MULTI-SOURCE DIFFUSION MODELS FOR SIMULTANEOUS MUSIC GENERATION AND SEPARATION

Giorgio Mariani*
Sapienza University of Rome
mariani@di.uniroma1.it

Irene Tallini*
Sapienza University of Rome
tallini@di.uniroma1.it

Emilian Postolache*
Sapienza University of Rome
postolache@di.uniroma1.it

Michele Mancusi*
Sapienza University of Rome
mancusi@di.uniroma1.it

Luca Cosmo†
Ca' Foscari University of Venice
luca.cosmo@unive.it

Emanuele Rodolà†
Sapienza University of Rome
rodola@di.uniroma1.it

ABSTRACT

In this work, we define a diffusion-based generative model capable of both music synthesis and source separation by learning the score of the joint probability density of sources sharing a context. Alongside the classic total inference tasks (i.e., generating a mixture, separating the sources), we also introduce and experiment on the partial generation task of source imputation, where we generate a subset of the sources given the others (e.g., play a piano track that goes well with the drums). Additionally, we introduce a novel inference method for the separation task based on Dirac likelihood functions. We train our model on Slakh2100, a standard dataset for musical source separation, provide qualitative results in the generation settings, and showcase competitive quantitative results in the source separation setting. Our method is the first example of a single model that can handle both generation and separation tasks, thus representing a step toward general audio models.

1 INTRODUCTION

Generative models have recently gained much attention thanks to their successful application in many fields, such as NLP (OpenAI, 2023; Touvron et al., 2023; Santilli et al., 2023), image synthesis (Ramesh et al., 2022; Rombach et al., 2022) or protein design (Shin et al., 2021; Weiss et al., 2023; Minello et al., 2024). Audio is no exception to this trend (Agostinelli et al., 2023; Liu et al., 2023).

A peculiarity of the audio domain is that an audio sample $\mathbf{y}$ can be seen as the sum of multiple individual sources $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, resulting in a mixture $\mathbf{y} = \sum_{n=1}^{N} \mathbf{x}_n$. Unlike in other sub-fields of the audio domain (e.g., speech), sources present in musical mixtures (stems) share a *context* given their strong interdependence. For example, the bass line of a song follows the drum's rhythm and

*Equal contribution. Listing order is random. G.M. wrote most of the code, performed most objective experiments, and contributed to the development of the Dirac separator. I.T. proposed and developed the idea of the Dirac separator and partly formalized its proof, contributed to the code and the objective experiments, especially concerning the Dirac separator, performed the subjective listening tests, and wrote substantial parts of the paper. E.P. proposed and developed the ideas of the source-joint Bayesian separator and the sub-FAD metric, partly formalized the proof of the Dirac separator, contributed to the code and the objective experiments, especially concerning source imputation, and wrote substantial parts of the paper. M.M. proposed the idea of using the source-joint model for music (and accompaniment) generation, proposed using the correction steps, and contributed to the objective experiments.

†Shared last authorship.

