ABCD: *A*ny-Subset Autoregressive *B*ridges in *C*ontinuous Time and Space via Non-Markovian *D*iffusion

Gabe Guo * **Thanawat Sornwanee** **Lutong Hao** **Elon Litman** **Stefano Ermon**

Jose Blanchet

Abstract

Generating continuous-time, continuous-space stochastic processes (*e.g.*, videos, weather forecasts) conditioned on partial observations (*e.g.*, first and last frames) is a fundamental challenge. Existing approaches, (*e.g.*, diffusion models), suffer from key limitations: (1) noise-to-data evolution fails to capture structural similarity between states close in physical time and has unstable integration in low-step regimes; (2) random noise injected is insensitive to the physical process's time elapsed, resulting in incorrect dynamics; (3) they overlook conditioning on arbitrary subsets of states (*e.g.*, irregularly sampled timesteps, future observations). We propose **ABCD**: ***A**ny-Subset Autoregressive **B**ridges in **C**ontinuous Time and Space via Non-Markovian **D**iffusion*. Crucially, we model the process with *one continual SDE* whose time variable and intermediate states track the *real time and process states*. This has provable advantages: (1) the starting point for generating future states is the already-close previous state, rather than uninformative noise; (2) random noise injection scales with physical time elapsed, encouraging physically plausible dynamics with similar time-adjacent states. We derive SDE dynamics via changes-of-measure on path space, yielding another advantage: (3) path-dependent conditioning on arbitrary subsets of the state history and/or future. To learn these dynamics, we derive a path- and time-dependent extension of denoising score matching. Our experiments show **ABCD**'s superiority to competing methods on multiple domains, including video generation and weather forecasting.

1 Introduction

Goal Generating continuous-time, continuous-space stochastic processes is a fundamental problem. Examples include: videos, turbulent flows, stock prices, EEG readings, and weather forecasts. We are particularly interested in probabilistically infilling missing states, given only partial observations of the process, which may be irregularly sampled and even from the future (*e.g.*, sporadically synced GPS readings, videos with low frame rate, first minute of a stock price).

Existing Approaches Lack Inductive Bias However, popular generative modeling techniques often do not fully leverage the underlying structure of such processes. Instead, they shoehorn existing techniques (typically some variants of flow [1, 38] and diffusion [56] models) made for static data. For example, naive approaches concatenate the states at all times of interest, and have flow/diffusion models generate them simultaneously; this runs into memory problems on long sequences.

Lack of State Continuity: To solve this, autoregressive approaches generate the next few states conditioned on previous observations. Typically, each state is generated by a diffusion/flow model

*Corresponding author: gabeguo@stanford.edu. All authors affiliated with Stanford University.

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

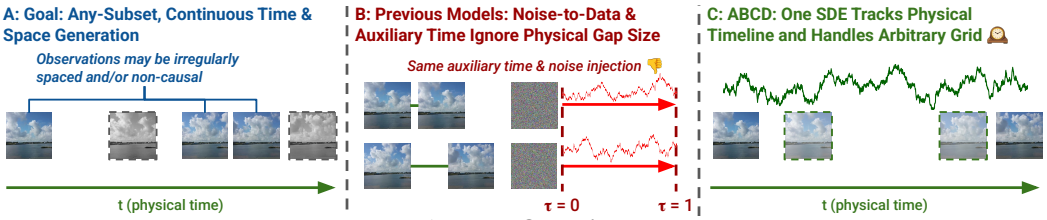


Figure 1: **Overview.**

that maps Gaussian noise to data [71, 72]. This ignores a crucial fact: adjacent states have minimal difference (*e.g.*, in a few seconds, videos barely evolve). So, starting from uninformative noise causes many issues, *e.g.*, poor structural preservation, stiff SDE integration [75]; also, the jump from noise to data is a physically implausible transition. Section 6 shows this gives incoherent generations.

Lack of Time-Scaled Volatility: As a partial fix, diffusion bridges and stochastic interpolants [75, 67, 40, 36, 39, 48, 49, 10] define generative SDEs whose endpoints are the states we transition between.[2] Yet, they still do not reflect the underlying process's true dynamics. Regardless of what physical times they interpolate, the SDE always starts at *auxiliary (non-physical) time* 0 and ends at auxiliary time 1, with the *same exact volatility (aka noise) schedule*. This is inappropriate for processes where change in state relates to physical time (*e.g.*, in movies, 1st to 2nd frame changes less than 1st to 20th frame), because these models inject the same volatility (*i.e.*, perturbation size) into both transitions. This causes problems, *e.g.*, flickering frames (Sec. 6), incorrect trajectories (Thm. 3).

Continuous-Time Any-Subset Generation is Underexplored Furthermore, previous works largely neglect handling arbitrary subsets of conditioning states. This limits applicability in cases where data was irregularly time-sampled (*e.g.*, stocks only record price when a transaction is made, GPS only transmits over strong 5G signal), or has non-causal conditioning (*e.g.*, first and last movie frames).

Our Approach We propose **ABCD**: ***A**ny-Subset Autoregressive **B**ridges in **C**ontinuous Time and Space via Non-Markovian **D**iffusion*. Our key insight is to model the underlying process with *one continual SDE* whose time variable and intermediate states correspond to *actual physical time and process states*. This makes volatility adaptive to times elapsed (Thm. 3 shows baseline methods cannot attain this) and encourages state change to scale with time change. To guarantee that the SDE matches the finite-dimensional distribution at times of interest, we set its drift via Doob h-transform (Thm. 1) [20, 17]. Via Gaussian process regression [68], we derive a simulation-free training objective for this drift that generalizes denoising score matching to the non-Markovian case with timestep-sensitive noising kernels (Thm. 2) [55, 56, 65]. We parameterize this drift with a cross-attention transformer [47]; it can condition on arbitrary (irregular) subsets of observed states and the future, similar to the re-popularized *any-subset autoregressive models* [23, 54]. Our work (1) unifies diffusion and any-subset autoregressive models; (2) generalizes any-subset autoregression to continuous time and space; (3) generalizes diffusion models to non-Markovian cases.

Results Our experiments show that (1) we address the long-overlooked problem of continuous-time any-subset generative modeling; (2) time-responsive volatility is important for modeling time series; (3) data-to-data bridges beat noise-to-data generation. Competing methods are essentially ablations on our method, up to minor derivation differences: conditional diffusion bridges (*e.g.*, DDBM) [75, 39] with fixed volatility schedules ablate (2); noise-to-data diffusion (*e.g.*, SMLD) [56] ablates (3). We validate ABCD's advantages over these baselines on video generation and weather forecasting. [3]

Our contributions are: ***(C1)*** We propose **ABCD**: an SDE generative model that generalizes autoregression to continuous time and space, and addresses the long-overlooked any-subset case. ***(C2)*** We prove that via its SDE time and state corresponding to physical process time and state, ABCD's unique inductive biases of data-to-data generative dynamics and time-adaptive volatility lead to faithful modeling. ***(C3)*** We derive a scalable training objective that extends denoising score matching to non-Markovian, time-dependent cases. ***(C4)*** We validate ABCD on practical tasks.

[2] State-to-state flow models/ODEs would be inappropriate, as they are deterministic, but the task is fundamentally stochastic.
[3] Sampled videos: https://abc-diffusion.github.io/.

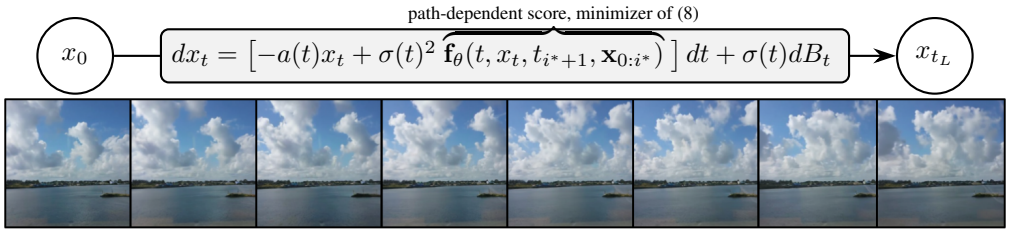


Figure 2: **ABCD Schematic:** The time series generation SDE from Thm. 1 is learned via Thm. 2. $\mathbf{f}_\theta(t, x_t, t_{i^*+1}, \mathbf{x}_{0:i^*})$ is the path-dependent score (parameterized by neural net), $\mathbf{x}_{0:i^*}$ is the conditioning frame history, t is the current *physical* time, t_{i^*+1} is the next frame's physical time.

2 Preliminaries

Any-Subset Autoregressive Models Consider a time series $x_{t_0}, \ldots, x_{t_{L-1}}$, where $t_0 < t_1 < t_2 < \ldots < t_{L-1}$. Vanilla autoregressive (AR) models predict $p(x_{t_{L-1}}, \ldots, x_{t_i} | x_{t_{i-1}}, \ldots, x_{t_0})$, *i.e.*, the future conditioned on the past. Any-subset autoregressive models predict $p(\mathbf{x}_{t_{\sigma(\geq i)}} | \mathbf{x}_{t_{\sigma(<i)}})$, where $\sigma : \{0, 1, \ldots, L-1\} \to \{0, 1, \ldots, L-1\}$ is a bijective mapping; that is, they predict the states in a permuted order that may not be causal in time [23].

Autoregressive models have four categories: (1) discrete space, discrete time; (2) continuous space, discrete time; (3) discrete space, continuous time; (4) continuous space, continuous time. Most existing models (*e.g.*, LLMs) are discrete-space, discrete-time (one token/timestep). There are also some continuous-space, discrete time models for video [28, 10, 71, 72] and images [37].

Very few popular generative models (discrete or continuous space) instantiate continuous-time autoregression, in which state-to-state transitions should mimic the physical time evolution of the real-world process. Approaches that could take in continuous time inputs, but construct auxiliary generative SDEs/ODEs to bridge between states (*e.g.*, noise-to-data diffusion/flow), do not count, as they only model a finite subset of process times [10].

Diffusion Bridge Models Diffusion bridge models map between two distributions $x_0 \sim p_{\text{data}}(x_0), x_1 \sim p_{\text{data}}(x_1|x_0)$, via an SDE whose drift is trained via denoising score matching [65]. There are multiple related formulations [75, 39, 1, 48, 49]; we show Zhou et al's [75] (switching time convention to generate in forward time), as it generalizes denoising diffusion models [56, 55]:

$$dx_t = \left(-a(t)x_t + \sigma(t)^2 \left[\mathbf{s}_\theta(t, x_t, x_0) - \nabla_{x_t} \log p(x_0|x_t)\right]\right) dt + \sigma(t) dB_t, x_0 \sim p_{\text{data}}(x_0) \quad (1)$$

$$\theta^* = \underset{\theta}{\text{argmin}} \, \mathbb{E}_{t \sim \mathcal{U}[0,1)} \underset{\substack{x_0, x_1 \sim p_{\text{data}} \\ x_t \sim p_{\text{base}}(x_t|x_0, x_1)}}{\mathbb{E}} \left[w(t) \|\mathbf{s}_\theta(t, x_t, x_0) - \nabla_{x_t} \log p_{\text{base}}(x_t|x_0, x_1)\|^2\right] \quad (2)$$

where $p_{\text{base}}(x_t|x_0, x_1)$ is a Gaussian transition kernel, *i.e.*, distribution of a diffusion bridge pinned at endpoints x_0, x_1 [75, 57]. To recover the familiar noise-to-data diffusion models, set $x_0 \sim \mathcal{N}(0, I)$ [56, 55]. Most diffusion models are Markovian, in that their dynamics are path-independent. A few works explore non-Markovian diffusions [43, 44], but rely on handcrafted memory kernels, *e.g.*, fBM.

Additional Related Works Comprehensive view of related works in Section M.

3 Generative SDE for Time Series

Intutition: To model time series, we have three major criteria: (1) The transition to the next state should interpolate from the previous state, rather than restarting from noise. These continual transitions are faithful to the physical nature of the process. (2) The trajectory should depend on the observed history (and future states that we have knowledge of). (3) The change from state-to-state should scale with the amount of physical time elapsed: smaller time differences mean smaller changes in the state.

We construct a solution based on conditional diffusion bridges [75]. Towards (1), diffusion bridges are specifically designed to transition from data-to-data, rather than noise-to-data. Towards (2), we can combine this with a continuous-time extension of any-subset autoregressive modeling [23], and feed the state history into the diffusion bridge network via cross-attention. Finally, towards (3), we

should give the conditional diffusion bridge a volatility coefficient (noise injection scale) that is proportional to the time elapsed, so that larger time gaps lead to more variation in state outcome.

3.1 Desired Time Series

Here, *time series* means continuous-space, continuous-time stochastic process in $\mathbb{R}^d$. Abusing notation, we use x_t and $X(t)$ interchangeably to denote process state at time t. WLOG, we want to generate the finite-dimensional state distribution $p_{data}(x_{t_1}, \ldots, x_{t_L}|x_0)$ of a time series at times $0 < t_1 < \ldots < t_L$, given initial condition x_0. We assume p_{data} is only observable via samples, *i.e.*, dataset. As an example, x_0 is a video's first frame, and $x_{t_1} \ldots x_{t_L}$ are the remaining frames (maybe irregularly spaced). Section 3.4 addresses non-causal conditioning and arbitrary time grids.

3.2 Data-Generating Dynamics via Change-of-Measure

Since we want our diffusion process to share the same physical time as the data generating process, we start by defining a data-independent base process:

$$dX_t = -a(t)X_t dt + \sigma(t)dB_t \tag{3}$$

where $a(t), \sigma(t)$ are appropriately chosen to represent the volatility structure of the data generating process, and B is a Brownian motion under the base measure $\mathbb{P}$. This is a Gaussian process conditioned on the initial point X_0. Since this process is data-independent, it can differ from the data generating process. However, we can use path-dependent change-of-measure (via Doob-h transform) [17, 20] to change the base measure $\mathbb{P}$ into $\mathbb{Q}$ *such that the finite-dimensional distribution matches with that of the data*. This allows the process X under the new measure $\mathbb{Q}$ to become a data-generating, non-Gaussian, non-Markov process whose diffusion time corresponds to physical process time. [4]

Given a process at time t, we know the current path history $(X_s)_{s\in[0,t]}$, and seek to find a drift. For Markov processes, we can write the drift as a function of (t, X_t). In the general non-Markov case, we may require the drift to condition on the entire path. We show that it is sufficient to match the finite-dimensional distribution of the data generating process by choosing the drift to be of the form $s\left(t, X_t; \mathbf{X}_{0:i^*}\right)$, where $\mathbf{X}_{0:i^*}$ is a past observation on the time grid $(0, t_1, \ldots, t_L)$:

$$\mathbf{X}_{0:i^*} := [(0, X_0), (t_1, X_1), \ldots, (t_{i^*}, X_{t_{i^*}})] \tag{4}$$

with $i^* := \max\{i : t_i < t\}$. The non-causal case in Section 3.4 will allow the argument of the drift u to include future times.

We denote p_{base} as a probability distribution of $(X_t)_{t\in[0,t_L]}$ under $\mathbb{P}$, so

$$p_{\text{base}}(x_t|x_{t_{i^*}}, x_{t_{i^*+1}}) \sim \mathcal{N}\left(\mu_{t|t_{i^*},t_{i^*+1}}(x_{t_{i^*}}, x_{t_{i^*+1}}), \Sigma_{t|t_{i^*},t_{i^*+1}}\right), \tag{5}$$

where the affine function $\mu_{t|t_{i^*},t_{i^*+1}}$ and the constant matrix $\Sigma_{t|t_{i^*},t_{i^*+1}}$ are primitives of (a, σ) obtained via Gaussian regression as shown in Lemma 1 in Appendix A.

Theorem 1. *Under regularities in Appendix A, by defining a process* $(X_t^{new})_{t\in[0,t_L]}$ *following*

$$dX_t^{new} = \left[-a(t)X_t^{new} + \sigma(t)^2 s\left(t, X_t^{new}; \mathbf{X}_{0:i^*}^{new}\right)\right] dt + \sigma(t)dB_t \tag{6}$$

with its initial condition $X_0^{new} \sim p_{data}(x_0)$ *and Brownian motion* B_t*, and where the score*

$$\begin{aligned} s\left(t, x_t; \mathbf{x}_{0:i^*}\right) = &-\nabla_{x_t} \log p_{base}(x_t|x_{t_{i^*}}) \\ &+ \nabla_{x_t} \log \int p_{data}(x_{t_{i^*+1}}|x_0, \ldots, x_{t_{i^*}}) p_{base}(x_t|x_{t_{i^*}}, x_{t_{i^*+1}}) dx_{t_{i^*+1}}, \end{aligned} \tag{7}$$

we have that $\left(X_0^{new}, X_{t_1}^{new}, \ldots, X_{t_L}^{new}\right)$ *will have the same joint distribution as* $p_{data}(x_0, x_{t_1}, \ldots, x_{t_L})$.

Formal proof in Appendix A. We use Doob's h-transform [17, 20] to find $\mathbb{Q}$ that makes the finite-dimension distribution of X under $\mathbb{Q}$ follow the data distribution.

This theorem suggests that we can transform any data-independent base process X (up to regularities) into a new process X^{new} whose finite-dimension distribution matches the data process. Intuitively, this comes from the added score term, which "pulls" the SDE towards regions of high data density at the prescribed times, conditioned on the relevant path history. In contrast to previous works, our score term $s\left(t, X_t^{\text{new}}; \mathbf{X}_{0:i^*}^{\text{new}}\right)$ *allows non-Markov structure, thereby admitting flexible correlations.*

[4] The importance of non-Markov continuous-time processes is also present even when the finite-dimensional (therefore discrete-time) observation is Markov. This happens because continuous-time Markov does not admit flexible correlation structure. An example and discussions are in Figure 5.

Sampling Algorithm See Algorithm 2. To generate time series data, we simulate the SDE according to Eqn. 6. That is, given $a(t), \sigma(t)$, and $s\left(t, x_t; \mathbf{x}_{0:i^*}\right)$, we discretize: $dt \leftarrow \Delta t$. Then, we generate Gaussian increments for $dB_t \sim \mathcal{N}(0, \Delta t)$, and plug all these variables into the equation to get dX_t. We iterate by adding dX_t to X_t and Δt to t. When t is sufficiently close to one of the t_i of interest, we output the corresponding X_t value, stopping after we cross t_L. *This can be interpreted as continuous-time autoregression, which is underexplored in prior work.*

A distinguishing feature of our sample generation is that the computational budget and volatility injected are determined by the physical time difference between consecutive states. This encourages the model to make fewer changes in the "easy" transitions (*i.e.*, similar states close in time), and spend more effort on "hard" transitions (*e.g.*, states far apart in time, with more variation in outcomes).

3.3 Path- and Time-Dependent Bridge Score Matching

Next, we will show how the score function s in Theorem 1 can be learned with neural networks.

Theorem 2. *Under regularities in App. A, by defining the path-dependent bridge score matching loss*

$$\mathcal{L}_{DSM}(\hat{s}) := \mathop{\mathbb{E}}_{t\sim\mathcal{U}(0,t_{L-1})} \mathop{\mathbb{E}}_{\substack{p_{data}(x_0,\ldots,x_{t_{i^*+1}}) \\ p_{base}(x_t|x_{t_{i^*+1}},x_{t_{i^*}})}} \left[\left\|\hat{s}\left(t, x_t; \mathbf{x}_{0:i^*}\right) - \nabla_{x_t} \log p_{base}(x_{t_{i^*+1}}|x_t)\right\|_2^2\right], \quad (8)$$

we have that the actual score function s in Theorem 1 minimizes $\mathcal{L}_{DSM}$.

The proof is in Appendix B.3 by a bias-variance decomposition and identifying the unique minimizer as the conditional mean of $\nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t)$.

This theorem suggests we can train a score function by considering the empirical distribution and class of functions specified by neural networks. Algorithm 1 shows the training loop. The flexibility of s allows time re-weighting, since the proof is done in a point-wise in time manner. We select weight $w(t, t_{i^*}, t_{i^*+1}) = \frac{C_t(t_{i^*+1}, t_{i^*+1})}{\Phi(t, t_{i^*+1})}$, where C is defined in Eq. 21 and Φ is defined in Eq. 20. This loss roughly generalizes score matching losses popularly used for diffusion models [56, 75], up to minor problem formulation differences (Appendix F). *Distinct from prior works, our loss is path-dependent, and the target scales with the physical process time.*

3.4 Any-Subset and Non-Causal Extension

This subsection underpins a continuous-time generalization of any-subset autoregression [23, 54].

Any-Subset We've focused on finite-dimensional distribution $p_{\text{data}}(x_0, x_{t_1}, \ldots, x_{t_L})$, but what if we want $p_{\text{data}}\left(x_0, x_{t'_1}, \ldots, x_{t'_L}\right)$ of the same process at *other* times $t'_1 < \ldots < t'_L$? Similarly, we can adjust the loss $\mathcal{L}_{\text{DSM}}(\hat{s})$ to also include different variations of time measurements while sharing the same neural network model $\hat{s} = f_\theta$. Note that in this multi-task setting, the next time on the grid is no longer uniquely determined by the past observation times, so we must input this desired future time t_{i^*+1} to the neural network (combined with path history, this indexes the tasks):

$$\mathbf{f}_{\theta^*}(t, x_t, t_{i^*+1}, \mathbf{x}_{0:i^*}) \quad (9)$$

This multi-task loss over arbitrary subsets of the path is a unique feature of our method.

Non-Causal There is sometimes "peek ahead" knowledge of future states the process will reach. We may want to generate X^{new} given that some observation

$$\mathcal{O} = [(t_j, x_{t_j})]_{j\in J} \quad (10)$$

for some $J \subseteq \{1, \ldots, L\}$ is specified beforehand. At the current physical time t, we denote the already-generated constraint $\mathcal{O}_{\leq t} := \{(s, x_s) \in \mathcal{O} : s \leq t\}$ and yet-to-be-generated constraint $\mathcal{O}_{>t} := \{(s, x_s) \in \mathcal{O} : s > t\}$. The constraints' values will not change with time t but will change their membership: from $\mathcal{O}_{>t}$ to $\mathcal{O}_{\leq t}$. Note that when the physical time t is sufficiently small, $\mathcal{O}_{>t} = \mathcal{O}$, meaning that all specified constraints are yet to be generated.

The already generated $\mathcal{O}_{\leq t}$ is already a part of the observed-and-generated $\mathbf{x}_{0:i^*}$, so it will not pose any problem to our framework. However, the yet-to-be-generated constraint $\mathcal{O}_{>t}$ is known but not included in the state $\mathbf{x}_{0:i^*}$ at time t. Therefore, we have to conditionally train our score

function s to incorporate this peek ahead knowledge: by changing the score from $s\left(t, X_t; \mathbf{X}_{0:i^*}\right)$ to $s\left(t, X_t; \mathbf{X}_{0:i^*}, \mathcal{O}\right)$, the score matching loss becomes

$$\mathcal{L}_{\text{DSM}}(\hat{s}) := \mathop{\mathbb{E}}_{t\sim\mathcal{U}(0,t_{L-1})} \mathop{\mathbb{E}}_{\substack{p_{data}(x_0,\ldots,x_{t_L}) \\ p_{base}(x_t|x_{t_{i^*+1}},x_{t_{i^*}}) \\ \mathcal{O}|x_0,\ldots,x_{t_L}}} \left[\left\|\hat{s}\left(t, x_t; \mathbf{x}_{0:i^*}, \mathcal{O}\right) - \nabla_{x_t} \log p_{base}(x_{t_{i^*+1}}|x_t)\right\|_2^2\right]. \tag{11}$$

Note that $\mathcal{O}$ is sampled by randomly selecting observation time indices $J \subseteq \{1, \ldots, L\}$ before setting $\mathcal{O}$ accordingly. In the full expressivity setting, this is equivalent to training a score function $s_{\mathcal{O}}$ specifically for each observation $\mathcal{O}$; this ensures the approach's validity. The implementation uses a shared network, which processes $\mathbf{x}_{0:t}$ and $\mathcal{O}$ identically, as an additional layer of multitask learning.

One caveat of this approach is that for any observation $(t_j, x_{t_j}) \in \mathcal{O}$, the corresponding process $(X_t)_{t\in[0,t_L]}$ must have X_{t_j} being exactly x_{t_j}. This makes the conditional distribution $p_{data}(x_{t_1}, \ldots, x_{t_L}|x_0, \mathcal{O})$ not have full-support, violating the regularity condition for Theorem 1. For tractability, observations at future times t_j can be modeled as Gaussian distributions centered on observed values x_{t_j} with vanishingly small variance, instead of Dirac distributions.

Furthermore, with the learned score (explicitly including future index t_{i^*+1} for multi-task learning) $\mathbf{f}_{\theta^*}(t, x_t, t_{i^*+1}, \mathbf{x}_{0:i^*}, \mathcal{O}) \approx s(t, x_t; \mathbf{x}_{0:i^*}, \mathcal{O})$, the sampling conditioned on observation $\mathcal{O}$ via

$$dX_t^{\text{new}} = \left[-a(t)X_t^{\text{new}} + \sigma(t)^2 \mathbf{f}_{\theta^*}\left(t, X_t^{\text{new}}, t_{i^*+1}, \mathbf{X}_{0:i^*}, \mathcal{O}\right)\right] dt + \sigma(t) dB_t \tag{12}$$

may fail to satisfy $X_{t_j}^{\text{new}} = x_{t_j}$ for some $(t_j, x_{t_j}) \in \mathcal{O}$. One engineering "hack" to circumvent this is to parameterize the score function

$$\mathbf{f}_{\theta^*}\left(t, X_t^{\text{new}}, t_{i^*+1}, \mathbf{X}_{0:i^*}, \mathcal{O}\right) = \mathbf{f}_{\theta^*}^{\text{res}}\left(t, X_t^{\text{new}}, t_{i^*+1}, \mathbf{X}_{0:i^*}, \mathcal{O}\right) + \nabla_{x_t} \log p_{base}(x_{t_{i_{\mathcal{O}_{>t}}}}|x_t) \tag{13}$$

where $t_{i_{\mathcal{O}_{>t}}} := \min\{s : (s, x_s) \in \mathcal{O}_{>t}\}$ is the next future constraint time to fulfill [5]. When the current physical time t approaches the next constraint time $t_{i_{\mathcal{O}_{>t}}}$, the drift $\nabla_{x_t} \log p_{base}(x_{t_{i_{\mathcal{O}_{>t}}}}|x_t)$ dominates, and the process behaves like a Brownian bridge (in practice, the denominator has a small stability epsilon) whose right endpoint is our desired constraint $(t_{i_{\mathcal{O}_{>t}}}, x_{t_{i_{\mathcal{O}_{>t}}}})$.

4 Special Cases and Ablations

AR Conditional Diffusion Bridges When autoregressively[6] modeling $p_{\text{data}}(x_{t_1,\ldots,x_{t_L}}|x_0)$:

$$p_{\text{data}}(x_{t_1,\ldots,x_{t_L}}|x_0) = \prod_{i=1}^{L} p_{\text{data}}(x_{t_i}|x_0, \ldots, x_{t_{i-1}}), \tag{14}$$

why do we need a continual SDE X^{new} as defined in Theorem 1?

