Unpaired audio restoration using Schrödinger Bridges

Andreas Hansen Bagge
DTU Compute
Technical University of Denmark
s214630@dtu.dk

Abstract

Unpaired audio-to-audio translation, also called audio domain transferring, is the problem of finding a mapping between audio domains which retains the content of the audio samples while applying the characteristics of another domain. A potential framework for performing unpaired audio domain transferring is to interpret the audio domains as probability distributions and then solving the Schrödinger Bridge problem, i.e. an entropy-regularized optimal transport problem between probability distributions. This paper examines the use of the Diffusion Schrödinger Bridge (DSB) algorithm applied to two unpaired audio restoration tasks, declipping and dereveberation. The DSB algorithm is computationally heavy and we therefore propose training on lower-dimensional Log Mel-spectrograms which can be decoded into speech using a vocoder. Additionally, we introduce a theoretically justified pretraining step to the DSB algorithm. We compare our method to a similar diffusion based approach called a Gaussian flow bridge (GFB) which maps the samples into an intermediate normal distribution, and we show that DSB outperforms GFB for few step diffusion by generating more straight trajectories while GFB is best at many step diffusion.

1 Introduction

A classic problem in audio processing is audio restoration tasks such as denoising, declipping, or dereverberation. Traditionally, such problems are solved with a paired, supervised approach in which the degraded audio is augmented by modifying the clean, target audio - for example by adding noise or clipping the amplitude - thereby obtaining paired examples of clean and degraded audio. However, this approach does not allow for training on actual, real-life degraded audio where we do not have access to both the degraded and clean signal in pairs. In this case, unsupervised methods are required. The problem of unpaired translation is also common in many other applications outside audio restoration tasks [20, 38, 17]. Examples of audio-to-audio tasks that could benefit from unsupervised methods also includes gender-to-gender, speaker-to-speaker, microphone-to-microphone, and so on.

Diffusion models are the current state-of-the-art models for image and audio synthesis [8, 30, 4, 13], and they have also showed great results in the field of paired speech enhancement [26]. Diffusion models generally consists of two processes, a forward and backward process. In traditional diffusion models, the forward process is capable of gradually turning data samples into Gaussian noise. The models can then synthesize high quality samples by employing the learned reverse process, i.e. the backward process, starting from noise. Though traditional diffusion methods yields high quality samples, they do not allow for direct transferring between two arbitrary data distributions. Diffusion is similar to flow matching and have been shown to fit into a unifying framework called stochastic interpolants [1].

A possible approach that allows for high quality sample generation via diffusion aswell as direct transferring between two arbitrary data distributions is the theory behind Schrödinger Bridges (SB), an entropy-regularized optimal transport problem of finding the most likely random evolution between two continuous probability distributions [29]. These distributions are most often refered to as the prior and data distribution, p_{prior} and p_{data}. By applying the SB problem to an audio restoration task and interpreting the clean and degraded audio as the data and prior distribution respectively, the problem boils down to finding the most likely random process than can transform degraded audio into clean audio, and vice versa. The SB problem is formulated in terms of stochastic differential equations (SDE's) which can be simulated with SDE solvers, and solving the SB problem therefore allows for a diffusion-based generative approach, having the potential of also yielding high-quality samples. Utilizing Schrödinger Bridges is therefore a valid approach for obtaining high quality samples on unpaired audio-to-audio tasks. Our code and audio examples are available at github.com/kommodeskab/Latent-DSB.

2 Similar work

I^{2}SB [15] (Image-to-Image Schrödinger Bridge) is a tractable, simulation-free solution to the SB problem betwen the distributions $p_A(X_0)$ and $p_B(X_1|X_0)$, i.e. a conditional probability distribution that requires paired samples. I^{2}SB is used for a range of image restoration tasks and yields state-of-the-art results. Though this approach is named after images, it generalizes to all kind of data and has also been applied to audio in an analogous approch called A^{2}SB (Audio-to-Audio Schrödinger Bridge) [14, 9].

[20] propose using Gaussian Flow Bridges, a flow-matching based approach where samples are first mapped to a simple Gaussian, for doing unpaired domain transferring on audio samples. The approach suffers from the fact that the simulated trajectories first have to map to a third distribution, the Gaussian, therefore yielding non-optimal transport.