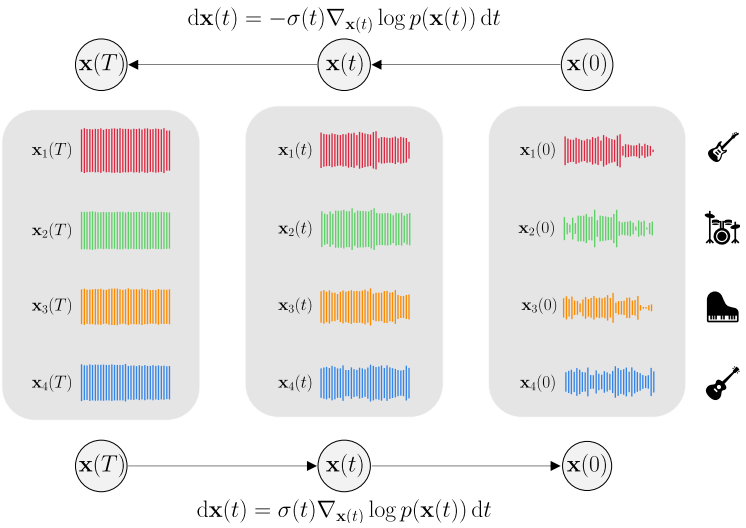

Figure 1: Proposed method. We leverage a forward Gaussian process (right-to-left) to learn the score over contextual sets (the large boxes) of instrumental sources (the waveforms) across different time steps t. During inference, the process is reversed (left-to-right), letting us perform the tasks of total generation, partial generation, or source separation (Figure 2).

harmonizes with the melody of the guitar. Mathematically, this fact can be expressed by saying that the joint distribution of the sources $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ does *not* factorize into the product of individual source distributions $\{p_n(\mathbf{x}_n)\}_{n=1,\ldots,N}$. Knowing the joint $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ implies knowing the distribution over the mixtures $p(\mathbf{y})$ since the latter can be obtained through the sum. The converse is more difficult mathematically, being an inverse problem.

Nevertheless, humans have developed the ability to process multiple sound sources simultaneously in terms of synthesis (i.e., musical composition or generation) and analysis (i.e., source separation). More specifically, composers can invent multiple sources $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ that sum to a consistent mixture $\mathbf{y}$ and, extract information about the individual sources $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ from a mixture $\mathbf{y}$.

This ability to compose and decompose sound is crucial for a generative music model. A model designed to assist in music composition should be capable of isolating individual sources within a mixture and allow for independent operation on each source. Such a capability would give the composer maximum control over what to modify and retain in a composition. Therefore, we argue that compositional (waveform) music generation is highly connected to music source separation.

To our knowledge, no model in deep learning literature can perform both tasks simultaneously. Models designed for the generation task directly learn the distribution $p(\mathbf{y})$ over mixtures, collapsing the information needed for the separation task. In this case, we have accurate mixture modeling but no information about the individual sources. It is worth noting that approaches that model the distribution of mixtures conditioning on textual data (Schneider et al., 2023; Agostinelli et al., 2023) face the same limitations. Conversely, models for source separation (Défossez et al., 2019) either target $p(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \mathbf{y})$, conditioning on the mixture, or learn a single model $p_n(\mathbf{x}_n)$ for each source distribution (e.g., in a weakly-supervised manner) and condition on the mixture during inference (Jayaram & Thickstun, 2020; Postolache et al., 2023a). In both cases, generating mixtures is impossible. In the first case, the model inputs a mixture, which hinders the possibility of unconditional modeling, not having direct access to $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ (or equivalently to $p(\mathbf{y})$). In the second case, while we can accurately model each source independently, all essential information about their interdependence is lost, preventing the possibility of generating coherent mixtures.

Contribution. Our contribution is three-fold. *(i)* First, we bridge the gap between source separation and music generation by learning $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$, the joint (prior) distribution of contextual sources (i.e., those belonging to the same song). For this purpose, we use the denoising score-matching framework to train a *Multi-Source Diffusion Model (MSDM)*. We can perform both source separation and music generation during inference by training this single model. Specifically, generation is achieved by sampling from the prior, while separation is carried out by conditioning the prior on

the mixture and then sampling from the resulting posterior distribution. *(ii)* This new formulation opens the doors to novel tasks in the generative domain, such as *source imputation*, where we create accompaniments by generating a subset of the sources given the others (e.g., play a piano track that goes well with the drums). *(iii)* Lastly, to obtain competitive results on source separation with respect to state-of-the-art regressor models (Manilow et al., 2022) on the Slakh2100 (Manilow et al., 2019) dataset, we propose a new procedure for computing the posterior score based on *Dirac delta functions*, exploiting the functional relationship between the sources and the mixture.