Since these are distribution matching problems, instead of sampling a single process ranging from the physical time of 0 directly to the terminal t_L, we could instead *autoregressively sample* L chained conditional diffusion bridges (with their own unscaled generative time variables $\tau \in [0, 1]$) that correspond to the transitions from physical times t_{i-1} to t_i. That is, generating

$$\left(\tilde{X}_\tau^{(i)}\right)_{\tau\in[0,1]} \tag{15}$$

for each $i \in \{1, 2, \ldots, L\}$, such that $\tilde{X}_1^{(i)} \sim p_{\text{data}}(x_{t_i}|x_0, \ldots, x_{t_{i-1}})$, with the initial condition coming from the previous bridge: $\tilde{X}_0^{(i)} = \tilde{X}_1^{(i-1)}$ (edge case $\tilde{X}_0^{(1)} = x_0$). Then, define a process $(Y_t)_{t\in[0,t_L]}$ by rescaling and concatenating the processes $\tilde{X}^{(i)}$'s together such that

$$Y_t := \tilde{X}^{(i^*)}_{\frac{t-t_{i^*}}{t_{i^*+1}-t_{i^*}}} \tag{16}$$

This will map the process $\tilde{X}^{(i)}$ and its corresponding bridge time $\tau \in [0, 1]$ into part of the process Y over the physical time $t \in [t_{i-1}, t_i]$. The path of Y is continuous (due to the initial condition of each

[5] Note $t_{i_{\mathcal{O}_{>t}}}$, in general, can be different from t_{i^*+1}: t_{i^*+1} is the time for the next state we will generate in the finite-dimensional distribution, while $t_{i_{\mathcal{O}_{>t}}}$ is the time for the next future state whose value we know in advance.

[6] For any-subset regression, we can analogously parameterize networks as in Section 3.4.

bridge being the previous bridge's terminal state), and Y agrees with p_{data} on finite-dimensional distribution at time $0, t_1, \ldots, t_L$ (due to the autoregressive factorization).

Each of the L conditional diffusion bridge processes $\tilde{X}^{(i)}, \ldots, \tilde{X}^{(L)}$ can *share the same coefficient functions* $\tilde{a}(\tau), \tilde{\sigma}(\tau)$ and *one common conditional network* $\tilde{\mathbf{f}}_\theta$:

$$d\tilde{X}_\tau^{(i)} = \left[-\tilde{a}(\tau)\tilde{X}_\tau^{(i)} + \tilde{\sigma}(\tau)^2 \tilde{\mathbf{f}}_\theta\left(\tau, \tilde{X}_\tau^{(i)}; \mathbf{x}_{0:t_{i-1}}\right)\right] d\tau + \tilde{\sigma}(\tau) dB_\tau^{(i)} \tag{17}$$

This fractured SDE formulation is almost the *same as probabilistic forecasting with stochastic interpolants (PFI) [10] and autoregressively conditioned denoising diffusion bridge models (DDBM) [75]* (*e.g.*, volatility doesn't adapt to physical time), save for minor parameterization differences (Sec. F, G). But, even as this model can be correct on finite-dimensional distributions, it generally cannot capture the correct volatility structure or path measure, as seen in the next theorem.

Theorem 3. *For some (generic) time grid $0, t_1, \ldots, t_L$, and volatility schedule σ, for any base drift schedules $a, \tilde{a}$, volatility schedule for the fractured process $\tilde{\sigma}$, score functions $\boldsymbol{f}_\theta, \tilde{\boldsymbol{f}}_\theta$, the process Y defined in 16 generally is not equal in distribution to the process X^{new} as defined in Theorem 1. Moreover, the two processes can be distinguished almost surely.*

Proof in App. C. It uses the fact that the fractured processes $\tilde{X}^{(i)}$ share the same volatility schedule $\tilde{\sigma}$. Under generic $0, t_1, \ldots, t_L$ and generic σ, we get mismatch in quadratic variations of Y and X^{new}.

Theorem 3 tells us that the shared choice of $\tilde{\sigma}$ across all the bridge processes $\tilde{X}_i$ (as in Equation 17) generally leads to different dynamics in the generation process than achieved with our ABCD method's continual SDE (Equation 6). This is important when we care about the dynamics of the process in between waypoints. For instance, stock prices shouldn't have the same total volatility in the first five seconds of the day ($t_1 - t_0 = 5$ seconds) as in the next five hours ($t_2 - t_1 = 5$ hours).

That being said, if we were to scale the volatility coefficient of each AR conditional diffusion bridge according to $\sqrt{t_{i^*+1} - t_{i^*}}$ (and *crucially, train with these coefficients*), the resultant construction would be equivalent to ABCD. This shows that the novel design elements of ABCD, as opposed to diffusion bridge models, can be broken into two parts: (1) We construct a new continuous-time formulation of the long-underexplored any-subset autoregressive conditioning [23], and introduce it to diffusion bridge models. Diffusion bridge models typically just condition on one endpoint ($O(1)$ possibilities), while we open the possibility to condition on *any subset of the trajectory* ($O(2^N), N \to \infty$ possibilities, depending on the grid resolution). (2) We emphasize the importance of physical-time-aware bridge dynamics. As Thm. 3 suggests, without time-scaled volatility, diffusion bridges would traverse physically implausible paths.

Noise-to-Data Diffusion We define conditional diffusion bridges, as in Eqn. 17, but *with noise as initial condition*: $\tilde{X}_0^{(i)} \sim \mathcal{N}(0, I)$. This is closely related to denoising diffusion models (DDPM, SMLD) [56], which are cases of DDBM [75]. However, noise-to-data diffusion (1) has physically nonsensical dynamics, only giving meaningful samples at $t = 1$; (2) lacks the inductive bias of similarity between adjacent states; (3) has stiff numerical integration at low discretization steps.

5 Experimental Setup

RQ1 (Task) Conventionally, autoregressive modeling is in discrete time and space; can we generalize it to continuous time and space? We also wish to address signals with non-causal conditioning, further extending the framework to *any-subset* modeling [23, 54].

RQ2 (Method) We investigate the impact of adjusting the generative SDE's volatility to the physical time elapsed, as opposed to a fixed volatility schedule that ignores real-world time scaling.

RQ3 (Method) We assess how effective a generative SDE that maps from data to data is in modeling time series, as opposed to the popular SDEs that map from data to noise [56].

Dataset We have two *video generation* datasets: CelebV-HQ [76] and Sky Timelapse [70]; see Section J.1. We evaluate *weather modeling* on the SEVIR VIL dataset [64]; see Section K.2.

Training Details For each clip, we sample the number of conditioning frames $L \sim \mathcal{U}[1, 16]$. After selecting L, the first frame is always used as conditioning, and all the other frames have uniform probability of getting picked: this gives irregularly sampled times. See Section J.2 and Algorithm 1.

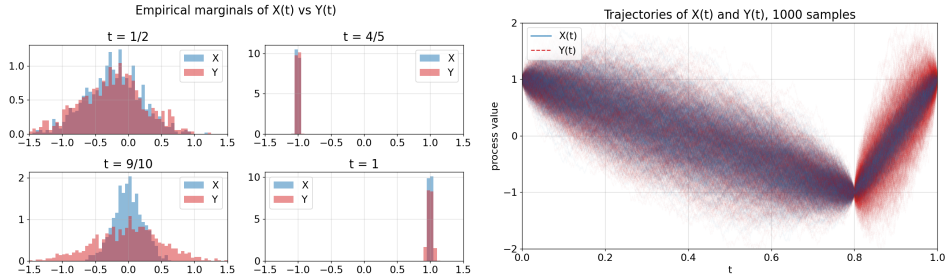


Figure 4: **Insufficiency of Autoregressive Conditional Diffusion Bridges:** See Section K.1.

Inference Settings Towards RQ1 (continuous-time any-subset modeling), we test video infilling at multiple subsampled time resolutions: 32 and 16 evenly spaced frames per SDE time unit (validating that continuous-time formulation aids in variable resolution modeling). Towards non-causal conditioning, we prompt every Kth frame, where $K \in \{4, 8, 16\}$. As the frames get filled in, the model is conditioned on its own generations (except that we teacher-force the conditioning frames), also adding a causal aspect. We conduct with $\{250, 500\}$ discretization steps of the SDE over $t \in [0, 1]$ (where $t = 1$ has the last frame). Section K.2 has weather experiment details. See Algorithm 2.

Metrics For video modeling, we use FVD [63, 19] and FID [25] metrics. FVD measures the temporal consistency of videos, and FID measures the per-frame quality of videos. See Section J.5. For SEVIR, we report pixelwise MAE, MSE, and RMSE, together with thresholded CSI (critical success index), POD (probability of detection), false alarm rate, and bias at the standard VIL thresholds $\{16, 74, 133, 160\}$ [64]. See Section K.2.

Model Architecture Our architecture extends DiT [47]. See Section I for more details.

Baselines The point of the evaluation is to answer RQ1, RQ2, and RQ3. We construct carefully controlled comparison against Section 4's baselines, which also correspond to competitive probabilistic modeling methods from recent literature. Comparison to autoregressively conditioned diffusion bridges answers RQ2; and is basically DDBM [75, 1, 48, 49, 39] and PFI [10] (see Sec. F, G). Comparison to noise-to-data diffusion answers RQ3; and is basically DDPM/SMLD [56, 55, 26]. All methods take the same conditioning and use the same architecture, so the computational complexity is the same; they also have the same training setup. See Section J.3.

6 Results

Our experiments show that our ABCD models outperform diffusion models at generating time series, due to our inductive biases of data-to-data transitions and time-adaptive computation.

Toy Experiment Fig. 4 illustrates Thm. 3 on Brownian motion pinned at $B(4/5) = -1, B(1) = B(0) = +1$. Chained diffusion bridges $Y(t)$, while capturing finite-dimensional marginals, have incorrect quadratic variation. Modeling with one continual SDE correctly captures the dynamics.

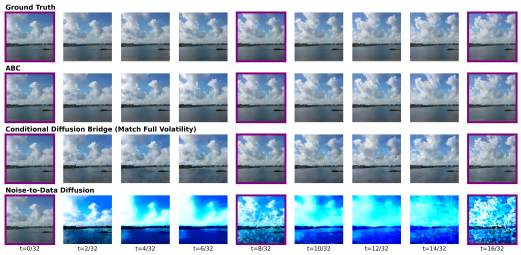


Figure 3: **Comparison:** ABCD beats conditional diffusion bridges (with equal σ) and noise-to-data diffusion models. Generated with 250 steps, and conditioning every eight (plus final) frames, although the model can overwrite prompts (to assess adherence). **Visit `https://abc-diffusion.github.io/` or supplementary material.**

Video Modeling ***View generated videos at*** `https://abc-diffusion.github.io/`. See Tables 1 and 2 for FVD scores. Judging by FVD on non-causal generation, ABCD is generally superior to the baselines across these video modeling datasets. The only baselines that are sometimes competitive are certain settings of the conditional diffusion bridge, depending on

the hyperparameters. The noise-to-data diffusion model is not competitive at all (as expected from its numerical stiffness and implausible dynamics), *validating our hypothesis from RQ3*.

However, FVD mainly assesses temporal consistency, but underemphasizes per-frame quality. FID scores in the appendix (Tables 16, 18), indicate that conditional diffusion bridge models (*e.g.*, Sky-Timelapse $\sigma = 0.3$) which beat ABCD on FVD are often not competitive on FID (per-frame quality); creating flickering videos. In the overall ranking across FVD and FID (Tables 17, 19), *ABCD clearly prevails, answering RQ2 in the affirmative: time-adaptive volatility is important*.

We also conduct experiments on strictly causal (forward in time) prompting (Section H.10, H.11); this is not our training's focus, but ABCD nonetheless achieves Pareto optimality. This *flexibility across causal and non-causal prompting further confirms RQ1*. Section H has additional results: ablation on full-path conditioning, sensitivity to hyperparameter choice, FID scores, causal evaluations, scaling, generalization to time grids unseen during training, constraint adherence, irregular time grid sampling, diversity analysis, runtime performance profiling.

Table 1: **Sky-timelapse Non-Causal FVD (↓):** On 1360 videos. BB stands for Brownian Bridge (Eq. 13). σ is the coefficient of $dB(t)$ in the SDE. Cos: (α, ϵ) and Exp: (B, K) are the parameters to the volatility schedules for noise-to-data diffusion (Sections L.1, L.4). We report mean and std results across three random seeds.

		Num Frames = 32						Num Frames = 16					
		pin_every = 4		pin_every =8		pin_every = 16		pin_every = 4		pin_every = 8		pin_every = 16	
Method	Steps	250	500	250	500	250	500	250	500	250	500	250	500
ABCD													
No BB: σ=0.4		**61.8** ± 0.2	***58.1*** ± 0.1	**162.5** ± 0.9	**137.9** ± 0.4	***346.5*** ± 0.2	***283.2*** ± 0.7	***123.2*** ± 0.5	115.1 ± 0.5	***234.1*** ± 0.2	**207.2** ± 0.3	313.4 ± 1.3	278.1 ± 1.1
W/ BB: σ=0.4		***63.3*** ± 0.1	**57.8** ± 0.2	***166.6*** ± 0.5	***140.9*** ± 0.4	377.7 ± 0.1	308.5 ± 3.0	131.9 ± 0.7	123.7 ± 1.0	235.7 ± 1.2	210.8 ± 0.5	***308.6*** ± 2.2	276.8 ± 1.3
Conditional Diffusion Bridge													
$\sigma = 0.071$		81.9 ± 0.2	72.7 ± 0.5	236.4 ± 0.7	208.2 ± 0.6	469.2 ± 1.1	437.4 ± 2.0	218.7 ± 0.2	193.3 ± 0.8	436.2 ± 1.9	386.4 ± 0.8	541.0 ± 1.3	482.3 ± 1.9
$\sigma = 0.1$		72.3 ± 0.2	65.9 ± 0.7	222.2 ± 0.5	199.9 ± 0.1	453.5 ± 1.5	434.2 ± 0.3	164.9 ± 0.3	147.0 ± 0.4	373.7 ± 1.5	326.2 ± 1.2	450.9 ± 0.8	404.9 ± 2.0
$\sigma = 0.3$		102.2 ± 0.2	72.7 ± 0.3	193.5 ± 0.3	159.0 ± 0.3	**303.4** ± 1.3	**277.7** ± 0.5	**121.4** ± 0.8	**111.8** ± 1.0	**217.6** ± 1.2	***208.9*** ± 1.4	**267.1** ± 1.6	**249.0** ± 0.6
$\sigma = 0.4$		142.8 ± 0.6	93.9 ± 0.7	252.2 ± 1.4	190.5 ± 0.6	375.3 ± 2.4	308.5 ± 4.2	136.4 ± 2.0	***114.1*** ± 0.5	250.9 ± 1.4	226.8 ± 2.4	316.0 ± 3.7	***267.2*** ± 1.5
$\sigma = 0.6$		223.3 ± 0.7	149.0 ± 2.1	409.6 ± 2.7	298.5 ± 1.3	592.1 ± 3.5	469.8 ± 2.3	206.6 ± 0.6	155.0 ± 1.5	356.5 ± 2.2	285.5 ± 1.3	506.8 ± 2.0	380.2 ± 2.5
Noise-to-Data Diffusion													
Cos: (3.0, 0.04)		509.2 ± 0.9	216.0 ± 1.0	822.0 ± 2.4	449.5 ± 1.3	761.4 ± 2.5	559.3 ± 3.3	315.5 ± 1.7	160.3 ± 0.3	508.1 ± 0.9	270.2 ± 0.7	542.1 ± 1.9	297.6 ± 1.0
Cos: (5.0, 0.05)		688.3 ± 0.5	345.5 ± 1.0	917.8 ± 1.3	676.0 ± 2.8	842.0 ± 3.1	783.0 ± 8.3	482.8 ± 1.9	208.7 ± 2.0	775.4 ± 0.7	364.4 ± 1.1	835.4 ± 3.4	423.0 ± 1.3
Exp: (4.0, 2.5)		574.8 ± 0.5	211.8 ± 0.9	716.8 ±0.8	442.7 ± 1.7	716.8 ± 0.7	637.9 ± 0.2	312.7 ±0.2	153.9 ± 0.4	534.8 ± 1.9	290.1 ± 1.6	611.2 ±2.4	361.9 ± 0.9
Exp: (5.0, 5.0)		799.4 ± 0.9	582.5 ± 1.9	783.9 ±1.5	707.8 ± 0.8	776.5 ± 0.5	678.2 ± 1.8	760.8 ±1.2	312.5 ± 3.9	772.0 ± 3.5	539.2 ± 3.0	696.2 ±0.9	452.3 ± 3.5

Table 2: **CelebV-HQ Non-Causal FVD (↓):** On 2048 videos. Mean and std over three random seeds.

		Num Frames = 32						Num Frames = 16					
		pin_every = 4		pin_every = 8		pin_every = 16		pin_every = 4		pin_every = 8		pin_every = 16	
Method	Steps	250	500	250	500	250	500	250	500	250	500	250	500
ABCD													
W/ BB: σ=0.5		***43.7*** ± 0.2	***35.3*** ± 0.1	**107.5** ± 0.5	**88.5** ± 0.2	285.8 ± 1.9	248.3 ± 0.4	***106.6*** ± 0.5	***96.4*** ± 0.3	201.6 ±1.6	***169.3*** ± 0.4	333.9 ± 0.3	278.4 ± 0.9
No BB: σ=0.5		**40.4** ± 0.2	**32.6** ± 0.1	112.2 ± 0.5	***93.8*** ± 0.4	311.7 ± 1.2	264.4 ± 1.1	**102.9** ± 0.7	**92.1** ± 0.2	**183.0** ± 0.6	**158.5** ±0.6	***301.6*** ± 1.4	**264.4** ± 1.0
Conditional Diffusion Bridge													
$\sigma = 0.5$		262.3 ± 1.0	181.7 ± 0.2	389.7 ± 0.9	265.1 ± 0.8	840.8 ± 3.0	421.0 ± 1.7	230.2 ± 0.7	168.9 ± 0.6	311.7 ± 0.6	246.7 ± 1.1	433.8 ± 0.7	319.4 ± 1.0
$\sigma = 0.125$		56.8 ± 0.2	41.6 ± 0.2	119.8 ± 0.3	97.3 ± 0.2	**233.8** ± 0.8	***217.7*** ± 0.3	113.1 ± 0.3	102.1 ± 0.3	***186.1*** ± 1.0	171.7 ± 1.0	**289.6** ± 0.6	299.4 ± 1.3
$\sigma = 0.09$		49.0 ± 0.0	38.8 ± 0.2	***111.5*** ± 0.5	93.9 ± 0.3	***237.1*** ± 1.0	**203.5** ± 1.3	120.4 ± 0.1	109.0 ± 0.9	205.1 ± 0.4	192.3 ± 0.8	339.6 ± 0.3	368.6 ± 0.7
Noise-to-Data Diffusion													
Cos: (3.0, 0.04)		706.0 ± 0.4	434.4 ± 0.3	908.6 ± 0.0	590.6 ± 2.2	970.5 ± 0.5	760.5 ± 3.9	565.2 ± 2.1	213.7 ± 1.6	672.8 ± 1.2	346.3 ± 1.7	825.8 ± 1.0	492.1 ± 3.3
Cos: (5.0, 0.05)		906.0 ± 0.8	572.5 ± 1.5	996.1 ± 0.8	756.0 ± 2.9	992.0 ± 2.3	907.3 ± 0.6	741.9 ± 1.7	295.8 ± 0.5	857.7 ± 1.5	484.0 ± 1.7	1023.2 ± 1.0	692.2 ± 2.8
Exp: (4.0, 2.5)		810.8 ± 1.4	337.5 ± 0.7	1019.7 ± 0.9	497.2 ± 0.8	1021.9 ± 0.6	670.3 ± 0.8	424.8 ± 0.7	138.9 ± 0.8	588.6 ± 0.9	224.0 ± 1.3	649.8 ± 1.7	***277.4*** ± 0.6
Exp: (5.0, 5.0)		1123.8 ± 0.6	748.6 ± 0.4	1206.2 ± 0.9	956.4 ± 1.5	1071.6 ± 1.3	956.7 ± 1.8	804.2 ± 2.4	420.8 ± 1.5	878.9 ± 0.6	591.1 ± 1.8	967.4 ± 0.7	763.5 ± 1.3

Weather Modeling Table 3 shows that ABCD also performs strongly on SEVIR. In the non-causal setting, ABCD Non-Causal achieves the best overall performance, outperforming both conditional diffusion bridges and noise-to-data diffusion; this again supports RQ2 and RQ3. ABCD also performs well in the strictly causal setting, showing that the same framework can handle both sparse non-causal conditioning and standard forecasting. See Sections K.2 and H.12 for more details.

Table 3: **SEVIR VIL**: Full Test Set (847 clips) Native Metrics. Three random inference seeds.

Method	MAE (↓)	RMSE (↓)	CSI_{16} (↑)	CSI_{74} (↑)	CSI_{133} (↑)	CSI_{160} (↑)	Avg. Rank (↓)
ABCD Non-causal	$\mathbf{18.228 \pm 0.003}$	$\mathbf{34.849 \pm 0.004}$	$\mathbf{0.562 \pm 0.00007}$	$\mathbf{0.699 \pm 0.00003}$	$\mathbf{0.644 \pm 0.00007}$	$\mathbf{0.664 \pm 0.00004}$	**1.0**
ABCD Causal	24.382 ± 0.010	54.793 ± 0.022	0.554 ± 0.00040	0.570 ± 0.00023	0.476 ± 0.00027	0.494 ± 0.00016	*2.0*
Conditional Diffusion Bridge	55.196 ± 0.098	88.038 ± 0.148	0.280 ± 0.00067	0.206 ± 0.00092	0.144 ± 0.00069	0.129 ± 0.00061	4.0
Noise-to-Data Diffusion	35.698 ± 0.121	62.457 ± 0.240	0.373 ± 0.00010	0.437 ± 0.00074	0.338 ± 0.00067	0.328 ± 0.00059	3.0

7 Discussion, Limitations, Conclusion

Future work should investigate sampling schemes, *e.g.*, hybrid ODE-SDE; coarse-to-fine by causally generating every K frames at low sampling rate before infilling the $K - 1$ frames in between. For

Table 4: **SEVIR VIL**: Full Test Set (847 clips) Detection/Calibration Metrics. Three random inference seeds.

Method	POD_{16} (↑)	POD_{74} (↑)	POD_{133} (↑)	POD_{160} (↑)	FAR_{16} (↓)	FAR_{74} (↓)	FAR_{133} (↓)	FAR_{160} (↓)	$Bias_{16}$	$Bias_{74}$	$Bias_{133}$	$Bias_{160}$
ABCD												
Non-causal	**0.926** ± **0.00007**	**0.843** ± **0.00002**	**0.774** ± **0.00001**	**0.762** ± **0.00001**	**0.412** ± **0.00005**	**0.197** ± **0.00003**	**0.206** ± **0.00011**	**0.162** ± **0.00008**	1.576 ± 0.00005	1.049 ± 0.00004	0.975 ± 0.00014	0.909 ± 0.00011
Causal	*0.814* ± *0.00039*	*0.712* ± *0.00020*	*0.621* ± *0.00030*	*0.638* ± *0.00016*	*0.366* ± *0.00033*	*0.258* ± *0.00022*	*0.329* ± *0.00019*	*0.314* ± *0.00018*	*1.285* ± *0.00053*	*0.959* ± *0.00027*	*0.924* ± *0.00019*	*0.930* ± *0.00027*
Conditional Bridge												
Diffusion Bridge	0.630 ± 0.00122	0.374 ± 0.00145	0.272 ± 0.00089	0.232 ± 0.00067	0.665 ± 0.00062	0.686 ± 0.00118	0.765 ± 0.00127	0.775 ± 0.00137	1.880 ± 0.00070	1.190 ± 0.00206	1.158 ± 0.00404	1.034 ± 0.00487
Noise-to-Data Diffusion												
Noise-to-Data	0.821 ± 0.00016	0.596 ± 0.00092	0.451 ± 0.00205	0.421 ± 0.00264	0.594 ± 0.00015	0.378 ± 0.00173	0.426 ± 0.00413	0.402 ± 0.00637	2.023 ± 0.00108	0.959 ± 0.00364	0.785 ± 0.00898	0.704 ± 0.01183

a mathematically faithful implementation, we made the transformer network attend to the entire history, but efficient architectures like SSMs [22] selectively compress history. Future work could also consider text-to-image generation [53]. Finally, future work should investigate generating chunks of states at a time, *e.g.*, augment SDE state to four frames; this fits into our framework.

In this study, we do not explicitly fit the volatility structure to the data-generating process, as quadratic variation is somewhat difficult to estimate given discrete time grids. Furthermore, our change-of-measure derivations to get the score matching loss rely on a predetermined volatility schedule: this makes volatility hard to learn jointly with score. As such, our time-adaptive volatility formulation merely enforces the inductive bias that accumulated quadratic variation scales as $\sigma^2(t_{i^*+1} - t_{i^*})$ with the physical time gap. Our main experiments on ABCD used the constant-magnitude volatility kernel. This is suitable because our data was sampled by clipping videos where the clip start time was roughly uniformly distributed. But, we do see evidence that the choice of kernel from App. L can sometimes affect performance. When doing the noise-to-data diffusion (which is an ablation on ABCD), there was some performance difference between different settings of the cosine decaying and exponentially decaying volatility settings. For now, we leave the volatility kernel to be a tunable hyperparameter as it is in diffusion model literature, but we hope to learn it from data in future work.

In summary, we introduce **ABCD**: an SDE-based generalization of any-subset autoregressive generative models to continuous time and space. We rigorously derived a training objective generalizing denoising score matching to path- and time-dependent cases. Finally, we theoretically and empirically showed that its inductive biases of data-to-data generative trajectories and time-adaptive volatility lead to faithful modeling of stochastic processes, such as videos and weather.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Department of Energy Computational Science Graduate Fellowship under Award Number(s) DE-SC0025528 to G. Guo. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 using NERSC award ASCR-ERCAP0038735 to G. Guo. Research supported by the NVIDIA Academic Grant Program using A100 GPU-Hours on Brev, awarded to S. Ermon. Research supported by the Stanford Institute for Human-Centered Artificial Intelligence (HAI)'s Google Cloud Credits Program, awarded to S. Ermon. J. Blanchet gratefully acknowledges support from DoD through the grants Air Force Office of Scientific Research under award number FA9550-20-1-0397 and ONR N00014-24-1-2655, also support from NSF via grants 2229012, 2312204, 2403007 is gratefully acknowledged.

We thank Jerry W Liu, Peter Pao-Huang, John Yao, Tristan Saidi for helpful feedback.

References

[1] Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. "Stochastic interpolants: A unifying framework for flows and diffusions". In: *Journal of Machine Learning Research* 26.209 (2025), pp. 1–80.