The ReFlow algorithm [18], which is an extension of ordinary flow matching, uses iterative fitting to map between unpaired probability distributions but it uses ordinary differential equations, therefore yielding deterministic results. Also, to the best of our knowledge, the ReFlow algorithm have not been applied to an audio-to-audio task.

RemixIT [35] is a example of a simple self-supervised method for training unpaired speech enhancement, though it can in theory be applied to many other audio restoration tasks. The RemixIT algorithm uses a teacher-student setup in which the teacher, which is a prior version of the student, provides pesudo targets for the student. However, this approach requires that the model is pretrained on an in-domain distribution and is therefore not fully unpaired.

3 Methods

Our work builds on Diffusion Schrödinger Bridge (DSB), an iterative approach for solving the Schrödinger Bridge problem (SBP) using the iterative proportional fitting (IPF) algorithm. The resulting approach is called the DSB algorithm and it uses iterative optimization for solving the problem. The algorithm was originally introduced by [2] but was later further simplified by [33] with the Reparametrized DSB (R-DSB) in which the performance of the algorithm was greatly improved. We will refer to the simplified algorithm as simply DSB. The DSB algorithm seeks to solve the discrete SBP by finding the random process π^* such that:

$$\pi^* = \arg\min_{\pi}\{D_{KL}(\pi||p),\ \pi \in \mathscr{P}_{N+1},\ \pi_0 = p_{data},\ \pi_N = p_{prior}\} \tag{1}$$

Where D_{KL} is the Kullback-Leibler divergence, $\pi_0 = \pi(x_0)$ and $\pi_N = \pi(x_N)$, and $\mathscr{P}_l = \mathscr{P}((\mathbb{R}^d)^l)$ is the set of l-state joint probability distributions for $l \in \mathbb{N}$, i.e. a probability distribution on the form $p(x_0, x_1, \dots x_{l-1})$ where $x_i \in \mathbb{R}^d$. p is the 'reference'-dynamic, i.e. some initial random process. Researchers have shown that the discrete SBP can be solved using the iterative proportional fitting algorithm (IPF) (also called Sinkhorn's algorithm) [19] using the following iterative approach:

$$\begin{aligned}\pi^{2n+1} &= \arg\min_{\pi}\{D_{KL}(\pi|\pi^{2n})\ :\ \pi_N = p_{prior}\}\\ \pi^{2n+2} &= \arg\min_{\pi}\{D_{KL}(\pi|\pi^{2n+1})\ :\ \pi_0 = p_{data}\}\end{aligned} \tag{2}$$

With $\pi^0 = p$ being the initial random process. As $n \to \infty$, π^{2n+1} and π^{2n+1} will approximate the optimal solution π^*. [2] shows that by parameterizing the joint probability distributions as Markov chains, the Kullback-Leibler divergence can be minimized using Bayes while upholding the marginal constraints. We refer to [33, 2] for the details.

Each iteration n of the DSB algorithm requires two processes, a forward and backward processes, defined as the conditional probabilities $p^n_{k+1|k}(x_{k+1}|x_k)$ and $q^n_{k|k+1}(x_k|x_{k+1})$ respectively. The forward process is used to (approximately) transfer points from p_{data} to p_{prior} and vice versa using

$$p^n_{k+1|k}(x_{k+1}|x_k) = \mathcal{N}(x_{k+1}; x_k + \gamma_k f^n_k(x_k), 2\gamma_{k+1}\mathbf{I}) \tag{3}$$

$$q^n_{k|k+1}(x_k|x_{k+1}) = \mathcal{N}(x_k; x_{k+1} + \gamma_{k+1} b^n_{k+1}(x_{k+1}), 2\gamma_{k+1}\mathbf{I}) \tag{4}$$

Where γ_k is the stepsize, and $f^n_k(x_k)$ and $b^n_{k+1}(x_{k+1})$ are the sought-after 'flows' of the processes which indicates the direction in which samples should move to successfully transfer from one distribution to the other. In traditional diffusion models, the forward process follows some simple, closed form, non-learnable process such as an Ornstein-Uhlenback process [36], which means that n and f^n are fixed. These non-learnable processes are only valid for transferring to a simple distribution, most often a Gaussian.
The objective of the DSB algorithm is to iteratively learn the reverse processes, i.e. to learn $p^n_{k|k+1}$ and $q^n_{k+1|k}$ respectively. For the first iteration, $n = 0$, the objective is therefore to first learn $p^0_{k|k+1}$ which can approximately be derived using Bayes (see equation 5 amd 6). Hereafter, we set $q^0_{k|k+1} = p^0_{k|k+1}$ in order to learn $q^0_{k+1|k}$. We then set $p^1_{k+1|k} = q^0_{k+1|k}$ and the process is repeated for $n = 1, 2, ..L$ where L is the number of "DSB iterations". [33] shows that the inverse processes $q^n_{k|k+1}$ and $p^{n+1}_{k+1|k}$ can approximately be derived as the posterior distributions:

$$q^n_{k|k+1}(x_k|x_{k+1}) \approx p^n_{k|k+1,0}(x_k|x_{k+1}, x_0) = \mathcal{N}(x_k; \mu^n_{k+1}(x_{k+1}, x_0), \sigma_{k+1}\mathbf{I}) \tag{5}$$

$$p^{n+1}_{k+1|k}(x_{k+1}|x_k) \approx q^n_{k+1|k,N}(x_{k+1}|x_k, x_N) = \mathcal{N}(x_{k+1}; \tilde{\mu}^n_k(x_k, x_N), \tilde{\sigma}_{k+1}\mathbf{I}) \tag{6}$$

Where:

$$\mu^n_{k+1}(x_{k+1}, x_0) \approx x_{k+1} + \frac{\gamma_{k+1}}{\bar{\gamma}_{k+1}}(x_0 - x_{k+1}), \qquad \sigma_{k+1} = \frac{2\gamma_{k+1}\bar{\gamma}_k}{\bar{\gamma}_{k+1}} \tag{7}$$

$$\tilde{\mu}^n_k(x_k, x_N) \approx x'_k + \frac{\gamma_{k+1}}{\bar{\gamma}_N - \bar{\gamma}_k}(x_N - x_k), \qquad \tilde{\sigma}_{k+1} = \frac{2\gamma_{k+1}(\bar{\gamma}_N - \bar{\gamma}_{k+1})}{\bar{\gamma}_N - \bar{\gamma}_k} \tag{8}$$

Where $\bar{\gamma}_k = \sum_{i=1}^k \gamma_i$ and $\bar{\gamma}_0 = 0$. The above derivation suggests that learning the forward and backward processes boils down to learning the straight line connecting the current state (x_k or x_{k+1}) to the terminal points (x_N or x_0). We call this the forward and backward flow respectively. In practice, learning the backward process works by sampling a forward trajectory $\{x_k\}_{k=0}^N$ using $p^n_{k+1|k}(x_{k+1}|x_k)$ and then learning the backward flow $\frac{x_0 - x_{k+1}}{\bar{\gamma}_{k+1}}$ and vice versa for learning the forward flow. The losses can then be written as:

$$\mathcal{L}^b = \mathbb{E}_{(x_0, x_{k+1}) \sim p^n_{0,k+1}}\left(\left\|\tilde{b}_{k+1}(x_{k+1}, k+1) - \frac{x_0 - x_{k+1}}{\bar{\gamma}_{k+1}}\right\|^2\right) \tag{9}$$

$$\mathcal{L}^f = \mathbb{E}_{(x_k, x_N) \sim q^n_{k,N}}\left(\left\|\tilde{f}_k(x_k, k) - \frac{x_N - x_k}{\bar{\gamma}_N - \bar{\gamma}_k}\right\|^2\right) \tag{10}$$

Where $\tilde{b}_{k+1}(x_{k+1}, k+1)$ and $\tilde{f}_k(x_k, k)$ are the flows parametrized as trainable neural networks. Here, we have dropped the superscript ^{n} from the parametrized flow to signify that the same neural networks can be iteratively finetuned to fit the new flows in each iteration, i.e. we do not need to initialize a new network for every n. As it is common in diffusion models, the model also takes the current state index k as an input to allow time-varying flows. As $n \to \infty$, the process will in theory converge to a solution to the SBP. In practice, [33] shows that only a few DSB iterations, around $L \approx 6$ are sufficient for attaining adequate results.

Assuming a symmetrical time schedule, i.e. $\gamma_k = \gamma_{N-k+1}$, the processes also becomes symmetrical. The time schedule can also be interpreted as the resolution of the steps taken when solving the SDE and smaller steps will generally result in better results at the cost of increased inference time. Symmetric time schedules makes sense when working with data distributions as p_{data} and p_{prior} (as opposed to a data distribution and Gaussian distribution) since preservation of details are important for both distributions. Symmetric time schedules are also adapted by [15, 33] and the like.

