Bernoulli Flow Models: Self-Consistent Generative Modeling for Binary Data

Hao Mo[1], **Liying Yang**[1], **Shumin Yao**[2], **Xinxing Yu**[1],
Ajian Liu[3]†, **Xudong Mao**[4], **Yanyan Liang**[1]†
[1]School of Computer Science and Engineering, Macau University of Science and Technology
[2]Department of Broadband Communication, Pengcheng Laboratory
[3]Institute of Automation, Chinese Academy of Sciences
[4]School of Artificial Intelligence, Sun Yat-sen University

Abstract

Binary diffusion models typically require a massive number of function evaluations (NFEs) to generate high-quality samples, making practical inference computationally expensive. Reducing NFEs while preserving sample quality without relying on distillation or additional training remains a significant challenge. However, existing binary diffusion models sequentially define a discrete one-step forward path and subsequently derive the reverse posterior. In low-NFE scenarios that require cross-step sampling, these models incorrectly approximate the true multi-step likelihood using a single-step likelihood transition, which severely degrades sample quality. To address this fundamental limitation and completely decouple the generative dynamics from fixed discrete time steps, we propose Bernoulli Flow Models (BFM). Rather than building upon sequential one-step Markov diffusion chains, we predefine a unified continuous global Bernoulli probability flow path between data distributions and pure noise, from which we derive analytical closed-form posterior transitions over arbitrary time intervals. Consequently, reducing the inference NFE in BFM is no longer an approximation of skipped discrete steps; it simply requires re-evaluating the analytical posterior on the new time intervals. This eliminates the structural training-inference mismatch inherent to discrete chains, yielding strictly self-consistent low-NFE sampling. Experimental results show that BFM is highly robust to aggressive NFE reduction. On the LSUN Churches 256x256 dataset, a 256-step-trained BFM achieves an FID of 9.22 when sampled with only 16 steps, whereas the state-of-the-art discrete baseline severely degrades to 204.10. BFM also remains competitive with both continuous and discrete generative baselines under standard full-step inference. Ultimately, these results establish BFM as a theoretically rigorous, self-consistent, and practically effective framework for fast binary data generation. Code is available at `https://github.com/minimo6666/BFM_NeurIPS2026`.

1 Introduction

Generative modeling over binary variables is a fundamental problem in binary image modeling, binarized latent representation learning, and statistical physics systems with two-state configurations. Recent binary generative models have shown that diffusion-style methods can achieve strong sample quality in binary spaces, including bit-based continuous diffusion formulations [4], Binary Latent Diffusion (BLD) [32], and recent binary diffusion models [18, 34]. However, these models still

†Corresponding authors.

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

typically require hundreds or even thousands of sequential function evaluations (NFEs) to generate high-fidelity samples, making practical inference computationally expensive.

A natural objective is to reduce the inference NFE while preserving sample quality without additional distillation, consistency training, or task-specific retraining. In continuous diffusion models, this problem has been extensively studied through training-based acceleration, such as progressive distillation [25] and consistency models [29], as well as training-free samplers, such as DDIM [28] and DPM-Solver [21], which evaluate the same continuous-time dynamics on a coarser inference grid [13, 30, 19]. In binary state spaces, however, such training-free acceleration is far less straightforward, because many binary diffusion models are defined through fixed one-step discrete Markov transitions rather than an arbitrary-interval continuous path.

The key difficulty is that low-NFE binary sampling is not merely a matter of skipping steps. Under the original full-step inference schedule, the reverse process remains aligned with the training-time Markov chain. Once the inference grid is coarsened, each reverse update must jump across multiple intermediate training intervals. For example, sampling a 256-step-trained binary model with only 16 steps requires each inference step to represent a large cross-step transition. A valid low-NFE sampler therefore needs a reverse posterior defined over arbitrary time intervals; reusing the original one-step posterior for such enlarged intervals no longer describes the same generative process.

This exposes a fundamental self-consistency issue in existing binary diffusion samplers. Although multi-step transitions may be derived in principle for a discrete Markov chain, practical binary diffusion samplers are often tied to the local transition rule used during training. When the sampling schedule changes after training, the required cross-step posterior is approximated by local one-step likelihood transitions, introducing a structural mismatch between the training-time process and the low-NFE inference-time process. As the step budget becomes smaller, this mismatch is amplified, leading to severe degradation in sample quality. Thus, the central obstacle to fast binary generation is not simply the lack of a faster numerical solver, but the lack of an analytical posterior mechanism that remains valid under changed inference grids.

Recent continuous-time discrete denoising and discrete flow-matching frameworks provide important foundations for discrete generative modeling [2, 31, 20, 3, 10, 6, 7]. Nevertheless, these general formulations do not directly resolve the Bernoulli-specific low-NFE problem considered here: deriving closed-form posterior transitions over arbitrary time intervals so that changing the inference NFE becomes a rigorous reparameterization of the same binary generative process, rather than an approximate cross-step sampler.

To address this limitation, we propose **Bernoulli Flow Models (BFM)**, a self-consistent generative framework for binary data. Instead of defining generation through fixed one-step Markov transitions, BFM starts from a unified continuous Bernoulli probability path between data and pure noise. From this path, we derive closed-form marginal, transition, and posterior distributions over arbitrary time intervals. Consequently, reducing the number of inference steps only requires re-evaluating the same analytical posterior on the new time intervals, which decouples the generative dynamics from a fixed training-time discretization and makes low-NFE sampling self-consistent by construction. Our contributions are summarized as follows: Our contributions can be summarized as follows:

- **A Closed-Form Bernoulli Generative Framework:** We propose Bernoulli Flow Models (BFM), which predefine a continuous Bernoulli probability path and derive analytical closed-form marginal, transition, and posterior distributions from this path.

- **Self-Consistent Low-NFE Inference:** The resulting arbitrary-interval posterior enables principled cross-step updates when the inference NFE is reduced after training. Instead of approximating a large multi-step jump with local one-step rules, BFM recomputes the analytical posterior on the new time intervals, yielding self-consistent low-NFE sampling.

- **Effective Training-Free Acceleration in Practice:** Experiments show that BFM is highly robust to aggressive NFE reduction. From a single 256-step checkpoint on LSUN Churches (256×256), BFM maintains an FID of 9.22 with only 16 sampling steps, whereas the corresponding baseline degrades to 204.10. Additional results show that BFM remains competitive under the original full-step inference setting across diverse tasks.

2 Related Work

Binary and Bit-Level Flow/Diffusion Models. Binary generative modeling has recently received increasing attention. Early work already explored diffusion-style modeling on binary state spaces [27]. Analog Bits [4] proposed a bit-encoded continuous diffusion approach, where discrete variables are represented as binary bits and then modeled as real-valued analog bits by a continuous diffusion model. Binary Latent Diffusion (BLD) [32] more directly studied diffusion-based generation in binarized latent spaces and demonstrated strong high-resolution image generation performance. Other recent works further investigated Bernoulli or binary diffusion formulations, such as masked Bernoulli diffusion for anomaly detection [34] and Binary Diffusion Probabilistic Models (BDPM) [18]. Concurrently, Binary Flow Matching [14] studies robust flow matching on binary manifolds from the perspective of prediction–loss space alignment, showing that aligning the training objective with the signal space can remove singular weighting and improve training stability. These works demonstrate the promise of binary or bit-level diffusion/flow modeling. However, their focus is mainly on representation, training objectives, or binary-space generative quality, while the problem of self-consistent low-NFE inference from a fixed checkpoint remains underexplored.

Fast Sampling and Training-Free Acceleration. A major limitation of diffusion-based generative models is their slow iterative sampling, which typically requires many sequential function evaluations (NFEs). In continuous domains, extensive work has studied acceleration. Training-based methods, such as progressive distillation [25] and consistency models [29], reduce sampling steps through an additional training stage. Training-free samplers, such as DDIM [28] and DPM-Solver [21], instead accelerate sampling directly from a pretrained model by evaluating the reverse dynamics on a coarser inference grid. These techniques have made low-NFE generation practical for continuous diffusion models. However, binary diffusion models cannot directly rely on continuous-domain ODE or SDE solvers, especially when the reverse process is defined through discrete Bernoulli transitions. Reducing the NFE then requires valid cross-step posterior updates over enlarged time intervals, rather than simply applying a continuous-domain solver. This makes training-free acceleration in binary generative modeling substantially less developed.

Relation to General Discrete Diffusion and Flow Models. A separate line of work studies generative modeling over broader discrete or categorical state spaces, including discrete diffusion models [1], continuous-time discrete denoising frameworks [2, 31, 20], and recent discrete flow-matching formulations [3, 11, 6, 7]. These works provide important general foundations for discrete generative modeling. Our work is complementary but more specialized: rather than proposing a generic categorical diffusion or flow construction, we focus on Bernoulli binary variables and derive closed-form posterior transitions over arbitrary time intervals, enabling self-consistent low-NFE inference from a single pretrained checkpoint.

3 Bernoulli Flow Models

In this section, we present the theoretical framework of Bernoulli Flow Models (BFM). We detail how a unified continuous global Bernoulli probability flow path is established to completely decouple the generative dynamics from fixed discrete time steps, and formally derive the resulting analytical probability distributions.

3.1 Notation and Problem Formulation

Consider an arbitrary binary data distribution $p_{\mathbf{X}}(\mathbf{x})$ over the sample space $\mathcal{X} = \{0,1\}^d$, where $d \in \mathbb{N}^+$ denotes the dimensionality, and $\mathbf{X}$ is a discrete random vector. We denote the Bernoulli distribution as $\mathcal{B}$.

Given target dataset observations $X_0 \sim \pi_0$ and a prior $X_1 \sim \pi_1$ (where $\pi_1 = \mathcal{B}(0.5)$ represents binary white noise), we define an optimal transport-style Bernoulli probability path. Analogous to linear interpolation in continuous spaces, we construct the path directly in the Bernoulli parameter space across continuous time $t_\tau \in [1, 0]$:

$$X_{t_\tau} \sim \mathcal{B}(t_\tau X_0 + (1 - t_\tau)X_1), \quad \tau = 0, 1, \ldots, T, \tag{1}$$

where $X_{t_0} \equiv X_0$ and $X_{t_T} \equiv X_1$. The random variable X_{t_τ} strictly follows the Bernoulli probability flow path from the data to the prior.

3.2 Bernoulli Probability Flow Dynamics

Building upon the optimal transport-style Bernoulli probability path defined above, and assuming the prior distribution π_1 is a uniform Bernoulli distribution $\mathcal{B}(0.5)$ (representing binary white noise), the marginal transition probability from the target observation X_0 (or X_{t_0}) to an arbitrary intermediate state X_{t_τ} can be analytically expressed as:

$$\begin{aligned} q(X_{t_\tau}|X_{t_0}) &= \mathcal{B}(X_{t_\tau}; \alpha_{t_\tau}), \\ \alpha_{t_\tau} &= g\left(X_{t_0}, \frac{1-t_\tau}{2}\right), \\ g(a,b) &= a(1-b) + b(1-a). \end{aligned} \tag{2}$$

For $0 \leq t_\tau < t_{\tau-1} \leq 1$, by exploiting the Markov property along the constructed path, we can further derive the one-step forward transition probability between sequential states $X_{t_{\tau-1}}$ and X_{t_τ}:

$$\begin{aligned} q(X_{t_\tau}|X_{t_{\tau-1}}) &= \mathcal{B}(X_{t_\tau}; g(X_{t_{\tau-1}}, \gamma_{t_\tau})), \\ \gamma_{t_\tau} &= \frac{0.5(t_{\tau-1} - t_\tau)}{t_{\tau-1}}, \end{aligned} \tag{3}$$

where γ_{t_τ} denotes the explicit flip probability from $X_{t_{\tau-1}}$ to X_{t_τ}.

