ACCELERATED SEQUENTIAL FLOW MATCHING: A BAYESIAN FILTERING PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential probabilistic inference from streaming observations requires tracking distributions over future trajectories as new observations arrive. Although diffusion models can effectively capture complex distributions, their generation in streaming settings typically requires repeatedly transporting a non-informative source such as Gaussian noise to the target distribution. This noise-to-target transport incurs substantial inference latency and makes efficient high-fidelity sampling challenging. In this work, we introduce *Sequential Bayesian Flow Matching*, a framework inspired by Bayesian filtering. By learning a probability flow that transports the posterior distribution from one time step to the next time step conditioned on new observations, it mirrors the recursive structure of Bayesian belief updates. Crucially, as consecutive posteriors share common structure, the sequential flow only needs to learn the distributional update which enables substantially faster sampling than naïve resampling from scratch. Across scientific forecasting tasks spanning accelerator beam spill dynamics, fluid dynamics, and weather forecasting, as well as decision-making benchmarks, our method achieves performance competitive with full-step diffusion on distributional metrics while using 5-20 times fewer sampling steps, substantially reducing inference latency.

1 INTRODUCTION

Modeling the evolution of real-world dynamic systems from streaming observations is a fundamental problem, with applications in weather forecasting (Kalnay, 2003; Lorenz, 2017), fluid dynamics modeling (Bewley, 2001), magnetic plasma control (Degrave et al., 2022), and video forecasting in autonomous systems (Yang et al., 2024). The inherent stochasticity of these dynamics makes deterministic prediction insufficient, as point estimates can average over multiple plausible futures and produce physically invalid or blurred predictions. As a result, reliable monitoring and planning require continuously tracking a predictive distribution over multi-step future trajectories as the system evolves and new observations arrive. Classical approaches such as Kalman filtering (Kalman, 1960) and Bayesian filtering (Gordon et al., 1993; Smith, 2013) explicitly leverage the system dynamics to propagate posterior distributions of future trajectories over time, but they typically require explicit dynamic models and often struggle to model high-dimensional distributions.

Probability flow-based models, such as diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) and flow matching (Lipman et al., 2023; Liu et al., 2023), have recently become powerful frameworks for modeling high-dimensional, multi-modal distributions and generating high-fidelity samples. This leads to a growing interest in applying flow models for probabilistic time-series and sequential applications, ranging from forecasting and state estimation to planning and control (Chen et al., 2024; Gao et al., 2023; Janner et al., 2022; Wei et al., 2024). However, high-quality generation typically requires iteratively transporting samples from a non-informative source distribution, such as Gaussian noise, to the target distribution over tens to hundreds of sampling steps. Generating this noise-to-target transport with only a few function evaluations often degrades sample quality (Salimans & Ho, 2022; Liu et al., 2023)

This sampling cost becomes particularly problematic in real-time streaming settings. Existing approaches typically re-sample future trajectories from scratch to incorporate new observations, incurring prohibitive inference latency (Janner et al., 2022; Chen et al., 2024; Zhou et al., 2025). The problem becomes more acute when multiple samples are needed to characterize the trajectory distri-

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

Table 1: Examples of sequential probabilistic inference $p(x_t|z_{\leq t})$. Here x_t is the abstract predictive variable, z_t is the observation and s_t is the physical state.

Task	Predictive Variable	Observation	Inference Goal
Forecasting	$x_t = s_{t+1:t+H}$	$z_t = s_t$	$p(s_{t+1:t+H}\|s_{\leq t})$
Planning	$x_t = (s_{t+1:t+H}, a_{t:t+H-1})$	$z_t = (s_t, a_{t-1})$	$p(s_{t+1:t+H}, a_{t:t+H-1}\|s_{\leq t}, a_{\leq t-1})$
State Estimation	$x_t = s_t$	$z_t \sim p(z_t\|s_t)$	$p(s_t\|z_{\leq t})$

bution. Recent efforts seek to mitigate this cost through heuristics such as asynchronous denoising or the warm-starting from historical predictions (Janner et al., 2022; Duan et al., 2025; Li et al., 2026; Wei et al., 2025; Høeg et al., 2025). However, these approaches emphasize efficient inference pipelines rather than accurate posterior modeling, which can compromise long-horizon predictive performance (see discussion in Sections 3.3 and 5).

In this paper, we propose **Sequential Bayesian Flow Matching** (SBFM), a flow-based framework that uses a probability flow ODE to track the evolution of posterior distributions, inspired by the recursive structure of Bayesian filtering. Instead of repeatedly transporting from a non-informative source to each updated posterior, SBFM directly transports the previous posterior $p(x_{t-1}|z_{\leq t-1})$ to the updated posterior $p(x_t|z_{\leq t})$. Our key insight is that *temporal continuity* makes consecutive posteriors closely aligned, providing a substantially more informative starting point than Gaussian scratch and enabling accurate sampling with far fewer flow steps. SBFM maintains a set of particles representing the posterior and evolves them as new observations arrive, progressively moving particles toward regions consistent with the updated posterior and filtering out inconsistent modes, as illustrated in Figure 2a. This particle-based view is closely related to particle flow filtering (Daum et al., 2010; Daum & Huang, 2011; Khan & Ulmke, 2014), but our flow is learned directly from data rather than derived from explicit system dynamics and observation models.

However, directly learning posterior-to-posterior flows from raw data requires some caution. A tempting idea is to train a flow matching model to transport between successive states (x_{t-1}, x_t) from the observed trajectory conditioned on the observations $z_{\leq t}$. But this construction does not faithfully realize Bayesian filtering: once conditioned on $z_{\leq t}$, the state x_{t-1} taken from an observed trajectory is effectively a sample from $p(x_{t-1}|z_{\leq t})$, not from the posterior $p(x_{t-1}|z_{\leq t-1})$ before assimilating the new observation at time t. This distributional mismatch is discussed formally in Appendix A and ablated empirically in Appendix C.3.

To address this, we propose a simple pretraining-finetuning strategy that adapts a pretrained Gaussian-to-posterior flow model to learn posterior-to-posterior transports. We use the pretrained model to generate source samples from the previous posterior $p(x_{t-1}|z_{\leq t-1})$, and finetune SBFM (initialized from the pretrained weights) to transport these samples to the ground-truth samples x_t. This strategy echoes the common practice of finetuning pretrained diffusion models for sampling acceleration, such as progressive diffusion distillation and consistency distillation (Salimans & Ho, 2022; Song et al., 2023), although our goal is to learn sequential belief updates rather than to compress the original Gaussian-to-target sampling path.

We primarily evaluate our method on scientific applications. As the primary scientific driver, we consider beam spill dynamics in the Mu2e particle accelerator system (Whitbeck et al., 2025), where particles are extracted over time and the beam intensity evolves stochastically, requiring real-time, ultra-low-latency monitoring. Our method improves distributional accuracy in forecasting future beam intensity while substantially reducing inference latency through fewer sampling steps. Similar efficiency gains are observed on fluid dynamics and weather forecasting tasks. We further demonstrate the applicability of SBFM to planning and control settings, where it likewise enables low-latency closed-loop inference.

2 Preliminary: Flow Matching and Diffusion Models

Flow matching is to learn the velocity of an ordinary differential equation (ODE) $\frac{d}{d\tau}x(\tau) = v(x(\tau), \tau)$ $(0 \leq \tau \leq 1)$ that transports between the source distribution $x(1) \sim p_1$ and the target distribution $x(0) \sim p_0$ (Lipman et al., 2023; Liu et al., 2023). To avoid confusion with physical time t, throughout the paper we use the symbol τ to denote the virtual time variable (or noise level) in flow matching. To construct such velocity field $v(x(\tau), \tau)$, we define an in-

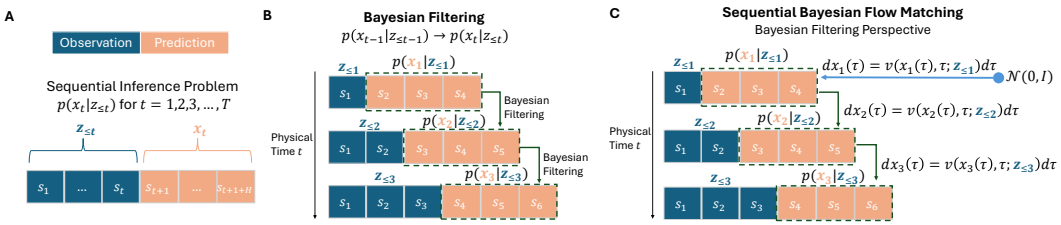


Figure 1: **A**: Streaming forecasting in sequential probabilistic inference framework. **B**: Bayesian filtering framework refines previous particles (samples) based on latest observations to obtain new particles. **C**: Following the idea of Bayesian filtering, SBFM leverages a probability flow to recursively transport from previous belief $p(x_{t-1}|z_{\le t-1})$ to current belief $p(x_t|z_{\le t})$.

terpolation path $x(\tau) = \alpha(\tau)x(0) + \sigma(\tau)x(1)$, where $\alpha(\tau), \sigma(\tau)$ are coefficients satisfying endpoint constraints $\alpha(0) = \sigma(1) = 1$ and $\alpha(1) = \sigma(0) = 0$. By defining conditional velocity $\dot{x}(\tau) = \dot{\alpha}(\tau)x(0) + \dot{\sigma}(\tau)x(1)$, it is shown that the velocity $v(x(\tau), \tau)$ can be expressed by $v(x(\tau), \tau) = \mathbb{E}(\dot{x}(\tau)|x(\tau))$, and can be learned by a neural network $v_\theta(x(\tau), \tau)$ through optimization: $\min_\theta \mathbb{E}_{(x(0),x(1))\sim\pi,\tau\sim\text{Uniform}(0,1)}\|v_\theta(x(\tau), \tau) - \dot{x}(\tau)\|^2$, where π is a coupling of p_0, p_1, i.e., π represents a joint distribution of $(x(0), x(1))$ whose marginals are p_0, p_1. To sample from target distribution p_0, we solve the ODE $dx(\tau)/d\tau = v_\theta(x(\tau), \tau)$ starting from source distribution $x(1) \sim p_1$ that is easy to sample from. The probability flow ODE counterpart of Diffusion models can be viewed as a special case of flow matching with $p_1 = \mathcal{N}(0, I)$ (Song et al., 2021b).

3 Sequential Bayesian Flow Matching

3.1 Problem Setup and Bayesian Filtering

We study a sequential probabilistic inference problem, i.e., modeling the evolving posterior distributions with new observations arriving over time. Formally, let x_t be a random variable to be predicted at the physical time t, and let z_t be the observation received at time t. Given streaming observations $z_{\le t} := (z_1, z_2, ..., z_t)$, the goal is to infer the posterior distribution $p(x_t|z_{\le t})$, over $t = 1, 2, ..., T$.

