Learning to Assimilate: Amortized Control for Autoregressive Diffusion

Prakhar Srivastava **Farrin Marouf Sofian** **Francesco Immorlano**
Kushagra Pandey **Stephan Mandt**
University of California, Irvine
{prakhs2,fmaroufs,fimmorla,pandeyk1,mandt}@uci.edu

Abstract

Autoregressive diffusion models (ARDMs) offer expressive priors for forecasting chaotic dynamics. However, maintaining consistency with sparse observations over long rollouts can require substantial test-time computation. We introduce Control-Augmented Data Assimilation (CADA), which trains a lightweight controller on top of a frozen ARDM to steer predictions toward future observations. Using a KL-regularized stochastic optimal control objective, the controller learns from short autoregressive rollouts to anticipate observations while staying close to the pretrained dynamics. At inference, it injects corrections at each denoising step without per-forecast optimization. Across two chaotic PDE benchmarks, CADA improves long-horizon accuracy, stability, and physical fidelity under sparse observations. On compact ERA5, it outperforms diffusion-based baselines in most regimes with over $10\times$ faster inference than conditional diffusion baselines. Applied to frozen GenCast, CADA also reduces temperature errors across 12–48-hour forecasts with spatially sparse observations. Code is available at https://github.com/mandt-lab/cada.

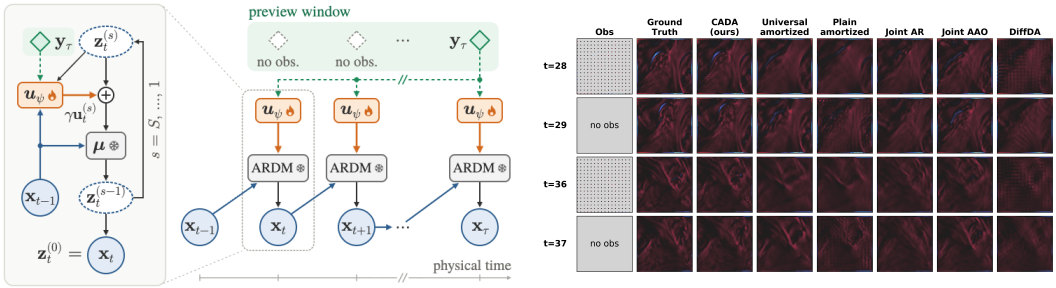
1 Introduction

Autoregressive diffusion models (ARDMs) are expressive models for high-dimensional spatiotemporal dynamics [1–3], including scientific and geophysical applications [4–7]. However, their reliance on one-step training with teacher forcing [8] leaves them prone to error accumulation over long rollouts, especially in chaotic systems where small deviations rapidly amplify [9–11]. Observations from sensors, weather stations, or satellites can correct these errors, yet incorporating sparse, intermittent measurements into a pretrained ARDM is challenging. This raises the question: *How can we steer long autoregressive trajectories toward observations without expensive test-time optimization?*

We study this problem through data assimilation (DA), where forecasts of evolving physical systems are constrained by noisy, incomplete measurements [9–12]. Classical DA approaches, including ensemble Kalman filters and variational methods [13–16], rely on approximations or optimization procedures that can be challenging to combine with stochastic, non-Gaussian ARDM priors. Because observations arrive intermittently, effective corrections must also influence unobserved steps preceding a measurement. Diffusion models offer expressive priors that can be adapted through observation-conditioned training [17] or inference-time guidance [18–21]. However, guidance is particularly expensive for ARDMs: correcting a prediction using an observation several steps ahead can require differentiating through an autoregressive rollout. Repeating this for every assimilation window incurs substantial test-time cost, motivating an approach that learns the corrections in advance.

Method Overview. We introduce Control-Augmented Data Assimilation (CADA), an amortized control framework that augments a frozen ARDM with a lightweight, observation-aware controller

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Overview (b) ERA5

Figure 1: **A learned controller steers a frozen autoregressive diffusion model toward upcoming observations.** (a) At each physical step, the controller $\boldsymbol{u}_\psi$ (orange, trained) shifts every denoising step of the frozen ARDM (gray) using the previous state $\mathbf{x}_{t-1}$ and the (possibly empty) preview window (green). (b) ERA5 500 hPa vorticity under MS-4 observations; rows are forecast times, columns show observations (gray: none; dots: sparse measurements), ground truth, and reconstructions. CADA stays closest to ground truth, including at unobserved steps (full comparison: Fig. 4).

(Fig. 1a). We formulate DA as a KL-regularized stochastic optimal control problem, balancing agreement with observations against deviations from pretrained dynamics. Instead of optimizing controls separately for each forecast, we train the controller offline through short autoregressive rollouts. At each physical step, it uses the noisy state, previous forecast, and a previewed observation to predict corrections injected into denoising steps. By learning how present corrections influence future observation consistency, CADA enables anticipatory guidance without changing ARDM parameters. At inference, assimilation reduces to a feed-forward rollout without per-instance optimization.

We evaluate CADA on two chaotic partial differential equations (PDEs), Kolmogorov flow and Kuramoto–Sivashinsky, and a compact ERA5 weather benchmark [22]. Across sparse and delayed observations, CADA improves long-horizon accuracy, temporal stability, and physical fidelity over diffusion-based baselines in most settings. On compact ERA5, it achieves over $10\times$ faster inference than conditional diffusion baselines. We also apply CADA to pretrained GenCast [23], reducing full-field temperature errors over 12–48-hour forecasts with sparse spatial observations.

Our main contributions are:

- **Control-based guidance for pretrained ARDMs.** We formulate observation-conditioned autoregressive generation as KL-regularized stochastic control, keeping the prior frozen.
- **Amortized, anticipatory corrections.** We learn a reusable controller on short windows, injecting observation-aware corrections at denoising steps without per-forecast optimization.
- **Accurate and efficient data assimilation.** We show improved long-horizon stability, physical fidelity, and efficiency on chaotic PDEs and weather benchmarks, including GenCast.

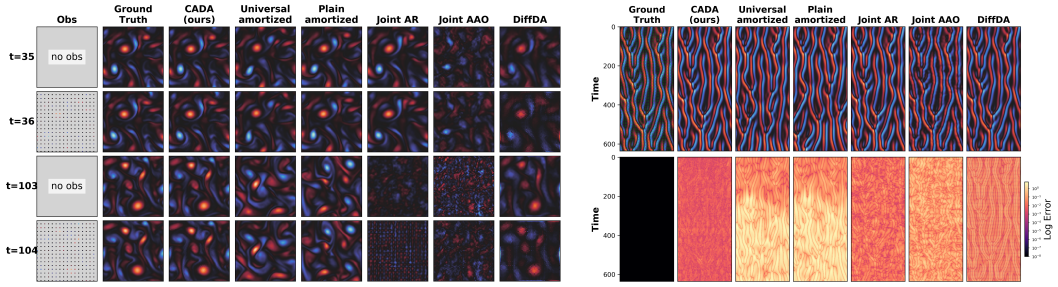
2 Control Augmented Data Assimilation

We now formalize the DA setting and develop the mathematical foundations underlying our approach.

2.1 Problem Statement: Chaotic Forecasting with Sparse Observations

Autoregressive forecasts of physical systems are vulnerable to error accumulation, especially under chaotic dynamics, where small deviations can rapidly amplify. In practice, such forecasts are stabilized by sequentially incorporating partial observations, a process known as DA. Our goal is to guide a pretrained autoregressive model toward forecasts that remain consistent with incoming sparse observations while reducing long-horizon drift.

We consider system states $\mathbf{x}_t$ at physical times $t \in \mathbb{N}$. Observations are available only at times $\mathcal{T} \subseteq \mathbb{N}$, denoted $\mathbf{y}_\mathcal{T} \triangleq \{\mathbf{y}_\tau\}_{\tau\in\mathcal{T}}$. In weather forecasting, for example, they may come from stations, radars, or satellites. We assume observations are noisy and partial, hence insufficient to recover the



(a) Kolmogorov Flow

(b) Kuramoto–Sivashinsky (KS)

Figure 2: **CADA stays accurate over long rollouts with sparse MS-4 observations.** (a) Kolmogorov vorticity at selected steps of a 180-step rollout (rows); columns show observations (gray: none; dots: sparse measurements), ground truth, and reconstructions. (b) KS space–time fields over 640 steps (vertical: time; horizontal: space); top: fields, with observations in green on the ground truth; bottom: log pointwise error (darker is lower). CADA preserves coherent structures with the least error.

full state $\mathbf{x}_t$. We are also given a pretrained autoregressive forecasting model

$$\mathcal{Q} \;=\; q_0(\mathbf{x}_0)\,\prod_{t\geq 1} q\big(\mathbf{x}_t \,\big|\, \mathbf{x}_{t-1}\big). \tag{1}$$

Here q_0 is the initial-state distribution; we later instantiate the transition q as a conditional diffusion model [24, 25] whose parameters stay fixed and are left implicit (Sec. 2.2). The objective is to incorporate $\mathbf{y}_{\mathcal{T}}$ into the forecast while keeping the pretrained prior $\mathcal{Q}$ fixed, i.e., without retraining it.

We need a cost function $\Phi(\mathbf{x}_\tau; \mathbf{y}_\tau)$ that measures the compatibility between a forecast state $\mathbf{x}_\tau$ and an observation $\mathbf{y}_\tau$ to optimally assimilate observations. For example, in a common observation model $\mathbf{y}_\tau = A(\mathbf{x}_\tau)$ (where A is some lossy data degradation operator such as blurring, masking, or spatial downsampling), one may take $\Phi = \|\mathbf{y}_\tau - A(\mathbf{x}_\tau)\|_2^2$ (more details on operators can be found in App. A). Equivalently, this cost induces a likelihood model $p(\mathbf{y}_\tau|\mathbf{x}_\tau) \propto \exp(-\frac{1}{\beta}\Phi(\mathbf{x}_\tau; \mathbf{y}_\tau))$, where $\beta > 0$ is a temperature parameter that controls tolerance to observation mismatch.

A principled way to assimilate observations is Bayesian inference. To this end, the optimal sampling distribution is the *tilted posterior* distribution that balances prior dynamics with observations:

$$\mathcal{P}^* \;\propto\; \mathcal{Q}\,\exp\!\left(-\frac{1}{\beta}\sum_{\tau\in\mathcal{T}}\Phi(\mathbf{x}_\tau; \mathbf{y}_\tau)\right). \tag{2}$$

In most cases of interest, this posterior distribution is intractable; however, notable exceptions exist in the literature. For example, classical DA methods such as Kalman filtering oftentimes assume that $\mathcal{Q}$ is a Gauss-Markov model and $A(\cdot)$ is linear [12], in which case the Bayesian updates are tractable. However, such assumptions severely limit the expressivity (e.g., multimodality) of $\mathcal{Q}$. Interestingly, $\mathcal{P}^*$ is the optimizer of the following variational problem,

$$\mathcal{C}(\mathcal{P}) \;\triangleq\; \sum_{\tau\in\mathcal{T}} \mathbb{E}_{\mathbf{x}_\tau\sim\mathcal{P}}\big[\Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau)\big] + \beta\, D_{\mathrm{KL}}\big(\mathcal{P} \,\|\, \mathcal{Q}\big) \tag{3}$$

(proof in App. B). This is a stochastic optimal control objective: a controlled process pays a β-weighted Kullback–Leibler (KL) cost for deviating from the uncontrolled dynamics $\mathcal{Q}$, the view that also underlies control-based guidance and fine-tuning of diffusion models [26, 27]. Restricting $\mathcal{P}$ to a tractable family yields a variational inference problem [28]. While one may sample directly from the unnormalized target in Eq. 2 [29, 30], we instead optimize Eq. 3 by learning guided autoregressive dynamics. We next define the unguided and guided dynamics (Sec. 2.2), how the guidance is learned (Sec. 2.3), and practical design choices (Sec. 2.4).