[2] Alexandre Alouadi, Baptiste Barreau, Laurent Carlier, and Huyên Pham. "Robust time series generation via Schrödinger Bridge: a comprehensive evaluation". In: *Proceedings of the 6th ACM International Conference on AI in Finance*. 2025, pp. 906–914.

[3] Brian DO Anderson. "Reverse-time diffusion equation models". In: *Stochastic Processes and their Applications* 12.3 (1982), pp. 313–326.

[4] Grigory Bartosh, Dmitry Vetrov, and Christian A Naesseth. "SDE Matching: Scalable and Simulation-Free Training of Latent Stochastic Differential Equations". In: *International Conference on Machine Learning*. PMLR. 2025, pp. 3054–3070.

[5] Christopher M Bishop. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[6] Theodore Allen Burton. *Volterra integral and differential equations*. Vol. 202. Elsevier, 2005.

[7] Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, Maneesh Agrawala, Lu Jiang, and Gordon Wetzstein. "Mixture of Contexts for Long Video Generation". In: *The Fourteenth International Conference on Learning Representations*. 2026. URL: https://openreview.net/forum?id=y6XJZlEC2x.

[8] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. "Diffusion forcing: Next-token prediction meets full-sequence diffusion". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 24081–24125.

[9] Tianrong Chen, Guan-Horng Liu, Molei Tao, and Evangelos Theodorou. "Deep momentum multi-marginal Schrödinger bridge". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 57058–57086.

[10] Yifan Chen, Mark Goldstein, Mengjian Hua, Michael Samuel Albergo, Nicholas Matthew Boffi, and Eric Vanden-Eijnden. "Probabilistic Forecasting with Stochastic Interpolants and Föllmer Processes". In: *International Conference on Machine Learning*. PMLR. 2024, pp. 6728–6756.

[11] Yu Chen, Wei Deng, Shikai Fang, Fengpei Li, Nicole Tianjiao Yang, Yikai Zhang, Kashif Rasul, Shandian Zhe, Anderson Schneider, and Yuriy Nevmyvaka. "Provably convergent Schrödinger bridge with applications to probabilistic time series imputation". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 4485–4513.

[12] Rembert Daems, Manfred Opper, Guillaume Crevecoeur, and Tolga Birdal. "Variational Inference for SDEs Driven by Fractional Noise". In: *The Twelfth International Conference on Learning Representations*. 2024.

[13] Tri Dao. "FlashAttention-2: Faster attention with better parallelism and work partitioning". In: *International Conference on Learning Representations*. Vol. 2024. 2024, pp. 35549–35562.

[14] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. "FlashAttention: Fast and memory-efficient exact attention with IO-awareness". In: *Advances in neural information processing systems* 35 (2022), pp. 16344–16359.

[15] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: *International Conference on Learning Representations*. 2017.

[16] Catherine Doléans-Dade. "Quelques applications de la formule de changement de variables pour les semimartingales". In: *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* 16.3 (1970), pp. 181–194.

[17] Joseph L Doob. *Classical potential theory and its probabilistic counterpart*. Vol. 262. Springer, 1984.

[18] Jhanvi Garg, Xianyang Zhang, and Quan Zhou. "Soft-constrained Schrödinger Bridge: a Stochastic Control Approach". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2024, pp. 4429–4437.

[19] Songwei Ge, Aniruddha Mahapatra, Gaurav Parmar, Jun-Yan Zhu, and Jia-Bin Huang. "On the content bias in Fréchet video distance". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 7277–7288.

[20] Igor Vladimirovich Girsanov. "On transforming a certain class of stochastic processes by absolutely continuous substitution of measures". In: *Theory of Probability & Its Applications* 5.3 (1960), pp. 285–301.

[21] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 249–256.

[22] Albert Gu and Tri Dao. "Mamba: Linear-time sequence modeling with selective state spaces". In: *First conference on language modeling*. 2024.

[23] Gabe Guo and Stefano Ermon. "Self-Speculative Decoding Accelerates Lossless Inference in Any-Order and Any-Subset Autoregressive Models". In: *The Fourteenth International Conference on Learning Representations*. 2026. URL: https://openreview.net/forum?id=hZnibTOke7.

[24] Mohamed Hamdouche, Pierre Henry-Labordère, and Huyên Pham. "Nonparametric generative modeling for time series via Schrödinger bridge". In: *Journal of Machine Learning Research* 27.58 (2026), pp. 1–23.

[25] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. "GANs trained by a two time-scale update rule converge to a local nash equilibrium". In: *Advances in neural information processing systems* 30 (2017).

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Advances in neural information processing systems* 33 (2020), pp. 6840–6851.

[27] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. "Cascaded diffusion models for high fidelity image generation". In: *Journal of Machine Learning Research* 23.47 (2022), pp. 1–33.

[28] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. "Video diffusion models". In: *Advances in neural information processing systems* 35 (2022), pp. 8633–8646.

[29] Wanli Hong, Yuliang Shi, and Jonathan Niles-Weed. "Trajectory Inference with Smooth Schrödinger Bridges". In: *International Conference on Machine Learning*. PMLR. 2025, pp. 23722–23750.

[30] Mohammad Mohaiminul Islam, Coen de Vente, Sharvaree Vadgama, Thijs P Kuipers, Afsana Khan, Clara I Sánchez, and Erik J Bekkers. "Longitudinal Flow Matching for Trajectory Modeling". In: *The 29th International Conference on Artificial Intelligence and Statistics*. 2025.

[31] Kiyosi Ito. *On stochastic differential equations*. Vol. 4. American Mathematical Society New York, 1951.

[32] Kiyosi Itô. "109. stochastic integral". In: *Proceedings of the Imperial Academy* 20.8 (1944), pp. 519–524.

[33] Ioannis Karatzas and Steven Shreve. *Brownian motion and stochastic calculus*. springer, 2014.

[34] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *The Third International Conference on Learning Representations*. 2015.

[35] Oskar Kviman, Kirill Tamogashev, Nicola Branchini, Víctor Elvira, Jens Lagergren, and Nikolay Malkin. "Multi-Marginal Flow Matching with Adversarially Learnt Interpolants". In: *The Fourteenth International Conference on Learning Representations* (2026).

[36] Bo Li, Kaitao Xue, Bin Liu, and Yu-Kun Lai. "BBDM: Image-to-image translation with Brownian bridge diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern Recognition*. 2023, pp. 1952–1961.

[37] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. "Autoregressive image generation without vector quantization". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 56424–56445.

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

[39] Xingchao Liu, Lemeng Wu, Mao Ye, and Qiang Liu. "Let us Build Bridges: Understanding and Extending Diffusion Generative Models". In: *NeurIPS 2022 Workshop on Score-Based Methods*. 2022.

[40] Zonglin Lyu, Ming Li, Jianbo Jiao, and Chen Chen. "Frame interpolation with consecutive brownian bridge diffusion". In: *Proceedings of the 32nd ACM International Conference on Multimedia*. 2024, pp. 3449–3458.

[41] Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. "Latte: Latent Diffusion Transformer for Video Generation". In: *Transactions on Machine Learning Research* (2025). ISSN: 2835-8856. URL: https://openreview.net/forum?id=ntGPYNUF3t.

[42] Otton Nikodym. "Sur une généralisation des intégrales de MJ Radon". In: *Fundamenta Mathematicae* 15.1 (1930), pp. 131–179.

[43] Gabriel Nobis, Maximilian Springenberg, Marco Aversa, Michael Detzel, Rembert Daems, Roderick Murray-Smith, Shinichi Nakajima, Sebastian Lapuschkin, Stefano Ermon, Tolga Birdal, Manfred Opper, Christoph Knochenhauer, Luis Oala, and Wojciech Samek. "Generative Fractional Diffusion Models". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=B9qg3wo75g.

[44] Gabriel Nobis, Maximilian Springenberg, Arina Belova, Rembert Daems, Christoph Knochenhauer, Manfred Opper, Tolga Birdal, and Wojciech Samek. "Fractional diffusion bridge models". In: *Advances in Neural Information Processing Systems* 38 (2026), pp. 114521–114571.

[45] Aleksandr Aleksandrovich Novikov. "On an identity for stochastic integrals". In: *Teoriya Veroyatnostei i ee Primeneniya* 17.4 (1972), pp. 761–765.

[46] Bernt Øksendal. "Stochastic differential equations". In: *Stochastic differential equations: an introduction with applications*. Springer, 2003, pp. 38–50.

[47] William Peebles and Saining Xie. "Scalable diffusion models with transformers". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 4195–4205.

[48] Stefano Peluchetti. "Diffusion bridge mixture transports, Schrödinger bridge problems and generative modeling". In: *Journal of Machine Learning Research* 24.374 (2023), pp. 1–51.

[49] Stefano Peluchetti. "Non-denoising forward-time diffusions". In: *arXiv preprint arXiv:2312.14589* (2023).

[50] Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael Samuel Albergo, and Yee Whye Teh. "Meta Flow Maps enable scalable reward alignment". In: *Forty-third International Conference on Machine Learning*. 2026. URL: https://openreview.net/forum?id=K5gV8Yptne.

[51] Johann Radon. *Theorie und Anwendungen der absolut additiven Mengenfunktionen*. Hölder, 1913.

[52] Hannes Risken. "Fokker-Planck equation". In: *The Fokker-Planck equation: methods of solution and applications*. Springer, 1989, pp. 63–95.

[53] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695.

[54] Andy Shih, Dorsa Sadigh, and Stefano Ermon. "Training and inference on any-order autoregressive models the right way". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 2762–2775.

[55] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution". In: *Advances in neural information processing systems* 32 (2019).

[56] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021.

[57] J Michael Steele. *Stochastic calculus and financial applications*. Vol. 1. Springer, 2001.

[58] Mingzhen Sun, Weining Wang, Gen Li, Jiawei Liu, Jiahui Sun, Wanquan Feng, Shanshan Lao, SiYu Zhou, Qian He, and Jing Liu. "AR-diffusion: Asynchronous video generation with auto-regressive diffusion". In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 7364–7373.

[59] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the Inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.

[60] Panagiotis Theodoropoulos, Augustinos Saravanos, Evangelos Theodorou, and Guan-Horng Liu. "Momentum multi-marginal Schrödinger bridge matching". In: *Advances in Neural Information Processing Systems* 38 (2025), pp. 85885–85922.

[61] Alexander Y Tong, Nikolay Malkin, Kilian Fatras, Lazar Atanackovic, Yanlei Zhang, Guillaume Huguet, Guy Wolf, and Yoshua Bengio. "Simulation-Free Schrödinger Bridges via Score and Flow Matching". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2024, pp. 1279–1287.

[62] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. "VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training". In: *Advances in neural information processing systems* 35 (2022), pp. 10078–10093.

[63] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. *FVD: A new Metric for Video Generation*. 2019. URL: https://openreview.net/forum?id=rylgEULtdN.

[64] Mark Veillette, Siddharth Samsi, and Chris Mattioli. "SEVIR: A storm event imagery dataset for deep learning applications in radar and satellite meteorology". In: *Advances in neural information processing systems* 33 (2020), pp. 22009–22019.

[65] Pascal Vincent. "A connection between score matching and denoising autoencoders". In: *Neural computation* 23.7 (2011), pp. 1661–1674.

[66] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. "VideoMAE v2: Scaling video masked autoencoders with dual masking". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 14549–14560.

[67] Yuji Wang, Zehua Chen, Chen Xiaoyu, Yixiang Wei, Jun Zhu, and Jianfei Chen. "FrameBridge: Improving Image-to-Video Generation with Bridge Models". In: *International Conference on Machine Learning*. PMLR. 2025, pp. 62596–62626.

[68] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

[69] Biao Zhang and Rico Sennrich. "Root mean square layer normalization". In: *Advances in neural information processing systems* 32 (2019).

[70] Jiangning Zhang, Chao Xu, Liang Liu, Mengmeng Wang, Xia Wu, Yong Liu, and Yunliang Jiang. "DTVNet: Dynamic time-lapse video generation via single still image". In: *European conference on computer vision*. Springer. 2020, pp. 300–315.

[71] Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. "Frame context packing and drift prevention in next-frame-prediction video diffusion models". In: *Advances in Neural Information Processing Systems* 38 (2025), pp. 30546–30566.

[72] Lvmin Zhang, Shengqu Cai, Muyang Li, Chong Zeng, Beijia Lu, Anyi Rao, Song Han, Gordon Wetzstein, and Maneesh Agrawala. "TinyHistory: Lightweight Video History Embeddings via Two-Stage Context Learning". In: *European Conference on Computer Vision* (2026).

[73] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.

[74] Xi Nicole Zhang, Yuan Pu, Yuki Kawamura, Andrew Loza, Yoshua Bengio, Dennis Shung, and Alexander Tong. "Trajectory flow matching with applications to clinical time series modelling". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 107198–107224.

[75] Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. "Denoising diffusion bridge models". In: *International Conference on Learning Representations*. Vol. 2024. 2024, pp. 8160–8171.

[76] Hao Zhu, Wayne Wu, Wentao Zhu, Liming Jiang, Siwei Tang, Li Zhang, Ziwei Liu, and Chen Change Loy. "CelebV-HQ: A large-scale video facial attributes dataset". In: *European conference on computer vision*. Springer. 2022, pp. 650–667.

A Data-Generating SDE via Doob h-Transform

In this section we prove Theorem 1 and establish the regularity conditions.

Let $0 = t_0 < t_1 < \cdots < t_L$ be fixed physical times and $x_0 \in \mathbb{R}^d$ an initial condition. The *base (prior) process* $(X_t)_{t\in[0,t_L]}$ is the Ornstein–Uhlenbeck (OU) SDE

$$dX_t = -a(t)X_t\,dt + \sigma(t)\,dW_t, \qquad X_0 = x_0, \tag{18}$$

where W is a d-dimensional $\mathbb{P}$-Brownian motion (the base-process driver), $a : [0, t_L] \to \mathbb{R}$, and $\sigma : [0, t_L] \to (0, \infty)$. We reserve the symbol B for the $\mathbb{Q}$-Brownian motion constructed in Step 5. Denote by $\mathbb{P}$ the induced law on path space $C([0, t_L]; \mathbb{R}^d)$, and by $p_{base}(x_{t_1}, \ldots, x_{t_L} \mid x_0)$ the finite dimensional density under $\mathbb{P}$.

Recall that at time $t \in (t_{i^*}, t_{i^*+1})$, we write the *observed past* as

$$\mathbf{x}_{0:i^*} := [(0, x_0)] \oplus [(t_j, x_{t_j}) \mid j \le i^*], \tag{19}$$

where i^* is the index of the most recent observation, $i^* := \max\{i : t_i < t\}$. The current state x_t is *not* included in $\mathbf{x}_{0:i^*}$; it appears as a separate explicit argument in all functions below.

A.1 Regularity Conditions

The following conditions are assumed throughout the proof. Each plays a distinct role.

- **(R1) Well-posedness of the base SDE.** a and σ are bounded and measurable on $[0, t_L]$, and $\sigma(t) \ge \sigma_{\min} > 0$ for all t. Under these conditions, the linear SDE (18) has a unique strong solution, and the finite dimensional distribution $p_{base}(x_{t_1}, \ldots, x_{t_L} \mid x_0)$ is a non-degenerate Gaussian for every x_0. The lower bound on σ is essential: it prevents the base process from degenerating into a deterministic ODE, which would violate absolute continuity.
- **(R2) Absolute continuity.** For $\mathbb{P}$-a.e. x_0, $p_{data}(x_{t_1}, \ldots, x_{t_L} \mid x_0) \ll p_{base}(x_{t_1}, \ldots, x_{t_L} \mid x_0)$, i.e., the data distribution assigns no mass to sets that the base process cannot reach. This guarantees the Radon–Nikodym derivative $\frac{d\mathbb{Q}}{d\mathbb{P}} = \frac{p_{data}}{p_{base}}$ exists as an $L^1(\mathbb{P})$ density. It is implied by **(R1)** whenever p_{data} is a continuous distribution without atoms, since the OU process has full Gaussian support.
- **(R3) Full support / strict positivity.** $p_{data}(x_{t_1}, \ldots, x_{t_L} \mid x_0) > 0$ p_{base}-almost everywhere. This ensures the change-of-measure martingale $Z_t := \mathbb{E}_{\mathbb{P}}[d\mathbb{Q}/d\mathbb{P} \mid \mathcal{F}_t]$ satisfies $Z_t > 0$ $\mathbb{P}$-almost surely, so that $\log Z_t$ and the score $\nabla_{x_t} \log u$ are well-defined.
- **(R4) Square-integrability of the Radon–Nikodym derivative.** $\mathbb{E}_{\mathbb{P}}[(d\mathbb{Q}/d\mathbb{P})^2] < \infty$, equivalently $Z_T \in L^2(\mathbb{P})$. This implies $Z_t \in L^2(\mathbb{P})$ for all t (by the conditional Jensen inequality), which is the integrability condition required for Z_t to be a bona fide (not merely local) $\mathbb{P}$-martingale. It holds whenever p_{data}/p_{base} is square-integrable under p_{base}, a mild condition on the tails of p_{data}.
- **(R5) Smoothness of u.** The function $u(t, x_t; \mathbf{x}_{0:i^*}) := Z_t$ satisfies $u > 0$, $u \in C^{1,2}$ jointly in (t, x_t) on each open interval (t_{i^*}, t_{i^*+1}), and the gradient $\nabla_{x_t} \log u$ is locally square-integrable in t. The $C^{1,2}$ condition is needed to apply Itô's formula to u and to $\log u$. It follows from standard parabolic PDE regularity when p_{data} has sufficiently smooth conditional densities; in our setting it holds on the open intervals because the OU Green's function (the base transition kernel) is smooth.
- **(R6) Novikov's condition.**

  $$\mathbb{E}_{\mathbb{P}}\left[\exp\left(\tfrac{1}{2}\int_0^{t_L} \|\sigma(s)\,\nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i_s^*})\|^2\,ds\right)\right] < \infty.$$

  This is the classical sufficient condition [45] that upgrades a local $\mathbb{P}$-martingale to a true $\mathbb{P}$-martingale. It is invoked in Step 4 of the proof below to guarantee that the Doléans–Dade exponential Z_t is a true martingale and that Girsanov's theorem [20] applies without qualification. In practice it is satisfied because the score $\nabla_{x_t} \log u$ coincides with the neural-network output $\mathbf{f}_\theta$, which is bounded on compact time intervals under any finite-parameter model.

(R7) Markov property of the base process. For $s < t < \tau$ all in $[0, t_L]$, $p_{base}(x_\tau \mid x_{[0,t]}) = p_{base}(x_\tau \mid x_t)$ under $\mathbb{P}$. The OU process is a Markov process driven by Brownian motion, so this holds automatically. It is used in Step 6 to factor the conditional base density and reduce the integral over future states to a one-step expectation over $x_{t_{i^*+1}}$ alone.

A.2 Gaussian Base Process: Score Target and Noising Kernel

The following lemma establishes closed-form expressions for the two quantities that enter both the SDE drift and the training loss: the *score target* $\nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t)$ and the *noising kernel* $p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$. Both follow from the Gaussian process structure of the OU base process.

Define the resolvent and covariance kernel of the OU base process (18) conditioned on $x_{t_{i^*}}$:

$$\Phi(s,t) := \exp\left(\int_s^t -a(u)\,du\right), \tag{20}$$

$$C_{t_{i^*}}(\tau_a, \tau_b) := \int_{t_{i^*}}^{\min(\tau_a,\tau_b)} \Phi(s,\tau_a)\,\Phi(s,\tau_b)\,\sigma(s)^2\,ds. \tag{21}$$

Lemma 1 (Closed-form score target and noising kernel)**.** *Let $t_{i^*} \le t < t_{i^*+1}$. Under the OU base process* (18) *with regularity condition **(R1)**:*

*(i) **Score target.** The conditional distribution $x_{t_{i^*+1}} \mid x_t$ under p_{base} is Gaussian with mean $\Phi(t, t_{i^*+1})x_t$ and variance $C_t(t_{i^*+1}, t_{i^*+1})I$. Consequently,*

$$\nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t) = \frac{\Phi(t, t_{i^*+1})}{C_t(t_{i^*+1}, t_{i^*+1})}\big(x_{t_{i^*+1}} - \Phi(t, t_{i^*+1})x_t\big). \tag{22}$$

This quantity is non-singular and linear in $(x_t, x_{t_{i^*+1}})$ for all $t \in [t_{i^*}, t_{i^*+1})$.

*(ii) **Noising kernel.** The conditional distribution $x_t \mid (x_{t_{i^*}}, x_{t_{i^*+1}})$ under p_{base} is Gaussian with*

$$\mu_{t|t_{i^*},t_{i^*+1}}(x_{t_{i^*}}, x_{t_{i^*+1}}) = \Phi(t_{i^*}, t)x_{t_{i^*}} + \frac{C_{t_{i^*}}(t, t_{i^*+1})}{C_{t_{i^*}}(t_{i^*+1}, t_{i^*+1})}\big(x_{t_{i^*+1}} - \Phi(t_{i^*}, t_{i^*+1})x_{t_{i^*}}\big), \tag{23}$$

$$\Sigma_{t|t_{i^*},t_{i^*+1}} = \left(C_{t_{i^*}}(t,t) - \frac{C_{t_{i^*}}(t, t_{i^*+1})^2}{C_{t_{i^*}}(t_{i^*+1}, t_{i^*+1})}\right) I. \tag{24}$$

The variance $\Sigma_{t|t_{i^*+1}}$ is strictly positive for $t \in (t_{i^*}, t_{i^*+1})$ and vanishes at both endpoints.

Proof. Part (i) follows from the explicit variation-of-constants solution of the linear SDE (18) and Itô's isometry; see [46], Chapter 5. The score formula is then the log-gradient of the resulting Gaussian density. Part (ii) is Gaussian process conditioning applied to the jointly Gaussian pair $(x_t, x_{t_{i^*+1}})$ given $x_{t_{i^*}}$; the mean and variance (23)–(24) follow from the standard formulas of [5] (equations 2.81–2.82), with covariances computed via Itô's isometry and the independence of non-overlapping Brownian increments [46].

The OU process (18) has the explicit variation-of-constants solution, verified by differentiating with the Leibniz integral rule:

$$x_t = \Phi(t_{i^*}, t)\,x_{t_{i^*}} + \int_{t_{i^*}}^t \Phi(s,t)\,\sigma(s)\,dW_s. \tag{25}$$

Part (i): Score target. Setting $t_{i^*} \leftarrow t$ in (25) and letting the process run to t_{i^*+1}:

$$x_{t_{i^*+1}} = \Phi(t, t_{i^*+1})\,x_t + \int_t^{t_{i^*+1}} \Phi(s, t_{i^*+1})\,\sigma(s)\,dW_s. \tag{26}$$

Since W has independent increments, the stochastic integral has zero mean. By Itô's isometry:

$$\mathrm{Var}(x_{t_{i^*+1}} \mid x_t) = \int_t^{t_{i^*+1}} \Phi(s, t_{i^*+1})^2\,\sigma(s)^2\,ds \cdot I = C_t(t_{i^*+1}, t_{i^*+1})\,I.$$

Hence $x_{t_{i^*+1}} \mid x_t \sim \mathcal{N}\left(\Phi(t, t_{i^*+1})x_t,\ C_t(t_{i^*+1}, t_{i^*+1})I\right)$. Taking the log-gradient of this Gaussian density with respect to x_t:

$$\begin{aligned}\nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t) &= \nabla_{x_t}\left[-\frac{\|x_{t_{i^*+1}} - \Phi(t, t_{i^*+1})x_t\|^2}{2\,C_t(t_{i^*+1}, t_{i^*+1})}\right] \\ &= \frac{\Phi(t, t_{i^*+1})}{C_t(t_{i^*+1}, t_{i^*+1})}\big(x_{t_{i^*+1}} - \Phi(t, t_{i^*+1})x_t\big),\end{aligned}$$

which is (22). Non-singularity holds because $C_t(t_{i^*+1}, t_{i^*+1}) = \int_t^{t_{i^*+1}} \Phi(s, t_{i^*+1})^2\sigma(s)^2\, ds > 0$ for all $t < t_{i^*+1}$, by **(R1)** ($\sigma \geq \sigma_{\min} > 0$).

Part (ii): Noising kernel. Conditioned on $x_{t_{i^*}}$, both x_t and $x_{t_{i^*+1}}$ are affine functions of W (by (25)), so the pair is jointly Gaussian with means

$$\mu_t(x_{t_{i^*}}) = \Phi(t_{i^*}, t)\, x_{t_{i^*}}, \qquad \mu_{t_{i^*+1}}(x_{t_{i^*}}) = \Phi(t_{i^*}, t_{i^*+1})\, x_{t_{i^*}}.$$

We compute the covariances using Itô's isometry. Off-diagonal entries of the covariance matrix are zero because W has independent components. For the diagonal entries, the cross-covariance involves the inner product of the two stochastic integrals:

$$\Sigma_{t,t_{i^*+1}}[j,j] = \mathbb{E}_{\mathbb{P}}\left[\int_{t_{i^*}}^{t} \Phi(s,t)\sigma(s)\, dW_s[j] \cdot \int_{t_{i^*}}^{t_{i^*+1}} \Phi(s, t_{i^*+1})\sigma(s)\, dW_s[j]\right].$$

Split the second integral at t; the piece over (t, t_{i^*+1}) is independent of the first integral (non-overlapping increments), contributing zero. Itô's isometry on the overlapping part $[t_{i^*}, t]$ gives

$$\Sigma_{t,t_{i^*+1}}[j,j] = \int_{t_{i^*}}^{t} \Phi(s,t)\,\Phi(s,t_{i^*+1})\,\sigma(s)^2\, ds = C_{t_{i^*}}(t, t_{i^*+1}).$$

By the same argument,

$$\Sigma_{t,t} = C_{t_{i^*}}(t,t)\, I, \qquad \Sigma_{t_{i^*+1},t_{i^*+1}} = C_{t_{i^*}}(t_{i^*+1}, t_{i^*+1})\, I, \qquad \Sigma_{t,t_{i^*+1}} = C_{t_{i^*}}(t, t_{i^*+1})\, I.$$

Applying the standard Gaussian conditional formulas ([5], equations 2.81–2.82):

$$\begin{aligned}\mu_{t|t_{i^*},t_{i^*+1}}(x_{t_{i^*}}, x_{t_{i^*+1}}) &= \mu_t(x_{t_{i^*}}) + \Sigma_{t,t_{i^*+1}}\Sigma^{-1}_{t_{i^*+1},t_{i^*+1}}(x_{t_{i^*+1}} - \mu_{t_{i^*+1}}(x_{t_{i^*}})), \\ \Sigma_{t|t_{i^*+1}} &= \Sigma_{t,t} - \Sigma_{t,t_{i^*+1}}\Sigma^{-1}_{t_{i^*+1},t_{i^*+1}}\Sigma_{t_{i^*+1},t},\end{aligned}$$

and substituting the scalar covariances yields (23)–(24).