Using Bayes' theorem, the ground-truth posterior distribution of the previous state $X_{t_{\tau-1}}$ conditioned on the current state X_{t_τ} and the source data X_{t_0} is formulated, for $x \in \{0, 1\}$, as:

$$q(X_{t_{\tau-1}} = x \mid X_{t_\tau}, X_{t_0}) = \frac{q(X_{t_\tau} \mid X_{t_{\tau-1}} = x) q(X_{t_{\tau-1}} = x \mid X_{t_0})}{\sum_{x' \in \{0,1\}} q(X_{t_\tau} \mid X_{t_{\tau-1}} = x') q(X_{t_{\tau-1}} = x' \mid X_{t_0})}, \tag{4}$$

which simplifies into the analytical closed-form solution:

$$q(X_{t_{\tau-1}}|X_{t_\tau}, X_{t_0}) = \mathcal{B}\left(X_{t_{\tau-1}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau})\alpha_{t_{\tau-1}}}{g(g(X_{t_\tau}, \gamma_{t_\tau}), 1 - \alpha_{t_{\tau-1}})}\right). \tag{5}$$

Similar to the x_0-prediction objective in continuous DDPM [13], we parameterize our generative flow model f_θ to directly predict the uncorrupted data X_{t_0}. This design aligns the model output with the analytical Bernoulli posterior probability path, thereby enabling the generative process to sample back to the target distribution π_0 from the prior π_1. The backward Bernoulli flow transition step is parameterized as:

$$p_\theta(X_{t_{\tau-1}}|X_{t_\tau}) = \mathcal{B}\left(X_{t_{\tau-1}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau}) g\left(f_\theta(X_{t_\tau}, t_\tau), \frac{1-t_{\tau-1}}{2}\right)}{g\left(g(X_{t_\tau}, \gamma_{t_\tau}), 1 - g\left(f_\theta(X_{t_\tau}, t_\tau), \frac{1-t_{\tau-1}}{2}\right)\right)}\right). \tag{6}$$

Here, f_θ is parameterized to directly predict the uncorrupted data X_{t_0}. Following the standard x_0-prediction paradigm widely adopted in diffusion and flow matching models, we employ binary cross-entropy (BCE) to formulate the training objective:

$$\begin{aligned} \mathcal{L}_{\text{BCE}}(\theta) := -\mathbb{E}_{t_\tau, X_{t_0}, X_{t_\tau}} \big[& X_{t_0} \log\left(f_\theta(X_{t_\tau}, t_\tau)\right) \\ & + (1 - X_{t_0}) \log\left(1 - f_\theta(X_{t_\tau}, t_\tau)\right) \big]. \end{aligned} \tag{7}$$

Detailed proofs for the derivations in this section are provided in Appendix A.

3.3 Self-Consistent Cross-Step Sampling

A fundamental advantage of BFM emerges during fast inference when the number of function evaluations (NFEs) is aggressively reduced. Suppose we change the inference time grid and need to jump from t_τ directly to a distant step $t_{\tau-k}$ ($k > 1$). In existing heuristic chain-based models, navigating this large stride requires accumulating multiple local one-step transitions, leading to severe approximation errors.

In contrast, BFM naturally supports analytical cross-step posterior sampling. Because the marginal parameters α and the transition flip probabilities γ in our framework are continuous analytical functions of time $t \in [0, 1]$, evaluated in the decreasing direction from data to noise, we can directly compute the analytical transition over any arbitrary interval. By generalizing Eq. 3, the analytical cross-step forward transition from $t_{\tau-k}$ to t_τ is uniquely determined by:

$$\gamma_{t_\tau|t_{\tau-k}} = \frac{0.5(t_{\tau-k} - t_\tau)}{t_{\tau-k}}. \tag{8}$$

By substituting $\gamma_{t_\tau|t_{\tau-k}}$ and the corresponding marginal $\alpha_{t_{\tau-k}}$ into Bayes' theorem, we establish the following core theoretical guarantee of our framework:

Theorem 1 (Analytical Cross-Step Posterior)**.** *For any reverse interval satisfying $0 \le t_\tau < t_{\tau-k} \le 1$ (and hence $t_{\tau-k} > 0$), where $k > 1$, the analytical cross-step posterior of the Bernoulli Flow Model admits a perfect closed-form solution:*

$$q(X_{t_{\tau-k}} \mid X_{t_\tau}, X_{t_0}) = \mathcal{B}\left(X_{t_{\tau-k}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\alpha_{t_{\tau-k}}}{g(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1 - \alpha_{t_{\tau-k}})}\right). \tag{9}$$

Proof. The rigorous derivation of Theorem 1 is provided in Appendix A. □

During fast sampling, we re-parameterize the decreasing schedule of $t \in [0, 1]$ from 1 to 0 to the new NFE budget K (where $K \ll T$). We refer to this property as the **self-consistency of BFM under cross-step sampling**. This establishes the structural consistency of the oracle posterior under arbitrary inference grids. The approximation errors introduced by the learned predictor are analyzed in the following section.

3.4 Error Bounds for Learned Reverse Sampling

Theorem 1 establishes the exact cross-step posterior conditioned on the clean state X_{t_0}. In practice, X_{t_0} is replaced by $f_\theta(X_t, t)$. We characterize the resulting error for both one-step and cross-step sampling.

Proposition 1 (Learned Reverse Transition Error)**.** *For any reverse interval $0 \le t < s < 1$, where $s = t_{\tau-k}$ and $t = t_\tau$, the coordinate-wise total variation (TV) distance between the oracle and learned posteriors satisfies*

$$\begin{aligned} &D_{\mathrm{TV}}\big(q(X_s \mid X_t, X_{t_0}), p_\theta(X_s \mid X_t)\big) \\ &\quad \le \frac{s^2 - t^2}{s(1-t)^2} \left| f_\theta(X_t, t) - X_{t_0} \right|. \end{aligned} \tag{10}$$

The corresponding KL divergence is also bounded by an interval-dependent constant times the binary cross-entropy (BCE) loss, connecting the training objective to the learned reverse-transition error.

Proposition 2 (Accumulated Sampling Error)**.** *For an inference grid $1 = t_0 > \cdots > t_K = 0$, the final sampling error satisfies*

$$\begin{aligned} &D_{\mathrm{TV}}\big(\pi_0, p_\theta(X_{t_0})\big) \\ &\quad \le \min\left\{1,\, E_{\mathrm{rec}} + \sum_{j=2}^{K} \frac{t_{j-1}^2 - t_j^2}{t_{j-1}(1-t_j)^2} \varepsilon_j \right\}, \end{aligned} \tag{11}$$

Here, ε_j bounds the expected soft ℓ_1 prediction error under the forward distribution, and E_{rec} denotes the final reverse-step error. These bounds distinguish the exact oracle cross-step posterior from the learned sampler, whose approximation error depends on prediction accuracy and inference intervals. Detailed proofs are provided in Appendix A.6.

Algorithm 1 Training procedure.

Given: Binary diffusion model f_θ parametrized by $\mathcal{T}_\theta$; A dataset $\mathbf{X}$.
Given: Diffusion steps T; Probability path defined by eq. (1); Training steps I.
Initializing $\mathcal{T}_\theta$.
for Step $i = 1 : I$ **do**
    Sampling data $\mathbf{x}_{t_0} \sim \mathbf{X}$, and time step $\tau \sim \{1, \ldots, T\}$.
    Obtaining $\mathbf{x}_{t_\tau}$ using $\mathbf{x}_{t_0}$, t_τ , and probability path with eq. (1).
    Predicting the probability that the state is $\mathbf{x}_{t_0}$ using $f_\theta(\mathbf{x}_{t_\tau}, t_\tau) = \sigma(\mathcal{T}_\theta(\mathbf{x}_{t_\tau}, t_\tau)/\kappa)$.
    Calculating loss $\mathcal{L}$ using eq. (7).
    Backpropagating $\mathcal{L}$ and updating θ.
end for
Return Binary diffusion model f_θ.

Algorithm 2 Sampling procedure.

Given: Trained binary diffusion model f_θ;The sample shape dimension $shape$.
Given: Diffusion steps T; Probability path defined by eq. (1); Temperature κ.
Sampling $\mathbf{x}_{t_T} = \text{Bernoulli}(\mathbf{x}^{\text{init}})$, where $\mathbf{x}^{\text{init}} \in \mathbb{R}^{shape}$ and contains 0.5 only.
for Step $\tau = T : 1$ **do**
    Predicting $p_\theta(\mathbf{x}_{t_{\tau-1}})$ with $f_\theta(\mathbf{x}_{t_\tau}, t_\tau) = \sigma(\mathcal{T}_\theta(\mathbf{x}_{t_\tau}, t_\tau)/\kappa)$ and eq. (6).
    Sampling $\mathbf{x}_{t_{\tau-1}} = \text{Bernoulli}(p_\theta(\mathbf{x}_{t_{\tau-1}}))$
end for
$\hat{\mathbf{x}}_{t_0} = \mathbf{x}_{t_{\tau-1}}$
Return the final sample $\hat{\mathbf{x}}_{t_0}$.

3.5 Model Training and Sampling

We now describe how BFM can be trained and sampled by minimal modification of existing binary diffusion [32] training and sampling architectures, as summarized in Algorithms 1 and 2, where we highlight the differences between BFM and existing binary diffusion models in blue. The prediction of X_{t_0} is implemented via a neural network $f_\theta(X_{t_\tau}, t_\tau) = \sigma(\mathcal{T}_\theta(X_{t_\tau}, t_\tau)/\kappa)$, where $\sigma(\cdot)$ is the sigmoid function, $\mathcal{T}_\theta$ is a neural network with parameters θ, and κ is a temperature hyperparameter. The temperature is used to control the diversity of the generated samples. A smaller temperature leads to less diverse samples, while a larger temperature increases diversity.

4 Experiments

We evaluate BFM from four perspectives. First, we validate our central claim that BFM maintains stability under *training-free* NFE reduction using a single pretrained checkpoint. Second, we verify its competitiveness on high-dimensional latent image benchmarks under standard sampling. Third, we demonstrate direct generative performance on binary manifolds via Binarized MNIST. Fourth, we conduct an ablation study to analyze the impact of different probability path schedulers. Furthermore, Appendix C details BFM's applicability beyond image synthesis by modeling the Ising binary physical system.

