Unifying Bayesian Flow Networks and Diffusion Models through Stochastic Differential Equations

Kaiwen Xue [* 1] **Yuhao Zhou** [* 2] **Shen Nie** [1] **Xu Min** [3]
Xiaolu Zhang [3] **Jun Zhou** [3] **Chongxuan Li** [1]

Abstract

Bayesian flow networks (BFNs) iteratively refine the parameters, instead of the samples in diffusion models (DMs), of distributions at various noise levels through Bayesian inference. Owing to its differentiable nature, BFNs are promising in modeling both continuous and discrete data, while simultaneously maintaining fast sampling capabilities. This paper aims to understand and enhance BFNs by connecting them with DMs through stochastic differential equations (SDEs). We identify the linear SDEs corresponding to the noise-addition processes in BFNs, demonstrate that BFN's regression losses are aligned with denoise score matching, and validate the sampler in BFN as a first-order solver for the respective reverse-time SDE. Based on these findings and existing recipes of fast sampling in DMs, we propose specialized solvers for BFNs that markedly surpass the original BFN sampler in terms of sample quality with a limited number of function evaluations (e.g., 10) on both image and text datasets. Notably, our best sampler achieves an increase in speed of $5 \sim 20$ times for free. Our code is available at https://github.com/ML-GSAI/BFN-Solver.

1. Introduction

Deep generative models (DGMs) are effective in capturing complex data distributions and producing realistic samples, substantially influencing fields such as computer vision (Rombach et al., 2022; Ramesh et al., 2022; Podell et al., 2023) and natural language processing (Brown et al., 2020; OpenAI, 2023). The fundamental challenge in DGMs is to represent a flexible probability distribution that facilitates effective parameter learning and efficient inference simultaneously, greatly depending on the data (or modality).

Autoregressive models (ARMs) (OpenAI, 2023), for example, excel in modeling sequential and discrete data (e.g., text) but face limitations in the inference speed, which is proportional to the number of variables. Diffusion models (DMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021), on the other hand, better balance generation quality and efficiency with a coarse-to-fine approach. Although considered state-of-the-art in image generation, DMs encounter challenges in handling discrete variables, where score matching algorithms (Hyvärinen, 2005; Vincent, 2011) do not directly apply.

A new class of generative models, Bayesian Flow Networks (BFNs) (Graves et al., 2023), has been developed recently to overcome these challenges. While inspired by DMs, BFNs distinguish themselves by focusing on iteratively refining the parameters (instead of the samples) of a distribution set at different noise levels through Bayesian inference (see Sec. 3 for more details). This strategy enables BFNs to facilitate fast sampling and maintain a continuous nature, even when processing discrete data. With carefully designed regression losses, BFNs have shown considerable promise in both image and language modeling. Notably, BFN is primarily developed based on a message-sending process with minimum communication length, and the exact relation between BFNs and DMs remains unclear.

As summarized in Table 1, this paper primarily contributes by unifying BFNs and DMs through stochastic differential equations (SDEs), a pivotal step in understanding their relationship and enhancing BFNs. Initially, by slightly truncating the time, we identify linear SDEs corresponding to the noise-adding processes in BFN on both continuous (see Sec. 4) and discrete data (see Sec. 5) and derive the reverse-time SDEs for sampling. Note that the SDEs for discrete data operate on a set of latent variables, which the original BFN formulation marginalizes out, rather than distribution parameters. Furthermore, we demonstrate that, especially

---
[*]Equal contribution [1]Gaoling School of AI, Renmin University of China, Beijing, China [2]Department of Computer Science and Technology, Tsinghua University, Beijing, China [3]Ant Group, Hangzhou, China. Correspondence to: Chongxuan Li <chongxuanli@ruc.edu.cn>.

Table 1. Technical contributions of the paper include the theory on unifying BFN and DM (in red) and new samplers for BFN inspired by the theory (in blue). "SDE-solver1" means a first-order solver for the corresponding SDE and "Approx." is a shorthand for "Approximate".

	NOISE-ADDING PROCESS	LOSS FUNCTION	ORIGINAL SAMPLER	NEW SAMPLERS
BFN ON CONTINUOUS DATA	CORRESPONDING SDE THEOREM 4.1	EQUIVALENT TO DSM TRIVIAL	SDE-SOLVER1 PROPOSITION 4.2	BFN-SOLVERS ALGOS. 1-3 IN APPENDIX
BFN ON DISCRETE DATA	CORRESPONDING SDE THEOREM 5.1	EQUIVALENT TO DSM THEOREM 5.2	APPROX. SDE-SOLVER1 PROPOSITION 5.3	BFN-SOLVERS ALGOS. 4-7 IN APPENDIX

on discrete data, BFN's regression losses align with denoising score matching (DSM) (Vincent, 2011) w.r.t. variables in the corresponding SDE, positioning the trained networks to naturally parameterize the reverse-time SDEs. Finally, the original BFN sampler is proven as an (approximate) first-order solver for the corresponding reverse-time SDE.

The explicit connection between BFNs and DMs brings immediate benefits, particularly in applying fast sampling methods (Lu et al., 2022b;c) from DMs to BFNs. We derive the corresponding probability flow ordinary differential equations (ODEs) (Song et al., 2021) for BFNs on both continuous and discrete data. We propose high-order solvers (named *BFN-Solvers*) tailored to BFNs' special (e.g., semilinear) structure, for both SDEs and ODEs. Empirically, using the same pre-trained model, our best solver significantly outperforms the original BFN sampler with a few (e.g., 10) number of function evaluations (NFE) under sample quality on both the CIFAR10 and text8 datasets, achieving a $5 \sim 20$ times increase in speed for free (see Sec. 6 for details).

We believe our discovery offers a rigorous and systematic perspective for analyzing and improving the training and inference processes of BFNs, grounded in the existing results of DMs, and may inspire future work as detailed in Sec. 7.

2. Related Work

Score-baesd DMs. Built upon the score matching algorithms (Hyvärinen, 2005; Vincent, 2011; Song et al., 2019; Pang et al., 2020), DMs (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021) are currently SOTA to model continuous variables (Dhariwal & Nichol, 2021; Chen et al., 2020; Kong et al., 2020; Ho et al., 2022; Singer et al., 2022; Poole et al., 2022; Wang et al., 2023). In particular, large-scale text-to-image models (Rombach et al., 2022; Ramesh et al., 2022; Saharia et al., 2022; Bao et al., 2023; Balaji et al., 2023; Xue et al., 2023b; Podell et al., 2023) have made remarkable progress and attracted significant attention.

Solvers for DMs. Since Song et al. (2021) introduced the SDE and probability flow ODE formulation of DMs, there have been extensive solvers for both SDE (Ho et al., 2020; Song et al., 2021; Karras et al., 2022; Lu et al., 2022c; Bao

et al., 2022b;a; Jolicoeur-Martineau et al., 2021; Xue et al., 2023a; Guo et al., 2023) and ODE (Song et al., 2020; Liu et al., 2022; Lu et al., 2022b;c; Zhang et al., 2022; Karras et al., 2022; Zhao et al., 2023) to improve the sampling process. In particular, ODE samplers are proven effective with limited NFEs while SDE samplers are robust to prior mismatch (Lu et al., 2022a; Nie et al., 2023) and perform better in a sufficient number of NFEs (Lu et al., 2022c).

Discrete DMs. Several DMs have been proposed to model discrete data with discrete states (Sohl-Dickstein et al., 2015; Hoogeboom et al., 2021; Austin et al., 2023), depending on a probability transition matrix. It is nontrivial to leverage the features associated with continuous-state DMs, such as guidance and ODE fast sampling. Efforts have been made to define the score in the discrete state (Lou et al., 2023; Meng et al., 2023; Campbell et al., 2022; Sun et al., 2023); however, this remains a challenging endeavor. Other works (Chen et al., 2022; Dieleman et al., 2022; Li et al., 2022) have attempted to identify a continuous equivalent for discrete data and apply continuous DMs, but this may result in information loss during the transformation and greatly rely on the noise schedule (Ye et al., 2023). Mahabadi et al. (2023) defines a continuous-time diffusion process on continuous latent variables but is trained with cross-entropy loss rather than regression loss. Several studies (Richemond et al., 2022; Lou & Ermon, 2023) have attempted to establish the diffusion process using SDEs on discrete data. Specifically, Richemond et al. (2022) introduced an SDE defined on the probability simplex, but it suffers from intractability in high-dimensional space. Lou & Ermon (2023) proposed a diffusion SDE with an additional boundary constraint, which also increases the complexity of discretization (e.g., requiring thresholding in SDE).

In comparison, this paper reveals that BFNs applied to discrete data solve a linear SDE and are trained using DSM, which aligns seamlessly with continuous DMs. Consequently, without changing the discrete data, BFNs are significantly simpler and more scalable and efficient than the related work, leveraging advancements in continuous DMs.

3. Background

In this section, we present the elementary notations and background of DMs and BFNs.

3.1. Elementary Notations

We use lowercase letters (e.g., t) and boldface lowercase letters (e.g., $\mathbf{x}$) to denote scalars and vectors respectively. Variables indexed by uncountable indices are denoted in the form of functions, (e.g., $\beta(t)$ and $\boldsymbol{\mu}(t)$). Given finite indices (e.g. $\{t_i\}_{i=1}^M$), the corresponding variables are denoted with subscripts (e.g., $\boldsymbol{\mu}_i$).

3.2. Score-based DMs

Score-based DMs (Kingma et al., 2021) characterize the data distribution through a diffusion process $\{\mathbf{x}(t) \sim \mathcal{N}(\alpha(t)\mathbf{x}, \sigma^2(t)\boldsymbol{I})\}$ indexed by a continuous-time variable $t \in [0, T]$ according to an Itô SDE as follows

$$\mathrm{d}\mathbf{x} = f(t)\mathbf{x}\,\mathrm{d}t + g(t)\,\mathrm{d}\boldsymbol{w}, \tag{1}$$

where $\boldsymbol{w}$ is the standard Wiener process, and $f(t) = \frac{\mathrm{d}\log\alpha(t)}{\mathrm{d}t}$ and $g(t) = \frac{\mathrm{d}\sigma^2(t)}{\mathrm{d}t} - \frac{1}{2}\frac{\mathrm{d}\log\alpha(t)}{\mathrm{d}t}\sigma^2(t)$ are the drift and diffusion coefficients respectively. For instance, denoising diffusion probabilistic models (Ho et al., 2020) consider a process given by the following SDE:

$$\mathrm{d}\mathbf{x} = -\frac{1}{2}\beta(t)\mathbf{x}\,\mathrm{d}t + \sqrt{\beta(t)}\,\mathrm{d}\boldsymbol{w}, \tag{2}$$

where $0 < \beta(t) < 1$. Let $p_t(\mathbf{x})$ denote the marginal density of $\mathbf{x}(t)$. The generative process of score-based DMs is given by a reverse-time SDE (Song et al., 2021; Anderson, 1982)

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

where $\bar{\boldsymbol{w}}$ is the time-reversed Wiener process. Then the score is parameterized with a time-dependent score-based model $\hat{\boldsymbol{s}}(\mathbf{x}, t)$ and trained with the following denoising score matching loss (Vincent, 2011)

$$\mathcal{L}_{\mathrm{DSM}} = \mathbb{E}_{\mathbf{x}(t),\mathbf{x}(0)}[\|\hat{\boldsymbol{s}}(\mathbf{x}(t), t) - \nabla_{\mathbf{x}} \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\|_2^2], \tag{4}$$

where the conditional distribution $p_{0t}(\mathbf{x}|\mathbf{x}(0))$ is designed as a Gaussian kernel with a closed form score function $\nabla_{\mathbf{x}} \log p_{0t}(\mathbf{x}|\mathbf{x}(0))$. For fast sampling, Song et al. (2021) introduce the corresponding *probability flow ODE* of the reverse SDE in Eq. (3) as follows

$$\mathrm{d}\mathbf{x} = \left[f(t)\mathbf{x} - \frac{1}{2}g(t)^2 \nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right]\mathrm{d}t, \tag{5}$$

which produces the same data distribution as the corresponding SDE with infinitesimally small stepsize and enjoys a

smaller discretization error with a large stepsize due to its deterministic nature (Kloeden et al., 1992). To solve the ODE in Eq. (5) efficiently, DPM-Solvers (Lu et al., 2022b;c) explicitly leverage the semi-linear property of Eq. (5) and further simplify it to an exponentially weighted integral of the neural network by applying change-of-variable. Consequently, the exact solution of ODE is given by

$$\mathbf{x}(t) = \frac{\alpha(t)}{\alpha(s)}\mathbf{x}(s) - \alpha(t)\int_{\lambda(s)}^{\lambda(t)} e^{-\lambda}\hat{\boldsymbol{e}}_{\theta}(\hat{\mathbf{x}}(\lambda), \lambda)\,\mathrm{d}\lambda, \tag{6}$$

where $\lambda(t) = \log(\alpha(t)/\sigma(t))$ is the half of the log signal-noise ratio. DPM-Solver solves Eq. (6) numerically leading to a small discretization error. Taking DPM-Solver1 as an example, given time steps $\{t_i\}_{i=1}^n$ and initial value $\mathbf{x}_0$, a sequence $\{\mathbf{x}_i\}_{i=1}^n$ can be solved iteratively as follows:

$$\mathbf{x}_i = \frac{\alpha(t_i)}{\alpha(t_{i-1})}\mathbf{x}_{i-1} - \sigma(t_i)(e^{h_i} - 1)\hat{\boldsymbol{\epsilon}}_{\theta}(\mathbf{x}_{i-1}, t_{i-1}) + \mathcal{O}(h_i^2), \tag{7}$$

where $h_i = \lambda(t_i) - \lambda(t_{i-1})$. Empirically, DPM-Solver achieves excellent results with a limited number of NFEs and is widely adopted.

3.3. Bayesian Flow Networks

Due to the space limit, we briefly present the motivation and formulation of BFNs (Graves et al., 2023) here and please refer to the original paper for more details. Inspired by DMs, BFNs iteratively refine the parameters of a distribution set at different noise levels through Bayesian inference. This strategy enables BFNs to facilitate fast sampling and be differentiable on both continuous and discrete data.

For D-dimensional continuous data[1] $\mathbf{x} \in \mathbb{R}^D$, a continuous-time BFN operates on parameters of a set of Gaussian distributions (of noisy data with different noise levels) with means $\{\boldsymbol{\mu}(t)\}_{t=0}^1$ and covariance matrices $\{\rho(t)\boldsymbol{I}\}_{t=0}^1$. Equivalently, $\boldsymbol{\mu}(t)$ can also be regarded as a noisy version of $\mathbf{x}$ by injecting a Gaussian noise and follows the distribution

$$q_{F}(\boldsymbol{\mu}(t)|\mathbf{x}, \gamma(t)) = \mathcal{N}(\gamma(t)\mathbf{x}, \gamma(t)(1 - \gamma(t))\boldsymbol{I}), \tag{8}$$

where $\gamma(t) = 1 - \sigma_1^{2(1-t)}$ is a schedule function[2] and $\sigma_1 \in (0, 1)$ is a hyperparameter. $\rho(t)$ is has a closed form as $\rho(t) = \frac{1}{1-\gamma(t)}$. Similar to DMs, a BFN on continuous data trains a neural network $\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t)$ to predict the injected

---

[1]We say $\mathbf{x}$ is a continuous data if its distribution has density w.r.t. the Lebesgue measure.

[2]For a clear alignment with DMs, we adopt a reverse time notation in this paper as originally used by Graves et al. (2023). Specifically, the schedule $\gamma(t)$ in our paper is equivalent to $\gamma(1-t)$ in Graves et al. (2023). We retain the other notational conventions for ease of reading, which do not affect our derivations.