Strict positivity of $\Sigma_{t|t_{i^*+1}}$ on (t_{i^*}, t_{i^*+1}) follows from the Cauchy–Schwarz inequality applied to the covariance kernel: by applying the Cauchy-Schwarz inequality and the non-negative integration from time $t = t$ to t_{i^*+1}, we have that

$$C_{t_{i^*}}(t, t_{i^*+1})^2 \ \leq\ C_{t_{i^*}}(t,t) \cdot C_{t_{i^*}}(t_{i^*+1}, t_{i^*+1}),$$

with strict inequality when $t \in (t_{i^*}, t_{i^*+1})$, since the integrand $\Phi(s,t)\Phi(s,t_{i^*+1})\sigma(s)^2$ is not a scalar multiple of $\Phi(s,t)^2\sigma(s)^2$ (the resolvent $\Phi(s,\cdot)$ is strictly monotone in its second argument for $a(t) \neq 0$, and bounded away from a constant ratio by **(R1)**). □

A.3 Proof of Theorem 1

Remark 1. *$s(t, x_t; \mathbf{x}_{0:i^*})$ in the theorem statement is simply another notation for the oracle score $\nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*})$, where u is the Doob h-function (Step 2 in proof). They are the same analytical object. The proof uses W_t for the $\mathbb{P}$-Brownian motion and B_t for the $\mathbb{Q}$-Brownian motion (Step 5 in proof).*

Learning s from data is addressed separately in Appendix B. The neural network $\mathbf{f}_\theta$ is a parametric approximation introduced that learns s; the two coincide, $\mathbf{f}_{\theta^*} = s$, only at the minimizer of the score matching loss.

Proof of Theorem 1. The proof has seven steps. Steps 1–2 establish the target measure $\mathbb{Q}$; Steps 3–6 extract the modified SDE dynamics via Girsanov's theorem; Step 7 derives the score.

Step 1: Base process and target change-of-measure. We take the OU process (18) as the base process under $\mathbb{P}$. Its transition densities are Gaussian by linearity, and by **(R1)** the finite dimensional distribution $p_{base}(x_{t_1}, \dots, x_{t_L} \mid x_0)$ is a non-degenerate Gaussian with full support on $(\mathbb{R}^d)^L$.

We wish to define a new measure $\mathbb{Q}$ on path space such that $(X_0, X_{t_1}, \dots, X_{t_L})$ under $\mathbb{Q}$ has joint density $p_{data}(x_0, x_{t_1}, \dots, x_{t_L})$. Equivalently, conditioned on $X_0 = x_0$, the grid $(X_{t_1}, \dots, X_{t_L})$ should have density $p_{data}(\cdot \mid x_0)$. By **(R2)**, the Radon–Nikodym derivative is well-defined on the σ-algebra generated by the grid:

$$\left.\frac{d\mathbb{Q}}{d\mathbb{P}}\right|_{\sigma(X_{t_1},\dots,X_{t_L})} = \frac{p_{data}(x_{t_1}, \dots, x_{t_L} \mid x_0)}{p_{base}(x_{t_1}, \dots, x_{t_L} \mid x_0)}. \tag{27}$$

For any event A in the path σ-algebra, $\mathbb{Q}(A) = \mathbb{E}_{\mathbb{P}}[\mathbb{1}_A \, d\mathbb{Q}/d\mathbb{P}]$. By construction, for every Borel set $S \subseteq (\mathbb{R}^d)^L$,

$$\mathbb{Q}\left((X_{t_1}, \dots, X_{t_L}) \in S\right) = \int_S p_{data}(x_{t_1}, \dots, x_{t_L} \mid x_0) \, dx_{t_1} \cdots dx_{t_L},$$

so the grid under $\mathbb{Q}$ has exactly the desired density.

Step 2: The Doob h-transform is a $\mathbb{P}$-martingale. In order to extract the SDE dynamics under $\mathbb{Q}$, we need the Radon–Nikodym derivative [51, 42] as a process indexed by t. Define

$$Z_t := h(t, x_t; \mathbf{x}_{0:i^*}) := \mathbb{E}_{\mathbb{P}}\left[\frac{d\mathbb{Q}}{d\mathbb{P}} \,\middle|\, \mathcal{F}_t\right] = \mathbb{E}_{\mathbb{P}}\left[\frac{p_{data}(X_{t_1}, \dots, X_{t_L} \mid X_0)}{p_{base}(X_{t_1}, \dots, X_{t_L} \mid X_0)} \,\middle|\, \mathcal{F}_t\right]. \tag{28}$$

This is the *Doob h-function* [17]: a positive $(\mathcal{F}_t, \mathbb{P})$-adapted process obtained by conditioning the terminal Radon–Nikodym derivative on the current information.

Z_t is a $\mathbb{P}$-martingale. For any $\tau > t$, the tower property of conditional expectation gives

$$\mathbb{E}_{\mathbb{P}}[Z_\tau \mid \mathcal{F}_t] = \mathbb{E}_{\mathbb{P}}\left[\mathbb{E}_{\mathbb{P}}\left[\frac{d\mathbb{Q}}{d\mathbb{P}} \,\middle|\, \mathcal{F}_\tau\right] \,\middle|\, \mathcal{F}_t\right] = \mathbb{E}_{\mathbb{P}}\left[\frac{d\mathbb{Q}}{d\mathbb{P}} \,\middle|\, \mathcal{F}_t\right] = Z_t.$$

By **(R4)**, $Z_t \in L^2(\mathbb{P})$ for all t, so this is a true (square-integrable) martingale, not merely a local one.

Normalisation $Z_0 = 1$. At $t = 0$, $\mathcal{F}_0 = \sigma(X_0)$ and conditioning on $X_0 = x_0$ gives

$$Z_0 = \mathbb{E}_{\mathbb{P}}\left[\frac{d\mathbb{Q}}{d\mathbb{P}}\right] = \int \frac{p_{data}(x_{t_1}, \dots, x_{t_L} \mid x_0)}{p_{base}(x_{t_1}, \dots, x_{t_L} \mid x_0)} p_{base}(x_{t_1}, \dots, x_{t_L} \mid x_0) \, dx_{t_1} \cdots dx_{t_L} = 1.$$

Step 3: Itô's lemma on Z_t. To obtain the dynamics of Z_t along the OU paths, write $Z_t = u(t, x_t; \mathbf{x}_{0:i^*})$, making explicit the dependence on (t, x_t) and on the already-realized path history $(x_0, x_{t_1}, \dots, x_{t_{i^*}})$ from the filtration. The realized history is $\mathcal{F}_t$-measurable and plays the role of a parameter; only the current state x_t is "live" in the Itô sense.

By **(R5)**, $u \in C^{1,2}$ in (t, x_t). Applying Itô's formula [32, 31] to $u(t, x_t; \mathbf{x}_{0:i^*})$ along (18):

$$dZ_t = \left(\frac{\partial u}{\partial t} + \tfrac{1}{2}\mathrm{Tr}\big(\sigma(t)^2 \nabla_{x_t}^2 u\big) - a(t) x_t \cdot \nabla_{x_t} u\right) dt + \nabla_{x_t} u \cdot \sigma(t) \, dW_t. \tag{29}$$

Since Z_t is a $\mathbb{P}$-martingale (Step 2), its dt-drift must vanish $\mathbb{P}$-almost surely. The dW_t term already has zero mean, so the bracketed expression in (29) equals zero, leaving

$$dZ_t = \nabla_{x_t} u(t, x_t; \mathbf{x}_{0:i^*}) \cdot \sigma(t) \, dW_t. \tag{30}$$

This identifies Z_t as a *stochastic integral*: it has no drift and its quadratic variation is $d\langle Z \rangle_t = \|\nabla_{x_t} u\|^2 \sigma(t)^2 \, dt$.

Step 4: Doléans–Dade exponential form. To apply Girsanov's theorem [20] we must write Z_t in the specific exponential martingale form of Doléans–Dade [16]. Apply Itô's formula to $\log Z_t$ (valid since $Z_t > 0$ by **(R3)**):

$$d \log Z_t = \frac{1}{Z_t} \, dZ_t - \frac{1}{2Z_t^2} (dZ_t)^2 = \frac{\nabla_{x_t} u}{u} \cdot \sigma(t) \, dW_t - \frac{1}{2} \left\| \frac{\nabla_{x_t} u}{u} \, \sigma(t) \right\|^2 dt. \tag{31}$$

Using the chain rule $\nabla_{x_t} u/u = \nabla_{x_t} \log u$ and integrating (31) from 0 to t:

$$\log Z_t = \int_0^t \sigma(s)\, \nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i_s^*}) \cdot dW_s - \frac{1}{2} \int_0^t \|\sigma(s)\, \nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i_s^*})\|^2 \, ds. \quad (32)$$

Exponentiating:

$$Z_t = \exp\left(\int_0^t Y_s \cdot dW_s - \frac{1}{2} \int_0^t \|Y_s\|^2 \, ds \right), \qquad Y_s := \sigma(s)\, \nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i_s^*}). \quad (33)$$

This is the Doléans–Dade exponential (Theorem 3.5.1 of Karatzas–Shreve [33]) with drift process Y_s. By **(R6)** (Novikov's condition), Z_t is a true (not just local) $\mathbb{P}$-martingale, confirming that the exponential in (33) has mean 1 and that all subsequent applications of Girsanov's theorem are valid.

Step 5: Girsanov's theorem — the $\mathbb{Q}$-Brownian motion. Having written Z_t in the required exponential form (33) with $Y_s = \sigma(s)\nabla_{x_s} \log u$, Girsanov's theorem ([20], Theorem 3.5.1 of [33]) states that

$$B_t := W_t - \int_0^t \sigma(s)\, \nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i_s^*})\, ds \quad (34)$$

is a standard Brownian motion under $\mathbb{Q}$. Equivalently:

$$dW_t = dB_t + \sigma(t)\, \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*})\, dt. \quad (35)$$

Under the new measure $\mathbb{Q}$, the $\mathbb{P}$-driver W_t is no longer a martingale; B_t is the process that plays the role of standard Brownian motion, acquiring the additional drift $\sigma(t)\nabla_{x_t} \log u$ relative to W_t.

Step 6: Substitution into the OU SDE. Substitute (35) into the base OU SDE (18):

$$\begin{aligned} dX_t &= -a(t)X_t\, dt + \sigma(t)\, dW_t \\ &= -a(t)X_t\, dt + \sigma(t)\, [dB_t + \sigma(t)\, \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*})\, dt] \\ &= \Big(-a(t)X_t + \sigma(t)^2\, \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*})\Big) dt + \sigma(t)\, dB_t. \end{aligned} \quad (36)$$

This is the claimed SDE (6). Since $B^{\mathbb{Q}}$ is a $\mathbb{Q}$-Brownian motion, the SDE (36) is driven by Gaussian increments under $\mathbb{Q}$, and by construction its finite dimensional law at times $t_1, \ldots, t_L$ is $p_{data}(\cdot \mid x_0)$.

Step 7: Analytic form of $\nabla_{x_t} \log u$. It remains to derive the explicit form (7) of the score. Fix $t \in (t_{i^*}, t_{i^*+1})$ and expand $Z_t = u(t, x_t; \mathbf{x}_{0:i^*})$ from its definition (28) by conditioning on the already-realized states $\mathcal{F}_{t_{i^*}} = (x_0, x_{t_1}, \ldots, x_{t_{i^*}})$:

$$\begin{aligned} Z_t &= \mathbb{E}_{\mathbb{P}} \left[\frac{p_{data}(X_{t_1}, \ldots, X_{t_L} \mid X_0)}{p_{base}(X_{t_1}, \ldots, X_{t_L} \mid X_0)} \,\middle|\, \mathcal{F}_t \right] \\ &= \underbrace{\frac{p_{data}(x_{t_1}, \ldots, x_{t_{i^*}} \mid x_0)}{p_{base}(x_{t_1}, \ldots, x_{t_{i^*}} \mid x_0)}}_{=:\, k,\ \mathcal{F}_{t_{i^*}}\text{-measurable}} \\ &\quad * \int \frac{p_{data}(x_{t_{i^*+1}}, \ldots, x_{t_L} \mid x_0, \ldots, x_{t_{i^*}})}{p_{base}(x_{t_{i^*+1}}, \ldots, x_{t_L} \mid x_0, \ldots, x_{t_{i^*}})}\, p_{base}(x_{t_{i^*+1}}, \ldots, x_{t_L} \mid \mathcal{F}_t)\, dx_{t_{i^*+1}} \cdots dx_{t_L}. \end{aligned} \quad (37)$$

By the Markov property **(R7)**, conditioned on $\mathcal{F}_t$ the future increments of the base process depend only on the current state x_t:

$$p_{base}(x_{t_{i^*+1}}, \ldots, x_{t_L} \mid \mathcal{F}_t) = p_{base}(x_{t_{i^*+1}} \mid x_t) \prod_{j=i^*+1}^{L-1} p_{base}(x_{t_{j+1}} \mid x_{t_j}).$$

Substituting into (37) and cancelling the telescoping Markov product against the denominator of the data/base ratio (noting the base Markov factorisation)

$$p_{base}(x_{t_{i^*+1}}, \ldots, x_{t_L} \mid x_{t_{i^*}}) = p_{base}(x_{t_{i^*+1}} \mid x_{t_{i^*}}) \prod_{j=i^*+1}^{L-1} p_{base}(x_{t_{j+1}} \mid x_{t_j}), \quad (38)$$

the product of future base transitions in numerator and denominator telescopes, and the integral over $x_{t_{i^*+2}}, \ldots, x_{t_L}$ of the normalised data density equals one:

$$Z_t = k \int p_{data}(x_{t_{i^*+1}} \mid x_0, \ldots, x_{t_{i^*}}) \cdot \frac{p_{base}(x_{t_{i^*+1}} \mid x_t)}{p_{base}(x_{t_{i^*+1}} \mid x_{t_{i^*}})} \, dx_{t_{i^*+1}}. \tag{39}$$

Applying Bayes' rule to the base process (we can apply Bayes' rule freely here because the Gaussian structure of the base process makes all conditional densities, whether forward or backward, analytically tractable),

$$\frac{p_{base}(x_{t_{i^*+1}} \mid x_t)}{p_{base}(x_{t_{i^*+1}} \mid x_{t_{i^*}})} = \frac{p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})}{p_{base}(x_t \mid x_{t_{i^*}})}, \tag{40}$$

and collecting:

$$Z_t = \frac{k}{p_{base}(x_t \mid x_{t_{i^*}})} \int p_{data}(x_{t_{i^*+1}} \mid x_0, \ldots, x_{t_{i^*}}) \, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) \, dx_{t_{i^*+1}}. \tag{41}$$

Taking $\nabla_{x_t} \log(\cdot)$ of (41), and noting that k is constant in x_t:

$$\begin{aligned} \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*}) = &-\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}) \\ &+ \nabla_{x_t} \log \int p_{data}(x_{t_{i^*+1}} \mid x_0, \ldots, x_{t_{i^*}}) \, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) \, dx_{t_{i^*+1}}. \end{aligned} \tag{42}$$

This establishes (7) in Theorem 1.

Combining Steps 1–7: the SDE (36) is driven by the $\mathbb{Q}$-Brownian motion B_t and has oracle drift $s = \nabla_{x_t} \log u$ as given by (42). Conditionally on $X_0 = x_0$ the finite dimensional law at $(t_1, \ldots, t_L)$ is $p_{data}(\cdot \mid x_0)$ by construction of $\mathbb{Q}$ in Step 1. Since $X_0 \sim p_{data}(x_0)$ by assumption,

$$\mathrm{Law}(X_0, X_{t_1}, \ldots, X_{t_L}) = p_{data}(x_0) \cdot p_{data}(x_{t_1}, \ldots, x_{t_L} \mid x_0) = p_{data}(x_0, x_{t_1}, \ldots, x_{t_L}).$$

□

B Learning the Score Function

The oracle score $s(t, x_t; \mathbf{x}_{0:i^*}) = \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*})$ from Theorem 1 is defined implicitly through the Doob h-function. This section makes its structure explicit, shows how it can be learned from data, and proves Theorem 2.

B.1 Analytic Form of the Score

Taking the log-gradient of the expression for u derived in Step 7 of Appendix A (equation (41)), the oracle score decomposes as

$$s(t, x_t; \mathbf{x}_{0:i^*}) = \nabla_{x_t} \log u(t, x_t; \mathbf{x}_{0:i^*}) \tag{43}$$

$$= -\underbrace{\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}})}_{\text{analytic}} + \underbrace{\nabla_{x_t} \log \int p_{data}(x_{t_{i^*+1}} \mid x_0, \ldots, x_{t_{i^*}}) \, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) \, dx_{t_{i^*+1}}}_{\text{to be learned}}. \tag{44}$$

The first term is the score of the OU marginal $p_{base}(x_t \mid x_{t_{i^*}})$, which is Gaussian and therefore available in closed form at any $t \in (t_{i^*}, t_{i^*+1})$. The second term is a log-normalizer over the next observation $x_{t_{i^*+1}}$ weighted by the data distribution, and is the term that must be approximated by a neural network.

The difficulty is that the first term diverges as $t \to t_{i^*}^+$: the OU marginal concentrates into a Dirac delta, making the gradient unbounded. A network that directly targets the sum would need to handle this singularity. We show below that the singularity cancels algebraically, leaving a well-posed training objective.

Remark 2 (Boundary interpretation at $t = t_{i^*}$)**.** *One might worry that when $t = t_{i^*}$ exactly, the gradient ∇_{x_t} in* (44) *accidentally differentiates through $x_{t_{i^*}}$, since the two coincide numerically. This is not the case for two reasons. First, t_{i^*} and t are distinct variables that happen to be equal at the boundary; ∇_{x_t} and $\nabla_{x_{t_{i^*}}}$ remain separate operators (analogous to pass-by-value versus pass-by-reference in programming). Second, Itô calculus is non-anticipative: in the Riemann approximation of the Itô integral, the integrand is evaluated at the left endpoint of each discretisation interval [57], so x_t is treated as a fixed value when taking ∇_{x_t}, independently of $x_{t_{i^*}}$, which was already realised in $\mathcal{F}_t$.*

B.2 Bridge Score Matching

A natural first learning objective uses the bridge kernel $p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$ as the noising distribution. By the denoising score matching identity [65], the minimizer of the squared-error loss against the bridge score $\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$ is the conditional expectation of that score given $(x_t, \mathbf{x}_{0:i^*})$, which equals the log-gradient of the mixture $\mathcal{M}(x_t) = \int p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})\, dx_{t_{i^*+1}}$. Comparing with (44), this is $s + \nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}})$, i.e. the oracle plus the singular term.

For a fixed trajectory segment and a score network $\mathbf{r}_\theta$, define the bridge loss

$$\mathcal{L}_{\text{bridge}}(\theta;\, t,\, \mathbf{x}_{0:t_{i^*}},\, x_{t_{i^*+1}}) \;:=\; \mathbb{E}_{p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})} \left[\left\| \mathbf{r}_\theta(t, t_{i^*+1}, x_t;\, \mathbf{x}_{0:i^*}) - \nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) \right\|_2^2 \right]. \quad (45)$$

Averaging over all trajectory segments and times gives the joint objective:

$$\mathcal{L}^{\mathbb{Q}}_{\text{bridge}}(\theta) \;:=\; \mathbb{E}_{t \sim \mathcal{U}(0,\, t_{L-1})}\, \mathbb{E}_{p_{data}(x_0, \ldots, x_{t_{i^*}})}\, \mathbb{E}_{p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})}\, \mathcal{L}_{\text{bridge}}(\theta;\, t,\, \mathbf{x}_{0:t_{i^*}},\, x_{t_{i^*+1}}). \quad (46)$$

This is a valid objective in the sense that its minimizer recovers the oracle (up to the singular additive term). However, because the target $\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$ diverges as $t \to t_{i^*}^+$, it is more natural to reparametrize the network so that the diverging term is absorbed analytically. This is carried out in the proof of Theorem 2 below.

B.3 Proof of Theorem 2

Proof of Theorem 2. The proof has two steps. Step 1 cancels the singularity via a reparametrization that converts the bridge loss into the simplified loss $\mathcal{L}_{\text{DSM}}$ of the theorem statement. Step 2 identifies the minimizer of this simplified loss with the oracle score s.

Step 1: Singularity cancellation.

By Bayes' rule applied to the Markov base process:

$$\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) \;=\; \nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t) \;+\; \nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}). \quad (47)$$

Define a reparametrized network $\mathbf{f}_\theta$ by

$$\mathbf{r}_\theta(t, t_{i^*+1}, x_t;\, \mathbf{x}_{0:i^*}) \;:=\; \mathbf{f}_\theta(t, t_{i^*+1}, x_t;\, \mathbf{x}_{0:i^*}) \;+\; \nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}), \quad (48)$$

so that $\mathbf{f}_\theta$ carries no singular term (it contributes the singular term analytically through (48)). Substituting (48) into the bridge loss (45) and applying (47):

$$\begin{aligned}
&\mathcal{L}_{\text{bridge}}(\theta) \\
&= \mathbb{E}\left[\left\|\mathbf{f}_\theta + \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}) - \nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t) - \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}})\right\|^2\right] \\
&= \mathbb{E}\left[\left\|\mathbf{f}_\theta(t, t_{i^*+1}, x_t;\, \mathbf{x}_{0:i^*}) - \nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\right\|^2\right]. \qquad (49)
\end{aligned}$$

The two $\nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}})$ terms cancel exactly. The residual target $\nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)$ is non-singular on (t_{i^*}, t_{i^*+1}) since $x_{t_{i^*+1}} \neq x_t$ throughout the open interval. Averaging over all segments gives the full objective

$$\begin{aligned}
&\mathcal{L}^{\mathbb{Q}}_{\text{DSM}}(\theta) \\
&= \mathbb{E}_{t\sim\mathcal{U}(0,\,t_{L-1})}\,\mathbb{E}_{\substack{p_{data}(x_0,\ldots,x_{t_{i^*+1}}) \\ p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})}}\left[\left\|\mathbf{f}_\theta(t, t_{i^*+1}, x_t;\, \mathbf{x}_{0:i^*}) - \nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\right\|^2\right], \qquad (50)
\end{aligned}$$

which is $\mathcal{L}_{\text{DSM}}(\hat{s})$ from Theorem 2 with $\hat{s} = \mathbf{f}_\theta$. Both quantities inside the expectation are available in closed form from Lemma 1: the noising kernel provides x_t given $(x_{t_{i^*}}, x_{t_{i^*+1}})$, and the score target is given by (22).

Step 2: The minimizer of $\mathcal{L}^{\mathbb{Q}}_{\text{DSM}}$ is the oracle score s.

Fix $t \in (t_{i^*}, t_{i^*+1})$ and $\mathbf{x}_{0:i^*}$. Let $p(x_{t_{i^*+1}}, x_t \mid \mathbf{x}_{0:i^*}) := p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$ denote the joint sampling distribution. By the bias-variance decomposition:

$$\begin{aligned}
&\mathbb{E}\left[\left\|\mathbf{f}(t, x_t;\, \mathbf{x}_{0:i^*}) - \nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\right\|^2\right] \\
&\quad= \mathbb{E}_{x_t}\left[\left\|\mathbf{f} - \mathbb{E}_{x_{t_{i^*+1}}\mid x_t}\left[\nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\right]\right\|^2\right] + \text{ const}, \qquad (51)
\end{aligned}$$

where the constant is independent of $\mathbf{f}$. The unique minimizer is therefore the conditional mean

$$\mathbf{f}^*(t, x_t;\, \mathbf{x}_{0:i^*}) \;=\; \mathbb{E}_{x_{t_{i^*+1}}\mid x_t,\, \mathbf{x}_{0:i^*}}\left[\nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\right], \qquad (52)$$

where $p(x_{t_{i^*+1}} \mid x_t, \mathbf{x}_{0:i^*}) \propto p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})$.

It remains to show $\mathbf{f}^* = s$. Define

$$\mathcal{M}(x_t) := \int p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\, p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})\, dx_{t_{i^*+1}},$$

the denominator of the conditional. Writing out (52) explicitly:

$$\mathbf{f}^* \;=\; \frac{1}{\mathcal{M}(x_t)}\int p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\; p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})\; \nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t)\; dx_{t_{i^*+1}}.$$

Apply (47) to substitute

$$\nabla_{x_t}\log p_{base}(x_{t_{i^*+1}} \mid x_t) = \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}}) - \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}),$$

we get

$$\begin{aligned}
\mathbf{f}^* \;=\; \frac{1}{\mathcal{M}(x_t)}\int p_{data}(x_{t_{i^*+1}} \mid \mathbf{x}_{0:i^*})\; \nabla_{x_t} p_{base}(x_t \mid x_{t_{i^*}}, x_{t_{i^*+1}})\; dx_{t_{i^*+1}} \\
-\; \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}).
\end{aligned}$$

Interchanging ∇_{x_t} with the integral (justified since $p_{base}(\cdot \mid x_{t_{i^*}}, x_{t_{i^*+1}})$ is Gaussian in x_t, so dominated convergence applies):

$$\mathbf{f}^* \;=\; \frac{\nabla_{x_t}\mathcal{M}(x_t)}{\mathcal{M}(x_t)} \;-\; \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}) \;=\; \nabla_{x_t}\log \mathcal{M}(x_t) \;-\; \nabla_{x_t}\log p_{base}(x_t \mid x_{t_{i^*}}). \qquad (53)$$

From (41) in Appendix A, $u(t, x_t; \mathbf{x}_{0:i^*}) = k\,\mathcal{M}(x_t)/p_{base}(x_t \mid x_{t_{i^*}})$ with $k > 0$ independent of x_t. Therefore

$$s(t, x_t; \mathbf{x}_{0:i^*}) \;=\; \nabla_{x_t} \log u \;=\; \nabla_{x_t} \log \mathcal{M}(x_t) - \nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}}) \;=\; \mathbf{f}^*.$$

The oracle score s is the unique minimizer of $\mathcal{L}^{\mathbb{Q}}_{\mathrm{DSM}}$ (equivalently, $\mathcal{L}_{\mathrm{DSM}}(\hat{s})$ with $\hat{s} = \mathbf{f}_\theta$). □

B.4 Non-Singularity and Novikov's Condition

The simplified loss (50) eliminates both practical and theoretical concerns associated with the original two-term score (44).

On the practical side, the target $\nabla_{x_t} \log p_{base}(x_{t_{i^*+1}} \mid x_t)$ is bounded and smooth on the interval $[t_{i^*}, t_{i^*+1})$ because $x_{t_{i^*+1}} \neq x_t$ by construction, and the singular term with $x_{t_{i^*}}$ got canceled out in (50). The network $\mathbf{f}_\theta$ therefore has a bounded, non-singular regression target throughout training.

