}}})$ and Borel $A \subseteq \mathbb{R}^{d_x}$, the current-frame marginalization operator is

$$\mathcal{C}_n(\lambda)(A) := \lambda\big(\{\boldsymbol{u} : \boldsymbol{u}_n \in A\}\big). \tag{30}$$

It is a Markov operator, so equation 29 applies to $\mathcal{C}_n$.

D.2 WINDOW POSTERIOR AND CONDITIONAL UPDATES

Fix $n > H$ and the observed values $\boldsymbol{y}_{n-H:n}$. The prior on $\tilde{\boldsymbol{X}}'_n(0)$ is constructed from the trajectory marginal $p(\boldsymbol{x}_{n-H:n+W} \in \cdot)$ of the fixed physical model. Its history and current coordinates remain clean, while independent Gaussian noise of variance $\sigma_w^2(0)$ is added to future frame $\boldsymbol{x}_{n+w}$, $w = 1, \dots, W$. All artificial noises are independent of the physical state–observation process.

Define the window posterior at $t = 0$ by

$$\Pi_n := p\big(\tilde{\boldsymbol{X}}'_n(0) \in \cdot \mid \boldsymbol{y}_{n-H:n}\big). \tag{31}$$

Let $\ell_i(\boldsymbol{u}, \boldsymbol{y})$ denote the time-i observation density relative to a sigma-finite reference measure μ_{obs}. Conditional independence of the observations gives the window likelihood

$$L_n(\boldsymbol{u}) := \prod_{h=0}^{H} \ell_{n-h}(\boldsymbol{u}_{n-h}, \boldsymbol{y}_{n-h}) \tag{32}$$

and the posterior can be written as

$$\Pi_n(\mathrm{d}\boldsymbol{u}) = \frac{L_n(\boldsymbol{u})p(\tilde{\boldsymbol{X}}'_n(0) \in \mathrm{d}\boldsymbol{u})}{\int L_n(\boldsymbol{v})p(\tilde{\boldsymbol{X}}'_n(0) \in \mathrm{d}\boldsymbol{v})}. \tag{33}$$

For a window law μ and a Borel set of windows A, define the re-noising and denoising operators by

$$\mathcal{R}_n(\mu)(A) := \int p\big(\tilde{\boldsymbol{X}}'_n(1) \in A \mid \tilde{\boldsymbol{X}}'_n(0) = \boldsymbol{u}\big)\mu(\mathrm{d}\boldsymbol{u}), \tag{34}$$

$$\mathcal{D}_n(\mu)(A) := \int p\big(\tilde{\boldsymbol{X}}'_n(0) \in A \mid \tilde{\boldsymbol{X}}'_n(1) = \boldsymbol{z}, \boldsymbol{y}_{n-H:n}\big)\mu(\mathrm{d}\boldsymbol{z}). \tag{35}$$

These conditionals are paired through the same forward noising model. The prior and observation likelihoods are fixed by the state–observation model; the incoming window law enters only through initialization.

Under the independent-increment VE coupling, re-noising a window $\boldsymbol{u}$ gives

$$\boldsymbol{z}_{n+w} = \begin{cases} \boldsymbol{u}_{n+w} + \sigma_0(1)\boldsymbol{\epsilon}_w, & -H \le w \le 0, \\ \boldsymbol{u}_{n+w} + \sqrt{\sigma_w^2(1) - \sigma_w^2(0)}\boldsymbol{\epsilon}_w, & 1 \le w \le W, \end{cases} \tag{36}$$

where the $\boldsymbol{\epsilon}_w$ are independent standard Gaussian vectors with the dimension of a single frame. Each coordinate receives the additional noise needed to return to its level at $t = 1$.

Let $r_n(\boldsymbol{z} \mid \boldsymbol{u})$ denote the Gaussian density of the re-noised window $\boldsymbol{z}$ conditional on the window $\boldsymbol{u}$ at $t = 0$, as specified by equation 36. Thus

$$\mathcal{R}_n(\delta_{\boldsymbol{u}})(\mathrm{d}\boldsymbol{z}) = r_n(\boldsymbol{z} \mid \boldsymbol{u})\mathrm{d}\boldsymbol{z}. \tag{37}$$

The state-independent, positive-definite re-noising covariance implies that $r_n(\boldsymbol{z} \mid \boldsymbol{u}) > 0$ for all $\boldsymbol{u}, \boldsymbol{z}$ and

$$\sup_{\boldsymbol{u},\boldsymbol{z} \in \mathbb{R}^{d_{\mathrm{w}}}} r_n(\boldsymbol{z} \mid \boldsymbol{u}) < \infty. \tag{38}$$

The additional forward increments are independent of the observations conditional on the window at $t = 0$. Hence the joint endpoint posterior is

$$p\big(\tilde{\boldsymbol{X}}'_n(0) \in \mathrm{d}\boldsymbol{u}, \tilde{\boldsymbol{X}}'_n(1) \in \mathrm{d}\boldsymbol{z} \mid \boldsymbol{y}_{n-H:n}\big) = \Pi_n(\mathrm{d}\boldsymbol{u})r_n(\boldsymbol{z} \mid \boldsymbol{u})\mathrm{d}\boldsymbol{z}. \tag{39}$$

Its noisy marginal has density

$$m_n(\boldsymbol{z}) := \int r_n(\boldsymbol{z} \mid \boldsymbol{u})\Pi_n(\mathrm{d}\boldsymbol{u}), \tag{40}$$

which is positive and finite at every $\boldsymbol{z}$. We therefore use the Bayes conditional

$$\mathcal{D}_n(\delta_{\boldsymbol{z}})(\mathrm{d}\boldsymbol{u}) = \frac{r_n(\boldsymbol{z} \mid \boldsymbol{u})\Pi_n(\mathrm{d}\boldsymbol{u})}{m_n(\boldsymbol{z})}. \tag{41}$$

This defines a measurable probability kernel for every noisy input.

Posterior normalization and nondegenerate re-noising. The Gaussian observation models used here have positive-definite noise covariances, so each likelihood $\ell_i(\cdot, \boldsymbol{y}_i)$ is strictly positive and bounded. Consequently, the normalizing constant in equation 33 is positive and finite for any window prior, including with the nonlinear observation operators used in our experiments. The strictly ordered rolling schedule in Section 3.1 satisfies $\sigma_0(1) = \sigma_1(0) > 0$ and $\sigma_w(0) = \sigma_{w-1}(1) < \sigma_w(1)$ for $1 \le w \le W$. Together with the finite terminal noise level, these relations ensure the nondegenerate re-noising required by Theorem 2. Exact pairing is specified by equation 41.

D.3 Fixed-window convergence

One complete re-noise–denoise cycle is the Markov operator

$$\mathcal{M}_n := \mathcal{D}_n \circ \mathcal{R}_n. \tag{42}$$

For $K \geq 1$ denoising passes at time n, the first pass applies $\mathcal{D}_n$ to the incoming noisy window. Each additional pass applies $\mathcal{R}_n$ followed by $\mathcal{D}_n$, using the same observations $\boldsymbol{y}_{n-H:n}$ and fresh conditional randomness. Starting from a noisy-window law μ, the final window law is $\mathcal{M}_n^{K-1}(\mathcal{D}_n(\mu))$, where $\mathcal{M}_n^0$ is the identity. Here K counts complete denoising passes, whereas S in Algorithm 1 counts numerical steps within each pass.

The window retains the same frame indices during these iterations; the current frame is committed and Algorithm 3 is applied only after the final pass. Thus $K = 1$ recovers the exact-conditional counterpart of the one-pass RDA procedure.

Theorem 2 (Convergence of fixed-window re-noise–denoise iterations)**.** *Fix $H, W \geq 1$, $n > H$, and the observed values $\boldsymbol{y}_{n-H:n}$. Suppose that Π_n has a positive finite normalizing constant, that every Gaussian variance increment in equation 36 is finite and strictly positive, and that $\mathcal{D}_n$ is the exact Bayes conditional paired with $\mathcal{R}_n$.*

The Markov chain of windows at $t = 0$ induced by $\mathcal{M}_n$ is time-homogeneous and reversible, with unique invariant probability measure Π_n. It converges in total variation from every initial state. In particular, for every probability law μ on noisy windows,

$$\lim_{K\to\infty} \left\| \mathcal{M}_n^{K-1}\big(\mathcal{D}_n(\mu)\big) - \Pi_n \right\|_{\mathrm{TV}} = 0. \tag{43}$$

Its invariant current-frame marginal is

$$\mathcal{C}_n(\Pi_n) = p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{n-H:n}). \tag{44}$$

Proof. At the fixed physical time n, each additional pass applies the same re-noising and denoising kernels with fresh randomness and without shifting the window. The resulting windows at $t = 0$ therefore form a time-homogeneous Markov chain with operator $\mathcal{M}_n = \mathcal{D}_n \circ \mathcal{R}_n$.

By equation 41 and Tonelli's theorem,

$$\Pi_n(\mathrm{d}\boldsymbol{u})\mathcal{M}_n(\delta_{\boldsymbol{u}})(\mathrm{d}\boldsymbol{v}) = \int m_n(\boldsymbol{z})\mathcal{D}_n(\delta_{\boldsymbol{z}})(\mathrm{d}\boldsymbol{u})\mathcal{D}_n(\delta_{\boldsymbol{z}})(\mathrm{d}\boldsymbol{v})\mathrm{d}\boldsymbol{z}. \tag{45}$$

The right-hand side is symmetric in $\boldsymbol{u}$ and $\boldsymbol{v}$, proving reversibility. Integrating over $\boldsymbol{u}$ gives $\mathcal{M}_n(\Pi_n) = \Pi_n$.

To establish convergence, strict positivity of r_n in equation 41 implies that $\mathcal{D}_n(\delta_{\boldsymbol{z}})(A) > 0$ if and only if $\Pi_n(A) > 0$. Integrating over the re-noising step gives

$$\mathcal{M}_n(\delta_{\boldsymbol{u}})(A) = 0 \Longleftrightarrow \Pi_n(A) = 0 \tag{46}$$

for every state $\boldsymbol{u}$ and Borel set A. Every set of positive Π_n-measure is therefore reachable in one step from every state, proving Π_n-irreducibility. This also rules out a nontrivial periodic decomposition: any periodic class of positive Π_n-measure would have zero one-step return probability, contradicting equation 46.

The Borel sigma-algebra is countably generated. Hence Roberts & Rosenthal (2004, Theorem 4) gives a measurable set G with $\Pi_n(G) = 1$ such that $\|\mathcal{M}_n^k(\delta_{\boldsymbol{v}}) - \Pi_n\|_{\mathrm{TV}} \to 0$ for every $\boldsymbol{v} \in G$.

