BDDM: BILATERAL DENOISING DIFFUSION MODELS FOR FAST AND HIGH-QUALITY SPEECH SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion probabilistic models (DPMs) and their extensions have emerged as competitive generative models yet confront challenges of efficient sampling. We propose a new bilateral denoising diffusion model (BDDM) that parameterizes both the forward and reverse processes with a scheduling network and a score network, which can train with a novel bilateral modeling objective. We show that the new surrogate objective can achieve a lower bound of the log marginal likelihood tighter than a conventional surrogate. We also find that BDDM allows inheriting pre-trained score network parameters from any DPMs and consequently enables speedy and stable learning of the scheduling network and optimization of a noise schedule for sampling. Our experiments demonstrate that BDDMs can generate high-fidelity audio samples with as few as 3 sampling steps. Moreover, comparing to other state-of-the-art diffusion-based neural vocoders, BDDMs produce comparable or higher quality samples indistinguishable from human speech, notably with only 7 sampling steps (143x faster than WaveGrad and 28.6x faster than DiffWave).

1 INTRODUCTION

Deep generative models have shown a tremendous advancement in speech synthesis (van den Oord et al., 2016; Kalchbrenner et al., 2018; Prenger et al., 2019; Kumar et al., 2019a; Kong et al., 2020b; Chen et al., 2020; Kong et al., 2021). Successful generative models can be mainly divided into two branches – the GAN (Goodfellow et al., 2014) based approach and the likelihood-based approach. The former uses an adversarial training procedure, but the training is sometimes unstable, and the relevant training objectives are not suitable to compare different GAN models; the latter uses log-likelihood or surrogate objectives for training, but they also have intrinsic limitations. For example, the autoregressive models (van den Oord et al., 2016; Kalchbrenner et al., 2018) have exceedingly slow sampling speed, and poor scaling properties on high-dimensional data, as its sampling is inherently a sequential process. Likewise, the flow-based models (Dinh et al., 2016; Kingma & Dhariwal, 2018; Chen et al., 2018; Papamakarios et al., 2021) rely on specialized architectures to build a normalized probability model so that the model training is less parameter-efficient and typically more difficult than other likelihood-based generative models (Bond-Taylor et al., 2021). On the other hand, for those using the surrogate objectives, such as the evidence lower bound in variational auto-encoders (Kingma & Welling, 2013; Rezende et al., 2014; Maaløe et al., 2019) and the contrastive divergence in energy-based models (Hinton, 2002; Carreira-Perpinan & Hinton, 2005), they improve inference speed but often yield lower-quality samples than autoregressive and GAN-based models and typically only work well for low-dimensional data (Bond-Taylor et al., 2021).

An up-and-coming class of likelihood-based models is the diffusion probabilistic models (DPMs) (Sohl-Dickstein et al., 2015), which shed light on the generation of high-quality samples comparable or even superior (Dhariwal & Nichol, 2021) to the state-of-the-art (SOTA) GAN-based models. Sohl-Dickstein et al. (2015) introduced the idea of using an iterative forward process for destroying the structure of a given distribution while learning the reverse process for restoring the structure in the data. With a different modeling objective, the score-based generative models (Song & Ermon, 2019) used a neural network trained with the score matching (Hyvarinen & Dayan, 2005) to produce samples via Langevin dynamics. As a seminal work of DPMs, Ho et al. (2020) proposed the denoising diffusion probabilistic models (DDPMs), which exploited a connection (Vincent, 2011) between

the above two ideas that the diffusion framework can be viewed as a specific approach to provide support to score matching. For the first time, DDPMs present high-quality image synthesis results on par with GAN models, especially on several higher-resolution image benchmarks. Subsequently, score matching and denoising diffusion models were applied to speech syntheses, such as Wavegrad (Chen et al., 2020) and a parallel contribution Diffwave (Kong et al., 2021), and demonstrated the capability of generating high fidelity audio samples outperforming non-autoregressive GAN-based models (Yamamoto et al., 2020; Kumar et al., 2019b; Yang et al., 2020; Bińkowski et al., 2020) meanwhile matching the quality of SOTA autoregressive methods (Chen et al., 2020).

Despite the compelling results, the diffusion models are two to three orders of magnitude slower than other generative models such as GANs and VAEs. The major limitation of the generative diffusion models is that it requires up to thousands of diffusion steps during training to learn the target distribution; therefore, a large number of reverse steps are often required at sampling time. Recently, extensive investigations have been conducted to accelerate the sampling process for efficiently generating high-quality outputs. In WaveGrad (Chen et al., 2020), a grid search algorithm was used to generate high-fidelity audio samples with six steps. From a different aspect, Song et al. (Song et al., 2021b) used a neural probability flow ODE to enable a fast deterministic multi-step sampling process. A parallel contribution called the denoising diffusion implicit models (DDIMs) (Song et al., 2021a) considered non-Markovian diffusion processes and used a subsequence of the noise schedule to accelerate the denoising process. A more recent work (Luhman & Luhman, 2021) explored a student-teacher method to distill the DDIMs sampling process into a single-step model.

Distinctively, we conceived that the reduction of sampling steps and the improvement of sampling quality significantly depends on the inference noise schedule. However, the critical inference noise schedule was conventionally tied with the predetermined training noise schedule and was suboptimal. We are thus motivated to investigate learning a noise schedule under the network training framework. With such an incentive, we propose the bilateral denoising diffusion models (BDDMs), which are derived from a bilateral modeling perspective – parameterizing the forward and reverse processes with a scheduling network and a score network. We show that a new lower bound for the log marginal likelihood tighter than the conventional ELBO can be obtained by training a scheduling network after optimizing the score network. Under the assumption of proposition 2 (see detailed descriptions in Section 3), our newly derived loss for the score network interestingly resembles the training loss of DDPMs, therefore the score network parameters can be directly inherited from a well-trained DDPM. Afterwards, our novel derived objective for the scheduling network converges very fast, and its training only adds a negligible overhead to DDPM's. Our neural vocoding experiments demonstrated that BDDMs can generate high-fidelity samples with as few as 3 sampling steps, and produce speech samples indistinguishable from human speech with only 7 sampling steps (143x faster than WaveGrad and 28.6x faster than DiffWave).

2 BACKGROUND

2.1 DIFFUSION PROBABILISTIC MODELS (DPMS)

Given i.i.d. samples $\{\boldsymbol{x}_0 \in \mathbb{R}^D\}$ from an unknown data distribution $p_{\text{data}}(\boldsymbol{x}_0)$, diffusion probabilistic models (DPMs) (Sohl-Dickstein et al., 2015) define a forward process $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$ that converts any complex data distribution into a simple, tractable distribution after T steps of diffusion. A reverse process $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ parameterized by θ is used to model the data distribution: $p_\theta(\boldsymbol{x}_0) = \int \pi(\boldsymbol{x}_T) \prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) d\boldsymbol{x}_{1:T}$, where $\pi(\boldsymbol{x}_T)$ is the prior distribution for starting the reverse process. Then, the variational parameters θ can be learned by maximizing the standard evidence lower bound (ELBO):

$$\mathcal{F}_{\text{elbo}} := \mathbb{E}_q \left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1) - \sum_{t=2}^{T} \mathbb{KL}\left(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) || p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right) - \mathbb{KL}\left(q(\boldsymbol{x}_T|\boldsymbol{x}_0) || \pi(\boldsymbol{x}_T)\right) \right]. \tag{1}$$

2.2 DENOISING DIFFUSION PROBABILISTIC MODELS (DDPMS)

As an extension to DPMs, de-noising diffusion probabilistic models (DDPMs) (Ho et al., 2020) applied the score matching technique (Hyvarinen & Dayan, 2005; Song & Ermon, 2019) to define

the reverse process. In particular, DDPMs considered a Gaussian diffusion process parameterized by a *noise schedule* $\boldsymbol{\beta} \in \mathbb{R}^T$ with $0 < \beta_1, \ldots, \beta_T < 1$:

$$q_{\boldsymbol{\beta}}(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) := \prod_{t=1}^{T} q_{\beta_t}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}), \quad \text{where} \quad q_{\beta_t}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) := \mathcal{N}(\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}). \tag{2}$$

Based on the nice property of isotropic Gaussians, one can directly express $\boldsymbol{x}_t$ in terms of $\boldsymbol{x}_0$:

$$\boldsymbol{x}_t = \alpha_t \boldsymbol{x}_0 + \left(1-\alpha_t^2\right)\boldsymbol{\epsilon}, \quad \text{where} \quad \alpha_t = \prod_{i=1}^{t} \sqrt{1-\beta_i} \quad \text{and} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{3}$$

To revert this forward process, DDPMs employ a score network[1] $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \alpha_t)$ to define

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) := \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, \alpha_t\right)\right), \Sigma_t\right), \tag{4}$$

where Σ_t is the co-variance matrix defined for the reverse process. Ho et al. (2020) showed that setting $\Sigma_t = \frac{1-\alpha_{t-1}^2}{1-\alpha_t^2}\beta_t \boldsymbol{I}$ is optimal for a deterministic $\boldsymbol{x}_0$, while setting $\Sigma_t = \beta_t \boldsymbol{I}$ is optimal for a white noise $\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Nichol & Dhariwal (2021) also proposed to learn Σ_t jointly with the score network by doubling the number of output channels, i.e., $\Sigma_t := \text{diag}(\Sigma_\theta(\boldsymbol{x}_t, \alpha_t))$, which is a learnable diagonal matrix based on a predefined noise schedule $\boldsymbol{\beta}$.

Note that the calculation of the complete ELBO in Eq. (1) requires T forward passes of the score network, which would make the training computationally prohibitive for a large T. To feasibly train the score network, instead of computing the complete ELBO, Ho et al. (2020) proposed an efficient training mechanism by sampling $t \sim \text{Uniform}(\{1, ..., T\})$, $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$, $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ at each training iteration to compute the training loss:

$$\mathcal{L}_{\text{ddpm}}^{(t)}(\theta) := \left\| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\alpha_t \boldsymbol{x}_0 + \sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t, \alpha_t\right)\right\|_2^2 \tag{5}$$

which is a re-weighted form of $\mathbb{KL}\left(q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right)$. Although Ho et al. (2020) reported the re-weighting effectively worked in practice, we show that it is deficient for learning the noise schedule $\boldsymbol{\beta}$ in the ablation experiment in Appendix C.