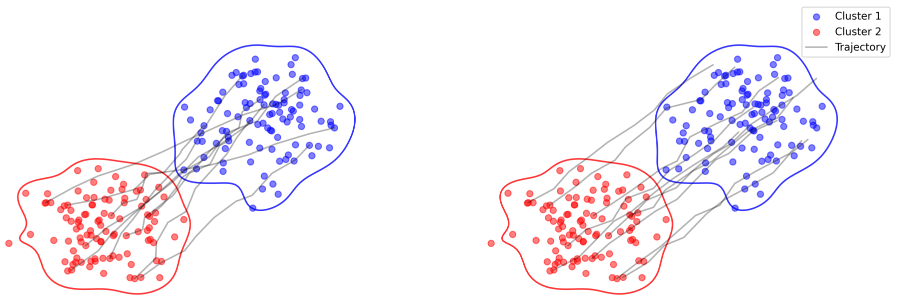

Figure 1: Left figure shows the initial process when using DSB pretraining. Here, the initial trajectories are calculated by interpolating between random pairs of points from p_{data} and p_{prior} but including some stochasticity. The resulting flows are non-optimal. Right figure shows ideal trajectories after some number of iterations of the DSB algorithm where the resulting flows are more optimal.

3.1 Smart initialization

The IPF algorithm requires some initial forward process for the fist iteration of the algorithm denoted $p^0_{k+1|k}$. Instead of initializing a simple but far from optimal process such as an Ornstein-Uhlenbeck process, [33] propose to initially use a pretrained Flow Matching (FM) model [18]. Traditional FM models are trained to approximate the ODE flow between two data distributions, p_{prior} and p_{data}, but requires p_{prior} to follow a normal distribution, i.e. $p_{prior} \sim \mathcal{N}(0,1)$. However, [33] shows that simply replacing the normal distribution with the desired data distribution yields a bridge that is non-optimal but still serves as a much better initialization than using a more simple process. To further adhere to the theory and stochasticity underlying DSB, we instead propose sticking to the DSB posterior transitions. In the first iteration of the DSB algorithm, instead of calculating the flow, f^0_{k+1}, of the forward process using a neural network, $\tilde{f}^0_{k+1}$, we instead sample $\tilde{x}_N$ randomly from the training data and calculate trajectories using the flow given by $\frac{\tilde{x}_N - x_k}{\bar{\gamma}_N - \bar{\gamma}_k}$. This approach does not rely on other theoretical frameworks, it is easily implemented in the DSB algorithm, and it can intuitively be interpreted as a process whose flow is a straight line connecting p_{prior} and p_{data} randomly. The approach also closely resemble the simulation-free approach outlined in [15]. We call this "DSB pretraining" and it is outlined alongside the general training algorithm in 3.2. The idea behind DSB pretraining is visualized in figure 1.

3.2 Dimensionality reduction

The DSB algorithm is computationally demanding due to two reasons: 1) the training data (generated trajectories) is generated during training by the model not currently being trained, and 2) said generated data needs to be periodically cached in memory. These problems can be mitigated by reducing the dimensionality of the training data since lower dimensionality allows for faster sampling and for more samples to be stored in the cache. We therefore propose training on Log Mel spectrograms instead of raw waveforms or normal spectrograms. During inference, the Log Mel spectrograms can be decoded using a vocoder such as [12]. Further reduction can be obtained by other audio codecs such as audio VAE's, and latent diffusion models exhibits increased computational efficiency at only slightly reduced sample quality [7, 27]. Additionally, the number of samples in the cache can be increased by reducing the precision of the samples.

4 Experiments

4.1 Experimental setup

We test the DSB algorithm on an unpaired speech declipping task and reverberation task. These tasks are traditionally not unpaired since degraded samples can very easily be augmented. However,