For any initial probability measure λ, equation 46 implies

$$\mathcal{M}_n(\lambda)(G) = \int \mathcal{M}_n(\delta_{\boldsymbol{u}})(G)\lambda(\mathrm{d}\boldsymbol{u}) = 1. \tag{47}$$

Therefore

$$\|\mathcal{M}_n^{k+1}(\lambda) - \Pi_n\|_{\mathrm{TV}} \leq \int \|\mathcal{M}_n^k(\delta_{\boldsymbol{v}}) - \Pi_n\|_{\mathrm{TV}}\mathcal{M}_n(\lambda)(\mathrm{d}\boldsymbol{v}) \longrightarrow 0 \tag{48}$$

by dominated convergence, since the integrand converges to zero for $\mathcal{M}_n(\lambda)$-almost every $\boldsymbol{v}$ and is bounded by 2. Taking $\lambda = \delta_{\boldsymbol{u}}$ proves convergence from every initial state. Taking $\lambda = \mathcal{D}_n(\mu)$ proves equation 43, because the first pass produces $\mathcal{D}_n(\mu)$ and the remaining $K - 1$ passes apply $\mathcal{M}_n$. Applying equation 48 to an invariant λ also proves uniqueness.

Finally, the current coordinate of the model window at $t = 0$ is the clean physical state $\boldsymbol{x}_n$, since $\sigma_0(0) = 0$. Thus

$$\mathcal{C}_n(\Pi_n) = p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{n-H:n}), \tag{49}$$

as claimed. □

D.4 OBSERVATION-WINDOW APPROXIMATION

To control observation truncation, we use the time-indexed version of Cappé et al. (2005, Assumption 4.3.21). Let $\mathsf{X} \subseteq \mathbb{R}^{d_x}$ be the Borel state space and $Q_i(\boldsymbol{u}, \mathrm{d}\boldsymbol{v})$ the physical transition kernels. This assumption concerns the physical state–observation model and is used only to compare the truncated posterior with the full filter.

Assumption 1 (Physical Markov model and strong mixing)**.** *The physical model has initial law $p(\boldsymbol{x}_0 \in \cdot)$ supported on X and satisfies*

$$p(\boldsymbol{x}_i \in \mathrm{d}\boldsymbol{v} \mid \boldsymbol{x}_{0:i-1}) = Q_i(\boldsymbol{x}_{i-1}, \mathrm{d}\boldsymbol{v}), \tag{50}$$

$$p(\boldsymbol{y}_{1:N} \in \mathrm{d}\boldsymbol{y}_{1:N} \mid \boldsymbol{x}_{0:N+W}) = \prod_{i=1}^{N} \big[\ell_i(\boldsymbol{x}_i, \boldsymbol{y}_i)\mu_{\mathrm{obs}}(\mathrm{d}\boldsymbol{y}_i)\big]. \tag{51}$$

For each observation time i, there exist a measurable probability kernel κ_i from the observation space to X and measurable functions $\varsigma_i^-, \varsigma_i^+ : \mathbb{R}^{d_y} \to (0, \infty)$ such that, for every $\boldsymbol{u} \in \mathsf{X}$, observation $\boldsymbol{y}$, and Borel set $A \subseteq \mathsf{X}$,

$$\varsigma_i^-(\boldsymbol{y})\kappa_i(\boldsymbol{y}, A) \le \int_A \ell_i(\boldsymbol{v}, \boldsymbol{y})Q_i(\boldsymbol{u}, \mathrm{d}\boldsymbol{v}) \le \varsigma_i^+(\boldsymbol{y})\kappa_i(\boldsymbol{y}, A). \tag{52}$$

Taking $A = \mathsf{X}$ gives $0 < \varsigma_i^-(\boldsymbol{y})/\varsigma_i^+(\boldsymbol{y}) \le 1$.

Sufficient condition with Gaussian observations. For the physical Markov model above, a sufficient condition for equation 52 is that, for each i, there exist a probability measure ν_i on X and constants $0 < a_i \le b_i < \infty$ such that

$$a_i\nu_i(A) \le Q_i(\boldsymbol{u}, A) \le b_i\nu_i(A) \tag{53}$$

for all $\boldsymbol{u} \in \mathsf{X}$ and Borel $A \subseteq \mathsf{X}$, together with $0 < \int_{\mathsf{X}} \ell_i(\boldsymbol{v}, \boldsymbol{y})\nu_i(\mathrm{d}\boldsymbol{v}) < \infty$ for every $\boldsymbol{y}$. When equation 53 holds, Gaussian observations with positive-definite noise covariance ensure this likelihood integrability condition. Then Cappé et al. (2005, Lemma 4.3.25) gives equation 52, with $\varsigma_i^-(\boldsymbol{y})/\varsigma_i^+(\boldsymbol{y}) = a_i/b_i$.

Lemma 1 (Observation-window approximation)**.** *Under Assumption 1, for every $1 \le H < n \le N$ and admissible observation record $\boldsymbol{y}_{1:n}$,*

$$\|p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{n-H:n}) - \pi_n\|_{\mathrm{TV}} \le 2 \prod_{i=n-H+1}^{n} \left(1 - \frac{\varsigma_i^-(\boldsymbol{y}_i)}{\varsigma_i^+(\boldsymbol{y}_i)}\right). \tag{54}$$

Proof. The full and truncated filters start at time $n - H$ from $p(\boldsymbol{x}_{n-H} \in \cdot \mid \boldsymbol{y}_{1:n-H})$ and $p(\boldsymbol{x}_{n-H} \in \cdot \mid \boldsymbol{y}_{n-H})$, respectively, and then assimilate the same observations $\boldsymbol{y}_{n-H+1:n}$ using the same filtering recursion. Apply Cappé et al. (2005, Proposition 4.3.23(iii)) with initial law $p(\boldsymbol{x}_1 \in \cdot)$ and physical time 1 reindexed as time 0. Its truncation index is $n - H - 1$, while its state index and observation horizon are both $n - 1$. The prediction factor therefore has exponent zero, leaving precisely the H factors in equation 54. The proof applies unchanged to the time-indexed kernels in Assumption 1. □

D.5 ASYMPTOTIC FILTERING ACCURACY

Fix $H, W \ge 1$ and a finite assimilation interval $n_0, \ldots, N$, where $n_0 > H$ and the physical model is defined through time $N + W$. At every assimilation time, apply K denoising passes before committing the current frame. The physical model, initial noisy-window law, and roll-out rule do not depend on K. Initialization may depend on $\boldsymbol{y}_{1:n_0-1}$, but not on later observations, and all auxiliary random seeds are independent of the data.

For one ensemble member, let $\widehat{\boldsymbol{x}}_n^{[K]}$ denote the sample committed at time n, computed using only $\boldsymbol{y}_{1:n}$ and the algorithm's randomization.

The following theorem is the precise version of the informal Theorem 1 in the main text.

Theorem 3 (Filtering accuracy under strong mixing)**.** *Consider the K-pass procedure described above. Suppose that the hypotheses of Theorem 2 hold at every assimilation time and that Assumption 1 holds. Fix an admissible observation record $\boldsymbol{y}_{1:N}$, meaning that the full and truncated posteriors under consideration have positive finite normalizing constants.*

There exists $\rho_N \in (0,1)$, depending only on the physical model and this observation record, but not on K, n, H, or W, such that, for every $n \in \{n_0, \dots, N\}$, the following limit exists and satisfies

$$\lim_{K\to\infty} \left\| \mathrm{Law}(\widehat{\boldsymbol{x}}_n^{[K]}) - \pi_n \right\|_{\mathrm{TV}} \le 2\rho_N^H. \tag{55}$$

Proof. Fix an admissible observation record $\boldsymbol{y}_{1:N}$. Let $\Lambda_n^{[K]}$ denote the law of the window at $t=0$ produced after K passes at physical time n, with randomness only from the algorithm.

For $n = n_0+1, \dots, N$, let $\mathcal{B}_{n-1}$ be the Markov operator induced by Algorithm 3, mapping a window at $t=0$ and physical time $n-1$ to the noisy input at physical time n. It shifts the window, re-noises the retained history, and appends a new terminal frame. Writing μ_{n_0} for the common initial noisy-window law, the output laws satisfy

$$\Lambda_{n_0}^{[K]} = \mathcal{M}_{n_0}^{K-1}\big(\mathcal{D}_{n_0}(\mu_{n_0})\big), \tag{56}$$

$$\Lambda_n^{[K]} = \mathcal{M}_n^{K-1}\big(\mathcal{D}_n(\mathcal{B}_{n-1}(\Lambda_{n-1}^{[K]}))\big). \tag{57}$$

Although μ_{n_0} and the roll-out operators are independent of K, the incoming law at later times may depend on K through preceding assimilation steps.

We establish convergence to Π_n by induction over the assimilation times. Theorem 2 gives $\|\Lambda_{n_0}^{[K]} - \Pi_{n_0}\|_{\mathrm{TV}} \to 0$. Suppose convergence has been established at time $n-1$. Adding and subtracting the output obtained from the fixed input law $\mathcal{B}_{n-1}(\Pi_{n-1})$, and using Markov nonexpansiveness, gives

$$\begin{aligned}\|\Lambda_n^{[K]} - \Pi_n\|_{\mathrm{TV}} \le{}& \|\Lambda_{n-1}^{[K]} - \Pi_{n-1}\|_{\mathrm{TV}} \\ &+ \left\|\mathcal{M}_n^{K-1}\big(\mathcal{D}_n(\mathcal{B}_{n-1}(\Pi_{n-1}))\big) - \Pi_n\right\|_{\mathrm{TV}}.\end{aligned} \tag{58}$$

The first term tends to zero by induction, and the second by Theorem 2 applied to the fixed noisy initialization law $\mathcal{B}_{n-1}(\Pi_{n-1})$. Since the assimilation interval is finite,

$$\lim_{K\to\infty} \max_{n_0\le n\le N} \|\Lambda_n^{[K]} - \Pi_n\|_{\mathrm{TV}} = 0. \tag{59}$$

The committed sample has law

$$\mathrm{Law}(\widehat{\boldsymbol{x}}_n^{[K]}) = \mathcal{C}_n(\Lambda_n^{[K]}). \tag{60}$$

Marginalization nonexpansiveness and equation 49 therefore imply

$$\lim_{K\to\infty} \left\| \mathrm{Law}(\widehat{\boldsymbol{x}}_n^{[K]}) - p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{n-H:n}) \right\|_{\mathrm{TV}} = 0 \tag{61}$$

for every $n \in \{n_0, \dots, N\}$.

To bound the remaining observation-truncation error, define

$$\rho_N := 1 - \frac{1}{2} \min_{1\le i\le N} \frac{\varsigma_i^-(\boldsymbol{y}_i)}{\varsigma_i^+(\boldsymbol{y}_i)}. \tag{62}$$

The finite minimum is positive, so $\rho_N \in (0,1)$, and every factor in equation 54 is at most ρ_N. This choice depends only on the physical model and the fixed observation record.

By the reverse triangle inequality, equation 61, and Lemma 1,

$$\begin{aligned}\lim_{K\to\infty} \left\| \mathrm{Law}(\widehat{\boldsymbol{x}}_n^{[K]}) - \pi_n \right\|_{\mathrm{TV}} &= \|p(\boldsymbol{x}_n \in \cdot \mid \boldsymbol{y}_{n-H:n}) - \pi_n\|_{\mathrm{TV}} \\ &\le 2 \prod_{i=n-H+1}^{n} \left(1 - \frac{\varsigma_i^-(\boldsymbol{y}_i)}{\varsigma_i^+(\boldsymbol{y}_i)}\right) \\ &\le 2\rho_N^H.\end{aligned} \tag{63}$$

This proves equation 55. □

The experiments in Section 5 evaluate the one-pass implementation with approximate guidance.