Noise scheduling has been shown crucial (Chen et al., 2020; Kong et al., 2021; Kong & Ping, 2021) for sampling acceleration and high-fidelity generation. In DDPMs, a linear noise schedule was shared by both the forward and reverse processes (Ho et al., 2019). A good approximation of the reverse process usually requires a large T (typically $T = 1000$). This leads to an unacceptably slow sampling process. To speed up the computation, Chen et al. (Chen et al., 2020) and Kong et al. (Kong et al., 2021) used a *noise schedule for sampling* (denoted as $\hat{\boldsymbol{\beta}} \in \mathbb{R}^N$, where $N < T$) different from the *noise schedule for training* (denoted as $\boldsymbol{\beta}$). For example, a grid search (GS) algorithm was applied in (Chen et al., 2020) to select $\hat{\boldsymbol{\beta}}$. Yet, GS would become prohibitively slow for $N > 6$. How to efficiently determine an effective noise schedule for sampling remains an unsolved issue. DPMs (Sohl-Dickstein et al., 2015) and DDPMs (Ho et al., 2019) mentioned that the noise schedule could possibly be learned by reparameterization yet not investigated in their work. We present an ablation study in Sec. 5 and demonstrate that a direct reparameterization of $\boldsymbol{\beta}$ is less effective than our proposed method. In the following sections, we alternatively derive a novel noise scheduling approach from a bilateral modeling perspective that can generate comparable and even higher quality of audio samples with remarkably fewer sampling steps.

3 BILATERAL DENOISING DIFFUSION MODELS (BDDMS)

Fig. 1 illustrates the underlying idea of BDDMs. First, a forward process with pre-specified $\boldsymbol{\beta}$ generates $\boldsymbol{x}_{n+1} \sim q_{\boldsymbol{\beta}}(\boldsymbol{x}_{n+1}|\boldsymbol{x}_0)$, which is used to reversely sample $\boldsymbol{x}_n \sim q_\phi(\boldsymbol{x}_n|\boldsymbol{x}_{n+1}, \boldsymbol{x}_0)$ using a

[1]Here, $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \alpha_t)$ is conditioned on the continuous noise scale α_t, as in (Song et al., 2021b; Chen et al., 2020). Alternatively, the score network can also be conditioned on a discrete time index $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, t)$, as in (Song et al., 2021a; Ho et al., 2020). An approximate mapping of a noise schedule to a time schedule (Kong & Ping, 2021) exists, therefore we consider conditioning on noise scales as the general case.

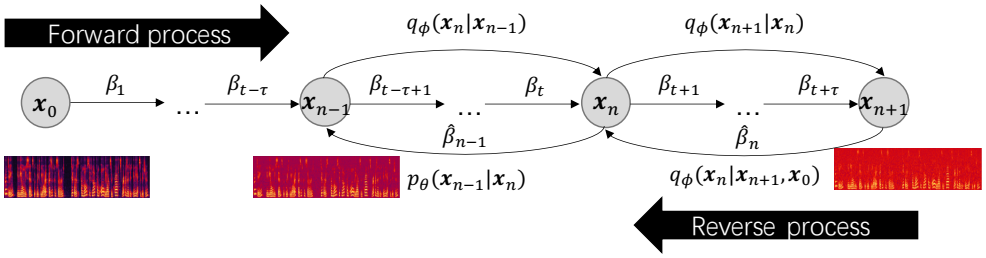

Figure 1: A block diagram of the bilateral denoising diffusion model (BDDM)

scheduling network ϕ. Then, BDDMs consider the intermediate variable $\boldsymbol{x}_n$ as the starting point for the reverse process, i.e., $p_\theta(\boldsymbol{x}_{0:n}) = p_\theta(\boldsymbol{x}_{0:n-1}|\boldsymbol{x}_n)q_\phi(\boldsymbol{x}_n|\boldsymbol{x}_{n+1}, \boldsymbol{x}_0)$. It is a new attempt to jointly parameterize both the forward process and the reverse process, which we call the *bilateral modeling*. Next, we will provide theoretical grounding for the follows: (1) the training of the score network performed at each step t, which relates to the parameterization of the reverse process, and we will show its objective in line with the training algorithm in (Ho et al., 2020), and (2) the optimization of the noise schedule for sampling, which relates to the parameterization of the forward process, and achieves a lower bound for the log marginal likelihood tighter than the conventional ELBO.

Formally, we first define the prior at step t as $\pi(\boldsymbol{x}_t)$ such that $p_\theta(\boldsymbol{x}_{0:t}) = \pi(\boldsymbol{x}_t)p_\theta(\boldsymbol{x}_{0:t-1}|\boldsymbol{x}_t)$ for $t \in \{2, ..., T\}$. Then, we derive a new lower bound to the log marginal likelihood:

Proposition 1. *Given a noise schedule $\boldsymbol{\beta}$, the following lower bound holds for $t \in \{2, ..., T\}$:*

$$\log p_\theta(\boldsymbol{x}_0) \geq \mathcal{F}^{(t)}_{score}(\theta) := -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{L}^{(t)}_{score}(\theta) + \mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)\right], \tag{6}$$

where

$$\mathcal{L}^{(t)}_{score}(\theta) := \mathbb{KL}\left(p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)||\pi(\boldsymbol{x}_{t-1})\right), \tag{7}$$

$$\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t) := -\mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\right]. \tag{8}$$

The proof is provided in Appendix A. In practice, $\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)$ can be approximated by a one-step prediction from $\boldsymbol{x}_t$ to $\boldsymbol{x}_1$ using the non-Markovian probability density function as defined in DDIMs (Song et al., 2021a). In this regard, the proposed lower bound $\mathcal{F}^{(t)}_{\text{score}}(\theta)$ allows us to consider only one t at each training step for efficient training, which is practically more advantageous than the standard ELBO in Eq. (1) that entails computing a sum of T KL terms. Below, we show that the proposed bound leads to an objective in line with $\mathcal{L}^{(t)}_{\text{ddpm}}(\theta)$ under the following conditions:

Proposition 2. *If we set $\pi(\boldsymbol{x}_{t-1}) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ for $t \in \{2, ..., T\}$, then any optimal solution satisfying $\theta^* = argmin_\theta \mathcal{L}^{(t)}_{ddpm}(\theta), \forall t \in \{1, ..., T\}$, also satisfies $\theta^* = argmax_\theta \mathcal{F}^{(t)}_{score}(\theta), \forall t \in \{2, ..., T\}$.*

The proof is provided in Appendix A. By this proposition, we see that optimizing $\mathcal{L}^{(t)}_{\text{ddpm}}(\theta)$ for training the score network θ assures maximizing the proposed lower bound $\mathcal{F}^{(t)}_{\text{score}}(\theta)$ under the given setting. In this sense, our new lower bound $\mathcal{F}^{(t)}_{\text{score}}(\theta)$ under certain setting ($\pi(\boldsymbol{x}_{t-1}) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$) can theoretically share the same optimization objective as the training algorithm used in DDPMs and DDIMs. Practically, it is also beneficial that our proposed BDDMs can re-use θ from any well-trained DDPM or DDIM.

Given that θ can be trained to maximize the log evidence with the pre-specified noise schedule $\boldsymbol{\beta}$ for training, the consequent question of interest in BDDMs is how to find a fast and good enough noise schedule $\hat{\boldsymbol{\beta}} \in \mathbb{R}^N$ for sampling given an optimized θ^*. To tackle this problem, we introduce a novel *scheduling network* $\sigma_\phi(\boldsymbol{x}_n)$. It learns to estimate the ratio between two consecutive noise scales ($\hat{\beta}_n$ and $\hat{\beta}_{n+1}$) from the current noisy observation $\boldsymbol{x}_n$ for inference in a descending order (from $n = N$ to 1). To make use of the scheduling, we first need to understand the relationship between the noise scale $\hat{\beta}_n$ at step n and the noise scale variables at step $n + 1$ (i.e. $\hat{\alpha}_{n+1}$ and $\hat{\beta}_{n+1}$). As a remark, we derive the range of $\hat{\beta}_n$ as below:

Remark 1. *Suppose the noise schedule for sampling is monotonic, i.e., $0 < \hat{\beta}_1 < \ldots < \hat{\beta}_N < 1$, then, for $1 \leq n < N$, $\hat{\beta}_n$ satisfies the following inequality:*

$$0 < \hat{\beta}_n < \min\left\{1 - \frac{\hat{\alpha}^2_{n+1}}{1 - \hat{\beta}_{n+1}}, \hat{\beta}_{n+1}\right\}. \tag{9}$$

The derivation is presented in Appendix A. The monotonic noise schedules put an upper limit for $\hat{\beta}_n$ and have been widely used in the prior arts (Song & Ermon, 2019; 2020; Song et al., 2021b). We also observed in practice that an upper limit is helpful for keeping the noise schedule short and effective for sampling. Consequently, we define the scheduling network $\sigma_\phi : \mathbb{R}^D \mapsto (0, 1)$ such that

$$\hat{\beta}_n = \min\left\{1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}, \hat{\beta}_{n+1}\right\}\sigma_\phi(\boldsymbol{x}_n), \tag{10}$$

where ϕ is a set of learnable neural network parameters. We call the inference process of the noise schedule for sampling a *noise scheduling* process. After setting a maximum number of reverse steps (denoted as N), we sequentially compute backward from N to 1 the noise scale, $\hat{\alpha}_n = \hat{\alpha}_N / \prod_{i=n+1}^{N} \sqrt{1-\hat{\beta}_i}$, using the output of scheduling network starting from a white noise $\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$.

The remaining problem is how to learn the network parameter ϕ. Note that $\mathcal{L}_{\text{score}}^{(t)}$ depends on β_t instead of $\hat{\beta}_n$, thus it cannot be applied to learn ϕ. Instead, we first build a mapping between sampling and training schedules by adding a constraint to the scheduling network:

$$q_\phi(\boldsymbol{x}_{n+1}|\boldsymbol{x}_n = \tilde{\boldsymbol{x}}) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t+\tau}|\boldsymbol{x}_t = \tilde{\boldsymbol{x}}), \tag{11}$$

where $1 \leq \tau < T$ is an positive integer, and $\tilde{\boldsymbol{x}}$ is a diffused variable. This constraint states that one step (indexed via n) of diffusion using $\hat{\beta}_n$ estimated by σ_ϕ maps to τ steps (indexed via t) of diffusion using $\beta_t, ..., \beta_{t+\tau}$. Then, we deduce the following proposition for learning ϕ:

Proposition 3. *Assuming θ has been optimized and hypothetically converged to the optimal θ^*, where by optimal it means with θ^* we have $p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0)$ as derived in Proposition 1 and 2, we can then define a new objective:*

$$\mathcal{L}_{step}^{(i)}(\phi; \theta^*) := \mathbb{KL}\left(p_{\theta^*}(\boldsymbol{x}_{i-1}|\boldsymbol{x}_i)||q_\phi(\boldsymbol{x}_{i-1}|\boldsymbol{x}_0)\right) \tag{12}$$

to represent the gap between $\log p_{\theta^*}(\boldsymbol{x}_0)$ and $\mathcal{F}_{score}^{(t)}(\theta^*)$ by ϕ:

$$\log p_{\theta^*}(\boldsymbol{x}_0) - \mathcal{F}_{score}^{(t)}(\theta^*) = \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\sum_{i=2}^{t} \mathcal{L}_{step}^{(i)}(\phi; \theta^*)\right]. \tag{13}$$