Algorithm 1 Training algorithm

```
Initialize two neural networks, $\tilde{f}_k(x_k, k)$ and $\tilde{b}_{k+1}(x_{k+1}, k+1)$
$L =$ number of DSB iterations
$K =$ training steps before refreshing cache

for $n = 0, 1, ..L$ do
    while not converged do
        if $n = 0$ then
            Sample $\tilde{x}_N \sim p_{prior}$ and set $f_k^n(x_k) = \frac{\tilde{x}_N - x_k}{\bar{\gamma}_N - \bar{\gamma}_k}$. {Pretraining step}
        end if
        Sample $x_0 \sim p_{data}$
        Calculate trajectory $\{x_k\}_{k=0}^N$ using (3) starting from $x_0$ and using $f_k^n(x_k) = \tilde{f}_k(x_k, k)$.
        Save trajectory in cache.
        for $c = 0, 1, ..K$ do
            Sample trajectory from cache, calculate loss using (9) and perform gradient descent.
        end for
    end while
    Clear cache
    while not converged do
        Sample $x_N \sim p_{prior}$
        Calculate trajectory $\{x_k\}_{k=0}^N$ using (4) starting from $x_N$ and using $b_{k+1}^n(x_{k+1}) = \tilde{b}_{k+1}(x_{k+1}, k+1)$.
        Save trajectory in cache.
        for $c = 0, 1, ..K$ do
            Sample trajectory from cache, calculate loss using (10) and perform gradient descent.
        end for
    end while
    Clear cache
end for
```

by augmenting the samples ourselves we gain access to the clean underlying signals and we can therefore better quantify our results. For our experiments, we train with $N = 100$ and a uniform (and therefore symmetric) time schedule with $\gamma_k = 0.01$. During sampling, we set $\gamma_k = 1/N$ where N is the number of diffusion steps. For the clean speech data denoted p_{data}, we used the VCTK clean dataset [11]. We used speakers `p225` to `p230` for validation. The degraded data denoted p_{prior} used the same speech dataset which is then degraded.

Clipped audio is generated by drawing and applying a random decibel gain between 5 and 30, capping the resulting signal at ± 1, and then dividing by the gain.

Reveberated audio is augmented by applying room-impulse-reponses (RIR) taken from various online sources [32, 31, 23, 21].

We test two different approaches; using raw STFT spectrograms and Log Mel spectrograms. Audio clips are 4.096 seconds long for the STFT approach and 4.47 seconds long for the Log Mel approach (to accomodate model architecture) and sampled at 16khz. The STFT spectrograms are calculated using `n_fft=win_length=510` and `hop_length=128`, and the resulting real and imaginary components are stacked on top of each other along the channel dimension. The log melspectrograms are calculated using `n_fft=win_length=1024`, `n_mels=64`, and `hop_length=160`. The spectrograms are decoded using a SpeechT5 vocoder trained by SpeechBrain [16]. DSB pretraining was used on both the forward and backward model for 200k iterations. Each model in each iteration of the DSB algorithm was trained for $\max(64000 \cdot 0.7^{n-1}, 16000)$ iterations for $n = 1 \dots 8$ with the same hyperparameters, resulting in a total of 426,110 training steps per forward/backward model. We observe that the first few DSB iterations takes longer to converge and therefore spend more training iterations here. For the STFT approach we use the same 2d-convolutional UNet achitecture as in [20] (approximately 41.1 million parameters), and for the Log Mel approach we use a 1d-convolutional UNet from huggingface based on [28] (approximately 48.1 million parameters). Throughout we use batch size 8 for the STFT approach and batch size 64 for the Log Mel approach and a AdamW optimizer with 10^{-4} learning rate. The models are trained using a single A100 GPU from DTU HPC [5].

4.2 Evaluation setup

We only evaluate the backward process, i.e. the process capable of generating clean speech samples. For test data, we use 256 samples from the LibriSpeech clean dataset [22] with the same parameters and augmentation as for the training dataset. The clipped audio samples are fixed at SDR=2. The reveberated test data is generated using a so far unseen RIR dataset [34]. The generated samples are evaluated using WER (word-error-rate), MOS (mean-opinion-score) [25], SR-CS (speaker recognition cosine similarity), and KAD (kernel audio distance) [3]. WER and SR-CS can be interpreted as content fidelity measures capturing how well content is preserved, while MOS and KAD can be interpreted as perceptual quality metrics capturing how good or natural the audio sounds. For most audio restoration tasks such as speech enhancement or declipping, signal-to-noise ratio or similar metrics are often used for estimating how much the estimated signal deviates from the ground-truth signal. However, for stochastic generative models such measures are unreliable since the objective is not to restore a ground truth signal but rather to generate a new, probable signal. We also wish to evaluate how well the generated samples match the marginal distribution p_{data} since this is the true objective of the proposed model. We evaluate this using KAD (kernel audio distance) which is simply a scaled version of maximum mean discrepancy (MMD). We compare our method with a baseline (i.e. the unaltered degraded samples) and a similar unpaired, diffusion-based approach called Gaussian Flow Bridge (GFB) [20]. GFB differs from our approach since it encodes and decodes audio samples to and from a Gaussian distribution. Additionally, said GFB model is trained with conditional knowledge about the ground-truth SDR for declipping and T_{60} and C_{60} (reveberation time and clarity) for dereveberation during training which we do not. GFB uses a cosine time schedule and is trained on STFT encodings. The methods are evaluated at various diffusion steps ranging from 50 to 1 step, though GFB can not do 1 step diffusion since it needs to both encode and decode at least once. For the GFB model, we use the exact implementation and model weights as in [20] which was trained for 300k iterations.