D.6 Interpretation and relation to the implementation

The limit in equation 61 forgets the incoming window, including any information it carries from observations before $n - H$. An exact sliding-window sampler using the same trajectory prior and observation window has the same current-frame target. At finite K, however, the incoming window may retain information from earlier observations. The analysis does not quantify the effect of this warm start or provide a convergence rate in K. Although the limiting current-frame marginal is independent of W, finite-pass behavior can depend on the window size and re-noising levels.

The exponential dependence on H comes from the observation-forgetting estimate in Lemma 1. Its physical Markov and strong-mixing assumptions are used for the comparison with the full filter, not for fixed-window convergence. The Markov condition is an additional restriction on the physical model relative to the general history-dependent formulation in equation 1. The constant ρ_N in equation 62 is uniform over assimilation times within a fixed record $\boldsymbol{y}_{1:N}$, but may approach one as N increases.

In the experiments, the prior is learned and denoising is numerically discretized using MMPS guidance with tuned strengths. MMPS remains approximate at unit strength, and the resulting updates need not preserve Π_n. The deterministic L63 and SQG transitions also fail the stated strong-mixing condition. Consequently, the theory does not predict monotone improvement of the reported SED and ES as K increases.

E Experimental details

E.1 Metrics

Let $\{\widehat{\boldsymbol{x}}_n^{(j)}\}_{j=1}^J$ denote the predicted ensemble at physical time n, and let $\boldsymbol{x}_n^\star$ denote the reference truth. The ensemble mean is

$$\bar{\boldsymbol{x}}_n = \frac{1}{J}\sum_{j=1}^{J} \widehat{\boldsymbol{x}}_n^{(j)}. \tag{64}$$

Unless otherwise stated, metrics are computed at each physical time step and then averaged over the evaluation window and test trajectories.

Root mean squared error The root mean squared error (RMSE) measures the error of the ensemble mean. For a state with flattened dimension D,

$$\mathrm{RMSE}_n = \left(\frac{1}{D}\sum_{i=1}^{D}\left(\bar{x}_{n,i} - x^\star_{n,i}\right)^2\right)^{1/2}. \tag{65}$$

CRPS For ensemble-valued forecasts, we use the fair finite-ensemble CRPS (Gneiting & Raftery, 2007; Ferro, 2014):

$$\begin{aligned}\mathrm{CRPS}_n = \frac{1}{J}\sum_{j=1}^{J}\left\|\widehat{\boldsymbol{x}}_n^{(j)} - \boldsymbol{x}_n^\star\right\|_{1,D} \\ - \frac{1}{2J(J-1)}\sum_{j=1}^{J}\sum_{k=1}^{J}\left\|\widehat{\boldsymbol{x}}_n^{(j)} - \widehat{\boldsymbol{x}}_n^{(k)}\right\|_{1,D}.\end{aligned} \tag{66}$$

Here,

$$\|\boldsymbol{u}\|_{1,D} = \frac{1}{D}\sum_{i=1}^{D}|u_i| \tag{67}$$

denotes the mean absolute value over the flattened state dimension. When $J = 1$, the pairwise correction term is set to zero.

Energy score The energy score (ES) is a multivariate generalization of CRPS:

$$\begin{aligned}\mathrm{ES}_n = \frac{1}{J}\sum_{j=1}^{J}\left\|\widehat{\boldsymbol{x}}_n^{(j)} - \boldsymbol{x}_n^{\star}\right\|_2 \\ - \frac{1}{2J^2}\sum_{j=1}^{J}\sum_{k=1}^{J}\left\|\widehat{\boldsymbol{x}}_n^{(j)} - \widehat{\boldsymbol{x}}_n^{(k)}\right\|_2 .\end{aligned} \tag{68}$$

Sliced energy distance When a reference posterior is available, we compare the predicted ensemble distribution against that reference. Let $X_{n,d}$ be a random draw from the predicted ensemble marginal in coordinate d, and let $Y_{n,d}$ be a random draw from the corresponding reference posterior marginal. We define

$$\begin{aligned}\mathrm{ED}_{n,d} = \mathbb{E}|X_{n,d} - Y_{n,d}| - \frac{1}{2}\mathbb{E}|X_{n,d} - X'_{n,d}| \\ - \frac{1}{2}\mathbb{E}|Y_{n,d} - Y'_{n,d}|,\end{aligned} \tag{69}$$

where $X'_{n,d}$ and $Y'_{n,d}$ are independent copies, and all draws in each expectation are independent. In practice, expectations are estimated using ensemble samples and reference posterior samples or quantiles.

The sliced energy distance (SED) averages these one-dimensional distances over state coordinates:

$$\mathrm{SED}_n = \frac{1}{D}\sum_{d=1}^{D}\mathrm{ED}_{n,d}. \tag{70}$$

E.2 Lorenz'63

The Lorenz–63 (L63) model is given by

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \sigma(y - x), \tag{71}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t} = x(\rho - z) - y, \tag{72}$$

$$\frac{\mathrm{d}z}{\mathrm{d}t} = xy - \beta z, \tag{73}$$

with the standard parameters $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$. In our experiments, trajectories have length $T = 1000$ and are generated with time step $\Delta t = 0.025$. All states are normalized using

$$\boldsymbol{w}_n = \frac{\boldsymbol{x}_n - \boldsymbol{\mu}}{\boldsymbol{\sigma}}, \qquad \boldsymbol{\mu} = (0, 0, 25), \qquad \boldsymbol{\sigma} = (8, 9, 8.6). \tag{74}$$

E.2.1 Model training

We generate L63 trajectories using the L63 ODE. After 1024 spinup transitions, we retain trajectories of 1024 states, using 819 trajectories for training and 102 for validation. We train a RDA diffusion prior on normalized L63 trajectories. The model uses a window length of 22 and commit index 14, so that each denoising window contains 14 history states, one committed state, and seven future states. The denoising network is an MLP with time embedding dimension 32, hidden width 256, depth 5, and SiLU activations. We use an ϵ-prediction objective. During training, each window uses the initialization noise schedule with probability $\beta_{\mathrm{train}} = 0.5$ and the editable-history rolling schedule otherwise.

The model is trained for 4096 epochs using AdamW (Loshchilov & Hutter, 2019) with learning rate 10^{-3}, weight decay 10^{-3}, batch size 64, and a linear learning-rate schedule. We use separate training and validation trajectory files, and select the checkpoint using validation loss.

E.2.2 DATA ASSIMILATION SETUP

We consider a nonlinear squared-observation setting. At each observed time, the first normalized coordinate is observed through

$$y_n = h(\boldsymbol{w}_n) + \eta_n, \qquad h(\boldsymbol{w}_n) = w_{n,1}^2, \qquad \eta_n \sim \mathcal{N}(0, \sigma_{\mathrm{obs}}^2). \tag{75}$$

The physical observation noise is $\sigma_{\mathrm{obs}}^{\mathrm{phys}} = 17.0$. Since the squared observation is applied in normalized coordinates, the corresponding normalized noise level is

$$\sigma_{\mathrm{obs}} = \frac{\sigma_{\mathrm{obs}}^{\mathrm{phys}}}{\sigma_1^2} = \frac{17.0}{8^2} = 0.265625. \tag{76}$$

We evaluate two observation frequencies. In the high-frequency setting, observations are available at every time step. In the low-frequency setting, observations are available every six time steps, corresponding to an inter-observation gap of 0.15 time units. The low-frequency observations are obtained by subsampling the same saved high-frequency observation cache, ensuring that all methods use identical observations.

All methods use $J = 512$ ensemble members. For RDA filtering, observations are assimilated with MMPS guidance inside each denoising window. Unless otherwise stated, the prefix states remain editable during denoising.

Initialization. For RDA, each ensemble member is initialized independently as

$$\boldsymbol{x}_0^{(j)} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0), \qquad \boldsymbol{\mu}_0 = (0, 0, 25), \qquad \boldsymbol{\Sigma}_0 = \begin{bmatrix} 64 & 50 & 0 \\ 50 & 81 & 0 \\ 0 & 0 & 75 \end{bmatrix}. \tag{77}$$

This Gaussian defines a common broad initialization prior over the three Lorenz–63 state variables. The mean centers the ensemble at zero in the first two coordinates and at (25) in the third coordinate, while the diagonal entries of ($\boldsymbol{\Sigma}_0$) correspond to marginal standard deviations of (8), (9), and ($\sqrt{75}$), respectively. The positive off-diagonal covariance between the first two coordinates introduces their expected co-variation, whereas the third coordinate is initially uncorrelated with the first two under this prior. Thus, the initialization provides substantial state uncertainty while preserving a simple correlation structure across the Lorenz–63 variables.

Each sampled state is copied across the initial window and perturbed according to the frame-specific VP schedule; RDA uses no numerical spinup or external forecaster. EnKF, ESRF, EnSF, and DAISI draw from the same Gaussian distribution but apply 64 numerical model transitions before assimilation. SDA samples its initial window from its diffusion prior conditioned on the available observations, whereas FlowDAS and the ForcingDAS variants use truth-conditioned initial histories. Since evaluation discards the first 100 steps, initialization transients are excluded from the reported metrics.

E.2.3 HYPERPARAMETER TUNING

We tune hyperparameters with Optuna (Akiba et al., 2019). Hyperparameters are selected on 1 validation sample by minimizing SED against a bootstrap particle filter reference. Final evaluation is performed on held-out samples $1, \ldots, 9$.

For RDA filtering, we tune the number of denoising steps, Guidance strength, denoising steps and guidance clipping. The selected configurations are shown in Table 4.

E.2.4 BASELINES

We compare against several classical and ML-based baselines.

DAISI. For DAISI (Andrae et al., 2026), we train a DAISI prior MLP with hidden size 256, depth 4, time embedding dimension 64, and SiLU activations. It is trained for 200 epochs with AdamW, learning rate 10^{-3}, weight decay 10^{-4}, and batch size 4096. We tune DAISI on SED objective.

Table 4: Selected RDA hyperparameters for the Lorenz–63 squared-observation experiment. Both settings use $K = 1$.

Hyperparameter	High frequency	Low frequency
Window length L	22	22
Denoising steps S	10	15
Guidance strength	1.5139	7.7711
Guidance clip	1.0	0

Table 5: Selected DAISI hyperparameters for Lorenz–63 squared observations.

Hyperparameter	High frequency	Low frequency
Guide method	MMPS	MMPS
Euler steps	100	50
Inverse steps	100	50
ϵ	0.03	0.03
Inverse ϵ	0.03	0.03
$t_{\min}$	0.3	0.3
Guidance strength	0.3023	0.5137
Guidance clip	0	0

SDA (filtering). We follow the sequential filtering adaptation of SDA described by Andrae et al. (2026). We trained SDA prior with an MLP of hidden size 256, depth 5, time embedding dimension 32, and SiLU activations. The prior was trained on nine-frame windows for 4096 epochs using AdamW, learning rate 10^{-3}, weight decay 10^{-3}, batch size 64, and a linear learning-rate schedule. At inference, we apply sequential sliding-window filtering with MMPS guidance, carrying an overlapping state from the previous analysis as a noisy initial-condition anchor and retaining the last frame as the filtering output. Sampling hyperparameters are tuned separately for each observation frequency on SED.