2.2 Diffusion-based Dynamics Modeling and Control

We use conditional ARDMs [1, 31, 23] and distinguish physical time indices $t \in \mathbb{N}$ (in subscripts) from diffusion denoising sub-steps $s \in \{S, \ldots, 1\}$ (in superscripts).

As in Sec. 2.1, let $\mathcal{Q} = q_0(\mathbf{x}_0)\prod_{t\geq 1} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ denote the pretrained autoregressive prior. Each transition $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ is defined as the marginal endpoint distribution of a conditional diffusion sampler:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \int \left[\prod_{s=1}^{S} q\left(\mathbf{z}_t^{(s-1)} \,\middle|\, \mathbf{z}_t^{(s)};\, \mathbf{x}_{t-1}\right)\right] q\left(\mathbf{z}_t^{(S)}\right)\, d\mathbf{z}_t^{(1:S)}. \tag{4}$$

The denoising transitions are Gaussian, $q(\mathbf{z}_t^{(s-1)} \mid \mathbf{z}_t^{(s)}; \mathbf{x}_{t-1}) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{z}_t^{(s)}, \mathbf{x}_{t-1}, s), \sigma_s^2 \boldsymbol{I}_d)$, where $\boldsymbol{\mu}$ is the pretrained denoiser, σ_s is the standard deviation of sub-step s, d is the state dimension, $\mathbf{z}_t^{(s)}$ are intermediate noisy states, and $q(\mathbf{z}_t^{(S)}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ is the initial noise prior. Thus, each autoregressive step samples $\mathbf{x}_t \equiv \mathbf{z}_t^{(0)}$ by running S denoising steps conditioned on $\mathbf{x}_{t-1}$. We refer to this process as the *unguided ARDM dynamics*.

As in Sec. 2.1, our goal is to steer the pretrained unguided process $\mathcal{Q}$ toward trajectories consistent with upcoming observations while remaining close to the original dynamics. We model the variational distribution $\mathcal{P}$ as an autoregressive process augmented with control variables $\boldsymbol{U}_t$:

$$\mathcal{P} \;=\; p_0(\mathbf{x}_0) \prod_{t\geq 1} p\big(\mathbf{x}_t \,\big|\, \mathbf{x}_{t-1}; \boldsymbol{U}_t\big). \tag{5}$$

Here $p_0 = q_0$, since controls act only on transitions. Analogously to $\mathcal{Q}$, each controlled transition is defined as

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}; \boldsymbol{U}_t) = \int \left[\prod_{s=1}^{S} p\Big(\mathbf{z}_t^{(s-1)} \,\Big|\, \mathbf{z}_t^{(s)}; \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}\Big)\right] q\left(\mathbf{z}_t^{(S)}\right) \mathrm{d}\mathbf{z}_t^{(1:S)}, \tag{6}$$

where $\boldsymbol{U}_t = (\mathbf{u}_t^{(1)}, \ldots, \mathbf{u}_t^{(S)})$ denotes the controls applied across denoising sub-steps. At each denoising sub-step, the control $\mathbf{u}_t^{(s)}$ perturbs the noisy input to the pretrained denoiser, yielding

$$p\Big(\mathbf{z}_t^{(s-1)} \,\Big|\, \mathbf{z}_t^{(s)}; \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}\Big) \triangleq \mathcal{N}\Big(\boldsymbol{\mu}\Big(\mathbf{z}_t^{(s)} + \gamma \mathbf{u}_t^{(s)},\, \mathbf{x}_{t-1},\, s\Big), \sigma_s^2 \boldsymbol{I}_d\Big), \tag{7}$$

where $\boldsymbol{\mu}$ is the fixed pretrained denoiser and $\gamma > 0$ controls the strength of the input shift. Thus, $\mathcal{P}$ preserves the pretrained denoising model while introducing additive controls in the noisy state space. Setting $\mathbf{u}_t^{(s)} \equiv \mathbf{0}$ recovers the unguided kernel, so $\mathcal{Q}$ is the special case of $\mathcal{P}$ without control.

Given controls $\boldsymbol{U}_t$, we can roll out the controlled autoregressive process, evaluate the observation costs along the trajectory, and differentiate through the rollout to optimize the variational objective in Eq. 3. We refer to this parameterization as a *guided ARDM*. We next describe how the controls are learned from upcoming observations over a fixed preview horizon.

2.3 Learning the controls

We learn the controls $\boldsymbol{U}_t$ by optimizing the variational objective in Eq. 3. A direct alternative is to treat $\{\mathbf{u}_t^{(s)}\}$ as free variables and optimize them separately for each assimilation window at test time, analogous to test-time scaling for diffusion models [18]. We refer to this ablation as *Test-Time Optimization based Data Assimilation (TTO-DA)*. Another inference-time alternative is *reconstruction guidance*, which adjusts each diffusion step using gradients of the observation cost, e.g., $\nabla_{\mathbf{x}}\Phi(\mathbf{x}; \mathbf{y})$, without a learned controller. However, without a direct estimate $\mathbb{E}[\mathbf{x}_\tau \mid \mathbf{z}_t^{(s)}]$ of the future state at an observation index $\tau > t$, both approaches require backpropagation through the ARDM rollout for each new forecast, making them prohibitively expensive for long-horizon assimilation.

We instead amortize the controls with a controller trained offline. During training, we roll out the guided ARDM in Eq. 5 and evaluate the observation costs in Eq. 3 at observation times τ. Since each controlled transition requires specifying $\boldsymbol{U}_t$, we parameterize the control at physical time t and denoising sub-step s as

$$\mathbf{u}_t^{(s)} = \boldsymbol{u}_\psi\big(\mathbf{z}_t^{(s)},\; \mathbf{x}_{t-1},\; \bar{\mathbf{y}}_t,\; s\big), \tag{8}$$

where $\mathbf{z}_t^{(s)}$ is the current noisy state, $\mathbf{x}_{t-1}$ is the previous forecast, and $\bar{\mathbf{y}}_t = (\mathbf{y}_{\tau_t}, \boldsymbol{M}_{\tau_t}, \Delta_{t,\tau_t})$ is the *preview*: the nearest observation $\tau_t \geq t$ in the current window, with its mask and lead time (null if none; App. C). We optimize only the controller parameters ψ, keeping the pretrained ARDM fixed.

Algorithm 1: Controlled Transition ($\mathbf{x}_{t-1} \to \mathbf{x}_t$)

Input: $\mathbf{x}_{t-1}$; future stream $\{\mathbf{y}_j\}_{j\in\mathcal{T}}$; horizon Λ; control scale γ
Output: Next state $\mathbf{x}_t$

1 **Context:** Build preview $\bar{\mathbf{y}}_t$ from $\{\mathbf{y}_j\}_{j\in\mathcal{T}}$ (App. C)
2 **Init:** Sample $\mathbf{z}_t^{(S)} \sim q(\mathbf{z}_t^{(S)})$
3 **for** $s = S, \dots, 1$ **do**
4 $\quad \mathbf{u}_t^{(s)} \leftarrow \boldsymbol{u}_\psi(\mathbf{z}_t^{(s)}, \mathbf{x}_{t-1}, \bar{\mathbf{y}}_t, s)$ // Eq. 8
5 $\quad \tilde{\mathbf{z}} \leftarrow \mathbf{z}_t^{(s)} + \gamma \mathbf{u}_t^{(s)}$ // Eq. 7
6 $\quad \mathbf{z}_t^{(s-1)} \leftarrow \textsc{DenoiseStep}(\tilde{\mathbf{z}}, \mathbf{x}_{t-1}, s)$ // DDIM
7 **return** $\mathbf{x}_t \leftarrow \mathbf{z}_t^{(0)}$

Highlighted lines 4–5 are the CADA additions: one controller call per sub-step predicts and injects the control; line 6 is the frozen ARDM's DDIM step. Full notation: Alg. 2.

Table 1: **CADA best preserves physical diagnostics.** Time-avg. abs. error (Eq. 9), MS-8: dissipation rate (Kolmogorov, 60 steps), total variation (KS, 640 steps).

Method	Diss. ↓	TV ↓
CADA	**2.8e-4**	**0.12**
Joint AAO	9.2e-3	3.98
Joint AR	9.0e-4	1.56
Plain amortized	5.7e-4	1.85
Universal amortized	8.5e-4	1.91
DiffDA	9.1e-3	2.70
TTO-DA	8.4e-3	3.12
BoN	1.0e-2	4.02

The amortized controller produces corrections with a single forward pass at test time. We call the full framework *CADA*.

CADA learns a reusable controller from short *preview windows*, so trajectory-level optimization is performed once offline rather than per forecast. At inference, assimilation reduces to rolling out the pretrained ARDM with feed-forward control corrections. Empirically, this amortized guidance substantially reduces inference cost while producing more stable and physically faithful DA trajectories than reconstruction guidance and TTO-DA baselines (see Sec. 4).

2.4 Training and Inference

In both training and inference, we assume access to the observation schedule $\mathcal{T}$ within a preview window of length Λ, together with the corresponding observations $\{\mathbf{y}_j\}_{j\in\mathcal{T}}$.

Forward pass through the preview. Given a preview window of length Λ, we sample an initial state and roll out the controlled dynamics in Eq. 6. At each physical step t in the window, we build the preview $\bar{\mathbf{y}}_t$, and the controller produces per-step denoising controls $\mathbf{u}_t^{(s)} = \boldsymbol{u}_\psi(\mathbf{z}_t^{(s)}, \mathbf{x}_{t-1}, \bar{\mathbf{y}}_t, s)$ as in Eq. 8, which are injected as noisy-state shifts in the reverse diffusion transitions in Eq. 7; Alg. 1 summarizes the resulting transition $\mathbf{x}_{t-1} \to \mathbf{x}_t$.

Training objective restricted to the preview. During training, we evaluate observation costs only at timesteps within the window that contain observations. The controller is optimized using only the information available in its preview context $\bar{\mathbf{y}}_t$. Let $\mathcal{P}_\psi(\mathbf{x}_{t_0+1:t_0+\Lambda} \mid \mathbf{x}_{t_0})$ denote the path distribution over a window of Λ steps induced by the controlled kernel with controls from $\boldsymbol{u}_\psi$, and let $\mathcal{T}^\Lambda = \mathcal{T} \cap [t_0{+}1,\, t_0{+}\Lambda]$ be the observation times in the window. We minimize the windowed variational objective $\mathcal{L}(\psi) = \sum_{\tau\in\mathcal{T}^\Lambda} \mathbb{E}_{\mathcal{P}_\psi}\big[\Phi(\mathbf{x}_\tau; \mathbf{y}_\tau)\big] + \beta\, D_{\mathrm{KL}}\big(\mathcal{P}_\psi(\cdot \mid \mathbf{x}_{t_0}) \,\|\, \mathcal{Q}(\cdot \mid \mathbf{x}_{t_0})\big)$, where $\mathcal{Q}(\cdot \mid \mathbf{x}_{t_0})$ is the corresponding unguided ARDM rollout. We approximate expectations with Monte Carlo rollouts, average the observation cost over the arrivals in $\mathcal{T}^\Lambda$ and also evaluate it on Tweedie estimates at intermediate denoising sub-steps, optimize ψ by stochastic gradient descent, and choose β by grid search. Since the marginal one-step KL is intractable, we use a tractable surrogate matched to our parameterization; see App. E. Alg. 3 in App. D gives the full training loop.

Inference by chunked re-anchoring. At test time, we apply the learned controller with a single forward pass per denoising sub-step, requiring no test-time optimization. We advance autoregressively and match the training setup by re-anchoring every Λ steps: the forecast horizon is partitioned into chunks of length Λ, and each preview context $\bar{\mathbf{y}}_t$ uses only observations scheduled within the current chunk. The final state of one chunk initializes the next. Alg. 4 in App. D specifies the full chunking and handoff procedure. We implement DENOISESTEP with DDIM [32], but the preview-and-control mechanism is sampler-agnostic and only requires access to diffusion sub-steps.