Gaussian noise ϵ by minimizing the following loss:

$$\mathop{\mathbb{E}}_{q_F(\boldsymbol{\mu}(t)|\mathbf{x},\gamma(t)),t\sim U(0,1)} -\frac{\ln\sigma_1}{\sigma_1^{2t}}\|\boldsymbol{\epsilon}-\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t),t)\|^2. \quad (9)$$

Given time steps $\{t_i\}_{i=0}^n$ and i.i.d. noises $\{\mathbf{u}_i\}_{i=0}^n \sim \mathcal{N}(0,\boldsymbol{I})$, the BFN sampler (Graves et al., 2023) iterates as follows.

$$\boldsymbol{\mu}_i = -\frac{\gamma(t_i)-\gamma(t_{i-1})}{\sqrt{\gamma(t_{i-1})(1-\gamma(t_{i-1}))}}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}_{i-1},t_{i-1}) + \frac{\gamma(t_i)}{\gamma(t_{i-1})}\boldsymbol{\mu}_{i-1}$$

$$+\sqrt{\frac{1-\gamma(t_i)}{1-\gamma(t_{i-1})}(\gamma(t_i)-\gamma(t_{i-1}))}\mathbf{u}_i. \quad (10)$$

On D-dimensional discrete data $\mathbf{x} \in \{1,\cdots,K\}^D$, where K is the number of classes, the BFN operates on parameters $\boldsymbol{\theta}(t)$ of the multivariate categorical distributions of noisy data. The distribution of $\boldsymbol{\theta}$ is

$$q_F(\boldsymbol{\theta}(t)|\mathbf{x},\beta(t)) = \mathop{\mathbb{E}}_{q(\mathbf{z}(t)|\mathbf{x},\beta(t))} \delta(\boldsymbol{\theta}(t)-\mathrm{softmax}(\mathbf{z}(t))),$$

where $\delta(\cdot)$ is the Dirac distribution, $\mathbf{z}(t)$ is a set of latent variables with Gaussian marginal distributions as

$$q(\mathbf{z}(t)|\mathbf{x},\beta(t)) = \mathcal{N}(\beta(t)\boldsymbol{w}_\mathbf{x}, K\beta(t)\boldsymbol{I}), \quad (11)$$

and $\boldsymbol{w}_\mathbf{x} := K\boldsymbol{e}_\mathbf{x} - \mathbf{1}$, $\boldsymbol{e}_\mathbf{x} := \{\boldsymbol{e}_{\mathbf{x}^{(1)}},\cdots,\boldsymbol{e}_{\mathbf{x}^{(D)}}\} \in \mathbb{R}^{KD}$ where $\boldsymbol{e}_j$ is the one-hot vector defined by $(\boldsymbol{e}_j)_k = \delta_{\mathbf{x}_j k}$ and $\mathbf{1}$ is a vector of length KD filled with ones. $\beta(t) = (1-t)^2\beta_1$ is a schedule function with a hyperparameter $\beta_1 > 0$. A BFN on discrete data trains a neural network $\hat{\boldsymbol{e}}(\boldsymbol{\theta}(t),t)$ that predicts the data in a one-hot form given noisy inputs using the following regression loss

$$\mathcal{L}^\infty(\mathbf{x}) = \mathop{\mathbb{E}}_{q_F(\boldsymbol{\theta}|\mathbf{x},t),t\sim U(0,1)} K\beta_1 t\|\boldsymbol{e}_\mathbf{x}-\hat{\boldsymbol{e}}(\boldsymbol{\theta}(t),t)\|^2. \quad (12)$$

Let $\{\mathbf{u}_i\}_{i=0}^n \sim \mathcal{N}(0,\boldsymbol{I})$ be independent and use $\hat{\boldsymbol{e}}_s(\mathbf{z}(t),t)$ as a shorthand for $\hat{\boldsymbol{e}}(\mathrm{softmax}(\mathbf{z}(t)),t)$. The sampling rule of BFN (Graves et al., 2023) can be written as follows

$$\boldsymbol{e}_k \sim \mathrm{Cat}(\hat{\boldsymbol{e}}_s(\mathbf{z}_{i-1},t_{i-1})), \quad (13)$$

$$\mathbf{z}_i = \mathbf{z}_{i-1} + \alpha_i(K\boldsymbol{e}_k-1) + \sqrt{K\alpha_i}\mathbf{u}_i, \quad (14)$$

where $\alpha_i = \beta(t_i) - \beta(t_{i-1})$ and Cat represents the one-hot categorical distribution.[3]

Based on the formulation, BFNs have shown considerable promise in both image and language modeling. Although inspired by DMs, and the exact relation between BFNs and DMs remains unclear. To this end, this paper unifies them through stochastic differential equations (SDEs) for understanding and accelerating BFNs on both continuous data (see Sec. 4) and discrete data (see Sec. 5).

---

[3]Originally, Graves et al. (2023) obtain samples through $\boldsymbol{\theta}(t)$, while we present the equivalent form in terms of $\mathbf{z}(t)$ for convenience.
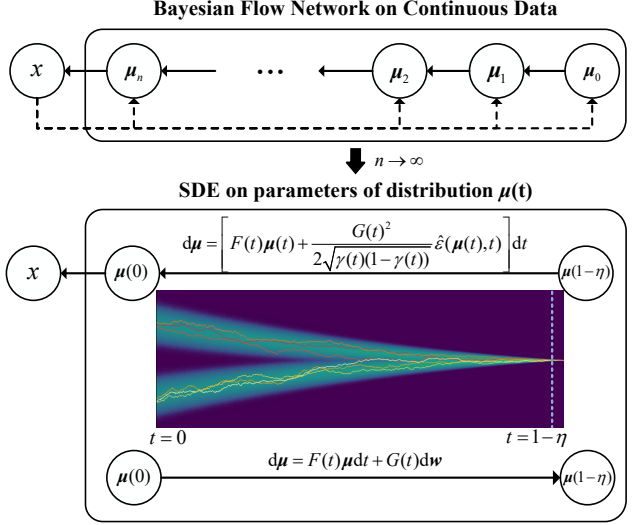

Figure 1. **Illustration of BFN on continuous data and the corresponding SDEs.** The SDEs are defined w.r.t. $\boldsymbol{\mu}$ on time $[0, 1-\eta]$.

4. Continuous-time BFN on Continuous Data

This section bridges BFNs on continuous data with DMs by establishing a linear SDE for noise modeling in BFN (Sec. 4.1), aligning training objectives with DSM (Sec. 4.2), and validating the sampler as discretization of the reverse-time SDE (Sec. 4.3). Further, fast samplers are developed based on the recipe in DMs in Sec. 4.4.

4.1. Formulating BFN on Continuous Data as SDEs

As illustrated in Fig. 1, we establish that the (truncated) noise-adding process of the continuous-time BFN on continuous data in Eq. (8) uniquely solves a linear SDE, summarized as follows.

Theorem 4.1 (Proof in Appendix A.1)**.** *Let $\eta > 0$ be an arbitrarily small constant. The BFN in Eq. (8) at time $[0, 1-\eta]$ is the unique solution of the following linear SDE:*

$$\mathrm{d}\boldsymbol{\mu} = F(t)\boldsymbol{\mu}\,\mathrm{d}t + G(t)\,\mathrm{d}\boldsymbol{w}. \quad (15)$$

Here $\boldsymbol{w}$ is a standard Wiener process and

$$F(t) = \frac{\gamma'(t)}{\gamma(t)} = 2\frac{\sigma_1^{2(1-t)}}{1-\sigma_1^{2(1-t)}}\ln\sigma_1, \quad (16)$$

$$G(t)^2 = -\gamma'(t) = -2\sigma_1^{2(1-t)}\ln\sigma_1, \quad (17)$$

where $\sigma_1 \in (0,1)$ is the hyperparameter defined in Eq. (8).

The time t is truncated by $1-\eta$ in Theorem 4.1 for two reasons. On one hand, the reverse-time SDE derived later (see Eq. (20)) is ill-defined at $t=1$ since the distribution of $\boldsymbol{\mu}$ collapses to a Dirac distribution whose score tends to infinity. On the other hand, it is convenient to satisfy certain

regularity conditions for the uniqueness of the solution in Theorem 4.1, as detailed in the proof. The exact distribution of $\boldsymbol{\mu}(1 - \eta)$ is unknown. We approximate it by an isotropic Gaussian with zero mean and small variance, as $p(\boldsymbol{\mu}(t))$ tends towards a Dirac delta function when $t \to 1$. (see details in Sec. 6). As η is small (e.g., $10^{-3} \sim 10^{-5}$) in our implementation, the effect of truncation is negligible.

Here a linear SDE also applies to the latent variable $\mathbf{z}$, a linear transformation of $\boldsymbol{\mu}$ in Eq. (8). The choice of $\boldsymbol{\mu}$ in Theorem 4.1 aligns with the sampling process in BFN (Graves et al., 2023), facilitating a later analysis in Sec. 4.3.

The finding in Theorem 4.1 directly connects to DMs (Song et al., 2021; Kingma et al., 2021), which are formulated as an SDE in Eq. (1) with a different noise schedule. We believe this may inspire new classes of BFNs and leave a systematic comparison of the schedules for future work.

Similar to Eq. (3), the linear SDE in Eq. (15) has an associated reverse-time SDE (Anderson, 1982; Song et al., 2021) in $[0, 1 - \eta]$ for generative modeling:

$$\mathrm{d}\boldsymbol{\mu} = [F(t)\boldsymbol{\mu} - G(t)^2 \nabla_{\boldsymbol{\mu}} \log p_t(\boldsymbol{\mu})] \,\mathrm{d}t + G(t) \,\mathrm{d}\bar{\boldsymbol{w}}, \quad (18)$$

where $\nabla_{\boldsymbol{\mu}} \log p_t(\boldsymbol{\mu})$ is the (time-conditional) score function to be estimated and $\bar{\boldsymbol{w}}$ is the time-reversed Wiener process.

4.2. Training as Parameterizing the Reverse-time SDE

The continuous-time BFN on continuous data trains a neural network to optimize the mean square error in Eq. (9), which directly aligns with the widely employed DSM loss in Eq. (4). In other words, BFN equivalently parameterizes the reverse-time SDE in Eq. (18) by estimating the time-conditional score function as

$$\hat{\boldsymbol{s}}(\boldsymbol{\mu}(t), t) = -\frac{1}{\sqrt{\gamma(t)(1 - \gamma(t))}} \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t), \quad (19)$$

where $\hat{\boldsymbol{s}}(\boldsymbol{\mu}(t), t)$ and $\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t)$ denote the estimate of the score function and the network trained by BFN, respectively, and $\gamma(t)$ follows Eq. (8).

4.3. Sampling as Discretizing the Reverse-time SDE

Plugging Eq. (19) into Eq. (18), we get a parameterized reverse-time SDE in $[0, 1 - \eta]$ for sampling as follows

$$\mathrm{d}\boldsymbol{\mu} = \left[F(t)\boldsymbol{\mu}(t) + \frac{G(t)^2 \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t)}{\sqrt{\gamma(t)(1 - \gamma(t))}} \right] \mathrm{d}t + G(t) \,\mathrm{d}\bar{\boldsymbol{w}}, \quad (20)$$

which is ill-defined at $t = 1$ because $\lim_{t \to 1} \gamma(t) = 1$. Interestingly, even without an explicit SDE formulation, the sampler proposed in the original BFN paper discretizes the reverse-time SDE, as characterized in the following Proposition 4.2.

Proposition 4.2 (Proof in Appendix A.2). *The BFN sampler in Eq. (10) is a first-order discretization of an equivalent form of the parameterized reverse-time SDE in Eq. (20).*

4.4. Probability Flow ODE and Faster Sampling

Establishing an explicit connection between BFNs and DMs through SDEs yields an immediate and significant benefit: the fast sampling recipe from DMs directly applies to BFN.

Formally, according to Eq. (5), we obtain the following equivalent probability flow ODE of the parameterized reverse-time SDE of Eq. (20):

$$\mathrm{d}\boldsymbol{\mu} = \left[F(t)\boldsymbol{\mu}(t) + \frac{G(t)^2}{2\sqrt{\gamma(t)(1 - \gamma(t))}} \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t) \right] \mathrm{d}t. \quad (21)$$

Further, we propose *BFN-Solver*, a customized ODE solver for BFN in analogy to DPM-Solver in Eq. (7). As detailed in Appendix A.3, we integrate all linear terms and apply a change of variable from t to $\lambda(t) = \frac{1}{2} \log \frac{\gamma(t)}{1 - \gamma(t)}$ to obtain a simplified exact solution of Eq. (21)

$$\boldsymbol{\mu}(t) = \frac{\gamma(t)}{\gamma(s)} \boldsymbol{\mu}(s) - \gamma(t) \int_{\lambda(s)}^{\lambda(t)} e^{-\lambda} \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda)) \,\mathrm{d}\lambda, \quad (22)$$

where $t_\lambda(\cdot)$ is the inverse function of $\lambda(t)$ for $0 \leq t < s < 1 - \eta$. Eq. (22) differs from Eq. (6) only in certain coefficients. Given an initial value $\boldsymbol{\mu}_0$ and time steps $\{t_i\}_{i=0}^n$ from $t_0 = 1 - \eta$ to $t_n = 0$, BFN-Solver1 is derived similarly to Eq. (7) and given by

$$\begin{aligned} \boldsymbol{\mu}_i = &-\sqrt{\gamma(t_i)(1 - \gamma(t_i))}(e^{h_i} - 1)\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}_{i-1}, t_{i-1}) \\ &+ \frac{\gamma(t_i)}{\gamma(t_{i-1})} \boldsymbol{\mu}_{i-1}, \end{aligned} \quad (23)$$

where $h_i = \lambda(t_i) - \lambda(t_{i-1})$. *BFN-Solver++* shares the same spirit with BFN-Solver and the difference is that BFN-Solver++ considers data prediction instead of noise prediction. We refer the readers to Appendix A.3 for higher-order solvers of both ODE and SDE.[4]

Empirically, as presented in Sec. 6.2, BFN-Solvers of different orders significantly outperform the original BFN sampler with a limited number of NFEs based on the same model.