2 Related Work

2.1 Generative Models for Audio

Deep generative models for audio learn, directly or implicitly, the distribution of mixtures, represented in our notation by $p(\mathbf{y})$, possibly conditioning on additional data such as text. Various general-purpose generative models, such as autoregressive models, GANs (Donahue et al., 2019), and diffusion models, have been adapted for use in the audio field.

Autoregressive models are well-established in audio modeling (van den Oord et al., 2016). Jukebox (Dhariwal et al., 2020) proposed to model musical tracks with Scalable Transformers (Vaswani et al., 2017) on hierarchical discrete representations obtained through VQ-VAEs (van den Oord et al., 2017). Furthermore, using a lyrics conditioner, this method generated tracks with vocals following the text. However, while Jukebox could model longer sequences in latent space, the audio output suffered from quantization artifacts. Newer latent autoregressive models (Borsos et al., 2023; Kreuk et al., 2023) can handle extended contexts, more coherent generations and, by incorporating residual quantization (Zeghidour et al., 2021), output more naturally sounding samples. State-of-the-art latent autoregressive models for music, such as MusicLM (Agostinelli et al., 2023), can guide generation by conditioning on textual embeddings obtained via large-scale contrastive pre-training (Manco et al., 2022; Huang et al., 2022). MusicLM can also input a melody and condition on text for style transfer. A concurrent work, SingSong (Donahue et al., 2023), introduces vocal-to-mixture accompaniment generation. Our accompaniment generation procedure differs from the latter since we perform generation at the stem level in a composable way, while the former outputs a single mixture.

DiffWave (Kong et al., 2021) and WaveGrad (Chen et al., 2021) were the first diffusion (score) based generative models in audio, tackling speech synthesis. Many subsequent models followed these preliminary works, mainly conditioned to solve particular tasks such as speech enhancement (Lu et al., 2021; Serrà et al., 2022; Sawata et al., 2023; Saito et al., 2023), audio upsampling (Lee & Han, 2021; Yu et al., 2023), MIDI-to-waveform (Mittal et al., 2021; Hawthorne et al., 2022), or spectrogram-to-MIDI generation (Cheuk et al., 2023). The first work in source-specific generation with diffusion models is CRASH (Rouard & Hadjeres, 2021). (Yang et al., 2023; Pascual et al., 2023; Liu et al., 2023) proposed text-conditioned diffusion models to generate general sounds, not focusing on restricted classes such as speech or music. Closer to our work, diffusion models targeting the musical domain are Riffusion (Forsgren & Martiros, 2022) and Moûsai (Schneider et al., 2023). Riffusion fine-tunes Stable Diffusion (Rombach et al., 2022), a large pre-trained text-conditioned vision diffusion model, over STFT magnitude spectrograms. Moûsai performs generation in a latent domain, resulting in context lengths that surpass the minute. Our score network follows the design of the U-Net proposed in Moûsai, albeit using the waveform data representation.

2.2 Audio Source Separation

Existing audio source separation models can be broadly classified into deterministic and generative. Deterministic source separators are parametric models that input the mixtures and systematically extract one or all sources. These models are typically trained with a regression loss (Gusó et al., 2022) on the estimated signal represented as waveform (Lluís et al., 2019; Luo & Mesgarani, 2019; Défossez et al., 2019), STFT (Takahashi et al., 2018; Choi et al., 2021), or both (Défossez, 2021). On the other hand, generative source separation models based on (independent) Bayesian inference learn a prior model for each source, thus targeting the distributions $\{p_n(\mathbf{x}_n)\}_{n=1,\dots,N}$. The mixture is observed only during inference, where a likelihood function connects it to its constituent sources. The literature has explored different priors, such as GANs (Subakan & Smaragdis, 2018; Kong et al.,