Table 6: Selected SDA filtering hyperparameters for Lorenz–63 squared observations. The anchor standard deviation is scaled relative to the ensemble spread of the carried state.

Hyperparameter	High frequency	Low frequency
Guidance method	MMPS	MMPS
Window length	9	9
Denoising steps	100	100
Guidance strength	2.3349	0.2278
MMPS solver iterations	1	3
Langevin correctors per step	1	0
Anchor standard-deviation multiplier	1.0	2.0
Guidance clip	0	0

ForcingDAS. We train a ForcingDAS (Jia et al., 2026) Lorenz prior using the official ForcingDAS framework adapted to the Lorenz squared-observation benchmark. The model uses a temporal transformer backbone with network size 128, 6 layers, 4 attention heads, feed-forward dimension 512, context length 15, and training sequence length 64. We train for $50{,}000$ optimization steps with batch size 64. We evaluate two variants: the original autoregressive ForcingDAS schedule and a rolling noise schedule variant using the same editable history-block schedule as our method.

For the autoregressive ForcingDAS variant, the context grows autoregressively during filtering. We found this variant substantially less stable and more expensive in the Lorenz squared-observation setting. Hyperparameters are tuned separately for each observation frequency on SED. The tuned high-frequency and low-frequency configuration is shown below.

FlowDAS. We train a conditional stochastic-interpolant forecaster following (Chen et al., 2025). The model generates the next Lorenz–63 state conditioned on the preceding seven states. Its drift

Table 7: Selected hyperparameters for the ForcingDAS-rolling variant.

Hyperparameter	High frequency	Low frequency
Sampling steps	25	25
Guidance strength	4.8569	7.1951
γ	0.001	0.03
Observation-noise multiplier	1.5	1.0
Guidance clip	1.0	0.0
Window length	15	15
Commit index	7	7
Future stride	7	10
Steps per commit	6	10

Table 8: Selected hyperparameters for the ForcingDAS-AR variant.

Hyperparameter	High frequency	Low frequency
Sampling steps	50	25
Guidance strength	0.3277	0.7738
γ	0.001	0.01
Observation-noise multiplier	0.5	0.25
Guidance clip	0.5	4.0
AR context window	100	100
Chunk size	1	1

network is an MLP with hidden width 256, five residual hidden layers, SiLU activations, and fixed Gaussian Fourier features with scale 30. The resulting model has approximately 1.78 million parameters. We train for 6000 epochs using Adam with learning rate 10^{-3}, batch size 64, and linear learning-rate decay. Batch normalization is disabled, and the checkpoint with the lowest validation loss is used. Hyperparameters are tuned separately for each observation frequency using the same validation sample and SED objective.

Table 9: Selected FlowDAS hyperparameters for Lorenz–63 squared observations.

Hyperparameter	High frequency	Low frequency
Conditioning history	7	7
Sampling steps	400	100
Guidance strength	0.002095	0.08300
MC guidance samples	1	21
Noise scale	0.5	0.5
Guidance clip	4.0	1.0
Observation-noise std.	0.265625	0.265625

EnKF. We use an ensemble Kalman filter (Evensen, 1994; 2003) with perturbed observations. The forecast step propagates each ensemble member with the Lorenz transition model. At observation times, we apply the nonlinear observation operator

$$h(\boldsymbol{x}) = \left(\frac{x_1 - \mu_1}{\sigma_1} \right)^2$$

inside the Kalman update. We use $J = 512$ ensemble members, observation noise $\sigma_{\mathrm{obs}} = 0.265625$ in normalized squared-observation units, no multiplicative inflation, and a warmup of 64 model steps before filtering.

ESRF. We use an ensemble square-root filter (Tippett et al., 2003) with the same forecast model, observation operator, observation noise, ensemble size, no-inflation setting, and warmup as EnKF. Unlike EnKF, ESRF uses a deterministic square-root analysis update rather than perturbed observations.

Table 10: Classical filtering baseline settings for Lorenz–63.

Setting	EnKF	ESRF
Ensemble size	512	512
Observation noise	0.265625	0.265625
Inflation	1.0	1.0
Warmup steps	64	64
Analysis update	perturbed obs.	deterministic square-root

EnSF. We use the ensemble score filter as a nonlinear filtering (Bao et al., 2024a) baseline. Since the original EnSF settings are not specified for this Lorenz squared-observation benchmark, we tune a small set of stability and score-correction hyperparameters on the same validation sample using SED as the objective. The sweep uses $J = 512$ particles, 1000 Euler steps, maximum correction horizon 500, pointwise clipped spread restoration, minimum spread clip 0.25, warmup 64, and seed 42. Hyperparameters are tuned separately for each observation frequency on SED.

Table 11: Selected EnSF hyperparameters for Lorenz–63 squared observations.

Hyperparameter	High frequency	Low frequency
ϵ_α	0.03	0.1
Score clip	1.0	4.0
Maximum spread clip	4.0	8.0
Euler steps	1000	1000
Maximum correction steps	500	500
Spread restoration	pointwise clipped	pointwise clipped
Minimum spread clip	0.25	0.25
Warmup steps	64	64

E.2.5 Evaluation

For evaluation, we use a bootstrap particle filter with 10^5 particles as the reference filtering posterior. We save marginal posterior quantiles at each time step and compute SED against these BPF quantiles. We also report the energy score against the ground-truth trajectory.

All trajectories have length $T = 1000$. We discard the first 100 time steps as burn-in and average metrics over the remaining 900 time steps. We report the mean and standard deviation over the nine held-out test samples.

E.2.6 Ablations on history length

We conduct ablations under low-frequency Lorenz–63 squared observations (stride 6). All configurations use $J = 512$ ensemble members, and seven future frames, giving a total window length $L = H + 8$. Sampling hyperparameters are tuned separately for each history length. We report mean $\pm$ standard deviation across nine held-out trajectories, after discarding the first 100 time steps.

Table 12: History-length ablation for low-frequency Lorenz–63 squared observations. The lowest mean in each metric is bolded.

History H	Window L	SED	ES
0	8	0.5896 ± 0.0637	6.6226 ± 0.3674
3	11	0.4411 ± 0.0468	6.5343 ± 0.2429
7	15	0.3517 ± 0.0397	$\mathbf{6.1824 \pm 0.3024}$
10	18	0.3485 ± 0.0348	6.4563 ± 0.2473
14	22	$\mathbf{0.3291 \pm 0.0429}$	6.1895 ± 0.2733
16	24	0.3302 ± 0.0494	6.3159 ± 0.3651
20	28	0.3480 ± 0.0543	6.3974 ± 0.5077

E.2.7 Ablations on number of denoising steps

We conduct ablations under low-frequency Lorenz–63 squared observations (stride 6). All configurations use $J = 512$ ensemble members, and 14 history frames and 7 future frames, giving a total window length $L = H + W + 1 = 22$. We report mean $\pm$ standard deviation across nine held-out trajectories, after discarding the first 100 time steps.

Table 13: Effect of denoising steps on low-frequency Lorenz–63 squared observations. All configurations use $H = 14$, $K = 1$, and $J = 512$. Values are mean $\pm$ standard deviation across nine held-out trajectories after discarding the first 100 time steps. The lowest mean in each metric is bolded.

Denoising steps S	SED	ES
2	0.3669 ± 0.0662	6.4279 ± 0.6904
5	$\mathbf{0.3239 \pm 0.0403}$	6.2369 ± 0.3078
10	0.3281 ± 0.0420	6.2271 ± 0.3102
15	0.3291 ± 0.0429	6.1895 ± 0.2733
20	0.3318 ± 0.0426	6.1849 ± 0.2885
40	0.3342 ± 0.0434	$\mathbf{6.1759 \pm 0.2872}$

E.2.8 Ablations on renoise/denoise cycles

We conduct ablations on different K under low-frequency Lorenz–63 squared observations (stride 6). All configurations use $J = 512$ ensemble members, and 14 history frames and 7 future frames, giving a total window length $L = H + W + 1 = 22$. We report mean $\pm$ standard deviation across nine held-out trajectories, after discarding the first 100 time steps.

Table 14: Effect of renoise/denoise cycles on low-frequency Lorenz–63 squared observations. All configurations use $H = 14$, $S = 15$ denoising steps per cycle, and $J = 512$ ensemble members. MMPS guidance strength is fixed to 1. Values are mean $\pm$ standard deviation across nine held-out trajectories after discarding the first 100 time steps.

Renoise/denoise cycles K	SED	ES
1	0.8432 ± 0.0815	7.0865 ± 0.3110
2	0.6863 ± 0.0790	6.8133 ± 0.1549
4	0.5044 ± 0.0583	6.4980 ± 0.1653
8	$\mathbf{0.3963 \pm 0.0418}$	$\mathbf{6.3858 \pm 0.1965}$
16	0.4652 ± 0.0820	6.4987 ± 0.5647

E.3 Surface Quasi-Geostrophic System

E.3.1 Dynamics and dataset

We consider a two-layer surface quasi-geostrophic (SQG) system following Tulloch & Smith (2009). The model evolves potential temperature on two horizontal boundaries, corresponding to the lower surface and an upper lid. The state at each time step is therefore a two-channel field $\boldsymbol{x}_n \in \mathbb{R}^{2\times 64\times 64}$, where the two channels represent potential-temperature anomalies on the two boundaries. The horizontal domain is periodic, and the velocity field is diagnosed from the boundary temperature fields through a nonlocal spectral inversion. This produces turbulent, multiscale dynamics while remaining computationally tractable for repeated data-assimilation experiments. The dataset contains 1598 training trajectories and a separate set of 400 validation trajectories. Each trajectory comprises 100 three-hourly timesteps of two state channels on a 64×64 grid.

We use SQG trajectories sampled every three physical hours. Each trajectory has length 100 time steps, giving a total assimilation time of 300 hours. For the reported data-assimilation results, we evaluate on 10 held-out trajectories.

The diffusion models are trained on normalized SQG states. The raw SQG fields have mean approximately zero and are normalized during training by the training standard deviation $\sigma_{\text{train}} = 2660$,

so that

$$\boldsymbol{w}_n = \frac{\boldsymbol{x}_n}{\sigma_{\text{train}}}. \tag{78}$$

For data assimilation and evaluation, samples are mapped back to the physical SQG units used by the observation model. All SQG observation operators, RMSE, and CRPS values are defined in physical units unless explicitly stated otherwise.

E.3.2 Rolling prior training

We train an RDA diffusion prior on normalized SQG trajectories windows. The model uses window length $L = 9$ and commit index $c = 4$, so that each window contains $H = 4$ history frames, one current frame and $W = 4$ future frames. Each frame has two channels and spatial resolution 64×64. The denoising network is a two-dimensional U-Net (Ronneberger et al., 2015) applied to the full trajectory window. Frames are stacked along the channel dimension before being passed to the U-Net, so the network processes the complete spatiotemporal block jointly. The architecture uses bases width 64, channel multiplier $(1, 2, 2)$, group normalization with 8 groups, learned sinusoidal time embeddings of dimension 32, and attention heads of dimension 64 with 4 heads. The model has approximately 5.5 million parameters.