5. Continuous-time BFN on Discrete Data

In a manner akin to Sec. 4, this section unifies BFNs on discrete data and (continuous) DMs through SDEs and de-

---

[4]A more straightforward way to get BFN-Solver on continuous data is to treat BFN as a DM with a special noise schedule $\alpha_t = \gamma_t$ and $\sigma_t^2 = \gamma_t(1 - \gamma_t)$. However, it is infeasible on discrete data. Therefore, we use a slightly complex yet coherent way to derive BFN-Solver throughout the paper.
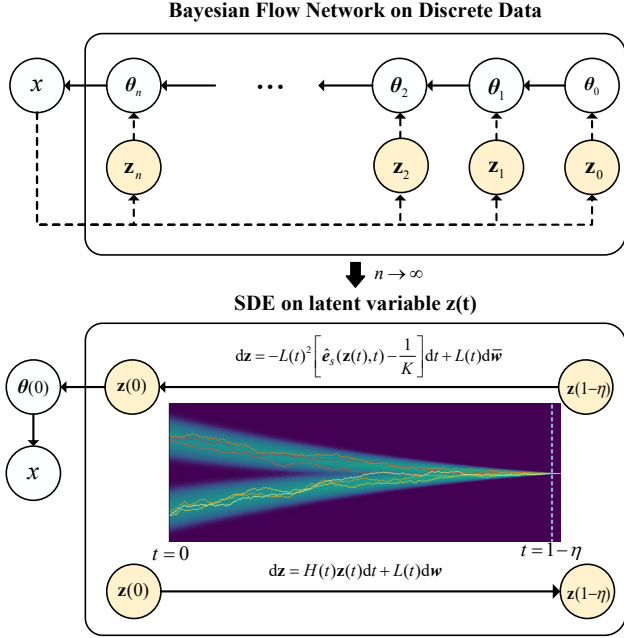
Bayesian Flow Network on Discrete Data

$$\mathrm{d}\mathbf{z} = -L(t)^2 \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(t), t) - \frac{1}{K} \right] \mathrm{d}t + L(t)\mathrm{d}\bar{\boldsymbol{w}}$$

SDE on latent variable z(t)

$$\mathrm{d}\mathbf{z} = H(t)\mathbf{z}(t)\mathrm{d}t + L(t)\mathrm{d}\boldsymbol{w}$$

Figure 2. **Illustration of BFN on discrete data and the corresponding SDEs.** The SDEs are defined w.r.t. the latent variables **z**, which are marginalized in BFN, on time $[0, 1 - \eta]$.

velops fast samplers for BFNs. However, this adaptation to discrete data is far from straightforward, as it involves SDEs operating on latent variables **z** — a significant departure from the original BFN formulation that marginalizes out these variables, rather than updating the distribution parameters $\boldsymbol{\theta}$. Consequently, it is surprising that the training and sampling of BFN on discrete data still connect to the SDE formulation on **z**.

5.1. Formulating BFN on Discrete Data as SDEs

Similar to Theorem 4.1, the truncated noise-adding process of the continuous-time BFN on discrete data in Eq. (11) uniquely solves a linear SDE, summarized as follows.

Theorem 5.1 (Proof in Appendix B.1). *Let $\eta > 0$ be an arbitrarily small constant. The BFN in Eq. (11) with $t \in [0, 1 - \eta]$ is the unique solution of the following linear SDE:*

$$\mathrm{d}\mathbf{z} = H(t)\mathbf{z}\,\mathrm{d}t + L(t)\,\mathrm{d}\boldsymbol{w}. \qquad (24)$$

Here $\boldsymbol{w}$ is a standard Wiener process and

$$H(t) = \frac{\beta'(t)}{\beta(t)} = -\frac{2}{1 - t}, \qquad (25)$$

$$L(t)^2 = -K\beta'(t) = 2K\beta_1(1 - t), \qquad (26)$$

where K and β_1 are hyperparameters defined in Eq. (11).

The rationale for truncation of t and the way to deal with η and $\mathbf{z}(1 - \eta)$ is similar to the continuous data case, detailed in the proof and Sec. 6.1, respectively.

Notably, Theorem 5.1 characterizes the dynamics of **z** instead of $\boldsymbol{\theta}$, as illustrated in Fig. 2. Indeed, the dynamics of $\boldsymbol{\theta}$ do not correspond to a linear SDE as $\boldsymbol{\theta}$ is a nonlinear transformation of **z** as shown in Eq. (11). It is implied that the original sampling process in Eq. (14) *does not directly discretize the linear SDE*, as detailed in Sec. 5.3.

The associated reverse-time SDE (Song et al., 2021) for the linear SDE in Eq. (24) in $[0, 1 - \eta]$ is given by

$$\mathrm{d}\mathbf{z} = [H(t)\mathbf{z} - L(t)^2 \nabla_{\mathbf{z}} \log p_t(\mathbf{z})]\,\mathrm{d}t + L(t)\,\mathrm{d}\bar{\boldsymbol{w}}, \quad (27)$$

where $\nabla_{\mathbf{z}} \log p_t(\mathbf{z})$ is the unknown score function, defined on **z** instead of $\boldsymbol{\theta}$.

5.2. Training as Parameterizing the Reverse-time SDE

It is nontrivial to see yet can be proven that the training objective of the continuous-time BFN on discrete data in Eq. (12) is a reparameterized form of DSM (Vincent, 2011) w.r.t. **z**, as summarized in the following Theorem 5.2.

Theorem 5.2 (Proof in Appendix B.2). *Minimizing the continuous-time loss of BFN on discrete data in Eq. (12) is equivalent to minimizing the DSM loss in Eq. (4). Besides, the corresponding estimate of the score function is given by*

$$\hat{s}(\mathbf{z}(t), t) = -\frac{\mathbf{z}(t)}{K\beta(t)} + \hat{\boldsymbol{e}}_s(\mathbf{z}(t), t) - \frac{1}{K}, \qquad (28)$$

where $\hat{\boldsymbol{e}}_s(\mathbf{z}(t), t)$ is the network trained by BFN.

Theorem 5.1 and Theorem 5.2 distinct BFNs from existing discrete DMs. Specifically, BFNs applied to discrete data solve a linear SDE and are trained using DSM, which aligns seamlessly with continuous DMs. Consequently, without changing the discrete data, BFNs are significantly simpler and more scalable and efficient than the related work, leveraging advancements in continuous DMs. We provide a comprehensive review and discussion in Sec. 2.

5.3. Sampling as Discretizing the Reverse-time SDE

Plugging Eq. (28) into Eq. (27), we get a parameterized reverse-time SDE in $[0, 1 - \eta]$ for sampling as follows

$$\mathrm{d}\mathbf{z} = -L(t)^2 \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(t), t) - \frac{1}{K} \right] \mathrm{d}t + L(t)\,\mathrm{d}\bar{\boldsymbol{w}}. \quad (29)$$

The following Proposition 4.2 suggests that the sampler proposed in the original BFN paper approximately discretizes the parameterized reverse-time SDE.

Proposition 5.3 (Proof in Appendix B.3). *If the categorical sampling step in the BFN sampler on discrete data (i.e., Eq. (13)) is omitted, then it is a first-order discretization of the parameterized reverse-time SDE in Eq. (29).*

The role of the categorical sampling step is still unclear in theory. However, experiments in Fig. 6 (Sec. 6.3) reveal that removing the categorical sampling step leads to consistently better performance in fewer than 50 NFEs, and almost the same results otherwise. We provide preliminary analyses of it in Appendix C.4.

5.4. Probability Flow ODE and Faster Sampling

Similar to the continuous case, the equivalent probability flow ODE of the parameterized reverse-time SDE on discrete data in Eq. (29) is

$$d\mathbf{z} = \left\{ -\frac{1}{1-t}\mathbf{z}(t) - \beta_1(1-t)[K\hat{\boldsymbol{e}}_s(\mathbf{z}(t), t) - 1] \right\} dt. \tag{30}$$

For $0 \le t < s < 1 - \eta$, its solution can be written as

$$\mathbf{z}(t) = \frac{1-t}{1-s}\mathbf{z}(s) + \beta_1(1-t)(t-s) \\ - K\beta_1(1-t)\int_s^t \hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) \, d\tau. \tag{31}$$

Again, we propose BFN-Solver on discrete data, and the first-order version is given by

$$\mathbf{z}_i = \beta_1(1-t_i)(t_i - t_{i-1})(1 - K\hat{\boldsymbol{e}}_s(\mathbf{z}(t_{i-1}), t_{i-1})) \\ + \frac{1-t_i}{1-t_{i-1}}\mathbf{z}_{i-1}. \tag{32}$$

Notably, we map the latent $\mathbf{z}_M$ to the distribution parameter $\boldsymbol{\theta}_M = \text{softmax}(\mathbf{z}_M)$ at the last step to obtain the final samples. We refer the readers to Appendix B.5 for higher-order solvers of both ODE and SDE. As presented in Sec. 6.3, the conclusion on the improvement of BFN-Solvers over the original BFN sampler remains the same on discrete data.

6. Experiments

We present the experimental setups in Sec. 6.1. We validate the proposed BFN-Solvers on continuous and discrete data, in Sec. 6.2 and Sec. 6.3 respectively.

6.1. Experimental Settings

Model. We employed the pre-trained models provided by the BFN (Graves et al., 2023) in all experiments for fairness.

Datasets. For continuous data, the model is trained on the CIFAR-10 (Krizhevsky & Hinton, 2009) dataset which contain 50K training images. For discrete data, the model is trained on the text8 (Mahoney, 2011) dataset which contains 90M consecutive characters, each character is a lower Latin

letter 'a'-'z' or the '_' whitespace token, giving a class number of 27. Each sample is a sequence of 256 characters.

Metrics. For continuous data, we adopt the widely used FID (Heusel et al., 2017) as the sample quality metric. We compute the FID metric on 10K generated samples for efficiency. For discrete data, there is no widely adopted sample-based metric comparable to FID in image modeling. Given our reliance on a simple character-level text dataset, we found that spelling accuracy (SA) is a straightforward yet effective metric for measuring the quality of text generation. Specifically, SA is defined as the ratio of correctly spelled words to the total words in the entire generated sequence, which is segmented by spaces. In each experiment, we collect 1,000 generated samples to calculate the metric. Additionally. we conducted a user study for text generation quality evaluation. For the user study, there are 100 questions for each one vs. one comparison (e.g., BFN vs. BFN-Solver1). In each question, participants were presented with two sentences randomly generated from two methods. Participants were instructed to choose a sentence of higher quality, which is known as the *two-alternative forced choice* methodology (Kawar et al., 2023; Bar-Tal et al., 2022; Park et al., 2020). Please see Appendix C for more experimental details.

Truncation. η is a manually tuned hyperparameter specified in each experiment. For both $p_{1-\eta}(\boldsymbol{\mu})$ and $p_{1-\eta}(\mathbf{z})$, we found an isotropic Gaussian with zero mean and a calculated variance works well. We provide preliminary analyses of the variance in Appendix C.1.

6.2. Fast Sampling on Continuous Data

We compare our proposed fast sampling methods with the original BFN continuous sampler in this section.

As illustrated in Fig. 3, with the NFE less than 100, BFN-Solver++1, BFN-Solver++2, and SDE-BFN-Solver++2 significantly outperform the BFN baseline. Moreover, BFN-Solver++2 achieves better results compared to BFN-Solver++1. When the NFE is higher (e.g., more than 500), our observations reveal that SDE-based samplers exhibit slightly better performance over ODE-based samplers, which aligns with the diffusion model (Song et al., 2020; Karras et al., 2022; Nie et al., 2023). Please see Appendix D.1 and Appendix D.5 for more quantitative results and randomly generated images, respectively.

We slightly tune the hyperparameter η for our methods on different NFEs to get the best results, as detailed in Appendix D.3.

6.3. Fast Sampling on Discrete Data

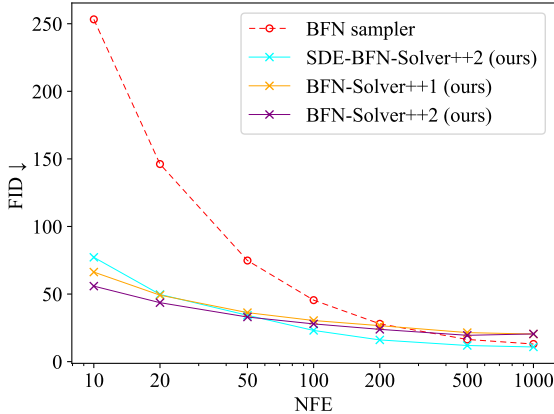We compare our proposed fast sampling methods with the origin BFN discrete sampler and other representative base-

Figure 3. **Fast sampling results on the continuous CIFAR-10 dataset.** Sampling quality is measured by FID ↓, varying the NFE.

lines in this section.

As illustrated in Fig. 4, with the NFE less than 30, BFN-Solver1, BFN-Solver2, and SDE-BFN-Solver2 significantly outperform the BFN baseline. Moreover, BFN-Solver2 and SDE-BFN-Solver2 achieve better results compared to BFN-Solver1, agreeing with the continuous case. We provide a preliminary user study in Fig. 5 with 10 NFEs and the results align with Fig. 4. When the NFE is higher (e.g., more than 500), we observe that SDE-based samplers exhibit slightly better performance than ODE-based samplers, which aligns with existing results in DMs. Please see Appendix D.2 and Appendix D.5 for more quantitative results and randomly generated texts.

We compare the efficiency of BFN-Solver2 with representative baselines including D3PM (Austin et al., 2023) and OAARDM (Hoogeboom et al., 2022). As summarized in Table 2, BFN-Solver2 outperforms all baselines with both 20 and 50 steps indirectly. Notably, evaluating the exact likelihood by the proposed samplers requires a nontrivial extension (Song et al., 2021). We use the original BFN sampler, evaluated under both metrics, as a bridge for an indirect comparison. For instance, according to Table 2, BFN-Solver2 (20 steps, SA 0.84) is better than the original BFN sampler (25 steps, SA 0.80, BPC 1.52) and better than OA-ARDM (20 steps, BPC 1.52).

We find that the hyperparameter $\eta = 0.001$ is sufficient for all NFEs for BFN-Solvers to get excellent results. We refer the readers to Appendix D.4 for more details.

Finally, we perform an ablation of the original BFN solver in Fig. 6 and find that an exact solver that just removes the categorical sampling step from the BFN sampler works better, conforming to our theory.

Table 2. **Efficiency evaluation results of different methods on discrete text8 dataset.** model performance is measured by bits-per-character (BPC) ↓ and SA↑, varying the number of evaluations. To get the strongest baseline, we use the D3PM absorbing model, which has a better BPC than D3PM uniform and NN models in the original paper (Austin et al., 2023).

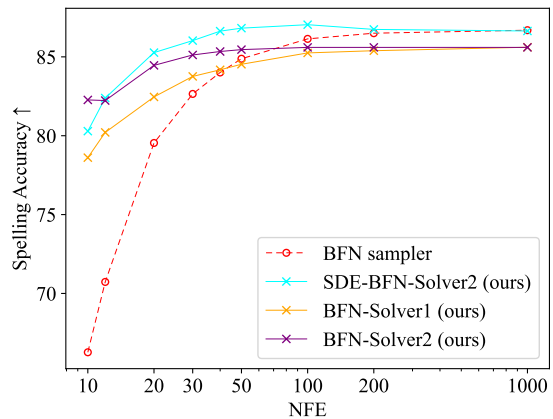| METHODS	NFE	BPC ↓	SA ↑
BFN-SOLVER2 (OURS)	50	-	**0.85**
BFN	50	1.47	0.84
D3PM (BEST)	50	1.52	-
BFN	25	1.52	0.80
BFN-SOLVER2 (OURS)	20	-	**0.84**
BFN	20	1.56	0.79
BFN-SOLVER2 (OURS)	20	1.56	-
OA-ARDM	20	1.52	-

Figure 4. **Fast sampling results on the discrete text8 dataset.** Sampling quality is measured by SA ↑, varying the NFE.
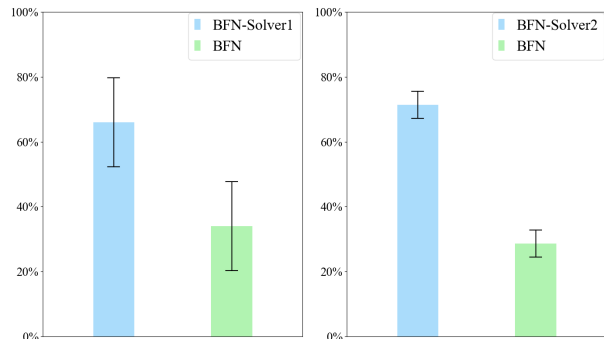

Figure 5. **User study results on the discrete text8 dataset with 10 NFE.** We present the preference rates (with 95% confidence intervals) of BFN-Solver1 and BFN-Solver2 over BFN baseline.
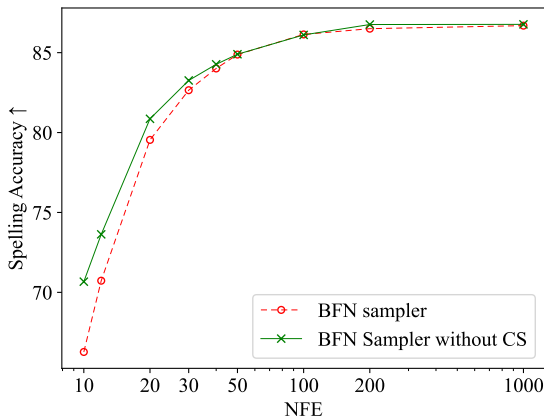
Figure 6. **Ablation of the categorical sampling (CS) step in the BFN sampler on the text8 dataset.** Sampling quality is measured by SA ↑, varying the NFE.