Table 2: **CADA attains the lowest RMSE in most regimes on Kolmogorov flow and Kuramoto–Sivashinsky (KS).** Columns: short/long rollouts (Kolmogorov 60/180, KS 140/640 steps) under DS-k and MS-k observations (Sec. 4). Rows: baselines and ablations (TTO-DA: no amortization; BoN: best of 16 unguided samples). Best in bold. HCT, ERA5, classical baselines: Tabs. 6, 3, 8.

	DS-2		**DS-4**		**DS-8**		**DS-16**		**MS-2**		**MS-4**		**MS-8**		**MS-16**	
	short	long	short	long	short	long	short	long	short	long	short	long	short	long	short	long
Kolmogorov flow																
CADA (ours)	**0.016**	**0.016**	**0.020**	**0.020**	**0.138**	0.351	0.428	0.638	**0.017**	**0.017**	**0.024**	**0.024**	**0.060**	**0.286**	0.424	0.641
Joint AAO	0.041	0.045	0.210	0.189	0.380	0.244	0.419	0.249	0.141	0.171	0.358	0.559	0.465	0.523	0.499	0.433
Joint AR	0.038	0.031	0.185	0.115	0.366	**0.218**	**0.413**	**0.245**	0.046	0.129	0.152	0.261	0.404	0.574	0.448	0.436
Plain amortized	0.109	0.814	0.229	1.033	0.712	1.276	0.799	1.175	0.245	0.454	0.302	0.477	0.316	0.479	**0.309**	0.471
Universal amortized	0.295	1.398	1.061	1.566	1.612	1.766	1.785	1.862	0.186	0.397	0.323	0.469	0.351	0.483	0.353	0.489
DiffDA	–	–	–	–	–	–	–	–	0.230	0.232	0.241	0.283	0.358	0.364	0.386	**0.391**
TTO-DA	0.040	0.243	0.027	0.115	0.156	0.401	0.491	1.280	0.078	0.298	0.113	0.357	0.215	0.433	0.526	0.611
BoN	0.258	0.420	0.264	0.440	0.299	0.442	1.759	1.992	0.265	0.447	0.266	0.454	0.306	0.488	0.846	2.089
Kuramoto–Sivashinsky																
CADA (ours)	**0.006**	**0.006**	**0.006**	**0.006**	**0.009**	**0.009**	**0.019**	**0.319**	**0.006**	**0.006**	**0.011**	**0.011**	**0.011**	**0.096**	**0.018**	**0.309**
Joint AAO	0.017	0.017	0.091	0.092	0.417	0.424	0.922	0.939	0.045	0.038	0.210	0.195	0.614	0.599	1.228	1.208
Joint AR	0.018	0.018	0.091	0.093	0.413	0.428	0.902	0.920	0.026	0.009	0.041	0.032	0.134	0.136	0.726	1.187
Plain amortized	0.041	9.787	0.146	10.73	1.859	11.81	5.979	12.830	0.034	1.163	0.036	1.165	0.039	1.211	0.034	1.189
Universal amortized	0.043	5.574	0.146	6.210	2.096	6.947	5.032	7.408	0.041	1.098	0.044	1.197	0.048	1.239	0.046	1.278
DiffDA	–	–	–	–	–	–	–	–	0.046	0.099	0.132	0.133	0.251	0.324	0.482	0.499
TTO-DA	0.016	8.288	0.009	0.053	0.418	0.634	0.930	2.613	0.016	0.298	0.114	0.363	0.081	0.580	1.573	3.559
BoN	0.046	1.257	0.046	1.498	0.048	3.122	2.827	4.598	0.045	1.987	0.046	1.644	0.049	2.128	1.241	1.948

3 Related Work

Guidance in Diffusion Models. Some existing works on guidance approximate the noisy likelihood score by estimating the diffusion posterior $q(\mathbf{z}^{(0)} \mid \mathbf{z}^{(s)})$ [33–37]. While this can result in accurate guidance and faster sampling, a large proportion of these methods are limited to linear inverse problems, which further limits their application. More recent works [26, 38] alleviate some of these problems by formulating guidance as optimal control. Our method amortizes the controls in a separate training stage in the framework of autoregressive diffusion models. There has also been recent work in fine-tuning diffusion models [27, 39–41] which is complementary to our proposed framework.

Data assimilation. Several recent works utilize diffusion models for DA. Rozet and Louppe [19] train score-based diffusion models on short trajectory segments to generate full long trajectories during inference. Their framework has been also applied in Qu et al. [20] and Manshausen et al. [21]. However, these approaches rely on inference-time-only guidance, lacking mechanisms that avoid error accumulation during observational gaps. To mitigate this, established practices utilize larger temporal windows of observations, as seen in classical 4D-Var [13] and iterative smoothers [42], which provide the multi-step constraints necessary to better constrain the physical manifold [43]. Autoregressive methods [17, 44, 45] improve stability but remain computationally expensive due to inference-time optimization. Latent space-based approaches [46, 47] reduce dimensionality but introduce reconstruction biases and latent-physical decoupling errors. We address these limitations by integrating learned feedback controls into autoregressive diffusion denoising, enabling inference as a single forward rollout with long-horizon stability and computational efficiency.

4 Experiments

We evaluate CADA on two chaotic PDE benchmarks, compact ECMWF Reanalysis v5 (ERA5), and larger-scale GenCast. Tests span six observation regimes, factor 16, nonlinear observations, and a one-step preview. We compare five diffusion-based baselines, two CADA ablations, and three classical methods. We ask: *(i)* Does preview-aware control reduce drift under sparse and delayed observations? *(ii)* Do gains persist over long chaotic rollouts? *(iii)* Are improvements due to amortized control rather than test-time optimization or brute-force search?

Datasets. We consider two canonical chaotic PDE benchmarks: the 1D KS equation and 2D Kolmogorov flow. Both exhibit nonlinear instabilities and long-range correlations, making them

challenging testbeds for sparse-observation DA [48, 49]. We additionally evaluate on a compact ERA5 assimilation setup and in a larger-scale GenCast weather study. The governing PDEs and data-generation protocols are detailed in App. F.

Observation Regimes. For each dataset, we evaluate six assimilation regimes designed to probe robustness to spatial sparsity and temporal delay, following settings inspired by Rozet and Louppe [19] and Shysheya et al. [44]. The first group, *downsampled* observations DS-k with $k \in \{2, 4, 8\}$, observes the full field at every simulator step, but only as averages over non-overlapping $k \times k$ blocks (App. A). The second group, *masked* observations MS-k with the same factors, observes the field only at every fourth simulator step and only at every k-th grid point along each spatial axis; all other steps and locations are unobserved. We extend both groups to factor 16, yielding a substantially tighter spatial information bottleneck without changing the temporal schedule. To test robustness beyond linear measurements, we also use the nonlinear masked operator $\mathbf{y} = (\boldsymbol{M} \odot \mathbf{x})^2 + 0.01(\boldsymbol{M} \odot \mathbf{n})$, with $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, at factors $\{2, 4, 8, 16\}$. The operator is applied in normalized model space and uses the same masks and four-step cadence as the linear MS setting. Squaring makes the observation map many-to-one by discarding sign information, while the additive noise prevents exact inversion even at observed locations. In all regimes, observation metadata, such as masks and operator parameters, are provided with each observation; see App. A.

Experimental Setup. For each observation regime, we train a separate *controller* on top of a fixed pretrained ARDM, using one ARDM per dataset. Architectural details for the ARDM and controller are given in App. G and App. H. CADA uses Denoising Diffusion Implicit Model (DDIM) sampling with $S{=}3$ denoising sub-steps per ARDM transition. The preview window is $\Lambda{=}16$ for Kolmogorov flow and ERA5, and $\Lambda{=}54$ for KS; within each anchored window, observations are selected by the nearest-arrival rule (App. C). The factor-16 and nonlinear experiments retain these preview lengths. We separately set $\Lambda{=}1$ for MS-2/4/8 while preserving the original four-step observation cadence, operator, forecast horizon, and pretrained prior. Consequently, only transitions whose one-step preview contains a scheduled arrival receive an observation; the remaining transitions use a null observation context. This design changes the amount of future information available to the controller without conflating preview length with observation frequency. We report both short- and long-horizon rollouts to separate near-term correction from long-term stability: 60/180 steps for Kolmogorov flow, 140/640 steps for KS, and 60 steps for ERA5.

Evaluation Metrics. We evaluate DA using trajectory-level and physics-aware metrics. Forecast accuracy is measured by time-averaged RMSE, and temporal stability by HCT. To assess physical fidelity beyond pointwise error, we compute a domain-specific diagnostic D, *total variation* (TV, $\int |\partial_\xi u| \, d\xi$) for KS and the *dissipation rate* ε for Kolmogorov flow (App. F), on every forecast state $\mathbf{x}_t$ and ground-truth state $\mathbf{x}_t^\star$ of an H-step rollout, and report the time-averaged absolute error

$$e_D \;=\; \frac{1}{H} \sum_{t=1}^{H} \big| D(\mathbf{x}_t) - D(\mathbf{x}_t^\star) \big|, \tag{9}$$

averaged over test trajectories.

Baselines. We compare against five diffusion-based DA baselines. *Joint AAO* [19] learns local joint scores over short segments under a k-order Markov factorization and samples trajectories *all-at-once*; although principled, it is memory-intensive for long-horizon stability at the cost of additional neural evaluations. We also include the conditional-score variants of Shysheya et al. [44]: *Plain amortized*, trained with a fixed conditioning length C, and *Universal amortized*, trained with $C \sim \mathrm{U}(\{0, \ldots, 2k\})$ (Markov order k as above) and masking to support any split of a fixed input window into C conditioning and P predicted states. Both conditional variants can be combined with reconstruction guidance for partial observations. *DiffDA* [17] conditions each denoising transition on a forecast state and sparse observations. Because it enforces observations through inpainting-style masks over observed grid columns, we report it only for masked (MS-k) regimes.

Classical baselines. On Kolmogorov flow, we additionally report *Ensemble Kalman Filter (EnKF)*, *three-dimensional variational assimilation (3D-Var)* [14], and *four-dimensional variational assimilation (4D-Var)* [15]. EnKF performs sequential ensemble updates using sample covariances; 3D-Var solves a per-arrival maximum-a-posteriori problem; and 4D-Var optimizes an assimilation window's initial state so its rollout matches all observations.

Table 3: **CADA is both the most accurate and the fastest on ERA5.** Left: RMSE under the DS-k and MS-k observation regimes defined in Sec. 4. Right: time to generate 8 trajectories (horizon 60) under MS-8, in seconds and relative to CADA. Best in bold.

	RMSE↓ on ERA5								**Time (MS-8)**	
	DS				MS					
	DS-2	DS-4	DS-8	DS-16	MS-2	MS-4	MS-8	MS-16	Time (s)	× CADA
CADA (ours)	**0.68**	**0.75**	**0.99**	**0.07**	1.23	**1.32**	**1.43**	**0.13**	**6.3**	**1.0**
Joint AAO	0.77	1.75	2.70	3.16	2.10	2.31	3.49	4.36	326.8	51.9
Joint AR	0.78	1.74	2.69	3.17	**0.98**	2.40	3.45	3.89	756.3	120.0
Plain amortized	3.23	3.31	3.69	4.60	1.61	2.10	3.03	3.68	63.4	10.1
Universal amortized	2.02	2.26	2.89	3.31	1.96	1.91	2.29	3.20	65.6	10.4
DiffDA	–	–	–	–	1.41	1.65	2.25	3.64	29.6	4.7
TTO-DA	2.78	2.77	2.81	4.82	1.39	1.73	2.20	5.79	230.4	36.6
BoN	3.27	3.96	3.97	5.96	3.91	3.91	4.34	6.23	25.8	4.1

Table 4: **CADA stays accurate under nonlinear observations and with a one-step preview.** Left: RMSE under the nonlinear operator (NL-MS-k; Sec. 4), CADA vs. the best baseline; better in bold. Right: CADA with preview Λ=1 (default 16; 54 for KS). Rollouts: 60 steps (640 for KS).