On the theoretical side, the SDE (6) in Theorem 1 requires Novikov's condition (Regularity R6 in Appendix A):

$$\mathbb{E}_{\mathbb{P}}\left[\exp\left(\tfrac{1}{2}\int_0^{t_L} \|\sigma(s)\,\nabla_{x_s} \log u(s, x_s; \mathbf{x}_{0:i^*_s})\|^2\, ds\right)\right] < \infty.$$

From (48) and (47), the learnable part of $\nabla_{x_t} \log u$ is $\mathbf{f}_{\theta^*} = s$, which depends on $x_{t_{i^*+1}} \neq x_t$ and is therefore non-singular on the open interval. The singular term $\nabla_{x_t} \log p_{base}(x_t \mid x_{t_{i^*}})$ is present in the oracle score s analytically (through the reparametrization (48)), but its contribution to the Itô integral is square-integrable on $[t_{i^*}, t_{i^*+1})$ since it got canceled out in (50). The integrand is therefore square-integrable on every compact sub-interval, and Novikov's condition is satisfied [45, 20, 57].

C Generic Conditional Diffusion Bridges Cannot Capture the Path Measure

We prove Theorem 3 here.

Proof. Marginal agreement at $(0, t_1, \ldots, t_L)$ holds by hypothesis. To see that the full path laws differ, compare quadratic variations, which are pathwise functionals and hence law-determined.

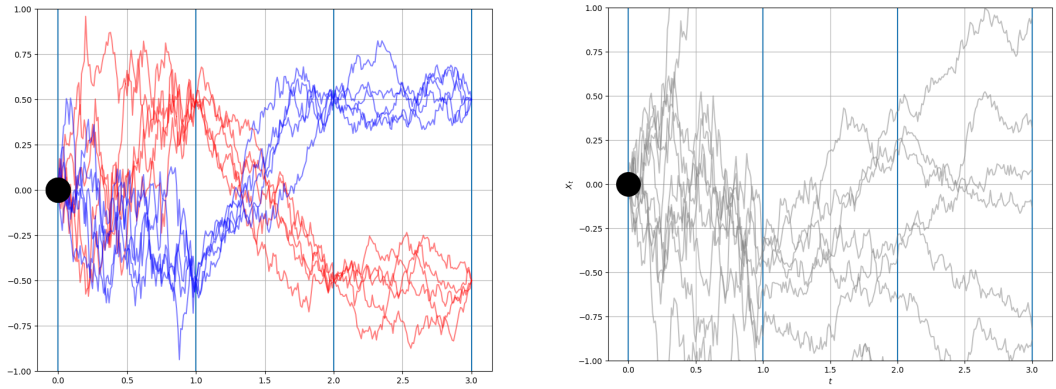
For X, quadratic variation is $\langle X \rangle_t = \int_0^t \sigma(s)^2\, ds$ (drift does not contribute) [32, 31, 33]. For Y, the deterministic time change on $[t_{i-1}, t_i]$ has rate $1/(t_i - t_{i-1})$, so on this subinterval Y has quadratic variation density

$$s \;\mapsto\; \tilde{\sigma}\big((s - t_{i-1})/(t_i - t_{i-1})\big)^2 \,/\, (t_i - t_{i-1}).$$

Equality of laws would force this to equal $\sigma(s)^2$ pointwise a.e. on every (t_{i-1}, t_i) — a rigid constraint independent of the marginal-matching hypothesis on $\tilde{\mathbf{f}}_\theta$. Generic choices violate it (*e.g.*, $\tilde{\sigma}$ that does not depend on i when $\exists i : t_i - t_{i-1} \neq t_L/L$, any constant $\tilde{\sigma}$ against a non-constant σ). So, $\mathrm{Law}(Y) \neq \mathrm{Law}(X)$. □

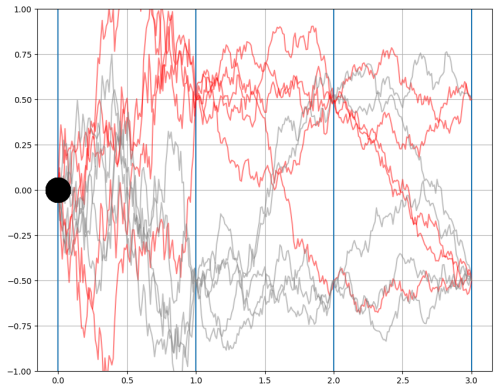
This shows that endpoint-matching alone does not identify the correct path law. In particular, unless the bridge volatility schedules are chosen to satisfy a specific time-rescaling compatibility condition, the chained-bridge construction leads to different path laws.

D Need for Path-Dependent Drift

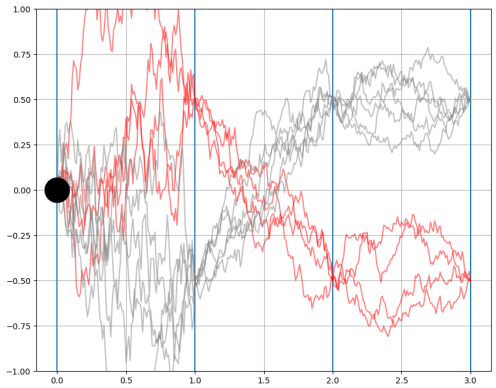


(a) Data

(b) Base Process



(c) Markov



(d) ABCD

Figure 5: We can consider the case where the data process is not Markov. This can happen for several reasons. For example, in a physical system, a snapshot will only reveal the position but not the velocity of the item. Even when the process restricted to the finite observation times is a discrete-time Markov process, we still cannot ensure that the continuous time extension is Markov. Indeed, we can find one that is not possible. This happens since continuous time extension while keeping the Markov structure will enforce some non-flexible temporal correlation. We can see such occurence in the example provided in (a). This is a mixture of two Markov processes: the red process and the blue process. The mixture itself is non-Markov. In the region around time $t = 1.5$, we will see that the drift for the red process tends to be negative, while that of the blue process tends to be positive. Still, we will see that the process restricted to the finite observations (which are at times $0, 1, 2$, and 3) is actually Markov: If $X_1 = 0.5$, then $X_2 = X_3 = -0.5$, and, if $X_1 = -0.5$, then $X_2 = X_3 = 0.5$. However, we have that X_1 and X_2 have a correlation of -1, while X_2 and X_3 have a correlation of 1. We then cannot find *any* continuous-time Markov SDE process such that the correlation structures can be satisfied (even when we allow the volatility schedule to be different from the data process). By trying to match the marginal of the observed data at each observation time, we will introduce positive correlation. We can see that the majority of the red process, which are the processes with $X_1 = 0.5$, will also have $X_2 = 0.5$ instead of -0.5. However, we will see that a similar problem also occurs from time $t = 2$ to 3. The value X_2 is supposed to be the same as X_3 under the data process. However, the volatility we add creates some mixing, thereby violating the joint finite-dimensional distribution matching desiderata. For our ABCD, we start from the base process, which is data-independent, but with matching volatility schedule. The path-dependent (on finite time observation) Doob-h transform helps change the base process (b) into our ABCD process (d).

E Algorithms

E.1 Training

See Algorithm 1. For simplicity, we denote it in the non-batched setting, but it is trivial to batch it, and our implementation is batched. We normalize the time horizon such that $t_L = 1$.

Algorithm 1 Training Procedure

Require: Time series distribution p_{data}, SDE parameters $\sigma(t), a(t)$, Waypoint time sampling distribution $p_{\text{waypoint_times}}$, Conditioning set size sampling distribution $p_{\text{waypoint_set_size}}$

1: Initialize $\mathbf{f}_\theta(x_t; t, t_{i^*+1}, \mathcal{K})$
2: **for** `iter` $= 1 \to$ `iter`$_{\text{max}}$ **do**
3: Sample $|\mathcal{T}_\mathcal{W}| \sim p_{\text{waypoint_set_size}}$ `// `$|\mathcal{T}_\mathcal{W}| \in \mathbb{Z}^+ \oplus \{0\}$
4: $\mathcal{T}_\mathcal{W} \leftarrow \{\tau_1, \ldots, \tau_{|\mathcal{T}_\mathcal{W}|}\} \sim p_{\text{waypoint_times}}$ `// ascending order,` $\tau_i \in (0, 1)$
5: $\mathcal{T}_\mathcal{W} \leftarrow \mathcal{T}_\mathcal{W} \oplus \{0, 1\}$ `// include initial condition and terminal state`
6: $t \sim \mathcal{U}[0, 1)$ `// current time (interpolates waypoints)`
7: $\mathcal{T}_\mathcal{P} \leftarrow \{\tau_i | \tau_i \le t \cap \tau_i \in \mathcal{T}_\mathcal{W}\}$ `// observe all past times (causal)`
8: $\mathcal{T}_\mathcal{K} \leftarrow \mathcal{T}_\mathcal{P} \cup \text{RandomSubset}(\mathcal{T}_\mathcal{W} \setminus \mathcal{T}_\mathcal{P})$ `// conditioning can include future`
9: $t_{i^*} \leftarrow \max\{\tau_i \in \mathcal{T}_\mathcal{K} | \tau_i \le t\}$ `// most recent waypoint time`
10: $t_{i^*+1} \leftarrow \min\{\tau_i \in \mathcal{T}_\mathcal{W} | \tau_i > t\}$ `// next waypoint time`
11: $x_0, x_{\tau_1}, \ldots, x_{\tau_{|\mathcal{T}_\mathcal{W}|}}, x_1 \sim p_{\text{data}}\left(x_0, x_{\tau_1}, \ldots, x_{\tau_{|\mathcal{T}_\mathcal{W}|}}, x_1\right)$ `// sample from joint data distribution in continuous time`
12: $\mathcal{K} \leftarrow \{(\tau, x_\tau) | \tau \in \mathcal{T}_\mathcal{K}\}$ `// waypoints to condition on (past and future)`
13: $x_t | x_{t_{i^*}}, x_{t_{i^*+1}} \overset{\mathbb{P}}{\sim} \mathcal{N}(\mu_{t|t_{i^*}, t_{i^*+1}}(x_{t_{i^*}}, x_{t_{i^*+1}}), \Sigma_{t|t_{i^*}, t_{i^*+1}})$ `// sample target, Eq 5`
14: $\nabla_{x_t} \log p_{\text{base}}(x_{t_{i^*+1}} | x_t) = \frac{\Phi(t, t_{i^*+1})}{C_t(t_{i^*+1}, t_{i^*+1})}(x_{t_{i^*+1}} - \Phi(t, t_{i^*+1})x_t)$ `// score target, Eq 22`
15: $\mathcal{L}(\theta) \leftarrow w(t, t_{i^*}, t_{i^*+1}) \|\mathbf{f}_\theta(x_t; t, t_{i^*+1}, \mathcal{K}) - \nabla_{x_t} \log p_{\text{base}}(x_{t_{i^*+1}} | x_t)\|^2$ `// DSM, Eq 8 (can be batched)`
16: $\theta \leftarrow \theta - \nabla_\theta \mathcal{L}(\theta)$ `// update (can be batched)`
17: **end for**
18: **return** $\mathbf{f}_\theta(x_t; t, t_{i^*+1}, \mathcal{K})$ `// trained score network`

E.2 Inference

See Algorithm 2. Normally, we would use a constant Δt discretization, but this may result in the discretization scheme overshooting the waypoints. As such, when we are near a waypoint, we decrease the timestep to prevent overshooting (Line 6; for numerical stability, we set $\epsilon_t = 10^{-6}$). To ensure that we still maintain the same total number of steps, we add back that difference to the next timestep after crossing the waypoint. (This strategy relies on the distance between waypoints being at least twice as large as the discretization interval, which is generally true in most reasonable SDE simulations.)

This inference procedure applies to both our ABCD model (Equation 6), and the autoregressively chained diffusion bridge construction from Theorem 3 (Equations 17, 16). For noise-to-data generation, reset $x_t \sim \mathcal{N}(0, I)$ after Line 19, and let $x_0 \sim \mathcal{N}(0, I)$.

Algorithm 2 Inference Procedure

Require: Trained score network $\mathbf{f}_\theta(x_t; t, t_{i^*+1}, \mathcal{K})$, Waypoint times $\mathcal{T}_\mathcal{W}$, Observed waypoints $\mathcal{K}$ ($\mathcal{T}_\mathcal{K} \subseteq \mathcal{T}_\mathcal{W}$, includes $(0, x_0)$ and possibly future conditioning), Number of steps N, Waypoint time exceedance threshold ϵ_t, SDE parameters $\sigma(t), a(t)$

$t \leftarrow 0$
$\Delta t_{\text{leftover}} \leftarrow 0$
while $t < 1$ **do**
  $t_{i^*} \leftarrow \max\{\tau_i \in \mathcal{T}_\mathcal{K} | \tau_i \leq t\}$ `// most recent waypoint time`
  $t_{i^*+1} \leftarrow \min\{\tau_i \in \mathcal{T}_\mathcal{W} | \tau_i > t\}$ `// next waypoint time`
  if $t + \frac{1}{N} > t_{i^*+1} + \epsilon_t$ **then**
    $\Delta t \leftarrow t_{i^*+1} - t + \epsilon_t$ `// Don't overshoot the waypoint`
    $\Delta t_{\text{leftover}} \leftarrow 1/N - \Delta t$ `// Give this back later`
  else
    $\Delta t \leftarrow 1/N + \Delta t_{\text{leftover}}$ `// Recover the leftover simulation time`
    $\Delta t_{\text{leftover}} \leftarrow 0$
  end if
  $dB_t^\mathbb{Q} \sim \mathcal{N}(\mu = \mathbf{0}, \Sigma = (\Delta t)I)$
  $dB_t^\mathbb{P} \leftarrow dB_t^\mathbb{Q} + \sigma(t)\mathbf{f}_\theta(x_t; t, t_{i^*+1}, \mathcal{K})\Delta t$ `// Eq 35`
  $dx_t \leftarrow -a(t)x_t\Delta t + \sigma(t)dB_t^\mathbb{P}$ `// Eq 18`
  $x_{t+\Delta t} \leftarrow x_t + dx_t$
  for $\tau_i \in \mathcal{T}_\mathcal{W} \setminus \mathcal{T}_\mathcal{K}$ **do**
    if $\tau_i \leq t < \tau_i + \frac{1}{N}$ **then**
      $\mathcal{K} \leftarrow \mathcal{K} \oplus (t, x_t)$ `// we've observed another waypoint,` $\tau_i \approx t$
    end if
  end for
end while
return $\mathcal{K}$ `// trajectory at waypoints`

F Comparison to Denoising Diffusion Bridge Models

Since our loss function and generative SDE bear some resemblance to the popular denoising diffusion bridge [75] and denoising diffusion [56] models, we show the similarities and differences. We conduct the analysis on denoising diffusion bridge models (DDBM) [75], as denoising diffusion models [56] are just a special case of DDBMs with $x_0 \sim \mathcal{N}(0, I)$.

F.1 Similarity: Two-Point Case

Let us consider the case where we only wish to model the marginal: $p_{\text{data}}(x_1|x_0)$; that is, a two-point time series where $t_L = 1$. Then (omitting the t_{i^*+1} argument, since it is always 1), our ABCD loss function (Equation 8) and dynamics (Equation 6) become:

$$\theta^* = \underset{\theta}{\text{argmin}}\, \mathbb{E}_{t\sim\mathcal{U}[0,1)}\mathbb{E}_{\substack{p_{\text{data}}(x_0,\dots,x_1)\\ p_{\text{base}}(x_t|x_0,x_1)}}\left[\|\mathbf{f}_\theta(t, x_0, x_t) - \nabla_{x_t}\log p_{\text{base}}(x_1|x_t)\|_2^2\right], \tag{54}$$

$$dx_t = \left(-a(t)x_t + \sigma(t)^2\mathbf{f}_\theta(t, x_0, x_t)\right)dt + \sigma(t)dB_t \tag{55}$$

Let us also recall the DDBM equations (Equations 1, 2):

$$dx_t = \left(-a(t)x_t + \sigma(t)^2\left[\mathbf{s}_\theta(t, x_t, x_0) - \nabla_{x_t}\log p(x_0|x_t)\right]\right)dt + \sigma(t)dB(t), x_0 \sim p_{\text{data}}(x_0)$$

$$\theta^* = \underset{\theta}{\text{argmin}}\, \mathbb{E}_{t\sim\mathcal{U}[0,1)} \underset{\substack{x_0,x_1\sim p_{\text{data}}\\ x_t\sim p_{\text{base}}(x_t|x_0,x_1)}}{\mathbb{E}} \left[w(t)\|\mathbf{s}_\theta(t, x_t, x_0) - \nabla_{x_t}\log p_{\text{base}}(x_t|x_0, x_1)\|^2\right]$$

In that case, the loss functions and dynamics are very similar between ABCD and diffusion bridge models. The losses both have the Gaussian noising kernel $p_{\text{base}}(x_t|x_0, x_1)$. The only difference is that diffusion bridge models have (1) doubly-conditioned score target $\nabla_{x_t}\log p_{\text{base}}(x_t|x_0, x_1)$ and (2) subtracted drift adjustment $\nabla_{x_t}\log p_{\text{base}}(x_0|x_t)$ from the score. But, if we make the reparameterization to the diffusion bridge score network:

$$\mathbf{s}_\theta(t, x_t, x_0) = \mathbf{b}_\theta(t, x_t, x_0) + \nabla_{x_t}\log p_{\text{base}}(x_0|x_t) \tag{56}$$

and apply Bayes' rule (and the Markov property):

$$\nabla_{x_t}\log p_{\text{base}}(x_t|x_0, x_1) = \nabla_{x_t}\left[\log p_{\text{base}}(x_t|x_1) + \log p_{\text{base}}(x_0|x_t, x_1) - \log p_{\text{base}}(x_0|x_1)\right] \tag{57}$$

$$= \nabla_{x_t}\log p_{\text{base}}(x_t|x_1) + \nabla_{x_t}\log p_{\text{base}}(x_0|x_t) \tag{58}$$

the loss (Eq. 2, $w(t)$ omitted) and dynamics of diffusion bridge models (Eq. 1) reduce to:

$$\theta^* = \underset{\theta}{\text{argmin}}\, \mathbb{E}_{t\sim\mathcal{U}[0,1)}\mathbb{E}_{x_0,x_1\sim p_{\text{data}}}\mathbb{E}_{x_t\sim p_{\text{base}}(x_t|x_0,x_1)}[\|\mathbf{b}_\theta(t, x_t, x_0) - \nabla_{x_t}\log p_{\text{base}}(x_t|x_1)\|^2], \tag{59}$$

$$dx_t = \left(-a(t)x_t + \sigma(t)^2\mathbf{b}_\theta(t, x_t, x_0)\right)dt + \sigma(t)dB_t \tag{60}$$

The dynamics match (Equations 55, 60), with appropriate choice of $\mathbf{b}_\theta, \mathbf{f}_\theta$. Now, the only difference is that the loss target is $\nabla_{x_t}\log p_{\text{base}}(x_1|x_t)$ in ABCD (Equation 54), and $\nabla_{x_t}\log p_{\text{base}}(x_t|x_1)$ in diffusion bridge models (Equation 59). Intuitively, both of these targets pull the intermediate x_t to be more "aligned" with the data x_1, although they represent slightly different things. ABCD's $\nabla_{x_t}\log p_{\text{base}}(x_1|x_t)$ can be interpreted as pulling an initially unconditioned diffusion towards certain waypoints, while diffusion bridges' (and denoising diffusion models') $\nabla_{x_t}\log p_{\text{base}}(x_t|x_1)$ can be interpreted as denoising the forward data-to-noise corruption. This comes from the fact that we derive ABCD with Girsanov's Theorem and change-of-measure [20, 17] in only one time direction (Section A.3), while the other methods derived their method via Anderson's time-reversal [3, 75, 56].

F.2 Difference: Path-Dependent Conditioning

Another key difference is that, in general, our ABCD has a path-dependent score network $\mathbf{f}_\theta(t, t_{i^*+1}, x_t; \mathbf{x}_{0:i^*})$ (where $\mathbf{x}_{0:i^*}$ is a concatenation of states as defined in Equation 4), as opposed to the diffusion bridge score network that depends only on the current state and the start point: $\mathbf{s}_\theta(t, x_t, x_0)$. Conceptually, this extends the framework from Markovian to *non-Markovian* autoregressive modeling in continuous time, which does not exist in prior work to our knowledge.

This suggests that DDBM will fail to capture the correlation structure, for example between x_{t_1} and x_{t_2}, while our approach can do so.

F.3 Difference: Time-Adaptive Volatility and Noising Kernels

Furthermore, even if we were to condition the diffusion bridge model's score network on the physical time, it still would not have the physical time-adaptive volatility that ABCD has. The implication of this is that the dynamics of the generation process are fundamentally inconsistent with the underlying real-world dynamics. We showed this in Theorem 3 and Section C.

F.4 Difference: Any-Subset Autoregression

Finally, the prior work [75, 56] does not consider any-subset autoregression. They only have the ability to bridge between two endpoints. In contrast, our framework allows us to model arbitrarily many finite-dimensional marginals of the SDE's path measure. In practical terms, what this means is that our method is naturally suited for handling time-series, especially those that are irregularly sampled or have future conditioning, e.g., infilling corrupted video files.

G Comparison to Probabilistic Forecasting with Interpolants

The stochastic interpolant framework is closely related to diffusion bridges and denoising diffusions [1, 10]. PFI (probabilistic forecasting with interpolants) uses state-to-state stochastic interpolants to generate time series data. We review some similarities and key differences.

G.1 Difference: Discrete-Time Modeling

PFI's stated aim is to model the discrete-time, continuous-space process $\{x_{kT}\}_{k\in\mathbb{Z}}$, where T is the sampling interval and k is the discrete time index. This already differs from our setting of *continuous-time* autoregression.

G.2 Connection: Autoregressive Diffusion Bridge Construction

PFI generates samples with the auxiliary non-physical SDE (using similar notation to Equation 17):

$$d\tilde{X}_s^{(k)} = \mathbf{b}_\theta^g\left(s, x_{kT}, \tilde{X}_s^{(k)}\right) ds + g(s) dB_s, \tilde{X}_0^{(k)} = x_{kT}, \tag{61}$$

where k is the discrete index of the state, $s \in [0,1]$ is a fake time variable they construct to interpolate between consecutive points in the time series, $\tilde{X}_s^{(k)}$ is the value of the auxiliary "bridge" SDE, and $\tilde{X}_1^{(k)} \sim p\left(x_{(k+1)T}|x_{kT}\right)$. The superscript ^{g} on $\mathbf{b}_\theta^g\left(s, x_{kT}, \tilde{X}_s^{(k)}\right)$ indicates that the drift is dependent on the volatility schedule.

Indeed, this is highly similar to the autoregressive diffusion bridge construction in Equation 17, if we set $\tilde{A} = 0$ and assume regular time sampling intervals:

$$d\tilde{X}_\tau^{(i)} = \tilde{\sigma}(\tau)^2 \tilde{\mathbf{f}}_\theta\left(\tau, \mathbf{x}_{0:i-1}, \tilde{X}_\tau^{(i)}\right) d\tau + \tilde{\sigma}(\tau) dB_\tau^{(i)}, \tilde{X}_0^{(i)} = x_{t_{i-1}} \tag{62}$$

Note the correspondence of variables: $i \leftrightarrow k, \tau \leftrightarrow s, x_{t_{i-1}} \leftrightarrow x_{kT}$. As such, to equate Equations 62 and 61, we just need:

$$\mathbf{x}_{0:i-1} = x_{t_{i-1}} \quad \texttt{//In Equation 62} \tag{63}$$

$$g(s) = \tilde{\sigma}(\tau) \quad \texttt{//match Eq 61's \& 62's volatilities} \tag{64}$$

$$\mathbf{b}_\theta^g\left(s, x_{kT}, \tilde{X}_s^{(k)}\right) = \tilde{\sigma}(\tau)^2 \tilde{\mathbf{f}}_\theta\left(\tau, \mathbf{x}_{0:i-1}, \tilde{X}_\tau^{(i)}\right) \quad \texttt{//match Eq 61's \& 62's drifts} \tag{65}$$

Whether the equality in Equation 65 is achieved depends on what PFI's trained network learned (which depends on the choice of interpolant schedule). In principle, however, both learned drifts have the same high-level intuition: they "pull towards" the data at the next waypoint time.

G.3 Difference to Connection: Constructing ABCD-Style Time-Adaptive Volatility

Considering the correspondence between Equations 61 and 62, PFI again runs into the same problem that autoregressively chained diffusion bridges do: the volatility doesn't naturally scale with physical time elapsed (Theorem 3). So, in the naive application, this leads to incorrect dynamics.

It's worth noting that via the Fokker-Planck equation [52], they do provide a mechanism for tuning the diffusion coefficient $g(s)$ and adjusting the drift accordingly, such that the single-time marginal laws of the resultant SDE are still the same as the single-time marginal laws of the data-generating SDE they had originally trained. This "trades diffusion for transport" [10].

In principle, we could tune this diffusion coefficient to match ABCD's inductive bias that volatility should scale with physical time: *e.g.*, $g(s) = \sqrt{T} * g_{\text{const}}$ if the model was originally trained with constant volatility. They do not consider this in their work (likely because they used regularly sampled time intervals, which is another limitation of their work).

G.4 Difference: Markovianization

Their experiments make the assumption that the transition kernel of the time series is physical time-invariant and Markov, which is not true in many important cases, *e.g.*, stock data driven by fractional

Brownian motion. As a consequence, they do not condition their models on the physical time variable, nor comprehensive past history (see Equation 61). The lack of conditioning on the physical time variable becomes problematic when we have a non-Markov and/or time-varying process, where we need to know the time relations between observed keypoints. For instance, in their video generation experiments, they randomly conditioned their model on a time-stamped previous frame as well as the most recent frame but *without* the timestamp, making it very difficult to ascertain where the previous history frame was in relation to the most recent frame, thereby hindering the estimation of the dynamics.

G.5 Difference: Any-Subset Modeling

They do not consider the any-subset inference case. In particular, they only deal with *causal* conditioning on a maximum of three states (previous waypoint, current auxiliary generative state, and sometimes one waypoint from the history, making for $O(N)$ conditions), on a regularly spaced physical time grid. They can also only infer the next waypoint in the regular discretized time grid. In contrast, our method is explicitly designed as an any-subset autoregressive model that can condition on any subset of observations in the time series, even those in the *future*. In the case of regularly spaced grids, we would have $O(2^N)$ capacity, versus their $O(N)$ capacity, but we also allow for irregularly spaced physical time grids. Our inference is also not limited to the immediate next waypoint on a fixed time grid, but is flexible to infer future states at arbitrary timestamps.

H Additional Results

H.1 Sensitivity of ABCD to Volatility Hyperparameter

Table 5 shows what happens when we change ABCD's volatility hyperparameter σ. Although we did not have the computational resources to retrain ABCD with a large hyperparameter sweep (each run takes 36 hours on 4 A100 80GB GPUs), we found that there seems to be a tolerable range of hyperparameters, as the alternate choices we tested did not significantly degrade the results.

Additionally, on the strictly causal training runs for CelebV-HQ (Table 22, 23), we had the chance to test a few more hyperparameter choices of ABCD $\sigma \in \{0.25, 0.5, 0.75\}$. We found that overall, *any* of these choices for ABCD beat the best of the conditional diffusion bridge methods (Table 24).

Table 5: **Sensitivity of ABCD to Volatility:** Infill videos of 32 frames, prompt every 8 frames, sample with 250 SDE steps. No Brownian Bridge.