The proof is shown in Appendix A. From this proposition, we can observe that minimizing ϕ over $\mathcal{L}_{\text{step}}^{(i)}(\phi; \theta^*)$ leads to minimizing the gap between $\log p_{\theta^*}(\boldsymbol{x}_0)$ and $\mathcal{F}_{\text{score}}^{(t)}(\theta^*)$. This newly proposed objective has practical merits for a prompt learning of the scheduling network. Besides, the above proposition is also theoretically appealing. Followed by this proposition, we propose a new lower bound $\mathcal{F}_{\text{bddm}}^{(t)}(\theta, \phi)$ that possesses nice theoretical properties:

Theorem 1. *A new lower bound*

$$\mathcal{F}_{bddm}^{(t)}(\theta, \phi) := \begin{cases} \mathcal{F}_{score}^{(t)}(\theta) & \text{if} \quad \theta \neq \theta^* \\ \mathcal{F}_{score}^{(t)}(\theta) + \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{L}_{step}^{(t)}(\phi; \theta)\right] & \text{if} \quad \theta = \theta^* \end{cases} \tag{14}$$

is a tighter lower bound to the log marginal likelihood than the standard ELBO:

$$\log p_\theta(\boldsymbol{x}_0) \geq \mathcal{F}_{bddm}^{(t)}(\theta, \phi) \geq \mathcal{F}_{elbo}^{(t)}(\theta), \tag{15}$$

where $\mathcal{F}_{elbo}^{(t)}(\theta)$ indicates the standard ELBO evaluated at step $t \in \{2, ..., T\}$:

$$\mathcal{F}_{elbo}^{(t)}(\theta) := -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{KL}\left(q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right) + \mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)\right]. \tag{16}$$

The proof can be found in Appendix A. Note that using $\mathcal{F}_{\text{elbo}}^{(t)}(\theta)$ to train the score network by randomly sampling the diffusion step t at each training step has been proven an effective and efficient training method for score-based DPMs, including DDPM (Ho et al., 2020), DDIM (Song et al., 2021a) and Improved DDPM (Nichol & Dhariwal, 2021). Theorem 1 suggests that using $\mathcal{F}_{\text{bddm}}^{(t)}(\theta^*, \phi)$ could be a better training method than $\mathcal{F}_{\text{elbo}}^{(t)}(\theta^*)$ towards maximizing the log evidence by additionally learning ϕ with $\mathcal{L}_{\text{step}}^{(t)}(\phi; \theta^*)$. Based on these propositions, we devised the training and inference algorithms for BDDMs as discussed in the next section from a practical perspective.

Algorithm 1 Training Score Network (θ)

Given $T, \{\beta_t\}_{t=1}^T$
$\{\alpha_t\}_{t=1}^T \leftarrow \{\prod_{i=1}^t \sqrt{1-\beta_t}\}_{t=1}^T$
repeat
  $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$
  $t \sim \text{Uniform}(\{1, \ldots, T\})$
  $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{x}_t \leftarrow \hat{\alpha}_t \boldsymbol{x}_0 + \sqrt{1-\hat{\alpha}_t^2}\boldsymbol{\epsilon}_t$
  $\mathcal{L}_{\text{ddpm}}^{(t)} \leftarrow \|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \alpha_t)\|_2^2$
  Take a gradient descent step on $\nabla_\theta \mathcal{L}_{\text{ddpm}}^{(t)}$
until converged

Algorithm 2 Training Scheduling Network (ϕ)

Given $\theta^*, \tau, T, \{\alpha_t, \beta_t\}_{t=1}^T$
repeat
  $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x}_0)$
  $t \sim \text{Uniform}(\{2, \ldots, T-\tau\})$
  $\hat{\alpha}_n \leftarrow \alpha_t,\ \hat{\beta}_{n+1} \leftarrow 1 - (\alpha_{t+\tau}/\alpha_t)^2$
  $\delta_n \leftarrow \sqrt{1-\hat{\alpha}_n^2}$
  $\boldsymbol{\epsilon}_n \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
  $\boldsymbol{x}_n \leftarrow \hat{\alpha}_n \boldsymbol{x}_0 + \delta_n \boldsymbol{\epsilon}_n$
  $\boldsymbol{\epsilon}_{\theta^*}^{(n)} \leftarrow \boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_n, \hat{\alpha}_n)$
  $\hat{\beta}_n \leftarrow \min\{\delta_n^2, \hat{\beta}_{n+1}\}\sigma_\phi(\boldsymbol{x}_n)$
  $C_n \leftarrow 4^{-1}\log(\delta_n^2/\hat{\beta}_n) + 2^{-1}D\left(\hat{\beta}_n/\delta_n^2 - 1\right)$
  $\mathcal{L}_{\text{step}}^{(n)} \leftarrow \frac{1}{2(\delta_n^2 - \hat{\beta}_n)}\left\|\delta_n \boldsymbol{\epsilon}_n - \frac{\hat{\beta}_n}{\delta_n}\boldsymbol{\epsilon}_{\theta^*}^{(n)}\right\|_2^2 + C_n$
  Take a gradient descent step on $\nabla_\phi \mathcal{L}_{\text{step}}^{(n)}$
until converged

Algorithm 3 Noise Scheduling

Given $\hat{\alpha}_N, \hat{\beta}_N$
$\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
for $n \leftarrow N$ to 2 **do**
  $\boldsymbol{x}_{n-1} \sim p_\theta(\boldsymbol{x}_{n-1}|\boldsymbol{x}_n; \hat{\alpha}_n, \hat{\beta}_n)$
  $\hat{\alpha}_{n-1} \leftarrow \frac{\hat{\alpha}_n}{\sqrt{1-\hat{\beta}_n}}$
  $\hat{\beta}_{n-1} \leftarrow \min\{1-\hat{\alpha}_{n-1}^2, \hat{\beta}_n\}\sigma_\phi(\boldsymbol{x}_{n-1})$
  if $\hat{\beta}_{n-1} < \beta_1$ **then**
    return $\hat{\beta}_n, \ldots, \hat{\beta}_N$
  end if
end for
return $\hat{\beta}_1, \ldots, \hat{\beta}_N$

Algorithm 4 Sampling

Given $\{\hat{\beta}_n\}_{n=1}^N, \boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$
$\{\alpha_n\}_{n=1}^N \leftarrow \left\{\prod_{i=1}^n \sqrt{1-\hat{\beta}_n}\right\}_{n=1}^N$
for $n \leftarrow N$ to 1 **do**
  $\boldsymbol{x}_{n-1} \sim p_\theta(\boldsymbol{x}_{n-1}|\boldsymbol{x}_n; \alpha_n, \hat{\beta}_n)$
end for
return $\boldsymbol{x}_0$

4 Training, noise scheduling, and inference

In BDDMs, we train two neural networks: i) a score network $\boldsymbol{\epsilon}_\theta$ for sampling, and ii) a scheduling network σ_ϕ for estimating a noise schedule for sampling. In this section, we first present the training objectives for learning θ and ϕ, respectively, based on the derived propositions.

4.1 Training objectives

By Theorem 1, we show that $\mathcal{F}_{\text{bddm}}^{(t)}$ contains two terms: $\mathcal{F}_{\text{score}}^{(t)}(\theta)$ and $\mathcal{L}_{\text{step}}^{(t)}(\phi; \theta)$. By Proposition 2, optimizing $\mathcal{F}_{\text{score}}^{(t)}(\theta)$ w.r.t θ can be simplified into minimizing the loss $\mathcal{L}_{\text{ddpm}}^{(t)}(\theta)$ for all $t \in \{1, ..., T\}$, which has been successfully employed in the prior works (Ho et al., 2020). Then, by Proposition 3, we show that ϕ should be trained after θ is well-optimized. In a realistic setting, we can assume the optimized θ^* to be closed to the hypothetically optimal values. Given the well-optimized parameters θ^*, we then optimize w.r.t ϕ the proposed loss $\mathcal{L}_{\text{step}}^{(t)}(\phi; \theta^*)$ that can be derived as a weighted ℓ_2 norm:

Remark 2.

$$\mathcal{L}_{step}^{(t)}(\phi; \theta^*) = \frac{1}{2(1-\hat{\beta}_t(\phi) - \alpha_t^2)}\left\|\sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t - \frac{\hat{\beta}_t(\phi)}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t, \alpha_t)\right\|_2^2 + C_t(\phi), \tag{17}$$

where

$$C_t(\phi) := \frac{1}{4}\log\frac{1-\alpha_t^2}{\hat{\beta}_t(\phi)} + \frac{D}{2}\left(\frac{\hat{\beta}_t(\phi)}{1-\alpha_t^2} - 1\right). \tag{18}$$

The detailed derivation for the simplification procedure can be found in Appendix A.

4.2 Efficient training

Suggested by Proposition 3, we know that θ should be optimized before learning ϕ. For training the score network $\boldsymbol{\epsilon}_\theta$, similar to the prior settings in (Ho et al., 2020; Chen et al., 2020; Song et al., 2021a), we define $\boldsymbol{\beta}$ as a linear noise schedule:

$$\beta_t = \frac{\varepsilon}{T-t+1}, \quad \text{for} \quad 1 \le t \le T, \tag{19}$$

where ε is a hyperparameter that specifies the maximum value in this linear schedule, i.e., β_T. By Proposition 2, we can sample $t \sim \text{Uniform}(\{1, ..., T\})$ for evaluating $\mathcal{L}^{(t)}_{\text{ddpm}}$, resulting in Alg. 1 that is similar to (Ho et al., 2020).

Next, based on the learned θ, we can start training the scheduling network σ_ϕ. For training σ_ϕ in Eq. (10), we first sample the noise scale from the preceding step, i.e., $\hat{\beta}_{n+1}$:

$$\hat{\beta}_{n+1} \sim \text{Uniform}\left(\left\{1 - (\alpha_{t+\tau}/\alpha_t)^2 \,|\, t \in \{2, ..., T-\tau\}\right\}\right), \tag{20}$$

where $0 < \tau < T$ is a hyperparameter defined in Eq. (11) to control the inference step size relative to the noise schedule for training $\boldsymbol{\beta}$. Expectedly, a larger τ should lead to a shorter noise schedule $\hat{\boldsymbol{\beta}}$. Note that Theorem 1 still holds when we change the index from t to n. Followed from this sampling strategy and Eq. (17) and (18), we can efficiently train the scheduling network, as presented in Alg. 2. Our empirical observation in section 5 shows that, although a linear schedule is used to define $\boldsymbol{\beta}$, the predicted schedule for sampling $\hat{\boldsymbol{\beta}}$ is not limited into a linear one.

4.3 Noise scheduling for fast and high-quality sampling

After the score network and the scheduling network are trained, the inference procedure can divide into two phases: (1) the noise scheduling phase and (2) the sampling phase.