WER WER is calculated using the small Whisper model from Open-AI [24] to obtain ground-truth transcriptions. We remove punctuations from the transcriptions and convert everything to lower case. The word-error-rate is then calculated as $WER = \frac{S+D+I}{S+D+C}$ where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and C is the number of correct words.

SR-CS SR-CS is calculated using a speaker embedding model trained by NVIDIA [12] to obtain speaker embeddings of the ground truth signal and the generated signal. The metric is then calculated as the cosine similarity between these embeddings.

MOS MOS is calculated using the MOS estimator DNSMOS [25]. DNSMOS is trained to estimate human subjective scores between 1 and 5 of audio and does not require access to a ground truth reference signal.

KAD KAD is calculated as KAD $= \alpha \cdot \widehat{MMD}^2_{unbiased}$, where $MMD^2_{unbiased}$ is the distribution-free, unbiased maximum mean discrepancy measure between two distributions, in this case our generated samples and p_{data}. [3] demonstrates that KAD is an advantagous alternative to FAD (Fréchet audio distance) which is otherwise the industry standard [10]. The samples used in the KAD metric is first encoded using a CLAP model [6, 37].

4.3 Trajectory analysis

Optimal transport paths, such as those learned by FM or SB models, are straight trajectories from the initial point to the end point, which we will denote x_0 and x_1 (not to be confused with the first and second samples in the trajectory). We therefore analyze the straightness of the generated trajectories by calculating curvature displacement as well as curvature lengths. Obviously, since SB models is stochastic, the generated trajectories are not truly straight, but the learned flow is. Therefore, to faily estimate the straightness of our trajectories, we use deterministic sampling during inference, i.e. we

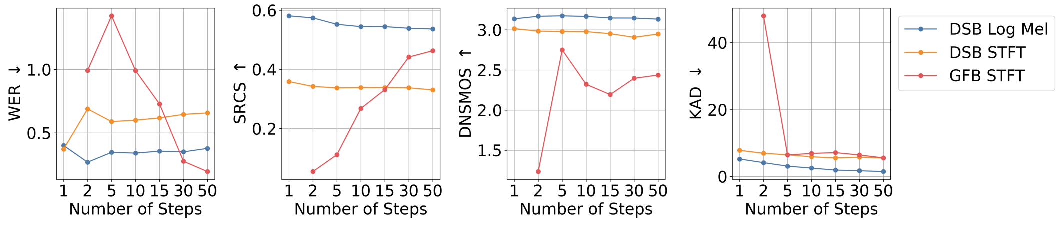

Figure 2: Results from declipping experiments.

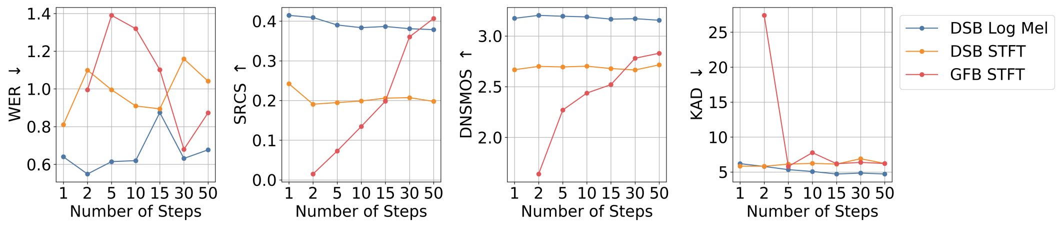

Figure 3: Results from dereveberation experiments.

set $\sigma_{k+1} = \tilde{\sigma}_{k+1} = 0$. Straight flows are not only theoretically interesting, it also opens the door for few step diffusion.