Many streaming tasks can be viewed as special instances of this formulation. A particularly interesting case we consider is when x_t represents a *long-horizon future trajectory*. For example, let s_t denote the real system state at time t. In online forecasting, the task is to infer H-step future states $x_t = s_{t+1:t+H}$ with historical states $z_{\le t}$ where $z_i = s_i$. In planning with generative models (Janner et al., 2022), the goal is to infer both future states and the actions to take $x_t = (s_{t+1:t+H}, a_{t:t+H-1})$ conditioned on historical states and actions $z_{\le t}$ where $z_i = (s_i, a_{i-1})$. Table 1 summarizes these representative examples.

Bayesian filtering. Bayesian filtering provides a principled framework to recursively update posterior distribution $p(x_t|z_{\le t})$ using new accumulated observations $z_{\le t}$. Concretely, by Bayesian rule, $p(x_t|z_{\le t})$ satisfies the key recursive formula (*note that the process may not be Markovian*):

$$\boldsymbol{p(x_t|z_{\le t})} \propto p(z_t|x_t, z_{\le t-1})p(x_t|z_{\le t-1}), \tag{1}$$

$$p(x_t|z_{\le t-1}) = \int p(x_t|x_{t-1}, z_{\le t-1})\boldsymbol{p(x_{t-1}|z_{\le t-1})}dx_{t-1}. \tag{2}$$

Equations 1,2 highlight a key structural property: the current posterior $p(x_t|z_{\le t})$ can be obtained by propagating the previous posterior $p(x_{t-1}|z_{\le t-1})$ via the dynamics $p(x_t|x_{t-1}, z_{\le t-1})$ and a correction using new observation z_t. This exact formulation is often intractable due to unknown system dynamics. We abstract this process as a distributional-level operator:

$$p(x_t|z_{\le t}) = \text{Bayesian_filtering}(p(x_{t-1}|z_{\le t-1}); z_{\le t}). \tag{3}$$

In practice, posteriors $p(x_t|z_{\le t})$ are represented by a set of particles (samples) $\{x_t^{(i)}\}_{i=1}^N$. Particle filtering provides a concrete realization of the recursive update in Eq. 3 by evolving this set of particles over time. For example, particles are propagated according to system dynamics and adjusted based on new observations through weighting and resampling (Gordon et al., 1993; Smith, 2013).

3.2 Sequential Inference with Flow-based Models: Existing Practices

A common approach to posterior inference of $p(x_t|z_{\leq t})$ is to treat each time step as a conditional generation problem and model $p(x_t|z_{\leq t})$ with a flow-based generative model. Specifically, the model transports samples from a fixed base distribution (e.g., $\mathcal{N}(0, I)$) to the target distribution $p(x_t|z_{\leq t})$ conditioned on $z_{\leq t}$. Concretely, we consider the following ODE:

$$\frac{d}{d\tau}x_t(\tau) = v(x_t(\tau), \tau; z_{\leq t}), \qquad x_t(0) \sim p(x_t|z_{\leq t}), \quad x_t(1) \sim \mathcal{N}(0, I). \tag{4}$$

where $\tau \in [0, 1]$ denotes the flow-matching time and t denotes the physical time step. Learning the velocity field and solving the ODE allow us to sample from $p(x_t \mid z_{\leq t})$.

To deploy such a model for sequential inference, one typically applies the conditional generative model repeatedly over physical time t. This receding-horizon paradigm is widely used in time-series forecasting (Box et al., 2015) and decision-making systems (Garcia et al., 1989). For flow-based models, however, this naive strategy requires restarting sampling and solving the ODE in Eq. 4 from scratch whenever a new observation arrives, often using tens or more solver steps (Janner et al., 2022). Naively reducing the number of function evaluations often degrades sample quality (Salimans & Ho, 2022; Liu et al., 2023). This incurs substantial latency and may fail to keep pace with the observation rate in real-time systems. This issue is further amplified when multiple particles are needed to represent the distribution over future trajectories.

3.3 Sequential Bayesian Flow Matching

Motivated by the recursive structure of Bayesian filtering, we propose *Sequential Bayesian Flow Matching* (SBFM), a simple framework that directly parameterizes a data-driven analogue of the filtering process in Eq. 3 as a probability flow. The key idea is to replace the non-informative Gaussian source with the previous posterior $p(x_{t-1}|z_{\leq t-1})$ in the flow ODE:

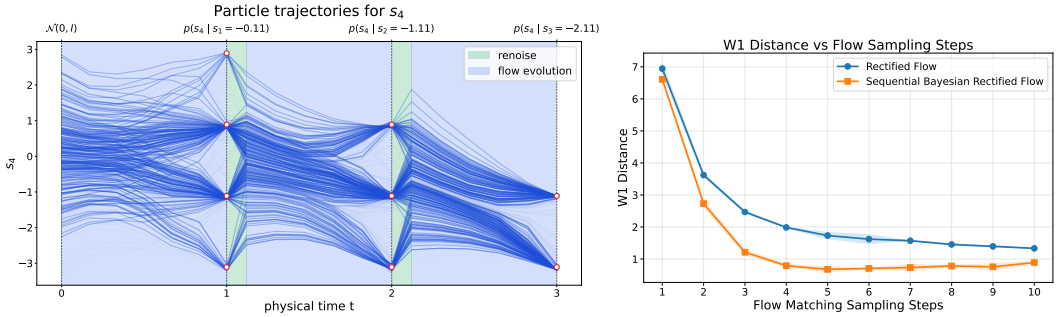
$$\frac{d}{d\tau}x_t(\tau) = v(x_t(\tau), \tau; z_{\leq t}), \qquad x_t(0) \sim p(x_t|z_{\leq t}), \quad \boldsymbol{x_t(1) \sim p(x_{t-1}|z_{\leq t-1})}. \tag{5}$$

By solving this ODE from $\tau = 1$ to 0, the flow model transports from the previous posterior to the current one. Figure 1 C illustrates the idea of SBFM.

Temporal continuity enables efficient sampling. In temporally coherent systems, consecutive posterior distributions are often closely aligned. The previous posterior therefore provides a more informative source distribution than an independent Gaussian, such that the flow only needs to model the change between consecutive posterior distributions rather than reconstructing the posterior from scratch. This informative and more-aligned source distribution reduces the transport gap, leading to smoother transport dynamics that are easier to learn and more efficient inference with fewer flow sampling steps (Ren et al., 2025; Scholz & Turner, 2025; Park et al., 2024).

Simple but not merely heuristic. While SBFM is simple in form, it differs from existing warm-start methods by directly learning the posterior update rather than applying an ad hoc reuse strategy. One line of work perturbs previous samples and denoises them under new observations using the original diffusion model (Janner et al., 2022; Duan et al., 2025; Li et al., 2026), but that model is still trained for Gaussian-to-target generation, not posterior-to-posterior transport. Another line of work trains asynchronous denoising diffusion models, which maintain future trajectories at different noise levels along the prediction horizon and progressively denoise them as new observations arrive (Wei et al., 2025; Høeg et al., 2025). However, because future states are intentionally kept partially denoised, the resulting estimates remain noisy. In contrast, SBFM learns the posterior-to-posterior flow induced by new observations. This also leads to a different particle-based mechanism for distribution modeling.

Distribution modeling. Modeling the full posterior distribution rather than a single point estimate is crucial for uncertainty quantification and risk-sensitive downstream forecasting and decision making Ghahramani (2015); Gneiting & Katzfuss (2014); Mesbah (2016). Standard diffusion models regenerate a particle set $x_t^{(i)} i = 1^N$ from Gaussian noise after each new observation to represent $p(x_t|z \leq t)$ (Rasul et al., 2021; Yan et al., 2021). SBFM instead maintains and evolves a set of particles $\{x_t^{(i)}\}_{i=1,2,\ldots,N}$ over time. When a new observation z_{t+1} arrives, each previous particle $x_t^{(i)}$ is evolved according to the flow ODE Eq. 5 conditioned on $z_{\leq t+1}$, resulting in an updated particle set $\{x_{t+1}^{(i)}\}_{i=1,2,\ldots,N}$ that approximates new posterior $p(x_{t+1}|z_{\leq t+1})$. Figure 2a illustrates the



(a) Particle evolution of $p(s_4|s_t)$ by SBFM

(b) Quantitative comparison

Figure 2: Forecasting on a bimodal Gaussian random walk $p(s_{t+1}|s_t) = \frac{1}{2}\mathcal{N}(s_t + 1, 10^{-6}) + \frac{1}{2}\mathcal{N}(s_t - 1, 10^{-6})$, with $s_1 \sim \mathcal{N}(0, 10^{-4})$. We infer $p(s_{t+1:t+5}|s_t)$ for $t = 1, \ldots, 4$, where each marginal $p(s_{t+\Delta}|s_t)$ has $(\Delta + 1)$ modes. (a) Evolution of 500 SBFM particles for s_4. Upon each observation s_t, SBFM transports particles from $p(s_4|s_{t-1})$ to $p(s_4|s_t)$ via Eq. 5. Red circles denote ground-truth modes, and inconsistent modes are progressively filtered out. (b) Average Wasserstein distance between generated and ground-truth distributions versus sampling steps. SBFM outperforms the full-step standard flow model with only three steps. Training details are in Appendix B.

particle evolution of SBFM on a synthetic dataset. Particles are moved away from low-probability regions toward regions consistent with the new observations, and old modes inconsistent with new observations are progressively filtered out.

Connection to particle filtering. This particle-evolution view is closely connected to particle-based Bayesian filtering. The classic particle filtering, or Sequential Monte Carlo Gordon et al. (1993); Smith (2013), also maintains a set of particles to approximate the posterior. The particles are propagated via system dynamics, re-weighted according to new observations and then re-sampled. The importance weighting is known to suffer from particle degeneracy in high-dimensional settings, where the posterior mass concentrates on only a few particles and most particles receiving negligible weights are discarded (Smith, 2013). Particle flow filtering is later developed to mitigate this issue by directly constructing a probability flow that bridges between posteriors (Daum et al., 2010; Daum & Huang, 2011; Khan & Ulmke, 2014). Particles are allowed to move to high-density regions rather than being discarded, similar to the mechanism of SBFM shown in Figure 2a. These classic approaches derive flow dynamics from explicit system and observation models, while our method learns the velocity field directly from data without requiring knowledge of the underlying system.