7. Conclusion

We unify BFNs and DMs by identifying the linear SDEs pertinent to the noise-addition processes in BFNs, illustrating that BFN's regression losses correspond with denoise score matching, and validating the sampler in BFN as an effective first-order solver for the related reverse-time SDE. Motivated by these insights, we implement fast sampling techniques from DMs in BFNs, yielding promising results.

Building upon the established results of DMs, this paper establishes a principled and systematic approach to the analysis and enhancement of BFNs and future work includes the development of predictor-corrector samplers (Song et al., 2020; Zhao et al., 2023), improved methods for likelihood evaluation (Bao et al., 2022b;a), and novel training strategies to refine (Karras et al., 2022) and scale BFNs (Rombach et al., 2022).

Limitations of the paper include the scale of the datasets and evaluation metrics. In our experiment, for a fair comparison, we leverage the pre-trained models of BFNs, which are all trained on small datasets. Further, the samplers cannot be directly used in likelihood evaluation and we mainly employ the FID and spelling accuracy as surrogates for the sample quality, potentially introducing bias. Hopefully, these limitations can be solved by scaling up BFNs to common benchmarks, as mentioned in future work.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Anderson, B. D. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

Atkinson, K., Han, W., and Stewart, D. E. *Numerical solution of ordinary differential equations*, volume 81. John Wiley & Sons, 2009.

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces, 2023.

Balaji, Y., Nah, S., Huang, X., Vahdat, A., Song, J., Zhang, Q., Kreis, K., Aittala, M., Aila, T., Laine, S., Catanzaro, B., Karras, T., and Liu, M.-Y. ediff-i: Text-to-image diffusion models with an ensemble of expert denoisers, 2023.

Bao, F., Li, C., Sun, J., Zhu, J., and Zhang, B. Estimating the optimal covariance with imperfect mean in diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 1555–1584. PMLR, 2022a.

Bao, F., Li, C., Zhu, J., and Zhang, B. Analytic-DPM: an analytic estimate of the optimal reverse variance in diffusion probabilistic models. In *International Conference on Learning Representations*, 2022b.

Bao, F., Nie, S., Xue, K., Li, C., Pu, S., Wang, Y., Yue, G., Cao, Y., Su, H., and Zhu, J. One transformer fits all distributions in multi-modal diffusion at scale. In *International Conference on Machine Learning*, pp. 1692–1717. PMLR, 2023.

Bar-Tal, O., Ofri-Amar, D., Fridman, R., Kasten, Y., and Dekel, T. Text2live: Text-driven layered image and video editing. In *European conference on computer vision*, pp. 707–723. Springer, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners.

Advances in neural information processing systems, 33: 1877–1901, 2020.

Campbell, A., Benton, J., Bortoli, V. D., Rainforth, T., Deligiannidis, G., and Doucet, A. A continuous time framework for discrete denoising models, 2022.

Chen, N., Zhang, Y., Zen, H., Weiss, R. J., Norouzi, M., and Chan, W. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Chen, T., Zhang, R., and Hinton, G. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *The Eleventh International Conference on Learning Representations*, 2022.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Dieleman, S., Sartran, L., Roshannai, A., Savinov, N., Ganin, Y., Richemond, P. H., Doucet, A., Strudel, R., Dyer, C., Durkan, C., Hawthorne, C., Leblond, R., Grathwohl, W., and Adler, J. Continuous diffusion for categorical data, 2022.

Graves, A., Srivastava, R. K., Atkinson, T., and Gomez, F. Bayesian flow networks, 2023.

Guo, H., Lu, C., Bao, F., Pang, T., Yan, S., Du, C., and Li, C. Gaussian mixture solvers for diffusion models. *arXiv preprint arXiv:2311.00941*, 2023.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ho, J., Chan, W., Saharia, C., Whang, J., Gao, R., Gritsenko, A., Kingma, D. P., Poole, B., Norouzi, M., Fleet, D. J., et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

Hoogeboom, E., Nielsen, D., Jaini, P., Forré, P., and Welling, M. Argmax flows and multinomial diffusion: Learning categorical distributions, 2021.

Hoogeboom, E., Gritsenko, A. A., Bastings, J., Poole, B., van den Berg, R., and Salimans, T. Autoregressive diffusion models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=Lm8T39vLDTE.

Hyvärinen, A. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research (JMLR)*, 6(Apr):695–709, 2005.

Jolicoeur-Martineau, A., Li, K., Piché-Taillefer, R., Kachman, T., and Mitliagkas, I. Gotta go fast when generating data with score-based models. *arXiv preprint arXiv:2105.14080*, 2021.

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

Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., and Irani, M. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6007–6017, 2023.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Kloeden, P. E., Platen, E., Kloeden, P. E., and Platen, E. *Stochastic differential equations*. Springer, 1992.

Kong, Z., Ping, W., Huang, J., Zhao, K., and Catanzaro, B. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Li, X. L., Thickstun, J., Gulrajani, I., Liang, P., and Hashimoto, T. B. Diffusion-lm improves controllable text generation, 2022.

Liu, L., Ren, Y., Lin, Z., and Zhao, Z. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022.

Lou, A. and Ermon, S. Reflected diffusion models. *arXiv preprint arXiv:2304.04740*, 2023.

Lou, A., Meng, C., and Ermon, S. Discrete diffusion language modeling by estimating the ratios of the data distribution, 2023.

Lu, C., Zheng, K., Bao, F., Chen, J., Li, C., and Zhu, J. Maximum likelihood training for score-based diffusion odes by high order denoising score matching. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 14429–14460.

PMLR, 2022a. URL https://proceedings.mlr.press/v162/lu22f.html.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022b.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022c.

Mahabadi, R. K., Tae, J., Ivison, H., Henderson, J., Beltagy, I., Peters, M. E., and Cohan, A. Tess: Text-to-text self-conditioned simplex diffusion. *arXiv preprint arXiv:2305.08379*, 2023.

Mahoney, M. Large text compression benchmark, 2011.

Meng, C., Choi, K., Song, J., and Ermon, S. Concrete score matching: Generalized score matching for discrete data, 2023.

Nie, S., Guo, H. A., Lu, C., Zhou, Y., Zheng, C., and Li, C. The blessing of randomness: Sde beats ode in general diffusion-based image editing. *arXiv preprint arXiv:2311.01410*, 2023.

Øksendal, B. *Stochastic differential equations*. Springer, 2003.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Pang, T., Xu, K., Li, C., Song, Y., Ermon, S., and Zhu, J. Efficient learning of generative models via finite-difference score matching. *Advances in Neural Information Processing Systems*, 33:19175–19188, 2020.

Park, T., Zhu, J.-Y., Wang, O., Lu, J., Shechtman, E., Efros, A., and Zhang, R. Swapping autoencoder for deep image manipulation. *Advances in Neural Information Processing Systems*, 33:7198–7211, 2020.

Podell, D., English, Z., Lacey, K., Blattmann, A., Dockhorn, T., Müller, J., Penna, J., and Rombach, R. Sdxl: improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

Poole, B., Jain, A., Barron, J. T., and Mildenhall, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

Richemond, P. H., Dieleman, S., and Doucet, A. Categorical sdes with simplex diffusion. *arXiv preprint arXiv:2210.14784*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E. L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35: 36479–36494, 2022.

Singer, U., Polyak, A., Hayes, T., Yin, X., An, J., Zhang, S., Hu, Q., Yang, H., Ashual, O., Gafni, O., et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Song, Y., Garg, S., Shi, J., and Ermon, S. Sliced score matching: A scalable approach to density and score estimation. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2019.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021.

Sun, H., Yu, L., Dai, B., Schuurmans, D., and Dai, H. Score-based continuous-time discrete diffusion models, 2023.

Vincent, P. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., and Zhu, J. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023.

Xue, S., Yi, M., Luo, W., Zhang, S., Sun, J., Li, Z., and Ma, Z.-M. Sa-solver: Stochastic adams solver for fast sampling of diffusion models. *arXiv preprint arXiv:2309.05019*, 2023a.

Xue, Z., Song, G., Guo, Q., Liu, B., Zong, Z., Liu, Y., and Luo, P. Raphael: Text-to-image generation via large mixture of diffusion paths. *arXiv preprint arXiv:2305.18295*, 2023b.

Ye, J., Zheng, Z., Bao, Y., Qian, L., and Wang, M. Dinoiser: Diffused conditional sequence learning by manipulating noises, 2023.

Zhang, Q., Tao, M., and Chen, Y. gddim: Generalized denoising diffusion implicit models. *arXiv preprint arXiv:2206.05564*, 2022.

Zhao, W., Bai, L., Rao, Y., Zhou, J., and Lu, J. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *arXiv preprint arXiv:2302.04867*, 2023.

A. Derivation for Continuous-time BFN on Continuous Data

A.1. Proof of Theorem 4.1

Proof. In this proof, we will show that the marginal distribution $\boldsymbol{\mu}(t)$ *conditioned on* $\mathbf{x}$ of Eq. (15) at time t is the BFN in Eq. (8) by the "variation of constants" formula (Atkinson et al., 2009) and the Itô's formula (Øksendal, 2003, Theorem 4.2.1).

To find the marginal distribution of $\boldsymbol{\mu}(t)$, we introduce a new process $\tilde{\boldsymbol{\mu}}(t) := e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}\boldsymbol{\mu}(t)$. Using Itô's formula, we see that $\tilde{\boldsymbol{\mu}}$ follows the equation below.

$$
\begin{aligned}
\mathrm{d}\tilde{\boldsymbol{\mu}} &= e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}\,\mathrm{d}\boldsymbol{\mu}(t) + \left(\frac{\mathrm{d}}{\mathrm{d}t}e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}\right)\boldsymbol{\mu}(t)\,\mathrm{d}t \\
&= e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}(F(t)\boldsymbol{\mu}(t)\,\mathrm{d}t + G(t)\,\mathrm{d}\boldsymbol{w}) - F(t)e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}\boldsymbol{\mu}(t)\,\mathrm{d}t \\
&= e^{-\int_0^t F(\tau)\,\mathrm{d}\tau}G(t)\,\mathrm{d}\boldsymbol{w}.
\end{aligned}
$$

Writing the above equation in the integral form we find that

$$
\tilde{\boldsymbol{\mu}}(t) = \tilde{\boldsymbol{\mu}}(0) + \int_0^t e^{-\int_0^s F(\tau)\,\mathrm{d}\tau}G(s)\,\mathrm{d}\boldsymbol{w}(s). \tag{33}
$$

It is known that from Itô's isometry (Øksendal, 2003, Corollary 3.1.7) the above Itô's integral is a Gaussian distribution with variance $\int_0^t e^{-\int_0^s 2F(\tau)\,\mathrm{d}\tau}G(s)^2\,\mathrm{d}s$.

Note that $F(t) = \frac{\mathrm{d}}{\mathrm{d}t}\ln\gamma(t)$ and $G(t)^2 = -\frac{\mathrm{d}}{\mathrm{d}t}\gamma(t)$, then Eq. (33) becomes

$$
\frac{\gamma(0)}{\gamma(t)}\boldsymbol{\mu}(t) \sim \boldsymbol{\mu}(0) + \mathcal{N}\left(0, \gamma(0)^2\left(\frac{1}{\gamma(t)} - \frac{1}{\gamma(0)}\right)\boldsymbol{I}\right),
$$

which shows that the distribution of $\boldsymbol{\mu}(t)$ conditioned on $\boldsymbol{\mu}(0)$ is

$$
\boldsymbol{\mu}(t) \mid \boldsymbol{\mu}(0) \sim \mathcal{N}\left(\frac{\gamma(t)}{\gamma(0)}\boldsymbol{\mu}(0), \left(\gamma(t) - \frac{\gamma(t)^2}{\gamma(0)}\right)\boldsymbol{I}\right). \tag{34}
$$

Recall that from Eq. (8) with $t = 0$, the initial distribution of $\boldsymbol{\mu}(0)$ given the data $\mathbf{x}$ is

$$
\boldsymbol{\mu}(0) \mid \mathbf{x} \sim \mathcal{N}(\gamma(0)\mathbf{x}, \gamma(0)(1 - \gamma(0))\boldsymbol{I}). \tag{35}
$$

In view of the above two equations, we see that

$$
\boldsymbol{\mu}(t) \mid \mathbf{x} \sim \mathcal{N}(\gamma(t)\mathbf{x}, \gamma(t)(1 - \gamma(t))\boldsymbol{I}), \tag{36}
$$

which is the BFN in Eq. (8) for any $t \in [0, 1 - \eta]$, and thus completes the proof. $\qquad\square$

Finally, we note that the coefficient $F(t)$ tends to infinity as $t \to 1$, which may violate the regularity assumptions used in Itô's formula, and in the existence and the uniqueness of the solution of Eq. (15). Nevertheless, as noted in the paragraph after Theorem 4.1, when we restrict on the time interval $t \in [0, 1 - \eta]$ for any fixed $\eta > 0$, the coefficients of Eq. (8) are well-behaved.

A.2. Proof of Proposition 4.2

In this section, we provide the proof of Proposition 4.2. As we will see, the BFN sampler is equivalent to solving a reparametrized equation using a variant of the first-order DPM-Solver++ (Lu et al., 2022c).