First, we run the noise scheduling process as a reverse process with N iterations maximum. Different from training, where α_t was started from $\sqrt{1-\beta_1}$ and forward-computed, $\hat{\alpha}_n$ is instead a backward-computed variable (from N to 1) that may deviates from the forward one because $\{\hat{\beta}_i\}_i^{n-1}$ are unknown in the noise scheduling phase during inference. Therefore, to start noise scheduling, we set two hyperparameters, $(\hat{\alpha}_N, \hat{\beta}_N)$. We use β_1, the smallest predetermined noise scale in training, as a threshold to determine an early stop of the noise scheduling process, so that we can ignore too small noise scales ($< \beta_1$) that were never seen by the score network to avoid numerical issues. In practice, we apply a grid search algorithm of M bins, which takes $\mathcal{O}(M^2)$ time, to find proper values for setting $(\hat{\alpha}_N, \hat{\beta}_N)$. This is computationally feasible in contrast to the unscalable $\mathcal{O}(M^N)$ grid search algorithm used in (Chen et al., 2020) that needs to search over the entire noise schedule $\hat{\boldsymbol{\beta}}$. Furthermore, the grid search for our noise scheduling algorithm can be evaluated on a very small subset of the training samples. Empirically, even as few as 1 sample for evaluation work well for our algorithm. Overall, the noise scheduling process is presented in Alg. 3.

Next, we use the resultant noise schedule, $\hat{\boldsymbol{\beta}} \in \mathbb{R}^N$, for the subsequent sampling over the whole testing set. By initializing the white noise $\boldsymbol{x}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, we can generate $\boldsymbol{x}_0$ after N iterations of the reverse process, as shown in Alg. 4.

5 Experiments

We conducted a series of experiments on neural vocoding tasks to evaluate the proposed BDDMs. First, we compared BDDMs against several strongest models that have been published: the mixture of logistics (MoL) WaveNet (Oord et al., 2018) implemented in (Yamamoto, 2020), the WaveGlow (Prenger et al., 2019) implemented in (Valle, 2020), the MelGAN (Kumar et al., 2019a) implemented in (Kumar, 2019), the HiFi-GAN (Kong et al., 2020b) implemented in (Kong et al., 2020a) and the two most recently proposed diffusion-based vocoders, i.e., WaveGrad (Chen et al., 2020) and DiffWave (Kong et al., 2021), both re-implemented in our code. The hyperparameter settings of BDDMs and all these models are detailed in Appendix B.

In addition, we also compared BDDMs to a variety of scheduling and acceleration techniques applicable to DDPMs, including the grid search (GS) approach in WaveGrad, the fast sampling (FS) approach based on a user-defined 6-step schedule in DiffWave, the DDIMs (Song et al., 2021a) and a noise estimation (NE) approach (San-Roman et al., 2021). Detailed implementation can be refer to Appendix B.

For fair and reproducible comparison with other models and approaches, we used the LJSpeech dataset (Ito & Johnson, 2017), which consists of 13,100 22kHz audio clips of a female speaker. All diffusion models were trained on the same training split as in (Chen et al., 2020). We also replicated

Table 1: Comparison of neural vocoders in terms of MOS with 95% confidence intervals, real-time factor (RTF) and model size in megabytes (MB) for inference. The MOSs showing no significant difference from the ground truth under student's t-tests (p-values ≥ 0.05) are bold-faced.

Neural Vocoder	**MOS** (p-value)	**RTF**	**Size**
Ground-truth	4.62 ± 0.24	—	—
WaveNet (MoL) (Oord et al., 2018)	3.63 ± 0.26 (1.05×10^{-9})	318.6	282MB
WaveGlow (Prenger et al., 2019)	3.06 ± 0.37 (1.10×10^{-10})	0.0198	645MB
MelGAN (Kumar et al., 2019a)	3.52 ± 0.35 (1.08×10^{-8})	0.00396	17MB
HiFi-GAN (Kong et al., 2020b)	4.32 ± 0.25 (3.25×10^{-3})	0.0134	54MB
WaveGrad - 1000 steps (Chen et al., 2020)	4.39 ± 0.23 (0.012)	38.2	183MB
DiffWave - 200 steps (Kong et al., 2021)	$\mathbf{4.52 \pm 0.24}$ (0.161)	7.30	27MB
BDDM - 3 steps ($\hat{\alpha}_N = 0.68$, $\hat{\beta}_N = 0.53$)	3.65 ± 0.23 (4.58×10^{-10})	0.110	27MB
BDDM - 7 steps ($\hat{\alpha}_N = 0.62$, $\hat{\beta}_N = 0.42$)	$\mathbf{4.43 \pm 0.31}$ (0.055)	0.256	27MB
BDDM - 12 steps ($\hat{\alpha}_N = 0.67$, $\hat{\beta}_N = 0.12$)	$\mathbf{4.47 \pm 0.33}$ (0.102)	0.438	27MB

Table 2: Comparison of sampling acceleration methods with the same score network and the same number of steps. The highest score and the scores that are not significantly different from the highest score in student's t-tests (p-values ≥ 0.05) are bold-faced.

Steps	**Acceleration Method**	**STOI**	**PESQ**	**MOS**
3	GS (Chen et al., 2020)	$\mathbf{0.965 \pm 0.009}$	$\mathbf{3.66 \pm 0.20}$	$\mathbf{3.63 \pm 0.25}$
	FS (Kong et al., 2021)	0.939 ± 0.023	3.09 ± 0.23	3.13 ± 0.29
	DDIM (Song et al., 2021a)	0.943 ± 0.015	3.42 ± 0.27	3.28 ± 0.33
	NE (San-Roman et al., 2021)	$\mathbf{0.966 \pm 0.010}$	$\mathbf{3.62 \pm 0.18}$	$\mathbf{3.56 \pm 0.30}$
	BDDM	$\mathbf{0.966 \pm 0.011}$	$\mathbf{3.63 \pm 0.24}$	$\mathbf{3.65 \pm 0.23}$
7	FS (Kong et al., 2021)	$\mathbf{0.981 \pm 0.006}$	3.68 ± 0.24	3.70 ± 0.35
	DDIM (Song et al., 2021a)	0.974 ± 0.008	3.85 ± 0.12	3.94 ± 0.28
	NE (San-Roman et al., 2021)	0.978 ± 0.007	3.75 ± 0.18	4.02 ± 0.26
	BDDM	$\mathbf{0.983 \pm 0.006}$	$\mathbf{3.96 \pm 0.09}$	$\mathbf{4.43 \pm 0.31}$
12	DDIM (Song et al., 2021a)	0.979 ± 0.006	3.90 ± 0.10	4.16 ± 0.27
	NE (San-Roman et al., 2021)	0.981 ± 0.007	3.82 ± 0.13	3.98 ± 0.35
	BDDM	$\mathbf{0.987 \pm 0.006}$	$\mathbf{3.98 \pm 0.12}$	$\mathbf{4.47 \pm 0.33}$

the comparative experiment of neural vocoding using a multi-speaker VCTK dataset (Yamagishi et al., 2019) and obtained a result consistent with that obtained from the LJSpeech dataset. The result of the VCTK dataset is detailed in Appendix C.

5.1 Sampling quality in objective and subjective metrics

To assess the quality of each generated audio sample, we used both objective and subjective measures for comparing different neural vocoders given the same ground-truth spectrogram $\boldsymbol{s}$ as the condition, i.e., $\boldsymbol{\epsilon}_\theta(\boldsymbol{x}, \boldsymbol{s}, \alpha_t)$. Specifically, we used two scale-invariant metrics: the perceptual evaluation of speech quality (PESQ) (Rix et al., 2001) and the short-time objective intelligibility (STOI) (Taal et al., 2010) to measure the noisiness and the distortion of the generated speech relative to the reference speech. Mean opinion score (MOS) was also used as a subjective metric for evaluating the naturalness of the generated speech. The assessment scheme of MOS is included in Appendix B.

In Table 1, we compared BDDMs against the state-of-the-art (SOTA) vocoders. To predict noise schedules with different sampling steps (3, 7, and 12), we set three pairs of $\{\hat{\alpha}_N, \hat{\beta}_N\}$ for BDDMs by running on Alg. 3 a quick hyperparameter grid search, which is detailed in Appendix B. Among the 9 evaluated vocoders, only our proposed BDDMs with 7 and 12 steps and DiffWave with 200 steps showed no statistic-significant difference from the ground-truth in terms of MOS. Moreover, BDDMs significantly outspeeded DiffWave in terms of RTFs. Notably, previous diffusion-based vocoders achieved high MOS scores at the cost of an unacceptable RTF for industrial deployment. In contrast, BDDMs managed to achieve a high standard of generation quality with only 7 sampling steps (143x faster than WaveGrad and 28.6x faster than DiffWave).

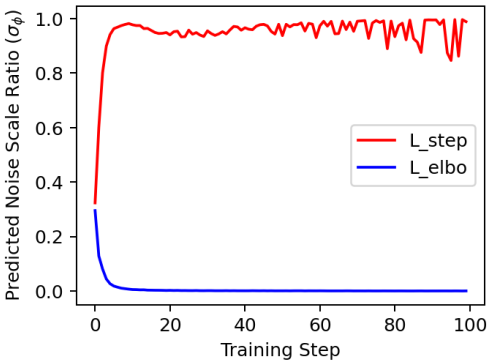

Figure 2: Different training losses for σ_ϕ

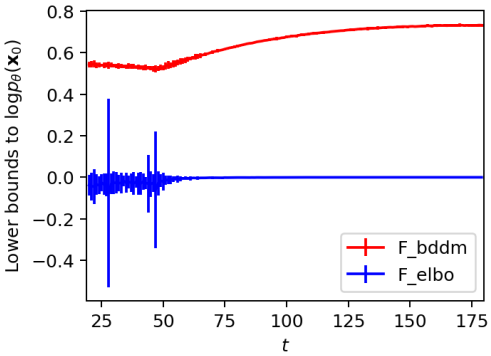

Figure 3: Different lower bounds of $\log p_\theta(\boldsymbol{x}_0)$

In Table 2, we evaluated BDDMs and alternative SOTA accelerated sampling methods, which used the same score network for a pair-to-pair comparison. The GS method performed stably when the step number was small (i.e., $N \leq 6$) but unfortunately not scalable towards more step numbers, which were therefore bypassed in further evaluation. The FS method by Song et al. (2021a) used a predefined 6-step noise schedule[2], which was linearly interpolated to 3 and 7 steps in our evaluation. Comparing its 7-step and 3-step results, we observed that the FS performance degraded drastically. Both the DDIM and the NE methods were stable across all the steps but were not performing competitively enough. In comparison, BDDMs consistently attained the leading scores across all the steps. This evaluation confirmed that BDDM was superior to other acceleration methods for DPMs in terms of both stability and quality.

5.2 Ablation Study and Analysis

We attribute the primary advantage of BDDMs to the newly derived objective $\mathcal{L}_{\text{step}}^{(t)}$ for learning ϕ. To better reason about this, we performed an ablation study, where we substituted the proposed loss with the standard negative ELBO for learning ϕ as mentioned by Sohl-Dickstein et al. (2015). We plotted the network outputs with different training losses in Fig. 2. It turned out that, when using $\mathcal{L}_{\text{elbo}}^{(t)}$ to learn ϕ, the network output rapidly collapsed to zero within several training steps; whereas, the network trained with $\mathcal{L}_{\text{step}}^{(t)}$ produced fluctuating outputs. The fluctuation is a desirable property showing the network properly predicts t-dependent noise scales, as t is a random time step drawn from a uniform distribution in training.