		Nonlinear observations				**$\Lambda = 1$**		
Dataset	**Method**	**NL-MS-2**	**NL-MS-4**	**NL-MS-8**	**NL-MS-16**	**MS-2**	**MS-4**	**MS-8**
Kolmogorov	**CADA (ours)**	**0.029**	**0.041**	**0.058**	**0.297**	0.041	0.041	0.095
	Best baseline	0.046	0.144	0.316	0.320	–	–	–
KS	**CADA (ours)**	**0.008**	**0.019**	0.847	**1.094**	0.009	0.021	0.024
	Best baseline	0.017	0.039	**0.154**	1.182	–	–	–
ERA5	**CADA (ours)**	**1.003**	**1.885**	**2.309**	**2.329**	1.312	1.327	1.507
	Best baseline	1.760	2.154	2.571	3.496	–	–	–

Ablations. (i) *TTO-DA* performs non-amortized test-time optimization of the controls under the same terminal-cost objective. In temporal DA, however, intermediate noisy states do not provide direct access to future clean-state estimates such as $\mathbb{E}[\mathbf{x}_\tau \mid \mathbf{z}_t^{(s)}]$. As a result, optimizing each denoising-step control requires a full autoregressive rollout to the relevant observation time τ, making TTO-DA prohibitively expensive. This ablation isolates the benefit of amortizing controls through the offline-trained controller $\boldsymbol{u}_\psi$. (ii) *Best-of-n (BoN)* samples n=16 independent latent seeds and selects the trajectory with the lowest terminal cost, providing a simple inference-time selection baseline inspired by contemporary alignment and scaling approaches [50, 51].

Qualitative and Quantitative Analysis. Tab. 2 compares CADA against state-of-the-art baselines and ablations. CADA performs best on most metrics and keeps RMSE nearly constant from short to long rollouts. The HCT results in Tab. 6 (App. I) show the same trend: CADA preserves high temporal correlations over full trajectories, whereas other ARDM-based methods, such as Plain and Universal amortized, degrade at long horizons and sometimes fail even in short rollouts. This stability is critical for operational DA, where forecasts must remain reliable over fixed-lag windows.

The factor-16 columns of Tab. 2 and Tab. 3 extend the standard linear operators to DS-16 and MS-16. Tab. 4 separates two further effects: robustness to a nonlinear observation map and sensitivity to the preview length Λ. These axes answer distinct questions. Factor 16 tests whether the controller can recover trajectories under an extreme spatial bottleneck; the nonlinear operator tests whether preview-aware control remains effective when observation compatibility is non-affine and sign-ambiguous; and Λ=1 removes anticipatory information while retaining the original sparse arrival process. All comparisons use the same rollout horizons and dataset-specific priors as the corresponding standard MS experiments, so differences isolate the observation operator or preview horizon.

The non-amortized control ablation, TTO-DA, performs worse than CADA, with the gap widening over long horizons (Kolmogorov MS-4: 0.113/0.357 vs. 0.024/0.024 for short/long rollouts). This suggests that optimizing controls only at test time and arrival points is insufficient for stable assimilation; learning a reusable controller over observations and preview windows is essential. The BoN

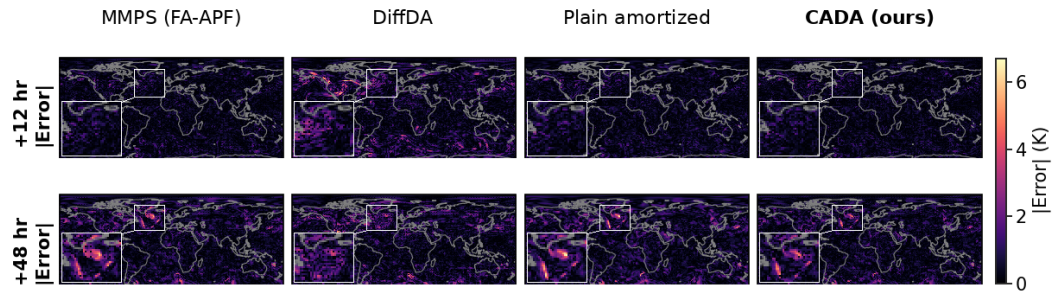


Figure 3: **CADA keeps GenCast forecast errors small and localized.** Absolute 400 hPa temperature error (K) at 12 h (top) and 48 h (bottom) under MS-4 observations; columns are methods. Insets magnify a North Atlantic storm whose error grows over time.

Table 5: **CADA attains the lowest GenCast temperature RMSE at every observation density.** Full-field RMSE averaged over the 4-step rollout (12–48 h leads), with observations at every step; rows are spatial strides (Sec. 4), columns are methods. Best in bold; the margin is largest at MS-4.

Observation mask	Unguided	MMPS (FA-APF)	DiffDA	Plain amortized	CADA (ours)
Spatial stride 4 (MS-4)	0.77	0.73	0.91	0.77	**0.65**
Spatial stride 8 (MS-8)	0.77	0.76	1.06	0.77	**0.71**
Spatial stride 16 (MS-16)	0.78	0.77	0.90	0.78	**0.74**

heuristic also underperforms (0.266/0.454 in the same setting): although best-of-n selection can help static generation, DA requires consistent stepwise correction rather than ex-post selection. These results highlight amortized control as the key mechanism for sequential inverse problems.

Fig. 2a and Fig. 2b visualize long-horizon DA under MS-4 observations for Kolmogorov flow (2D, horizon 180) and KS (1D, horizon 640). Although trained with finite preview windows (Λ=16 and 54, respectively), CADA remains stable on horizons more than $10\times$ longer than Λ.

Tab. 1 reports these physics-aware errors under MS-8 observations, over 640-step KS and 60-step Kolmogorov rollouts. TV measures the spatial roughness of the KS field, so a low TV error means the forecast keeps the ground truth's fine-scale structure instead of smoothing it out or adding spurious oscillations; CADA's error (0.12) is $13\times$ lower than the next best (Joint AR, 1.56). The dissipation rate is a standard diagnostic of turbulent energy transfer to small scales; CADA's error is $2\times$ lower than the next best (Plain amortized). Joint-score methods underestimate dissipation, while amortized ARDMs tend to overshoot it. Together, these results show that control augmentation not only reduces forecast drift but also improves domain-relevant physical fidelity.

In addition to improving accuracy and stability, CADA is substantially more efficient at inference. Tab. 3 reports wall-clock time for generating 8 ERA5 trajectories of 60 steps in the MS-8 regime. CADA is over $10\times$ faster than conditional ARDM baselines (Plain and Universal amortized) and 50–$120\times$ faster than joint-score methods (Joint AAO/AR). This efficiency follows from using a single controlled rollout of the pretrained ARDM, whereas guidance-only and joint-score methods require repeated score or gradient evaluations during sampling.

We further validate CADA in three additional settings. First, we evaluate a compact ERA5 weather setup: we train our own ARDM from scratch on a 64×64 North America patch using only (T, u, v) at 500 hPa from 2006–2016, and run the controller on top of this small prior. This shows that the controller transfers from synthetic PDEs to a realistic geophysical setting. Across most observation regimes, CADA achieves lower RMSE than conditional ARDM, joint-score baselines, and ablations (Tab. 3); Fig. 1b shows vorticity forecasts, and Fig. 4 (App. J) the full comparison. Second, classical EnKF, 3D-Var, and 4D-Var are competitive only under dense downsampled observations (3D-Var beats CADA at DS-8) but degrade sharply under masked, temporally strided ones, where CADA's error is 5–$15\times$ lower (Tab. 8, Fig. 5; App. J). Third, CADA also attains the lowest RMSE under randomized, non-grid-aligned spatial masks, without architectural changes (Tab. 7; App. J).

Large-Scale GenCast Assimilation. We additionally evaluate CADA in a larger-scale weather assimilation setting built on GenCast. Unlike in the compact ERA5 experiment, where we train a small prior from scratch, here the pretrained ARDM prior is GenCast [23] itself. The experiment uses 1° atmospheric fields at 12-hour cadence. The controller is trained on February 2019 data and evaluated on January–March 2022. GenCast is rolled out over its full set of prognostic fields, while only temperature (all 13 pressure levels) is observed, through sparse masks.

Observation setting and baselines. We use spatial MS-4/8 masked-observation regimes and include MS-16 as an extreme-sparsity follow-up, with observations available at every 12-hour forecast step. MS-4 follows the protocol of Savary et al. [52] (one grid point every 4° in latitude and longitude); MS-8 and MS-16 use 8° and 16° spacing. Unlike the PDE MS regimes, this isolates spatial sparsity: observations are not temporally strided. The controller is trained with 2-step preview windows and evaluated on 4-step rollouts, corresponding to forecast leads of 12, 24, 36, and 48 hours. We compare against unguided GenCast, DiffDA, Plain amortized guidance, and MMPS (FA-APF) [52], which combines a Monte Carlo posterior sampler with a fully adapted auxiliary particle filter.

Results. Tab. 5 reports average full-field temperature RMSE over the 4-step rollout. CADA achieves the lowest error across all three spatial strides. For stride 4, it reduces average RMSE by about 11% relative to MMPS (FA-APF) and 15% relative to unguided GenCast; at stride 16, it improves on the next-best method by about 4.6% (from unrounded values). MMPS (FA-APF) is consistently the strongest baseline, while Plain amortized remains close to unguided GenCast and DiffDA does not improve over the uncontrolled prior under this protocol. Although CADA's margin narrows as the mask becomes sparser, it remains best at MS-16. Fig. 3 shows selected 400 hPa absolute-error maps; full forecast and error maps are in Fig. 6 (App. K). The errors for CADA remain more localized, indicating that the learned controller can scale to a high-dimensional weather prior while using sparse observations from a subset of fields.

5 Conclusion

We presented CADA, an amortized control framework that recasts sparse-observation data assimilation as a single feed-forward rollout of a pretrained ARDM. A small, observation-aware controller is trained once to inject anticipatory corrections inside each denoising transition, replacing expensive test-time guidance, trajectory search, and adjoint-based optimization with a reusable policy.

Across two canonical chaotic PDEs and a compact ERA5 setup, CADA outperforms strong diffusion-based and classical DA baselines in most regimes, and most clearly under sparse, temporally strided observations, delivering more accurate forecasts, markedly better long-horizon stability, and closer agreement with domain-standard physical diagnostics. The same recipe applies without modification to a foundation-scale autoregressive weather model: a small controller trained on top of a frozen GenCast prior improves on both unguided GenCast and bespoke training-free DA methods designed for that prior (Sec. 4). Our ablations isolate amortization as the decisive ingredient: neither test-time optimization of the same objective nor best-of-n trajectory selection recovers comparable robustness, indicating that what makes controlled diffusion practical here is not the control objective itself but the fact that its solution is learned offline and reused.

More broadly, our results point to a simple recipe for sequential inverse problems with generative priors: cast assimilation as control over a frozen ARDM, and amortize that control. The GenCast experiment suggests this recipe is compatible with off-the-shelf foundation models for the dynamics, opening a lightweight path to assimilating sparse, delayed, or noisy observations into priors that need no retraining, from operational weather and climate to robotics and scientific simulation.

Limitations and scope. We study two chaotic PDEs, compact ERA5, and public GenCast, using one controller per observation regime. ARDM pretraining costs are comparable to existing diffusion-based PDE surrogates, while controllers add modest overhead and can use a frozen foundation prior. The GenCast study is limited to temperature observations and four-step forecasts, and therefore does not test longer operational cycling or heterogeneous observing systems. Moreover, each controller inherits the support and biases of its frozen prior: control can correct trajectories toward available observations, but cannot supply unresolved dynamics absent from the underlying model. Multivariate observations, cross-regime controllers, adaptive previews, and richer operators are natural extensions.