Proof. We consider the following reparameterization of $\hat{\boldsymbol{\epsilon}}$:

$$
\hat{\mathbf{x}}(\boldsymbol{\mu}, t) := \frac{\boldsymbol{\mu}}{\gamma(t)} - \sqrt{\frac{1 - \gamma(t)}{\gamma(t)}}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, t) \quad\Leftrightarrow\quad \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}, t) = \sqrt{\frac{\gamma(t)}{1 - \gamma(t)}}\left[\frac{\boldsymbol{\mu}}{\gamma(t)} - \hat{\mathbf{x}}(\boldsymbol{\mu}, t)\right], \tag{37}
$$

under which the reverse SDE in Eq. (20) is

$$
\begin{aligned}
\mathrm{d}\boldsymbol{\mu} &= \left(F(t)\boldsymbol{\mu} + \frac{G(t)^2}{\sqrt{\gamma(t)(1-\gamma(t))}} \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t) \right) \mathrm{d}t + G(t)\,\mathrm{d}\bar{\boldsymbol{w}} \\
&= \left(\frac{\gamma'(t)}{\gamma(t)}\boldsymbol{\mu} - \frac{\gamma'(t)}{\sqrt{\gamma(t)(1-\gamma(t))}} \hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t), t) \right) \mathrm{d}t + \sqrt{-\gamma'(t)}\,\mathrm{d}\bar{\boldsymbol{w}} \\
&= \left(\frac{\gamma'(t)}{\gamma(t)}\boldsymbol{\mu} - \frac{\gamma'(t)}{1-\gamma(t)} \left(\frac{\boldsymbol{\mu}}{\gamma(t)} - \hat{\mathbf{x}}(\boldsymbol{\mu}(t), t) \right) \right) \mathrm{d}t + \sqrt{-\gamma'(t)}\,\mathrm{d}\bar{\boldsymbol{w}} \\
&= \left(-\frac{\gamma'(t)}{1-\gamma(t)}\boldsymbol{\mu} + \frac{\gamma'(t)}{1-\gamma(t)}\hat{\mathbf{x}}(\boldsymbol{\mu}(t), t) \right) \mathrm{d}t + \sqrt{-\gamma'(t)}\,\mathrm{d}\bar{\boldsymbol{w}}.
\end{aligned}
\tag{38}
$$

In the integral form similar to Eq. (33), for $0 \le t < s < 1$, the above equation is

$$
\boldsymbol{\mu}(t) = \frac{1-\gamma(t)}{1-\gamma(s)}\boldsymbol{\mu}(s) + (1-\gamma(t))\int_s^t \frac{\gamma'(\tau)}{(1-\gamma(\tau))^2}\hat{\mathbf{x}}(\boldsymbol{\mu}(\tau), \tau)\,\mathrm{d}\tau + \int_s^t \frac{1-\gamma(t)}{1-\gamma(\tau)}\sqrt{-\gamma'(\tau)}\,\mathrm{d}\bar{\boldsymbol{w}}(\tau).
\tag{39}
$$

Approximating $\hat{\mathbf{x}}(\boldsymbol{\mu}(\tau), \tau)$ using $\hat{\mathbf{x}}(\boldsymbol{\mu}(s), s)$, and compute the Itô's integral exactly, we can obtain the following first-order discretization

$$
\begin{aligned}
\boldsymbol{\mu}(t) &\approx \frac{1-\gamma(t)}{1-\gamma(s)}\boldsymbol{\mu}(s) + \left(\frac{\gamma(t)-\gamma(s)}{1-\gamma(s)} \right) \hat{\mathbf{x}}(\boldsymbol{\mu}(s), s) + \sqrt{\frac{1-\gamma(t)}{1-\gamma(s)}(\gamma(t)-\gamma(s))}\,\mathbf{u}_s \\
&= \frac{\gamma(t)}{\gamma(s)}\boldsymbol{\mu}(s) - \frac{\gamma(t)-\gamma(s)}{\sqrt{\gamma(s)(1-\gamma(s))}}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(s), s) + \sqrt{\frac{1-\gamma(t)}{1-\gamma(s)}(\gamma(t)-\gamma(s))}\,\mathbf{u}_s,
\end{aligned}
$$

where $\mathbf{u}_s = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, which matches the BFN's sampling rule in Eq. (10), and thus completes the proof. $\square$

A.3. Derivation of BFN-Solvers on Continuous Data

We follow the principles in DPM-Solvers (Lu et al., 2022b;c) to derive samplers of the ODE in Eq. (21). Let times $0 \le t < s \le 1 - \eta$ and $\lambda(t) = \frac{1}{2}\log\frac{\gamma(t)}{1-\gamma(t)}$. The exact solution of this ODE can be formulated by "variation of constants" formula (Atkinson et al., 2009):

$$
\boldsymbol{\mu}(t) = e^{\int_s^t F(\tau)\,\mathrm{d}\tau}\boldsymbol{\mu}(s) + \int_s^t \left(e^{\int_\tau^t F(r)\,\mathrm{d}r}\frac{G(\tau)^2}{2\sqrt{\gamma(\tau)(1-\gamma(\tau))}}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(\tau), \tau) \right) \mathrm{d}\tau.
\tag{40}
$$

Note that $F(t) = \frac{\mathrm{d}}{\mathrm{d}t}\ln\gamma(t)$ and $\frac{\mathrm{d}\lambda(t)}{\mathrm{d}t} = -\frac{G(t)^2}{\gamma(t)(1-\gamma(t))}$, then the above equation becomes

$$
\boldsymbol{\mu}(t) = \frac{\gamma(t)}{\gamma(s)}\boldsymbol{\mu}(s) - \gamma(t)\int_s^t \frac{\mathrm{d}\lambda(\tau)}{\mathrm{d}\tau}e^{-\lambda(\tau)}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(\tau), \tau)\,\mathrm{d}\tau
\tag{41}
$$

$$
= \frac{\gamma(t)}{\gamma(s)}\boldsymbol{\mu}(s) - \gamma(t)\int_{\lambda(s)}^{\lambda(t)} e^{-\lambda}\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda))\,\mathrm{d}\lambda,
\tag{42}
$$

where the last equation uses the fact that $\lambda(\tau)$ is strict monotone and $t_\lambda(\cdot)$ is the inverse function of $\lambda(t)$.

Let $\hat{\boldsymbol{e}}^{(m)}(\lambda) := \hat{\boldsymbol{e}}^{(m)}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda)) := \frac{\mathrm{d}^m\hat{\boldsymbol{e}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda))}{\mathrm{d}\lambda^m}$ be the m-th order derivative of the map $\lambda \mapsto \hat{\boldsymbol{e}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda))$, then we can approximate $\hat{\boldsymbol{e}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda))$ by the Taylor's expansion at $\lambda = \lambda(s)$ for any $k \ge 0$:

$$
\hat{\boldsymbol{\epsilon}}(\boldsymbol{\mu}(t_\lambda(\lambda)), t_\lambda(\lambda)) = \sum_{m=0}^{k-1} \frac{(\lambda - \lambda(s))^m}{m!}\hat{\boldsymbol{e}}^{(m)}(\lambda(s)) + O((\lambda - \lambda(s))^k).
\tag{43}
$$

Subsituting the above equation into Eq. (42) yields

$$
\boldsymbol{\mu}(t) = \frac{\gamma(t)}{\gamma(s)}\boldsymbol{\mu}(s) - \sum_{m=0}^{k-1} \gamma(t)\hat{\boldsymbol{e}}^{(m)}(\lambda(s))\int_{\lambda(s)}^{\lambda(t)} e^{-\lambda}\frac{(\lambda - \lambda(s))^m}{m!}\,\mathrm{d}\lambda + O((\lambda - \lambda(s))^{k+1}).
\tag{44}
$$

Algorithm 1 BFN-Solver++1 (on continuous data)

Require: time steps $\{t_i\}_{i=0}^{M}$ from $t_0 = 1 - \eta$ to $t_M = 0$, noise prediction model $\hat{\epsilon}(\boldsymbol{\mu}, t)$
Denote $\bar{\alpha}_t = \gamma(t)$, $\bar{\sigma}_t = \sqrt{\gamma(t)(1 - \gamma(t))}$, $\lambda_t = \log \frac{\bar{\alpha}_t}{\bar{\sigma}_t}$
$\boldsymbol{\mu}_0 \sim \mathcal{N}(\mathbf{0}, \gamma(t_0)(1 - \gamma(t_0))\boldsymbol{I}), K\beta(t_0)\boldsymbol{I})$
for $i = 1$ **to** M **do**
$\quad h_i = (\lambda_{t_i} - \lambda_{t_{i-1}})$
$\quad \hat{\mathbf{x}}_i = \frac{\boldsymbol{\mu}_{i-1}}{\gamma(t_{i-1})} - \sqrt{\frac{1 - \gamma(t_{i-1})}{\gamma(t_{i-1})}} \hat{\epsilon}(\boldsymbol{\mu}_{i-1}, t_{i-1})$
$\quad \boldsymbol{\mu}_i = \frac{\bar{\sigma}_{t_i}}{\bar{\sigma}_{t_{i-1}}} \boldsymbol{\mu}_{i-1} - \bar{\alpha}_{t_i}(e^{-h_i} - 1)\hat{\mathbf{x}}_i$
end for
return $\hat{\mathbf{x}}_M$

Algorithm 2 BFN-Solver++2 (on continuous data)

Require: time steps $\{t_i\}_{i=0}^{M}$ from $t_0 = 1 - \eta$ to $t_M = 0$, noise prediction model $\hat{\epsilon}(\boldsymbol{\mu}, t)$
Denote $\bar{\alpha}_t = \gamma(t)$, $\bar{\sigma}_t = \sqrt{\gamma(t)(1 - \gamma(t))}$, $\lambda_t = \log \frac{\bar{\alpha}_t}{\bar{\sigma}_t}$
$\boldsymbol{\mu}_0 \sim \mathcal{N}(\mathbf{0}, \gamma(t_0)(1 - \gamma(t_0))\boldsymbol{I}), K\beta(t_0)\boldsymbol{I})$, Initialize an empty buffer Q.
$\hat{\mathbf{x}}_0 = \frac{\boldsymbol{\mu}_0}{\gamma_{t_0}} - \sqrt{\frac{1 - \gamma_{t_0}}{\gamma_{t_0}}} \hat{\epsilon}(\boldsymbol{\mu}_0, t_0)$
$Q \leftarrow \hat{\mathbf{x}}_0$
$h_1 = (\lambda_{t_1} - \lambda_{t_0})$
$\boldsymbol{\mu}_1 = \frac{\bar{\sigma}_{t_1}}{\bar{\sigma}_{t_0}} \boldsymbol{\mu}_0 - \bar{\alpha}_{t_1}(e^{-h_1} - 1)\hat{\mathbf{x}}_0$
$\hat{\mathbf{x}}_1 = \frac{\boldsymbol{\mu}_1}{\gamma_{t_1}} - \sqrt{\frac{1 - \gamma_{t_1}}{\gamma_{t_1}}} \hat{\epsilon}(\boldsymbol{\mu}_1, t_1)$
$Q \leftarrow \hat{\mathbf{x}}_1$
for $i = 2$ **to** M **do**
$\quad h_i = (\lambda_{t_i} - \lambda_{t_{i-1}})$
$\quad r_i = \frac{h_{i-1}}{h_i}$
$\quad \boldsymbol{D}_i = \left(1 + \frac{1}{2r_i}\right) \hat{\mathbf{x}}_{i-1} - \frac{1}{2r_i} \hat{\mathbf{x}}_{i-2}$
$\quad \boldsymbol{\mu}_i = \frac{\bar{\sigma}_{t_i}}{\bar{\sigma}_{t_{i-1}}} - \bar{\alpha}_{t_i}(e^{-h_i} - 1)\boldsymbol{D}_i$
$\quad \hat{\mathbf{x}}_i = \frac{\boldsymbol{\mu}_i}{\gamma_{t_i}} - \sqrt{\frac{1 - \gamma(t_i)}{\gamma(t_i)}} \hat{\epsilon}(\boldsymbol{\mu}_i, t_i)$
$\quad$ If $i < M$, then $Q \leftarrow \hat{\mathbf{x}}_i$
end for
return $\hat{\mathbf{x}}_M$

Given an initial value $\boldsymbol{\mu}_0$ and time steps $\{t_i\}_{i=0}^{n}$ from $t_0 = 1 - \eta$ to $t_n = 0$. The solver uses n steps to iteratively compute a sequence $\{\boldsymbol{\mu}_i\}_{i=1}^{n}$ to approximate the true solutions at time $\{t_i\}_{i=1}^{n}$ using Eq. (44) by setting $t = t_i$ and $s = t_{i-1}$ for each $i = 1, \cdots, n$. We can use the well-established finite difference method or the Runge-Kutta method to avoid the computation of the high-order derivatives $\hat{e}^{(m)}(\lambda)$, as we will illustrate in the case of discrete data in Appendix. B (see also, e.g., Kloeden et al. (1992); Lu et al. (2022b)).

BFN as a DM with a special noise schedule   The derivation above with a noise prediction network are presented mainly to illustrate the idea. In our implementation, we use a data prediction network $\hat{\mathbf{x}}$ as defined in Appendix. A.2. For simplicity, we do not derive the sampler step by step. Instead, a more straightforward way to get BFN-Solver on continuous data is to treat BFN as a DM with a special noise schedule $\alpha(t) = \gamma(t)$ and $\sigma^2(t) = \gamma(t)(1 - \gamma(t))$. We can directly plugin the relation to the samplers proposed in Lu et al. (2022b;c) to obtain our first-order ODE solver BFN-Solver++1, second-order ODE solver BFN-Solver++2 and second-order SDE solver SDE-BFN-Solver++2, as presented in Algorithm 1, 2 and 3 respectively.

The plugging idea does not apply to discrete data and we provide a detailed derivation in Appendix. B.

Algorithm 3 SDE-BFN-Solver++2 (on continuous data)

Require: time steps $\{t_i\}_{i=0}^M$ from $t_0 = 1 - \eta$ to $t_M = 0$, noise prediction model $\hat{\epsilon}(\boldsymbol{\mu}, t)$
Denote $\bar{\alpha}_t = \gamma(t), \bar{\sigma}_t = \sqrt{\gamma(t)(1 - \gamma(t))}, \lambda_t = \log \frac{\bar{\alpha}_t}{\bar{\sigma}_t}$
$\boldsymbol{\mu}_0 \sim \mathcal{N}(\mathbf{0}, \gamma(t_0)(1 - \gamma(t_0))\boldsymbol{I}), K\beta(t_0)\boldsymbol{I})$, Initialize an empty buffer Q.
$\hat{\mathbf{x}}_0 = \frac{\boldsymbol{\mu}_0}{\gamma_{t_0}} - \sqrt{\frac{1-\gamma_{t_0}}{\gamma_{t_0}}} \hat{\epsilon}(\boldsymbol{\mu}_0, t_0)$
$Q \leftarrow \hat{\mathbf{x}}_0$
$h_1 = (\lambda_{t_1} - \lambda_{t_0})$
$\boldsymbol{\mu}_1 = \frac{\bar{\sigma}_{t_1}}{\bar{\sigma}_{t_0}} \boldsymbol{\mu}_0 - \bar{\alpha}_{t_1}(e^{-h_1} - 1)\hat{\mathbf{x}}_0$
$\hat{\mathbf{x}}_1 = \frac{\boldsymbol{\mu}_1}{\gamma_{t_1}} - \sqrt{\frac{1-\gamma_{t_1}}{\gamma_{t_1}}} \hat{\epsilon}(\boldsymbol{\mu}_1, t_1)$
$Q \leftarrow \hat{\mathbf{x}}_1$
for $i = 2$ **to** M **do**
   $h_i = (\lambda_{t_i} - \lambda_{t_{i-1}})$
   $r_1 = \frac{h_{i-1}}{h_i}$
   $\boldsymbol{D}_i = \frac{1}{r_1}(\hat{\mathbf{x}}_{i-1} - \hat{\mathbf{x}}_{i-2})$
   $\mathbf{u}_{t_{i-1}} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
   $\boldsymbol{\mu}_i = \frac{\bar{\sigma}_{t_i}}{\bar{\sigma}_{t_{i-1}}} e^{-h_i} \boldsymbol{\mu}_{i-1} + \bar{\alpha}_{t_i}(1 - e^{-2h_i})\hat{\mathbf{x}}_{i-1} + \frac{\bar{\alpha}_t(1-e^{-2h_i})}{2} \boldsymbol{D}_i + \bar{\sigma}_t \sqrt{1 - e^{-2h_i}} \mathbf{u}_{t_{i-1}}$
   $\hat{\mathbf{x}}_i = \frac{\boldsymbol{\mu}_i}{\gamma_{t_i}} - \sqrt{\frac{1-\gamma(t_i)}{\gamma(t_i)}} \hat{\epsilon}(\boldsymbol{\mu}_i, t_i)$
   If $i < M$, then $Q \leftarrow \hat{\mathbf{x}}_i$
end for
return $\hat{\mathbf{x}}_M$