We train the prior as a VP diffusion model with an ϵ-prediction objective. The per-frame diffusion times follow the editable-history rolling schedule described in Section 3.2, using the VP parameterization described in Appendix C. Training is performed on normalized SQG windows sampled from the training trajectories; no observations are used during prior training.

The model is trained for 2000 epochs using AdamW with learning rate 2×10^{-4}, weight decay 10^{-3}, batch size 16, and cosine learning-rate decay. We clip gradients to norm 1.0 during training. The checkpoint with the lowest validation loss is used for all SQG RDA data-assimilation experiments.

Table 15: Training configuration for the SQG RDA diffusion prior.

Hyperparameter	Value
Window length L	9
Commit index c	4
History / future frames	4 / 4
Objective	ϵ-prediction
Base U-Net width	64
Channel multipliers	$(1, 2, 2)$
Attention heads	4
Attention head dimension	64
Learned sinusoidal embedding dim.	32
Parameters	5.5M
Epochs	2000
Batch size	16
Optimizer	AdamW
Learning rate	2×10^{-4}
Weight decay	10^{-3}
Learning-rate schedule	cosine
Gradient clipping	1.0

E.3.3 Observation settings

Let $\boldsymbol{x}_n^{\text{phys}}$ denote the physical-unit SQG state at time n. Observations are generated as

$$\boldsymbol{y}_n = Ah(\boldsymbol{x}_n^{\text{phys}}) + \boldsymbol{\eta}_n, \qquad \boldsymbol{\eta}_n \sim \mathcal{N}(0, \sigma_{\text{obs}}^2 I), \tag{79}$$

where A is a fixed sparse spatial observation mask and h is the observation operator. The operator h is applied before adding Gaussian noise; therefore, for the nonlinear settings, the noise is added in observation space rather than to the state before applying the observation operator.

We consider three observation settings. In the sparse linear setting, $h(x) = x$, and 5% of the spatial grid points are observed. In the multimodal setting, we use the squared observation operator $h(x) = (x/7)^2$ and observe 25% of the grid points. In the saturating setting, we use $h(x) = \arctan(x)$ and observe 25% of the grid points. This is summarized in Table 16

Table 16: SQG observation settings. Observation noise is specified in physical observation units and is added after applying the observation operator.

Setting	Observation operator $h(x)$	Observed fraction	σ_{obs}	Seeds
Sparse	x	5%	1.0	mask 42, noise 42
Multimodal	$(x/7)^2$	25%	1.0	mask 42, noise 42
Saturating	$\arctan(x)$	25%	0.01	mask 42, noise 42

E.3.4 Data assimilation setup

All SQG methods are evaluated on ten held-out trajectories that are disjoint from the training trajectories. Each trajectory has length 100, and data assimilation is performed over the full trajectory using the corresponding saved observation cache from Table 16. The shared caches store the same held-out trajectory set for every method, with a fixed observation mask seed and noise seed. This ensures that all methods are compared under identical observations. All ensemble methods use $J = 20$ particles.

Initialization. RDA, DAISI, LETKF, and EnSF initialize their ensembles by perturbing the first truth state with Gaussian noise of standard deviation 3 in physical units. For RDA, each perturbed state is repeated across the initial window, after which the copies receive independent perturbations according to their frame-specific VP coefficients; no numerical spinup or external forecaster is used. SDA constructs its initial five-frame window by conditioning the diffusion prior on a noisy dense pseudo-observation of the first truth state (with standard deviation 1 in physical units) and the available observations. FA-APF uses the first truth frame as its one-frame conditioning state. FlowDAS uses the first six truth frames as conditioning history and the following truth frame as its initial state, with Gaussian perturbations of standard deviation 0.1 in physical units. Both ForcingDAS variants use the first ten truth frames as context. Since metrics are computed only over the final 20 frames, these startup frames are excluded from evaluation.

Metrics are computed on the final 20 frames of each held-out trajectory. We report the mean and standard deviation across the ten trajectories. RMSE and CRPS are computed after mapping samples back to physical SQG units, so all reported SQG metrics are in physical units.

E.3.5 Hyperparameter tuning

We tune RDA and some baseline's guidance hyperparameters with (Akiba et al., 2019). Hyperparameters are selected on validation trajectories that are disjoint from the held-out evaluation trajectories, and the final selected configuration is then evaluated once on the held-out shared-cache trajectories. The tuning objective is CRPS.

For all three SQG observation settings, the rolling prior is fixed to window length $L = 9$ and commit index $c = 4$. The selected hyperparameters are shown in Table 17.

Table 17: Selected RDA guidance hyperparameters for SQG. Low-pass cutoffs follow denoising order; equal values indicate a constant cutoff. The rolling prior is fixed across observation settings.

Hyperparameter	Sparse	Multimodal	Saturating
Guidance method	MMPS	MMPS	MMPS
Denoising steps	20	20	20
Guidance strength	22.0836	12.0306	15.9755
Guidance clip	1.0	4.0	0.1
Low-pass schedule	linear	constant	off
Low-pass cutoff (start $\rightarrow$ end)	$0.25 \rightarrow 1.0$	$0.4 \rightarrow 0.4$	–

E.3.6 BASELINES

All baselines use the same physical-space observations and observation masks as RDA filtering. Final results use $J = 20$ ensemble members and are evaluated with the common last-20-frame RMSE and CRPS.

SDA (filtering). We follow the sequential filtering adaptation of SDA described by Andrae et al. (2026). SDA is trained on the same SQG data, with a five-frame window and MMPS guidance. The initial window uses 80 denoising steps and two Langevin corrector steps per denoising step; subsequent analyses use 100 denoising steps and one corrector step, with $\tau = 0.5$ throughout. Only the guidance strength varies between observation settings. Their guidance parameters are tuned with Optuna (Akiba et al., 2019) on CRPS.

LETKF. LETKF propagates each ensemble member with the numerical SQG model and performs a localized ensemble-space analysis. It is commonly used in the filed of geosciences Wang et al. (2021). We use 20 members, truth-perturbed initialization, horizontal covariance localization, and relaxation-to-prior spread (RTPS). The localization radius and RTPS coefficient are given in Table 18, which are consistent to Andrae et al. (2026) and Liang et al. (2025).

EnSF. We evaluate the original ensemble score filter using 1000 Euler steps and $\epsilon_\alpha = 0.05$. These settings are fixed across all three observation operators. The setup is consistent to Liang et al. (2025); Andrae et al. (2026).

DAISI. We use the official DAISI implementation and the authors' released `daisi_64.pth` SQG checkpoint. The hyperparameters used follows Andrae et al. (2026). We use MMPS guidance with 100 forward Euler steps, 100 inversion steps, $\epsilon = 0.03$, $t_{\min} = 0.3$. We use the observation-specific guidance strengths reported by the authors. We additionally apply spatial low-pass filtering and guidance clipping to the MMPS correction, with setting chosen on validation data.

Table 18: Selected SQG baseline hyperparameters. Low-pass cutoffs are Fourier cutoff fractions at the first and last denoising steps; equal values indicate a constant cutoff. A guidance clip of 0 disables clipping. Clipping thresholds are method-specific and not directly comparable. ForcingDAS settings are reported in Table 19.

Method	Hyperparameter	Sparse	Multimodal	Saturating
SDA	Guidance strength	1.2920	1.0475	1.0912
SDA	Guidance clip	0	0	0
SDA	Low-pass schedule	off	off	off
SDA	Low-pass cutoff (first → last)	–	–	–
LETKF	Localization radius (km)	1500	1500	4500
LETKF	RTPS coefficient	0.5	0.5	0.9
EnSF	ϵ_α	0.05	0.05	0.05
DAISI	Guidance strength	1	1	20
DAISI	Guidance clip	0.1	0.5	0.1
DAISI	Low-pass schedule	linear	linear	linear
DAISI	Low-pass cutoff (first → last)	$1.0 \to 0.4$	$1.0 \to 0.4$	$0.7 \to 0.1$
FlowDAS	Guidance strength	3	0.5	5
FA-APF	Guidance strength	15.6277	14.8863	0.9629
FA-APF	Guidance clip	0.1	0.25	2.0
FA-APF	Low-pass schedule	constant	linear	constant
FA-APF	Low-pass cutoff (first → last)	$0.4 \to 0.4$	$1.0 \to 0.2$	$0.6 \to 0.6$

ForcingDAS. We train one causal diffusion prior and use it for both (Jia et al., 2026) variants. The prior uses a DiT backbone with hidden dimension 128, depth 8, four attention heads, and patch size 4. It is trained with the velocity-prediction objective for 60,000 optimization steps using batch size 4, learning rate 1.4×10^{-4}, weight decay 10^{-3}, and 2000 warmup steps. The model uses 100 diffusion sampling steps.

ForcingDAS-AR uses the original autoregressive inference schedule with chunk size one. ForcingDAS-rolling uses the same trained prior with an editable window of length 9, commit index 4, future stride 20, and 20 sampling steps per commit. Both variants use MMPS guidance. Their guidance parameters are tuned with Optuna (Akiba et al., 2019) on CRPS. We additionally apply spatial low-pass filtering and guidance clipping to the MMPS correction, with setting chosen on validaiton data, for both variants.

FlowDAS. We train three-hourly SQG FlowDAS (Chen et al., 2025) checkpoint with a window of seven frames, comprising six conditioning states and one generated state. At each assimilation time, FlowDAS generates the next state using Monte Carlo observation guidance with 500 Euler–Maruyama steps and 25 Monte Carlo samples. We use a noise scale of 0.2 and an initial perturbation standard deviation of 0.1 in physical units. The hyperparameters used follows the configurations in DAISI Andrae et al. (2026).

FA-APF. We implement FA-APF (Savary et al., 2026) using a conditional SQG diffusion model with EDM preconditioning (Karras et al., 2022) and a U-Net backbone. The conditional prior operates on normalized two-channel 64×64 states, with the clean conditioning state concatenated channel-wise with the noisy target. Its U-Net uses base width 64, channel multipliers $(1, 2, 4)$, two residual blocks per resolution level in the encoder and decoder, SiLU activations, group normalization with eight groups, a 256-dimensional Fourier noise embedding, and four-head attention at the bottleneck. Filtering starts from the first truth frame. Proposals use MMPS guidance with 32 noise levels, a third-order Adams–Bashforth predictor, and two Langevin corrector steps at each nonzero target noise level, with correction parameter $\delta = 0.2$. We use two MMPS solver iterations and systematic resampling with adaptive likelihood tempering. Their guidance parameters are tuned with (Akiba et al., 2019) on CRPS. We apply spatial low-pass filtering and guidance clipping to the MMPS correction, with setting chosen on validation data.

Table 19: Selected guidance hyperparameters for ForcingDAS-AR and ForcingDAS rolling. Both variants use the same trained prior. A low-pass cutoff of 0 denotes disabled filtering.