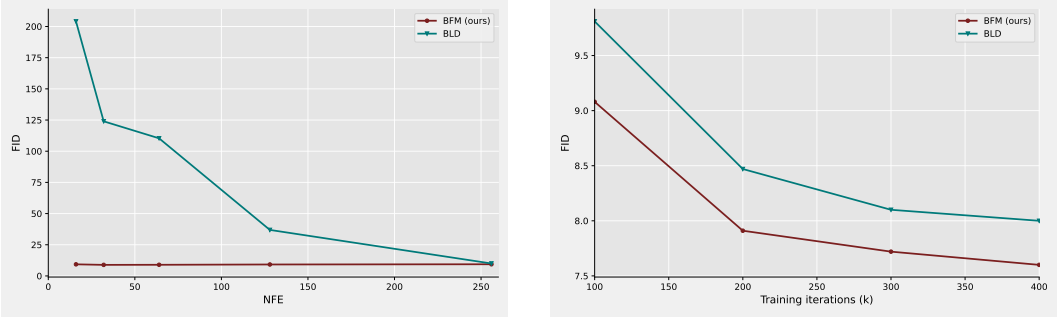
4.1 Experimental Setup

To ensure a rigorous and fair comparison with BLD [32], we implement BFM directly on top of the official BLD codebase. We fully follow the experimental settings of BLD, using the identical full transformer backbone and hyperparameters. All experiments are conducted on two NVIDIA GeForce RTX 4090 GPUs. We use the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\epsilon = 1 \times 10^{-8}$, together with a warm-up scheduler whose peak learning rate is 2×10^{-4} after 10K iterations. Unless otherwise specified, we set the temperature κ to 0.9 as in the default BLD setup, and compute all FID, Precision, and Recall metrics from 50K generated samples compared against the corresponding training datasets.

4.2 Self-Consistent Low-NFE Sampling

We first evaluate the main practical regime targeted by our method: *training-free acceleration*, where the number of function evaluations (NFEs) is reduced at inference time without retraining or distillation. This setting directly tests whether BFM remains stable when the inference grid is coarsened after training.

To this end, we compare BFM and BLD on LSUN Churches 256×256 under a fixed-checkpoint protocol. Specifically, both models are trained with 256 sampling steps using the linear probability path scheduler, and we evaluate the 100K-iteration checkpoints while varying only the inference NFE. All other training and evaluation settings follow Sec. 4.1.



(a) Low-NFE sampling robustness. (b) Training dynamics.

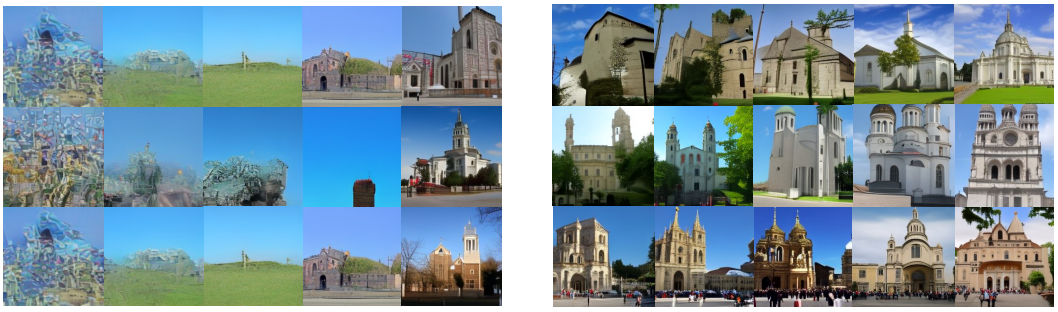
Figure 1: **Comparison of BFM and BLD under reduced inference steps and during training.** (a) FID versus NFE on LSUN Churches (256×256), evaluated from the 100K-iteration checkpoints trained with 256 sampling steps using the linear probability path scheduler, where only the inference NFE is varied. (b) Training convergence comparison on LSUN Bedrooms (256×256). We visualize the FID scores from 100K to 400K training iterations ($\times 10^3$). Both BFM and BLD are trained under identical experimental settings using a linear scheduler. BFM reaches lower FID consistently throughout this range.

Fig. 1a reports the core low-NFE result. Starting from the same 256-step training setup, BFM remains highly stable as the inference NFE is reduced, whereas BLD deteriorates rapidly under aggressive NFE reduction. In particular, when sampled with only 16 inference steps, BFM achieves an FID of 9.22 under the linear path scheduler, while the discrete baseline BLD severely degrades to 204.10. This result directly supports our central claim that BFM is substantially more robust than chain-based Bernoulli diffusion under training-free NFE reduction.

Fig. 1b further compares the training dynamics from 100K to 400K iterations on LSUN Bedrooms 256×256. Under identical architectures, schedulers, and optimization settings, BFM consistently reaches lower FID values than BLD throughout this training range. This behavior is consistent with the structural design of BFM. In our framework, the supervision at different sampled time points is induced by the same closed-form Bernoulli probability path, so different sampled times correspond to different intervals of one unified transition family. In contrast, BLD is defined on a fixed one-step discrete chain, where reverse transitions are tied to specific grid steps. This path-consistent parameterization gives BFM a more coherent training signal across time, which empirically translates into faster FID reduction during training.

To make this behavior visually explicit, Fig. 2 compares samples generated from the respective 256-step checkpoints of BLD and BFM under different inference NFEs. As the NFE decreases, BLD quickly becomes blurry and structurally unstable, while BFM preserves global scene structure and semantic layout across a broad range of NFEs. This qualitative evidence is fully consistent with the quantitative FID curve.

The practical significance of these results is that BFM supports *training-free acceleration* from a single pretrained checkpoint. Reducing the NFE does not require retraining, distillation, or switching to a different sampler. Instead, the model remains stable because the reverse updates can be recomputed on the new coarser inference grid. A more detailed discussion of the corresponding cross-step posterior discrepancy in BLD and the self-consistency mechanism of BFM is deferred to Appendix D.



(a) BLD (b) BFM

Figure 2: Visual comparison of samples generated with different NFEs from the respective checkpoints of BLD and BFM, both trained with 256 sampling steps on LSUN Churches 256×256. For each method, each row corresponds to a different NFE setting, while images in the same column share the same initial latent code.

4.3 Standard Sampling Performance on High-Dimensional Image Generation

We next evaluate BFM under the standard sampling setting to examine its generation quality with the original inference budget. This experiment complements the low-NFE analysis and verifies that BFM remains competitive in the conventional full-step sampling regime.

Methods	Steps	LSUN-Bedrooms 256x256			LSUN-Churches 256x256			FFHQ 256x256		
		FID ↓	Prec. ↑	Recall ↑	FID ↓	Prec. ↑	Recall ↑	FID ↓	Prec. ↑	Recall ↑
Continuous Generative Models										
StyleGAN [16, 17]	-	**2.35**	0.59	**0.48**	**3.86**	0.60	0.43	**4.16**	0.71	0.46
LDM-4/8/4 [24]	200	2.95	0.66	**0.48**	4.02	0.64	0.52	4.98	**0.73**	**0.50**
Patch-DM [9]	50	6.04	0.56	0.44	5.49	0.62	**0.53**	10.02	0.68	0.44
VQ-LCMD [22]	200	4.16	**0.72**	0.40	4.99	**0.75**	0.42	7.25	0.72	0.46
Discrete Generative Models										
D3PM [1]	200	6.60	0.60	0.35	6.02	0.68	0.39	9.49	**0.71**	0.41
VQ-Diffusion [12]	200	7.19	0.54	**0.37**	6.88	**0.72**	0.37	**8.79**	0.70	0.43
BLD* [32]	64	6.22	**0.69**	0.36	5.48	0.66	**0.41**	11.46	0.68	0.48
BFM (ours)	64	**5.86**	**0.69**	**0.37**	**5.32**	0.66	**0.41**	10.87	0.68	**0.49**

Table 1: Comparison of various methods for image generation on LSUN Bedrooms, LSUN Churches, and FFHQ. All images are of resolution 256×256. BLD* denotes our reproduction of BLD using its official codebase under the same machine environment and training configuration as BFM. The reproduced BLD* and BFM results are evaluated from checkpoints trained for 800K iterations. Within each method category, the best and second-best results in each column are highlighted in **bold** and underline, respectively.

To evaluate BFM on challenging image synthesis tasks, we use it as a latent-space generator. Following BLD [32], we employ the pretrained encoder and decoder to map images from LSUN Bedrooms and Churches [35], and FFHQ [16] into a binary latent space of size $16 \times 16 \times 64$. Detailed implementation specifics of the autoencoder can be found in [32].

A direct comparison to the originally reported BLD numbers is not fully controlled, because the corresponding training checkpoints are not publicly available and the original training environment differs from ours. Therefore, instead of mixing unmatched results in the main comparison, we reproduce BLD using its official codebase under the same machine environment and the same training configuration as BFM, including the backbone, optimizer, learning rate, loss function, and iteration budget. Table 1 reports only this unified comparison. Under this strictly matched setting, BFM consistently outperforms the reproduced BLD across all three datasets (e.g., FID 5.86 vs. 6.22 on LSUN Bedrooms and 10.87 vs. 11.46 on FFHQ).

Beyond the quantitative improvements presented in Table 1, we also provide a qualitative assessment to showcase BFM's ability to generate high-fidelity images across diverse datasets. Representative samples are shown in Fig. 3. Additionally, Appendix B provides extended visualizations demonstrating how increasing the sampling temperature shifts the outputs from stable, high-fidelity structures to greater visual diversity.



LSUN Bedrooms LSUN Churches FFHQ

Figure 3: Samples from BFM on LSUN-Bedrooms, LSUN-Churches and FFHQ datasets. All samples resolution are 256x256.

4.4 Performance on Binarized MNIST



Model	Base method	FID
SFM [6]	Flow Matching	4.62
α-Flow [5]	Flow Matching	5.02
DMPM [23]	Score-based Diffusion	2.89
CoVAE [26]	VAE	<u>0.58</u>
BFM (ours)	Flow Matching & Diffusion	**0.42**

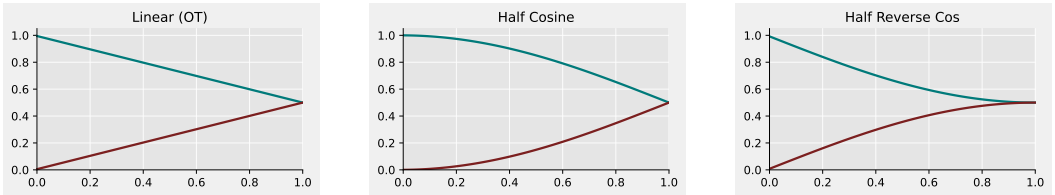
Figure 4: Left: Binarized MNIST samples generated by BFM. Right: Comparison of FID scores. The best and second-best results are highlighted in **bold** and <u>underline</u>, respectively.

To validate the generative performance of BFM on high-dimensional binary manifolds, we conduct experiments on the binarized MNIST [8] dataset. Following standard practices, we apply a thresholding scheme where pixels greater than 0 are set to 1, and others are set to 0. We evaluate the performance of BFM and report the FID scores in Figure 4, comparing it with other state-of-the-art methods.

As shown, BFM achieves the best FID score of 0.42 among the evaluated models. The FID metric used to evaluate our generated binarized MNIST samples strictly follows the codebase provided in the SFM [6] repository. We visualize the generated high-quality samples on the left side of Figure 4.