3.4 Training and Inference Algorithms

We discuss how to learn SBFM (Eq. 5) via flow matching. The flow matching objective requires pairs of samples (x_{t-1}, x_t) as well as condition $z_{\leq t}$. Suppose we have an offline dataset of state-observation trajectories $\{(x^{*,m}_{1:T}, z^{*,m}_{1:T})\}_{m=1,2,\ldots,M}$. For notational simplicity, we omit the trajectory index, denoting a general ground-truth trajectory from the dataset as $(x^*_{1:T}, z^*_{1:T})$.

Training Data Construction. A naïve approach is to draw successive ground-truth states (x^*_{t-1}, x^*_t) together with observations $z^*_{\leq t}$ from the offline dataset, and train a conditional flow matching to transport x^*_{t-1} to x^*_t conditioned on $z^*_{\leq t}$. However, this construction introduces a fundamental distributional mismatch. Since the data $(x^*_{t-1}, x^*_t, z^*_{\leq t})$ are drawn from the joint distribution $p(x_{t-1}, x_t, z_{\leq t})$, the source sample x^*_{t-1} exposed to the conditional flow model $v_\theta(\cdot|z^*_{\leq t})$ has marginal distribution $p(x_{t-1}|z^*_{\leq t})$. However, Bayesian filtering recursion requires the previous posterior as $p(x_{t-1}|z^*_{\leq t-1})$, without observing z^*_t. As a result, the learned flow matching is not aligned with the posterior-to-posterior flow ODE Eq. 5. A formal analysis of this mismatch is in Appendix A and this training approach empirically leads to a degraded performance shown in Appendix C.3.

To address this mismatch, we use a simple pretraining-finetuning framework that adapts a pretrained diffusion or flow model to learn the posterior-to-posterior transport. In the pretraining stage, we train a standard flow-based model v_{pre} that transports Gaussian noise to x^*_t conditioned on $z^*_{\leq t}$ (Eq. 4). In the finetuning stage, we freeze the pretrained model and use it to generate source samples that match the filtering recursion. Specifically, we (a) draw a ground-truth state-observation pair $(x^*_t, z^*_{\leq t})$, and

Algorithm 1 Sequential Bayesian Flow Matching Finetuning

Require: Ground-truth x_t, context $z_{\leq t}$, renoise level τ_{renoise}, and a pretrained flow model v_{pre}

1: Generate $x_{t-1}^{\text{pre}} \sim p_{\text{pre}}(x_{t-1}|z_{\leq t-1})$ using v_{pre} (solve ODE Eq. 4)
2: Renoise $\tilde{x}_{t-1} = \alpha(\tau_{\text{renoise}}) \cdot x_{t-1}^{\text{pre}} + \sigma(\tau_{\text{renoise}}) \cdot \mathcal{N}(0, I)$
3: Sample interpolation time $\tau \sim \text{Uniform}(0, 1)$
4: Compute $x_t(\tau) = \alpha(\tau)x_t + \sigma(\tau)\tilde{x}_{t-1}$
5: **return** loss $L(\theta) = \|v_\theta(x_t(\tau), \tau; z_{\leq t}) - \dot{x}_t(\tau)\|^2$

Algorithm 2 Sequential Bayesian Flow Matching Inference

Require: Pretrained flow model v_{pre}, SBFM v_θ finetuned by Algorithm 1, initial observation z_1, particle size N

1: Initialize particle set $\{x_1^{(i)}\}_{i=1,2,\ldots,N}$ with $x_1^{(i)} \overset{\text{i.i.d.}}{\sim} p_{\text{pre}}(x_1|z_1)$ (solve ODE Eq. 4)
2: **for** $t = 2$ to T **do**
3: Receive new observation z_t
4: **for** $i = 1$ to N **do** (run in parallel)
5: $\tilde{x}_{t-1}^{(i)} = \alpha(\tau_{\text{renoise}}) \cdot x_{t-1}^{(i)} + \sigma(\tau_{\text{renoise}}) \cdot \mathcal{N}(0, I)$
6: Solve $dx_t^{(i)}(\tau)/d\tau = v_\theta(x_t^{(i)}(\tau), \tau; z_{\leq t})$ from $x_t^{(i)}(1) = \tilde{x}_{t-1}^{(i)}$, and obtain $x_t^{(i)}(0)$
7: **end for**
8: Obtain new particle set $\{x_t^{(i)}\}_{i=1,2,\ldots,N}$ by assigning $x_t^{(i)} = x_t^{(i)}(0)$
9: **end for**

(b) sample $x_{t-1}^{\text{pre}} \sim p_{\text{pre}}(x_{t-1}|z_{\leq t-1}^*)$ from the pretrained model conditioned only on the previous observations. We then initialize SBFM v_θ with pretrained model weights and finetune it to transport x_{t-1}^{pre} to x_t^* conditioned on new observations $z_{\leq t}^*$. This makes the source samples approximately follow the previous posterior, so the finetuned model learns the desired posterior update.

Renoising mechanism. As the pretrained model only approximates the true distribution, train–test mismatch in the source distribution can accumulate over successive belief updates. We mitigate this with controlled renoising during both training and inference: $\tilde{x}_{t-1} := \alpha(\tau_{\text{renoise}})x_{t-1} + \sigma(\tau_{\text{renoise}})\mathcal{N}(0, I)$, where τ_{renoise} is a tunable hyperparameter. The renoise level trades off preserving historical posterior information against accommodating uncertainty. When $\tau_{\text{renoise}} = 1$, the source becomes Gaussian and recovers restart-from-scratch sampling. When $\tau_{\text{renoise}} = 0$, the method fully relies on the previous posterior and is more brittle to error accumulation. Appendix C.3 studies this trade-off by tuning τ_{renoise}.

Flexible particle set size. The number of particles N controls the granularity of the posterior representation and can be adjusted according to the latency budget. The particle set can be expanded by independently re-noising existing particles ($x_t^{(i)} \to \{\tilde{x}_t^{(i,1)}, \tilde{x}_t^{(i,2)}, \ldots\}$), or by drawing fresh particles from the pretrained model when additional latency is acceptable. In our experiments, we keep N fixed across sequential updates.

Algorithms 1 and 2 summarize the finetuning and inference pipelines. At inference, the initial particles $x_1^{(i)}$ are generated by the pretrained flow model since no previous posterior is available.

4 Related Works

Flow-based models for trajectory generation. Diffusion and flow matching models have been widely used for trajectory generation in an offline setting. Existing approaches mainly differ in how denoising is scheduled across the prediction horizon, including full-sequence denoising (Li et al., 2022; Ho et al., 2022), autoregressive denoising (Hoogeboom et al., 2021; Rasul et al., 2021) and asynchronous denoising (Ruhe et al., 2024; Wu et al., 2023; Chen et al., 2024). Asynchronous denoising allows assigning and denoising variables of different noise levels along the prediction axis. These methods are primarily designed for static trajectory generation. When applied in streaming settings, they rely on repeated sampling from scratch.

Efficient sampling of flow-based models. Prior work accelerates sampling by applying advanced numerical solvers (Lu et al., 2022; Zhang & Chen, 2023) or distillation-based methods that compress

Table 2: Results of beam spill forecasting. Prediction horizon $H = 300$.

Method	NFE	Energy Score ↓	RMSE ↓
Autoregressive	H	$1.2509_{\pm 0.0013}$	$2.199_{\pm 0.021}$
Diffusion	5	$0.0196_{\pm 0.0001}$	$\mathbf{0.051}_{\pm 0.002}$
Rectified Flow	5	$0.0231_{\pm 0.0003}$	$0.072_{\pm 0.004}$
Diffusion	1	$0.0220_{\pm 0.0001}$	$0.062_{\pm 0.002}$
Rectified Flow	1	$0.0302_{\pm 0.0007}$	$0.115_{\pm 0.009}$
MeanFlow	1	$0.1780_{\pm 0.0022}$	$0.849_{\pm 0.280}$
Consistency Model	1	$0.0269_{\pm 0.0007}$	$0.155_{\pm 0.044}$
Asynchronous Diffusion	1	$0.0336_{\pm 0.0009}$	$0.358_{\pm 0.011}$
Warm-start Diffusion	1	$0.0197_{\pm 0.0002}$	$0.053_{\pm 0.002}$
Warm-start Diffusion	2	$0.0203_{\pm 0.0001}$	$0.054_{\pm 0.001}$
Warm-start Rectified Flow	1	$0.0236_{\pm 0.0001}$	$0.076_{\pm 0.001}$
Warm-start Rectified Flow	2	$0.0239_{\pm 0.0002}$	$0.083_{\pm 0.001}$
SBFM-Rectified Flow	1	$0.0179_{\pm 0.0003}$	$0.054_{\pm 0.001}$
SBFM-Rectified Flow	2	$\mathbf{0.0170}_{\pm 0.0002}$	$\mathbf{0.050}_{\pm 0.002}$

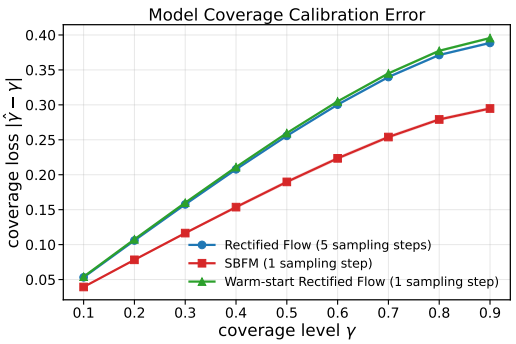


Figure 3: Predictive coverage error on beam-spill forecasting.

the sampling path of a teacher model into a fewer-step student model (Salimans & Ho, 2022; Yin et al., 2024; Geng et al., 2023). More recently, flow map matching methods directly learn the solution operator of the flow ODE and enable few-step generation (Song et al., 2023; Boffi et al., 2025; Geng et al., 2025a). However, the training objective is known to be unstable to optimize (Lu & Song, 2025; Geng et al., 2025b). In our streaming experiments, consistency models (Geng et al., 2025c) and MeanFlow (Geng et al., 2025a) were difficult to optimize reliably and generalized poorly. In contrast, SBFM leverages the previous posterior to construct a more informative source distribution rather than compressing the flow path into one step. Extending such one-step generative models to streaming tasks is a nontrivial direction, and could potentially be combined with our framework in future work.

Bayesian inference of diffusion models. Diffusion priors have been studied for Bayesian inverse problems, where the goal is to infer $p(x|y)$ from measurements $y = f(x)$ and a diffusion prior $p(x)$ (Cardoso et al., 2024; Chung et al., 2023; Dou & Song, 2024; Daras et al., 2024). These approaches mainly focus on static inference with single-shot measurements. While recent work has begun to explore flow-based data assimilation (Chen et al., 2025), sequential probabilistic inference with diffusion or flow priors remains underexplored, especially in low-latency streaming settings. Crucially, most approaches rely on likelihood-based guidance, requiring explicit, often linear, observation models and system dynamics that are unavailable in the real world.