Acknowledgments and Disclosure of Funding

Stephan Mandt acknowledges funding from the National Science Foundation (NSF) through an NSF CAREER Award IIS-2047418, IIS-2611549, the NSF LEAP Center, the Chan Zuckerberg Initiative, SAP, and the Hasso Plattner Research Center at UCI.

References

[1] Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation. *Entropy*, 25(10):1469, 2023.

[2] Lijun Yu, Jose Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Agrim Gupta, Xiuye Gu, Alexander G Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A Ross, and Lu Jiang. Language model beats diffusion - tokenizer is key to visual generation. In *The Twelfth International Conference on Learning Representations*, 2024.

[3] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[4] Jaideep Pathak, Yair Cohen, Piyush Garg, Peter Harrington, Noah D. Brenowitz, Dale R. Durran, Morteza Mardani, Arash Vahdat, Shaoming Xu, Karthik Kashinath, and Michael S. Pritchard. Kilometer-scale convection allowing model emulation using generative diffusion modeling. *CoRR*, abs/2408.10958, 2024.

[5] Morteza Mardani, Noah Brenowitz, Yair Cohen, Jaideep Pathak, Chieh-Yu Chen, Cheng-Chin Liu, Arash Vahdat, Mohammad Amin Nabian, Tao Ge, Akshay Subramaniam, Karthik Kashinath, Jan Kautz, and Mike Pritchard. Residual corrective diffusion modeling for km-scale atmospheric downscaling. *Communications Earth & Environment*, 6(1), February 2025. ISSN 2662-4435.

[6] Prakhar Srivastava, Ruihan Yang, Gavin Kerrigan, Gideon Dresdner, Jeremy McGibbon, Christopher Bretherton, and Stephan Mandt. Precipitation downscaling with spatiotemporal video diffusion. In *Advances in Neural Information Processing Systems*, volume 37, pages 56374–56400, 2024.

[7] Kushagra Pandey, Jaideep Pathak, Yilun Xu, Stephan Mandt, Michael Pritchard, Arash Vahdat, and Morteza Mardani. Heavy-tailed diffusion models. In *International Conference on Learning Representations*, 2025.

[8] Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, June 1989. ISSN 1530-888X.

[9] Eugenia Kalnay. *Atmospheric Modeling, Data Assimilation and Predictability*. Cambridge University Press, 2002.

[10] Alberto Carrassi, Marc Bocquet, Laurent Bertino, and Geir Evensen. Data assimilation in the geosciences: An overview of methods, issues, and perspectives. *WIREs Climate Change*, 9(5): e535, 2018.

[11] Geir Evensen, Femke C Vossepoel, and Peter Jan Van Leeuwen. *Data assimilation fundamentals: A unified formulation of the state and parameter estimation problem*. Springer Nature, 2022.

[12] Daniel Sanz-Alonso, Andrew Stuart, and Armeen Taeb. *Inverse Problems and Data Assimilation*. London Mathematical Society Student Texts. Cambridge University Press, 2023.

[13] François-Xavier Le Dimet and Olivier Talagrand. Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus A: Dynamic Meteorology and Oceanography*, 38(2):97–110, 1986.

[14] Philippe Courtier, E Andersson, W Heckley, D Vasiljevic, M Hamrud, A Hollingsworth, F Rabier, M Fisher, and J Pailleux. The ecmwf implementation of three-dimensional variational assimilation (3d-var). i: Formulation. *Quarterly Journal of the Royal Meteorological Society*, 124(550):1783–1807, 1998.

[15] F. Rabier, H. Järvinen, E. Klinker, J.-F. Mahfouf, and A. Simmons. The ecmwf operational implementation of four-dimensional variational assimilation. i: Experimental results with simplified physics. *Quarterly Journal of the Royal Meteorological Society*, 126(564):1143–1170, 2000.

[16] Yannick Tr'emolet. Accounting for an imperfect model in 4d-var. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 132(621):2483–2504, 2006.

[17] Langwen Huang, Lukas Gianinazzi, Yuejiang Yu, Peter Dominik Dueben, and Torsten Hoefler. DiffDA: a diffusion model for weather-scale data assimilation. In *Forty-first International Conference on Machine Learning*, 2024.

[18] Masatoshi Uehara, Yulai Zhao, Chenyu Wang, Xiner Li, Aviv Regev, Sergey Levine, and Tommaso Biancalani. Inference-time alignment in diffusion models with reward-guided generation: Tutorial and review, 2025.

[19] François Rozet and Gilles Louppe. Score-based data assimilation. *Advances in Neural Information Processing Systems*, 36:40521–40541, 2023.

[20] Yongquan Qu, Juan Nathaniel, Shuolin Li, and Pierre Gentine. Deep generative data assimilation in multimodal setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 449–459, June 2024.

[21] 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.

[22] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna De Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia de Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume, and Jean-Noël Thépaut. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

[23] 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(8044): 84–90, 2025. ISSN 1476-4687.

[24] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[25] Kushagra Pandey and Stephan Mandt. A complete recipe for diffusion generative models. In *Proceedings of the International Conference on Computer Vision*, pages 4261–4272, 2023.

[26] Kushagra Pandey, Farrin Marouf Sofian, Felix Draxler, Theofanis Karaletsos, and Stephan Mandt. Variational control for guidance in diffusion models. In *Forty-second International Conference on Machine Learning*, 2025.

[27] Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *The Thirteenth International Conference on Learning Representations*, 2025.

[28] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

[29] Francisco Vargas, Will Sussman Grathwohl, and Arnaud Doucet. Denoising diffusion samplers. In *The Eleventh International Conference on Learning Representations*, 2023.

[30] Lorenz Richter and Julius Berner. Improved sampling via learned diffusions. In *The Twelfth International Conference on Learning Representations*, 2024.

[31] Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. *Advances in neural information processing systems*, 36:45259–45287, 2023.

[32] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[33] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2022.

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

[35] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. *Advances in Neural Information Processing Systems*, 35:23593–23606, 2022.

[36] Kushagra Pandey, Ruihan Yang, and Stephan Mandt. Fast samplers for inverse problems in iterative refinement models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[37] Ashwini Pokle, Matthew J. Muckley, Ricky T. Q. Chen, and Brian Karrer. Training-free linear image inverses via flows. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

[38] Litu Rout, Yujia Chen, Nataniel Ruiz, Abhishek Kumar, Constantine Caramanis, Sanjay Shakkottai, and Wen-Sheng Chu. RB-modulation: Training-free stylization using reference-based modulation. In *The Thirteenth International Conference on Learning Representations*, 2025.

[39] Kevin Clark, Paul Vicol, Kevin Swersky, and David J. Fleet. Directly fine-tuning diffusion models on differentiable rewards. In *The Twelfth International Conference on Learning Representations*, 2024.

[40] Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Reinforcement learning for fine-tuning text-to-image diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[41] Kushagra Pandey, Farrin Marouf Sofian, Jan Niklas Groeneveld, Felix Draxler, and Stephan Mandt. Hierarchical variational policies for reward-guided diffusion, 2026. URL https://arxiv.org/abs/2605.21661.

[42] M. Bocquet and P. Sakov. An iterative ensemble kalman smoother. *Quarterly Journal of the Royal Meteorological Society*, 140(682):1521–1535, 2014. doi: https://doi.org/10.1002/qj.2236. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.2236.

[43] Suping Nie, Xiaolong Jia, Weitao Deng, Yixiong Lu, Dongyan He, Liang Zhao, Weihua Cao, and Xueliang Deng. The influence of fy-4a high-frequency lst data on data assimilation in a climate model. *Remote Sensing*, 15(1), 2023. ISSN 2072-4292. doi: 10.3390/rs15010059. URL https://www.mdpi.com/2072-4292/15/1/59.

[44] Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard Turner, and Emile Mathieu. On conditional diffusion models for pde simulations. *Advances in Neural Information Processing Systems*, 37:23246–23300, 2024.

[45] Han Gao, Xu Han, Xiantao Fan, Luning Sun, Li-Ping Liu, Lian Duan, and Jian-Xun Wang. Bayesian conditional diffusion models for versatile spatiotemporal turbulence generation. *Computer Methods in Applied Mechanics and Engineering*, 427:117023, 2024.

[46] Ehsan Foroumandi and Hamid Moradkhani. Harnessing generative deep learning for enhanced ensemble data assimilation. *Water Resources Research*, 61(7):e2025WR040078, 2025. e2025WR040078 2025WR040078.

[47] Hang Fan, Lei Bai, Ben Fei, Yi Xiao, Kun Chen, Yubao Liu, Yongquan Qu, Fenghua Ling, and Pierre Gentine. Physically consistent global atmospheric data assimilation with machine learning in a latent space. *Science Advances*, 12(1):eaea4248, 2026.

[48] Yifan Du and Tamer A Zaki. Evolutional deep neural network. *Physical Review E*, 104(4): 045303, 2021.

[49] Phillip Lippe, Bas Veeling, Paris Perdikaris, Richard Turner, and Johannes Brandstetter. Pderefiner: Achieving accurate long rollouts with neural pde solvers. *Advances in Neural Information Processing Systems*, 36:67398–67433, 2023.

[50] Raghav Singhal, Zachary Horvitz, Ryan Teehan, Mengye Ren, Zhou Yu, Kathleen McKeown, and Rajesh Ranganath. A general framework for inference-time scaling and steering of diffusion models. In *Forty-second International Conference on Machine Learning*, 2025.

[51] Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023.

[52] Thomas Savary, François Rozet, and Gilles Louppe. Training-free data assimilation with gencast, 2025.

[53] Johannes Brandstetter, Max Welling, and Daniel E Worrall. Lie point symmetry data augmentation for neural pde solvers. In *International Conference on Machine Learning*, pages 2241–2256. PMLR, 2022.

A Observation Operators

Our experiments use linear observation operators A that map a full state $\mathbf{x}$ to an observed signal $\mathbf{y}$. These operators define the compatibility costs $\Phi(\mathbf{x}_\tau; \mathbf{y}_\tau)$ used at observation times in Eq. 2.

Masked observations. For spatially sparse observations, including temporally strided regimes when measurements arrive only at selected times, we use a binary mask $\boldsymbol{M} \in \{0,1\}^{1\times D}$, where D is the number of spatial grid points, broadcast across all channels of the state (so d is D times the number of channels):

$$A_{\text{mask}}(\mathbf{x}) \;=\; \boldsymbol{M} \odot \mathbf{x}, \qquad \Phi^{\text{mask}}(\mathbf{x}; \mathbf{y}) \;=\; \frac{\|\boldsymbol{M} \odot \mathbf{x} - \mathbf{y}\|_2^2}{\|\boldsymbol{M}\|_1}.$$

The normalization by $\|\boldsymbol{M}\|_1$ makes the cost comparable across different sparsity levels.

Downsampled observations. For coarse-resolution observations, we apply average pooling P_f over non-overlapping $f\times f$ blocks, followed by nearest-neighbor upsampling U_f back to the model grid:

$$A_{\downarrow f}(\mathbf{x}) \;=\; U_f(P_f\mathbf{x}), \qquad \Phi^{\text{ds}}(\mathbf{x}; \mathbf{y}) \;=\; \|U_f(P_f\mathbf{x}) - \mathbf{y}\|_2^2.$$

These operators generate the observed signals $\mathbf{y}_\tau$ and the corresponding arrival-time costs used throughout the assimilation objective. Although our experiments focus on masking and downsampling, the framework can incorporate any differentiable observation operator and associated cost without modifying the controlled ARDM.

B Proof of the Tilted Distribution

We show that the optimization problem in Eq. 3 admits the tilted distribution in Eq. 2 as its optimal solution.

Setup. Recall the objective

$$\mathcal{C}(\mathcal{P}) \;=\; \sum_{\tau\in\mathcal{T}} \mathbb{E}_{\mathbf{x}_\tau\sim\mathcal{P}}\Big[\Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau)\Big] + \beta\, D_{\text{KL}}(\mathcal{P} \,\|\, \mathcal{Q}),$$

where $\mathcal{P}$ is the guided CADA distribution over trajectories, $\mathcal{Q}$ the unguided distribution, and Φ an arrival-time cost.