4.5 Ablation Study on Bernoulli Probability Path Scheduling Strategies

Fig. 5 visualizes how, under different probability paths, the probabilities associated with 1 (blue curves) and 0 (red curves) are progressively driven towards 0.5 along the diffusion trajectory. To systematically assess the impact of these schedulers on model performance, we select 3 different probability path schedulers and conduct experiments on the LSUN Churches 256×256 dataset using the official BLD codebase. We keep the network architecture, learning rate, loss function, and all optimization hyperparameters exactly the same, and train with a batch size of 96. The only difference lies in the algorithm at the training and sampling stages.



(a) Linear (b) Half cos (c) Reverse half cos

Figure 5: Bernoulli probability path schedulers.

Table 2: FID comparison of different Bernoulli probability path schedulers for BFM and BLD on LSUN Churches 256×256, evaluated at 100K training iterations.

Scheduler	**FID** ↓	
	BFM	**BLD**
Linear	**9.22**	9.55
Half-cosine	9.41	9.53
Reverse half-cos	9.27	9.54

We summarize the ablation results in Table 2. Under all three probability paths (linear, half-cosine, and reverse half-cosine), BFM consistently achieves lower FID than BLD. Among them, the linear scheduler gives the best overall performance for BFM (9.22). We also note that the FID of BFM varies slightly more across different schedulers than that of BLD, although the overall variation remains small. This indicates that while the specific choice of scheduling function has a measurable impact on the final generation quality, BFM maintains stable performance across different probability paths. Consequently, given its superior empirical performance, we adopt the linear schedule as the default configuration for all subsequent BFM experiments.

5 Discussion

We presented Bernoulli Flow Models (BFM), a self-consistent generative framework that defines a continuous global Bernoulli probability path between data and noise and derives analytical posterior transitions over arbitrary time intervals. This construction decouples inference from a fixed discrete grid: when the NFE changes, BFM re-evaluates the posterior on the new intervals rather than approximating skipped one-step transitions. Experiments show robust training-free low-NFE sampling while remaining competitive under standard inference. Our evaluation is currently limited to representative image-generation and scientific binary benchmarks, leaving broader domain-specific validation for future work. We also plan to combine BFM with distillation or consistency-style training for very-few-step or one-step generation.

Acknowledgments and Disclosure of Funding

This work is supported by the Science and Technology Development Fund of Macau Project (grant No. 0096/2023/RIA2, 0044/2024/AGJ, 0140/2024/AGJ, 0084/2024/RIB2), and is partially supported by the AI Supercomputing Center of Macau University of Science and Technology (MUST). We acknowledge the technical support provided by the Information Technology Development Office of MUST. We thank the anonymous reviewers for their constructive feedback and valuable suggestions. The authors declare no competing interests.

References

[1] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021.

[2] Andrew Campbell, Joe Benton, Valentin De Bortoli, Thomas Rainforth, George Deligiannidis, and Arnaud Doucet. A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, 35:28266–28279, 2022.

[3] Andrew Campbell, Jason Yim, Regina Barzilay, Tom Rainforth, and Tommi Jaakkola. Generative flows on discrete state-spaces: Enabling multimodal flows with applications to protein co-design. *arXiv preprint*, 2024.

[4] Ting Chen, Ruixiang Zhang, and Geoffrey E. Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *ArXiv*, abs/2208.04202, 2022.

[5] Chaoran Cheng, Jiahan Li, Jiajun Fan, and Ge Liu. α-Flow: A Unified Framework for Continuous-State Discrete Flow Matching Models. *arXiv preprint arXiv:2504.10283*, 2025.

[6] Chaoran Cheng, Jiahan Li, Jian Peng, and Ge Liu. Categorical flow matching on statistical manifolds. *Advances in Neural Information Processing Systems*, 37:54787–54819, 2024.

[7] Oscar Davis, Samuel Kessler, Mircea Petrache, Ismail Ceylan, Michael Bronstein, and Joey Bose. Fisher flow matching for generative modeling over discrete data. *Advances in Neural Information Processing Systems*, 37:139054–139084, 2024.

[8] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[9] Zheng Ding, Mengqi Zhang, Jiajun Wu, and Zhuowen Tu. Patched denoising diffusion models for high-resolution image synthesis. In *The twelfth international conference on learning representations*, 2023.

[10] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

[11] Itai Gat, Tal Remez, Neta Shaul, Felix Kreuk, Ricky TQ Chen, Gabriel Synnaeve, Yossi Adi, and Yaron Lipman. Discrete flow matching. *Advances in Neural Information Processing Systems*, 37:133345–133385, 2024.

[12] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10696–10706, 2022.

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

[14] Jiadong Hong, Lei Liu, Xinyu Bian, Wenjie Wang, and Zhaoyang Zhang. Binary flow matching: Prediction-loss space alignment for robust learning. *arXiv preprint arXiv:2602.10420*, 2026.

[15] Ernst Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31(1):253–258, 1925.

[16] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[17] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[18] Volodymyr Kinakh and Slavi Voloshynovskiy. Binary diffusion probabilistic model. *arXiv preprint*, 2025. Under review.

[19] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[20] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

[21] 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. *arXiv preprint arXiv:2206.00927*, 2022.

[22] Bac Nguyen, Chieh-Hsin Lai, Yuhta Takida, Naoki Murata, Toshimitsu Uesaka, Stefano Ermon, and Yuki Mitsufuji. Improving vector-quantized image modeling with latent consistency-matching diffusion. *arXiv preprint arXiv:2410.14758*, 2024.

[23] Le-Tuyet-Nhi Pham, Dario Shariatian, Antonio Ocello, Giovanni Conforti, and Alain Durmus. Discrete markov probabilistic models. *arXiv preprint arXiv:2502.07939*, 2025.

[24] 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*, pages 10684–10695, 2022.

[25] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[26] Gianluigi Silvestri and Luca Ambrogioni. Covae: Consistency training of variational autoencoders. *arXiv preprint arXiv:2507.09103*, 2025.

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

[28] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[29] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023.

[30] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[31] Haoran Sun, Lijun Yu, Bo Dai, Dale Schuurmans, and Hanjun Dai. Score-based continuous-time discrete diffusion models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[32] Ze Wang, Jiang Wang, Zicheng Liu, and Qiang Qiu. Binary latent diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22576–22585, 2023.

[33] Ulli Wolff. Collective monte carlo updating for spin systems. *Physical Review Letters*, 62(4):361, 1989.

[34] Julia Wolleb, Florentin Bieder, Paul Friedrich, Peter Zhang, Alicia Durrer, and Philippe C Cattin. Binary noise for binary tasks: Masked bernoulli diffusion for unsupervised anomaly detection. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 135–145. Springer, 2024.

[35] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

A Proofs of Bernoulli Probability Flow Dynamics

In this section, we provide the detailed and rigorous proofs for the Bernoulli Probability Flow Dynamics introduced in Section 3.

A.1 Mathematical Preliminaries

We first formalize the core algebraic operator used to simplify the binary probability space.

Definition 1 (Symmetric Binary Combination)**.** *We define the symmetric combination function* $g : [0,1] \times [0,1] \to [0,1]$ *as:*

$$g(a,b) = a(1-b) + b(1-a). \tag{12}$$

A.2 Marginal Distribution Dynamics

We begin by establishing the marginal distribution from the data X_{t_0} to an arbitrary state X_{t_τ}.

Proposition 3 (Marginal Probability)**.** *Let* $X_{t_0} \in \{0,1\}$ *be a data sample and* $X_1 \sim \mathcal{B}(0.5)$ *be a uniform prior. For time parameters* $t_\tau \in [0,1]$ *evaluated along a decreasing schedule, let* X_{t_τ} *be defined conditionally as* $X_{t_\tau} \sim \mathcal{B}\left(t_\tau X_{t_0} + (1-t_\tau)X_1\right)$*. The exact marginal distribution is given by:*

$$q(X_{t_\tau} \mid X_{t_0}) = \mathcal{B}\left(X_{t_\tau}; g\left(X_{t_0}, \frac{1-t_\tau}{2}\right)\right). \tag{13}$$

Proof. We aim to derive the conditional distribution $q(X_{t_\tau} \mid X_{t_0})$ by marginalizing over the uniform prior $X_1 \sim \mathcal{B}(0.5)$. By definition, the Bernoulli parameter for the intermediate state is $p = t_\tau X_{t_0} + (1-t_\tau)X_1$. Using the linearity of expectation, the marginal probability of observing state 1 is computed directly as:

$$\begin{aligned}
\mathbb{P}(X_{t_\tau} = 1 \mid X_{t_0}) &= \mathbb{E}_{X_1}\left[t_\tau X_{t_0} + (1-t_\tau)X_1\right] \\
&= t_\tau X_{t_0} + (1-t_\tau) \cdot \mathbb{E}[X_1] \\
&= t_\tau X_{t_0} + 0.5(1-t_\tau).
\end{aligned}$$

Notice that this compact expression naturally evaluates to $0.5(1-t_\tau)$ when $X_{t_0} = 0$, and to $0.5(1+t_\tau)$ when $X_{t_0} = 1$.

To unify this parameter into a symmetric form, we algebraically rewrite the expression as:

$$\alpha_{t_\tau} := \mathbb{P}(X_{t_\tau} = 1 \mid X_{t_0}) = \frac{1 + (2X_{t_0} - 1)t_\tau}{2}. \tag{14}$$

It is straightforward to algebraically verify that $g\left(X_{t_0}, \frac{1-t_\tau}{2}\right) = X_{t_0}\left(1 - \frac{1-t_\tau}{2}\right) + \frac{1-t_\tau}{2}(1 - X_{t_0})$, which simplifies exactly to $\frac{1+(2X_{t_0}-1)t_\tau}{2}$. Therefore, the marginal probability is exactly parameterized by $\alpha_{t_\tau} = g\left(X_{t_0}, \frac{1-t_\tau}{2}\right)$. □

A.3 One-Step Forward Transition Dynamics

We now prove the transition probability between two sequential states $X_{t_{\tau-1}}$ and X_{t_τ}.