5 Experiment

We evaluate SBFM on scientific forecasting tasks with streaming observations, including beam spill dynamics (Whitbeck et al., 2025), fluid dynamics, and weather forecasting (Moro & Chamon, 2025; Rasp et al., 2024), as well as planning and control (Fu et al., 2020; Holl et al., 2020). SBFM maintains distributional accuracy while reducing inference latency and enabling faster closed-loop re-planning. We report Number of Function Evaluations (NFE), i.e., the number of first-order Euler steps used to solve the flow ODE, to measure the sampling efficiency. We use a single NVIDIA RTX 6000 Ada Generation (48 GB). Dataset and implementation details are in Appendix B.

Pretrained Model. We mainly adopt Diffusion with DDIM sampler (Song et al., 2021a) and Rectified flow (Liu et al., 2023) as the pretrained model. Note that SBFM is compatible with general flow-based models, though we evaluate it using these two representative models for demonstration.

Baselines. We consider several classes of efficient flow-based baselines: (1) *One-step diffusion*: We evaluate consistency models (Song et al., 2023) and MeanFlow (Geng et al., 2025a). For consistency models, we adopt ECT (Geng et al., 2025c), an efficient fine-tuning method for consistency distillation. (2) *Warm-start diffusion*: Following Janner et al. (2022); Duan et al. (2025), we add a controlled amount of noise to the previous prediction x_{t-1} and denoise it under the updated condition z_t using the same pretrained model. For direct comparison, we set the noise level to τ_{renoise}, matching SBFM's re-noise level. This warm-start heuristic can be viewed as an ablation of SBFM, where fine-tuning stage is removed and the pretrained model is directly applied. (3) *Asynchronous denoising diffusion*: We adopt the idea from (Wei et al., 2025; Høeg et al., 2025), training an asynchronous diffusion and maintaining a partially denoised future trajectories at inference time.

Long-horizon x_t. In our experiments, x_t typically denotes a long-horizon trajectory, e.g., $x_t = [s_{t+1}, \ldots, s_{t+H}]$. We define $\text{shift}(x_t) = [s_{t+2}, \ldots, s_{t+H}, s_{t+H}]$ and practically train SBFM to transport $p(\text{shift}(x_{t-1}) \mid z_{\leq t-1})$ to $p(x_t \mid z_{\leq t})$, aligning states at the same physical time.

Table 3: Results of fluid system and weather forecasting. Prediction horizon $H = 10$ and 12 for Burgers'equation and weather forecasting.

Method	Burgers' Equation			Weather Forecasting		
	NFE	Energy Score ↓	RMSE ↓	NFE	Temperature Energy Score ↓	Temperature RMSE ↓
Autoregressive	H	$0.133_{\pm 0.009}$	$0.253_{\pm 0.022}$	H	$2.583_{\pm 0.026}$	$3.764_{\pm 0.042}$
Diffusion	10	$\mathbf{0.100}_{\pm 0.001}$	$0.258_{\pm 0.005}$	20	$2.173_{\pm 0.084}$	$4.249_{\pm 0.162}$
Rectified Flow	5	$0.103_{\pm 0.001}$	$0.262_{\pm 0.001}$	10	$\mathbf{2.030}_{\pm 0.080}$	$3.990_{\pm 0.146}$
Diffusion	1	$0.108_{\pm 0.002}$	$0.247_{\pm 0.006}$	1	$2.849_{\pm 0.197}$	$4.974_{\pm 0.279}$
Rectified Flow	1	$0.114_{\pm 0.002}$	$0.250_{\pm 0.006}$	1	$2.608_{\pm 0.157}$	$4.509_{\pm 0.228}$
MeanFlow	1	$0.234_{\pm 0.047}$	$0.418_{\pm 0.070}$	1	$5.700_{\pm 0.017}$	$8.632_{\pm 0.030}$
Consistency Model	1	$0.107_{\pm 0.003}$	$0.252_{\pm 0.004}$	1	$2.610_{\pm 0.033}$	$6.269_{\pm 0.067}$
Asynchronous Diffusion	1	$0.264_{\pm 0.015}$	$0.559_{\pm 0.011}$	1	$3.368_{\pm 0.124}$	$6.248_{\pm 0.086}$
Warm-start Diffusion	1	$0.231_{\pm 0.013}$	$0.487_{\pm 0.033}$	1	$4.064_{\pm 0.065}$	$7.072_{\pm 0.014}$
Warm-start Rectified Flow	1	$0.218_{\pm 0.009}$	$0.475_{\pm 0.004}$	1	$3.209_{\pm 0.003}$	$6.236_{\pm 0.108}$
SBFM-Rectified Flow	1	$\mathbf{0.101}_{\pm 0.002}$	$\mathbf{0.239}_{\pm 0.005}$	1	$2.353_{\pm 0.032}$	$\mathbf{3.660}_{\pm 0.061}$

5.1 Mu2e Beam Spill Monitoring

The Mu2e experiment at Fermilab is a particle-physics experiment designed to search for new physics Whitbeck et al. (2025). In its beam spill system, charged particles are extracted over time, producing a beam intensity that evolves stochastically. The beam intensity must remain highly uniform to preserve detector live time and prevent irreversible loss of valuable physics data (Narayanan, 2022). Since the spill regulation system operates at 10 kHz with an overall feedback loop on the order of 1 ms, low-latency monitoring and prediction of the evolving beam state are crucial.

Datasets. We adopt a physics-based beam spill simulator to simulate beam intensity trajectories $s_{1:430}$ regulated by a controller over a 43ms time window Narayanan (2022). The task to infer 300-step future trajectories $s_{t+1:t+300}$ given historical states $s_{\leq t}$ and input control signals $a_{\leq t+300}$.

Distribution evaluation. We use $N = 20$ particles to represent the predictive distribution. To evaluate distributional quality, we adopt the Energy Score, a proper scoring rule widely used in probabilistic time-series forecasting (Gneiting & Raftery, 2007). Given a ground-truth trajectory $x_t = s_{t+1:t+300}$ and the model's predictive distribution p_θ, the energy score is defined as $\mathbb{E}_{\hat{x}_t \sim p_\theta} \|\hat{x}_t - x_t\|_1 - \frac{1}{2}\mathbb{E}_{\hat{x}_t, \hat{x}'_t \sim p_\theta} \|\hat{x}_t - \hat{x}'_t\|_1$. The energy score measures the absolute error but with a penalty of over-deterministic prediction. The energy score is proper, meaning its expected value is minimized when the predictive distribution matches the ground-truth distribution (Gneiting & Raftery, 2007). As a complementary pointwise metric, we also report root mean squared error (RMSE), although RMSE alone does not directly assess distributional quality.

Results. Table 2 shows that SBFM, finetuned from a pretrained rectified flow, achieves the best energy score with substantially fewer sampling steps. The autoregressive baseline accumulates errors over the 300-step horizon, while asynchronous diffusion remains inferior even after its partially denoised trajectories are fully denoised with one-step DDIM. Warm-start performs reasonably on this task but generalizes less reliably across datasets.

Figure 3 evaluates calibration using coverage loss $|\hat{\gamma} - \gamma|$, where $\hat{\gamma}$ is the empirical coverage of the central γ quantile interval. One-step SBFM consistently achieves lower coverage loss than the full-step pretrained model and warm-start heuristics, indicating better-calibrated predictive uncertainty and improved distributional quality.

Performance-latency trade-offs. Appendix C.2, Figures 4 and 5 shows that across varying sampling steps, SBFM matches or outperforms the full-step pretrained model with fewer sampling steps.

5.2 Fluid and Weather System Forecasting

Datasets. We further consider forecasting on two physical systems: (a) fluid dynamic system governed by the 1D Burgers' equation, adapted from (Wei et al., 2025). (b) real-world weather forecasting adapted from WeatherBench2 (Rasp et al., 2024). The model takes historical states $z_{\leq t} := s_{\leq t}$ as conditions and infers H-step future states $x_t := s_{t+1:t+H}$ over t. The prediction horizon is $H = 10$ for fluid system and $H = 12$ for weather forecasting.

Results. We use $N = 20$ particles and Table 3 reports the energy score and RMSE. We observe acceleration heuristics such as asynchronous denoising and warm starts underperform, while SBFM still maintains a competitive energy score with only one flow sampling step.

Table 4: Total reward on maze planning task. Planning horizon is $H = 600/800$ for Maze Medium/Large.

Method	NFE	Maze Medium ↑	Maze Large ↑
MPPI (Williams et al., 2015)	N/A	10.2	5.1
CQL (Kumar et al., 2020)	N/A	5.0	12.5
IQL (Kostrikov et al., 2022)	N/A	34.9	58.6
Diffuser	256	$121.5_{\pm 2.7}$	$123.0_{\pm 6.4}$
Diffusion Forcing	50	$149.4_{\pm 7.5}$	$159.0_{\pm 2.7}$
Diffusion Forcing	3	$110.4_{\pm 11.6}$	$67.3_{\pm 27.1}$
Diffusion Forcing	1	$22.1_{\pm 12.3}$	$2.2_{\pm 1.9}$
Asynchronous Diffusion	1	$74.2_{\pm 2.4}$	$96.1_{\pm 3.6}$
Asynchronous Diffusion	3	$78.3_{\pm 0.7}$	$103.9_{\pm 4.1}$
Warm-start Diffusion Forcing	1	$85.1_{\pm 12.6}$	$61.9_{\pm 22.1}$
Warm-start Diffusion Forcing	3	$81.8_{\pm 8.2}$	$62.3_{\pm 20.7}$
SBFM-Diffusion Forcing	1	$57.5_{\pm 29.3}$	$114.6_{\pm 49.0}$
SBFM-Diffusion Forcing	3	$\mathbf{168.6}_{\pm 11.9}$	$\mathbf{233.9}_{\pm 8.8}$

Table 5: Off-target leakage rate on smoke control task. Planning horizon is $H = 10$.