Variational form. Expanding the KL divergence,

$$D_{\text{KL}}(\mathcal{P} \,\|\, \mathcal{Q}) \;=\; \mathbb{E}_{\mathcal{P}}\left[\log \frac{\mathcal{P}}{\mathcal{Q}}\right].$$

Thus the objective reads

$$\mathcal{C}(\mathcal{P}) \;=\; \mathbb{E}_{\mathcal{P}}\left[\sum_{\tau\in\mathcal{T}} \Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau) + \beta \log \frac{\mathcal{P}}{\mathcal{Q}}\right].$$

Lagrangian minimization. Consider minimizing $\mathcal{C}(\mathcal{P})$ over distributions $\mathcal{P}$ subject to normalization $\int \mathcal{P} = 1$. The corresponding Lagrangian is

$$\mathfrak{L}(\mathcal{P}, \lambda) \;=\; \mathbb{E}_{\mathcal{P}}\left[\sum_{\tau\in\mathcal{T}} \Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau) + \beta \log \frac{\mathcal{P}}{\mathcal{Q}}\right] + \lambda\Big(\int \mathcal{P} - 1\Big).$$

Stationary point. Taking the functional derivative w.r.t. $\mathcal{P}$ gives

$$\frac{\delta\mathfrak{L}}{\delta\mathcal{P}} \;=\; \sum_{\tau\in\mathcal{T}} \Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau) + \beta\Big(1 + \log \tfrac{\mathcal{P}}{\mathcal{Q}}\Big) + \lambda.$$

Setting this derivative to zero yields

$$\log \mathcal{P} \;=\; \log \mathcal{Q} - \tfrac{1}{\beta} \sum_{\tau\in\mathcal{T}} \Phi(\mathbf{x}_\tau;\, \mathbf{y}_\tau) - \tfrac{\lambda+\beta}{\beta}.$$

Closed form. Exponentiating both sides gives

$$\mathcal{P}^* \;\propto\; \mathcal{Q}\exp\left(-\tfrac{1}{\beta}\sum_{\tau\in\mathcal{T}}\Phi(\mathbf{x}_\tau;\,\mathbf{y}_\tau)\right),$$

which is exactly the tilted distribution in Eq. 2.

C Active Observation Selector

We maintain a preview buffer containing observations in $\mathcal{T}$ that fall within a preview window of length Λ anchored at t_0. Each observation entry is represented by its observed signal and metadata, $(\mathbf{y}_j, \boldsymbol{M}_j)$, where $j \in \mathcal{T}$ is the physical observation time, $\mathbf{y}_j$ is the observed signal lifted to the model resolution when needed, and $\boldsymbol{M}_j$ is an auxiliary mask as in App. A. For non-masking operators, $\boldsymbol{M}_j$ may be omitted or replaced by the corresponding operator metadata.

For transition $\mathbf{x}_{t-1} \to \mathbf{x}_t$, we define the set of future observations available within the anchored preview window as

$$\mathcal{T}_t^{\Lambda} \;\triangleq\; \{\, j \in \mathcal{T} : t \le j \le t_0{+}\Lambda \,\}.$$

The active preview is chosen as the nearest future observation in this set:

$$\tau_t = \arg\min_{j\in\mathcal{T}_t^{\Lambda}} (j - t), \qquad \Delta_{t,\tau_t} = \tau_t - (t-1),$$

and the preview context passed to the controller is

$$\bar{\mathbf{y}}_t = (\mathbf{y}_{\tau_t},\, \boldsymbol{M}_{\tau_t},\, \Delta_{t,\tau_t}).$$

Thus, at each step, the selector provides the controller with the nearest previewed observation inside the current anchored window, together with its metadata and lead time. Details on how this selector is used during training and inference are given in App. D. When $\mathcal{T}_t^{\Lambda} = \emptyset$, the selector returns a null preview context with empty metadata; the controller receives no active observation, and the observation cost is skipped until the next available arrival.

D Training and Sampling Algorithm

Algorithm 2: Preview-aware controlled DDIM step ($\mathbf{x}_t \to \mathbf{x}_{t+1}$)

Input: Current state $\mathbf{x}_t$; preview $\bar{\mathbf{y}}_{t+1} = (\mathbf{y}_{\tau_{t+1}}, \boldsymbol{M}_{\tau_{t+1}}, \Delta_{t+1,\tau_{t+1}})$; pretrained ARDM kernels q; controller $\boldsymbol{u}_\psi$; control scale $\gamma > 0$

Output: Next state $\mathbf{x}_{t+1}$ and arrival-time cost ℓ_{t+1}

Sample parent latent $\mathbf{z}_{t+1}^{(S)} \sim q(\mathbf{z}_{t+1}^{(S)})$
$\ell_{t+1} \leftarrow 0$
for $s = S, S-1, \ldots, 1$ **do**
  $\mathbf{u}_{t+1}^{(s)} \leftarrow \boldsymbol{u}_\psi\left(\mathbf{z}_{t+1}^{(s)}, \mathbf{x}_t, \bar{\mathbf{y}}_{t+1}, s\right)$
  $\tilde{\mathbf{z}} \leftarrow \mathbf{z}_{t+1}^{(s)} + \gamma\, \mathbf{u}_{t+1}^{(s)}$ `// latent shift`
  if $t+1 \in \mathcal{T}$ **then**
    $\widehat{\mathbf{x}}_{t+1}^{(s)} \leftarrow \mathbb{E}\left[\mathbf{z}_{t+1}^{(0)} \mid \tilde{\mathbf{z}}\right]$ `// Tweedie estimate`
    $\ell_{t+1} \leftarrow \ell_{t+1} + \Phi\left(\widehat{\mathbf{x}}_{t+1}^{(s)}; \mathbf{y}_{t+1}\right)$
  $\mathbf{z}_{t+1}^{(s-1)} \leftarrow \textsc{DenoiseStep}_q(\tilde{\mathbf{z}}, \mathbf{x}_t, s)$ `// DDIM step`
$\mathbf{x}_{t+1} \leftarrow \mathbf{z}_{t+1}^{(0)}$
if $t+1 \in \mathcal{T}$ **then**
  $\ell_{t+1} \leftarrow \ell_{t+1} + \Phi(\mathbf{x}_{t+1}; \mathbf{y}_{t+1}) + \beta\, D_{\mathrm{KL}}$ `// For the KL surrogate, see App. E`
return $\mathbf{x}_{t+1}, \ell_{t+1}$

Algorithm 3: Training the controller network

Input: Pretrained ARDM kernel q; observation stream $\{\mathbf{y}_j\}_{j\in\mathcal{T}}$; preview horizon Λ; regularization strength $\beta > 0$; optimizer for ψ; controller $\boldsymbol{u}_\psi$; control scale $\gamma > 0$

Output: Trained controller parameters ψ

repeat
  Sample rollout start t_0 and initial state $\mathbf{x}_{t_0} \sim p_0$
  $\mathcal{T}^\Lambda \leftarrow \mathcal{T} \cap [t_0 + 1,\, t_0 + \Lambda]$
  $\widehat{\mathcal{C}} \leftarrow 0$
  for $t = t_0, t_0 + 1, \ldots, t_0 + \Lambda - 1$ **do**
    $\mathcal{T}^\Lambda_{t+1} \leftarrow \{\, j \in \mathcal{T} : t + 1 \le j \le t_0 + \Lambda \,\}$ `// Anchored preview; App. C`
    Build $\bar{\mathbf{y}}_{t+1}$ from $\mathcal{T}^\Lambda_{t+1}$ `// App. C`
    $(\mathbf{x}_{t+1}, \ell_{t+1}) \leftarrow \textsc{ControlledStep}(\mathbf{x}_t, \bar{\mathbf{y}}_{t+1}, q, \boldsymbol{u}_\psi, \gamma)$ `// Alg. 2`
    $\mathbf{x}_t \leftarrow \mathbf{x}_{t+1}$
    $\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} + \ell_{t+1}\, \mathbf{1}\{t + 1 \in \mathcal{T}^\Lambda\}$
  $\widehat{\mathcal{C}} \leftarrow \widehat{\mathcal{C}} / \max\{|\mathcal{T}^\Lambda|, 1\}$ `// Arrival normalization`
  $\mathcal{L}(\psi) \leftarrow \widehat{\mathcal{C}}$
  Update ψ by descending $\nabla_\psi \mathcal{L}(\psi)$
until *convergence*

Algorithm 4: Preview-aware forecasting with Λ-chunk anchoring

Input: Pretrained ARDM kernel q; trained controller $\boldsymbol{u}_\psi$; initial state $\mathbf{x}_{t_0}$; forecast horizon H; preview horizon Λ; observation stream $\{\mathbf{y}_j\}_{j\in\mathcal{T}}$; control scale $\gamma > 0$

Output: Forecast trajectory $\mathbf{x}_{t_0+1:t_0+H} = (\mathbf{x}_{t_0+1}, \ldots, \mathbf{x}_{t_0+H})$

$N \leftarrow \lceil H/\Lambda \rceil$ `// Number of chunks`
for $c = 0, 1, \ldots, N - 1$ **do**
  $t_0^{(c)} \leftarrow t_0 + c\Lambda$
  $\Lambda_c \leftarrow \min\{\Lambda,\ H - c\Lambda\}$ `// Last chunk may be shorter`
  for $t = t_0^{(c)}, \ldots, t_0^{(c)} + \Lambda_c - 1$ **do**
    $\mathcal{T}^\Lambda_{t+1} \leftarrow \{\, j \in \mathcal{T} : t + 1 \le j \le t_0^{(c)} + \Lambda_c \,\}$ `// Anchored preview; App. C`
    Build $\bar{\mathbf{y}}_{t+1}$ from $\mathcal{T}^\Lambda_{t+1}$ `// App. C`
    $(\mathbf{x}_{t+1},\, \ell_{t+1}) \leftarrow \textsc{ControlledStep}(\mathbf{x}_t, \bar{\mathbf{y}}_{t+1}, q, \boldsymbol{u}_\psi, \gamma)$ `// Alg. 2`
  if $c + 1 < N$ **then**
    $\mathbf{x}_{t_0^{(c+1)}} \leftarrow \mathbf{x}_{t_0^{(c)} + \Lambda_c}$ `// Autoregressive handoff`
return $\mathbf{x}_{t_0+1:t_0+H}$

E KL regularization for guided ARDM training

We give a tractable surrogate for the windowed KL term $D_{\mathrm{KL}}\big(\mathcal{P}_\psi(\mathbf{x}_{t_0+1:t_0+\Lambda} \,|\, \mathbf{x}_{t_0}) \,\|\, \mathcal{Q}(\mathbf{x}_{t_0+1:t_0+\Lambda} \,|\, \mathbf{x}_{t_0})\big)$. Throughout, $\mathcal{P}_\psi$ is induced by the controlled kernel (Eq. 6–7) and $\mathcal{Q}$ by the unguided ARDM (Eq. 1–4).

Path KL and reduction to one-step KLs. Conditioned on $\mathbf{x}_{t_0}$, both $\mathcal{P}_\psi$ and $\mathcal{Q}$ define Markov rollouts over $t = t_0, \ldots, t_0 + \Lambda - 1$. The chain rule for KL therefore gives the exact decomposition

$$
\begin{aligned}
&D_{\mathrm{KL}}\Big(\mathcal{P}_\psi(\mathbf{x}_{t_0+1:t_0+\Lambda} \mid \mathbf{x}_{t_0}) \,\Big\|\, \mathcal{Q}(\mathbf{x}_{t_0+1:t_0+\Lambda} \mid \mathbf{x}_{t_0})\Big) \\
&= \sum_{t=t_0+1}^{t_0+\Lambda} \mathbb{E}_{\mathbf{x}_{t-1}\sim\mathcal{P}_\psi}\Big[D_{\mathrm{KL}}\big(p(\mathbf{x}_t \mid \mathbf{x}_{t-1}; \boldsymbol{U}_t) \,\|\, q(\mathbf{x}_t \mid \mathbf{x}_{t-1})\big)\Big].
\end{aligned}
\tag{10}
$$

The challenge is that $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}; \boldsymbol{U}_t)$ and $q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$ are defined as marginals of diffusion latent chains and the one-step KL is generally intractable.