Proposition 4 (Forward Transition Probability)**.** *Let* $0 \le t_\tau < t_{\tau-1} \le 1$*, so that* $t_{\tau-1} > 0$*. Assuming a Markovian transition process from* $X_{t_{\tau-1}}$ *to* X_{t_τ} *with an error rate* γ_{t_τ}*, the transition probability is uniquely specified by:*

$$q(X_{t_\tau} \mid X_{t_{\tau-1}}) = \mathcal{B}\left(X_{t_\tau}; g(X_{t_{\tau-1}}, \gamma_{t_\tau})\right), \quad \textit{where} \quad \gamma_{t_\tau} = \frac{0.5(t_{\tau-1} - t_\tau)}{t_{\tau-1}}. \tag{15}$$

Proof. Let α_{t_τ} and $\alpha_{t_{\tau-1}}$ be the marginal probabilities of state 1 at their respective time steps, defined via Eq. 14. By the law of total probability, the marginal probability α_{t_τ} must aggregate all paths from the previous state:

$$\alpha_{t_\tau} = \sum_{x \in \{0,1\}} q(X_{t_\tau} = 1 \mid X_{t_{\tau-1}} = x) \cdot q(X_{t_{\tau-1}} = x \mid X_{t_0}). \tag{16}$$

Substituting the parameterized transition probabilities yields:

$$\alpha_{t_\tau} = \alpha_{t_{\tau-1}}(1-\gamma_{t_\tau}) + (1-\alpha_{t_{\tau-1}})\gamma_{t_\tau}. \tag{17}$$

Rearranging the terms and solving for the flip probability γ_{t_τ}:

$$\alpha_{t_\tau} = \alpha_{t_{\tau-1}} - 2\gamma_{t_\tau}\alpha_{t_{\tau-1}} + \gamma_{t_\tau} \implies \gamma_{t_\tau} = \frac{\alpha_{t_{\tau-1}} - \alpha_{t_\tau}}{2\alpha_{t_{\tau-1}} - 1}. \tag{18}$$

Substituting $\alpha_{t_\tau} = \frac{1+(2X_{t_0}-1)t_\tau}{2}$ and $\alpha_{t_{\tau-1}} = \frac{1+(2X_{t_0}-1)t_{\tau-1}}{2}$ into the equation exactly cancels out the data-dependent term $(2X_{t_0}-1)$, resulting in:

$$\gamma_{t_\tau} = \frac{\frac{1}{2}(2X_{t_0}-1)(t_{\tau-1}-t_\tau)}{(2X_{t_0}-1)t_{\tau-1}} = \frac{0.5(t_{\tau-1}-t_\tau)}{t_{\tau-1}}. \tag{19}$$

This confirms the exact functional form of the forward transition. □

A.4 Analytical One-Step Posterior Derivation

Proposition 5 (One-Step Closed-Form Posterior)**.** *The ground-truth posterior distribution of $X_{t_{\tau-1}}$ conditioned on X_{t_τ} and the source data X_{t_0} admits the exact closed-form solution:*

$$q(X_{t_{\tau-1}} \mid X_{t_\tau}, X_{t_0}) = \mathcal{B}\left(X_{t_{\tau-1}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau})\alpha_{t_{\tau-1}}}{g(g(X_{t_\tau}, \gamma_{t_\tau}), 1-\alpha_{t_{\tau-1}})}\right). \tag{20}$$

Proof. Our objective is to find the analytical Bernoulli parameter for the reverse transition, which requires calculating the probability of $X_{t_{\tau-1}} = 1$ using Bayes' theorem:

$$q(X_{t_{\tau-1}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{q(X_{t_\tau} \mid X_{t_{\tau-1}} = 1) \cdot q(X_{t_{\tau-1}} = 1 \mid X_{t_0})}{\sum_{x\in\{0,1\}} q(X_{t_\tau} \mid X_{t_{\tau-1}} = x) \cdot q(X_{t_{\tau-1}} = x \mid X_{t_0})}. \tag{21}$$

Based on the dynamics established in Propositions 3 and 4, we define the constituent transition probabilities:

$$\begin{aligned} q(X_{t_\tau} \mid X_{t_{\tau-1}} = 1) &= g(X_{t_\tau}, \gamma_{t_\tau}), & q(X_{t_{\tau-1}} = 1 \mid X_{t_0}) &= \alpha_{t_{\tau-1}}, \\ q(X_{t_\tau} \mid X_{t_{\tau-1}} = 0) &= 1 - g(X_{t_\tau}, \gamma_{t_\tau}), & q(X_{t_{\tau-1}} = 0 \mid X_{t_0}) &= 1 - \alpha_{t_{\tau-1}}. \end{aligned} \tag{22}$$

Substituting these precise terms into the fractional Bayes' formulation in Eq. 21, we obtain:

$$q(X_{t_{\tau-1}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{g(X_{t_\tau}, \gamma_{t_\tau})\alpha_{t_{\tau-1}}}{g(X_{t_\tau}, \gamma_{t_\tau})\alpha_{t_{\tau-1}} + (1 - g(X_{t_\tau}, \gamma_{t_\tau}))\,(1-\alpha_{t_{\tau-1}})}. \tag{23}$$

By recognizing the symmetric structure of the denominator, we apply the definition $g(a,b) = a(1-b) + b(1-a)$, which folds the denominator perfectly into $g\left(g(X_{t_\tau}, \gamma_{t_\tau}), 1-\alpha_{t_{\tau-1}}\right)$. Thus, the final expression simplifies to:

$$q(X_{t_{\tau-1}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{g(X_{t_\tau}, \gamma_{t_\tau})\alpha_{t_{\tau-1}}}{g(g(X_{t_\tau}, \gamma_{t_\tau}), 1-\alpha_{t_{\tau-1}})}, \tag{24}$$

concluding the derivation of the one-step posterior. □

A.5 Exact Cross-Step Posterior (Proof of Theorem 1)

Finally, we extend the one-step derivation to any arbitrary interval to prove the fast-sampling mechanism introduced in the main text.

Theorem 1 (Exact Cross-Step Posterior, restated). *For any reverse interval satisfying $0 \le t_\tau < t_{\tau-k} \le 1$ (and hence $t_{\tau-k} > 0$), where $k > 1$, the exact cross-step posterior of the Bernoulli Flow Model admits a perfect closed-form solution:*

$$q(X_{t_{\tau-k}} \mid X_{t_\tau}, X_{t_0}) = \mathcal{B}\left(X_{t_{\tau-k}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\alpha_{t_{\tau-k}}}{g(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1-\alpha_{t_{\tau-k}})}\right). \tag{25}$$

Proof. Because our parameters α and γ are defined continuously across the interpolation path, the algebraic relationships proven in Propositions 3 and 4 hold for any arbitrary temporal stride. To jump from t_τ directly back to $t_{\tau-k}$, we generalize the transition error rate over this specific interval as $\gamma_{t_\tau|t_{\tau-k}} = \frac{0.5(t_{\tau-k}-t_\tau)}{t_{\tau-k}}$.

Using Bayes' theorem, the probability of $X_{t_{\tau-k}} = 1$ is calculated exactly as:

$$q(X_{t_{\tau-k}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{q(X_{t_\tau} \mid X_{t_{\tau-k}} = 1) \cdot q(X_{t_{\tau-k}} = 1 \mid X_{t_0})}{\sum_{x\in\{0,1\}} q(X_{t_\tau} \mid X_{t_{\tau-k}} = x) \cdot q(X_{t_{\tau-k}} = x \mid X_{t_0})}. \tag{26}$$

Substituting the generalized constituent parameters for the k-step stride:

$$\begin{aligned} &q(X_{t_\tau} \mid X_{t_{\tau-k}} = 1) = g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), && q(X_{t_{\tau-k}} = 1 \mid X_{t_0}) = \alpha_{t_{\tau-k}}, \\ &q(X_{t_\tau} \mid X_{t_{\tau-k}} = 0) = 1 - g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), && q(X_{t_{\tau-k}} = 0 \mid X_{t_0}) = 1 - \alpha_{t_{\tau-k}}. \end{aligned} \tag{27}$$

Inserting these into Eq. 26 yields the expanded fraction:

$$q(X_{t_{\tau-k}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\alpha_{t_{\tau-k}}}{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\alpha_{t_{\tau-k}} + \left(1 - g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\right)\left(1 - \alpha_{t_{\tau-k}}\right)}. \tag{28}$$

Applying the symmetric operator definition $g(a, b) = a(1-b)+b(1-a)$ to the denominator precisely folds it into $g\left(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1 - \alpha_{t_{\tau-k}}\right)$. Consequently, the analytical cross-step posterior is cleanly obtained:

$$q(X_{t_{\tau-k}} = 1 \mid X_{t_\tau}, X_{t_0}) = \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}})\alpha_{t_{\tau-k}}}{g(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1 - \alpha_{t_{\tau-k}})}, \tag{29}$$

which establishes the structural consistency of the oracle posterior under changes of the inference grid. The learned sampler remains subject to the prediction error analyzed in the following subsection. □

A.6 Error Bounds for Learned Reverse Sampling

We provide the proofs of Propositions 1 and 2. We first establish the coordinate-wise transition bounds, then extend them to the full binary state and the accumulated sampling error.

A.6.1 Prediction-Induced TV Bound

Consider an arbitrary reverse interval $0 \le t < s < 1$, where $s = t_{\tau-k}$ and $t = t_\tau$. Following Eq. 8, the transition flip probability is

$$\gamma_{t|s} = \frac{s - t}{2s}. \tag{30}$$

For a fixed noisy state X_t, we express the analytical posterior parameter as a function of the clean-state input $u \in [0, 1]$:

$$r(u) = \frac{g(X_t, \gamma_{t|s})g\left(u, \frac{1-s}{2}\right)}{g\left(g(X_t, \gamma_{t|s}), 1 - g\left(u, \frac{1-s}{2}\right)\right)}. \tag{31}$$

The oracle and learned posterior parameters are therefore

$$q(X_s = 1 \mid X_t, X_{t_0}) = r(X_{t_0}), \qquad p_\theta(X_s = 1 \mid X_t) = r(f_\theta(X_t, t)). \tag{32}$$

Using the definition of g in Eq. (2), we have

$$\frac{d}{du} g\left(u, \frac{1-s}{2}\right) = s. \tag{33}$$

Differentiating Eq. (31) with respect to u gives

$$r'(u) = \frac{s\, g(X_t, \gamma_{t|s})\left[1 - g(X_t, \gamma_{t|s})\right]}{\left[g\left(g(X_t, \gamma_{t|s}), 1 - g\left(u, \frac{1-s}{2}\right)\right)\right]^2}. \tag{34}$$

Since $X_t \in \{0, 1\}$,

$$g(X_t, \gamma_{t|s}) \in \left\{ \frac{s-t}{2s}, \frac{s+t}{2s} \right\}. \tag{35}$$

Consequently,

$$g(X_t, \gamma_{t|s}) \left[1 - g(X_t, \gamma_{t|s}) \right] = \frac{s^2 - t^2}{4s^2}. \tag{36}$$

Moreover, the denominator before squaring in Eq. (34) can be expressed as

$$g\left(g(X_t, \gamma_{t|s}), 1 - g\left(u, \frac{1-s}{2} \right) \right) = \begin{cases} \frac{1-t}{2} + tu, & X_t = 1, \\ \frac{1+t}{2} - tu, & X_t = 0. \end{cases} \tag{37}$$

For all $u \in [0, 1]$, this quantity is bounded below by $(1 - t)/2$. Therefore,

$$|r'(u)| \leq \frac{s^2 - t^2}{s(1-t)^2}. \tag{38}$$

Applying the mean-value theorem yields

$$|r(f_\theta(X_t, t)) - r(X_{t_0})| \leq \frac{s^2 - t^2}{s(1-t)^2} \left| f_\theta(X_t, t) - X_{t_0} \right|. \tag{39}$$

For two Bernoulli distributions, the TV distance equals the absolute difference between their Bernoulli parameters. Hence,

$$D_{\mathrm{TV}}\big(q(X_s \mid X_t, X_{t_0}), p_\theta(X_s \mid X_t)\big) \leq \frac{s^2 - t^2}{s(1-t)^2} \left| f_\theta(X_t, t) - X_{t_0} \right|. \tag{40}$$

This proves Proposition 1 for both adjacent-step and arbitrary cross-step reverse transitions.