Method	NFE	Fixed Map ↓	Random Map ↓
BC (Pomerleau, 1988)	N/A	0.672	0.705
BPPO (Zhuang et al., 2023)	N/A	0.634	0.652
DiffPhyCon	600	0.545	0.375
Diffusion Forcing	10	$0.166_{\pm 0.036}$	$0.291_{\pm 0.042}$
Rectified Flow	10	$0.173_{\pm 0.078}$	$0.190_{\pm 0.020}$
Diffusion Forcing	1	$0.362_{\pm 0.068}$	$0.280_{\pm 0.019}$
Rectified Flow	1	$0.432_{\pm 0.009}$	$0.260_{\pm 0.020}$
MeanFlow	1	$0.369_{\pm 0.117}$	$0.462_{\pm 0.166}$
Consistency Model	1	$0.295_{\pm 0.013}$	$0.293_{\pm 0.002}$
Asynchronous Diffusion	60	0.337	0.346
Warm-start Diffusion Forcing	1	$0.547_{\pm 0.026}$	$0.268_{\pm 0.016}$
warm-start Rectified Flow	1	$0.291_{\pm 0.039}$	$0.286_{\pm 0.017}$
SBFM-Diffusion Forcing	1	$\mathbf{0.097}_{\pm 0.012}$	$\mathbf{0.231}_{\pm 0.006}$

5.3 Planning and Control

Datasets. We evaluate SBFM on: (a) maze planning (Fu et al., 2020), where an agent navigates around obstacles to a goal, and (b) smoke control (Holl et al., 2020; Wei et al., 2025), where smoke is steered toward a target exit in a 2D incompressible fluid governed by the Navier–Stokes equations. For smoke control, we test both fixed maps and random maps with perturbed obstacle positions. The model conditions on $z_{\leq t} := (s_t, a_{t-1})$ and predicts $x_t := (s_{t+1:t+H}, a_{t:t+H-1})$. The planning horizon is $H = 600, 800, 10$ for Maze-Medium, Maze-Large, and smoke control. It re-plans (incorporates new observations) every 50 steps for maze planning and every step for smoke control.

We use a variant of Diffusion tailored for sequence data named Diffusion Forcing as the pretrained model (Chen et al., 2024). We also include other classic baselines (Williams et al., 2015; Kumar et al., 2020; Kostrikov et al., 2022; Pomerleau, 1988; Zhuang et al., 2023), whose reported numbers are from Chen et al. (2024) and Wei et al. (2025) in the same benchmark settings.

Results. We use $N = 1$ particle. Tables 4 and 5 report results on maze planning and smoke control respectively. SBFM consistently achieves highly competitive downstream performance compared to full-step diffusion on both tasks with a small number of sampling steps.

5.4 Ablation Study

Ablation Study. The warm-start baselines can be viewed as an ablation of SBFM in which the finetuning stage is removed and the pretrained model is directly applied. We further ablate the re-noise level and the use of model-generated versus ground-truth trajectories for finetuning. Detailed results are provided in Appendix C.3.

Finetuning data efficiency. Overall SBFM requires only a small number of finetuning data to effectively generalize (see Table 6). We vary the number of finetuning trajectories while keeping the pretrained model fixed. SBFM requires only a small amount of finetuning data to generalize well and outperform standard diffusion in the few-NFE regime. Results are deferred to Appendix C.4.

6 Conclusion and Limitations

This work studies the efficient deployment of flow-based models in sequential probabilistic inference, where models are required to infer the distribution of long-horizon future trajectory and continuously assimilate streaming observations under latency constraints. We propose Sequential Bayesian Flow Matching, which transports the predictive distribution across time steps and can be viewed as parameterizing the Bayesian belief update. Experiments show that it achieves distributional metrics competitive to full-step diffusion with significantly fewer sampling steps.

A limitation of our approach is that recursive posterior-to-posterior updates may accumulate errors over time. A promising future direction is to develop adaptive re-noising strategies that regulate this accumulation, or to design criteria for when the model should reset by sampling from the Gaussian to recover from compounding errors.

AI USE STATEMENT

In this work, we used generative AI tools to assist with manuscript editing, including improving grammar, clarity, and readability. We did not use generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, provide ingredients for or assist in proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean or reformat datasets, analyze or interpret results, or draw scientific conclusions. Translation and qualitative or thematic data analysis were not applicable to this work. All AI-assisted text was manually reviewed and revised by the authors for technical accuracy, consistency with the underlying experiments and methods, and appropriate attribution. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

The code for reproducing the experiments is provided in the supplementary materials. The dataset description is provided in Appendix B.1.

REFERENCES

Thomas R Bewley. Flow control: new challenges for a new renaissance. *Progress in Aerospace sciences*, 37(1):21–58, 2001.

Nicholas Matthew Boffi, Michael Samuel Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *Transactions on Machine Learning Research*, 2025.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

Gabriel Cardoso, Yazid Janati el idrissi, Sylvain Le Corff, and Eric Moulines. Monte carlo guided denoising diffusion models for bayesian linear inverse problems. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nHESwXvxWK.

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

Siyi Chen, Yixuan Jia, Qing Qu, He Sun, and Jeffrey A Fessler. Flowdas: A stochastic interpolant-based framework for data assimilation. *arXiv preprint arXiv:2501.16642*, 2025.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=OnD9zGAGT0k.

Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *CoRR*, 2024.

Fred Daum and Jim Huang. Particle flow for nonlinear filters. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5920–5923. IEEE, 2011.

Fred Daum, Jim Huang, and Arjang Noushin. Exact particle flow for nonlinear filters. In *Signal processing, sensor fusion, and target recognition XIX*, volume 7697, pp. 92–110. SPIE, 2010.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Pierre Del Moral. Nonlinear filtering: Interacting particle resolution. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 325(6):653–658, 1997.

Zehao Dou and Yang Song. Diffusion posterior sampling for linear inverse problem solving: A filtering perspective. In *The Twelfth International Conference on Learning Representations*, 2024.

Yufei Duan, Hang Yin, and Danica Kragic. Real-time iteration scheme for diffusion policy. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11758–11764. IEEE, 2025.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Zhihan Gao, Xingjian Shi, Boran Han, Hao Wang, Xiaoyong Jin, Danielle Maddix, Yi Zhu, Mu Li, and Yuyang Bernie Wang. Prediff: Precipitation nowcasting with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:78621–78656, 2023.

Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

Zhengyang Geng, Ashwini Pokle, and J Zico Kolter. One-step diffusion distillation via deep equilibrium models. *Advances in Neural Information Processing Systems*, 36:41914–41931, 2023.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025a.

Zhengyang Geng, Yiyang Lu, Zongze Wu, Eli Shechtman, J Zico Kolter, and Kaiming He. Improved mean flows: On the challenges of fastforward generative models. *arXiv preprint arXiv:2512.02012*, 2025b.

Zhengyang Geng, Ashwini Pokle, Weijian Luo, Justin Lin, and J Zico Kolter. Consistency models made easy. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=xQVxo9dSID.

Zoubin Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553): 452–459, 2015.

Tilmann Gneiting and Matthias Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1(1):125–151, 2014.

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

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

Yangguang He, Wenhao Li, Minzhe Li, Juan Zhang, Xiangfeng Wang, and Bo Jin. Trackdiffuser: Nearly model-free bayesian filtering with diffusion model. *arXiv preprint arXiv:2502.05629*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

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

Sigmund H Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4821–4828. IEEE, 2025.

Philipp Holl, Nils Thuerey, and Vladlen Koltun. Learning to control pdes with differentiable physics. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HyeSin4FPB.

Emiel Hoogeboom, Alexey A Gritsenko, Jasmijn Bastings, Ben Poole, Rianne van den Berg, and Tim Salimans. Autoregressive diffusion models. *arXiv preprint arXiv:2110.02037*, 2021.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pp. 9902–9915. PMLR, 2022.

Simon J Julier and Jeffrey K Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pp. 182–193. Spie, 1997.

RE Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, D*, 82:35–44, 1960.

Eugenia Kalnay. *Atmospheric modeling, data assimilation and predictability*. Cambridge university press, 2003.

Muhammad Altamash Khan and Martin Ulmke. Non-linear and non-gaussian state estimation using log-homotopy based particle flow filters. In *2014 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*, pp. 1–6. IEEE, 2014.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

H Kushner. Approximations to optimal nonlinear filters. *IEEE Transactions on Automatic Control*, 12(5):546–556, 1967.

Jinhao Li, Yuxuan Cong, Yingqiao Wang, Hao Xia, Shan Huang, Yijia Zhang, Ningyi Xu, and Guohao Dai. Step: Warm-started visuomotor policies with spatiotemporal consistency prediction. *arXiv preprint arXiv:2602.08245*, 2026.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35: 4328–4343, 2022.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *11th International Conference on Learning Representations, ICLR 2023*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023.

Edward N Lorenz. Deterministic nonperiodic flow 1. In *Universality in Chaos, 2nd edition*, pp. 367–378. Routledge, 2017.

Cheng Lu and Yang Song. Simplifying, stabilizing and scaling continuous-time consistency models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LyJi5ugyJx.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Ali Mesbah. Stochastic model predictive control: An overview and perspectives for future research. *IEEE Control Systems Magazine*, 36(6):30–44, 2016.

Viggo Moro and Luiz F. O. Chamon. Solving differential equations with constrained learning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=5KqveQdXiZ.

Aakaash Narayanan. Machine learning for slow spill regulation in the fermilab delivery ring for mu2e. Technical report, Fermi National Accelerator Laboratory (FNAL), Batavia, IL (United States), 2022.

Jinseong Park, Seungyun Lee, Woojin Jeong, Yujin Choi, and Jaewook Lee. Leveraging priors via diffusion bridge for time series generation. *arXiv preprint arXiv:2408.06672*, 2, 2024.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

Stephan Rasp, Stephan Hoyer, Alexander Merose, Ian Langmore, Peter Battaglia, Tyler Russell, Alvaro Sanchez-Gonzalez, Vivian Yang, Rob Carver, Shreya Agrawal, et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6):e2023MS004019, 2024.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International conference on machine learning*, pp. 8857–8868. PMLR, 2021.

Hao Ren, Yiming Zeng, Zetong Bi, Zhaoliang Wan, Junlong Huang, and Hui Cheng. Prior does matter: Visual navigation via denoising diffusion bridge models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12100–12110, 2025.

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42818–42835. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ruhe24a.html.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Jonas Scholz and Richard E Turner. Warm starts accelerate conditional diffusion. *arXiv preprint arXiv:2507.09212*, 2025.

Adrian Smith. *Sequential Monte Carlo methods in practice*. Springer Science & Business Media, 2013.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. pmlr, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

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*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 32211–32252, 2023.

Long Wei, Peiyan Hu, Ruiqi Feng, Haodong Feng, Yixuan Du, Tao Zhang, Rui Wang, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Diffphycon: A generative approach to control complex physical systems. *Advances in Neural Information Processing Systems*, 37:4090–4147, 2024.