B. Derivation for Continuous-time BFN on Discrete Data

B.1. Proof of Theorem 5.1

Proof. Similar to the proof in Appendix. A.1, let $\tilde{\mathbf{z}}(t) := e^{-\int_0^t H(\tau) \mathrm{d}\tau} \mathbf{z}(t)$, then we have

$$\tilde{\mathbf{z}}(t) = \tilde{\mathbf{z}}(0) + \int_0^t e^{-\int_0^s H(\tau) \mathrm{d}\tau} L(s) \mathrm{d}\boldsymbol{w}(s). \tag{45}$$

Since $H(t) = \frac{\mathrm{d}}{\mathrm{d}t} \ln \beta(t)$ and $L(t)^2 = -K \frac{\mathrm{d}}{\mathrm{d}t} \beta(t)$, then the above equation becomes

$$\frac{\beta(0)}{\beta(t)} \mathbf{z}(t) \sim \mathbf{z}(0) + \mathcal{N}\left(0, K\beta(0)^2 \left(\frac{1}{\beta(t)} - \frac{1}{\beta(0)}\right) \boldsymbol{I}\right),$$

which shows that the distribution of $\mathbf{z}(t)$ conditioned on $\mathbf{z}(0)$ is

$$\mathbf{z}(t) \mid \mathbf{z}(0) \sim \mathcal{N}\left(\frac{\beta(t)}{\beta(0)} \mathbf{z}(0), K\left(\beta(t) - \frac{\beta(t)^2}{\beta(0)}\right) \boldsymbol{I}\right). \tag{46}$$

Recall that from Eq. (11) with $t = 0$ we know that

$$\mathbf{z}(0) \mid \mathbf{x} \sim \mathcal{N}(\beta(0)\boldsymbol{w}_{\mathbf{x}}, K\beta(0)\boldsymbol{I}). \tag{47}$$

Combining the above two equations, we find that

$$\mathbf{z}(t) \mid \mathbf{x} \sim \mathcal{N}(\beta(t)\boldsymbol{w}_{\mathbf{x}}, K\beta(t)\boldsymbol{I}), \tag{48}$$

which is the BFN in Eq. (11) for any $t \in [0, 1 - \eta]$. □

B.2. Proof of Theorem 5.2

Proof. Recall that the loss function on discrete data in Eq. (12) is

$$\mathcal{L}^\infty(\mathbf{x}) = \mathbb{E}_{q_F(\boldsymbol{\theta}|\mathbf{x},t), t\sim U(0,1)} K\beta_1 t \|\boldsymbol{e}_{\mathbf{x}} - \hat{\boldsymbol{e}}(\boldsymbol{\theta}, t)\|^2, \tag{49}$$

where $q_F(\boldsymbol{\theta}|\mathbf{x}, t)$ is specially defined with softmax function to ensure $\boldsymbol{\theta}$ lies in the simplex as follows, making its density complex.[5]

$$q_F(\boldsymbol{\theta}|\mathbf{x}, t) = \mathop{\mathbb{E}}_{\mathbf{z}(t) \sim \mathcal{N}(\beta(t)(K\boldsymbol{e}_\mathbf{x} - \mathbf{1}), \beta(t)K\boldsymbol{I})} \delta(\boldsymbol{\theta} - \mathrm{softmax}(\mathbf{z}(t))).$$

Note that to obtain a sample $\boldsymbol{\theta}(t) \sim q_F(\boldsymbol{\theta}|\boldsymbol{x}, t)$, we can first sample $\mathbf{z}(t) \sim \mathcal{N}(\beta(t)(K\boldsymbol{e}_\mathbf{x} - 1), \beta(t)K\boldsymbol{I})$ and apply the deterministic transform $\boldsymbol{\theta}(t) := \mathrm{softmax}(\mathbf{z}(t))$. Then, we can rewrite the loss function as

$$\mathcal{L}^\infty(\mathbf{x}) = \mathop{\mathbb{E}}_{t \sim U(0,1)} \mathop{\mathbb{E}}_{\mathbf{z}(t) \sim q(\mathbf{z}(t)|\mathbf{x})} K\beta_1 t \|\boldsymbol{e}_\mathbf{x} - \hat{\boldsymbol{e}}(\mathrm{softmax}(\mathbf{z}(t)), t)\|^2$$

$$= \mathop{\mathbb{E}}_{t \sim U(0,1)} \mathop{\mathbb{E}}_{\mathbf{z}(t) \sim q(\mathbf{z}(t)|\mathbf{x})} K\beta_1 t \|\boldsymbol{e}_\mathbf{x} - \hat{\boldsymbol{e}}_s(\mathbf{z}(t), t)\|^2,$$

where $q(\mathbf{z}(t)|\mathbf{x}) = \mathcal{N}(\beta(t)(K\boldsymbol{e}_\mathbf{x} - 1), \beta(t)K\boldsymbol{I})$ whose score function is given by

$$\nabla_\mathbf{z} \log q(\mathbf{z}(t)|\mathbf{x}) = -\frac{\mathbf{z}(t)}{K\beta(t)} + \boldsymbol{e}_\mathbf{x} - \frac{1}{K}. \tag{50}$$

Let

$$\hat{\boldsymbol{s}}(\mathbf{z}(t), t) := -\frac{\mathbf{z}(t)}{K\beta(t)} + \hat{\boldsymbol{e}}_s(\mathbf{z}(t), t) - \frac{1}{K}. \tag{51}$$

Substituting $\nabla_\mathbf{z} \log q(\mathbf{z}(t)|\mathbf{x})$ and $\hat{\boldsymbol{s}}(\mathrm{softmax}(\mathbf{z}(t)), t)$ into the loss function yields

$$\mathcal{L}^\infty(\mathbf{x}) = \mathop{\mathbb{E}}_{t \sim U(0,1)} \mathop{\mathbb{E}}_{\mathbf{z}(t) \sim q(\mathbf{z}(t)|\mathbf{x})} K\beta_1 t \|\boldsymbol{e}_\mathbf{x} - \hat{\boldsymbol{e}}_s(\mathbf{z}(t), t)\|^2$$

$$= \mathop{\mathbb{E}}_{t \sim U(0,1)} \mathop{\mathbb{E}}_{\mathbf{z}(t) \sim q(\mathbf{z}(t)|\mathbf{x})} K\beta_1 t \|\nabla_\mathbf{z} \log q(\mathbf{z}(t)|\mathbf{x}) - \hat{\boldsymbol{s}}(\mathbf{z}(t), t)\|^2. \tag{52}$$

The loss in Eq. (52) performs the denosing score matching (DSM), and according to Vincent (2011), the optimal solution of it w.r.t. $\hat{s}$ is the score of the distribution of $\mathbf{z}(t)$. Inspecting Eq. (51), we find that there is a one-to-one correspondance between the parameterized score function $\hat{s}$ in the DSM loss and the function $\hat{e}_s$ in BFN, showing that the optimization of BFN is equivalent to that of DM. $\qquad\square$

B.3. Proof of Proposition 5.3

Proof. Recall that the BFN sampler is defined in Eqs. (13) and (14), and $\beta(t) = \beta_1(1 - t)^2$. If we remove the categorical sampling step in Eq. (13), the sampling rule becomes

$$\mathbf{z}_i = \mathbf{z}_{i-1} + (\beta(t_i) - \beta(t_{i-1}))(K\hat{\boldsymbol{e}}_s(\mathbf{z}_{i-1}, t_{i-1}) - 1) + \sqrt{K(\beta(t_i) - \beta(t_{i-1}))}\mathbf{u}_i, \tag{53}$$

where $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Next, to show that the update rule in Eq. (53) is a first-order discretization of the reverse SDE, we write the SDE defined in Eq. (29) in the integral form as follows for any $0 \le t < s \le 1 - \eta$:

$$\mathbf{z}(t) = \mathbf{z}(s) - \int_s^t L(\tau)^2 \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) - \frac{1}{K}\right] \mathrm{d}\tau + \int_s^t L(\tau) \mathrm{d}\bar{\boldsymbol{w}}(\tau). \tag{54}$$

Note that the Itô integral follows the Gaussian distribution, and by Eq. (26), we know that

$$\int_s^t L(\tau) \mathrm{d}\bar{\boldsymbol{w}}(\tau) \sim \mathcal{N}\left(\mathbf{0}, \int_t^s L(\tau)^2 \mathrm{d}t\boldsymbol{I}\right) = \mathcal{N}\left(\mathbf{0}, K(\beta(t) - \beta(s))\boldsymbol{I}\right).$$

Then, by approximating $\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) = \hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + O(\tau - s)$ in Eq. (54), we find

$$\mathbf{z}(t) = \mathbf{z}(s) - \int_s^t L(\tau)^2 \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + O(\tau - s) - \frac{1}{K}\right] \mathrm{d}\tau + \int_s^t L(\tau) \mathrm{d}\bar{\boldsymbol{w}}(\tau)$$

$$= \mathbf{z}(s) + K(\beta(t) - \beta(s)) \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + O(t - s) - \frac{1}{K}\right] + \sqrt{K(\beta(t) - \beta(s))}\mathbf{u}_s$$

$$= \mathbf{z}(s) + (\beta(t) - \beta(s)) [K\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) - 1] + \sqrt{K(\beta(t) - \beta(s))}\mathbf{u}_s + O((t - s)^2),$$

---

[5]Note that $\boldsymbol{\theta}$ lies in a low-dimensional simplex, so the "density" should be defined w.r.t. a low-dimensional Lebesgue measure. We intend to define the distribution q_F by a sampling procdure as introduced later.

where $\mathbf{u}_s \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Now, the proof is completed by setting $t = t_i$ and $s = t_{i-1}$ in the above equation, and comparing the result with Eq. (53). □

B.4. Derivation of SDE-BFN-Solvers on Discrete Data

In this section we derive the SDE-BFN-Solvers of solving Eq. (29). We shall highlight that the SDE to be solved is different from those solved in BFNs and DMs on continuous data, as it contains no linear terms. We also follow the recipe of DPM-Solvers (Lu et al., 2022b;c) in reducing the discretization error to analytically simplify the equation as much as possible, instead of approximating it directly. The derived algorithms can be found in Algorithm 4, 5.

As we have seen in Appendix. B.3, the BFN sampler without the categorical sampling step (Eq. (53)) is the desired first-order solver, which is called SDE-BFN-Solver1. To further reduce the discretization error, we consider the first-order approximation of $\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau)$ in Eq. (54):

$$\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) = \hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + \hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(s), s)(\tau - s) + O((\tau - s)^2), \tag{55}$$

where $\hat{\boldsymbol{e}}_s^{(1)}$ is the derivative of $\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s)$ w.r.t. s. Then, setting $t = t_i$ and $s = t_{i-1}$, Eq. (54) becomes

$$\begin{aligned}
\mathbf{z}(t_i) &= \mathbf{z}(t_{i-1}) - \int_{t_{i-1}}^{t_i} L(\tau)^2 \left[\hat{\boldsymbol{e}}_s(\mathbf{z}(t_{i-1}), t_{i-1}) + \hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(t_{i-1}), t_{i-1})(\tau - t_{i-1}) - \frac{1}{K} \right] \mathrm{d}\tau \\
&\quad + \int_{t_{i-1}}^{t_i} L(\tau) \, \mathrm{d}\bar{\boldsymbol{w}}(\tau) + O((t_i - t_{i-1})^3) \\
&= \mathbf{z}(t_{i-1}) + (\beta(t_i) - \beta(t_{i-1})) \left[K\hat{\boldsymbol{e}}_s(\mathbf{z}(t_{i-1}), t_{i-1}) - 1 \right] \\
&\quad - \frac{1}{3} K\beta_1(t_i - t_{i-1})^2(t_{i-1} + 2t_i - 3)\hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(t_{i-1}), t_{i-1}) \\
&\quad + \sqrt{K(\beta(t_i) - \beta(t_{i-1}))} \mathcal{N}(0, \boldsymbol{I}) + O((t_i - t_{i-1})^3).
\end{aligned}$$

Finally, we approximate the derivative $\hat{\boldsymbol{e}}_s^{(1)}$ by a finite difference as follows to obtain the final discretization algorithm, namely SDE-DPM-Solver2.

$$\hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}_{i-1}, t_{i-1}) \approx \frac{\hat{\boldsymbol{e}}_s(\mathbf{z}_{i-2}, t_{i-2}) - \hat{\boldsymbol{e}}_s(\mathbf{z}_{i-1}, t_{i-1})}{t_{i-2} - t_{i-1}}. \tag{56}$$

B.5. Derivation of BFN-Solvers on Discrete Data

Eq. In this section, we derive the BFN-Solvers on discrete data using the following integral form of the parameterized ODE in (30).

$$\mathbf{z}(t) = \frac{1 - t}{1 - s} \mathbf{z}(s) + \beta_1(1 - t)(t - s) - K\beta_1(1 - t) \int_s^t \hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) \, \mathrm{d}\tau. \tag{57}$$

BFN-Solver1 The first-order solver approximates $\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau)$ in the above integral by $\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s)$ directly, yielding the following update rule:

$$\mathbf{z}_i = \frac{1 - t_i}{1 - t_{i-1}} \mathbf{z}_{i-1} + \beta_1(1 - t_i)(t_i - t_{i-1})(1 - K\hat{\boldsymbol{e}}_s(\mathbf{z}_{i-1}, t_{i-1})). \tag{58}$$

BFN-Solver2 The second-order solver approximates $\hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau)$ with $\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + \hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(s), s)(\tau - s)$, where $\hat{\boldsymbol{e}}_s(\mathbf{z}(s), s)$ is the derivative of $\hat{\boldsymbol{e}}_s$ w.r.t. s. Using such an approximation, we have

$$\begin{aligned}
\int_s^t \hat{\boldsymbol{e}}_s(\mathbf{z}(\tau), \tau) \, \mathrm{d}\tau &= \int_s^t \hat{\boldsymbol{e}}_s(\mathbf{z}(s), s) + \hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(s), s)(\tau - s) + O(\tau - s)^2 \, \mathrm{d}\tau \\
&= \hat{\boldsymbol{e}}_s(\mathbf{z}(s), s)(t - s) + \hat{\boldsymbol{e}}_s^{(1)}(\mathbf{z}(s), s)\frac{(t - s)^2}{2} + O((t - s)^3). \tag{59}
\end{aligned}$$

Following the method used in DPM-Solver (Lu et al., 2022b), we use an intermediate time $r \in (s, t)$ to approximate the derivative term. We use the first-order approximation in Eq. (58) to compute $\mathbf{z}(r)$ and the finite difference to estimate the

---

Algorithm 4 SDE-BFN-Solver1 (on discrete data)

Require: time steps $\{t_i\}_{i=0}^{M}$, from $t_0 = 1 - \eta$ to $t_M = 0$, model $\hat{e}_s(\mathbf{z}, t)$, $\beta(t) = \beta_1(1-t)^2$
$\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, K\beta(t_0)\mathbf{I})$
for $i = 1$ **to** $M - 1$ **do**
    $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{z}_i = \mathbf{z}_{i-1} + (\beta(t_i) - \beta(t_{i-1}))(K\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})) - 1) + \sqrt{K(\beta(t_i) - \beta(t_{i-1}))}\mathbf{u}_i$
end for
$\hat{\mathbf{x}} = \operatorname{argmax}(\hat{e}_s(\mathbf{z}_{M-1}, t_{M-1}))$
return $\hat{\mathbf{x}}$

Algorithm 5 SDE-BFN-Solver2 (on discrete data)