Variant	Hyperparameter	Sparse	Multimodal	Saturating
AR	Guidance strength	3.4941	8.4685	20.0084
AR	Guidance clip	2.0	2.0	4.0
AR	Low-pass schedule	constant	constant	disabled
AR	Low-pass cutoff (first $\rightarrow$ last)	$0.2 \rightarrow 0.2$	$0.2 \rightarrow 0.2$	0
Rolling	Guidance strength	19.3039	16.3081	23.0904
Rolling	Guidance clip	1.0	2.0	2.0
Rolling	Low-pass schedule	constant	constant	constant
Rolling	Low-pass cutoff (first $\rightarrow$ last)	$0.4 \rightarrow 0.4$	$0.2 \rightarrow 0.2$	$0.4 \rightarrow 0.4$

E.4 ERA5

ERA5 is a global atmospheric reanalysis that combines observations with numerical weather prediction to estimate past atmospheric states (Hersbach et al., 2020). We use six-hourly ERA5 fields on a 121×240 latitude–longitude grid. Following APPA (Andry et al., 2025), we use 11 of the ERA5 measurement variables (see Table 20) from the year 1993 to 2021 where the "Atmospheric"-type variables are measured at 13 pressure levels (50, 100, 150, 200, 250, 300, 400, 500, 600, 700, 850, 925, and 1000 hPa), resulting in 71 variables total. The training set ranges from 1993 to 2019, the validation set is 2020, and the test set is 2021. Each variable's measurements are normalized by its mean and standard deviation computed over all grid points and snapshots in the training set.

E.4.1 Model training

To scale training and inference of RDA to this high dimensional dynamical system, we follow APPA (Andry et al., 2025) and train an autoencoder to compress the states into a latent encoding of 128 channels on a 16×32 grid. A generative model prior (e.g., RDA) is trained on the dataset of latent encodings. The autoencoder and generative model together enable a latent data assimilation procedure.

Table 20: Variables used from ERA5. Snapshots of these are encoded using an autoencoder, and then a generative model (e.g., RDA) is trained on these latent snapshots to sample from their probability distribution. During training and inference, the generative model is conditioned on the time of day and elapsed year progress.

Measurement	**Type**
2m temperature	Near surface
10m U wind component	Near surface
10m V wind component	Near surface
Mean sea level pressure	Near surface
Sea surface temperature	Near surface
Total precipitation	Near surface
Temperature	Atmospheric
U wind component	Atmospheric
V wind component	Atmospheric
Geopotential	Atmospheric
Specific humidity	Atmospheric

Table 21: Training hyperparameters for the ERA5 autoencoder

Hyperparameter	**Value**
Batch size per GPU (4 total)	48
Initial learning rate	2×10^{-4}
Optimizer	SOAP (Vyas et al., 2025)
Learning rate schedule	linear decay
Gradient clipping	1.0
Training time	3,794 out of a target of 100,000 steps (6 hours)
Input, hidden, and output channels	$(71, 128, 256, 384, 512, 128)$
Latent noise regularization	0.01

Autoencoder We use the autoencoder architecture from APPA (Andry et al., 2025; Rozet et al., 2025) adapted for the ERA5 dataset resolution in this experiment. The encoder uses a convolutional neural network to progressively downsample the spatial resolution while expanding the number of channels until a 16×32 spatial resolution is reached. A final 1×1 convolutional layer projects the hidden channels to 128 output channels. The decoder is a convolutional neural network that reverses this compression. All convolutional layers have periodic padding for longitude. These models are trained jointly using a mean-square reconstruction error with a weighting dependent on latitude and pressure level (Lam et al., 2023). Hyperparameters are given in Table 21.

RDA and APPA (Latent SDA) Both generative models use the denoiser model architecture from APPA (Andry et al., 2025) adapted to the temporal and latent spatial resolution. The model is a Diffusion Transformer (DiT) (Peebles & Xie, 2023) that processes a window of snapshots. The snapshots are tokenized into tokens of shape $(1, 1, 1)$ where the first dimension is the snapshot index in the window, and the last two dimensions are spatial, and these are passed to the DiT. The generative models are trained using the rectified noise schedule (Liu et al., 2022) with $\sigma_{\min} = 0.001$ and $\sigma_{\max} = 1{,}000$ to predict the clean snapshot window. APPA noises each snapshot index uniformly whereas RDA follows the rolling diffusion snapshot noise schedule. Hyperparameters are given in Table 22.

E.4.2 Data assimilation setup

The observation operator design follows APPA (Andry et al., 2025). Observations for each snapshot come from ground weather stations, which measure near-surface variables, and satellites, which measure all atmospheric variables (see Table 20). The location of 11,000 weather stations provided by (NOAA National Centers for Environmental Information, 1999) are projected onto the 121×240 grid points to determine what points are observed, resulting in about 15% of grid points being

Table 22: Training hyperparameters for the ERA5 latent denoisers.

Hyperparameter	Value
Batch size per GPU (4 total)	32
Initial learning rate	5×10^{-4}
Optimizer	AdamW (Loshchilov & Hutter, 2019)
Learning rate schedule	One-cycle (Smith & Topin, 2019)
Gradient clipping	1.0
Gradient accumulation batch count	2
Training time	Two cycles of 3 epochs (12 hours total)
Window length (APPA)	8
Window length (RDA)	9
History/future frames	4/4
Commit index (RDA)	4

observed. For satellite observations, a satellite trajectory from the current time back one hour is computed; all grid points under the satellite's field of view (about 5%) along that trajectory are observed and all observed measurements are from the current time. The noise standard deviation for the observations is 10^{-1} and 10^{-2} for weather stations and satellites, respectively.

Initialization For RDA, the first frame of each ensemble member's window is initialized independently by sampling from a Gaussian centered at the the encoding of the true initial ERA5 state with standard deviation 0.01. The RDA window is then populated by copying the first frame forward to every other frame, and then each frame is noised according to its rolling diffusion noise schedule.

APPA does not need initialization because it samples the entire trajectory from noise simultaneously.

E.4.3 Hyperparameter tuning

Due to limited compute, we did not tune the window size or the history count. The hyperparameter tuning was focused on MMPS gradient guidance strength and clipping based on the root-mean-square magnitude of the guidance vector). We tuned these parameters using Optuna (Akiba et al., 2019) for about 50 trials to minimize the RMSE averaged over the last 9 snapshots of a 13-snapshot trajectories beginning at four validation start dates: 2020-01-21, 2020-03-21, 2020-06-21, and 2020-09-21. Optuna optimized the gradient guidance strength in $[1, 50]$ and the gradient RMS clipping in $[200, 600]$. Guidance strength values around 10, and RMS clipping between 200 and 400 performed best. The chosen hyperparmeter values are given in Table 23.

E.4.4 Other baselines

Climatological estimation This estimates the true state at any time of year with the mean of the same time of year across all years in the training set. For computing the CRPS, we treat each year in the dataset as an ensemble member.

Forecasting We compute forecasts using the APPA and RDA generative priors by setting the MMPS guidance strength to zero.

E.4.5 Evaluation

Each method is evaluated by performing assimilation to generate 96-snapshot (24-day) ensemble trajectories at four start dates 2021-01-21, 2021-03-21, 2021-06-21, and 2021-09-21. The RDA ensemble is initialized using the first snapshot, and each method has ten ensemble members. The inference settings are given in Table 23.

Table 23: Inference for ERA5 data assimilation

	Value	
Hyperparameter	**APPA**	**RDA**
MMPS guidance strength	1.0	10.05
MMPS guidance clipping RMS threshold	None	441.79
Sampler	Predictor(Euler)-Corrector(2 steps, 0.1 step size)	
Denoiser sampling steps	32	3 (per window shift)

E.5 FULL RESULTS

Table 24: RMSE on held-out SQG trajectories, averaged over the final 20 frames. Metrics are reported in physical units. All methods use 20 ensemble members and the same saved noisy observations in each observation setting. Best scores are bolded and second-best scores are underlined.

Method	Multimodal	Saturating	Sparse
LETKF	5.7188 ± 1.1453	9.6092 ± 0.4398	4.2364 ± 0.6191
SDA	6.5468 ± 0.5069	6.8023 ± 0.4789	3.7591 ± 0.3430
EnSF	9.0296 ± 1.0524	$\underline{2.6033 \pm 0.3773}$	7.3520 ± 0.7373
DAISI	$\underline{3.5535 \pm 0.4785}$	2.9296 ± 0.2518	$\underline{3.2248 \pm 0.3645}$
FlowDAS	7.8581 ± 0.5998	7.4516 ± 0.8651	6.1228 ± 0.646
ForcingDAS-AR	4.9794 ± 0.3560	6.0304 ± 0.2516	4.5144 ± 0.4169
ForcingDAS-Rolling	4.2552 ± 0.5518	3.6532 ± 0.3048	4.2293 ± 0.5578
FA-APF	8.2665 ± 0.7015	5.3952 ± 0.7094	6.9357 ± 0.4845
RDA	$\mathbf{3.2095 \pm 0.5113}$	$\mathbf{2.0139 \pm 0.2631}$	$\mathbf{3.0730 \pm 0.3794}$

Table 25: CRPS on held-out SQG trajectories, averaged over the final 20 frames. Metrics are reported in physical units. All methods use 20 ensemble members and the same saved noisy observations in each observation setting. Best scores are bolded and second-best scores are underlined.

Method	Multimodal	Saturating	Sparse
LETKF	3.0673 ± 0.6996	5.2281 ± 0.2086	2.1778 ± 0.3117
SDA	3.5628 ± 0.2985	3.7159 ± 0.2646	1.9259 ± 0.1675
EnSF	5.8158 ± 0.8277	$\underline{1.3496 \pm 0.1712}$	4.4494 ± 0.5151
DAISI	$\underline{1.6942 \pm 0.2130}$	1.5660 ± 0.1415	$\underline{1.6382 \pm 0.1690}$
FlowDAS	4.4680 ± 0.3601	4.1944 ± 0.5418	3.3458 ± 0.3709
ForcingDAS-AR	2.4828 ± 0.1856	2.1052 ± 0.1417	2.4017 ± 0.2198
ForcingDAS-Rolling	2.2260 ± 0.3083	1.9464 ± 0.1841	2.2866 ± 0.2978
FA-APF	5.1289 ± 0.4834	3.4642 ± 0.4575	4.2867 ± 0.3083
RDA	$\mathbf{1.6098 \pm 0.2005}$	$\mathbf{1.0172 \pm 0.1413}$	$\mathbf{1.6023 \pm 0.1825}$

Table 27: ERA5 trajectory average results with ten ensemble members on four held-out 96-snapshot trajectories. We report mean ± standard deviation across trajectories. APPA performs smoothing and RDA performs filtering.