As illustrated in Fig. 3, we empirically validated that $\mathcal{F}_{\text{bddm}}^{(t)} \geq \mathcal{F}_{\text{elbo}}^{(t)}$ with their respective values at $t \in [20, 180]$ using the same optimized theta θ^*, and each value with 95% confidence intervals. In this experiment, we used the LJ speech dataset and set $T = 200$ and $\tau = 20$. Notably, we dropped their common entropy term $\mathbb{E}_{q_\phi(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)\right] < 0$ to mainly compare their KL terms. This explains those positive lower bound values in the plot. The graph shows that our proposed bound $\mathcal{F}_{\text{bddm}}^{(t)}$ is always a tighter lower bound than the standard one across all examined t. Moreover, we found that $\mathcal{F}_{\text{bddm}}^{(t)}$ attained low values with a relatively much lower variance around $t \in [20, 50]$, whereas $\mathcal{F}_{\text{elbo}}^{(t)}$ became highly volatile. This implies $\mathcal{F}_{\text{bddm}}^{(t)}$ better tackles the difficult training part, when t is approaching zero such that the score becomes more challenging to estimate.

6 Conclusions

We introduced the bilateral denoising diffusion models (BDDMs), which parameterizes the forward and reverse processes with a scheduling network and a score network. From a bilateral modeling perspective, we derived a new lower bound that leads to the same training loss as in DDPMs (Ho et al., 2020), which thus enables inheriting any pre-trained score networks in DDPMs. We also showed that training the scheduling network after a well-optimized score network can be viewed as tightening the lower bound. Followed from the theoretical results, an efficient training algorithm and a noise scheduling algorithm were respectively designed for BDDMs. Finally, in our experiments, BDDMs showed a clear edge over the previous diffusion-based vocoders.

[2]The authors of FS did not unfold their method on how to define the 6-step noise schedule.

REFERENCES

Mikołaj Bińkowski, Jeff Donahue, Sander Dieleman, Aidan Clark, Erich Elsen, Norman Casagrande, Luis C. Cobo, and Karen Simonyan. High fidelity speech synthesis with adversarial networks. *ICLR*, 2020.

Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *https://arxiv.org/abs/2103.04922*, 2021.

Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. *AISTATS*, pp. 33–40, 2005.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J. Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *NeurIPs*, pp. 6571–6583, 2018.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis. *arXiv preprint arXiv:2105.05233*, 2021.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv:1406.2661*, 2014.

G. E. Hinton. Training products of experts by minimizing contrastive divergence. neural computation. *Neural computation*, pp. 14(8):1771–1800, 2002.

Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design. *ICML*, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020.

Aapo Hyvarinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, pp. 6(4), 2005.

Keith Ito and Linda Johnson. The lj speech dataset. *https://keithito.com/LJ-Speech-Dataset/*, 2017.

Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *International Conference on Machine Learning*, pp. 2410–2419. PMLR, 2018.

Diederik P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *NeurIPS*, pp. 10215–10224, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. `https://github.com/jik876/hifi-gan`, 2020a.

Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33, 2020b.

Zhifeng Kong and Wei Ping. On fast sampling of diffusion probabilistic models. *arXiv preprint arXiv:2106.00132*, 2021.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *ICLR*, 2021.

Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron Courville. Melgan: Generative adversarial networks for conditional waveform synthesis. *arXiv preprint arXiv:1910.06711*, 2019a.

Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brebisson, Yoshua Bengio, and Aaron Courville. Melgan: Generative adversarial networks for conditional waveform synthesis. *NeurIPS*, 2019b.

Rithesh Kumar. Official repository for the paper melgan: Generative adversarial networks for conditional waveform synthesis. `https://github.com/descriptinc/melgan-neurips`, 2019.

Max WY Lam, Jun Wang, Dan Su, and Dong Yu. Effective low-cost time-domain audio separation using globally attentive locally recurrent networks. *arXiv preprint arXiv:2101.05014*, 2021.

Eric Luhman and Troy Luhman. Knowledge distillation in iterative generative models for improved sampling speed. *arXiv:2101.02388*, 2021.

Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. Biva: A very deep hierarchy of latent variables for generative modeling. *NeurIPS*, pp. 6548–6558, 2019.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021.

Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pp. 3918–3926. PMLR, 2018.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *JMLR*, pp. 22(57):1–64, 2021.

Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621. IEEE, 2019.

Flavio Protasio Ribeiro, Dinei Florencio, Cha Zhang, and Mike Seltzer. CROWDMOS: An approach for crowdsourcing mean opinion score studies. In *ICASSP*. IEEE, 2011. Edition: ICASSP.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, pp. 1278–1286, 2014.

Antony W Rix, John G Beerends, Michael P Hollier, and Andries P Hekstra. Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)*, volume 2, pp. 749–752. IEEE, 2001.

Robin San-Roman, Eliya Nachmani, and Lior Wolf. Noise estimation for generative diffusion models. *arXiv preprint arXiv:2104.02600*, 2021.

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

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021a.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 2019.

Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *arXiv preprint arXiv:2006.09011*, 2020.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *ICLR*, 2021b.

Cees H Taal, Richard C Hendriks, Richard Heusdens, and Jesper Jensen. A short-time objective intelligibility measure for time-frequency weighted noisy speech. In *2010 IEEE international conference on acoustics, speech and signal processing*, pp. 4214–4217. IEEE, 2010.

Rafael Valle. Waveglow: a flow-based generative network for speech synthesis. `https://github.com/NVIDIA/waveglow`, 2020.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *Proc. 9th ISCA Speech Synthesis Workshop*, pp. 125–125, 2016.

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

Junichi Yamagishi, Christophe Veaux, and Kirsten MacDonald. CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92), 2019.

Ryuichi Yamamoto. Wavenet vocoder. `https://github.com/r9y9/wavenet_vocoder`, 2020.

Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel. Wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. *ICASSP*, 2020.

Geng Yang, Shan Yang, Kai Liu, Peng Fang, Wei Chen, and Lei Xie. Multi-band melgan: Faster waveform generation for high-quality text-to-speech. *arXiv preprint arXiv:2005.0510*, 2020.

A MATHEMATICAL PROOFS FOR DERIVING BDDMS

Proposition 1. *Given a noise schedule $\boldsymbol{\beta}$, the following lower bound holds for $t \in \{2, ..., T\}$:*

$$\log p_\theta(\boldsymbol{x}_0) \geq \mathcal{F}^{(t)}_{score}(\theta) := -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{L}^{(t)}_{score}(\theta) + \mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)\right], \tag{21}$$

where

$$\mathcal{L}^{(t)}_{score}(\theta) := \mathbb{KL}\left(p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)||\pi(\boldsymbol{x}_{t-1})\right), \tag{22}$$

$$\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t) := -\mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\right]. \tag{23}$$

Proof.

$$\log p_\theta(\boldsymbol{x}_0) = \log \int p_\theta(\boldsymbol{x}_{0:t-1}) d\boldsymbol{x}_{1:t-1} \tag{24}$$

$$= \log \int p_\theta(\boldsymbol{x}_{0:t-1}) \left(\int \frac{p_\theta(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}{p_\theta(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)} q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0) d\boldsymbol{x}_t\right) d\boldsymbol{x}_{1:t-1} \tag{25}$$

$$= \log \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)} \mathbb{E}_{p_\theta(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\left[\frac{p_\theta(\boldsymbol{x}_{0:t-1})}{p_\theta(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\right] \tag{26}$$

$$= \log \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)} \mathbb{E}_{p_\theta(\boldsymbol{x}_{1,t-1}|\boldsymbol{x}_t)}\left[\frac{p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\pi(\boldsymbol{x}_{t-1})}{p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}\right] \tag{27}$$

$$\text{[Jensen's Inequality]} \quad \geq \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)} \mathbb{E}_{p_\theta(\boldsymbol{x}_1, \boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}\left[\log \frac{p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\pi(\boldsymbol{x}_{t-1})}{p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}\right] \tag{28}$$

$$= \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\right] - \mathbb{KL}\left(p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)||\pi(\boldsymbol{x}_{t-1})\right)\right] \tag{29}$$

$$= -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t) + \mathcal{L}^{(t)}_{\text{score}}(\theta)\right] \tag{30}$$

□

Proposition 2. *If we set $\pi(\boldsymbol{x}_{t-1}) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ for $t \in \{2, ..., T\}$, then any optimal solution satisfying $\theta^* = argmin_\theta \mathcal{L}^{(t)}_{ddpm}(\theta)\, \forall t \in \{1, ..., T\}$, also satisfies $\theta^* = argmax_\theta \mathcal{F}^{(t)}_{score}(\theta)\, \forall t \in \{2, ..., T\}$.*

Proof. By definition, we have

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, \alpha_t\right)\right), \frac{(1-\alpha_{t-1})}{(1-\alpha_t)}\beta_t \boldsymbol{I}\right), \tag{31}$$

$$q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) = \mathcal{N}\left(\frac{\alpha_{t-1}\beta_t}{1-\alpha_t^2}\boldsymbol{x}_0 + \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{1-\alpha_t^2}\boldsymbol{x}_t, \frac{(1-\alpha_{t-1})}{(1-\alpha_t)}\beta_t \boldsymbol{I}\right) \tag{32}$$

$$= \mathcal{N}\left(\frac{\alpha_{t-1}\beta_t}{1-\alpha_t^2}\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t}{\alpha_t} + \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{1-\alpha_t^2}\boldsymbol{x}_t, \sigma_t^2 \boldsymbol{I}\right) \tag{33}$$

$$= \mathcal{N}\left(\left(\frac{\alpha_{t-1}\beta_t}{\alpha_t(1-\alpha_t^2)} + \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{1-\alpha_t^2}\right)\boldsymbol{x}_t - \frac{\alpha_{t-1}\beta_t}{\alpha_t\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_t, \sigma_t^2 \boldsymbol{I}\right) \tag{34}$$

$$= \mathcal{N}\left(\frac{\alpha_{t-1}\beta_t + \alpha_t\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{\alpha_t(1-\alpha_t^2)}\boldsymbol{x}_t - \frac{\alpha_{t-1}\beta_t}{\alpha_t\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_t, \sigma_t^2 \boldsymbol{I}\right). \tag{35}$$

Given that $\pi(\boldsymbol{x}_{t-1}) := q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$, we now can simplify $\mathcal{L}^{(t)}_{\text{score}}(\theta)$ and $\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)$ as follows.