A.6.2 KL Bound and Connection to BCE

We next connect the prediction-induced reverse-transition error to the BCE objective.

For $0 \leq t < s < 1$, the posterior parameter in Eq. (31) satisfies

$$r(u) \in [\delta_{s,t}, 1 - \delta_{s,t}], \qquad u \in [0, 1], \tag{41}$$

where

$$\delta_{s,t} = \frac{(s-t)(1-s)}{2s(1+t)} > 0. \tag{42}$$

To see this, both $g(X_t, \gamma_{t|s})$ and $1 - g(X_t, \gamma_{t|s})$ are at least $(s-t)/(2s)$. Similarly, both $g(u, (1-s)/2)$ and $1 - g(u, (1-s)/2)$ are at least $(1-s)/2$, while the common denominator in Eq. (31) is at most $(1+t)/2$.

For Bernoulli distributions with parameters $a, b \in (0, 1)$, the KL divergence satisfies

$$D_{\mathrm{KL}}\big(\mathcal{B}(a) \| \mathcal{B}(b)\big) \leq \frac{(a-b)^2}{b(1-b)}. \tag{43}$$

Combining this inequality with Eq. (38) gives

$$D_{\mathrm{KL}}\big(q(X_s \mid X_t, X_{t_0}) \| p_\theta(X_s \mid X_t)\big) \leq C_{s,t} \left| f_\theta(X_t, t) - X_{t_0} \right|^2, \tag{44}$$

where

$$C_{s,t} = \frac{1}{\delta_{s,t}(1 - \delta_{s,t})} \left(\frac{s^2 - t^2}{s(1-t)^2} \right)^2. \tag{45}$$

For any binary $x \in \{0, 1\}$ and $r \in (0, 1)$,

$$|r - x|^2 \leq -x \log r - (1 - x) \log(1 - r). \tag{46}$$

Therefore, the coordinate-wise KL divergence is bounded by

$$D_{\mathrm{KL}}\big(q(X_s \mid X_t, X_{t_0}) \| p_\theta(X_s \mid X_t)\big) \le C_{s,t}\Big[- X_{t_0} \log f_\theta(X_t, t) \\ - (1 - X_{t_0}) \log(1 - f_\theta(X_t, t))\Big]. \tag{47}$$

For the d-dimensional extension, assume that the forward corruption acts independently across coordinates conditional on the preceding state and that the learned reverse transition is product Bernoulli conditional on its current state. Under this assumption, the oracle posterior conditioned on X_{t_0} and the learned transition both factorize coordinate-wise. The full-state KL divergence is therefore bounded by the sum of the corresponding coordinate-wise BCE terms.

The true reverse transition is obtained by marginalizing the oracle posterior over the clean state:

$$q(X_s \mid X_t) = \mathbb{E}_{q(X_{t_0} \mid X_t)} \left[q(X_s \mid X_t, X_{t_0}) \right]. \tag{48}$$

By convexity of KL divergence,

$$D_{\mathrm{KL}}\big(q(X_s \mid X_t) \| p_\theta(X_s \mid X_t)\big) \le \mathbb{E}_{q(X_{t_0} \mid X_t)} \Big[D_{\mathrm{KL}}\big(q(X_s \mid X_t, X_{t_0}) \| p_\theta(X_s \mid X_t)\big) \Big]. \tag{49}$$

Combining this with Eq. (47) shows that, for each fixed non-terminal interval (s, t), the discrepancy from the true reverse transition is controlled by the expected BCE terms with the interval-dependent factor $C_{s,t}$. For any fixed finite collection of such intervals, the maximum of $C_{s,t}$ over that collection is finite, so summing over coordinates and averaging over the corresponding training distribution gives an upper bound proportional to the BCE objective in Eq. (7), restricted to those intervals. This is an interval-wise control statement; no grid-uniform constant is claimed as $s \uparrow 1$.

A.6.3 Accumulated Low-NFE Sampling Error

Consider an arbitrary inference grid

$$1 = t_0 > t_1 > \cdots > t_K = 0, \tag{50}$$

where K is the number of reverse transitions.

For each intermediate step $j \ge 2$, assume the expected soft prediction error satisfies

$$\mathbb{E}_{q(X_{t_0}, X_{t_j})} \Big[\big\| f_\theta(X_{t_j}, t_j) - X_{t_0} \big\|_1 \Big] \le \varepsilon_j. \tag{51}$$

Under the same coordinate-wise product structure, the full-state TV distance is bounded by the sum of the coordinate-wise TV distances. Combining this property with Proposition 1 and the convexity of TV gives

$$\mathbb{E}_{q(X_{t_j})} \Big[D_{\mathrm{TV}}\big(q(X_{t_{j-1}} \mid X_{t_j}), p_\theta(X_{t_{j-1}} \mid X_{t_j})\big) \Big] \le \frac{t_{j-1}^2 - t_j^2}{t_{j-1}(1 - t_j)^2} \varepsilon_j. \tag{52}$$

The final transition to $t_0 = 1$ is treated separately.

We define its expected reverse-transition discrepancy as

$$E_{\mathrm{rec}} := \mathbb{E}_{q(X_{t_1})} \Big[D_{\mathrm{TV}}\big(q(X_{t_0} \mid X_{t_1}), p_\theta(X_{t_0} \mid X_{t_1})\big) \Big]. \tag{53}$$

By the contraction property of Markov kernels and the triangle inequality, each reverse step satisfies

$$D_{\mathrm{TV}}\big(q(X_{t_{j-1}}), p_\theta(X_{t_{j-1}})\big) \le D_{\mathrm{TV}}\big(q(X_{t_j}), p_\theta(X_{t_j})\big) \\ + \mathbb{E}_{q(X_{t_j})} \Big[D_{\mathrm{TV}}\big(q(X_{t_{j-1}} \mid X_{t_j}), p_\theta(X_{t_{j-1}} \mid X_{t_j})\big) \Big]. \tag{54}$$

Since the true and learned reverse processes start from the same Bernoulli prior,

$$q(X_{t_K}) = p_\theta(X_{t_K}) = \pi_1, \tag{55}$$

the initial TV discrepancy is zero.

Applying Eq. (54) recursively and using Eq. (52) and Eq. (53) yields

$$D_{\mathrm{TV}}\big(\pi_0, p_\theta(X_{t_0})\big) \le \min\left\{1,\ E_{\mathrm{rec}} + \sum_{j=2}^{K} \frac{t_{j-1}^2 - t_j^2}{t_{j-1}(1-t_j)^2}\varepsilon_j\right\}. \tag{56}$$

This proves Proposition 2.

The result shows that the oracle cross-step posterior remains analytically exact, while the learned sampler accumulates errors determined by prediction accuracy and the selected inference intervals.

A.7 Connecting the Analytical Posterior to the Practical Sampler

This subsection clarifies how the clean-data-conditioned posterior in Theorem 1 is related to the practical reverse transition used during generation. Specifically, we explain three points: why the posterior conditioned on X_{t_0} is available as a ground-truth denoising target during training, why likelihood-based training requires the learned reverse transition to approximate this posterior, and why the resulting reverse transition can be parameterized by predicting X_{t_0} rather than directly predicting the complete posterior probability.

Forward and reverse Bernoulli processes. Let

$$1 = t_0 > t_1 > \cdots > t_T = 0 \tag{57}$$

denote a finite time grid. Here, X_{t_0} is a clean binary observation drawn from the data distribution π_0, whereas X_{t_T} follows the product-Bernoulli prior $\pi_1 = \mathcal{B}(\frac{1}{2}\mathbf{1}_d) = \bigotimes_{i=1}^{d} \mathcal{B}(\frac{1}{2})$.

The predefined forward process progressively transforms the clean data into Bernoulli noise:

$$q(X_{t_{1:T}} \mid X_{t_0}) = \prod_{\tau=1}^{T} q(X_{t_\tau} \mid X_{t_{\tau-1}}). \tag{58}$$

All transitions in this process are analytically specified and contain no learnable parameters.

The generative model operates in the opposite direction:

$$p_\theta(X_{t_{0:T}}) = p(X_{t_T}) \prod_{\tau=1}^{T} p_\theta(X_{t_{\tau-1}} \mid X_{t_\tau}). \tag{59}$$

Generation begins from $X_{t_T} \sim \pi_1$ and repeatedly samples from the learned reverse transitions until a clean sample X_{t_0} is obtained. The learning problem is therefore to construct $p_\theta(X_{t_{\tau-1}} \mid X_{t_\tau})$ using only the current noisy state X_{t_τ}.

Why the oracle posterior appears during training. Ideally, the reverse sampler should reproduce the true reverse transition

$$q(X_{t_{\tau-1}} \mid X_{t_\tau}). \tag{60}$$

This distribution is generally intractable because evaluating it requires marginalizing over the unknown data distribution. During training, however, the clean sample X_{t_0} is observed from the dataset, and X_{t_τ} is generated by applying the predefined forward process to this known sample. We can therefore evaluate the tractable conditional posterior

$$q(X_{t_{\tau-1}} \mid X_{t_\tau}, X_{t_0}). \tag{61}$$

We refer to this distribution as the oracle posterior because it has access to the clean state toward which X_{t_τ} should be denoised. The conditioning variable X_{t_0} is available only when constructing the training target; it is not assumed to be available during generation.

The role of this oracle posterior follows directly from maximum-likelihood training. The objective of the generative model is to maximize the likelihood assigned to the observed clean data, $\log p_\theta(X_{t_0})$.

Marginalizing over the intermediate binary states gives

$$\begin{aligned}
\log p_\theta(X_{t_0}) &= \log \sum_{X_{t_1},\dots,X_{t_T}} p_\theta(X_{t_0}, X_{t_1}, \dots, X_{t_T}) \\
&= \log \sum_{X_{t_1},\dots,X_{t_T}} p_\theta(X_{t_{0:T}}) \frac{q(X_{t_{1:T}} \mid X_{t_0})}{q(X_{t_{1:T}} \mid X_{t_0})} \\
&= \log \sum_{X_{t_1},\dots,X_{t_T}} q(X_{t_{1:T}} \mid X_{t_0}) \frac{p_\theta(X_{t_{0:T}})}{q(X_{t_{1:T}} \mid X_{t_0})} \\
&= \log \mathbb{E}_{q(X_{t_{1:T}} \mid X_{t_0})} \left[\frac{p_\theta(X_{t_{0:T}})}{q(X_{t_{1:T}} \mid X_{t_0})} \right].
\end{aligned} \tag{62}$$

Applying Jensen's inequality produces a tractable lower bound:

$$\log p_\theta(X_{t_0}) \geq \mathbb{E}_q \left[\log \frac{p_\theta(X_{t_{0:T}})}{q(X_{t_{1:T}} \mid X_{t_0})} \right]. \tag{63}$$

Substituting the forward factorization in Eq. (58) and the reverse factorization in Eq. (59), and rearranging the forward transitions using Bayes' rule, decomposes this lower bound as

$$\begin{aligned}
\log p_\theta(X_{t_0}) \geq\ & \mathbb{E}_q \left[\log p_\theta(X_{t_0} \mid X_{t_1}) \right] \\
& - D_{\mathrm{KL}} \left(q(X_{t_T} \mid X_{t_0}) \| p(X_{t_T}) \right) \\
& - \sum_{\tau=2}^{T} \mathbb{E}_{q(X_{t_\tau} \mid X_{t_0})} \left[D_{\mathrm{KL}} \left(q(X_{t_{\tau-1}} \mid X_{t_\tau}, X_{t_0}) \| p_\theta(X_{t_{\tau-1}} \mid X_{t_\tau}) \right) \right].
\end{aligned} \tag{64}$$

Equation (64) makes the learning objective explicit. The first term encourages reconstruction of the clean sample from the least corrupted state. The second term matches the endpoint of the forward process to the prescribed Bernoulli prior and contains no learnable reverse-transition parameters. Most importantly, maximizing the lower bound requires minimizing the denoising-matching terms

$$D_{\mathrm{KL}} \left(q(X_{t_{\tau-1}} \mid X_{t_\tau}, X_{t_0}) \| p_\theta(X_{t_{\tau-1}} \mid X_{t_\tau}) \right). \tag{65}$$

Thus, the oracle posterior is not an additional assumption made at inference time. It is the tractable ground-truth reverse transition that the practical sampler is trained to approximate. By making these per-step reverse transitions accurate, the learned process can iteratively transform samples from the Bernoulli prior back toward the data distribution.