Long Wei, Haodong Feng, Yuchen Yang, Ruiqi Feng, Peiyan Hu, Xiang Zheng, Tao Zhang, Dixia Fan, and Tailin Wu. CL-diffphycon: Closed-loop diffusion control of complex physical systems. In *The Thirteenth International Conference on Learning Representations*, 2025.

A Whitbeck, J Berlioz, K Danison-Fieldhouse, K Hazelwood, M Khan, J Mitrevski, A Narayanan, J St John, N Tran, J Ji, et al. Fast adaptive neural control of resonant extraction at fermilab. *arXiv preprint arXiv:2511.19099*, 2025.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv preprint arXiv:1509.01149*, 2015.

Tong Wu, Zhihao Fan, Xiao Liu, Hai-Tao Zheng, Yeyun Gong, Jian Jiao, Juntao Li, Jian Guo, Nan Duan, Weizhu Chen, et al. Ar-diffusion: Auto-regressive diffusion model for text generation. *Advances in Neural Information Processing Systems*, 36:39957–39974, 2023.

Tijin Yan, Hongwei Zhang, Tong Zhou, Yufeng Zhan, and Yuanqing Xia. Scoregrad: Multivariate probabilistic time series forecasting with continuous energy-based generative models. *arXiv preprint arXiv:2106.10121*, 2021.

Jiazhi Yang, Shenyuan Gao, Yihang Qiu, Li Chen, Tianyu Li, Bo Dai, Kashyap Chitta, Penghao Wu, Jia Zeng, Ping Luo, et al. Generalized predictive model for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14662–14672, 2024.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024.

Qinsheng Zhang and Yongxin Chen. Fast sampling of diffusion models with exponential integrator. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Loek7hfb46P.

Guangyao Zhou, Sivaramakrishnan Swaminathan, Rajkumar Vasudeva Raju, J Swaroop Guntupalli, Wolfgang Lehrach, Joseph Ortiz, Antoine Dedieu, Miguel Lazaro-Gredilla, and Kevin Patrick Murphy. Diffusion model predictive control. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=pvtgffHtJm.

Zifeng Zhuang, Kun LEI, Jinxin Liu, Donglin Wang, and Yilang Guo. Behavior proximal policy optimization. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3c13LptpIph.

A TRAINING SBFM FROM RAW DATA

Here we formally justify why training a conditional flow matching from successive ground-truth states (x_{t-1}, x_t) with conditions $z_{\leq t}$ cannot implement filtering ODE Eq. equation 5, due to a mismatch of source distribution.

Goal. The goal is to train a flow following Eq. equation 5, with initial distribution $p(x_{t-1}|z_{\leq t-1})$ and target distribution $p(x_t|z_{\leq t})$.

Training data construction. Let us describe the naive training data construction protocol. Suppose we have a ground-truth trajectory $(x^*_{1:T}, z^*_{1:T})$ drawn from an offline dataset. Formally, this trajectory is a sample $(x^*_{1:T}, z^*_{1:T}) \sim p(x_{1:T}, z_{1:T})$ from the joint distribution of states and observations. Suppose we want to train the filtering from $p(x_{t-1}|z_{\leq t-1})$ to $p(x_t|z_{\leq t})$, the naive approach extract (x^*_{t-1}, x^*_t) and $z^*_{\leq t}$ from the full trajectory. This means $(x^*_{t-1}, x^*_t, z^*_{\leq t}) \sim p(x_{t-1}, x_t, z_{\leq t})$, i.e., they are drawn jointly from the marginal distribution.

Based on these settings, we can argue the following straightforward proposition, which shows the flow ODE's initial distribution is $p(x_{t-1}|z_{\leq t})$ rather than the desired $p(x_{t-1}|z_{\leq t-1})$.

Suppose a conditional flow matching is trained by minimizing

$$\theta^* = \arg\min_\theta \mathbb{E}_{(x^*_{t-1}, x^*_t, z^*_{\leq t}) \sim p(x_{t-1}, x_t, z_{\leq t})} \mathbb{E}_{\tau \sim \text{Unif}(0,1)} \left\| v_\theta(x_t(\tau), \tau; z^*_{\leq t}) - \dot{x}_\tau(t) \right\|^2, \tag{6}$$

where $x_t(\tau) := \alpha_\tau x^*_{t-1} + \sigma_\tau x^*_t$ and $\dot{x}_t(\tau) = \dot{\alpha}_\tau x^*_{t-1} + \dot{\sigma}_\tau x^*_t$. Then given sufficient model capacity, the optimal velocity field $v_{\theta^*}(x_t(\tau), \tau; z^*_{\leq t}) = \mathbb{E}_{x_{t-1}, x_t | z_{\leq t}} \dot{x}_\tau(t)$ induces a flow ODE from initial distribution $p(x_{t-1}|z^*_{\leq t})$ to $p(x_t|z^*_{\leq t})$.

Let us factorize the distribution $p(x_{t-1}, x_t, z_{\leq t}) = p(z_{\leq t}) p(x_t, x_{t-1}|z_{\leq t})$ and we rewrite the flow matching loss as

$$\mathbb{E}_{z^*_{\leq t} \sim p(z_{\leq t})} \mathbb{E}_{(x^*_{t-1}, x^*_t) \sim p(x_{t-1}, x_t | z_{\leq t})} \mathbb{E}_{\tau \sim \text{Unif}(0,1)} \left\| v_\theta(x_t(\tau), \tau; z^*_{\leq t}) - \dot{x}_\tau(t) \right\|^2. \tag{7}$$

The minimizer of the loss above is an expected loss over all conditions $z^*_{\leq t}$. Therefore, the minimizer must also minimize a loss conditioned on any $z^*_{\leq t}$, i.e., the loss below:

$$\mathbb{E}_{(x^*_{t-1}, x^*_t) \sim p(x_{t-1}, x_t | z_{\leq t})} \mathbb{E}_{\tau \sim \text{Unif}(0,1)} \left\| v_\theta(x_t(\tau), \tau; z^*_{\leq t}) - \dot{x}_\tau(t) \right\|^2. \tag{8}$$

The optimal solution to this loss leads to a conditional velocity $v(x_t(\tau), \tau; z_{\leq t}) = \mathbb{E}_{x_{t-1}, x_t | z_{\leq t}} \dot{x}_\tau(t)$, whose flow ODE transports from initial distribution $p(x_{t-1}|z_{\leq t})$ to $p(x_t|z_{\leq t})$. As a result, the initial distribution does not align with the previous posterior $p(x_{t-1}|z_{\leq t-1})$.

B EXPERIMENTAL DETAILS

B.1 DATASET

Table 6: Dataset statistics. The trajectory length refers to the total length of prediction for a specific task, and the prediction horizon is the length of rolling prediction window per physical time. The prediction horizon = "shrinking" means the task requires a shrinking prediction window whose length equals to the trajectory length at the beginning but decreases over time. #pretraining and #finetuning refer to the number of pretraining and finetuning trajectories.

Task	Dataset	Trajectory length	Prediction horizon	Feature dim.	#pretraining	#finetuning
Forecasting	Synthetic	8	5	1	10,000	5,000
	Beam Spill	430	300	1	40,000	5,000
	Burger's Equation	16	10	64	90,000	10,000
	WeatherBench2	28	12	32×32×16	73,112	5,000
State Estimation	Lorentz Attractor	100	1	3	75,000	5,000
Planning&Control	Maze Medium	600	600 (Shrinking)	2	1,999,400	16,384
	Maze Large	800	800 (Shrinking)	2	3,999,200	32,768
	Smoke Control	65	10	64×64× 6	36,000	2,000

We present the dataset statistics in Table 6. The trajectory length refers to the total prediction duration (or episode) and prediction horizon is the length of prediction window at each time step.

#pretrain and #finetuning refer to the number of trajectories we use for model pretraining and finetuning. We provide details data construction below.

Synthetic Bimodal Gaussian random walk process. We construct a bimodal Gaussian random walk $p(s_{t+1}|s_t) = \frac{1}{2}\mathcal{N}(s_t + 1, 10^{-6}) + \frac{1}{2}\mathcal{N}(s_t - 1, 10^{-6})$, with initial state $s_1 \sim \mathcal{N}(0, 10^{-4})$. The models infer posterior distribution $p(s_{t+1:t+5}|s_t)$ over $t = 1, 2, 3, 4$. Ground-truth $p(s_{t+\Delta}|s_t)$ is a discrete distribution with $(\Delta + 1)$ modes. We first train a rectified flow model Liu et al. (2023) on 10,000 trajectories as the pretrained model, and then fine-tune it on 5,000 trajectories to obtain SBFM. We evaluate the accuracy of the learned distribution using the 1-Wasserstein distance $\mathcal{W}_1(t) = \mathcal{W}_1\big(p_\theta(x_{t+1:t+5}|x_t),\, p(x_{t+1:t+5}|x_t)\big)$, where p_θ denotes the empirical distribution induced by 1000 generated particles. We report the average Wasserstein distance $\langle W_1 \rangle = \frac{1}{T}\sum_{t=1}^{T} W_1(t)$, which summarizes performance across time steps ($T = 4$). Figure 2b plots $\langle W_1 \rangle$ with respect to flow matching sampling steps. The results show that SBFM consistently achieves lower Wasserstein distance than the pretrained model under the same sampling budget. The uncertainty band is one time standard deviation of three runs of the models.

Beam spill dynamic. We adopt the simulator from (Narayanan, 2022). The state s_t is a one-dimensional variable representing the beam intensity, and a PID controller takes the state and returns an one-dimensional control signal a_t. The simulator will simulate a 43 ms time window, with 0.1 ms per time step, resulting in a full state-action trajectory $x_{1:T}, a_{1:T}$ with $T = 430$. The simulator introduces a stochastic dynamic, by injecting a time-correlated noises $\epsilon_{1:T}$ for the process. As a result, it induces a probabilistic distribution $p(x_{1:T}, a_{1:T})$ where the future trajectory $x_{t+1:t+H}$ given history is uncertain.

Burgers' equation. We adopt the dataset from (Wei et al., 2025). The 1D Burgers' equation follows:

$$\begin{cases} \dfrac{\partial s}{\partial t} = -s \cdot \dfrac{\partial s}{\partial x} + \nu \dfrac{\partial^2 s}{\partial x^2} + a(x,t), & \text{in } [0,T] \times \Omega, \\ s(x,t) = 0, & \text{in } [0,T] \times \partial\Omega, \\ s(x,0) = s_0(x), & \text{in } \{\tau = 0\} \times \Omega. \end{cases} \tag{9}$$

Here state states $s(x,t)$ is a field over space x and time t, and input (action) $a(x,t)$ is also a function of space and time. This system dynamic is a deterministic process. To impose uncertainty, we introduce: (1) partial observation: only half of the space is observable, i.e., $s(x,t)$ for $x \in \Omega_{\text{right}}$ is removed from model input; (2) agnostic input: the input $a(x,t)$ (which is randomly generated in training and test dataset) is also excluded from model input.