	10 m U-wind [m/s]		10 m V-wind [m/s]		2 m temperature [K]		850 hPa temperature [K]	
Method	RMSE	CRPS	RMSE	CRPS	RMSE	CRPS	RMSE	CRPS
Trajectory average								
Climatology	4.14 ± 0.16	2.03 ± 0.08	4.19 ± 0.21	2.06 ± 0.11	3.26 ± 0.40	1.44 ± 0.15	3.99 ± 0.37	2.01 ± 0.16
APPA forecast	4.22 ± 0.21	2.10 ± 0.11	4.25 ± 0.21	2.11 ± 0.12	3.36 ± 0.47	1.52 ± 0.19	4.10 ± 0.44	2.12 ± 0.22
RDA forecast	4.26 ± 0.67	2.15 ± 0.31	4.15 ± 0.55	2.08 ± 0.25	3.71 ± 0.68	1.70 ± 0.28	4.07 ± 0.90	2.11 ± 0.46
APPA	1.92 ± 0.14	0.89 ± 0.06	1.91 ± 0.13	0.89 ± 0.05	1.68 ± 0.20	0.77 ± 0.07	1.78 ± 0.19	0.90 ± 0.07
RDA	2.26 ± 0.33	1.03 ± 0.13	2.20 ± 0.27	1.02 ± 0.12	1.84 ± 0.27	0.84 ± 0.10	2.20 ± 0.45	1.08 ± 0.18

Table 26: Wallclock evaluation time in minutes for one held-out SQG trajectory on an NVIDIA RTX 5000 Ada Generation GPU (32 GB).

Method	Multimodal	Saturating	Sparse
RDA	4.89	6.92	4.87
SDA	39.72	39.63	39.83
LETKF	2.09	3.36	0.90
EnSF	0.56	0.64	0.63
DAISI	27.29	27.23	27.21
FA-APF	12.55	12.12	11.90
FlowDAS	258.48	259.18	259.16
ForcingDAS-Rolling	13.73	13.48	9.62
ForcingDAS-AR	362.13	360.41	365.24

E.6 Additional Figures

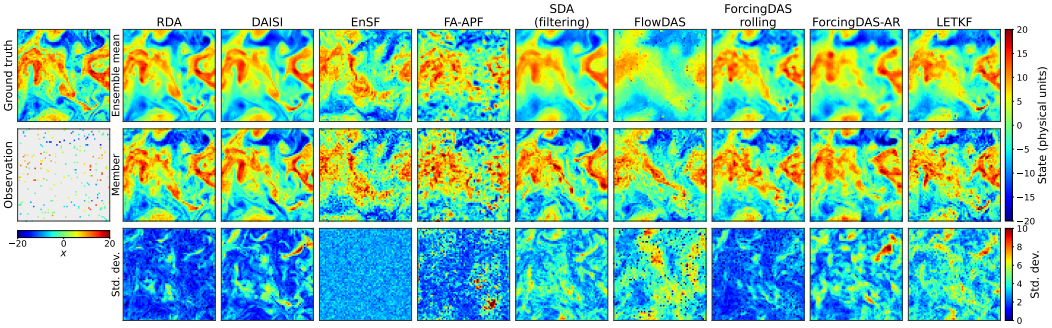


Figure 5: Filtering ensembles for the sparse SQG setting on held-out trajectory at assimilation step 100. The left panels show the ground truth and sparse noisy observations. For each method, the rows show the ensemble mean, one ensemble member, and the ensemble standard deviation. All methods use 20 ensemble members. We display channel 0 in physical units, with shared state and standard-deviation colour scales. Unobserved locations are shown in grey.

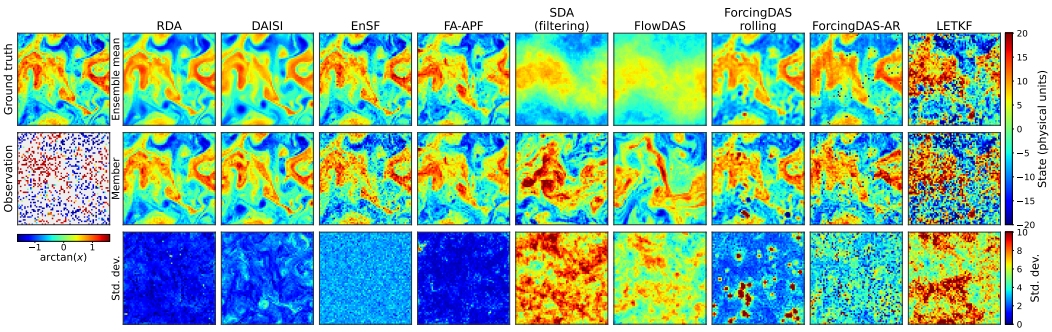


Figure 6: Filtering ensembles for the saturating SQG setting on held-out trajectory at assimilation step 100. The layout and colour scales follow Figure 5, with observations displayed in the transformed space $\arctan(x)$.

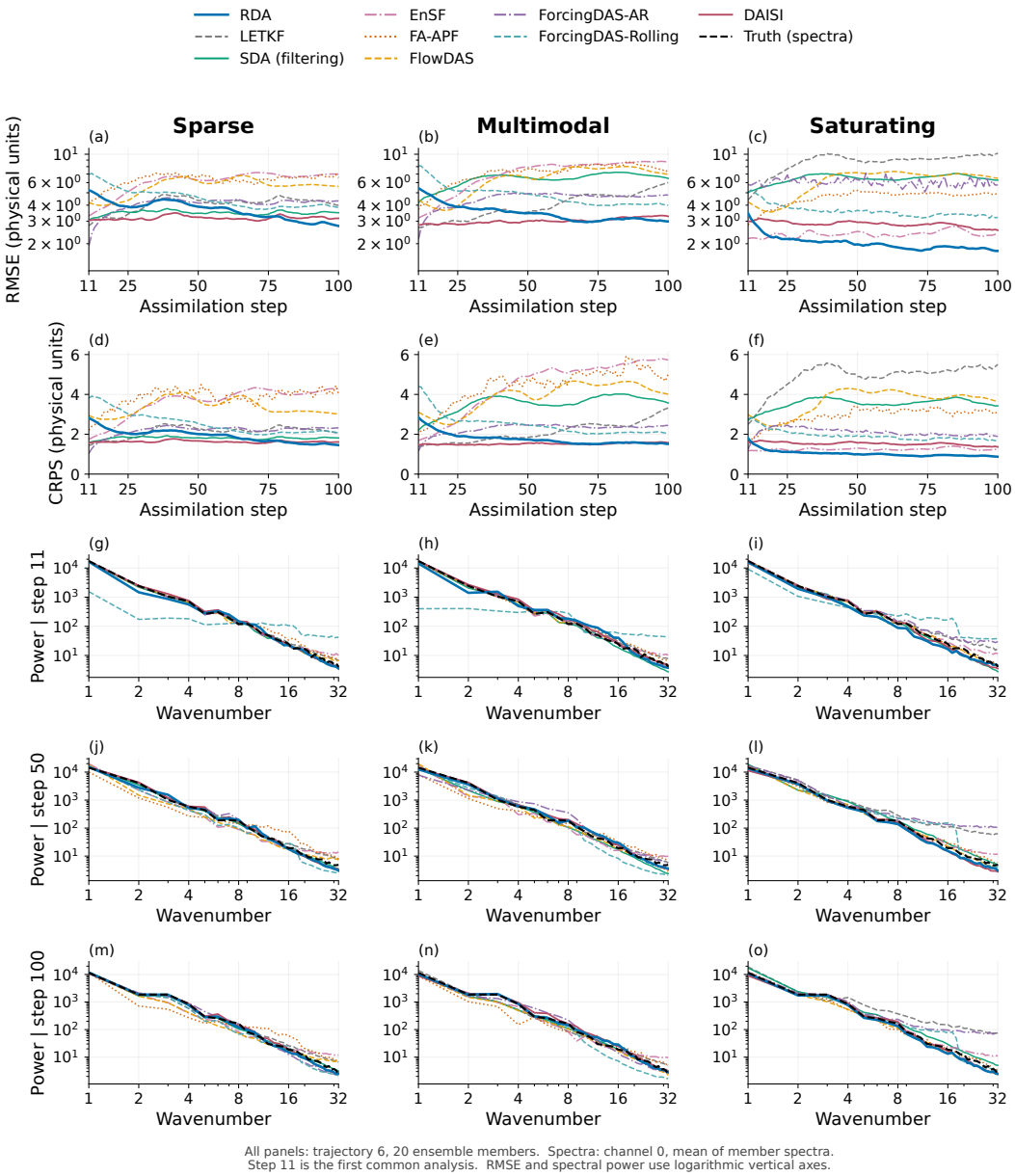


Figure 7: SQG filtering results on held-out trajectory. Columns correspond to sparse, multimodal, and saturating observations. The first two rows show RMSE and CRPS over time. The remaining rows show channel-0 power spectra at assimilation steps 11, 50, and 100, averaged over individual ensemble members, alongside the truth spectrum. All panels use the same trajectory and 20 ensemble members. Comparisons begin at step 11, after all methods' startup periods. RMSE and spectral power use logarithmic vertical axes.

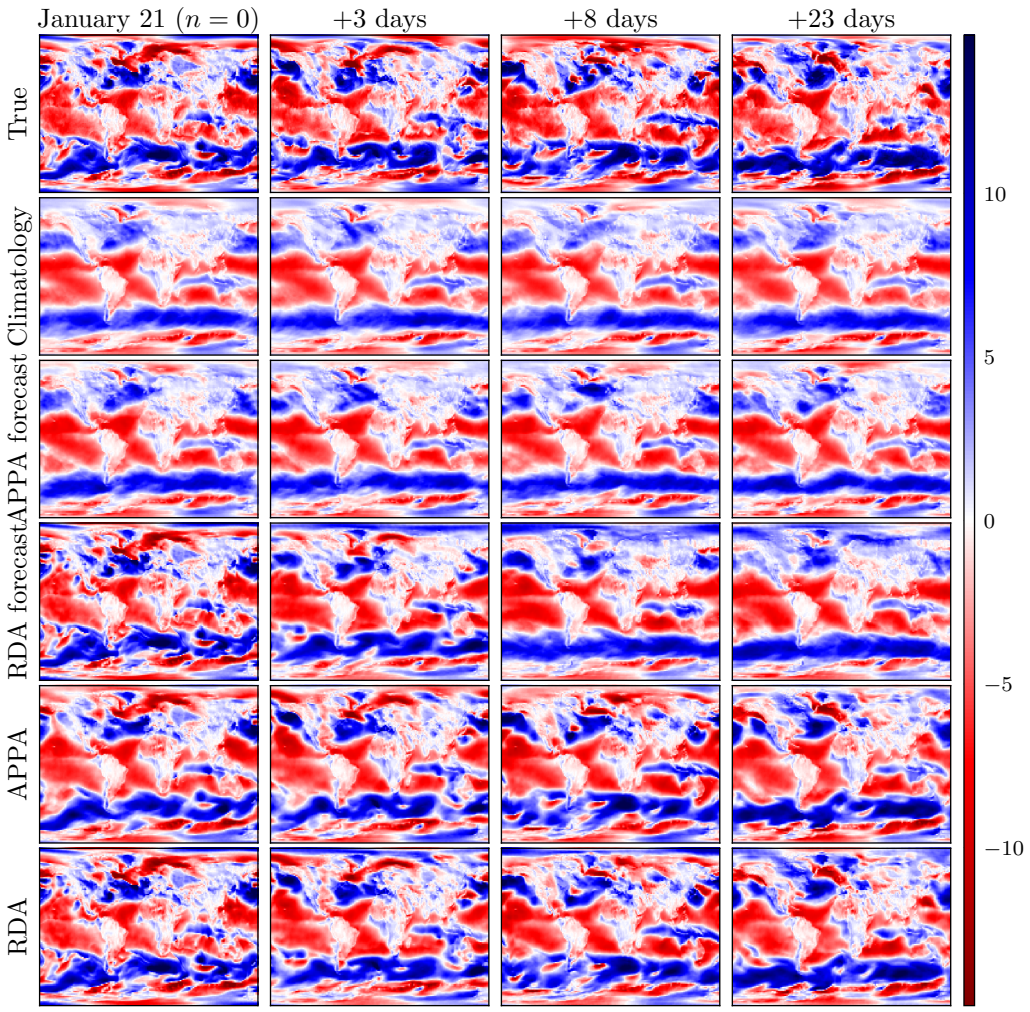


Figure 8: Snapshots of ERA5 U-wind [m/s].