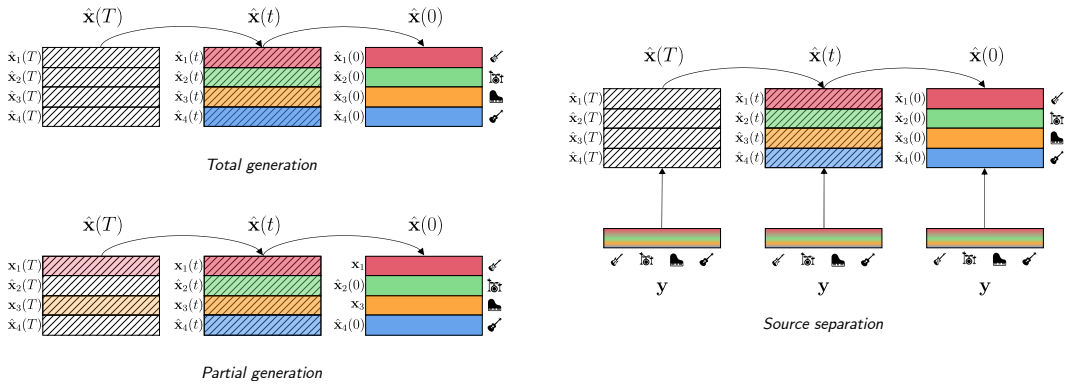

Figure 2: Inference tasks with MSDM. Oblique lines represent the presence of noise in the signal, decreasing from left to right, with the highest noise level at time T when we start the sampling procedure. *Top-left:* We generate all stems in a mixture, obtaining a total generation. *Bottom-left:* We perform partial generation (source imputation) by fixing the sources $\mathbf{x}_1$ (Bass) and $\mathbf{x}_3$ (Piano) and generating the other two sources $\hat{\mathbf{x}}_2(0)$ (Drums) and $\hat{\mathbf{x}}_4(0)$ (Guitar). We denote with $\mathbf{x}_1(t)$ and $\mathbf{x}_3(t)$, the noisy stems obtained from $\mathbf{x}_1$ and $\mathbf{x}_3$ via the perturbation kernel in Eq. (1). *Right:* We perform source separation by conditioning the prior with a mixture $\mathbf{y}$, following Algorithm 1.

2019; Narayanaswamy et al., 2020), normalizing flows (Jayaram & Thickstun, 2020; Zhu et al., 2022), and autoregressive models (Jayaram & Thickstun, 2021; Postolache et al., 2023a).

The separation method closer to ours is NCSN-BASIS (Jayaram & Thickstun, 2020). This method was proposed for image source separation, using Langevin Dynamics to separate the mixtures with an NCSN score-based model. It employs a Gaussian likelihood function during inference, which, as we demonstrate experimentally, is sub-optimal compared to our novel Dirac-based likelihood function. The main difference between our method and other generative source separation methods (including NCSN-BASIS) is the modeling of the full joint distribution. As such, we can perform source separation and synthesize mixtures or subsets of stems with a single model.

Contextual information between sources is explicitly modeled in (Manilow et al., 2022) and (Postolache et al., 2023b). The first work models the relationship between sources by training an orderless NADE estimator, which predicts a subset of the sources while conditioning on the input mixture and the remaining sources. The subsequent study achieves universal source separation (Kavalerov et al., 2019; Wisdom et al., 2020) through adversarial training, utilizing a context-based discriminator to model the relationship between sources. Both methods are deterministic and conditioned on the mixtures architecturally. The same architectural limitation is present in diffusion-based (Scheibler et al., 2023; Lutati et al., 2023) or diffusion-inspired (Plaja-Roglans et al., 2022) conditional approaches. Our method sets itself apart as it proposes a model not constrained architecturally by a mixture conditioner, so we can also perform unconditional generation.

3 Background

The foundation of our model lies in estimating the joint distribution of the sources $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$. Our approach is generative because we model an unconditional distribution (the prior). The different tasks are then solved at inference time, exploiting the prior.