Curvature displacement Curvature displacement is calculated as $C(t) = \left\|(x_1 - x_0) - \frac{\partial x_t}{\partial t}\right\|$. For a given timestep t, we calculate ∂x_t as $x_{k=i+1} - x_{k=i}$ and set $\partial t = \gamma_{k=i+1}$ for the index i associated with the time step t. We report the normalized curvature displacement, i.e. $\hat{C}(t) = \frac{C(t)}{\|x_1 - x_0\|}$. $\hat{C}(t) = 0$ indicates a perfectly straight flow [18].

Since GFB first transfer to an intermediate gaussian distribution, we examine the forward and backward flow independently. We only compare

5 Results and discussion

The results from the declipping and dereveberation experiments are shown in figures 2 and 3 and in table 1. The experiments show that DSB outperforms GFB in most cases when doing few step diffusion, i.e. when the number of diffusion steps is low. However, as the number of diffusion steps increase, GFB often closes the gap and surpasses DSB in some metrics. The DSB models using Log Mel spectrograms most often outperforms other models using traditional STFT spectrograms. Interestingly, performance drops off for DSB on the metrics WER and SR-CS when the number of diffusion steps increases while DNSMOS and KAD stays mostly the same. This means that while the model correctly learns to model the marginal distribution and generate convincing samples, the content is not necessarily conserved. This might be an effect of the random nature of Schrödinger Bridges which adds stochasticity during sampling.
Results from the trajectory analysis can be seen in figure 4. For both GFB and DSB, the biggest displacement happens at the end points near the marginal distributions resulting in a U-shape. Since GFB encodes and decodes to and from an intermediate Gaussian distribution, this U shape is exhibited twice. Oppositely, since DSB maps directly between p_{data} and p_{prior}, the U shape is only exhibited once. DSB generates more straight trajectories which also explains why it excels at few step diffusion.

6 Conclusion

We investigate the use of diffusion Schrödinger Bridges on unpaired audio restoration tasks, namely declipping and dereveberation. We test the algorithm on objective content and perceptual metrics. The results show that the DSB algorithm is capable of generating audio samples that resemble the target distribution while approximately conserving content in the audio samples. The DSB algorithm benefits from being trained on Log Mel spectrograms and the models trained on Log Mel

Table 1: Evaluation metrics for different methods

	MOS ↑	WER ↓	SR-CS ↑	KAD ↓
Declipping				
DSB STFT (30 step)	2.91	0.65	0.34	5.80
DSB STFT (1 step)	3.02	0.37	0.36	7.82
DSB Log Mel (30 step)	**3.15**	0.35	0.54	**1.73**
DSB Log Mel (1 step)	3.14	0.40	**0.58**	5.24
GFB STFT (30 step)	2.40	**0.28**	0.44	6.44
Baseline	2.68	0.15	0.47	15.18
Dereveberation				
DSB STFT (30 step)	2.66	1.16	0.21	6.90
DSB STFT (1 step)	2.67	0.81	0.24	5.84
DSB Log Mel (30 step)	3.17	**0.63**	0.38	**4.85**
DSB Log Mel (1 step)	**3.17**	0.64	**0.41**	6.20
GFB STFT (30 step)	2.78	0.68	0.36	6.38
Baseline	1.88	0.14	0.56	11.41

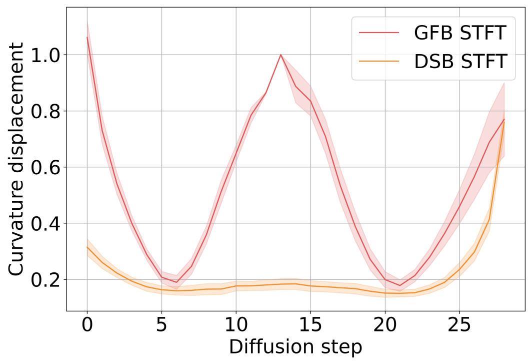

Figure 4: Curvature displacement with 30 diffusion steps. The DSB model is using deterministic sampling, i.e. $\sigma_{k+1} = \tilde{\sigma}_{k+1} = 0$.

spectrograms generally performs better than the STFT alternative on all metrics. Trajectory analysis show that the generated trajectories are more straight than those of a Gaussian Flow Bridge which also explains why DSB excels at few step diffusion, though the DSB model is usually outperformed by GFB when doing many step diffusion. Despite the benefit of few step diffusion, the DSB algorithm is more difficult to implement and requires many training steps before convergence. This is a problem regarding scalability of the model and further research is required for addressing its computational inefficiency.