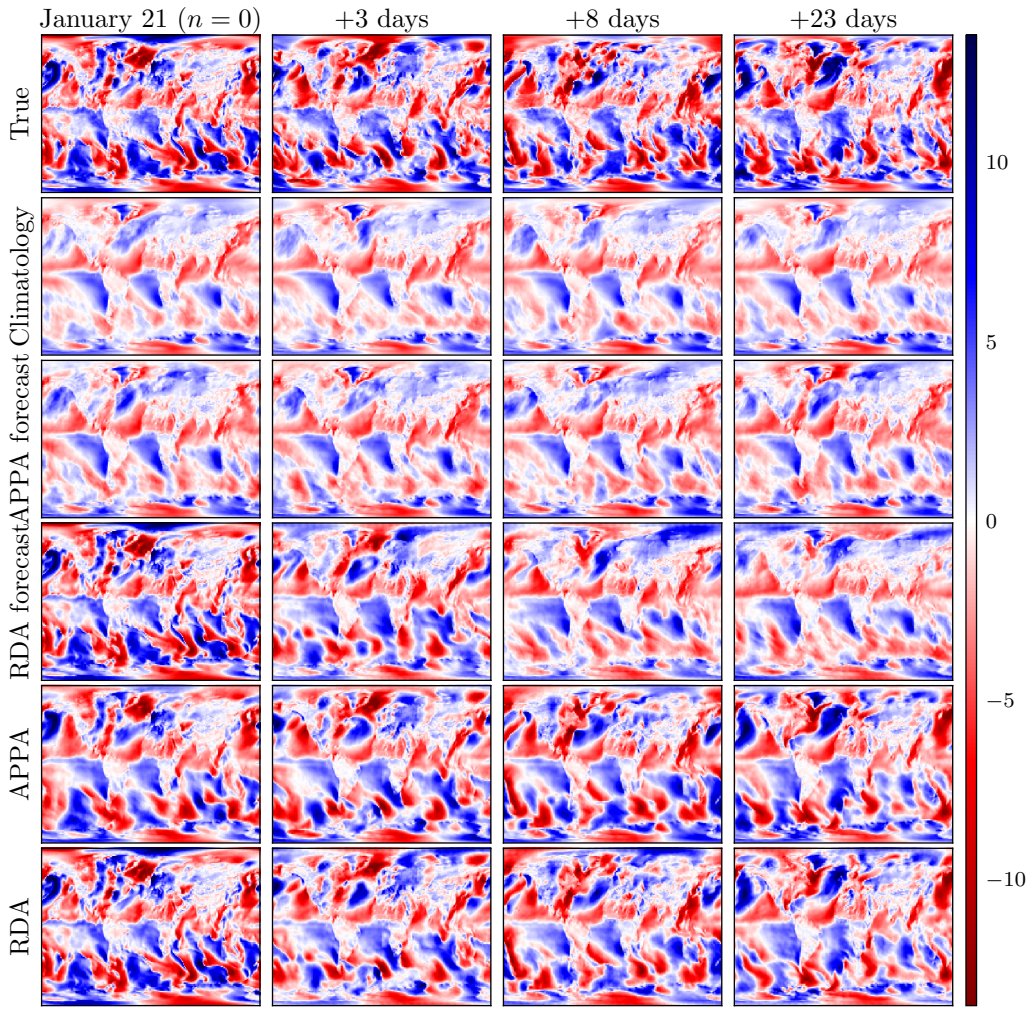


Figure 9: Snapshots of ERA5 V-wind [m/s].

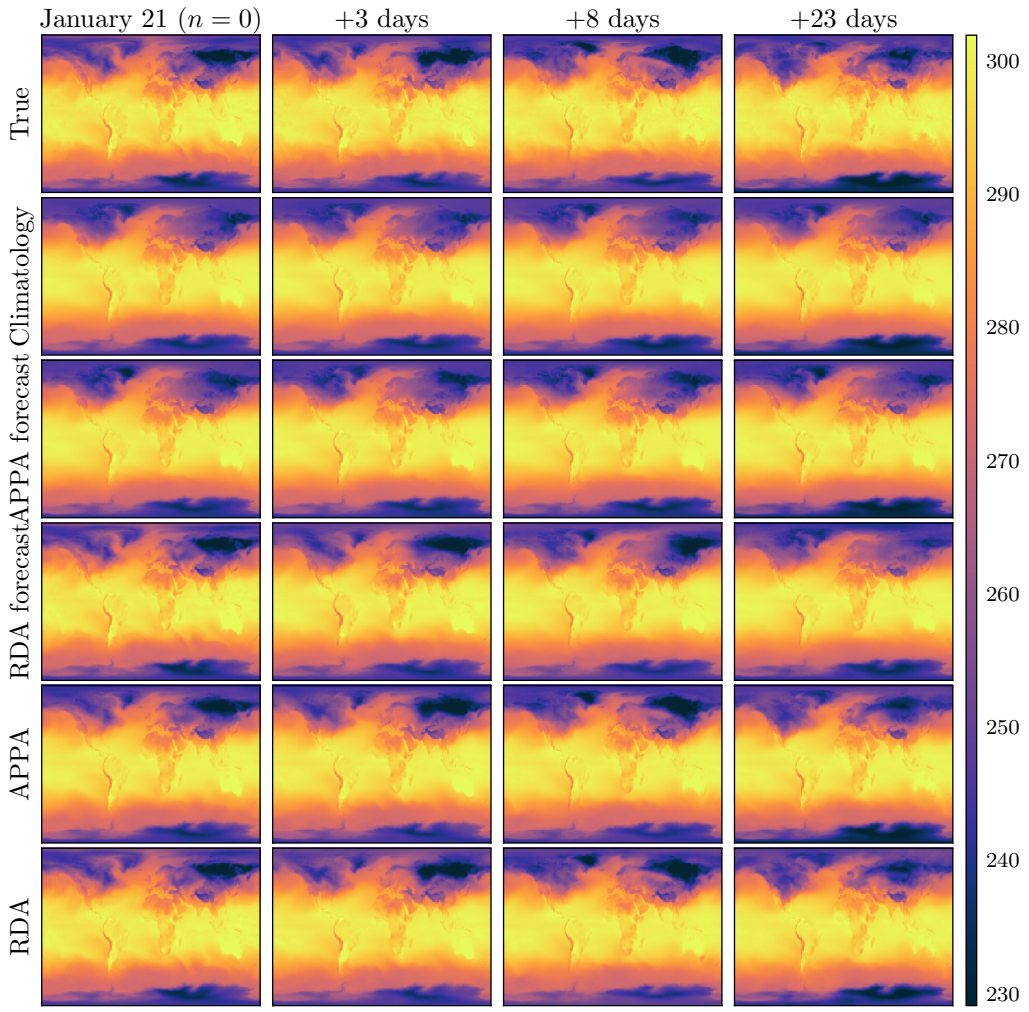


Figure 10: Snapshots of ERA5 2 m temperature [K].

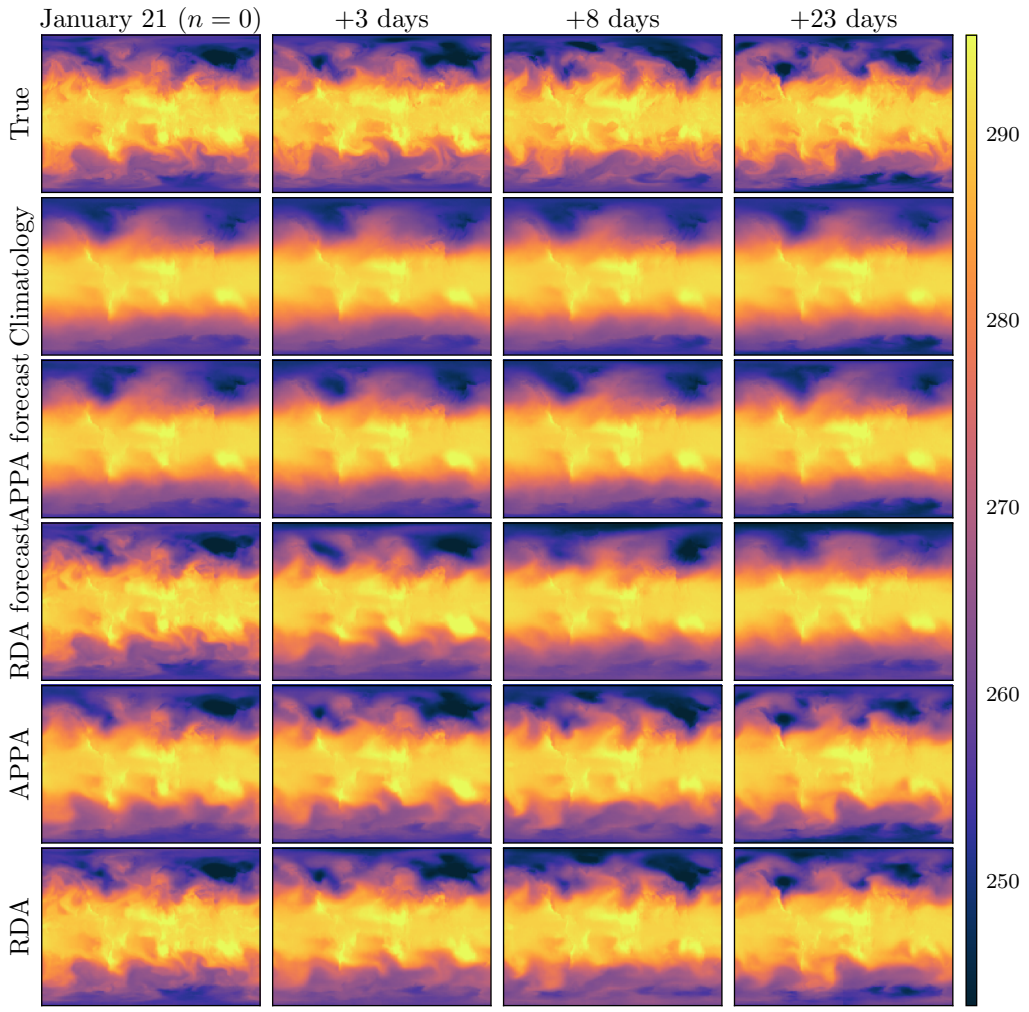


Figure 11: Snapshots of ERA5 temperature at 850 hPa [K].