We employ a diffusion-based (Sohl-Dickstein et al., 2015; Ho et al., 2020) generative model trained via denoising score-matching (Song & Ermon, 2019) to learn the prior. Specifically, we present our formalism by utilizing the notation and assumptions established in (Karras et al., 2022). The central idea of score-matching (Hyvärinen, 2005; Kingma & LeCun, 2010; Vincent, 2011) is to approximate the "score" function of the target distribution $p(\mathbf{x})$, namely $\nabla_{\mathbf{x}} \log p(\mathbf{x})$, rather than the distribution itself. To effectively approximate the score in sparse data regions, denoising diffusion methods introduce controlled noise to the data and learn to remove it. Formally, the data distribution is perturbed with a Gaussian perturbation kernel:

$$p(\mathbf{x}(t) \mid \mathbf{x}(0)) = \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0), \sigma^2(t)\mathbf{I}) \,, \tag{1}$$

where the parameter $\sigma(t)$ regulates the degree of noise added to the data. Following the authors in (Karras et al., 2022), we consider an optimal schedule given by $\sigma(t) = t$. With that choice of $\sigma(t)$, the forward evolution of a data point $\mathbf{x}(t)$ in time is described by a probability flow ODE (Song et al., 2021):

$$\mathrm{d}\mathbf{x}(t) = -\sigma(t)\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t))\,\mathrm{d}t\,. \tag{2}$$

For $t = T >> 0$, a data point $\mathbf{x}(T)$ is approximately distributed according to a Gaussian distribution $\mathcal{N}(\mathbf{x}(t); \mathbf{0}, \sigma^2(T)\mathbf{I})$, from which sampling is straightforward. Eq. (2) can be inverted in time, resulting in the following backward ODE that describes the denoising process:

$$\mathrm{d}\mathbf{x}(t) = \sigma(t)\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t))\,\mathrm{d}t\,. \tag{3}$$

Sampling can be performed integrating Eq. (3) with a standard ODE solver, starting from an initial (noisy) sample drawn from $\mathcal{N}(\mathbf{x}(t); \mathbf{0}, \sigma^2(T)\mathbf{I})$. The score function, is approximated by a neural network $S^\theta(\mathbf{x}(t), \sigma(t))$, minimizing the following score-matching loss:

$$\mathbb{E}_{t\sim\mathcal{U}([0,T])}\mathbb{E}_{\mathbf{x}(0)\sim p(\mathbf{x}(0))}\mathbb{E}_{\mathbf{x}(t)\sim p(\mathbf{x}(t)|\mathbf{x}(0))} \left\| S^\theta(\mathbf{x}(t), \sigma(t)) - \nabla_{\mathbf{x}(t)} \log p\left(\mathbf{x}(t) \mid \mathbf{x}(0)\right) \right\|_2^2 .$$

By expanding $p(\mathbf{x}(t) \mid \mathbf{x}(0))$ with Eq. (1), the score-matching loss simplifies to:

$$\mathbb{E}_{t\sim\mathcal{U}([0,T])}\mathbb{E}_{\mathbf{x}(0)\sim p(\mathbf{x}(0))}\mathbb{E}_{\epsilon\sim\mathcal{N}(\mathbf{0},\sigma^2(t)\mathbf{I})} \left\| D^\theta(\mathbf{x}(0) + \epsilon, \sigma(t)) - \mathbf{x}(0) \right\|_2^2 ,$$

where we define $S^\theta(\mathbf{x}(t), \sigma(t)) =: (D^\theta(\mathbf{x}(t), \sigma(t)) - \mathbf{x}(t))/\sigma^2(t)$.