References

[1] Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions, 2023.

[2] Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion schrödinger bridge with applications to score-based generative modeling, 2023.

[3] Yoonjin Chung, Pilsun Eu, Junwon Lee, Keunwoo Choi, Juhan Nam, and Ben Sangbae Chon. Kad: No more fad! an effective and efficient evaluation metric for audio generation, 2025.

[4] Prafulla Dhariwal and Alex Nichol. Diffusion models beat gans on image synthesis, 2021.

[5] DTU Computing Center. DTU Computing Center resources, 2024.

[6] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap: Learning audio concepts from natural language supervision, 2022.

[7] Zach Evans, Julian D. Parker, CJ Carr, Zack Zukowski, Josiah Taylor, and Jordi Pons. Stable audio open, 2024.

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[9] Ante Jukić, Roman Korostik, Jagadeesh Balam, and Boris Ginsburg. Schrödinger bridge for generative speech enhancement, 2024.

[10] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi. Fréchet audio distance: A metric for evaluating music enhancement algorithms, 2019.

[11] Yamagishi Junichi; Veaux Christophe; MacDonald Kirsten. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92), 2019.

[12] Nithin Rao Koluguri, Taejin Park, and Boris Ginsburg. Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global context. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8102–8106, 2022.

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

[14] Zhifeng Kong, Kevin J Shih, Weili Nie, Arash Vahdat, Sang gil Lee, Joao Felipe Santos, Ante Jukic, Rafael Valle, and Bryan Catanzaro. A2sb: Audio-to-audio schrodinger bridges, 2025.

[15] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A. Theodorou, Weili Nie, and Anima Anandkumar. I^2sb: Image-to-image schrödinger bridge, 2023.

[16] Haohe Liu, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Qiao Tian, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D. Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining, 2024.

[17] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks, 2018.

[18] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow, 2022.

[19] Simone Di Marino and Augusto Gerolin. An optimal transport approach for the schrödinger bridge problem and convergence of sinkhorn algorithm, 2019.

[20] Eloi Moliner, Sebastian Braun, and Hannes Gamper. Gaussian flow bridges for audio domain transfer with unpaired data, 2024.

[21] Damian Murphy and Frank Stevens. open air library_2025, 2025.

[22] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015.

[23] RWCP (Real World Computing Partnership). Rwcp (rwcp-ssd), mar 2007.

[24] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022.

[25] Chandan K A Reddy, Vishak Gopal, and Ross Cutler. Dnsmos: A non-intrusive perceptual objective speech quality metric to evaluate noise suppressors, 2021.

[26] Julius Richter, Simon Welker, Jean-Marie Lemercier, Bunlong Lay, and Timo Gerkmann. Speech enhancement and dereverberation with diffusion-based generative models, 2023.

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[29] E. Schrödinger. Sur la théorie relativiste de l'électron et l'interprétation de la mécanique quantique. *Annales de l'institut Henri Poincaré*, 2(4):269–310, 1932.

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

[31] Rebecca Stewart and Mark Sandler. Database of omnidirectional and b-format impulse responses. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Dallas, Texas, March 2010.

[32] Igor Szöke, Miroslav Skácel, Ladislav Mošner, Jakub Paliesek, and Jan Černocký. Building and evaluation of a real room impulse response dataset. *IEEE Journal of Selected Topics in Signal Processing*, 13(4):863–876, 2019.

[33] Zhicong Tang, Tiankai Hang, Shuyang Gu, Dong Chen, and Baining Guo. Simplified diffusion schrödinger bridge, 2024.

[34] James Traer and Josh H. McDermott. Statistics of natural reverberation enable perceptual separation of sound and space. *Proceedings of the National Academy of Sciences*, 113(48):E7856–E7865, 2016.

[35] Efthymios Tzinis, Yossi Adi, Vamsi K. Ithapu, Buye Xu, Paris Smaragdis, and Anurag Kumar. Remixit: Continual self-training of speech enhancement models via bootstrapped remixing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1329–1341, October 2022.

[36] Wikipedia contributors. Ornstein–uhlenbeck process — Wikipedia, the free encyclopedia, 2025. [Online; accessed 29-March-2025].

[37] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Marianna Nezhurina, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation, 2024.

[38] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.