Setting	*FVD* (↓)	*FID* (↓)
CelebV-HQ		
$\sigma = 0.4$	99.7	2.6
$\sigma = 0.5$	111.5	2.9
Sky-Timelapse		
$\sigma = 0.3$	183.5	4.8
$\sigma = 0.4$	162.3	4.4

H.2 Ablation on Conditioning

Towards answering RQ1, we conduct ablations on how much conditioning is necessary for the model to achieve high-quality generations, *i.e.*, how much we get from a path-dependent formulation. For simplicity, we test on causal rollout. We test with a few conditioning options:

1. *Full autoregressive conditioning* on all the waypoints elapsed in the path.
2. *Monte Carlo estimate* (single waypoint, uniformly sampled) of the intermediate path history, along with times 0 and t_{i^*} (that is, initial condition and most recent waypoint observation), making for a total of three conditioning points. This is *similar to PFI's [10] conditioning strategy*, except that they omitted the initial condition.
3. *Near-Markov* conditioning, where we only use the initial condition from time 0 and the most recent waypoint at time t_{i^*} as conditioning. This is *similar to LFM's [30] Markovian conditioning strategy*, except that they omitted the initial condition.
4. *Prefix only* conditioning, where we only condition on the prompt frames from the prefix that are before time t.

See Tables 6, 7, 8, 9. We can draw a few conclusions from this experiment:

- Our *path-dependent* conditioning provides the best results on a majority of scenarios: 15 out of 24 tests on CelebV-HQ, and on 18 out of 24 tests on Sky-Timelapse. This confirms that *non-Markovian dependence in the drift is beneficial for generative modeling of time series, underscoring the benefit of our modeling approach over that of previous works (PFI [10], LFM [30])*.
- To give the other approaches credit, having less conditioning can still be competitive (except for prefix-only). This actually supports RQ1, as it means that our model effectively learned to extract information across a variety of subsets of the path history. This also means it could be possible to mostly maintain quality while reducing computational resources used; the optimal settings for this can be a subject of exploration in future work.

Table 6: Conditioning ablation, FVD (lower is better). Best per column in **bold**. CelebV-HQ dataset. Model is causally trained ABCD with $\sigma = 0.5$ and no Brownian Bridge (see Section H.11).

	32 frames						16 frames					
	prefix_length_1		prefix_length_4		prefix_length_8		prefix_length_1		prefix_length_4		prefix_length_8	
Conditioning	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500
Full AR Cond $(0, \ldots, t_{i^*})$	**303.93**	**317.08**	350.54	248.25	281.24	199.37	**281.84**	278.45	**209.82**	**174.93**	**117.75**	**96.14**
Monte Carlo $(0, t_{\text{rand}}, t_{i^*})$	332.86	328.25	**310.63**	**217.15**	**224.36**	**160.10**	297.76	275.99	241.55	201.88	142.10	117.08
Near-Markov $(0, t_{i^*})$	364.95	329.51	344.95	242.50	260.68	185.04	311.54	**270.01**	289.44	226.39	163.98	128.18
Prefix Only $(0, \ldots, t_{\text{pre len}})$	448.23	472.08	426.77	447.01	343.07	374.03	456.68	461.18	353.44	374.38	244.85	262.89

Table 7: Conditioning ablation, FID (lower is better). Best per column in **bold**. CelebV-HQ dataset. Model is causally trained ABCD with $\sigma = 0.5$ and no Brownian Bridge (see Section H.11).

	32 frames						16 frames					
	prefix_length_1		prefix_length_4		prefix_length_8		prefix_length_1		prefix_length_4		prefix_length_8	
Conditioning	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500
Full AR Cond $(0, \ldots, t_{i^*})$	**15.30**	**7.05**	22.65	13.74	14.74	9.34	**9.41**	**6.27**	**10.93**	**7.59**	**3.82**	**2.73**
Monte Carlo $(0, t_{\text{rand}}, t_{i^*})$	16.52	7.94	**22.60**	13.25	**12.92**	**7.51**	10.29	6.72	12.77	8.65	4.70	3.01
Near-Markov $(0, t_{i^*})$	16.52	8.31	22.86	**13.23**	14.17	7.94	10.94	7.17	14.62	9.72	5.14	3.16
Prefix Only $(0, \ldots, t_{\text{pre len}})$	33.05	43.07	55.24	61.59	35.89	41.35	36.04	41.57	33.64	37.94	12.47	13.82

Table 8: Conditioning ablation, FVD (lower is better). Best per column in **bold**. Sky Timelapse dataset. ABCD $\sigma = 0.4$ without Brownian Bridge.

	32 frames						16 frames					
	prefix_length_1		prefix_length_4		prefix_length_8		prefix_length_1		prefix_length_4		prefix_length_8	
Conditioning	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500
Full AR Cond $(0, \ldots, t_{i^*})$	**369.06**	**285.94**	**357.69**	**276.78**	**297.03**	**229.90**	**242.40**	**207.81**	**186.79**	**164.61**	**119.81**	**106.15**
Monte Carlo $(0, t_{\text{rand}}, t_{i^*})$	377.28	302.48	391.86	323.46	333.51	275.35	275.88	228.71	254.09	213.07	181.17	150.86
Near-Markov $(0, t_{i^*})$	405.34	332.55	421.13	357.90	368.54	313.05	302.90	258.34	300.64	268.23	219.64	191.57
Prefix Only $(0, \ldots, t_{\text{pre len}})$	928.86	923.94	871.58	874.81	730.85	738.43	994.55	991.96	733.17	734.32	467.81	468.97

Table 9: Conditioning ablation, FID (lower is better). Best per column in **bold**. Sky Timelapse dataset. ABCD $\sigma = 0.4$ without Brownian Bridge.

	32 frames						16 frames					
	prefix_length_1		prefix_length_4		prefix_length_8		prefix_length_1		prefix_length_4		prefix_length_8	
Conditioning	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500	step_250	step_500
Full AR Cond $(0, \ldots, t_{i^*})$	19.74	14.22	**16.84**	12.29	**11.62**	8.13	10.10	8.60	**6.65**	**5.45**	**2.55**	**2.17**
Monte Carlo $(0, t_{\text{rand}}, t_{i^*})$	**18.31**	**12.80**	17.27	**11.55**	12.17	**7.95**	**10.04**	**8.45**	7.02	5.66	2.84	2.35
Near-Markov $(0, t_{i^*})$	19.08	13.31	18.92	13.14	13.87	9.19	10.13	8.49	7.41	5.86	3.11	2.50
Prefix Only $(0, \ldots, t_{\text{pre len}})$	253.48	256.25	225.76	227.64	177.73	180.75	215.40	216.65	148.42	150.49	71.14	73.63

H.3 Exploring Impact of Scaling

We also investigate whether increasing computational resources can improve model performance. We launched another training run on CelebV-HQ, this time with a 594,067,472 parameter model (depth=20, hidden_size=1024, num_heads=16). We increased the number of scheduled cosine decay steps to 380K, with 20K warmup steps; and increased the total number of training steps to 320K. We used 8 A100 80GB GPUs for this.

See Tables 11, 10. The experiments are done with the infilling setting. We see that on FVD (Table 10), increasing parameter count and training time boosts the performance, especially as the prompting gets sparser. With respect to FID (Table 11), while the model with the most resources is not always the best, we note that all the FIDs are low to begin with, and the differences between FIDs are much smaller than those between FVDs.

Table 10: Model-scaling ablation, FVD (lower is better) on CelebV-HQ. We fix the video length to 32 frames and use 250 diffusion steps. Best per column in **bold**.

Model	pin_every_4	pin_every_8	pin_every_16
Base (202M params, 240K steps)	37.75	99.69	315.49
Medium (594M params, 240K steps)	36.40	94.17	268.67
Medium (594M params, 320K steps)	**35.02**	**88.47**	**238.77**

Table 11: Model-scaling ablation, FID (lower is better) on CelebV-HQ. We fix the video length to 32 frames and use 250 diffusion steps. Best per column in **bold**.

Model	pin_every_4	pin_every_8	pin_every_16
Base (202M params, 240K steps)	1.29	**2.62**	**5.91**
Medium (594M params, 240K steps)	1.26	2.73	6.12
Medium (594M params, 320K steps)	**1.21**	2.67	6.51

H.4 Time Super-Resolution

Given the continuous-time formulation, we are also interested to see whether ABCD can generalize to times not seen during training. As such, we trained ABCD on Sky Timelapse with the same settings ($\sigma = 0.4$) as before, except that we eliminate roughly half of the time grid from the training data. Specifically, we excluded every odd-indexed frame (under 0-based indexing) except the final frame from the possible predicted or conditioning times during training. (So, out of 64 frames, the model was only allowed to see some subset of frames $0, 2, 4, \ldots, 62, 63$.) Then, we evaluated it to generate the full time-resolution data, including those parts of the time grid that were not seen during training (that is, all 64 frames).

We then compare against an ABCD model that was trained on the full-grid data. The experiment shows that if we include the Brownian bridge inductive bias from (13) to the score network, it can generalize fairly well to times that were not seen during training. Out of all the compared methods, it has the best frame-wise FID. On FVD (Pin Every 16), it even beat the network that was trained on the whole time grid albeit without the Brownian bridge inductive bias. However, a network trained on the partial time grid without having a Brownian bridge inductive bias generally fails when predicting times not seen during training. The best-performing network in terms of FVD is that which has the full-grid training, along with the Brownian bridge inductive bias. See Table 12.

	Pin Every 8		Pin Every 16	
Model	FVD	FID	FVD	FID
Full-Grid Train, No BB	107.35	2.94	323.65	7.68
Full-Grid Train, W/ BB	77.17	2.45	244.83	5.20
Half-Grid Train, No BB	875.72	97.06	1003.98	121.58
Half-Grid Train, W/ BB	155.27	1.96	300.27	3.33

Table 12: Time super-resolution. Videos of 64 frames total, 500 inference steps. All models have $\sigma = 0.4$. Sky-Timelapse dataset.

H.5 Satisfaction of Future Constraints

We conduct an experiment on Sky-Timelapse where we generate 16-frame videos, having future knowledge of the 4th, 8th, 12th, and 16th frames. Over 1360 generated videos, we measure the adherence of our model to these ground truth frames. Just via training the model with our loss function and without any inductive architectural bias, we achieve near-perfect adherence to the conditioning, as measured by SSIM (scale of 0-1, 1 is exact reconstruction) and PSNR (values above 40 are considered to be near-perfect reconstructions). See Table 13.

Metric	SSIM (↑)	PSNR (↑)
Value	0.995	45.6

Table 13: **Satisfaction of Future Constraints.** We test how well the model lands on the ground truth future conditioning when following its learned score function.

H.6 Irregular Time Sampling

We want to assess how ABCD compares against diffusion bridge baselines in sampling with irregular physical time intervals. We expect ABCD to prevail, as the diffusion bridge baselines do not adapt their volatility to the time elapsed.

We set $\sigma = 0.4$ for both ABCD (no Brownian bridge) and the AR conditional diffusion bridge baseline. We tested causal generation starting from a single frame on 1360 videos from the Sky-Timelapse dataset, and use 250 SDE solver steps. Out of 64 ground-truth frames provided in the Sky-Timelapse dataset samples, we had the models generate the irregularly-spaced frames (corresponding to the repeating spacing pattern of 1-1-2-4-8): $1, 2, 4, 8, 16, 17, 18, 20, 24, 32, 33, 34, 36, 40, 48, 49, 50, 52, 56$. ABCD is clearly the best, as seen in Table 14.

H.6.1 Adaptive Inference-Time Volatility Scaling in Diffusion Bridge Baseline

Theorem 3 also suggests that we could select the volatility of an autoregressive conditional diffusion bridge baseline to match ABCD's, via the scaling

$$\tilde{\sigma}_{\text{new}}(\tau) = \sqrt{t_{i^*+1} - t_{i^*}} * \sigma_{\text{ABCD}}(t_{i^*} + (t_{i^*+1} - t_{i^*})\tau) \tag{66}$$

If we adopted this construction at *train-time* and inference-time, then it would indeed be equivalent to ABCD. However, adopting it *only at inference time would give incorrect results*. This is because the training target scores (Lemma 1) are carefully constructed to fulfill the prescribed volatility schedule. As such, changing the volatility schedule at inference time would lead to evaluation on a completely different implied change-of-measure than it was trained for. Intuitively, if the score network was trained on interpolating paths with very low volatility, it should not be expected to give the correct score on interpolating paths with very high volatility that it had never seen.

To empirically illustrate the folly of inference-time volatility schedule changes, we add an additional baseline to this experiment, where the volatility is scaled *at inference-time only* according to Equation 66. See Table 14.

Metric	ABCD	AR Cond: σ	AR Cond: $\sqrt{\Delta t} * \sigma$
FVD (↓)	**301.3**	465.0	658.1
FID (↓)	**12.2**	75.8	46.1

Table 14: **Irregular Time Sampling.** "AR Cond: σ" refers to the autoregressive conditional diffusion bridge baseline from Theorem 3 *without* time-adaptive volatility. "AR Cond: $\sqrt{\Delta t} * \sigma$" refers to a version of that baseline which, *at inference time only*, has its volatility coefficient scaled as $\sqrt{t_{i^*+1} - t_{i^*}}$ according to the gap in physical time over an interval, such that its volatility coefficient matches ABCD's.

H.7 Diversity Analysis

We want to assess how diverse ABCD's completions for a given video prompt are.

Over 1360 Sky-Timelapse videos, we gave ABCD 4 frames from each video, then had it complete the next 12 frames. To measure diversity, we had the model generate 4 different completions per video, and calculated the frame-wise LPIPS [73] between each of the 6 pairs of different video completions. LPIPS of 0 indicates deterministic completions.

Quantitatively, the completions are diverse: as time went further from the conditioning frames (0-3), diversity increased, as shown in Table 15. Qualitatively, we confirmed by manual inspection that the model produces diverse plausible completions (*e.g.*, cloud paths vary).

Frame Index	4	8	12
LPIPS	0.056 ± 0.026	0.096 ± 0.041	0.132 ± 0.064

Table 15: **LPIPS Scores by Frame:** Out of 16 total frames, frames 0-3 are the prompts. Reporting mean and std LPIPS.

H.8 CelebV-HQ: Additional Non-Causal Results

See Table 16 for FID metrics and Table 17 for overall rankings. Table 17 is calculated by first calculating the average rank in each of Tables 16 and 2 (that is, getting a summary rank for FID and FVD), then averaging (and getting the standard deviation of) the two summary metric ranks. The rank is 1-based, where lower is better.

Table 16: CelebV-HQ — Non-Causal — FID (↓)

		Num Frames = 32						Num Frames = 16					
		pin_every = 4		pin_every = 8		pin_every = 16		pin_every = 4		pin_every = 8		pin_every = 16	
Method	Steps	250	500	250	500	250	500	250	500	250	500	250	500
ABCD													
W/ BB: σ=0.5		***1.3*** $\pm$ 0.0	***1.0*** $\pm$0.0	***3.4*** $\pm$ 0.0	***2.4*** $\pm$ 0.0	12.5 $\pm$ 0.0	8.3 $\pm$ 0.0	***2.7*** $\pm$ 0.0	***2.2*** $\pm$ 0.0	6.7 $\pm$ 0.0	4.8 $\pm$ 0.0	15.9 $\pm$ 0.0	11.3 $\pm$ 0.1
No BB: σ=0.5		**1.3** $\pm$ 0.0	**0.9** $\pm$ 0.0	**2.9** $\pm$ 0.0	**2.2** $\pm$ 0.0	***8.8*** $\pm$ 0.1	5.8 $\pm$ 0.0	**2.6** $\pm$ 0.0	**2.1** $\pm$ 0.0	5.5 $\pm$ 0.0	4.1 $\pm$ 0.1	9.5 $\pm$0.0	6.8 $\pm$ 0.1
Conditional Diffusion Bridge													
$\sigma = 0.5$		24.2 $\pm$ 0.0	10.0 $\pm$ 0.0	55.0 $\pm$ 0.0	24.2 $\pm$ 0.1	148.8 $\pm$ 0.2	70.9 $\pm$ 0.2	12.5 $\pm$ 0.0	5.5 $\pm$ 0.0	28.0 $\pm$ 0.0	13.7 $\pm$ 0.1	68.3 $\pm$ 0.1	33.6 $\pm$ 0.1
$\sigma = 0.125$		1.7 $\pm$ 0.0	1.2 $\pm$ 0.0	4.2 $\pm$ 0.0	2.6 $\pm$ 0.0	9.4 $\pm$ 0.1	***5.3*** $\pm$ 0.1	2.9 $\pm$ 0.0	2.3 $\pm$ 0.0	***5.2*** $\pm$ 0.1	***3.5*** $\pm$ 0.0	***6.9*** $\pm$ 0.1	***5.0*** $\pm$ 0.0
$\sigma = 0.09$		1.5 $\pm$ 0.0	1.1 $\pm$ 0.0	3.5 $\pm$ 0.0	2.6 $\pm$ 0.0	**7.4** $\pm$ 0.0	**4.1** $\pm$ 0.0	3.0 $\pm$ 0.0	2.4 $\pm$ 0.0	**4.9** $\pm$ 0.0	**3.4** $\pm$ 0.0	**4.9** $\pm$ 0.0	**4.0** $\pm$ 0.0
Noise-to-Data Diffusion													
Cos: (3.0, 0.04)		110.9 $\pm$ 0.1	18.1 $\pm$ 0.1	185.8 $\pm$ 0.0	65.1 $\pm$ 0.2	240.1 $\pm$ 0.2	135.8 $\pm$ 0.4	25.6 $\pm$ 0.1	3.7 $\pm$ 0.0	82.1 $\pm$ 0.2	16.7 $\pm$ 0.0	132.7 $\pm$ 0.3	42.8 $\pm$ 0.3
Cos: (5.0, 0.05)		154.3 $\pm$ 0.1	42.2 $\pm$ 0.1	236.4 $\pm$ 0.3	114.2 $\pm$0.2	285.9 $\pm$ 0.1	196.5 $\pm$ 0.2	51.2 $\pm$ 0.0	6.1 $\pm$ 0.0	126.8 $\pm$ 0.1	29.4 $\pm$ 0.1	190.6 $\pm$ 0.1	75.7 $\pm$ 0.4
Exp: (4.0, 2.5)		125.5 $\pm$ 0.1	33.1 $\pm$ 0.0	188.8 $\pm$ 0.1	67.9 $\pm$ 0.1	238.4 $\pm$ 0.2	117.2 $\pm$ 0.1	32.6 $\pm$ 0.1	3.0 $\pm$ 0.0	74.3 $\pm$ 0.1	9.4 $\pm$ 0.0	101.3 $\pm$ 0.2	15.9 $\pm$ 0.0
Exp: (5.0, 5.0)		190.3 $\pm$ 0.0	116.1 $\pm$ 0.1	253.1 $\pm$ 0.1	190.0 $\pm$0.0	301.8 $\pm$ 0.0	226.1 $\pm$ 0.1	116.3 $\pm$ 0.1	9.1 $\pm$ 0.0	183.7 $\pm$ 0.2	33.1 $\pm$ 0.1	215.6 $\pm$ 0.1	63.8 $\pm$ 0.4

Table 17: CelebV-HQ (Non-Causal) — Overall Average Rank (↓)

Method	Avg Rank	Rank
ABCD		
W/ BB: σ=0.5	***2.6*** $\pm$ 0.4	2
No BB: σ=0.5	**1.9** $\pm$ 0.0	1
Conditional Diffusion Bridge		
$\sigma = 0.5$	5.2 $\pm$ 0.1	5
$\sigma = 0.125$	2.9 $\pm$ 0.0	4
$\sigma = 0.09$	2.8 $\pm$ 0.6	3
Noise-to-Data Diffusion		
Cos: (3.0, 0.04)	6.6 $\pm$ 0.1	7
Cos: (5.0, 0.05)	8.0 $\pm$ 0.1	8
Exp: (4.0, 2.5)	6.0 $\pm$ 0.1	6
Exp: (5.0, 5.0)	8.9 $\pm$ 0.0	9

H.9 Sky-Timelapse: Additional Non-Causal Results

See Table 18 for FID. See Table 19 for overall ranking across FVD and FID in the non-causal setting for Sky-Timelapse. Table 19 was calculated analogously to Table 17; summarizing the winner across both FID and FVD.

Table 18: Sky-timelapse — Non-Causal — FID (↓)

		Num Frames = 32						Num Frames = 16					
		pin_every = 4		pin_every =8		pin_every = 16		pin_every = 4		pin_every = 8		pin_every = 16	
Method	Steps	250	500	250	500	250	500	250	500	250	500	250	500
ABCD													
No BB: σ=0.4		1.8 ± 0.0	1.1 ± 0.0	4.3 ± 0.0	2.9 ± 0.0	8.8 ± 0.0	6.6 ± 0.0	***2.1*** ± 0.0	1.7 ± 0.0	4.6 ± 0.0	3.9 ± 0.0	8.8 ±0.1	7.4 ± 0.0
W/ BB: σ=0.4		***1.7*** ± 0.0	***1.1*** ± 0.0	***3.9*** ± 0.0	***2.6*** ± 0.0	***7.4*** ± 0.0	***5.4*** ± 0.0	**2.0** ± 0.0	**1.7** ± 0.0	**4.3** ± 0.0	**3.6** ± 0.0	***8.2*** ±0.1	6.9 ± 0.0
Conditional Diffusion Bridge													
$\sigma = 0.071$		2.2 ± 0.0	1.4 ± 0.0	5.2 ± 0.0	3.2 ± 0.0	9.0 ± 0.0	6.2 ± 0.0	3.1 ± 0.0	2.1 ± 0.0	5.6 ± 0.0	4.1 ± 0.0	8.9 ± 0.0	***6.6*** ± 0.0
$\sigma = 0.1$		**1.7** ± 0.0	**1.1** ± 0.0	**3.8** ± 0.0	**2.5** ± 0.0	**7.1** ± 0.0	**5.2** ± 0.0	2.2 ± 0.0	***1.7*** ± 0.0	***4.5*** ± 0.0	***3.7*** ± 0.0	**8.0** ± 0.0	**6.5** ± 0.0
$\sigma = 0.3$		4.1 ± 0.0	1.8 ± 0.0	9.5 ± 0.0	4.5 ± 0.0	21.3 ± 0.2	10.3 ± 0.0	2.5 ± 0.0	1.7 ± 0.0	5.8 ± 0.0	4.0 ± 0.0	10.5 ± 0.1	7.1 ± 0.1
$\sigma = 0.4$		8.5 ± 0.0	3.3 ± 0.0	19.7 ± 0.0	7.9 ± 0.0	46.4 ± 0.2	18.3 ± 0.1	4.1 ± 0.0	2.3 ± 0.0	9.5 ± 0.1	5.4 ± 0.0	17.5 ± 0.0	9.7 ± 0.0
$\sigma = 0.6$		28.0 ± 0.1	9.4 ± 0.0	63.8 ± 0.1	23.1 ± 0.0	140.2 ± 0.4	59.3 ± 0.3	9.8 ± 0.0	4.2 ± 0.0	24.6 ± 0.1	10.4 ± 0.1	50.6 ± 0.2	21.3 ± 0.2
Noise-to-Data Diffusion													
Cos: (3.0, 0.04)		57.9 ± 0.1	6.0 ± 0.0	108.0 ± 0.2	15.2 ± 0.0	153.4 ± 0.2	27.1 ± 0.3	7.3 ± 0.1	2.5 ± 0.0	17.0 ± 0.1	6.2 ± 0.0	26.4 ± 0.1	10.5 ± 0.1
Cos: (5.0, 0.05)		133.9 ± 0.2	14.0 ± 0.0	192.7 ±0.0	33.4 ± 0.2	227.3 ± 0.1	59.2 ± 0.3	15.3 ± 0.1	3.4 ± 0.0	36.6 ± 0.1	9.1 ± 0.1	58.4 ± 0.2	15.8 ± 0.2
Exp: (4.0, 2.5)		85.1 ± 0.1	9.9 ± 0.0	113.8 ± 0.2	24.9 ± 0.1	160.3 ± 0.3	85.5 ± 0.1	9.5 ± 0.0	2.3 ± 0.0	24.1 ± 0.1	7.2 ± 0.0	69.2 ± 0.2	15.5 ± 0.2
Exp: (5.0, 5.0)		153.9 ± 0.1	25.9 ± 0.0	163.3 ± 0.1	49.8 ± 0.1	189.7 ± 0.3	92.9 ± 0.1	23.2 ± 0.0	4.2 ± 0.0	44.9 ± 0.1	12.5 ± 0.0	80.1 ± 0.4	22.8 ± 0.2

Table 19: Sky-timelapse (Non-Causal) — Overall Average Rank (↓)

Method	Avg Rank	Rank
ABCD		
No BB: σ=0.4	***2.6*** ± 0.6	2
W/ BB: σ=0.4	**2.2** ± 0.5	1
Conditional Diffusion Bridge		
$\sigma = 0.071$	5.5 ± 1.5	6
$\sigma = 0.1$	3.3 ± 1.9	4
$\sigma = 0.3$	3.3 ± 1.3	3
$\sigma = 0.4$	5.0 ± 1.0	5
$\sigma = 0.6$	7.5 ± 1.0	7
Noise-to-Data Diffusion		
Cos: (3.0, 0.04)	7.7 ± 0.3	8
Cos: (5.0, 0.05)	10.0 ± 0.3	10
Exp: (4.0, 2.5)	8.3 ± 0.4	9
Exp: (5.0, 5.0)	10.6 ± 0.2	11

H.10 Sky-Timelapse: Causal Results

We use the same frame counts and discretization step counts as in the other experiments. Here, we do causal (regular autoregressive) generation, where we don't prompt with the future, but rather, only give a prefix of 1 or 4 frames.

See Tables 20 (FVD), 21 (FID), Figure 6 (combined ranking). We report mean and std over three random inference seeds. Under this setting, *ABCD without Brownian Bridge is Pareto-optimal* (Figure 6) across FVD and FID scores on causal rollout. As compared to the conditional diffusion bridge methods on the Pareto front, we see that ABCD has low discrepancy in its overall FID (3) and FVD (2) rankings, indicating consistency and robustness across multiple aspects of quality measure.