4 METHOD

4.1 MULTI-SOURCE AUDIO DIFFUSION MODELS

In our setup, we have N distinct source waveforms $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with $\mathbf{x}_n \in \mathbb{R}^D$ for each n. The sources coherently sum to a mixture $\mathbf{y} = \sum_{n=1}^{N} \mathbf{x}_n$. We also use the aggregated form $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N) \in \mathbb{R}^{N\times D}$. In this setting, multiple tasks can be performed: one may generate a consistent mixture $\mathbf{y}$ or separate the individual sources $\mathbf{x}$ from a given mixture $\mathbf{y}$. We refer to the first task as *generation* and the second as *source separation*. A subset of sources can also be fixed in the generation task, and the others can be generated consistently. We call this task *partial generation* or *source imputation*. Our key contribution is the ability to perform all these tasks simultaneously by training a single multi-source diffusion model (MSDM), capturing the prior $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$. The model, illustrated in Figure 1, approximates the noisy score function:

$$\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t)) = \nabla_{(\mathbf{x}_1(t),\ldots,\mathbf{x}_N(t))} \log p(\mathbf{x}_1(t), \ldots, \mathbf{x}_N(t))\,,$$

with a neural network:

$$S^\theta(\mathbf{x}(t), \sigma(t)) : \mathbb{R}^{N\times D} \times \mathbb{R} \to \mathbb{R}^{N\times D}\,, \tag{4}$$

where $\mathbf{x}(t) = (\mathbf{x}_1(t), \ldots, \mathbf{x}_N(t))$ denotes the sources perturbed with the Gaussian kernel in Eq. (1). We describe the three tasks (illustrated in Figure 2) using the prior distribution:

- *Total Generation.* This task requires generating a plausible mixture $\mathbf{y}$. It can be achieved by sampling the sources $\{\mathbf{x}_1, ..., \mathbf{x}_N\}$ from the prior distribution and summing them to obtain the mixture $\mathbf{y}$.
- *Partial Generation.* Given a subset of sources, this task requires generating a plausible accompaniment. We define the subset of fixed sources as $\mathbf{x}_\mathcal{I}$ and generate the remaining sources $\mathbf{x}_{\overline{\mathcal{I}}}$ by sampling from the conditional distribution $p(\mathbf{x}_{\overline{\mathcal{I}}} \mid \mathbf{x}_\mathcal{I})$.
- *Source Separation.* Given a mixture $\mathbf{y}$, this task requires isolating the individual sources that compose it. It can be achieved by sampling from the posterior distribution $p(\mathbf{x} \mid \mathbf{y})$.

4.2 INFERENCE

The three tasks of our method are solved during inference by discretizing the backward Eq. (3). Although different tasks require distinct score functions, they all originate directly from the prior score function in Eq. (4). We analyze each of these score functions in detail. For more details on the discretization method, refer to Section C.3.

Algorithm 1 'MSDM Dirac' sampler for source separation.

Require: I number of discretization steps for the ODE, R number of corrector steps, $\{\sigma_i\}_{i\in\{0,\dots,I\}}$ noise schedule, S_{churn}

Initialize $\hat{\mathbf{x}} \sim \mathcal{N}(0, \sigma_I^2\mathbf{I})$
$\alpha \leftarrow \min(S_{\text{churn}}/I, \sqrt{2}-1)$
for $i \leftarrow I$ **to** 1 **do**
  for $r \leftarrow R$ **to** 0 **do**
    $\hat{\sigma} \leftarrow \sigma_i \cdot (\alpha+1)$
    $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
    $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \sqrt{\hat{\sigma}^2 - \sigma_i^2}\epsilon$
    $\mathbf{z} \leftarrow [\hat{\mathbf{x}}_{1:N-1}, \mathbf{y} - \sum_{n=1}^{N-1}\hat{\mathbf{x}}_n]$
    for $n \leftarrow 1$ **to** $N-1$ **do**
      $\mathbf{g}_n \leftarrow S_n^\theta(\mathbf{z}, \hat{\sigma}) - S_N^\theta(\mathbf{z}, \hat{\sigma})$
    end for
    $\mathbf{g} \leftarrow [\mathbf{g}_1, \dots, \mathbf{g}_{N-1}]$
    $\hat{\mathbf{x}}_{1:N-1} \leftarrow \hat{\mathbf{x}}_{1:N-1} + (\sigma_{i-1} - \hat{\sigma})\mathbf{g}$
    $\hat{\mathbf{x}} \leftarrow [\hat{\mathbf{x}}_{1:N-1}, \mathbf{y} - \sum_{n=1}^{N-1}\hat{\mathbf{x}}_n]$
    if $r > 0$ **then**
      $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
      $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\epsilon$
    end if
  end for
end for
return $\hat{\mathbf{x}}$