Why predicting X_{t_0} is sufficient. Equation (65) specifies the learning target: the practical reverse transition $p_\theta(X_{t_{\tau-k}} \mid X_{t_\tau})$ should approximate the ground-truth posterior $q(X_{t_{\tau-k}} \mid X_{t_\tau}, X_{t_0})$. It does not, however, specify how p_θ should be parameterized.

One possible way to parameterize the practical reverse transition is to use a neural network to directly predict the complete Bernoulli probability of $p_\theta(X_{t_{\tau-k}} \mid X_{t_\tau})$ and train it to match the Bernoulli parameter of the ground-truth posterior $q(X_{t_{\tau-k}} \mid X_{t_\tau}, X_{t_0})$. In other words, the network could directly regress the full oracle posterior probability for each noisy state and time interval. However, as noted in BLD [32], directly modeling such reverse-transition probabilities can be challenging, because the model must accurately regress sophisticated interpolations that vary jointly with the current noisy state and the noise level.

In BFM, Theorem 1 provides the analytical form of this posterior. For an arbitrary reverse interval from t_τ to $t_{\tau-k}$, the oracle posterior is

$$q(X_{t_{\tau-k}} \mid X_{t_\tau}, X_{t_0}) = \mathcal{B} \left(X_{t_{\tau-k}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}) \alpha_{t_{\tau-k}}}{g\left(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1 - \alpha_{t_{\tau-k}}\right)} \right), \tag{66}$$

where

$$\alpha_{t_{\tau-k}} = g\left(X_{t_0}, \frac{1 - t_{\tau-k}}{2}\right), \qquad \gamma_{t_\tau|t_{\tau-k}} = \frac{0.5(t_{\tau-k} - t_\tau)}{t_{\tau-k}}. \tag{67}$$

For a given reverse update, X_{t_τ} and the inference times are known, and $\gamma_{t_\tau|t_{\tau-k}}$ is analytically determined. Thus, the only unknown quantity in Eq. (66) is X_{t_0}. We use the neural network $f_\theta(X_{t_\tau}, t_\tau)$ to predict X_{t_0}. Hence, the posterior p_θ parameterized by f_θ can be written as

$$
\begin{aligned}
&p_\theta(X_{t_{\tau-k}} \mid X_{t_\tau}) \\
&= \mathcal{B}\left(X_{t_{\tau-k}}; \frac{g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}) g\left(f_\theta(X_{t_\tau}, t_\tau), \frac{1-t_{\tau-k}}{2}\right)}{g\left(g(X_{t_\tau}, \gamma_{t_\tau|t_{\tau-k}}), 1 - g\left(f_\theta(X_{t_\tau}, t_\tau), \frac{1-t_{\tau-k}}{2}\right)\right)} \right).
\end{aligned} \tag{68}
$$

Therefore, X_{t_0}-prediction provides a structured parameterization of posterior matching. In the ideal case where $f_\theta(X_{t_\tau}, t_\tau) = X_{t_0}$, Eq. (68) exactly recovers the oracle posterior in Eq. (66). This reparameterization through X_{t_0}-prediction corresponds to the standard x_0-prediction strategy widely used in DDPMs and SDE diffusion models [13, 30].

B Visualization of Different Temperatures of BFM Generated Samples



Figure 6: Unconditional 256×256 samples on LSUN Bedrooms. Sampling temperatures κ are linearly interpolated from 0.5 to 1.0 (left to right).



Figure 7: Unconditional 256×256 samples on LSUN Churches. Sampling temperatures κ are linearly interpolated from 0.5 to 1.0 (left to right).



Figure 8: Unconditional 256×256 samples on FFHQ. Sampling temperatures κ are linearly interpolated from 0.5 to 1.0 (left to right).

C Beyond Image Synthesis: Ising Model Generation

Finally, we evaluate BFM on the Ising model, a canonical probabilistic model in statistical physics defined over interacting spin variables $\mathbf{s} \in \{+1, -1\}^D$ [15]. This experiment demonstrates that BFM is not restricted to image generation, but also applies to general binary scientific data.

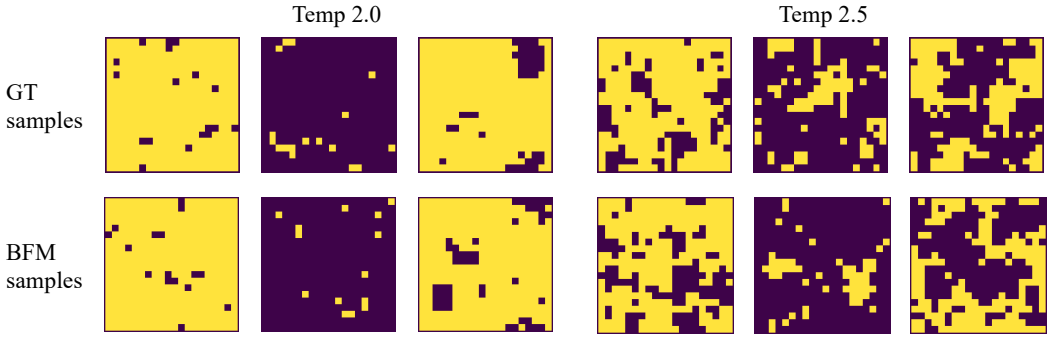
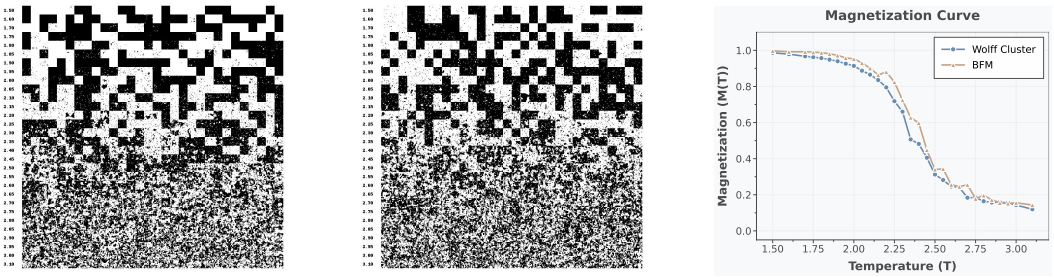


Figure 9: Comparison of 20×20 Ising lattice samples generated by BFM and ground-truth samples at temperatures 2.0 and 2.5.

To obtain reference Ising samples at different temperatures, we use the Wolff cluster algorithm [33]. We consider a 20×20 2D lattice Ising model with periodic boundary conditions, where each spin interacts with its four nearest neighbors. The Markov chain is run for 1000 steps to reach thermal equilibrium. We collect 500 training samples and 100 validation samples at each temperature from 1.50 to 3.10. We then train a conditional BFM for 100K iterations with batch size 512. During inference, we use 64 sampling steps to generate Ising samples from pure noise.

Fig. 9 shows representative comparisons between BFM-generated samples and Wolff-cluster reference samples at temperatures 2.0 and 2.5. BFM generates high-quality Ising configurations that are visually consistent with the reference samples.



(a) BFM Ising Samples (b) Wolff Cluster Ising Samples (c) Magnetization Curve

Figure 10: Comprehensive evaluation of Ising model generation across the continuous temperature spectrum (1.50 to 3.10). (a) and (b) show the visual evolution of spin lattices generated by BFM and the ground-truth Wolff cluster [33] algorithm, respectively. (c) Quantitative comparison of the absolute magnetization $M(T)$, demonstrating that BFM accurately captures the thermodynamic properties.

To further investigate our model's ability to simulate the macroscopic thermodynamic behavior of the Ising system, we extend our evaluation across the entire temperature spectrum in Fig. 10. In statistical physics, visual inspection is often insufficient; a generative model must faithfully capture the underlying Boltzmann distribution. To rigorously substantiate our claim, we calculate the absolute magnetization $M(T)$ of the generated samples as a quantitative physical observable.

Visually, the BFM-generated samples (Fig. 10a) smoothly capture the structural evolution of the spin lattices from highly ordered to completely disordered states, strictly mirroring the Wolff cluster ground truth (Fig. 10b). More importantly, the quantitative magnetization curve of BFM (Fig. 10c) tightly tracks the ground truth simulations. This quantitative alignment proves that BFM learns

the complex physical interaction rules of the system rather than merely memorizing visual training patterns.

D Discussion on Cross-step Sampling Consistency of BFM

In this section, we discuss the reason why BFM exhibits better sampling quality than BLD when using different NFE from the same trained checkpoint. This property is particularly important in practical applications, as it allows for flexible adjustment of sampling speed and quality trade-offs without the need for retraining the model. We follow the notation used in BLD to explain this issue. In the sampling scenario of Fig. 2, when we sample with K NFE from a model trained with N steps, we need to perform cross-step sampling with stride $m = N/K$. According to Eq. (9) in BLD, the posterior transition can be written as

$$q(\mathbf{z}^{t-m}|\mathbf{z}^t, \mathbf{z}^0) = \frac{q(\mathbf{z}^t|\mathbf{z}^{t-m}, \mathbf{z}^0)q(\mathbf{z}^{t-m}|\mathbf{z}^0)}{q(\mathbf{z}^t|\mathbf{z}^0)}. \tag{69}$$

In the official BLD code implementation, the evidence transition term is implemented as

$$q(\mathbf{z}^t|\mathbf{z}^{t-m}, \mathbf{z}^0) = \mathcal{B}(\mathbf{z}^t; \mathbf{z}^{t-m}(1-\beta^{t-m}) + 0.5\beta^{t-m}), \tag{70}$$

which effectively uses only the last-step noise parameter β^{t-m} for this cross-step update; however, the corresponding exact evidence transition should be

$$q(\mathbf{z}^t \mid \mathbf{z}^{t-m}) = \mathcal{B}\left(\mathbf{z}^t; \left(\prod_{j=t-m+1}^{t}(1-\beta^j)\right)\mathbf{z}^{t-m} + 0.5\left(1 - \prod_{j=t-m+1}^{t}(1-\beta^j)\right)\right) \tag{71}$$

In contrast, in the framework of BFM, the cross-step posterior sampling is naturally supported by our derived posterior transition in (3) and (5), which can be directly applied to any arbitrary cross-step posterior transition with slight modification. The only modification is to replace the original N-step decreasing schedule of $t_\tau \in [0, 1]$ from 1 to 0 with a new schedule containing K NFEs at sampling time. We refer to this property as the **self-consistency of BFM under cross-step sampling**.