Table 20: Sky-timelapse — Causal — FVD (↓)

	Num Frames = 32				Num Frames = 16			
	prefix_length = 1		prefix_length = 4		prefix_length = 1		prefix_length = 4	
Method Steps	250	500	250	500	250	500	250	500
ABCD								
No BB: σ=0.4	***373.1*** $\pm$ 2.1	***283.3*** $\pm$ 0.9	***359.5*** $\pm$ 0.6	***277.3*** $\pm$ 0.8	***246.4*** $\pm$ 1.5	***209.0*** $\pm$ 0.5	***189.4*** $\pm$ 0.8	164.4 $\pm$ 0.3
W/ BB: σ=0.4	462.8 $\pm$ 1.4	364.0 $\pm$ 3.0	416.3 $\pm$ 1.2	330.5 $\pm$ 2.7	307.5 $\pm$ 1.5	241.0 $\pm$ 0.6	225.8 $\pm$ 1.6	181.5 $\pm$ 0.5
Conditional Diffusion Bridge								
$\sigma = 0.071$	485.9 $\pm$ 1.1	428.4 $\pm$ 1.9	438.4 $\pm$ 1.1	359.0 $\pm$ 0.9	399.0 $\pm$ 1.0	354.3 $\pm$ 1.8	308.8 $\pm$ 0.2	281.7 $\pm$ 1.0
$\sigma = 0.1$	409.0 $\pm$ 2.2	329.3 $\pm$ 0.3	389.0 $\pm$ 0.3	314.3 $\pm$ 0.5	326.4 $\pm$ 1.4	274.7 $\pm$ 1.4	251.5 $\pm$ 1.7	223.0 $\pm$ 0.5
$\sigma = 0.3$	**303.4** $\pm$ 1.2	**225.8** $\pm$ 1.8	**278.1** $\pm$ 1.1	**194.6** $\pm$ 1.1	**206.2** $\pm$ 1.7	**185.5** $\pm$ 1.4	**156.5** $\pm$ 1.9	**137.4** $\pm$ 1.0
$\sigma = 0.4$	442.0 $\pm$ 1.3	311.8 $\pm$ 0.6	435.4 $\pm$ 2.5	293.6 $\pm$ 2.8	269.8 $\pm$ 0.4	215.8 $\pm$ 1.8	220.0 $\pm$ 1.5	***159.9*** $\pm$ 2.4
$\sigma = 0.6$	746.4 $\pm$ 3.1	559.9 $\pm$ 5.7	753.3 $\pm$ 2.6	567.8 $\pm$ 3.7	456.6 $\pm$ 1.5	302.9 $\pm$ 2.2	438.3 $\pm$ 4.1	279.1 $\pm$ 2.1
Noise-to-Data Diffusion								
Cos: (3.0, 0.04)	698.6 $\pm$ 1.4	512.7 $\pm$ 2.9	750.4 $\pm$ 4.2	508.8 $\pm$ 5.0	504.1 $\pm$ 3.7	258.3 $\pm$ 2.8	475.5 $\pm$ 1.6	210.4 $\pm$ 2.7
Cos: (5.0, 0.05)	862.3 $\pm$ 1.0	782.4 $\pm$ 2.1	929.5 $\pm$ 2.1	842.2 $\pm$ 2.2	761.9 $\pm$ 3.3	363.0 $\pm$ 2.3	752.9 $\pm$ 4.0	324.7 $\pm$ 2.2
Exp: (4.0, 2.5)	764.8 $\pm$ 1.2	658.0 $\pm$ 1.1	618.5 $\pm$1.0	535.2 $\pm$ 2.2	621.0 $\pm$ 1.7	317.3 $\pm$ 2.7	419.8 $\pm$2.0	210.8 $\pm$ 0.5
Exp: (5.0, 5.0)	798.0 $\pm$ 1.7	680.1 $\pm$ 1.0	644.0 $\pm$0.4	580.1 $\pm$ 0.9	693.5 $\pm$ 0.9	411.1 $\pm$ 3.8	471.5 $\pm$1.2	300.5 $\pm$ 2.6

Table 21: Sky-timelapse — Causal — FID (↓)

	Num Frames = 32				Num Frames = 16			
	prefix_length = 1		prefix_length = 4		prefix_length = 1		prefix_length = 4	
Method Steps	250	500	250	500	250	500	250	500
ABCD								
No BB: σ=0.4	19.9 $\pm$ 0.1	14.1 $\pm$ 0.0	17.3 $\pm$ 0.1	12.2 $\pm$ 0.1	10.2 $\pm$ 0.1	8.5 $\pm$ 0.1	**6.6** $\pm$ 0.0	5.5 $\pm$ 0.0
W/ BB: σ=0.4	25.1 $\pm$ 0.1	17.4 $\pm$ 0.2	21.5 $\pm$ 0.1	14.7 $\pm$ 0.2	12.5 $\pm$ 0.0	10.5 $\pm$ 0.1	8.0 $\pm$ 0.1	6.5 $\pm$ 0.1
Conditional Diffusion Bridge								
$\sigma = 0.071$	**13.8** $\pm$ 0.1	**10.0** $\pm$ 0.1	**15.3** $\pm$ 0.1	**10.8** $\pm$ 0.0	**8.6** $\pm$ 0.1	**6.6** $\pm$ 0.0	8.1 $\pm$ 0.0	5.9 $\pm$ 0.0
$\sigma = 0.1$	***17.8*** $\pm$ 0.1	***13.1*** $\pm$ 0.1	***15.6*** $\pm$ 0.1	***11.2*** $\pm$ 0.1	***9.3*** $\pm$ 0.0	***7.4*** $\pm$ 0.1	6.8 $\pm$ 0.1	***5.2*** $\pm$ 0.0
$\sigma = 0.3$	43.5 $\pm$ 0.2	19.9 $\pm$ 0.1	30.0 $\pm$ 0.1	14.3 $\pm$ 0.1	11.8 $\pm$ 0.0	8.0 $\pm$ 0.0	***6.7***$\pm$ 0.0	**4.7** $\pm$ 0.0
$\sigma = 0.4$	95.7 $\pm$ 0.5	38.2 $\pm$ 0.2	68.4 $\pm$ 0.2	27.4 $\pm$ 0.1	20.0 $\pm$ 0.1	11.0 $\pm$ 0.1	11.3 $\pm$ 0.0	6.3$\pm$ 0.1
$\sigma = 0.6$	220.9 $\pm$ 0.6	122.1 $\pm$ 0.5	171.2 $\pm$ 0.9	95.2 $\pm$ 0.3	59.7 $\pm$ 0.1	25.1 $\pm$ 0.2	35.3 $\pm$ 0.2	15.4 $\pm$ 0.1
Noise-to-Data Diffusion								
Cos: (3.0, 0.04)	175.0 $\pm$ 0.4	39.1 $\pm$ 0.1	140.4 $\pm$0.1	28.1 $\pm$ 0.2	29.0 $\pm$ 0.2	12.1 $\pm$ 0.1	16.3 $\pm$ 0.2	6.6 $\pm$ 0.0
Cos: (5.0, 0.05)	246.0 $\pm$ 0.2	88.0 $\pm$ 0.2	203.3 $\pm$0.2	68.3 $\pm$ 0.1	62.7 $\pm$ 0.3	17.3 $\pm$ 0.2	37.8 $\pm$ 0.2	9.8 $\pm$ 0.1
Exp: (4.0, 2.5)	191.8 $\pm$ 0.3	141.2 $\pm$ 0.4	159.6 $\pm$0.4	116.2 $\pm$ 0.2	83.7 $\pm$ 0.2	18.5 $\pm$ 0.2	50.1 $\pm$ 0.1	9.7 $\pm$ 0.0
Exp: (5.0, 5.0)	211.6 $\pm$ 0.2	145.8 $\pm$ 0.1	181.5 $\pm$0.1	118.1 $\pm$ 0.1	90.4 $\pm$ 0.3	26.4 $\pm$ 0.2	55.4 $\pm$ 0.0	14.3 $\pm$ 0.1

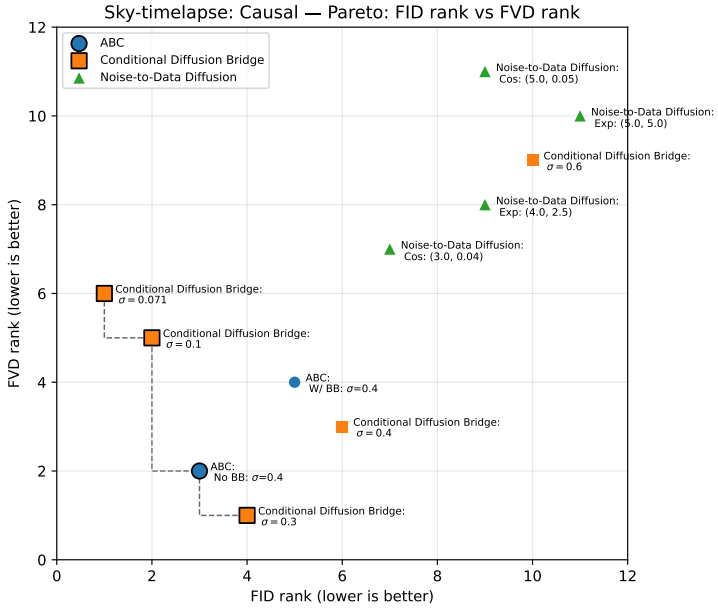


Figure 6: **Pareto Front for Sky-Timelapse Causal Generation:** ABCD (shortened to ABC in the chart) is Pareto-optimal across FVD and FID scores for causally-ordered generation on Sky-Timelapse. Corresponds to results in Tables 21, 20. Horizontal axis is the ordering of average rank in Table 21's columns, and vertical axis is the ordering of average rank in Table 20's columns.

H.11 CelebV-HQ: Causal Results

We retrained CelebV-HQ on a strictly causal conditioning structure (same frame sampling procedure as before, and then masking out any future conditioning frames that inadvertently got selected), with otherwise the same settings as in the previous experiments. We report mean and std over two random seeds for inference, with each trial calculated over 4096 videos. See Tables 22, 23, 24. Among the settings we tested, *ABCD is clearly better than its conditional diffusion bridge and noise-to-data cousins*.

Table 22: CelebV-HQ — Causal — FVD (↓)

	Num Frames = 32				Num Frames = 16			
	prefix_length = 1		prefix_length = 4		prefix_length = 1		prefix_length = 4	
Method Steps	250	500	250	500	250	500	250	500
ABCD								
No BB: σ=0.5	**301.7** $\pm$ 2.1	***278.3*** $\pm$ 3.0	***362.7*** $\pm$ 5.0	***242.1*** $\pm$ 0.5	**234.6** $\pm$ 0.5	227.2 $\pm$ 0.8	***181.2*** $\pm$ 0.4	139.1 $\pm$ 0.4
No BB: σ=0.75	***331.0*** $\pm$ 0.5	**275.6** $\pm$ 2.1	410.7 $\pm$ 0.5	284.5 $\pm$ 0.1	246.5 $\pm$ 1.5	***200.2*** $\pm$ 0.3	**177.8** $\pm$ 0.1	**133.3** $\pm$ 0.1
No BB: σ=1.0	343.7 $\pm$ 0.6	298.6 $\pm$ 2.1	**295.7** $\pm$ 1.3	**216.5** $\pm$ 0.3	281.8 $\pm$ 1.6	212.2 $\pm$ 0.1	194.9 $\pm$ 0.2	138.9 $\pm$ 0.2
Conditional Diffusion Bridge								
$\sigma = 0.25$	897.2 $\pm$ 4.4	517.9 $\pm$ 3.3	803.2 $\pm$ 0.5	490.7 $\pm$ 0.1	***239.2*** $\pm$ 0.8	**196.4** $\pm$ 1.3	193.8 $\pm$ 0.7	***137.4*** $\pm$ 0.7
$\sigma = 0.5$	1014.1 $\pm$ 0.6	541.5 $\pm$ 2.1	889.6 $\pm$ 0.7	471.8 $\pm$ 0.6	430.9 $\pm$ 2.3	273.4 $\pm$ 0.0	365.3 $\pm$ 0.4	220.2 $\pm$ 0.9
$\sigma = 0.75$	977.5 $\pm$ 1.0	726.1 $\pm$ 1.8	887.9 $\pm$ 2.9	653.4 $\pm$ 2.3	674.7 $\pm$ 0.3	426.7 $\pm$ 0.3	575.1 $\pm$ 0.2	368.0 $\pm$ 1.0
$\sigma = 1.0$	938.3 $\pm$ 1.5	796.6 $\pm$ 0.5	764.7 $\pm$ 1.6	702.9 $\pm$ 1.1	726.5 $\pm$ 1.1	535.8 $\pm$ 1.3	629.6 $\pm$ 1.9	489.2 $\pm$ 2.4
Noise-to-Data Diffusion								
Cos: (3.0, 0.04)	1030.9 $\pm$ 1.4	1055.9 $\pm$ 0.5	981.4 $\pm$ 1.3	1059.6 $\pm$ 0.3	1059.6 $\pm$ 1.5	591.9 $\pm$ 0.8	990.0 $\pm$ 0.3	458.4 $\pm$ 0.9

Table 23: CelebV-HQ — Causal — FID (↓)

	Num Frames = 32				Num Frames = 16			
	prefix_length = 1		prefix_length = 4		prefix_length = 1		prefix_length = 4	
Method Steps	250	500	250	500	250	500	250	500
ABCD								
No BB: σ=0.5	**13.8** $\pm$ 0.1	**5.8** $\pm$ 0.0	**19.8** $\pm$ 0.1	**11.5** $\pm$ 0.0	**6.8** $\pm$ 0.0	**4.1** $\pm$ 0.0	**8.3** $\pm$ 0.0	**5.4** $\pm$ 0.0
No BB: σ=0.75	***27.3*** $\pm$ 0.0	***12.0*** $\pm$ 0.0	***33.5*** $\pm$ 0.0	***17.5*** $\pm$ 0.0	***13.2*** $\pm$ 0.0	***7.2*** $\pm$ 0.0	***9.6*** $\pm$ 0.0	***5.8*** $\pm$ 0.0
No BB: σ=1.0	40.6 $\pm$ 0.1	20.8 $\pm$ 0.1	34.7 $\pm$ 0.2	19.1 $\pm$ 0.0	21.8 $\pm$ 0.0	11.8 $\pm$ 0.0	13.0 $\pm$ 0.0	7.5 $\pm$ 0.1
Conditional Diffusion Bridge								
$\sigma = 0.25$	117.1 $\pm$ 0.2	51.4 $\pm$ 0.1	91.4 $\pm$ 0.0	43.0 $\pm$ 0.1	16.4 $\pm$ 0.0	8.9 $\pm$ 0.0	11.8 $\pm$ 0.0	6.7 $\pm$ 0.0
$\sigma = 0.5$	182.2 $\pm$ 0.2	99.0 $\pm$ 0.1	132.5 $\pm$ 0.2	74.2 $\pm$ 0.0	64.8 $\pm$ 0.1	33.1 $\pm$ 0.0	37.5 $\pm$ 0.1	19.2 $\pm$ 0.1
$\sigma = 0.75$	256.9 $\pm$ 0.1	169.8 $\pm$ 0.2	195.9 $\pm$ 0.3	127.7 $\pm$ 0.1	122.9 $\pm$ 0.1	68.3 $\pm$ 0.1	73.9 $\pm$ 0.1	40.6 $\pm$ 0.1
$\sigma = 1.0$	283.7 $\pm$ 0.4	232.7 $\pm$ 0.1	221.6 $\pm$ 0.4	184.2 $\pm$ 0.0	170.4 $\pm$ 0.1	105.9 $\pm$ 0.1	107.6 $\pm$ 0.1	65.3 $\pm$ 0.1
Noise-to-Data Diffusion								
Cos: (3.0, 0.04)	281.0 $\pm$ 0.3	185.0 $\pm$ 0.0	237.6 $\pm$ 0.2	143.5 $\pm$ 0.1	140.8 $\pm$ 0.1	40.0 $\pm$ 0.2	83.0 $\pm$ 0.2	18.2 $\pm$ 0.1

Table 24: CelebV-HQ (Causal) — Overall Average Rank (↓)

Method	Avg Rank	Rank
ABCD		
No BB: σ=0.5	**1.6** $\pm$ 0.6	1
No BB: σ=0.75	***2.0*** $\pm$ 0.0	2
No BB: σ=1.0	3.1 $\pm$ 0.4	3
Conditional Diffusion Bridge		
$\sigma = 0.25$	3.4 $\pm$ 0.1	4
$\sigma = 0.5$	5.2 $\pm$ 0.1	5
$\sigma = 0.75$	6.1 $\pm$ 0.1	6
$\sigma = 1.0$	7.2 $\pm$ 0.7	7
Noise-to-Data Diffusion		
Cos: (3.0, 0.04)	7.3 $\pm$ 0.6	8

H.12 SEVIR: Additional Results

H.12.1 Qualitative Comparison

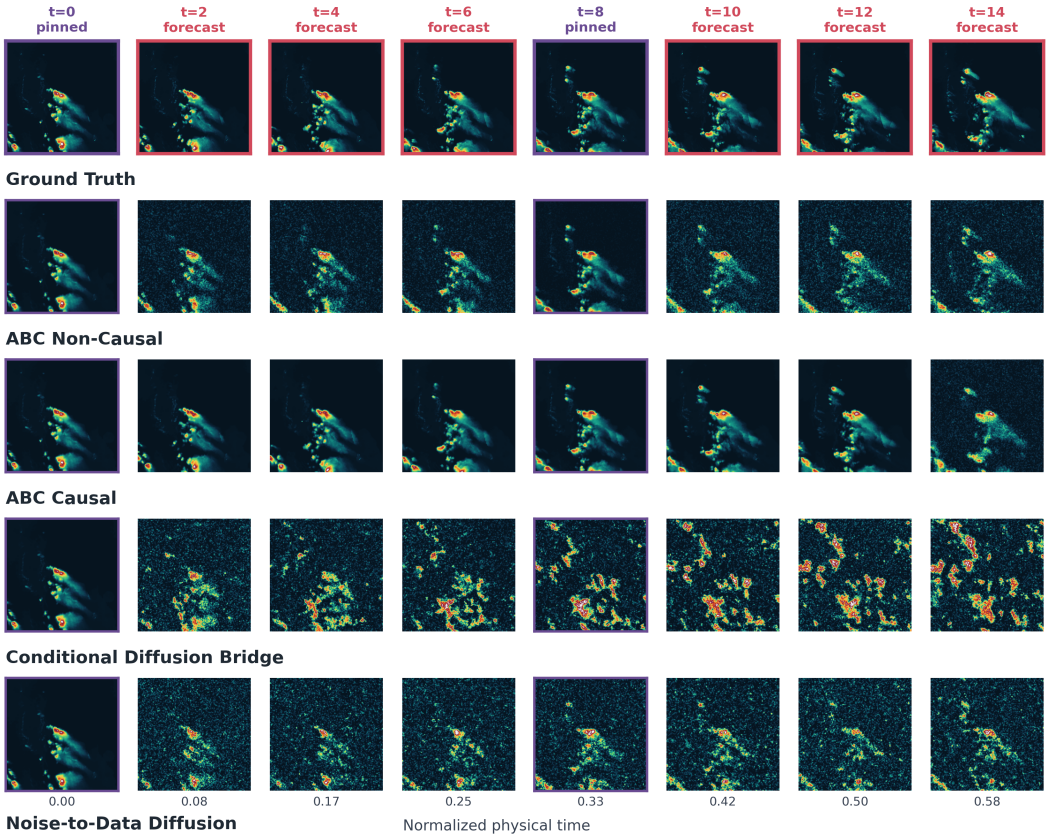


Figure 7: **SEVIR qualitative comparison.** ABCD Non-Causal, Conditional Diffusion Bridge, and Noise-to-Data Diffusion are evaluated under the pinned non-causal protocol, while ABCD Causal is shown separately under the stricter past-only protocol. Individual examples may favor either protocol visually, but the aggregate metrics in Table 3 favors ABCD Non-Causal among the pinned methods.

H.12.2 SEVIR: Additional Hyperparameter Sweep

After the initial round of SEVIR experiments referenced in the main text (which used constant $\sigma = 1.0$), we had some extra computational resources and time; so we ran another hyperparameter sweep over $\sigma \in \{0.25, 0.5, 0.75\}$. The results again clearly show ABCD's advantage. See Tables 25, 26.

Table 25: SEVIR results: error and CSI metrics. Best per column in **bold**, second best in *italic*.

Method	MAE (↓)	RMSE (↓)	CSI_{16} (↑)	CSI_{74} (↑)	CSI_{133} (↑)	CSI_{160} (↑)	Avg. Rank (↓)
ABCD Non-Causal							
$\sigma = 0.25$	**14.00**	**32.21**	**0.739**	**0.731**	**0.662**	**0.678**	**1.00**
$\sigma = 0.5$	***15.08***	***33.29***	***0.705***	***0.716***	***0.656***	***0.671***	***2.00***
$\sigma = 0.75$	17.14	34.05	0.602	0.709	0.651	0.669	3.50
ABCD Causal							
$\sigma = 0.25$	21.38	54.52	0.665	0.583	0.487	0.502	4.17
$\sigma = 0.5$	22.40	54.26	0.631	0.583	0.485	0.501	4.67
$\sigma = 0.75$	23.15	54.55	0.595	0.582	0.483	0.499	6.17
Autoregressive Diffusion Bridge							
$\sigma = 0.25$	40.87	79.26	0.423	0.349	0.313	0.326	9.50
$\sigma = 0.5$	43.45	81.05	0.381	0.346	0.254	0.245	10.67
$\sigma = 0.75$	54.83	89.82	0.303	0.231	0.155	0.142	12.00
Noise-to-Data Diffusion							
$\sigma = 0.25$	24.44	59.37	0.633	0.535	0.382	0.363	6.50
$\sigma = 0.5$	33.16	64.93	0.462	0.399	0.206	0.145	9.33
$\sigma = 0.75$	33.13	65.36	0.424	0.476	0.339	0.326	8.50

Table 26: SEVIR results: POD, FAR, and Bias metrics. Best per column in **bold**, second best in *italic*; for Bias, "best" is the value closest to 1.0.

Method	POD_{16} (↑)	POD_{74} (↑)	POD_{133} (↑)	POD_{160} (↑)	FAR_{16} (↓)	FAR_{74} (↓)	FAR_{133} (↓)	FAR_{160} (↓)	$Bias_{16}$	$Bias_{74}$	$Bias_{133}$	$Bias_{160}$
ABCD Non-Causal												
$\sigma = 0.25$	**0.947**	**0.863**	0.776	0.760	***0.230***	**0.174**	**0.181**	**0.138**	1.230	1.045	0.948	0.882
$\sigma = 0.5$	0.930	***0.855***	**0.787**	**0.774**	0.255	***0.185***	0.203	0.165	1.249	1.050	**0.987**	**0.926**
$\sigma = 0.75$	***0.936***	0.852	***0.777***	***0.765***	0.373	0.191	***0.200***	***0.157***	1.491	1.053	***0.971***	***0.908***
ABCD Causal												
$\sigma = 0.25$	0.823	0.728	0.628	0.636	**0.225**	0.254	0.316	0.297	**1.062**	**0.976**	0.917	0.905
$\sigma = 0.5$	0.822	0.726	0.620	0.629	0.269	0.252	0.311	0.290	1.125	***0.971***	0.900	0.886
$\sigma = 0.75$	0.817	0.725	0.625	0.634	0.314	0.253	0.320	0.299	1.190	0.970	0.920	0.904
Autoregressive Diffusion Bridge												
$\sigma = 0.25$	0.670	0.582	0.579	0.613	0.466	0.534	0.595	0.589	1.255	1.249	1.431	1.491
$\sigma = 0.5$	0.697	0.545	0.461	0.457	0.544	0.513	0.640	0.654	1.529	1.118	1.279	1.320
$\sigma = 0.75$	0.654	0.427	0.313	0.279	0.640	0.665	0.765	0.775	1.813	1.275	1.334	1.240
Noise-to-Data Diffusion												
$\sigma = 0.25$	0.800	0.656	0.487	0.461	0.249	0.255	0.361	0.369	***1.066***	0.881	0.762	0.732
$\sigma = 0.5$	0.770	0.512	0.255	0.167	0.463	0.355	0.481	0.478	1.435	0.794	0.492	0.319
$\sigma = 0.75$	0.826	0.621	0.477	0.461	0.534	0.330	0.459	0.474	1.775	0.926	0.882	0.878

H.13 Runtime Performance Profiling

We ran profiling experiments generating Sky-Timelapse videos from a single prompt frame, and report mean±SEM across batches. Per setting, we generate 800 videos in batches of 16 videos per GPU (A100, 80GB).

Table 27: Cost vs. number of conditioning frames (32 generated frames, 250 SDE steps)

Conditioning frames	1	5	9	13	17	21	25	29
Denoising steps measured	400	400	400	400	400	400	400	400
GPU time per denoising step (ms)	108.03 ± 1.76	111.64 ± 1.74	118.57 ± 1.76	131.84 ± 1.85	141.39 ± 1.79	149.08 ± 1.27	157.12 ± 1.28	169.99± 1.26
Peak memory (GiB)	6.440 ± 0.000	6.632 ± 0.000	6.842 ± 0.000	7.056 ± 0.000	7.272 ± 0.000	7.485 ± 0.000	7.696 ± 0.000	7.911 ± 0.000

Table 28: Cost vs. number of generated frames (250 SDE steps)

Generated frames	8	16	32
Sampling GPU time (s)	23.46 ± 1.60	27.01 ± 1.37	34.74 ± 1.02
VAE decode GPU time (s)	1.51 ± 0.00	2.85 ± 0.00	5.53 ± 0.00
Total wall time (s)	24.96 ± 1.60	29.85 ± 1.37	40.27 ± 1.02
Wall time per video (s)	1.56 ± 0.10	1.87 ± 0.09	2.52 ± 0.06
Peak memory, sampling (GiB)	6.21 ± 0.00	6.83 ± 0.00	8.07 ± 0.00
Peak memory, sampling + decode (GiB)	12.95 ± 0.00	13.24 ± 0.00	13.82 ± 0.00

Table 29: Cost vs. number of SDE steps (8 generated frames)

SDE steps	125	250	375
Sampling GPU time (s)	15.00 ± 0.88	23.46 ± 1.60	32.57 ± 2.10
VAE decode GPU time (s)	1.51 ± 0.00	1.51 ± 0.00	1.51 ± 0.00
Total wall time (s)	16.50 ± 0.88	24.96 ± 1.60	34.07 ± 2.10
Wall time per video (s)	1.03 ± 0.05	1.56 ± 0.10	2.13 ± 0.13
Peak memory, sampling (GiB)	6.21 ± 0.00	6.21 ± 0.00	6.21 ± 0.00
Peak memory, sampling + decode (GiB)	12.95 ± 0.00	12.95 ± 0.00	12.95 ± 0.00

I Architecture

We build on the Diffusion Transformer (DiT) [47] and extend it with image-based conditioning, conditioning on the next waypoint $t_{\text{next}} = t_{i^*+1}$, and an optional Brownian-bridge drift term. The output is the estimated drift from Equation 8. *All methods compared in the paper use the same architecture.*

Inputs and embeddings. In addition to the noisy latent $x_t \in \mathbb{R}^{C\times H\times W}$, diffusion timestep $t \in [0,1]$, and class label y used by DiT, our model takes a next-waypoint timestep $t_{\text{next}} \in [0,1]$, a set of L conditioning frames $\{x^{(\ell)}\}_{\ell=1}^{L}$ with associated timesteps $\{t^{(\ell)}\}$, and a boolean validity mask $m \in \{0,1\}^L$ (with $m_0 = 1$ always). The conditioning frames are patchified by a dedicated embedder $\mathrm{PatchEmbed}_{\text{cond}}$ and additively tagged with the shared sin-cos positional embedding plus a timestep embedding $\phi_{\text{cond}}(t^{(\ell)})$, yielding tokens $z^{(\ell)} \in \mathbb{R}^{T\times D}$. Timesteps are rescaled by a factor of 1000 before sinusoidal embedding. The AdaLN conditioning vector becomes

$$c \;=\; \tfrac{1}{2}\big(\phi_t(t) + \phi_{t_{\text{next}}}(t_{\text{next}})\big) + \phi_y(y),$$

where ϕ_t, $\phi_{t_{\text{next}}}$, and ϕ_{cond} are independent timestep MLPs.