Require: time steps $\{t_i\}_{i=0}^{M}$, from $t_0 = 1 - \eta$ to $t_M = 0$, model $\hat{e}_s(\mathbf{z}, t)$, $\beta(t) = \beta_1(1-t)^2$
$\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, K\beta(t_0)\mathbf{I})$, Initialize an empty buffer Q
$\mathbf{u}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
$Q \leftarrow \hat{e}_s(\mathbf{z}_0, t_0)$
$\mathbf{z}_1 = \mathbf{z}_0 + (\beta(t_1) - \beta(t_0))(K\hat{e}_s(\mathbf{z}_0, t_0)) - 1) + \sqrt{K(\beta(t_1) - \beta(t_0))}\mathbf{u}_i$
for $i = 2$ **to** $M - 1$ **do**
    $D_1 = \frac{\hat{e}_s(\mathbf{z}_{i-2}, t_{i-2}) - \hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})}{t_{i-2} - t_{i-1}}$
    $\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{z}_i = \mathbf{z}_{i-1} + (\beta(t_i) - \beta(t_{i-1}))(K\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})) - 1) - \frac{1}{3}K\beta_1(t_i - t_{i-1})^2(t_{i-1} + 2t_i - 3)D_1 + \sqrt{K(\beta(t_i) - \beta(t_{i-1}))}\mathbf{u}_i$
    If $i < M - 1$, then $Q \leftarrow \hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})$
end for
$\hat{\mathbf{x}} = \operatorname{argmax}(\hat{e}_s(\mathbf{z}_{M-1}, t_{M-1}))$
return $\hat{\mathbf{x}}$

---

derivative $\hat{e}_s^{(1)}$:

$$\mathbf{z}(r) = \frac{1-t}{1-s}\mathbf{z}(s) + \beta_1(1-r)(r-s)(1 - K\hat{e}_s(\mathbf{z}(s), s)) + O((r-s)^2), \tag{60}$$

$$\hat{e}_s^{(1)}(\mathbf{z}(s), s) = \frac{\hat{e}_s(\mathbf{z}(r), r) - \hat{e}_s(\mathbf{z}(s), s)}{r-s} + O(r-s). \tag{61}$$

Combining the above equations with Eq. (59), we find that

$$\int_s^t \hat{e}_s(\mathbf{z}(\tau), \tau)\, d\tau = \hat{e}_s(\mathbf{z}(s), s)(t-s) + \frac{(\hat{e}_s(\mathbf{z}(r), r) - \hat{e}_s(\mathbf{z}(s), s))(t-s)^2}{2(r-s)} + O((t-s)^3). \tag{62}$$

Finally, let $\eta > 0$ be an arbitrarily small constant and choose time steps $\{t_i\}_{i=0}^{M}$ from $t_0 = 1 - \eta$ to $t_M = 0$. Given an initial value sample, $\{\mathbf{z}_i\}_{i=1}^{M}$ is computed iteratively as follows, by choosing $(s, t, r) := \left(t_{i-1}, t_i, \frac{t_i + t_{i-1}}{2}\right)$ for each i in the above derivation.

$$\mathbf{z}_{i-1/2} = \frac{1 - t_{i-1/2}}{1 - t_{i-1}}\mathbf{z}_{i-1} + \beta_1(1 - t_{i-1/2})(t_{i-1/2} - t_{i-1})(1 - K\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})), \tag{63}$$

$$\mathbf{z}_i = \frac{1 - t_i}{1 - t_{i-1}}\mathbf{z}_{i-1} + \beta_1(1 - t_i)(t_i - t_{i-1}) - c(t_i)(t_i - t_{i-1})\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})$$

$$- c(t_i)\frac{(t_i - t_{i-1})^2}{2(t_{i-1/2} - t_{i-1})}(\hat{e}_s(\mathbf{z}_{i-1/2}, t_{i-1/2}) - \hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})). \tag{64}$$

where $t_{i-1/2} = (t_i + t_{i-1})/2$ and $c(t) = K\beta_1(1 - t)$.

Algorithm 6 BFN-Solver1 (on discrete data)

Require: time steps $\{t_i\}_{i=0}^M$, from $t_0 = 1 - \eta$ to $t_M = 0$, model $\hat{e}_s(\mathbf{z}, t)$, $\beta(t) = \beta_1(1-t)^2$
$\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, K\beta(t_0)\boldsymbol{I})$
for $i = 1$ **to** $M - 1$ **do**
$\quad \tilde{\mathbf{z}}_i = \frac{1-t_i}{1-t_{i-1}}\mathbf{z}_{i-1} + \beta_1(1-t_i)(t_i - t_{i-1})(1 - K\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1}))$
end for
$\hat{\mathbf{x}} = \text{argmax}(\hat{e}_s(\mathbf{z}_{M-1}, t_{M-1}))$
return $\hat{\mathbf{x}}$

Algorithm 7 BFN-Solver2 (on discrete data)

Require: time steps $\{t_i\}_{i=0}^M$, from $t_0 = 1 - \eta$ to $t_M = 0$, model $\hat{e}_s(\mathbf{z}, t)$, $\beta(t) = \beta_1(1-t)^2$
$\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, K\beta(t_0)\boldsymbol{I})$
for $i = 1$ **to** $M - 1$ **do**
$\quad t_{i-1/2} = (t_i + t_{i-1})/2$
$\quad c_i = K\beta_1(1-t_i)$
$\quad D_1 = \frac{\hat{e}_s(\mathbf{z}_{i-1/2}, t_{i-1/2}) - \hat{e}_s(\mathbf{z}_{i-1}, t_{i-1})}{t_{i-1/2} - t_{i-1}}$
$\quad \mathbf{z}_i = \frac{1-t_i}{1-t_{i-1}}\mathbf{z}_{i-1} + \beta_1(1-t_i)(t_i - t_{i-1}) - c_i(t_i - t_{i-1})\hat{e}_s(\mathbf{z}_{i-1}, t_{i-1}) - \frac{c_i(t_i - t_{i-1})^2}{2}D_1$
end for
$\hat{\mathbf{x}} = \text{argmax}(\hat{e}_s(\mathbf{z}_{M-1}, t_{M-1}))$
return $\hat{\mathbf{x}}$

C. Experimental Details

C.1. Choices of the Initialization Distribution

Since the exact distribution of $\boldsymbol{\mu}(1-\eta)$ is unknown, we need to choose an approximation of it as the initialization distribution. The following proposition identifies the best initial distribution among isotropic Gaussian distributions.

Proposition C.1. *Let $p_t(\boldsymbol{\mu}) = \mathbb{E}_{\mathbf{x} \sim p_{data}} q_F(\boldsymbol{\mu}|\mathbf{x}, \gamma(t))$ be the distribution of $\boldsymbol{\mu}(t)$ in Eq. (8), then $q_t(\boldsymbol{\mu}) := \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}^*(t), (\sigma^*(t))^2\boldsymbol{I})$ minimizes the Kullback-Leibler (KL) divergence between p_t and any isotropic Gaussian distributions, where*

$$\boldsymbol{m}^*(t) = \gamma(t)\boldsymbol{m}_{data}, \quad and \quad (\sigma^*(t))^2 = \gamma(t)(1 - \gamma(t)) + \gamma(t)^2\frac{\text{Tr}\,\Sigma_{data}}{D},$$

D is the dimensionality of $\boldsymbol{\mu}$, and $\boldsymbol{m}_{data}, \Sigma_{data}$ are the mean and the covariance matrix of the data distribution p_{data}, respectively. In other words, we have

$$(\boldsymbol{m}^*(t), \sigma^*(t)) = \underset{\boldsymbol{m}, \sigma}{\arg\min}\, D_{\text{KL}}(p_t \| \mathcal{N}(\boldsymbol{m}, \sigma^2\boldsymbol{I})), \tag{65}$$

where D_{KL} is the Kullback-Leibler divergence.

Proof. Since $p_F(\boldsymbol{\mu}(t)|\mathbf{x}, \gamma(t)) = \mathcal{N}(\gamma(t)\mathbf{x}, \gamma(t)(1 - \gamma(t))\boldsymbol{I})$, the KL divergence can be written as

$$D_{\text{KL}}(p_t \| \mathcal{N}(\boldsymbol{m}, \sigma^2\boldsymbol{I})) = -\underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \log \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{m}, \sigma^2\boldsymbol{I}) + \underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \log p_t(\boldsymbol{\mu})$$

$$= \underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \left[\frac{D}{2}\log(2\pi) + D\log\sigma + \frac{\|\boldsymbol{\mu} - \boldsymbol{m}\|^2}{2\sigma^2} \right] + \underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \log p_t(\boldsymbol{\mu}),$$

where D is the dimensionality of $\boldsymbol{\mu}$. By taking the derivative, we know the minimizer is

$$\boldsymbol{m}^*(t) = \underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \boldsymbol{\mu} = \gamma(t)\boldsymbol{m}_{data},$$

$$(\sigma^*(t))^2 = \frac{1}{D} \underset{\boldsymbol{\mu} \sim p_t}{\mathbb{E}} \|\boldsymbol{\mu} - \boldsymbol{m}^*(t)\|^2 = \gamma(t)(1 - \gamma(t)) + \gamma(t)^2\frac{\text{Tr}\,\Sigma_{data}}{D}.$$

$\square$

However, the optimal distribution in the above proposition depends on the mean and the covariance matrix of the data distribution, which are still unknown. Fortunately, we find that for sufficiently small η, the effect of data-dependent terms is negligible. Since when $\eta \to 0$ we know $\gamma(t) \to 0$ and by the proposition the optimal variance is $\gamma(t)(1 - \gamma(t)) + o(\gamma(t))$, so a typical sample from the optimal distribution has the norm $\sqrt{D\gamma(t)(1 - \gamma(t))}$, which dominates the optimal mean $\gamma(t)\boldsymbol{m}_{data}$ as $\gamma(t) \to 0$. Therefore, we suggest the data-independent distribution $\mathcal{N}(\mathbf{0}, \gamma(t)(1 - \gamma(t))\boldsymbol{I})$ as the initial distribution.

We note that although the above proposition is proved for the BFN with continuous data, a similar result also holds for discrete data.

C.2. Final Step of Sampling

Theoretically, we need to solve the reverse SDEs or ODEs from time $1 - \eta$ to 0 to generate samples. In practice, the BFN sampler adds a step where it uses the state obtained from the initial n iterations, denoted by $\boldsymbol{\theta}_n$, to run the network an additional time. The output from this final network is then used as the final sample. We follow this sampling trick as our default sampling method at the final step.

C.3. User Study for Text Generation

In this section, we give the details of the user study designed to compare the quality of text generated from BFN-Solvers and BFN from the human perspective. We instruct participants to select the text samples they perceive as higher quality, providing them with authentic examples from the test dataset to serve as benchmarks for their evaluations. For each study, we collect 500 human answers from 10 participants to ensure a sufficient sample size to analyze the preferences.

C.4. Ablation of Categorical Sampling Step in BFN Sampler on Discrete Data

In this section, we present the detailed results of Sec. 6.2 in Table 3. We explicitly analyze the sampling dynamics of the original BFN sampler with and without CS. We generate two sets of trajectories, denoted as $\mathbf{z}_{\text{w/cat}}(t)$ and $\mathbf{z}_{\text{w/o cat}}(t)$, using identical Gaussian noises and then compute the 1-norm of the difference between these trajectories at each time point t and sum them. We replicate the sampling 1000 times independently and compute the mean, which represents the disparity between two samplers. As shown in Table 4, $\|\mathbf{z}_{\text{w/cat}} - \mathbf{z}_{\text{w/o cat}}\|_1$ gets smaller as the step size decreases.

Table 3. **Ablation of the categorical sampling (CS) step in the BFN sampler on discrete data.** We present the SA↑.

NFE	10	12	20	30	40	50	100	200	1000
BFN	66.27	70.73	79.54	82.65	84.01	84.88	**86.14**	86.50	86.69
BFN W/O CS (OURS)	**70.67**	**73.63**	**80.86**	**83.26**	**84.27**	**84.90**	86.11	**86.76**	**86.77**

Table 4. **Ablation of the categorical sampling (CS) step in the BFN sampler on discrete data.** We present the $L1$ norm of the difference between sampling trajectories of the original BFN sampler with and without CS, varying the step size.

STEP SIZE	0.1	0.05	0.02	0.01	0.005	0.002	0.001
$\|\mathbf{z}_{\text{W/CAT}} - \mathbf{z}_{\text{W/O CAT}}\|_1$	0.18	0.16	0.13	0.10	0.087	0.066	0.049

D. Additional results

D.1. Additional Results on Continuous Dataset

In this section, we present the detailed results of Sec. 6.2. As shown in Tab. 5, our proposed methods obtain the best results under all NFE.

Table 5. **Image generation results on continuous CIFAR-10 dataset.** Sampling quality is measured by FID ↓, varying the number of function evaluations (NFE). We **bold** the best result under the corresponding setting. For instance, we underline the result of BFN at 50 steps and our BFN-Solvers2++ solver at 10 steps, where we achieve a speed-up of 5 times.

NFE	10	20	50	100	200	500	1000
BFN	253.23	146.19	<u>74.81</u>	45.50	27.98	16.24	13.05
SDE-BFN-Solver++2 (ours)	77.19	49.73	34.29	**23.09**	**16.07**	**11.94**	**10.86**
BFN-Solver++1 (ours)	66.27	49.18	36.27	30.43	26.62	21.49	20.39
BFN-Solver++2 (ours)	**<u>55.87</u>**	**43.63**	**33.01**	27.92	23.89	19.48	20.51

Table 6. **Text generation results on discrete text8 dataset.** Sampling quality is measured by SA ↑, varying the number of function evaluations (NFE). For instance, we underline the result of BFN at 1000 steps and our SDE-BFN-Solvers2 solver at 50 steps, where we achieve a speed-up of 20 times.

NFE	10	12	20	30	40	50	100	200	1000
BFN	66.27	70.73	79.54	82.65	84.01	84.88	86.14	86.50	<u>86.69</u>
SDE-BFN-Solver2 (ours)	80.29	82.39	**85.27**	**86.03**	**86.63**	**<u>86.82</u>**	**87.04**	**86.74**	**86.78**
BFN-Solver1 (ours)	78.61	80.21	82.46	83.76	84.18	84.53	85.25	85.39	85.61
BFN-Solver2 (ours)	**82.27**	**83.23**	84.46	85.12	85.34	85.46	85.53	85.57	85.60

Table 7. **Empirical study of η.** We present the FID ↓ results of image generation on continuous CIFAR-10 dataset varying η.

	NFE=50			NFE=500		
Values of η	0.01	0.001	0.0001	0.001	0.0001	0.00001
BFN-Solver++1 (ours)	50.79	48.68	111.14	21.49	49.13	220.37
BFN-Solver++2 (ours)	48.29	43.84	106.44	19.48	45.95	214.17
SDE-BFN-Solver++2 (ours)	41.23	36.13	34.29	12.33	14.49	27.39

D.2. Additional Results of Discrete Dataset

In this section, we present the detailed results of Sec. 6.3. As shown in Tab. 6, our best solver significantly outperforms the original BFN sampler with a few (e.g., 10) NFEs under the SA metric.

D.3. Analysis of η on Continuous Data

In this section, we provide an analysis of the hyperparameter η for continuous data. According to the analyses in Appendix C.1, we start sampling from the approximate prior distribution $\tilde{p}(\boldsymbol{\mu}(1-\eta)) = \mathcal{N}(\mathbf{0}, \gamma(1-\eta)(1-\gamma(1-\eta))\boldsymbol{I})$. Firstly, we present the FID results on the continuous CIFAR-10 dataset. We show the results with 50 and 500 NFE for efficiency. As shown in Tab. 7, excessively small or large values of η detrimentally affect image generation quality.