Weather forecasting. We adopt the WeatherBench2 (Rasp et al., 2024). The dataset consists of global weather data from year 1959 to 2023, measured with 6 hours as interval. The data is a 2D/3D temporal data, including 2D features (surface variables) such as `sea_level_pressure`, `2m_temperature`, and 3D features such `geopotential` as a function as height (measured by Atmospheric pressure). Due to computational resources, we only choose partial features (2D:`2m_temperature`, `10m_u_component_of_wind`, `10m_v_component_of_wind`, `mean_sea_level_pressure`, 3D: `geopotential`,`temperature`,`u_component_of_wind`,`v_component_of_wind`, at height $500, 850, 1000$ Pa.) as our system states. We also constraint the latitude from 37.25 to 45.0 and longitude from 115.0 to 122.75. We use the data from 1959 to 2011 for pretraining, 2011 to 2015 for finetning and 2021 to 2023 for testing. We report `2m_temperature` in our main table 3.

Maze planning. We adopt maze-medium and maze-large from D4RL benchmark (Fu et al., 2020). The offline dataset consists of random-walk trajectories and at test time the goal is to reach a target position.

Smoke Control. We adopt the dataset from Wei et al. (2025). The system state is a 2D incompressible fluid following the Navier-Stokes equations:

$$\begin{cases} \dfrac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} - \nu \nabla^2 \mathbf{v} + \nabla p = \mathbf{f}, \\ \nabla \cdot \mathbf{v} = 0, \\ \mathbf{v}(\mathbf{x}, 0) = \mathbf{v}_0(\mathbf{x}). \end{cases} \tag{10}$$

Here the system state s consists of velocity field $\mathbf{v}$ and pressure field p, and external force field $\mathbf{f}$ is the action. The task is to generate $\mathbf{f}$ to guide an initial smoko in the field to avoid obstacles and reach

a target exit area. There are two settings in (Wei et al., 2025): large domain control and boundary control. We adopt large domain control setting, where force signals are applied to all peripheral regions outside the obstacles, consisting of 1,792 cells.

State Estimation. We adopt the Lorenz attractor simulator from (He et al., 2025) to generate data. The system state s_t is a 3D vector following an nonlinear state-space model:

$$s_t = \mathbf{F}(s_{t-1})s_{t-1} + w_t, \tag{11}$$

$$\mathbf{F}(s_{t-1}) = \exp\left(\begin{bmatrix} -10 & 10 & 0 \\ 28 & -1 & -s_{t-1,1} \\ 0 & s_{t-1,1} & -\frac{8}{3} \end{bmatrix} \Delta\right), \tag{12}$$

where w_t is a noise term (we use Gaussian noise $w_t \sim \mathcal{N}(0, q^2 I)$) and Δ is the time interval. The observation is $z_t = g(x_t) + \eta_t$, where $g(x_t)$ is a rotation operation and $\eta_t \sim \mathcal{N}(0, r^2 I)$. The task is to estimation current state system s_t given historical observation $z_{\leq t}$ (s_t cannot be directly observed).

B.2 Implementation Details of Algorithms 1,2

Forecasting. In forecasting task we have $x_t = s_{t+1:t+H}$ and $z_t = x_t$, where s_t is the real physical states we want to predict. To apply Algorithm 1, we take a small finetuning dataset $\{s^{(i)}_{1:T}, z^{(i)}_{1:T}\}_{i=1,2,\dots,n}$ and leverage a pretrained Rectified Flow model v_{θ_0} to generate some trajectories $\hat{s}^{(i)}_{t+1:t+H} \sim p_{\theta_0}(s_{t+1:t+H}|z^{(i)}_{\leq t})$. We could directly use $\hat{s}^{(i)}_{t:t+H-1}$ as the source distribution and $s^{(i)}_{t+1:t+H}$ as the target distribution in SBFM ODE Eq. equation 5. In practice, to better align the prediction at the same physical time, we instead drop the first $\hat{s}^{(i)}_t$ (as it is already observed at time t) and pad a $\hat{s}^{(i)}_{t+H-1}$ (can be seen as a moving average) and construct $(\hat{s}^{(i)}_{t+1:t+H-1}, \hat{s}^{(i)}_{t+H-1})$ as the source distribution instead.

Planning and Control. In planning and control task we have $x_t = (s_{t+1:t+H}, a_{t:t+H-1})$ and $z_t = (s_t, a_{t-1})$. To apply Algorithm 1, we require an imitation learning dataset consisting of expert state-action trajectories. We treat the pretrained flow model p_{θ_0} as an expert policy and let it interact with the environment to collect a few state-action trajectories. Ultimately, we have model predicted trajectory $\hat{s}^{(i)}_{t+1:t+H}, \hat{a}^{(i)}_{t:t+H-1} \sim p_{\theta_0}(s_{t+1:t+H}, a_{t:t+H-1}|s^{(i)}_{1:t}, \hat{a}^{(i)}_{1:t-1})$ at any time t and a resulting actual full trajectory $(s^{(i)}_{1:T}, a^{(i)}_{1:T})$, where $a^{(i)}_t$ is the actual executed action (i.e., the first predicted action at each time). We treat $(\hat{s}^{(i)}_{t:t+H-1}, \hat{a}^{(i)}_{t-1:t+H-2})$ as the source distribution and $(s^{(i)}_{t+1:t+H}, a^{(i)}_{t:t+H-1})$ as the target distribution in SBFM. Again, since at time t we observe $z^{(i)}_t = (s^{(i)}_t, a^{(i)}_{t-1})$, in practice we remove already-observed state-action $\hat{s}^{(i)}_t, \hat{a}^{(i)}_{t-1}$ and instead adopt $((\hat{s}^{(i)}_{t+1:t+H-1}, \hat{s}^{(i)}_{t+H-1}), (\hat{a}^{(i)}_{t:t+H-2}, \hat{a}^{(i)}_{t+H-2})$ as the source distribution.

State Estimation. In state estimation task we have $x_t = s_t$. Similarly we call a pretrained model to generate $\hat{s}^{(i)}_t \sim p_{\theta_0}(s_t|z^{(i)}_{\leq t})$. We use $\hat{s}^{(i)}_t$ as the source distribution and the actual physical state $s^{(i)}_t$ as the target distribution.

B.3 Implementation Details of Baselines

Warm-start Diffusion. The warm-start diffusion is a heuristic that denoise a previously noisy x_{t-1}, but conditioning on new observation z_t, to obtain updated prediction of x_t. In our implementation, warm-start diffusion share the exact pipeline as our SBFM, the only difference is that it use the pretrained model v_{pre} for denoising, while SBFM uses a finetuned flow v_θ for generation.

Asynchronous Diffusion (CL-Diffusion (Wei et al., 2025)). We implement asynchronous diffusion following the idea of Wei et al. (2025). The original CL-Diffusion requires training two diffusion models, a synchronous diffusion and an asynchronous diffusion that allows different denoising scheduling. In our implementation, we use Diffusion Forcing (Chen et al., 2024) as the pretrained model, which naturally allows to denoise in an arbitrary schedule (either synchronous or asynchronous). So we use a single pretrained diffusion forcing to replace the two models in CL-Diffusion.

B.4 Diffusion Guidance in Maze Planning

The inference of diffusion models on Maze planning task often relies on diffusion guidance based on rewards, i.e., modifying local score/velocity function during sampling. The consistency model and MeanFlow baselines lack a well-defined score/velocity function, and therefore we omit these baselines on maze planning. On the other hand, SBFM are finetuned on the imitation learning dataset of trajectories produced by guided pretrained models. Therefore, the finetuned SBFM learns the transportation between guided trajectories, thus removing the need for explicit guidance in our implementation.

C Additional Experimental Results

C.1 State Estimation

Datasets. We consider state estimation of Lorenz attractor, a three-dimensional chaotic dynamical system following the nonlinear state-space model: $s_t = f(s_t)s_t + \mathcal{N}(0, q^2 I)$ and $z_t = g(s_t) + \mathcal{N}(0, r^2 I)$, where $s_t \in \mathbb{R}^3, f(s_t) \in \mathbb{R}^{3\times 3}, g(s_t) \in \mathbb{R}^3$. It is a simplified mathematical model used to capture chaotic behavior and to understand atmospheric convection (Lorenz, 2017). We follow He et al. (2025) to set g be a rotation matrix operation, $q = r/10$ and varies r to test the model performance under different environmental stochasticity. The model takes historical observations $z_{\leq t}$ as conditions and predicts current state $x_t := s_t$. The prediction horizon is $H = 1$.

Except for learning-based approaches, we also compare against model-based approaches, including Extended Kalman Filtering (Kushner, 1967), Unscented Kalman Filtering (Julier & Uhlmann, 1997), and Particle Filtering (Del Moral, 1997). These methods have explicit access to the underlying system dynamics $f(x_t)$ and measurement function $g(x_t)$, and therefore serve as oracle-style baselines.

Results. Table 7 reports performance measured by $10\log_{10}(\text{MSE})$ under varying levels of environmental uncertainty (larger $1/r^2$[**dB**] indicates lower stochasticity). As expected, the performance of all methods degrades as environmental stochasticity increases. Notably, warm-start approaches perform reasonably well when environmental stochasticity is low, but their performance becomes worse as uncertainty increases. This behavior is expected, since warm-start methods could approximate the correct predictive distribution when successive states x_{t-1}, x_t follow similar distributions, which is more likely to hold in near-deterministic system dynamics. In contrast, SBFM explicitly learns the probability flow across successive time steps. It consistently achieves competitive performance across all uncertainty levels with one sampling step among the learning-based methods.

Table 7: Results of state estimation. The reported numbers are the $10\log_{10}(\text{MSE})$ (lower the better) with MSE averaged over the entire episode (100 steps). $1/r^2$[**dB**] $= 10\log_{10}(1/r^2)$ represents different uncertainty levels of measurements (smaller the $1/r^2$[**dB**] larger the stochasticity).