Block structure. Each block extends the DiT block with a cross-attention sub-layer inserted between self-attention and the MLP:

$$x \leftarrow x + g_{\text{sa}}\,\mathrm{SA}(\mathrm{mod}(x; s_{\text{sa}}, \sigma_{\text{sa}})), \tag{67}$$

$$x \leftarrow x + g_{\text{xa}}\,\mathrm{XA}\big(\mathrm{mod}(x; s_{\text{xa}}, \sigma_{\text{xa}}),\ \mathrm{mod}(z; s_z, \sigma_z)\big), \tag{68}$$

$$x \leftarrow x + g_{\text{mlp}}\,\mathrm{MLP}(\mathrm{mod}(x; s_{\text{mlp}}, \sigma_{\text{mlp}})), \tag{69}$$

so the AdaLN-Zero MLP emits $11D$ modulation parameters per block (vs. $6D$ in DiT). Two additional LayerNorms are introduced for the cross-attention query and key/value streams. We apply RMSNorm [69] to the per-head queries and keys in both self- and cross-attention to stabilize training on long contexts (the orignal DiT did not include this, but we found the exclusion to be very unstable).

Variable-length cross-attention. For cross-attention, we use PyTorch's `varlen_attn`, which is based on FlashAttention [14, 13].

Brownian-bridge drift. We have an option to add the Brownian Bridge drift specified in Section 3.4. We add a small epsilon of 10^{-7} to the denominator for stability.

Initialization. We follow DiT's AdaLN-Zero initialization, additionally applying Xavier initialization [21] to the conditioning patch embedder and $\mathcal{N}(0, 0.02^2)$ to the new timestep MLPs (ϕ_{cond} and $\phi_{t_{\text{next}}}$).

Model Size We use the DiT-B model size. With our cross-attention layers and other architectural modifications, this works out to 202,192,912 parameters.

EMA We use EMA 0.9999, following DiT defaults.

J Video Experimental Setup Details

J.1 Dataset Details

We train on 178,688 and evaluate on 2,048 clips (disjoint) of 32 frames each from CelebV-HQ [76]. We train on 17,152 and evaluate on 1,360 clips (disjoint) of 64 frames each from Sky-Timelapse [70]. We diffuse in the Stable Diffusion latent space [53] on 32x32x4 codes, corresponding to 256x256x3 images.

J.2 Training

We sample the time $t \sim \mathcal{U}[0,1]$, then clip it so that $|t - t_i| > 10^{-4}$, for every finite-dimensional marginal time t_i. For the autoregressively conditioned diffusion bridge (and noise-to-data diffusion), we get the auxiliary time $\tau_i = \frac{t-t_{i^*}}{t_{i^*+1}-t_{i^*}}$, as constructed in Equation 16.

We have cosine decay learning rate, with linear warmup for 10K steps up to rate 10^{-4}, then decay for 290K steps. Due to compute limits, we evaluate the model from 240K steps. We use Adam [34] without weight decay, and $\beta = (0.9, 0.999)$. We clip the gradient to maximum norm of 5.0. We use batch size 256. We run each training run on 4 A100 80GB GPUs, and each run takes about 36 hours.

J.3 SDE Parameter Selection

There is a very large design space of SDE parameters that could have been used in Equations 6 and 17. To isolate the effects of RQ2 (impact of time-dependent quadratic variation), RQ3 (impact of data-to-data bridge), we picked relatively simple choices. Future work can explore more complicated parameterizations.

We always set $t_L = 1$, ensuring that the SDE is defined on $[0, 1]$.

We set the base (non-score driven) drift to be $a(t) = 0$. The score network should be able to learn the drift, without need for an explicit specification.

We use constant volatility $\sigma(t) = \sigma$ in ABCD and the conditional diffusion bridge baseline. This is because a non-constant volatility (say, small at the finite-dimensional marginal times, but large in between them, as is popular in DDBM [75]) would mean that the generative process would not have physically meaningful intermediate values in between the times at which we observe the finite-dimensional marginals. This is a non-starter for continuous-time modeling, as it kills any hope of generalization to modeling times that were not on the grid that the training data was sampled from. In contrast, using a constant volatility decouples the sampling rate of the data from the underlying model of the stochastic process.

We selected the volatility by testing reconstruction fidelity in the Stable Diffusion VAE latent space [53] with various Gaussian noise perturbations. We found that perturbations up to $\mathcal{N}(0, 0.1^2 I)$ minimally affected the reconstruction quality. Assuming simulation step sizes up to $\Delta t \leq 0.01$, the upper limit on the volatility introduced from Brownian motion should be $\sigma\sqrt{\Delta t} \leq 0.1 \Rightarrow \sigma \leq 1.0$.

To select volatility for the baseline conditional diffusion bridge, we tried a few options. The first, most naive one was to set the volatility coefficient exactly the same as that of ABCD's:

$$\sigma_{\text{cond bridge}} = \sigma_{\text{ABCD}} \tag{70}$$

Another option was to calculate the volatility coefficient based on the quadratic variation (up to rounding) that would have been introduced by ABCD in between frames (given a frame sampling rate):

$$\sigma_{\text{cond bridge}} = \sqrt{t_{i^*+1} - t_{i^*}} * \sigma_{\text{ABCD}}, \tag{71}$$

in accordance with time-scaling rules for Brownian motion [57]. Finally, when compute resources permitted, we tried a few other volatility values.

For the noise-to-data diffusion baseline, we used exponentially decaying volatility (Section L.1) and cosine decaying volatility (Section L.4).

J.4 Inference

We also concatenate the final frame from $t = 1$ as a prompt.

J.5 Evaluation

FVD and FID are computed on the infilled videos (including the prompt frames), with the same number of videos (1360 in Sky Timelapse, 2048 in CelebV-HQ) for each of the ground truth and fake sets. We exclude the final frame ($t = 1$) from computations. FVD computations use VideoMAE v2 backbone with interpolated time embeddings [66, 62, 19]. FID computations use the default Inception net [59].

K Other Experimental Setups

K.1 Toy Experiment: Incorrect Dynamics in Chained Conditional Bridges

We conduct a toy experiment to empirically illustrate Theorem 3. Although the finite-dimensional marginals of autoregressively concatenated conditional diffusion bridges can agree with those of an underlying stochastic process, the path measures and volatility in general will not coincide. This underscores the need for one continual SDE, as in our ABCD framework.

Reference process. Let $X(t)$ be standard Brownian motion on $[0,1]$ with $X(0) = x_0$, Doob-conditioned [17] to satisfy $X(4/5) = -x_0$ and $X(1) = +x_0$. Its SDE is piecewise [33, 46, 57]

$$dX(t) = \begin{cases} \dfrac{-x_0 - X(t)}{4/5 - t}\, dt + dW(t), & t \in [0,\, 4/5], \\ \dfrac{x_0 - X(t)}{1 - t}\, dt + dW(t), & t \in [4/5,\, 1]. \end{cases} \tag{72}$$

Stitched construction. As an alternative, we build $Y(t)$ from two unit-time Brownian bridges defined on their own time variables. Let $Y_0(\tau_0)$ be a Brownian bridge on $\tau_0 \in [0,1]$ with $Y_0(0) = x_0$, conditioned on $Y_0(1) = -x_0$, and let $Y_1(\tau_1)$ be a Brownian bridge on $\tau_1 \in [0,1]$ with $Y_1(0) = Y_0(1) = -x_0$, conditioned on $Y_1(1) = -Y_0(1) = +x_0$. Their SDEs are the standard Brownian-bridge forms

$$dY_0(\tau_0) = \frac{-x_0 - Y_0(\tau_0)}{1 - \tau_0}\, d\tau_0 + dB_0(\tau_0), \tag{73}$$

$$dY_1(\tau_1) = \frac{x_0 - Y_1(\tau_1)}{1 - \tau_1}\, d\tau_1 + dB_1(\tau_1), \tag{74}$$

where B_0, B_1 are independent standard Brownian motions in their respective time variables. We then define

$$Y(t) = \begin{cases} Y_0\big(\frac{5}{4}t\big)\,, & t \in [0,\, 4/5], \\ Y_1\big(5\,(t - 4/5)\big)\,, & t \in [4/5,\, 1], \end{cases} \tag{75}$$

so that each unit-time bridge is compressed into its respective sub-window.

Derivation of the SDE for $Y(t)$. Consider the first segment $t \in [0, 4/5]$ with $\tau_0 = \frac{5}{4}t$, hence $d\tau_0/dt = 5/4$. The drift transforms by the chain rule (multiply by $d\tau_0/dt$):

$$\frac{-x_0 - Y_0(\tau_0)}{1 - \tau_0} \cdot \frac{d\tau_0}{dt} = \frac{5}{4} \cdot \frac{-x_0 - Y(t)}{1 - \frac{5}{4}t} = \frac{-x_0 - Y(t)}{4/5 - t}. \tag{76}$$

For the diffusion term, $B_0(\tau_0) \sim \mathcal{N}(0, \tau_0) \overset{\text{dist}}{=} \mathcal{N}\left(0, \frac{5}{4}t\right)$, so the diffusion coefficient is $\sqrt{5/4}$ [33, 46, 57]. An identical argument on the second segment with $\tau_1 = 5(t - 4/5)$, $d\tau_1/dt = 5$, gives drift $(x_0 - Y(t))/(1 - t)$ and diffusion coefficient $\sqrt{5}$. Hence

$$dY(t) = \begin{cases} \dfrac{-x_0 - Y(t)}{4/5 - t}\, dt + \sqrt{\frac{5}{4}}\, dW(t), & t \in [0,\, 4/5], \\ \dfrac{x_0 - Y(t)}{1 - t}\, dt + \sqrt{5}\, dW(t), & t \in [4/5,\, 1]. \end{cases} \tag{77}$$

The drifts of X and Y coincide, but the diffusion coefficients of Y are inflated by $\sqrt{d\tau/dt}$ on each segment—a factor that is easy to overlook when "reusing" bridges across rescaled time windows.

Empirical Results. X and Y have identical finite-dimensional marginals at the checkpoint set $\{0,\, 4/5,\, 1\}$ (all three are point masses at $x_0, -x_0, x_0$), yet their path measures do not coincide: their quadratic variations differ pathwise, e.g. $\langle Y \rangle_t = \frac{5}{4}t$ versus $\langle X \rangle_t = t$ on the first segment. Figure 4 confirms this empirically with 1000 simulated trajectories (4000 discretization steps): the two processes coincide in distribution at the pinned times $t \in \{4/5, 1\}$ but differ markedly at intermediate times $t \in \{1/2, 9/10\}$, with Y exhibiting the predicted variance inflation. In this experiment, we use the analytic form of the drift (with small ϵ in denominator to promote numerical stability) to isolate the effect of neural network learning from the fundamental issue of path measures coinciding.

K.2 Weather Forecasting Experimental Setup

For SEVIR, we use the VIL modality [64] and construct a benchmark protocol from 25-frame clips taken from the 2019 radar sequences, with a center clip start at frame 12. This yields 13 context frames and 12 forecast frames per clip. We split events at 2019-06-01, reserve 10% of the pre-cutoff events for validation, and obtain 2,729 training events, 334 validation events, and 847 test events. Training is performed on resized 128×128 single-channel VIL fields, while evaluation is reported at native resolution by upsampling predictions before scoring. We train for 20 epochs at a constant learning rate of 10^{-4}, with a batch size of 4, on a single GPU: this takes around three hours. We gradient clip the norms to maximum 1, and use EMA 0.999.

We compare four methods: ABCD Non-Causal, ABCD Causal, Conditional Diffusion Bridge, and Noise-to-Data Diffusion. For the SEVIR experiments mentioned in the main text and the qualitative figure (Table 3, Figure 7), all the benchmarked methods used the same constant volatility schedule of $\sigma = 1.0$. For completeness, we also try different hyperparameters (Section H.12.2).

In the pinned non-causal protocol, the model conditions on the first, every 8th, and final frame of the clip and predicts the remaining frames; in the causal protocol, it conditions on the first 13 frames and predicts the final 12. All SEVIR results use 250 sampling steps and one sample per test example. We report MAE, MSE, RMSE, and thresholded CSI, POD, FAR, and bias at VIL thresholds $\{16, 74, 133, 160\}$, computed only on the unobserved/forecast frames.

L SDE Kernel Choices

L.1 Exponentially Decaying Volatility

We set

$$A(t) = A \tag{78}$$

$$\sigma(t) = Ke^{-Bt}, \tag{79}$$

where $A, B, K \in \mathbb{R}_{+0}$ are positive constants. This gives us an analytic form for the Gaussian process statistics (Equation 20):

$$\Phi(s,t) = e^{-A(t-s)}, \tag{80}$$

where $s < t$. Furthermore, plugging Equation 80 into Equation 21,

$$C_{t_{i^*}}(\tau_a, \tau_b) = \int_{t_{i^*}}^{\min(\tau_a,\tau_b)} e^{-A(\tau_a - s)} e^{-A(\tau_b - s)} K^2 e^{-2Bs} ds \tag{81}$$

$$= K^2 e^{-A(\tau_a+\tau_b)} \int_{t_{i^*}}^{\min(\tau_a,\tau_b)} e^{2(A-B)s} ds \tag{82}$$

$$= K^2 e^{-A(\tau_a+\tau_b)} \frac{1}{2(A-B)} \left[e^{2(A-B)s} \right]_{s=t_{i^*}}^{s=\min(\tau_a,\tau_b)} \tag{83}$$

$$= K^2 e^{-A(\tau_a+\tau_b)} \frac{1}{2(A-B)} \left[e^{2(A-B)\min(\tau_a,\tau_b)} - e^{2(A-B)t_{i^*}} \right] \tag{84}$$

$$\tag{85}$$

L.2 Periodic Volatility

$$\sigma(t) = \frac{\alpha}{2}\left(1 - \cos\left(2\pi k t\right)\right) + \epsilon \tag{86}$$

$$A(t) = 0 \tag{87}$$

$$\Phi(s,t) = 1 \tag{88}$$

$$C_{t_{i^*}}(\tau_a, \tau_b) = \int_{t_{i^*}}^{\min(\tau_a,\tau_b)} 1 * 1 * \sigma(s)^2 ds \tag{89}$$

$$= \left[\left(\frac{3\alpha^2}{8} + \alpha\epsilon + \epsilon^2 \right) s - \frac{\alpha(\alpha + 2\epsilon)}{4\pi k} \sin(2\pi k s) + \frac{\alpha^2}{32\pi k} \sin(4\pi k s) \right]_{s=t_{i^*}}^{s=\min(\tau_a,\tau_b)} \tag{90}$$

L.3 Constant Volatility

Simply set $\alpha = 0, k = 1$ in the periodic volatility kernel (Section L.2).

L.4 Cosine Decaying Volatility

Use $\sigma_{\text{cos decay}} = \sigma_{\text{cos periodic}}(t-1)$, where $k = 0.5$ (from Section L.2).

L.5 General Case

In general, we should set the volatility to match the dynamics of the underlying process, which may result in a non-monotonic function. In this case, we may have to integrate numerically.

M Related Works

M.1 Non-Markovian Diffusions

Our method is non-Markovian, as it makes use of path history. There is also recent work in non-Markovian diffusion modeling [43, 44] that approximately parameterizes the driving noise as fractional Brownian motion. Our framework actually supersedes such frameworks, assuming we have to discretize the SDE to simulate it. This is because many non-Markovian processes (including fractional Brownian Motion) can be written as Volterra SDEs [6]:

$$dX(t) = -AX(t)dt + \left(\int_0^t \sigma(t,s) dB(s) \right) dt + \sigma(t,t) dB(t) \tag{91}$$

In principle, if we constructed our martingale to consider every point on the discretization grid, our score network would take this entire history as input. It could then learn the middle integral over past noise history as part of the score-driven drift, assuming the underlying time series data actually has Volterra dynamics.

In fact, our framework is strictly more flexible, because it can learn from data what the best way to incorporate the path history is when generating the next sample. In contrast, the fractional Brownian Motion diffusion models use a handcrafted kernel to incorporate path history. Furthermore, our framework allows us to take the intermediate steps of the generative process as valid samples, whereas the other works do not.

Daems et al [12] also explore variational inference for (approximate) fractional Brownian motion driven SDEs to generate videos. However, they do not make the connection to any-subset autoregressive models nor score matching.

M.2 Time-Series Modeling

Longitudinal Flow Matching Longitudinal flow matching [30] also uses diffusion/flow processes to generate time series data, where the time variable in the generative process corresponds to the actual time in the physical time series. However, their work has a few key differences from ours: (1) We explicitly unify diffusion (bridge) models and any-subset autoregressive models (Section 3.4). Their procedure does not support any-subset conditioning (in the future or arbitrary past), nor is it even autoregressive, as it only conditions on the most recent observation, making their predictions Markovian. This stems from their choice to focus on multi-marginal optimal transport (rather than learning interdependencies along the trajectory). (2) Their loss function and simulation require three objects (velocity, score, volatility) parameterized by learned neural networks. This complicates the learning problem with three losses: flow matching (on velocity), score matching (on score), confidence estimation (on volatility; using the velocity and scores to form a distillation-like target). We just require one item learned with one loss: the path-dependent drift (Equation 8). (3) Our method is rigorously derived from stochastic calculus, measure theory, and Gaussian process regression. Their method is derived from different ideas, such as flow matching and optimal transport. (4) The scope of their experiments is mostly limited to medical imaging; we tackle video generation and weather forecasting (Section 6). (5) They do not provide detailed analysis on the benefits of time-adaptive volatility, and how that affects learning of path measures (*i.e.*, correctness of trajectories). We provide both theoretical (Theorem 3) and empirical evidence (Figure 4) of this phenomenon. See Section H.2 for an empirical comparison.

Time Series Schrödinger Bridge SBTS diffusion [24, 2] proposes to frame time series generation as a Schrödinger bridge problem between path measures, and also uses a similar measure-theoretic argument with Girsanov's Theorem to derive their method. However, their work has a few key limitations: (1) They assume a fixed discretization, and only model causally through time. As such, they do not support any-subset autoregressive modeling [23], either with respect to the past conditioning or future conditioning. (2) Their learning objectives for the drift do not scale to high-dimensions. In contrast, we show that the change-of-measure and Gaussian process regression can ultimately be used to derive a tractable, scalable denoising score matching objective [65]; they fail to see this connection. (3) They do not show success on any high-dimensional tasks at scale (largest is MNIST). (4) They were generally unable to leverage the power of modern deep learning architectures, at most using LSTMs on low-dimensional problems.

Multi-Marginal Schrödinger Bridges DMSB [9], SF^{2}M [61], 3MSBM [60], and SSB [18] (this one lays out the theory, but no experiments) learn multi-marginal Schrödinger Bridges, but (1) only respect the marginal rather than joint distributions, (2) do not make the connection to any-subset autoregressive modeling, (3) have limited experimental validation on high-dimensional problems. Smooth Schrödinger Bridges [29] also tackles trajectory inference, but (1) their belief-propagation-based method does not scale to high-dimensional problems, (2) they do not use neural networks, (3) while they do draw some connections to autoregressive modeling and Gaussian processes like our work, they do not show any connection to any-subset autoregressive models. Chen et al [11] also tried to generate time series with Schrödinger bridges, but they modeled the time series as the high-dimensional endpoint distribution of the SDE, rather than a distribution over intermediate variables in the generative SDE itself.

Multi-Marginal Flow Matching Multi-Marginal flow matching also tries to respect certain marginal distributions at intermediate times in flow matching trajectories [35]. However, they do not actually learn the joint distribution (but only the marginals), and do not account for any-subset inference (in discrete or continuous time).

Trajectory Flow Matching (TFM) Like ABCD, TFM [74] attempts to model time series and indeed has a time-sensitive velocity. However, they do not actually guarantee to learn the joint time series distribution.

They use a reweighted flow-matching objective (TFM Sec. 3.2) to train the SDE drift, which learns a history-conditioned point estimate of the next state. Their learned "volatility" (TFM Sec. 3.4), as they note, is more like an epistemic uncertainty estimate for their next-state predictor of x_{next}.

They admit their model only learns the correct joint probability distribution when the trajectory is effectively deterministic given observation history (TFM Prop. 3.2, (A1)-(A3)). Intuitively, the loss described in their Prop. 3.2 only achieves zero if there is no stochasticity in the target. They say they need a Monge map (deterministic OT) in order for their model to preserve the couplings.

In contrast, ABCD is explicitly designed to model the correct joint distributions of a time series, and can handle distributions where the trajectory is *not* deterministic given the input.

SDE Matching Bartosh et al. [4] also use an SDE to model time-series. Their method is quite different from ours. Their score function estimator relies on a network mapping from noise to latent state (their Section 4.2, Eqs. 17-20) to parameterize an auxiliary ODE that becomes part of the SDE construction. They generally require invertible parameterizations (*e.g.*, Real NVP [15], affine maps) for which the log-determinant of the Jacobian matrix is efficiently computable, restricting their architectural choices. In contrast, we do not need an auxiliary ODE nor noise-to-latent mapping, and we can use more flexible architectures to directly learn scores.

Probabilistic Forecasting with Stochastic Interpolants See Section G.

M.3 Video Generative Models

There is a rich body of work in video generative models. Generally, the strategy is to generate groups of frames autoregressively [28]. Within this paradigm, there is work on efficiently compressing the frame context [72, 71, 7]; architectural improvements [41]; noising schedule [8, 58]. Fundamentally, they *all stay within the noise-to-data diffusion paradigm*: we introduce a new paradigm of a continual SDE data-to-data bridge. Furthermore, they largely treat videos as discrete-time signals, rather than our continuous-time treatment.

Our contribution is on the probabilistic modeling side, and therefore orthogonal to these engineering improvements; it could, in principle, be applied jointly with many of them. We accordingly design our experiments to isolate methodological gains via controlled ablations of our own method, rather than to chase SOTA against models that differ along multiple architectural and scaling axes.

M.4 Diffusion Bridges

See Section F for detailed comparison. The popular denoising diffusion models/score-based generative models [56, 26] are special cases of denoising diffusion bridge models [75]. While these works only account for bridges between two endpoints (*e.g.*, noise to data), we extend the bridging methodology to the non-Markovian (any-subset, non-causal) case. Another difference between these

works and ours is that they derive their method through Anderson's time-reversal [3], while we only deal with the forward-time process via a change-of-measure to force it to generate data; this results in slightly different parameterizations of the score function.

There are also works that have a similar formulation of forward-in-time diffusion bridges [39, 48, 49]. [39] actually does consider non-Markovian stochastic processes in their initial general formulation, but they unfortunately do not take this idea further. Ultimately, they resort to matching individual marginals $\prod_1^L p(x_{t_i})$, rather than the finite-dimensional marginals $p(x_{t_1}, \ldots, x_{t_L})$. Furthermore, their work only considers *two-endpoint* bridges, *i.e.*, they only care about the sample at $t = 0$ and $t = 1$, and completely ignores time series data that could be modeled by an SDE.

M.5 Autoregressive Models

Continuous-Space Autoregressive Models Our method also is related to a line of work that uses diffusion models to parameterize the next-token prediction in continuous-space autoregressive models (*e.g.*, image generation) [37, 27]. However, we have a few key differences: (1) Our method operates in continuous time, whereas the previous methods conceptually run in discrete time with respect to the autoregressive generation (*e.g.*, they predict a fixed number of tokens at fixed "grid" points). This means that our method can predict arbitrarily many points in the time series, whereas the previous ones cannot. (2) Our method uses diffusion bridges to parameterize the model, which is conceptually closer to how certain time series (*e.g.*, dynamical systems) operate. Whereas previous methods transform noise into data, we have the inductive bias that we really only need to make some small perturbations to the previous data to get the next data point. (3) The time index of our sampling process also corresponds to the physical time index of the generated time series data, since we have the property that intermediate distributions generated by the SDE adhere to the data distribution.

Any-Order and Any-Subset Autoregressive Models Our method is related to any-order and any-subset autoregressive models [54, 23], as we can infill trajectories and condition on arbitrary subsets of the trajectory. However, our method advances the paradigm in a few substantial ways: (1) We operate in continuous time. (2) We unearth a novel parameterization of any-subset autoregressive rollouts as diffusion bridge modeling. (3) Any-subset autoregressive models typically operate on discrete data, not continuous data.

M.6 Meta Flow Maps

Meta Flow Maps [50] are a recently introduced framework that touches on some similar ideas: namely, conditioning differential equation-based generative models on some intermediate point in the trajectory. That is, they generate samples from the posterior $p_{1|t}(|x_t)$, where 1 is the time at which we get data, and t is an intermediate timestep we want to condition on.

However, we have a few crucial differences: (1) When simulating the conditional flow, their method does not actually hit the intermediate trajectory point x_t that the model was conditioned on: it only produces samples from the posterior $p_{1|t}(|x_t)$. Therefore, the modeling choice is fundamentally different from the dynamics it hopes to simulate, and lacks interpretability when the paths are examined. In contrast, our method is specifically trained to hit all the conditioning points over the course of the trajectory, as our generative time index is actually the same as the time series index. This is a consequence of our choice to model with a stochastic diffusion bridge, rather than their deterministic flow. (2) Although they briefly mentioned time series data and multiple conditioning inputs, they did not further implement or explore the connection to autoregressive models, let alone the continuous time or any-subset variants.

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: We describe our method, its relation to previous work, and results in the abstract and introduction.

   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 this in the Conclusion section, and in the results.

   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: See appendix, *e.g.*, Section C, B.2, A.3.

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: See appendix, *e.g.*, Section J.2, J.5.

   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]

See https://github.com/gabeguo/abc_diffusion.

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: See appendix, *e.g.*, Section J.2, J.5.

   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: [Yes]

   See Section 6. We have large sample size.

   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: See appendix, *e.g.*, Section J.2.

   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: We adhered to the 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: [N/A]

    Justification: Beyond the general risks of generative AI, there are not societal impacts in particular we feel that were unique to our work. Furthermore, the contribution of the paper is primarily theoretical, and the numerical experiments are done at medium-scale: not large enough to have any disinformation campaigns, etc.

    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: At our scale, we don't have models that would be misused in such a way.

    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: We own our assets, and give citations to those we don't own. The ones we don't own are open-source.

    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: Code and data will be released upon acceptance. We did document them well.

    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 human subjects.

    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 human subjects.

    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: We used LLMs to assist in code writing and paper writing, but it's not a part of the actual methodology.

    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.