Augmenting with diffusion latents. Fix a physical transition $t-1 \to t$ and introduce the joint latent chains $\mathbf{z}_t^{(0:S)}$ with $\mathbf{x}_t \equiv \mathbf{z}_t^{(0)}$. Define the corresponding augmented distributions

$$\bar{q}(\mathbf{z}_t^{(0:S)} \mid \mathbf{x}_{t-1}) \triangleq q(\mathbf{z}_t^{(S)}) \prod_{s=1}^{S} q(\mathbf{z}_t^{(s-1)} \mid \mathbf{z}_t^{(s)}; \mathbf{x}_{t-1}), \tag{11}$$

$$\bar{p}_\psi(\mathbf{z}_t^{(0:S)} \mid \mathbf{x}_{t-1}, \bar{\mathbf{y}}_t) \triangleq q(\mathbf{z}_t^{(S)}) \prod_{s=1}^{S} p(\mathbf{z}_t^{(s-1)} \mid \mathbf{z}_t^{(s)}; \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}). \tag{12}$$

Since $\mathbf{x}_t$ is a marginal of $\mathbf{z}_t^{(0:S)}$, data processing yields the bound

$$D_{\mathrm{KL}}\big(p(\mathbf{x}_t \mid \mathbf{x}_{t-1}; \boldsymbol{U}_t) \,\|\, q(\mathbf{x}_t \mid \mathbf{x}_{t-1})\big) \;\le\; D_{\mathrm{KL}}\big(\bar{p}_\psi(\mathbf{z}_t^{(0:S)} \mid \mathbf{x}_{t-1}, \bar{\mathbf{y}}_t) \,\|\, \bar{q}(\mathbf{z}_t^{(0:S)} \mid \mathbf{x}_{t-1})\big). \tag{13}$$

We use the latent-chain KL on the right as a computable surrogate for the intractable marginal KL on the left.

Sub-step KL and closed form. Because $\bar{p}_\psi$ and $\bar{q}$ share the same prior $q(\mathbf{z}_t^{(S)})$, the joint KL decomposes exactly along the reverse chain:

$$D_{\mathrm{KL}}\big(\bar{p}_\psi \,\|\, \bar{q}\big) = \sum_{s=1}^{S} \mathbb{E}_{\mathbf{z}_t^{(s)} \sim \bar{p}_\psi}\Big[D_{\mathrm{KL}}\big(p(\mathbf{z}_t^{(s-1)} \mid \mathbf{z}_t^{(s)}; \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}) \,\|\, q(\mathbf{z}_t^{(s-1)} \mid \mathbf{z}_t^{(s)}; \mathbf{x}_{t-1})\big)\Big]. \tag{14}$$

Under Eq. 7 and the corresponding unguided kernel, both sub-step conditionals are Gaussians with identical covariance $\sigma_s^2 \boldsymbol{I}_d$, hence each KL term admits the closed form

$$D_{\mathrm{KL}}(\cdot\|\cdot) = \frac{1}{2\sigma_s^2}\big\|\boldsymbol{\mu}(\mathbf{z}_t^{(s)} + \gamma \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}, s) - \boldsymbol{\mu}(\mathbf{z}_t^{(s)}, \mathbf{x}_{t-1}, s)\big\|_2^2. \tag{15}$$

Combining Eq. 13–15 yields a tractable KL regularizer that is fully aligned with our guided-ARDM parameterization.

DDIM specialization. For DDIM, the per-substep KL in Eq. 15 admits a standard two-part form: a *control-energy* term and a *denoiser/score-discrepancy* term induced by the shift $\mathbf{z}_t^{(s)} \mapsto \mathbf{z}_t^{(s)} + \gamma \mathbf{u}_t^{(s)}$, i.e., $\|\mathbf{u}_t^{(s)}\|_2^2$ and $\|\epsilon(\mathbf{z}_t^{(s)} + \gamma \mathbf{u}_t^{(s)}, \mathbf{x}_{t-1}, s) - \epsilon(\mathbf{z}_t^{(s)}, \mathbf{x}_{t-1}, s)\|_2^2$, where $\epsilon(\cdot)$ is the noise-prediction network underlying $\boldsymbol{\mu}$, with step-dependent weights determined by the noise schedule.

F Data and Evaluation

Datasets. The KS equation is a fourth-order nonlinear PDE modeling flame-front instabilities and solidification dynamics:$\partial_t u + u\,\partial_\xi u + \partial_\xi^2 u + \nu\,\partial_\xi^4 u = 0$, where $\nu > 0$ is the viscosity. We solve it on a periodic domain with 256 spatial points and time step $\Delta t = 0.2$. Training trajectories span $140\Delta t$, while validation and test trajectories extend to $640\Delta t$. Data generation and splits follow Shysheya et al. [44], Brandstetter et al. [53].

Kolmogorov flow is a two-dimensional incompressible Navier–Stokes system: $\partial_t \boldsymbol{v} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} - \nu \nabla^2 \boldsymbol{v} + \frac{1}{\rho}\nabla p - f = 0, \nabla \cdot \boldsymbol{v} = 0$, with velocity field $\boldsymbol{v} = (v_\xi, v_\eta)$, viscosity ν, density ρ, pressure p, and external forcing f. Trajectories contain 64 states for training and 180 states for validation and test, represented on a 64×64 grid with $\Delta t = 0.2$. Data generation and splits follow Shysheya et al. [44], Rozet and Louppe [19]. We evaluate on the scalar vorticity field $\Omega = \partial_\xi v_\eta - \partial_\eta v_\xi$, which captures rotational structures.

For a compact real-world DA benchmark, we use ERA5 reanalysis fields at 500 hPa, extracting temperature and horizontal winds (T, u, v) over a North America patch at 6-hour cadence (00/06/12/18 UTC). We use years 2006–2016, split by year, with the final year held out for validation/evaluation and the remaining years used for training. Each snapshot is interpolated to a 64×64 regular grid and partitioned into fixed-length 64-step trajectories. Per-channel min/max normalization statistics are computed on the training set only, and results are reported on all fields.

Evaluation Metrics. HCT is defined as the last index $m_{\max}$ for which the Pearson correlation $r(m)$ between forecast and ground truth remains above a fixed threshold ϕ: $m_{\max} = \max\{m : r(m) \geq \phi\}$, $t_{\max} = m_{\max}\,\Delta t$. We use $\phi = 0.9$, stricter than the 0.8 of Shysheya et al. [44], so our HCT values are not directly comparable to theirs.

To assess physical fidelity beyond pointwise error, we also report domain-specific diagnostics. For the 1D KS system, with state field $u(\xi)$ over spatial coordinate ξ, we compute the TV, $\mathrm{TV}(u) = \int |\partial_\xi u(\xi)|\, d\xi$, which measures spatial oscillation and the sharpness of evolving patterns. For 2D Kolmogorov flow, with velocity field $\boldsymbol{v} = (v_\xi, v_\eta)$ and scalar vorticity $\Omega = \partial_\xi v_\eta - \partial_\eta v_\xi$, we report the kinetic-energy dissipation rate $\varepsilon(\boldsymbol{v}) = \nu \int \|\nabla \boldsymbol{v}(\xi,\eta)\|_F^2\, d\xi\, d\eta$. This diagnostic measures the rate at which kinetic energy is dissipated at small scales. For both diagnostics, Tab. 1 reports the error e_D of Eq. 9, which excludes the initial state.

G Pretraining ARDM

Implementation details. We adapt the 1D and 2D diffusion codebase from `lucidrains`[1] into an ARDM tailored for PDE forecasting. Each ARDM transition corresponds to one-step forecasting via a DDIM sampler with S=3 denoising steps, v-parameterization, and a sigmoid noise schedule.

The backbone is a residual U-Net with multi-resolution attention and learned sinusoidal time embeddings:

```
dim = 64,
dim_mults = (1, 2, 4, 8),
learned_sinusoidal_dim = 128
```

Attention layers are applied at intermediate and coarse resolutions, while residual blocks follow the standard Conv–Norm–SiLU design.

Training configuration. Models are trained on a single NVIDIA RTX A6000 GPU with mixed precision (FP16) and exponential moving average. The configuration is:

```
train_batch_size = 32
train_lr = 3.2e-4
train_num_steps = 1000000
gradient_accumulate_every = 1
ema_decay = 0.995
ema_every = 10
```

H Control Network Architecture

Overview. The *controller* network $\boldsymbol{u}_\psi$ produces controls $\mathbf{u}_t^{(s)}$ used at each denoising sub-step. In the rollout, we write $\mathbf{u}_{t+1}^{(s)} = \boldsymbol{u}_\psi(\cdot)$ for brevity; here we provide a detailed description of the architecture and conditioning.

Inputs. At each sub-step, the network receives five spatial tensors concatenated along channels: (i) the current latent $\mathbf{z}_{t+1}^{(s)}$, (ii) the previous state $\mathbf{x}_t$, (iii) the preview observation $\mathbf{y}_{\tau_t}$, (iv) the auxiliary mask $\boldsymbol{M}_{\tau_t}$, and (v) the previous control $\mathbf{u}_{\mathrm{prev}}$. In addition, it conditions on scalar metadata: the preview lag Δ_{t,τ_t}, the local frame index k within the preview window, and the current $\log$ of the Signal-to-Noise Ratio (SNR) at diffusion sub-step s from the DDIM schedule.

Backbone encoder. The concatenated inputs are passed through a shallow convolutional encoder with two 3×3 layers and group normalization. A downsample/upsample path provides limited multi-scale context: features are reduced to half resolution, then upsampled and fused back with the original resolution. A 1×1 fusion convolution followed by group normalization yields the encoded representation.

[1] https://github.com/lucidrains/denoising-diffusion-pytorch

Feature-wise Linear Modulation (FiLM) conditioning. Each scalar input is normalized to $[0, 1]$ and embedded via a two-layer Multi-Layer Perceptron of hidden dimension `hid` (768 in all experiments). The three embeddings (lag, frame index, SNR) are concatenated and mapped to scale/shift coefficients (a, b) through a linear layer. These coefficients modulate the encoded features in a FiLM style, feat $\mapsto$ feat $\cdot\,(1 + a) + b$.

Residual head. A 3×3 convolutional head outputs the control increment δ_ψ. This is added to a normalized copy of the previous control $\mathbf{u}_{\text{prev}}$, producing $\mathbf{u}_t^{(s)} = \text{GroupNorm}(\mathbf{u}_{\text{prev}}) + \delta_\psi$. At the first denoising step, $\mathbf{u}_{\text{prev}}$ is set to zero.

Configuration. In our experiments we instantiate the control network as `hid=768`, with group normalization (8 groups), hidden dimension `hid` for the encoder, and FiLM embeddings of dimension `hid`. The architecture is sufficiently expressive to incorporate preview information into the denoising dynamics.

Training details. Gradients flow only into ψ (the pretrained UNet is frozen). We use gradient checkpointing at each UNet call and detach $\mathbf{u}_{\text{prev}}$ within a frame to avoid deep denoising-step recurrences; memory scales with the number of checkpoints. Experiments work with $\beta = 10^{-2}$ found via grid search while $\gamma = 0.1$.

I HCT Metric

Tab. 6 reports high correlation time (HCT) for the two PDE benchmarks.

Table 6: **CADA stays correlated with the ground truth longest in nearly all regimes.** High-correlation time (HCT↑: last step at which the Pearson correlation with ground truth is at least 0.9; App. F, capped at the rollout length) for Kolmogorov flow (short/long: 60/180 steps) and KS (140/640) under DS-k and MS-k observations (Sec. 4). Rows: baselines and ablations (TTO-DA: no amortization; BoN: best of 16 unguided samples). Best in bold, including ties.