Firstly, to express $\mathcal{L}_{\text{score}}^{(t)}(\theta)$, we know that $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ and $q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ are two isotropic Gaussians with the same variance, thus the KL becomes a scaled squared error between their means:

$$
\begin{aligned}
\mathcal{L}_{\text{score}}^{(t)}(\theta) :=& D_{\text{KL}}\left(p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)||q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)\right) && (36)\\
=& \frac{(1-\beta_t)(1-\alpha_t^2)}{2(1-\beta_t-\alpha_t^2)\beta_t}\left\|\frac{\alpha_{t-1}\beta_t+\alpha_t\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{\alpha_t(1-\alpha_t^2)}\boldsymbol{x}_t-\frac{\alpha_{t-1}\beta_t}{\alpha_t\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_t\right. && (37)\\
& \left.-\frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t-\frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)\right)\right\|_2^2 && (38)\\
=& \frac{(1-\beta_t)(1-\alpha_t^2)}{2(1-\beta_t-\alpha_t^2)\beta_t}\left\|\left(\frac{\alpha_{t-1}\beta_t+\alpha_t\sqrt{1-\beta_t}(1-\alpha_{t-1}^2)}{\alpha_t(1-\alpha_t^2)}-\frac{1}{\sqrt{1-\beta_t}}\right)\boldsymbol{x}_t\right. && (39)\\
& \left.-\frac{\alpha_{t-1}\beta_t}{\alpha_t\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_t+\frac{\beta_t\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)}{\sqrt{(1-\alpha_t^2)(1-\beta_t)}}\right\|_2^2, && (40)\\
=& \frac{(1-\beta_t)(1-\alpha_t^2)}{2(1-\beta_t-\alpha_t^2)\beta_t}\left\|0\cdot\boldsymbol{x}_t-\frac{\beta_t\boldsymbol{\epsilon}_t}{\sqrt{(1-\alpha_t^2)(1-\beta_t)}}+\frac{\beta_t\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)}{\sqrt{(1-\alpha_t^2)(1-\beta_t)}}\right\|_2^2 && (41)\\
=& \frac{(1-\beta_t)(1-\alpha_t^2)}{2(1-\beta_t-\alpha_t^2)\beta_t}\frac{\beta_t^2}{(1-\alpha_t^2)(1-\beta_t)}\left\|\boldsymbol{\epsilon}_t-\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)\right\|_2^2 && (42)\\
=& \frac{\beta_t}{2(1-\beta_t-\alpha_t^2)}\left\|\boldsymbol{\epsilon}_t-\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)\right\|_2^2, && (43)
\end{aligned}
$$

which is proportional to $\mathcal{L}_{\text{ddpm}}^{(t)} := \|\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t, \alpha_t\right)\|_2^2$.

Next, we show that $\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t)$ can be expressed in terms of $\mathcal{L}_{\text{ddpm}}^{(t)}$ evaluated at $t = 1$:

$$
\begin{aligned}
\mathcal{R}_\theta(\boldsymbol{x}_0, \boldsymbol{x}_t) :=& -\mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)\right] && (44)\\
=& \mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\log\mathcal{N}\left(\frac{1}{\sqrt{1-\beta_1}}\left(\boldsymbol{x}_1-\frac{\beta_1}{\sqrt{1-\alpha_1^2}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_1,\alpha_1),\beta_1\mathbf{I}\right)\right)\right] && (45)\\
=& \mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\frac{D}{2}\log 2\pi\beta_1+\frac{1}{2\beta_1}\left\|\boldsymbol{x}_0-\frac{1}{\sqrt{1-\beta_1}}\left(\boldsymbol{x}_1-\frac{\beta_1}{\sqrt{\beta_1}}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_1,\alpha_1)\right)\right\|_2^2\right] && (46)\\
=& \frac{D}{2}\log 2\pi\beta_1+\frac{1}{2\beta_1}\mathbb{E}_{p_\theta(\boldsymbol{x}_1|\boldsymbol{x}_t)}\left[\left\|\frac{\boldsymbol{x}_1-\sqrt{\beta_1}\boldsymbol{\epsilon}_1}{\sqrt{1-\beta_1}}-\frac{\boldsymbol{x}_1-\sqrt{\beta_1}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_1,\alpha_1)}{\sqrt{1-\beta_1}}\right\|_2^2\right] && (47)\\
=& \frac{1}{2(1-\beta_1)}\|\boldsymbol{\epsilon}_1-\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_1,\alpha_1\right)\|_2^2+\frac{D}{2}\log 2\pi\beta_1. && (48)
\end{aligned}
$$

Let $\theta^* = \text{argmin}_\theta \mathcal{L}_{\text{ddpm}}^{(t)}(\theta)$ such that $\boldsymbol{\epsilon}_{\theta^*}\left(\boldsymbol{x}_t, \alpha_t\right) = \boldsymbol{\epsilon}_t + \Delta\boldsymbol{\epsilon}_t$ for all $t \in \{1, ..., T\}$, where $\Delta\boldsymbol{\epsilon}_t$ denotes the minimum error vector that can be achieved by θ^* given the score network $\boldsymbol{\epsilon}_\theta$. By substituting θ^* into $\mathcal{F}_{\text{score}}^{(t)}(\theta)$ for $t \in \{2, ..., T\}$, we can see that

$$
\begin{aligned}
\mathcal{F}_{\text{score}}^{(t)}(\theta^*) :=& -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{L}_{\text{score}}^{(t)}(\theta^*)+\mathcal{R}_{\theta^*}(\boldsymbol{x}_0,\boldsymbol{x}_t)\right] && (49)\\
=& -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\frac{\beta_t}{2(1-\beta_t-\alpha_t^2)}\|\Delta\boldsymbol{\epsilon}_t\|_2^2+\frac{1}{2(1-\beta_1)}\|\Delta\boldsymbol{\epsilon}_1\|_2^2+\frac{D}{2}\log 2\pi\beta_1\right] && (50)\\
=& -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\min_\theta\mathcal{L}_{\text{score}}^{(t)}(\theta)+\min_\theta\mathcal{R}_\theta(\boldsymbol{x}_0,\boldsymbol{x}_t)\right] && (51)\\
=& \max_\theta\mathcal{F}_{\text{score}}^{(t)}(\theta) && (52)
\end{aligned}
$$

□

Remark 1. *Suppose the noise schedule for sampling is monotonic, i.e.,* $0 < \hat{\beta}_1 < \ldots < \hat{\beta}_N < 1$*, then, for* $1 \leq n < N$*,* $\hat{\beta}_n$ *satisfies the following inequality:*

$$0 < \hat{\beta}_n < \min\left\{1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}, \hat{\beta}_{n+1}\right\}. \tag{53}$$

Proof. By the definition of noise schedule, we know that $0 < \hat{\beta}_1, \ldots, \hat{\beta}_N < 1$ (Note: no inequality sign in between). Given that $\hat{\alpha}_n = \prod_{i=1}^n \sqrt{1-\hat{\beta}_i}$, we also have $0 < \hat{\alpha}_1, \ldots, \hat{\alpha}_t < 1$. First, we show that $\hat{\beta}_n < 1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}$:

$$\hat{\alpha}_{n-1} = \frac{\hat{\alpha}_n}{\sqrt{1-\hat{\beta}_n}} < 1 \Longleftrightarrow \hat{\beta}_n < 1 - \hat{\alpha}_n^2 = 1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}. \tag{54}$$

Next, we show that $\hat{\beta}_n < 1 - \hat{\alpha}_{n+1}$:

$$\frac{\hat{\alpha}_n}{\sqrt{1-\hat{\beta}_n}} = \frac{\hat{\alpha}_n\sqrt{1-\hat{\beta}_n}}{1-\hat{\beta}_n} = \frac{\hat{\alpha}_{n+1}}{1-\hat{\beta}_n} < 1 \Longleftrightarrow \hat{\beta}_n < 1 - \hat{\alpha}_{n+1}. \tag{55}$$

Now, we have $\hat{\beta}_n < \min\left\{1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}, 1 - \hat{\alpha}_{n+1}\right\}$. When $1 - \hat{\alpha}_{n+1} < 1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}$, we can show that $\hat{\beta}_{n+1} < 1 - \hat{\alpha}_{n+1}$:

$$1 - \hat{\alpha}_{n+1} < 1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}} = 1 - \hat{\alpha}_n^2 \Longleftrightarrow \hat{\alpha}_{n+1} > \hat{\alpha}_n^2 \Longleftrightarrow \frac{\hat{\alpha}_{n+1}^2}{\hat{\alpha}_n^2} > \hat{\alpha}_{n+1} \tag{56}$$

$$\Longleftrightarrow 1 - \frac{\hat{\alpha}_{n+1}^2}{\hat{\alpha}_n^2} < 1 - \hat{\alpha}_{n+1} \Longleftrightarrow \hat{\beta}_{n+1} < 1 - \hat{\alpha}_{n+1}. \tag{57}$$

By the assumption of monotonic sequence, we also have $\hat{\beta}_n < \hat{\beta}_{n+1}$. Knowing that $\hat{\beta}_{n+1} < 1 - \hat{\alpha}_{n+1}$ is always true, we obtain a tighter bound for $\hat{\beta}_n$: $0 < \hat{\beta}_n < \min\left\{1 - \frac{\hat{\alpha}_{n+1}^2}{1-\hat{\beta}_{n+1}}, \hat{\beta}_{n+1}\right\}$. □

Proposition 3. *Assuming* θ *has been optimized and hypothetically converged to the optimal* θ^**, where by optimal it means with* θ^* *we have* $p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0)$ *as derived in Proposition 1 and 2, we can then define a new objective:*

$$\mathcal{L}_{step}^{(i)}(\phi;\theta^*) := \mathbb{KL}\left(p_{\theta^*}(\boldsymbol{x}_{i-1}|\boldsymbol{x}_i)||q_\phi(\boldsymbol{x}_{i-1}|\boldsymbol{x}_0)\right) \tag{58}$$

to represent the gap between $\log p_{\theta^*}(\boldsymbol{x}_0)$ *and* $\mathcal{F}_{score}^{(t)}(\theta^*)$ *by* ϕ*:*

$$\log p_{\theta^*}(\boldsymbol{x}_0) - \mathcal{F}_{score}^{(t)}(\theta^*) = \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\sum_{i=2}^t \mathcal{L}_{step}^{(i)}(\phi;\theta^*)\right]. \tag{59}$$

Proof. We have

$$\log p_{\theta^*}(\boldsymbol{x}_0) - \mathcal{F}_{\text{score}}^{(t)}(\theta^*) = \log p_{\theta^*}(\boldsymbol{x}_0) - \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{E}_{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\left[\log\frac{p_{\theta^*}(\boldsymbol{x}_{0:t-1})}{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\right]\right] \tag{60}$$

$$= \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{E}_{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\left[\log\frac{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0)}\right]\right] \tag{61}$$

$$= \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{E}_{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}\left[\log\frac{p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)}{q_{\boldsymbol{\beta}}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0)}\right]\right] \tag{62}$$

$$= \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{KL}\left(p_{\theta^*}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_t)||q_{\boldsymbol{\beta}}(\boldsymbol{x}_{1:t-1}|\boldsymbol{x}_0)\right)\right] \tag{63}$$

$$= \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\sum_{i=2}^t \mathcal{L}_{\text{step}}^{(i)}(\phi;\theta^*)\right]. \tag{64}$$