Method	**NFE**	Environment Stochasticity $1/r^2$**[dB]**			
		-10	0	10	20
Model-based					
Extended KF	N/A	2.69	-6.19	-16.49	-25.18
Unscented KF	N/A	9.05	2.58	-3.52	-16.24
Particle Filtering	N/A	3.76	-4.76	-14.68	-22.93
Diffusion Forcing	10	$11.74_{\pm 0.01}$	$0.68_{\pm 0.02}$	$-9.57_{\pm 0.07}$	$-17.08_{\pm 1.56}$
Rectified Flow	5	$11.93_{\pm 0.02}$	$0.57_{\pm 0.02}$	$-9.30_{\pm 0.30}$	$-18.65_{\pm 0.33}$
Diffusion Forcing	1	$10.59_{\pm 0.07}$	$1.15_{\pm 0.17}$	$-8.74_{\pm 0.30}$	$-15.01_{\pm 2.07}$
Rectified Flow	1	$15.39_{\pm 0.10}$	$4.90_{\pm 1.04}$	$-5.94_{\pm 0.26}$	$-13.25_{\pm 1.41}$
MeanFlow	1	$12.87_{\pm 0.62}$	$6.20_{\pm 0.44}$	$2.82_{\pm 2.00}$	$0.85_{\pm 1.70}$
Consistency Model	1	$10.85_{\pm 0.00}$	$4.15_{\pm 0.16}$	$2.78_{\pm 0.15}$	$2.45_{\pm 0.04}$
Warm-start Heuristic					
Warm-start Diffusion Forcing	1	$13.52_{\pm 0.03}$	$1.39_{\pm 0.17}$	$-9.42_{\pm 0.04}$	$-17.11_{\pm 1.51}$
Warm-start Rectified Flow	1	$16.86_{\pm 0.11}$	$5.07_{\pm 0.40}$	$-7.66_{\pm 0.42}$	$-18.45_{\pm 0.48}$
Ours					
SBFM-Rectified Flow	1	$10.12_{\pm 0.01}$	$0.26_{\pm 0.20}$	$-9.79_{\pm 0.06}$	$-19.58_{\pm 0.09}$

C.2 Performance-Latency Trade-offs

We further show the performance-latency trade-off on spill beam forecasting (Figure 4) and maze planning (Figure 5). We find SBFM only a few sampling steps (≤ 3) to achieve superior performance.

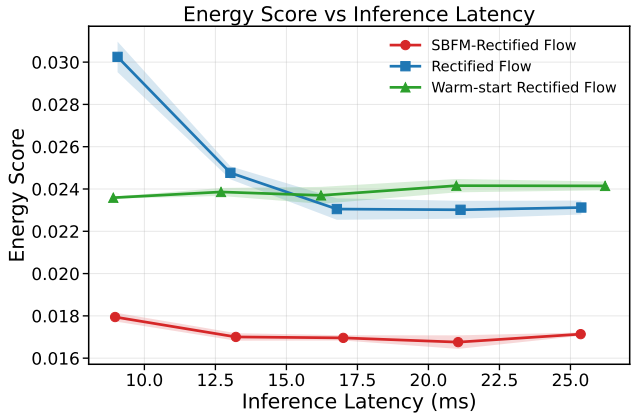


Figure 4: Inference latency (x axis) v.s. energy score (y axis) for beam spill forecasting under varying sampling steps (1, 2, 3, 4, 5 steps).

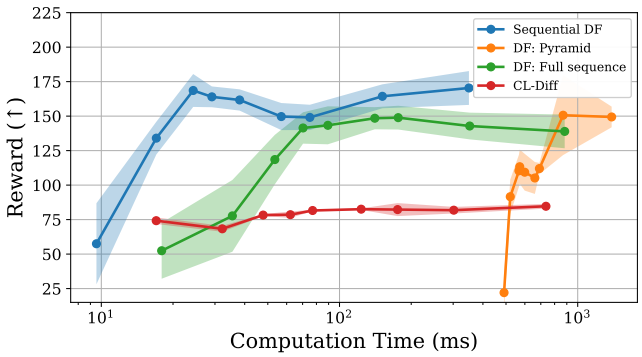


Figure 5: Inference latency (x axis) v.s. performance (y axis) for maze planning under varying sampling steps (1, 2, 3, 4, 5, 8, 10, 20 and 50 steps). DF: Pyramid and DF: Full-sequence refer to two denoising schedule of diffusion forcing. Sequential DF uses full sequence denoising by default.

C.3 Ablation Study

We conduct ablation study on re-noise mechanisms and training with model-generated trajectories.

Re-noise level. Figure 6 reports performance under different re-noise levels τ_{renoise}. We observe that both using a fully clean previous estimate ($\tau_{\text{renoise}} = 0$) and completely discarding the previous estimate ($\tau_{\text{renoise}} = 1$) lead to degraded performance. In contrast, there exists an intermediate range of τ_{renoise} that yields consistently strong and robust results. Moreover, this optimal range shifts toward larger values as system uncertainty increases (e.g., the optimal renoise level is $0.4 \sim 0.6$ for high system uncertainty $1/r^2$[dB]=-10, and is $0.2 \sim 0.4$ for low system uncertainty $1/r^2$[dB]=10). This behavior is expected: higher system uncertainty induces greater uncertainty in the model predictions, which in turn requires a higher re-noise level to adequately accommodate this stochasticity.

Training with model generated trajectories. Table 8 compares the performance of Burgers' equation forecasting and state estimation ($1/r^2$[dB]=-10) of using model-generated trajectories ($(\hat{x}_{t-1}, x_t)$) against pure ground-truth trajectories (x_{t-1}, x_t) in SBFM finetuning. We see a significant performance degrade when exclusively using ground-truth trajectories on forecasting task,

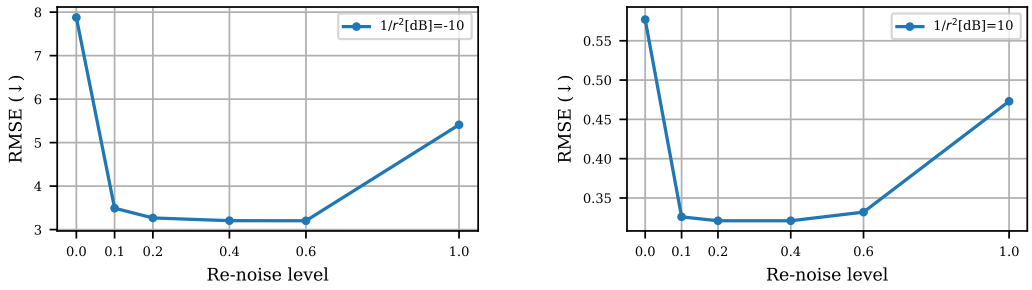


Figure 6: Performance as a function of re-noise level τ_{renoise} on state estimation. Left and right figures are at different system uncertainty levels.

while comparable performance on state tracking. We hypothesis for long-horizon task like Burgers' equation forecasting, the model's prediction will be largely deviated from ground truth, while for short-horizon task like state estimation ($H = 1$), the model prediction is close to ground truth. The different levels of train-test mismatch will decide if model-generated trajectories are necessary for finetuning SBFM.

Table 8: Ablation study of the choice of finetuning on model-generated trajectories or ground-truth trajectories.

Method		**Burgers' Equation**		**State Estimation**
	NFE	RMSE ↓	Energy Score ↓	$10\log_{10}$ MSE ↓
SBFM (finetuned from model-generated trajectories)	1	0.239	0.101	10.12
SBFM (finetuned from ground-truth trajectories)	1	0.251	0.108	10.11

C.4 Finetuning Data Efficiency

We further study how the performance of SBFM varies with the amount of finetuning data. Table 9 reports an ablation over the number of finetuning trajectories on Burgers' equation forecasting. SBFM requires substantially fewer finetuning trajectories than are used for pretraining (90k). With only 1,500–3,000 finetuning trajectories, the finetuned model begins to generalize well and outperforms the pretrained model under the same sampling budget, while the pretrained model is trained on 90,000 trajectories. These results indicate that SBFM does not require a large finetuning dataset for the tasks considered.

Table 9: Energy score and RMSE (both lower is better) on Burgers' equation forecasting. SBFM is finetuned using different numbers of trajectories. All models use 1 NFE for sampling.

# Finetuning trajectories	0 (pretrained)	300	600	1,500	3,000	6,000	10,000
Energy Score ↓	0.114	0.120	0.118	0.111	0.106	0.103	0.101
RMSE ↓	0.250	0.266	0.257	0.252	0.247	0.243	0.239

D VISUALIZATION

We provide a visualization of maze planning (Figure 8) and smoke control (Figure 7) using pre-trained diffusion model and finetuned sequential diffusion model.

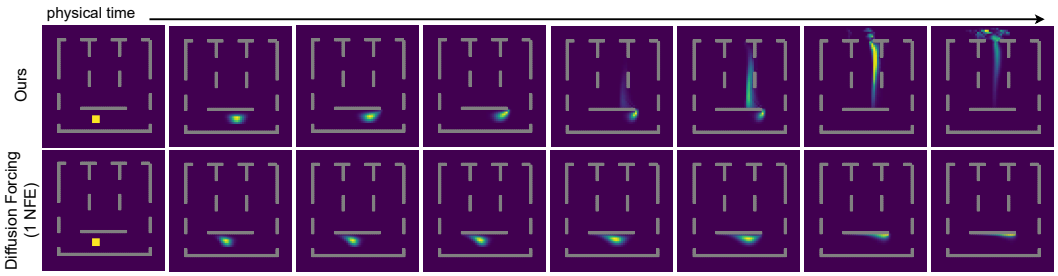


Figure 7: **Smoke Control.** With the same NFE, Sequential Diffusion Forcing effectively controls the smoke to reach the target exit, whereas Diffusion Forcing fails to circumvent the bottom obstacle.

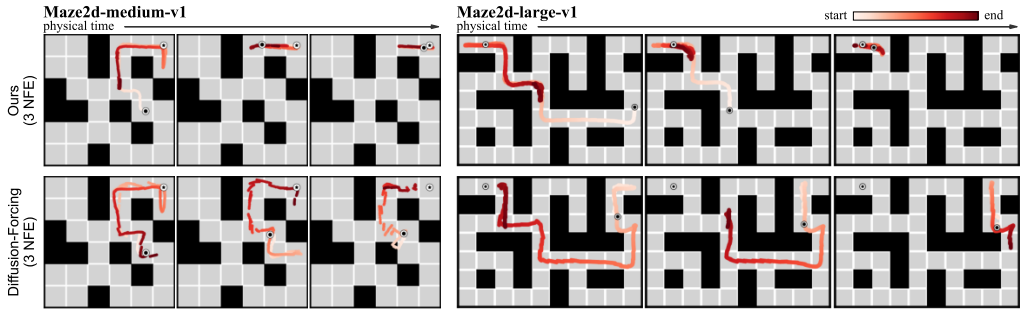


Figure 8: **Maze Planning.** With the same NFE, Sequential Diffusion Forcing can utilize the previous plans to effectively reach the target, while Diffusion Forcing fails drastically with small NFE.