Here we explain why BFM has this self-consistency property. The evidence transition $q(X_{t_\tau}|X_{t_{\tau-1}}, X_{t_0})$ from $X_{t_{\tau-1}}$ to X_{t_τ} depends on γ_{t_τ}, we use the marginal distribution from X_{t_0} to X_{t_τ} and $X_{t_{\tau-1}}$ to derive γ_{t_τ} and find it only depends on t_τ and $t_{\tau-1}$ as (19) shows. Therefore, when we perform cross-step sampling with different NFE, we can simply re-calculate γ_{t_τ} based on the new t_τ and $t_{\tau-1}$ without any approximation, which ensures the correctness of the posterior transition. This is the key mechanism of BFM's self-consistency property in cross-step sampling. The self-consistency property makes BFM more flexible and robust in cross-step sampling scenarios.

Evaluation with Corrected BLD. We further evaluate BLD [32] using the corrected cross-step transition in Eq. (71). Both BFM and corrected BLD are evaluated on LSUN Churches 256×256 using the same pretrained binary autoencoder, evaluation protocol, and 50,000 generated samples at each NFE.

Table 3: Low-NFE comparison between BFM and corrected BLD on LSUN Churches 256×256. Relative reduction denotes the FID improvement of BFM over corrected BLD.

NFE	BFM FID ↓	Corrected BLD FID ↓	Reduction ↑
16	**10.41**	10.49	0.76%
32	**8.28**	8.37	1.08%
64	**7.78**	7.98	2.51%
128	**7.70**	7.85	1.91%
256	**7.59**	7.79	2.57%

Correcting the BLD cross-step transition substantially improves its low-NFE performance compared with the original implementation. Nevertheless, BFM consistently achieves lower FID across all evaluated NFEs, with relative reductions ranging from 0.76% to 2.57%.

These results indicate that the severe low-NFE degradation observed in the original BLD implementation is largely attributable to its cross-step transition approximation, rather than an inherent limitation of discrete Markov diffusion. BFM remains competitive with the corrected baseline while retaining a direct analytical construction for arbitrary time intervals.

E Additional Implementation Details

In this section, we provide detailed network configurations and training hyperparameters for all tasks, as summarized in Table 4. For high-dimensional image synthesis (LSUN and FFHQ), the training process operates within a discrete latent space. Regarding these latent representations, we utilize the official pre-trained Binary Autoencoder (BAE) provided by BLD [32]. Rather than training a dedicated autoencoder for each specific dataset, we adopt their general-purpose **C64** model, which was trained on over 600 million images from the LAION dataset. This robust autoencoder explicitly compresses a high-resolution 256×256 image into a compact $16 \times 16 \times 64$ binary tensor, providing a highly generalized discrete space for our Bernoulli Flow Models. The pre-trained checkpoints for this autoencoder (including both the encoder and decoder) are officially provided by the BLD authors and can be accessed publicly[1]. To ensure a fair comparison and robust performance in modeling these binary latents, our core generative network directly adopts the Transformer-based backbone proposed in BLD. While this standard configuration is seamlessly applied to the latent image representations, we introduce minor structural adaptations, specifically to the sequence length and positional embedding layers, to properly accommodate the unique spatial formats of Binarized MNIST and the 2D Ising model. Unlike the high-dimensional image tasks, Binarized MNIST [8] and the Ising model [15] do not rely on any encoder-decoder mappings; our models are trained directly on their native binary spaces. For the Ising model specifically, the spin configurations were synthetically simulated using the Wolff cluster [33] algorithm, yielding a custom dataset of 15000 training samples and 3000 testing samples. Across all datasets, we maintain a consistent optimization strategy using the Adam optimizer coupled with a linear learning rate warmup.

Table 4: Configuration and training details for BFM across different datasets.

Component	**LSUN & FFHQ**	**Binarized MNIST**	**Ising Model**
Transformer layers	24	24	24
Attention heads	12	12	12
Embedding dimension	768	768	768
Sequence length (block size)	256	28	20
Training iterations	500K	500K	100K
Batch size	96	512	512
Optimizer	Adam	Adam	Adam
Learning rate	2×10^{-4}	2×10^{-4}	2×10^{-4}
Weight decay	0.0	0.0	0.0
Warmup iterations	10,000	10,000	10,000
Hardware	$2\times$ RTX 4090	$2\times$ RTX 4090	$2\times$ RTX 4090

E.1 Compute Resources and Inference Efficiency

To improve reproducibility and address the computational resource requirements of our experiments, we further report the inference-time compute cost for each task. All measurements are conducted with batch size 1 and 64 sampling steps, corresponding to the single-sample generation latency rather than throughput under large-batch sampling. For each benchmark, we first perform 5 warm-up runs to exclude CUDA initialization and caching overhead, and then average the wall-clock time over 20 repeated runs. We enforce CUDA synchronization before and after each measured run. For LSUN and FFHQ, we report a shared Image-256 setting because both tasks use the same 256×256 image resolution and the same $16 \times 16 \times 64$ binary latent representation produced by the pre-trained Binary Autoencoder (BAE). This benchmark includes both binary latent sampling and decoding through the BAE decoder. In contrast, Binarized MNIST and the Ising model are sampled directly in their native binary spaces without any encoder-decoder module.

[1] `https://drive.google.com/drive/folders/1CFlixXLnEHZ0jaRLXLS4d_bJMZwG-Iih`

Table 5: **Single-sample inference resource usage.** We report the average wall-clock latency and peak GPU memory for one generated sample using 64 sampling steps. The Image-256 setting is shared by LSUN and FFHQ since they use the same image resolution and binary latent format.

Task	Representation	Batch Size	Sampling Steps	Avg. Time / Sample (s)	Peak GPU Memory (GB)
LSUN / FFHQ Image-256	Binary latent + BAE decoder	1	64	0.3837	1.0743
Binarized MNIST	Native binary space	1	64	0.3507	0.6482
Ising Model	Native binary space	1	64	0.3531	0.6480

As shown in Table 5, all tasks can be sampled with less than 1.1 GB of peak GPU memory under the single-sample setting. The Image-256 benchmark has slightly higher latency and memory consumption because it operates on a higher-dimensional binary latent space and additionally invokes the BAE decoder to reconstruct RGB images. Binarized MNIST and Ising require less memory since they are generated directly in low-dimensional binary spaces. These measurements indicate that our 64-step sampler remains lightweight at inference time across both image-latent generation and native binary-domain generation.

F Bootstrap Uncertainty Analysis of FID

To assess the statistical uncertainty of the FID comparisons, we perform bootstrap analysis on LSUN Bedrooms, LSUN Churches, and FFHQ. For each method, we generate 50,000 samples and perform 50 bootstrap resamples of the generated Inception features with replacement. Within each dataset, BFM and BLD use identical real-data statistics, feature extractors, sample counts, and bootstrap protocols.

Table 6: Bootstrap uncertainty analysis of FID on LSUN Bedrooms, LSUN Churches, and FFHQ. We report the full-sample FID, bootstrap mean $\pm$ standard deviation, and 95% bootstrap percentile interval from 50 resamples.

Dataset	Method	Full FID $\downarrow$	Boot. Mean $\pm$ Std. $\downarrow$	95% Interval
Bedrooms	BFM	**5.86**	$\mathbf{6.14 \pm 0.04}$	[6.03, 6.22]
	BLD	6.21	6.48 ± 0.04	[6.39, 6.55]
Churches	BFM	**5.32**	$\mathbf{5.59 \pm 0.04}$	[5.50, 5.67]
	BLD	5.47	5.71 ± 0.03	[5.66, 5.77]
FFHQ	BFM	**10.87**	$\mathbf{11.21 \pm 0.08}$	[11.11, 11.36]
	BLD	11.46	11.79 ± 0.07	[11.71, 11.94]

As shown in Table 6, BFM consistently achieves lower bootstrap mean FID than BLD across all three datasets. All bootstrap standard deviations are below 0.08, indicating limited finite-sample variability. The 95% intervals are separated on Bedrooms and FFHQ, while those on Churches overlap slightly.

The bootstrap means are higher than the corresponding full-sample FIDs because resampling with replacement reduces the number of unique generated samples. This effect applies to both methods under the same evaluation protocol.

Overall, the bootstrap results preserve the relative performance ranking between BFM and BLD under finite-sample resampling.

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: The claims made in the abstract and introduction are fully supported by the theoretical framework in Section 3 (with detailed proofs in Appendix A) and the empirical evaluations in Section 4.

   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: The paper includes a discussion of limitations in Section 5.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [Yes]

Justification: For each theoretical result, we provide a complete set of assumptions and a correct proof. Detailed proofs for our model architecture are provided in Appendix A.

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: We fully disclose all necessary information to reproduce our main experimental results. Detailed descriptions of the experimental setups, training procedures, and model architectures are provided in Section 4.1, with comprehensive hyperparameters, dataset configurations, and compute resources further detailed in Appendix E.

Guidelines:

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

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

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

   Answer: [Yes]

   Justification: We will provide data, code, and instructions for the final paper.

   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: We fully disclose the necessary information to reproduce our main experimental results. Comprehensive details regarding the training procedures, model architectures, and hyperparameters are provided in Section 4.1 (Experimental Setup) and Appendix E (Additional Implementation Details).

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: We evaluate our models using standard generative metrics (e.g., FID, Precision, Recall) computed over a massive set of 50,000 samples to ensure statistical stability. We do not report error bars across multiple training runs due to the prohibitive computational cost of repeatedly training and sampling large-scale models.

   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: Detailed information regarding the compute resources is provided in Appendix E and Appendix E.1.

   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 go through the NeurIPS Code of Ethics and confirm that we adhere to them.

   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: [No]

Justification: While Section 5 discusses the positive scientific potential of our framework (e.g., modeling social thermodynamics and biological systems), we do not explicitly discuss negative societal impacts. As foundational algorithmic research, our work focuses on mathematical derivations and scientific simulations rather than a deployable application, thus limiting immediate direct paths to societal harm.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

Answer: [N/A]

Justification: Our work focuses on the fundamental theoretical dynamics of binary generative modeling. The models evaluated are proofs-of-concept trained on standard academic benchmarks (e.g., Ising models, Binarized MNIST, LSUN) rather than large-scale, high-fidelity deployable generators, and thus do not pose a high risk for malicious misuse that would necessitate specific release safeguards.

Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All existing assets utilized in our experiments, including standard publicly available datasets (e.g., LSUN, FFHQ, Binarized MNIST) and open-source codebases (e.g., BLD), are properly credited with citations in Section 4 and are used in accordance with their respective open-source research licenses.

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: The paper does not release any new assets.

    Guidelines:

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

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

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

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with 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: The paper does not involve crowdsourcing nor research with 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 an LLM solely for editing assistance.

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.