In addition, we found that SDE-based method (i.e., SDE-BFN-Solver++2) is less sensitive to η compared to ODE-based methods (i.e., BFN-Solver++1, BFN-Solver++2). Assuming the discretization error is negligible, Nie et al. (2023) theoretically elucidates why SDE-based samplers outperform ODE-based samplers in sampling from the approximate prior distribution. This provides a theoretical foundation for our observations.

At last, as present in Sec. 6.2 and Tab. 5, after tuning η, ODE-based samplers still outperform SDE-based samplers and BFN baseline with a few NFEs (e.g., 10) under sample quality.

D.4. Analysis of η on Discrete Data

In this section, we provide an analysis of hyperparameter η for discrete data. According to the analyses in Appendix C.1, we start sampling from the approximate prior distribution $\tilde{p}(\mathbf{z}(1-\eta)) = \mathcal{N}(\mathbf{0}, K\beta(1-\eta)\boldsymbol{I})$ which works well empirically. As shown in Tab. 8, For lower NFEs(20) or higher NFEs(200), the variation in η shows minor differences in SA for BFN-Solver1, BFN-Solver2 and SDE-BFN-Solver2. This suggests that the choice of η does not significantly impact the

Table 8. **Empirical study of** η**.** We present the SA ↑ results of text generation on continuous text8 dataset varying η.

	NFE=20			NFE=200		
VALUES OF η	0.01	0.001	0.0001	0.01	0.001	0.0001
BFN-SOLVER1 (OURS)	82.56	82.39	82.46	85.40	85.38	85.39
BFN-SOLVER2 (OURS)	84.46	84.43	84.46	85.57	85.56	85.52
SDE-BFN-SOLVER2 (OURS)	84.98	85.28	85.27	86.65	86.68	86.74

performance of these solvers on discrete data.

D.5. Random Samples

Additional sampling results on CIFAR-10 are shown in Fig. 7.

Additional sampling results on text8 are shown in Figs. 8 and 9.

NFE=50   NFE=200   NFE=1000

Figure 7. Randomly generated images by BFN and BFN-Solvers (ours) with **50**, **200**, **1000** NFEs, using the same pre-trained model on CIFAR-10.

```
ected_a_la_a_capo_off_one_diary_about_vein_ille_ramp_in_nine_two
_one_could_solutely_be_credited_on_becoming_alro_terrorist_high_
rankengeros_mikara_even_tool_sword_bitter_played_the_when_you_di
e_teuru_you_re_not_wisdom_but_you_used_must_choose_to_feel_why_y

of_israel_against_throne_festiny_sadara_meign_voi_gerand_crisis_
of_the_killing_cowkey_he_isamy_thu_serpent_man_of_the_us_three_r
d_indoorh_gundams_limz_mult_wied_ropes_and_univ_used_gundam_line
s_external_links_downtille_rod_yamatonf_at_ken_cruftweb_site_que

lligisms_et_six_zero_ff_is_noose_the_first_was_to_form_a_more_bi
tter_phase_the_oxoset_like_musical_airs_in_the_final_phase_of_th
e_foster_and_thu_pricks_out_his_exariments_von_in_the_lifte_steo
_of_alp_death_sea_in_the_baltic_and_inudanu_trees_iv_five_five_g

rties_university_toys_the_no_tops_newly_for_most_people_than_are
_the_speakers_and_hike_inside_to_upper_c_and_even_part_up_to_lea
d_ridge_longests_but_face_can_not_be_heard_only_a_small_number_o
f_people_beating_by_methane_steen_has_taken_no_one_to_see_actual

e_got_the_fry_prizes_en_sonor_a_fmz_the_shyierkin_the_brothers_e
rich_hormanut_olipy_meducine_which_originated_as_one_of_nust_cha
racters_in_the_fimms_sonwycke_lihenwer_a_leona_pogganut_l_c_kazy
nutt_posz_bord_s_book_the_dust_mountains_is_a_polish_new_school_
```

(a) BFN

```
first_published_in_for_instance_the_earliest_of_the_word_usage_i
n_dext_was_the_si_unit_anda_and_the_errosphere_then_pried_was_fo
r_many_measurements_such_as_sound_unit_or_soive_pressure_the_hyp
othetical_inverses_of_extensive_note_as_there_are_no_clear_units

him_and_manifestation_most_belonging_to_the_western_yinying_grou
p_the_chinese_jeers_and_the_pit_the_petin_jeers_in_the_center_th
ey_are_mainly_in_slated_off_parts_of_the_world_kehak_and_council
_the_jews_reap_several_name_aliers_judlusm_bank_of_scerpian_seta

ed_with_a_car_also_known_as_cats_in_hong_kong_pass_the_bus_syste
m_directly_from_kung_hongpass_a_hong_kong_taiwan_taxi_toualin_ta
le_an_ecroke_will_be_emergencies_purchase_his_tale_s_used_disusi
_e_flowers_in_ease_of_swimming_many_people_are_used_to_hours_on_

calvin_s_list_for_lymia_sells_his_father_was_going_a_snapper_and
_crally_for_the_joys_of_the_then_shark_falls_for_his_mob_honest_
roundings_of_customs_penalties_he_attacks_him_on_his_way_to_taxi
cal_raysel_warning_him_troubled_by_the_back_of_a_bumper_ry_chair

joint_committee_on_american_fair_land_activists_pro_ott_avoided_
voting_and_also_in_a_good_letter_now_end_of_the_campaign_two_zer
o_zero_one_wiscoe_nimes_bob_book_and_franklin_roosevelt_and_near
ly_the_four_zero_zero_more_hour_four_th_edition_bob_storie_arise
```

(b) SDE-BFN-Solver2 (ours)

```
arting_evidence_for_local_schools_initial_archaeological_series_
reputed_roman_raising_power_of_carthage_in_cornwall_or_swynsea_t
enis_references_to_roman_authorities_tacitus_mxvi_tull_ad_plani_
s_honoribis_bona_xxcutiningles_dag_dig_s_goats_british_nomores_b

_of_technological_species_wes_know_in_pre_existing_fossil_patter
n_thus_an_intevergence_of_that_begion_orbiting_the_germ_paraharm
_is_seen_as_the_method_of_thruvitus_learning_we_still_experience
_formal_tachymulty_which_is_would_be_turned_upon_by_the_aerologi

_six_one_three_tets_telecan_evil_date_airplane_accident_computer
_three_four_zero_four_six_painted_film_drinetware_on_duck_morlan
d_was_flinted_inaocho_one_nine_eight_zero_s_disarke_by_such_air_
prices_as_dogative_in_a_condition_to_ant_humanism_political_spim

six_zero_yelena_charariovinn_russian_clown_one_nine_five_four_he
rman_prichheae_kick_writer_don_reves_american_national_cartosn_o
ne_nine_six_zero_mark_holman_australian_automaker_one_nine_six_z
ero_matt_beard_american_actor_d_one_nine_nine_zero_one_nine_six_

osts_now_amound_throughout_the_natural_sciences_and_malletta_man
y_universities_in_the_uk_and_were_scored_for_postby_a_causea_gav
edeup_to_more_than_five_million_as_the_first_post_approved_one_o
f_one_act_weft_of_the_million_was_shortinesed_by_neal_collidge_p
```

(c) BFN-Solver1 (ours)

```
arting_evidence_for_local_schools_initial_archaeological_series_
reputed_roman_raising_power_of_carthage_in_cornwall_or_swansea_t
unis_references_to_roman_authorities_tacitus_xxvi_tull_ad_plany_
s_honorifis_to_a_positing_elen_dam_dia_s_goa_daughter_from_red_b

_of_technological_species_responds_to_pre_existing_fossil_patter
n_thus_an_inteligence_of_that_begins_orbiting_the_ierm_paraharm
_is_seen_as_the_netsup_of_th_ss_this_viewing_we_still_experience
_formal_tachymuncy_which_is_would_be_turned_upon_by_the_aerologi

_six_one_three_mets_were_an_evil_date_airplane_accident_at_pipot
_three_four_zero_four_six_faineen_killed_when_are_on_duck_morlan
d_was_fainted_inancho_one_nine_eight_zero_indiearingly_such_air_
prices_as_invasive_in_a_police_enclosant_pacafism_political_crim

ive_zero_yelena_charariovinn_russian_clown_one_nine_five_four_go
rdon_priest_am_kick_writer_the_rives_american_national_tartist_o
ne_nine_six_zero_mark_holman_australian_automaker_one_nine_six_z
ero_matt_beard_american_actor_d_one_nine_nine_zero_one_nine_six_

osts_now_around_throughout_the_natural_sciences_and_mauletta_man
y_universities_in_the_uk_and_were_scored_for_football_causes_gav
ed_up_to_more_than_five_million_as_the_first_post_approved_one_o
f_one_two_zero_of_the_million_was_shortinesed_by_seal_collidge_p
```

(d) BFN-Solver2 (ours)

Figure 8. Randomly generated texts by BFN and BFN-Solvers (ours) with **10** NFEs, using the same pre-trained models on text8.

```
b_one_four_eight_eight_one_six_two_nine_albrecht_spener_german_t
heologian_b_one_five_eight_zero_one_six_three_five_aighaz_mahams
udart_king_of_kochi_singh_zugu_warder_and_politician_b_one_five_
five_two_one_six_four_five_ashokawa_sunemuna_japanese_silence_is

s_the_babylonian_talmud_to_one_more_than_the_first_half_of_six_t
wo_five_with_a_recent_interpretat_to_date_the_decline_of_the_pop
ulations_from_the_first_time_however_the_third_cultural_branch_r
epresenting_a_flux_in_the_number_of_workers_appears_to_have_stro

e_citizens_trained_under_personal_safety_laws_less_than_one_zero
_years_of_potency_but_strictly_speaking_the_livis_taskforce_used
_for_magistations_amounting_to_older_magistrates_nineteen_years_
later_working_on_home_education_and_and_various_reforms_across_t

ne_at_least_in_daily_use_of_the_important_lakes_named_after_mann
ipies_image_flag_of_this_island_state_of_macedonia_syria_s_itali
an_shi_h_bela_divided_sixth_horns_edegis_turke_howkver_kjan_bent
unhorle_deneves_mykior_subias_vlashbirerojs_in_one_five_seven_tw

ld_county_soub_army_britain_christian_democratic_party_moral_phi
losophy_scottish_philosophers_scaptons_dualistic_philosophers_ge
igntonians_physicians_george_burnell_leeves_physicists_british_m
athematicians_mathematical_theory_of_non_human_worlds_j_barnard_
```

(a) BFN

```
the_chambers_divisive_among_stephen_shepherd_who_is_still_in_con
tent_for_the_production_nicholas_de_france_in_one_nine_nine_nine
_trial_by_maria_dizvosa_two_zero_zero_one_the_shadow_in_the_simp
sons_two_zero_zero_three_trial_by_david_rockland_two_zero_zero_f

ork_after_its_commercial_and_initial_acquisition_by_fan_of_prese
nt_enlarged_by_graphical_publishers_that_now_it_was_hired_for_no
_third_idea_use_such_work_a_large_portion_of_microsoft_specifica
lly_an_adaptation_an_open_tp_program_as_a_requirement_for_the_su

er_than_the_now_here_mawes_psa_led_worldwide_however_motorcycle_
became_less_seen_in_formula_one_eight_factories_by_one_nine_one_
one_was_open_production_until_one_nine_one_three_designer_niel_j
ones_maldini_made_him_up_a_psa_car_and_in_one_nine_one_four_bega

n_man_mumahuadeed_wrote_that_this_may_say_he_has_made_of_a_horse
bow_against_use_which_is_wise_what_to_make_and_what_did_isn_t_do
_nothing_about_it_with_mind_and_what_used_to_exist_in_the_islami
c_world_philosophy_canton_p_two_five_five_for_now_with_mind_alwa

ernisler_german_pianist_d_one_nine_four_zero_one_nine_zero_eight
_lobe_te_seitel_canadian_bandleader_d_one_nine_six_eight_one_nin
e_one_three_samuel_pahn_american_author_d_one_nine_four_three_on
e_nine_one_four_thomas_dolf_beck_american_musician_and_band_desi
```

(b) SDE-BFN-Solver2 (ours)

```
cult_references_parbula_in_english_december_two_zero_zero_two_ve
rsion_two_zero_zero_two_with_illustrations_idea_users_dictionary
_of_colligions_devil_org_christianism_from_the_beginner_translat
ed_into_the_web_ten_quotes_from_hippocrate_richard_paul_geneva_w

her_sical_dating_is_found_byron_s_date_ch_one_nine_six_titus_of_
macedon_had_decided_to_have_a_babylonian_calendar_in_february_on
e_two_zero_five_so_according_to_some_sical_dating_the_greek_adda
ration_of_five_three_nine_one_bc_five_three_three_one_cut_him_bu

ion_three_etc_and_three_d_seven_zero_five_four_nine_nine_seven_f
ive_zero_one_riemann_euler_german_mathematician_and_an_astronome
r_who_worked_in_the_one_eight_th_century_after_two_fields_gone_i
n_line_mount_zero_and_a_bright_orbitary_more_commonly_moon_numbe

arting_evidence_for_roman_schools_initial_archaeological_series_
reputed_roman_raising_power_of_carthage_in_cornwall_or_swynsea_t
unis_references_to_roman_authorities_tacitus_christyla_ad_pliny_
s_honorifis_bond_xxcutiningles_dag_dig_s_goats_british_romores_b

_of_technological_species_responds_to_pre_existing_fossil_patter
n_thus_an_intelligence_of_that_begins_orbiting_the_germ_paradorm
_is_seen_as_the_extent_of_which_the_permlike_we_still_experience
_fossil_beings_and_which_we_would_be_turned_upon_by_the_aerologi
```

(c) BFN-Solver1 (ours)

```
cult_references_parbula_in_english_december_two_zero_zero_two_ve
rsion_two_zero_zero_two_with_illustrations_idea_users_dictionary
_of_colligions_devil_org_christianism_from_the_beginner_translat
ed_into_the_web_ten_quotes_from_hippocrate_richard_paul_geneva_w

her_sical_dating_is_found_byron_s_date_ch_one_nine_six_titus_of_
macedon_had_decided_to_have_a_babylonian_calendar_in_february_on
e_two_zero_five_so_according_to_some_sical_dating_the_greek_adda
ration_of_five_three_nine_one_bc_five_three_three_one_cut_him_bu

ion_three_etc_and_three_d_seven_zero_five_four_nine_nine_seven_f
ive_zero_one_riemann_euler_german_mathematician_and_an_astronome
r_who_worked_in_the_one_eight_th_century_after_two_fields_gone_i
n_line_mount_zero_and_a_bright_orbitary_more_commonly_moon_numbe

arting_evidence_for_roman_schools_initial_archaeological_series_
reputed_roman_raising_power_of_carthage_in_cornwall_or_swynsea_t
unis_references_to_roman_authorities_tacitus_christyla_ad_pliny_
s_honorifis_bond_xxcutiningles_dag_dig_s_goats_british_romores_b

_of_technological_species_responds_to_pre_existing_fossil_patter
n_thus_an_intelligence_of_that_begins_orbiting_the_germ_paradorm
_is_seen_as_the_extent_of_which_the_permlike_we_still_experience
_fossil_beings_and_which_we_would_be_turned_upon_by_the_aerologi
```

(d) BFN-Solver2 (ours)

Figure 9. Randomly generated texts by BFN and BFN-Solvers (ours) with **1000** NFEs, using the same pre-trained models on text8.