Note that the last equality holds as we considered $q_\phi(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}) = q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. □

Theorem 1. *A new lower bound*

$$\mathcal{F}_{bddm}^{(t)}(\theta,\phi) := \begin{cases} \mathcal{F}_{score}^{(t)}(\theta) & \text{if} \quad \theta \neq \theta^* \\ \mathcal{F}_{score}^{(t)}(\theta) + \mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathcal{L}_{step}^{(t)}(\phi;\theta)\right] & \text{if} \quad \theta = \theta^* \end{cases} \tag{65}$$

is a tighter lower bound than the standard ELBO:

$$\log p_\theta(\boldsymbol{x}_0) \geq \mathcal{F}_{bddm}^{(t)}(\theta,\phi) \geq \mathcal{F}_{elbo}^{(t)}(\theta), \tag{66}$$

where $\mathcal{F}_{elbo}^{(t)}(\theta)$ indicates the standard ELBO evaluated at step $t \in \{2,...,T\}$:

$$\mathcal{F}_{elbo}^{(t)}(\theta) := -\mathbb{E}_{q_{\boldsymbol{\beta}}(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathbb{KL}\left(q_{\boldsymbol{\beta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t,\boldsymbol{x}_0)||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)\right) + \mathcal{R}_\theta(\boldsymbol{x}_0,\boldsymbol{x}_t)\right]. \tag{67}$$

Proof. For the case of $\theta \neq \theta^*$, the left inequality can be directly justified by the Proposition 1. We can prove the right inequality by Proposition 2, which states that $-\mathcal{L}_{\text{score}}^{(t)}(\theta) = \mathcal{F}_{\text{elbo}}^{(t)}(\theta)$. For the case of $\theta = \theta^*$, by Proposition 3, we have $\log p_{\theta^*}(\boldsymbol{x}_0) = \mathcal{F}_{\text{score}}^{(t)}(\theta^*) + \sum_{i=2}^{t}\mathcal{L}_{\text{step}}^{(i)}(\phi;\theta^*) = \mathcal{F}_{\text{bddm}}^{(t)}(\theta^*,\phi) + \sum_{i=2}^{t-1}\mathcal{L}_{\text{step}}^{(i)}(\phi;\theta^*) \geq \mathcal{F}_{\text{bddm}}^{(t)}(\theta^*,\phi) \geq \mathcal{F}_{\text{elbo}}^{(t)}(\theta^*)$, since $\mathcal{L}_{\text{step}}^{(i)}(\phi;\theta^*) \geq 0, \forall i$. □

Remark 2.

$$\mathcal{L}_{step}^{(t)}(\phi;\theta^*) = \frac{1}{2(1-\hat{\beta}_t(\phi)-\alpha_t^2)}\left\|\sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t - \frac{\hat{\beta}_t(\phi)}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t,\alpha_t)\right\|_2^2 + C_t(\phi), \tag{68}$$

where

$$C_t(\phi) := \frac{1}{4}\log\frac{1-\alpha_t^2}{\hat{\beta}_t(\phi)} + \frac{D}{2}\left(\frac{\hat{\beta}_t(\phi)}{1-\alpha_t^2} - 1\right). \tag{69}$$

Proof. By the definition of the following probability density functions:

$$p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}\left(\frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_\theta\left(\boldsymbol{x}_t,\alpha_t\right)\right), \frac{(1-\beta_t-\alpha_t^2)\beta_t}{(1-\beta_t)(1-\alpha_t^2)}\boldsymbol{I}\right) \tag{70}$$

$$q_\phi(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0) = \mathcal{N}\left(\alpha_{t-1}\boldsymbol{x}_0, \left(1-\alpha_{t-1}^2\right)\boldsymbol{I}\right) = \mathcal{N}\left(\frac{\alpha_t}{\sqrt{1-\beta_t}}\boldsymbol{x}_0, \left(1-\frac{\alpha_t^2}{1-\beta_t}\right)\boldsymbol{I}\right), \tag{71}$$

we have

$$\mathcal{L}_{\text{step}}^{(t)}(\phi;\theta^*) := D_{\text{KL}}\left(p_{\theta^*}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)||q_\phi(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)\right) \tag{72}$$

$$= \frac{1-\beta_t}{2(1-\beta_t-\alpha_t^2)}\left\|\frac{\alpha_t}{\sqrt{1-\beta_t}}\boldsymbol{x}_0 - \frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}\left(\boldsymbol{x}_t,\alpha_t\right)\right)\right\|_2^2 + C_t \tag{73}$$

$$= \frac{1-\beta_t}{2(1-\beta_t-\alpha_t^2)}\left\|\frac{\alpha_t}{\sqrt{1-\beta_t}}\boldsymbol{x}_0 - \frac{1}{\sqrt{1-\beta_t}}\left(\alpha_t\boldsymbol{x}_0 + \sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}\left(\boldsymbol{x}_t,\alpha_t\right)\right)\right\|_2^2 + C_t \tag{74}$$

$$= \frac{1-\beta_t}{2(1-\beta_t-\alpha_t^2)}\left\|\sqrt{\frac{1-\alpha_t^2}{1-\beta_t}}\boldsymbol{\epsilon}_t - \frac{\beta_t}{\sqrt{(1-\beta_t)(1-\alpha_t^2)}}\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t,\alpha_t)\right\|_2^2 + C_t \tag{75}$$

$$= \frac{1}{2(1-\beta_t-\alpha_t^2)}\left\|\sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t - \frac{\beta_t}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t,\alpha_t)\right\|_2^2 + C_t, \tag{76}$$

where

$$C_t = \frac{1}{4}\log\frac{1-\alpha_t^2}{\beta_t} + \frac{D}{2}\left(\frac{\beta_t}{1-\alpha_t^2} - 1\right). \tag{77}$$

As we parameterize β_t by ϕ to estimate the noise schedule for inference, we obtain the final step loss:

$$\mathcal{L}_{\text{step}}^{(t)}(\phi;\theta^*) = \frac{1}{2(1-\hat{\beta}_t(\phi)-\alpha_t^2)}\left\|\sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_t - \frac{\hat{\beta}_t(\phi)}{\sqrt{1-\alpha_t^2}}\boldsymbol{\epsilon}_{\theta^*}(\boldsymbol{x}_t,\alpha_t)\right\|_2^2 + C_t(\phi), \tag{78}$$

where

$$C_t(\phi) := \frac{1}{4}\log\frac{1-\alpha_t^2}{\hat{\beta}_t(\phi)} + \frac{D}{2}\left(\frac{\hat{\beta}_t(\phi)}{1-\alpha_t^2} - 1\right). \tag{79}$$

□

B EXPERIMENTAL DETAILS

B.1 CONVENTIONAL GRID SEARCH ALGORITHM FOR DDPMS

We reproduced the grid search algorithm in (Chen et al., 2020), in which a 6-step noise schedule was searched. In our paper, we generalized the grid search algorithm by similarly sweeping the N-step noise schedule over the following possibilities with a bin width $M = 9$:

$$\{1,2,3,4,5,6,7,8,9\} \otimes \{10^{-6\cdot N/N}, 10^{-6\cdot(N-1)/N}, ..., 10^{-6\cdot 1/N}\}, \tag{80}$$

where $\otimes$ denotes the cartesian product applied on two sets. LS-MSE was used as a metric to select the solution during the search. When $N = 6$, we resemble the GS algorithm in (Chen et al., 2020). Note that above searching method normally does not scale up to $N > 6$ steps for its exponential computational cost $\mathcal{O}(9^N)$.

B.2 HYPERPARAMETER SETTING IN BDDMS

Algorithm 2 took a skip factor τ to control the stride for training the scheduling network. The value of τ would affect the coverage of step sizes when training the scheduling network, hence affecting the predicted number of steps N for inference – the higher τ is, the shorter the predicted inference schedule tends to be. We set $\tau = 66$ for training the BDDM vocoders in this paper.

For initializing Algorithm 3 for noise scheduling, we could take as few as 1 training sample for validation, perform a grid search on the hyperparameters $\{(\hat{\alpha}_N = 0.1\alpha_T i, \hat{\beta}_N = 0.1j)\}$ for $i, j = 1, ..., 9$, i.e., 81 possibilities in total, and use the PESQ measure as the selection metric. Then, the predicted noise schedule corresponding to the maximum PESQ was stored and applied to the online inference afterward, as shown in Algorithm 4. Note that this searching has a complexity of only $\mathcal{O}(M^2)$ (e.g., $M = 9$ in this case), which is much more efficient than $\mathcal{O}(M^N)$ in the conventional grid search algorithm in (Chen et al., 2020), as discussed in Section B.1.

B.3 IMPLEMENTATION DETAILS

Our proposed BDDMs and the baseline methods were all implemented with the Pytorch library. The score networks for the LJ and VCTK speech datasets were trained from scratch on a single NVIDIA Tesla P40 GPU with batch size 32 for about 1M steps, which took about 3 days.

For the model architecture, we used the same architecture as in DiffWave (Kong et al., 2021) for the score network with 128 residual channels; we adopted a lightweight GALR network (Lam et al., 2021) for the scheduling network. GALR was originally proposed for speech enhancement, so we considered it well suited for predicting the noise scales. For the configuration of the GALR network, we used a window length of 8 samples for encoding, a segment size of 64 for segmentation and only two GALR blocks of 128 hidden dimensions, and other settings were inherited from (Lam et al., 2021). To make the scheduling network output with a proper range and dimension, we applied a sigmoid function to the last block's output of the GALR network. Then the result was averaged over the segments and the feature dimensions to obtain the predicted ratio: $\sigma_\phi(\boldsymbol{x}) = \text{AvgPool2D}(\sigma(\text{GALR}(\boldsymbol{x})))$, where $\text{GALR}(\cdot)$ denotes the GALR network, $\text{AvgPool2D}(\cdot)$ denotes the average pooling operation applied to the segments and the feature dimensions, and

Table 4: Performances of different noise schedules on the multi-speaker VCTK speech dataset, each of which used the same score network (Chen et al., 2020) $\boldsymbol{\epsilon}_\theta(\cdot)$ that was trained on VCTK for about 1M iterations.