	DS-2		**DS-4**		**DS-8**		**MS-2**		**MS-4**		**MS-8**	
	short	long	short	long	short	long	short	long	short	long	short	long
Kolmogorov flow												
CADA (ours)	**60**	**180**	**60**	**180**	**50**	50	**60**	**180**	**60**	**180**	**60**	**70**
Joint AAO	**60**	**180**	**60**	**180**	**50**	**180**	**60**	173	25	25	10	18
Joint AR	**60**	**180**	**60**	**180**	**50**	**180**	**60**	177	**60**	62	13	13
Plain amortized	**60**	64	41	45	13	17	38	42	32	33	31	32
Universal amortized	37	13	17	17	8	8	50	50	32	32	28	28
DiffDA	–	–	–	–	–	–	38	39	31	33	27	28
TTO-DA	**60**	73	**60**	**180**	37	40	**60**	60	48	40	27	25
BoN	40	40	32	26	21	26	40	26	32	26	32	27
Kuramoto–Sivashinsky												
CADA (ours)	**140**	**640**	**140**	**640**	**140**	**640**	**140**	**640**	**140**	**640**	**140**	**640**
Joint AAO	**140**	**640**	139	**640**	139	**640**	**140**	**640**	**140**	**640**	129	633
Joint AR	139	639	139	639	139	639	139	639	139	639	139	639
Plain amortized	**140**	211	**140**	147	76	56	**140**	253	**140**	264	**140**	242
Universal amortized	**140**	262	**140**	168	62	62	**140**	286	**140**	217	**140**	274
DiffDA	–	–	–	–	–	–	**140**	**640**	**140**	638	**140**	627
TTO-DA	**140**	218	**140**	**640**	139	633	**140**	638	**140**	637	**140**	634
BoN	**140**	260	**140**	250	**140**	240	**140**	243	**140**	250	**140**	246

J Additional Results

To contextualize our method relative to classical techniques, we additionally evaluate standard EnKF, 3D-Var, and 4D-Var implementations (Tab. 8). With downsampled observations at every step, classical methods remain accurate (RMSE at most 0.19, except EnKF at DS-8), and 3D-Var even beats CADA at DS-8 (0.09 vs. 0.138). Under masked, temporally strided observations, all

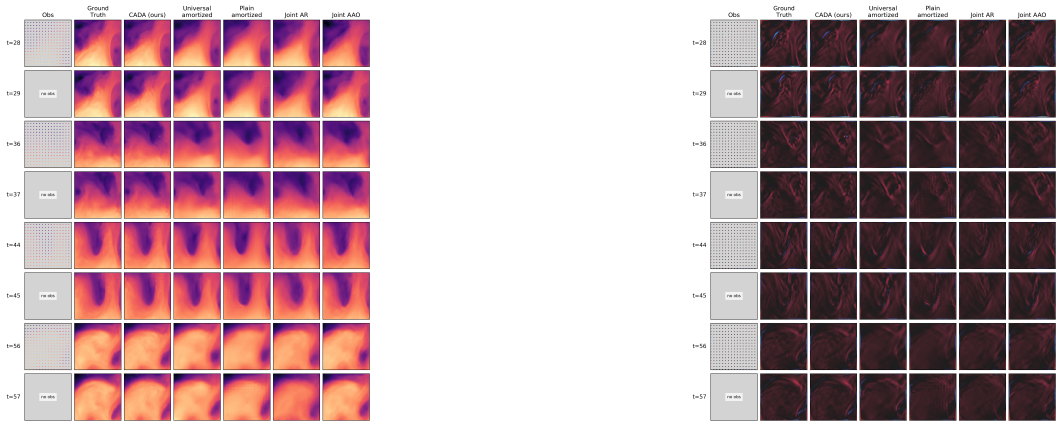


Figure 4: **CADA keeps sharper temperature fronts and more coherent vorticity filaments on compact ERA5, at observed and unobserved steps alike.** 500 hPa temperature (left) and vorticity (right) over North America under MS-4 observations; rows are forecast times, and columns show observations (gray: none; dots: sparse measurements), ground truth, and reconstructions by CADA, Universal amortized, Plain amortized, Joint AR, and Joint AAO. Full comparison for Fig. 1b; RMSE in Tab. 3.

Table 7: **CADA is the most accurate under irregular, randomly masked observations.** RMSE↓ on Kolmogorov flow when each grid point is observed independently with probability $p = 0.125$, at irregular times 2–6 steps apart; rows are methods. Best in bold.

Method	RMSE ↓
Plain amortized	0.28
Universal amortized	0.26
Joint AAO	0.18
Joint AR	0.07
DiffDA	0.23
TTO-DA	0.13
BoN	0.32
CADA (ours)	**0.02**

Table 8: **Classical DA methods do best with downsampled observations at every step and degrade sharply under masked, strided ones.** RMSE↓ on Kolmogorov flow (60-step rollouts); rows are observation regimes (Sec. 4), columns are EnKF, 3D-Var, 4D-Var, and CADA (short-rollout values from Tab. 2). Best in bold.

Regime	EnKF	3D-Var	4D-Var	CADA
DS-2	0.08	0.08	0.06	**0.016**
DS-4	0.09	0.11	0.18	**0.020**
DS-8	0.37	**0.09**	0.19	0.138
MS-2	0.31	0.38	0.26	**0.017**
MS-4	0.32	0.41	0.28	**0.024**
MS-8	0.35	0.42	0.33	**0.060**

three degrade sharply (0.26–0.42), whereas CADA stays at 0.017–0.060, 5–15× lower than the best classical method. This behavior is expected: linear–Gaussian assumptions and reliance on second-order statistics limit EnKF's ability to recover fine-scale nonlinear structures in chaotic PDE systems, whereas CADA combines an expressive diffusion prior with amortized control for non-Gaussian, intermittently observed dynamics. Fig. 5 shows a qualitative comparison.

Lastly, assessing robustness beyond structured downsampling and regular masking, we conduct an additional experiment using *irregular* spatial observations. Each grid point is independently revealed with probability $p = 0.125$, producing non-aligned, non-regular measurement patterns. Observations follow an irregular cadence, with a minimum separation of 2 time steps and a maximum of 6 time steps between consecutive observations. As shown in Tab. 7, CADA attains the lowest RMSE (0.02 vs. 0.07 for the next best, Joint AR), while conditional ARDM baselines experience substantial degradation. These results reinforce that preview-based amortized control remains stable even when observations deviate significantly from regular masks, and that no architectural changes are required to accommodate such irregular regimes.

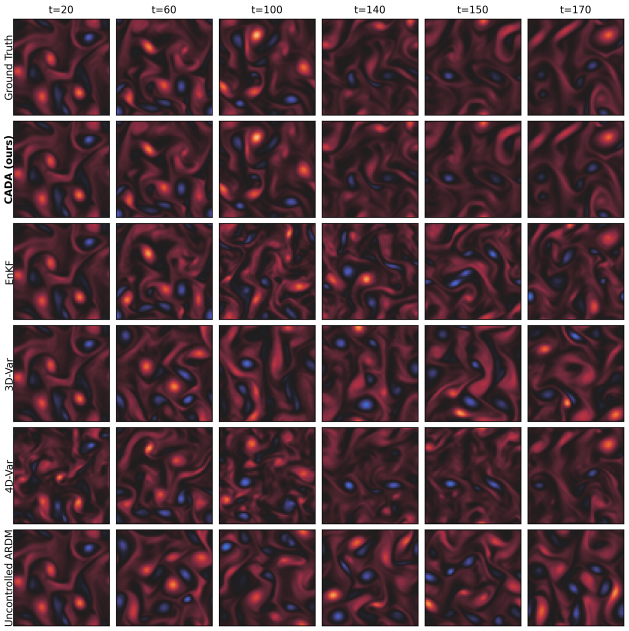


Figure 5: **CADA stays aligned with the ground truth over 170 steps, while classical DA methods and the uncontrolled ARDM drift.** Kolmogorov vorticity under MS-4 observations; columns are rollout times from t=20 to t=170, and rows show ground truth, CADA, EnKF, 3D-Var, 4D-Var, and the uncontrolled ARDM. RMSE at 60 steps in Tab. 8.

K Qualitative GenCast Results

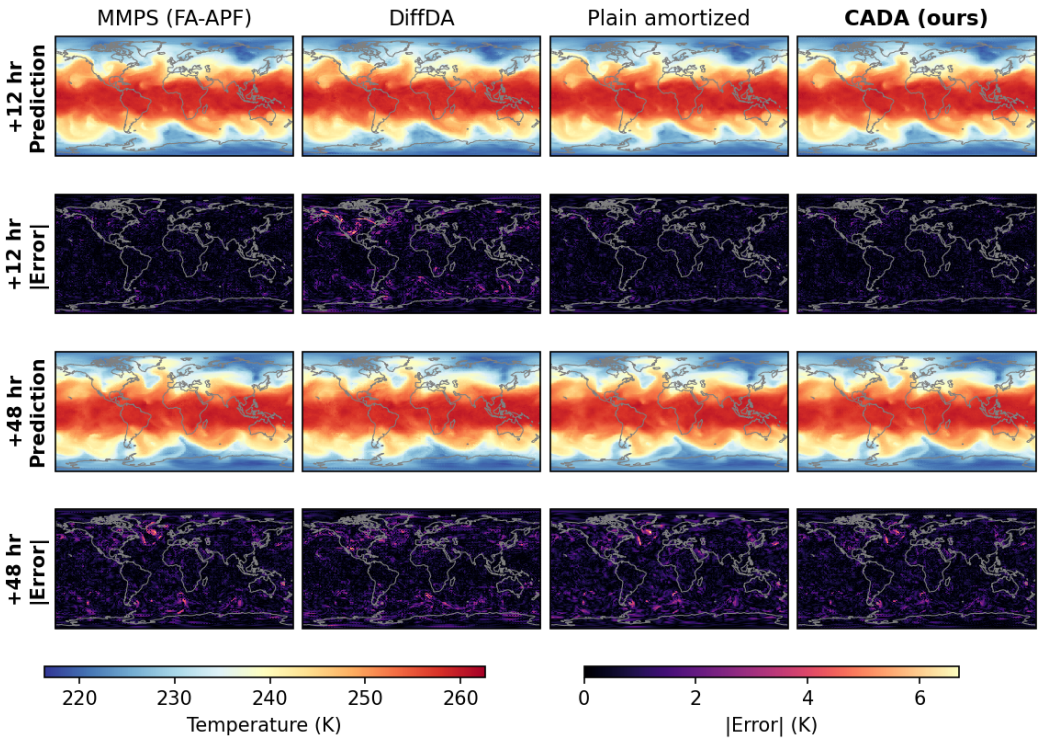


Figure 6: **CADA keeps GenCast temperature errors lower and more localized over 48 hours.** Global 400 hPa temperature under MS-4 observations; columns are methods (MMPS (FA-APF), DiffDA, Plain amortized, CADA), and rows show the forecast (K) and its absolute error (K) at 12 h and 48 h leads. Full maps for Fig. 3.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Sec. 4 outlines the experiments and Tables 1, 2, 7, 6, 8, 5, and 3 clearly demonstrate that the performance metrics for our model are better than the state-of-the-art baselines.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss some limitations of our approach in Section 5.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: App. B and App. E carry relevant assumptions and proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Architecture details have been discussed in App. G and App. H and experiment setup details in Sec. 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Algorithm details along with hyperparameters have been discussed in App. D, G and H. Code will be released with the camera ready version.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: All details can be found in App. G and H along with Sec. 4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Training the models takes a long time and reporting meaningful error bars is computationally expensive.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details have been provided in Tab. 3 along with App. G and App. H.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, our paper adheres to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The broader impacts of our work are discussed in Section 5.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Our proposed model poses no such risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All the reported baselines have been cited in Sec. 4.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: We do not release any new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects were involved.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Core method development did not involve LLM.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.