Noise schedule	LS-MSE (↓)	MCD (↓)	STOI (↑)	PESQ (↑)	MOS (↑)
DDPM (Ho et al., 2020; Chen et al., 2020)					
8 steps (Grid Search)	101	**2.09**	**0.787**	**3.31**	**4.22 ± 0.04**
1,000 steps (Linear)	85.0	2.02	0.798	3.39	4.40 ± 0.05
DDIM (Song et al., 2021a)					
8 steps (Linear)	553	3.20	0.701	2.81	3.83 ± 0.04
16 steps (Linear)	412	2.90	0.724	3.04	3.88 ± 0.05
21 steps (Linear)	355	2.79	0.739	3.12	4.12 ± 0.05
100 steps (Linear)	259	2.58	0.759	3.30	4.27 ± 0.04
NE (San-Roman et al., 2021)					
8 steps (Linear)	208	2.54	0.740	3.10	4.18 ± 0.04
16 steps (Linear)	183	2.53	0.742	3.20	4.26 ± 0.04
21 steps (Linear)	852	3.57	0.699	2.66	3.70 ± 0.03
BDDM $(\hat{\alpha}_N, \hat{\beta}_N)$					
8 steps $(0.2, 0.9)$	**98.4**	2.11	0.774	3.18	4.20 ± 0.04
16 steps $(0.5, 0.5)$	**73.6**	**1.93**	**0.813**	**3.39**	**4.35 ± 0.05**
21 steps $(0.5, 0.1)$	**76.5**	**1.83**	**0.827**	**3.43**	**4.48 ± 0.06**

$\sigma(x) := 1/(1 + e^{-x})$. The same network architecture was used for the NE approach for estimating α_t^2 and was shown better than the ConvTASNet used in the original paper (San-Roman et al., 2021). It is also notable that the computational cost of a scheduling network is indeed fractional compared to the cost of a score network, as predicting a noise scalar variable is intrinsically a relatively much easier task. Our GALR-based scheduling network, while being able to produce stable and reliable results, was about 3.6 times faster than the score network. The training of scheduling networks for BDDMs took only 10k steps to converge, which consumed no more than an hour on a single GPU.

B.4 Crowd-sourced subjective evaluation

All our Mean Opinion Score (MOS) tests were crowd-sourced. We refer to the MOS scores in (Protasio Ribeiro et al., 2011), and the scoring criteria have been included in Table 3 for completeness. The samples were presented and rated one at a time by the testers.

Table 3: Ratings that have been used in evaluation of speech naturalness of synthetic samples.

Rating	Naturalness	Definition
1	Unsatisfactory	Very annoying, distortion is objectionable.
2	Poor	Annoying distortion, but not objectionable.
3	Fair	Perceptible distortion, slightly annoying.
4	Good	Slight perceptible level of distortion, but not annoying.
5	Excellent	Imperceptible level of distortion.

C Additional experiments

A demonstration page at https://bilateral-denoising-diffusion-model.github.io shows some samples generated by BDDMs trained on LJ speech and VCTK datasets.

C.1 MULTI-SPEAKER SPEECH SYNTHESIS

In addition to the single-speaker speech synthesis, we evaluated BDDMs on the multi-speaker speech synthesis benchmark VCTK (Yamagishi et al., 2019). VCTK consists of utterances sampled at 48 KHz by 108 native English speakers with various accents. We split the VCTK dataset for training and testing: 100 speakers were used for training the multi-speaker model and 8 speakers for testing. We trained on a 44257-utterance subset (40 hours) and evaluated on a held-out 100-utterance subset. For the score network, we used the Wavegrad architecture (Chen et al., 2020) so as to examine whether the superiority of BDDMs remains in a different dataset and with a different score network architecture.

Results are presented in Table 4. For this multi-speaker VCTK dataset, we obtained consistent observations with that for the single-speaker LJ dataset presented in the main paper. Again, the proposed BDDM with only 16 or 21 steps outperformed the DDPM with 1,000 steps. To the best of our knowledge, ours was the first work that reported this degree of superior. When reducing to 8 steps, BDDM obtained performance on par with (except for a worse PESQ) the costly grid-searched 8 steps (which were unscalable to more steps) in DDPM. For NE, we could again observe a degradation from its 16 steps to 21 steps, indicating the instability of NE for the VCTK dataset likewise. In contrast, BDDM gave continuously improved performance while increasing the step number.

C.2 COMPARING DIFFERENT REVERSE PROCESSES FOR BDDMS

This section demonstrates that BDDMs do not restrict the sampling procedure to a specialized reverse process in Algorithm 4. In particular, we evaluated different reverse processes, including that of DDPMs (see Eq. 4) and DDIMs (Song et al., 2021a), for BDDMs and compared the objective scores on the generated samples. DDIMs (Song et al., 2021a) formulate a non-Markovian generative process that accelerates the inference while keeping the same training procedure as DDPMs. The original generative process in Eq. 4 in DDPMs is modified into

$$p_\theta^{(\tau)}(\boldsymbol{x}_{0:T}) := \pi(\boldsymbol{x}_T) \prod_{i=1}^{S} p_\theta^{(\gamma_i)}(\boldsymbol{x}_{\gamma_{i-1}}|\boldsymbol{x}_{\gamma_i}) \times \prod_{t\in\bar{\boldsymbol{\gamma}}} p_\theta^{(t)}(\boldsymbol{x}_0|\boldsymbol{x}_t), \tag{81}$$

where $\boldsymbol{\gamma}$ is a sub-sequence of length N of $[1, ..., T]$ with $\gamma_N = T$, and $\bar{\boldsymbol{\gamma}} := \{1, ..., T\} \setminus \boldsymbol{\gamma}$ is defined as its complement; Therefore, only part of the models are used in the sampling process.

To achieve the above, DDIMs defined a prediction function $f_\theta^{(t)}(\boldsymbol{x}_t)$ that depends on $\boldsymbol{\epsilon}_\theta$ to predict the observation $\boldsymbol{x}_0$ given $\boldsymbol{x}_t$ directly:

$$f_\theta^{(t)}(\boldsymbol{x}_t) := \frac{1}{\alpha_t}\left(\boldsymbol{x}_t - \sqrt{1-\alpha_t^2}\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_t, \alpha_t)\right). \tag{82}$$

By leveraging this prediction function, the conditionals in Eq. 81 are formulated as

$$p_\theta^{(\gamma_i)}(\boldsymbol{x}_{\gamma_{i-1}}|\boldsymbol{x}_{\gamma_i}) = \mathcal{N}\left(\frac{\alpha_{\gamma_{i-1}}}{\alpha_{\gamma_i}}\left(\boldsymbol{x}_{\gamma_i} - \varsigma\boldsymbol{\epsilon}_\theta(\boldsymbol{x}_{\gamma_i}, \alpha_{\gamma_i})\right), \sigma_{\gamma_i}^2\boldsymbol{I}\right) \quad \text{if} i \in [N], i > 1 \tag{83}$$

$$p_\theta^{(t)}(\boldsymbol{x}_0|\boldsymbol{x}_t) = \mathcal{N}(f_\theta^{(t)}(\boldsymbol{x}_t), \sigma_t^2\boldsymbol{I}) \quad \text{otherwise}, \tag{84}$$

where the detailed derivation of σ_t and ς can be referred to (Song et al., 2021a). In the original DDIMs, the accelerated reverse process produces samples over the subsequence of $\boldsymbol{\beta}$ indexed by $\boldsymbol{\gamma}$: $\hat{\boldsymbol{\beta}} = \{\beta_n | n \in \boldsymbol{\gamma}\}$. In BDDMs, to apply the DDIM reverse process, we use the $\hat{\boldsymbol{\beta}}$ predicted by the scheduling network in place of a subsequence of the training schedule $\boldsymbol{\beta}$.

Finally. the objective scores are given in Table 5. Note that the subjective evaluation (MOS) is omitted here since the other assessments above have shown that the MOS scores are highly correlated with the objective measures, including STOI and PESQ. They indicate that applying BDDMs to either DDPM or DDIM reverse process leads to comparable and competitive results. Meanwhile, the results show some subtle differences: BDDMs over a DDPM reverse process gave slightly better samples in terms of signal error and consistency metrics (i.e., LS-MSE and MCD), while BDDM over a DDIM reverse process tended to generate better samples in terms of intelligibility and perceptual metrics (i.e., STOI and PESQ).

Table 5: Performances of different reverse processes for BDDMs on the VCTK speech dataset, each of which used the same score network (Chen et al., 2020) $\boldsymbol{\epsilon}_\theta(\cdot)$ and the same noise schedule.

Noise schedule	LS-MSE (↓)	MCD (↓)	STOI (↑)	PESQ (↑)
BDDM (DDPM reverse process)				
8 steps $(0.3, 0.9, 1e^{-5})$	**91.3**	**2.19**	0.936	3.22
16 steps $(0.7, 0.1, 1e^{-6})$	**73.3**	**1.88**	0.949	3.32
21 steps $(0.5, 0.1, 1e^{-6})$	**72.2**	**1.91**	0.950	3.33
BDDM (DDIM reverse process)				
8 steps $(0.3, 0.9, 1e^{-5})$	91.8	**2.19**	**0.938**	**3.26**
16 steps $(0.7, 0.1, 1e^{-6})$	77.7	1.96	**0.953**	**3.37**
21 steps $(0.5, 0.1, 1e^{-6})$	77.6	1.96	**0.954**	**3.39**

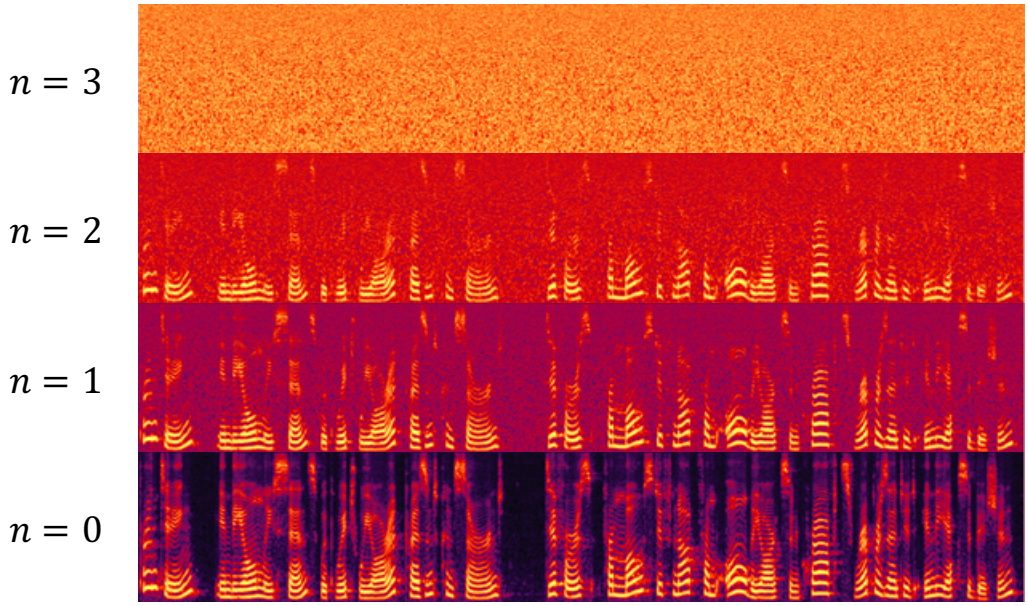

Figure 4: Spectrum plots of the speech samples produced by BDDM within 3 sampling steps. The first row shows the spectrum of a random signal for starting the reverse process. Then, from the top to the bottom, we show the spectrum of the resultant signal after each step of the reverse process performed by the BDDM. We also provide the corresponding WAV files on our demo page.