4.2.1 Total Generation

The total generation task is performed by sampling from Eq. (3) using the score function in Eq. (4). The mixture is then obtained by summing over all the generated sources.

4.2.2 Partial Generation

In the partial generation task, we fix a subset of source indices $\mathcal{I} \subset \{1, \dots, N\}$ and the relative sources $\mathbf{x}_\mathcal{I} := \{\mathbf{x}_n\}_{n\in\mathcal{I}}$. The goal is to generate the remaining sources $\mathbf{x}_{\overline{\mathcal{I}}} := \{\mathbf{x}_n\}_{n\in\overline{\mathcal{I}}}$ consistently, where $\overline{\mathcal{I}} = \{1, \dots, N\} - \mathcal{I}$. To do so, we estimate the gradient of the conditional distribution:

$$\nabla_{\mathbf{x}_{\overline{\mathcal{I}}}(t)} \log p(\mathbf{x}_{\overline{\mathcal{I}}}(t) \mid \mathbf{x}_\mathcal{I}(t)). \tag{5}$$

This falls into the setting of imputation or, as it is more widely known in the image domain, inpainting. We approach imputation using the method in (Song et al., 2021). The gradient in Eq. (5) is approximated as follows:

$$\nabla_{\mathbf{x}_{\overline{\mathcal{I}}}(t)} \log p([\mathbf{x}_{\overline{\mathcal{I}}}(t), \hat{\mathbf{x}}_\mathcal{I}(t)])\,,$$

where $\hat{\mathbf{x}}_\mathcal{I}$ is a sample from the forward process: $\hat{\mathbf{x}}_\mathcal{I}(t) \sim \mathcal{N}(\mathbf{x}_\mathcal{I}(t); \mathbf{x}_\mathcal{I}(0), \sigma(t)^2\mathbf{I})$. The square bracket operator denotes concatenation. Approximating the score function, we write:

$$\nabla_{\mathbf{x}_{\overline{\mathcal{I}}}(t)} \log p(\mathbf{x}_{\overline{\mathcal{I}}}(t) \mid \mathbf{x}_\mathcal{I}(t)) \approx S_{\overline{\mathcal{I}}}^\theta([\mathbf{x}_{\overline{\mathcal{I}}}(t), \hat{\mathbf{x}}_\mathcal{I}(t)], \sigma(t))\,,$$

where $S_{\overline{\mathcal{I}}}^\theta$ denotes the entries of the score network corresponding to the sources indexed by $\overline{\mathcal{I}}$.

4.2.3 Source Separation

We view source separation as a specific instance of conditional generation, where we condition the generation process on the given mixture $\mathbf{y} = \mathbf{y}(0)$. This requires computing the score function of the posterior distribution:

$$\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t) \mid \mathbf{y}(0))\,. \tag{6}$$

Standard methods for implementing conditional generation for diffusion models involve directly estimating the posterior score in Eq. (6) at training time (i.e., Classifier Free Guidance (Ho & Salimans, 2021)) or estimating the likelihood function $p(\mathbf{y}(0) \mid \mathbf{x}(t))$ and using the Bayes formula

to derive the posterior. The second approach typically involves training a separate model, often a classifier, for the likelihood score (i.e., Classifier Guidance (Dhariwal & Nichol, 2021)).

In diffusion-based generative source separation, learning a likelihood model is typically unnecessary because the relationship between $\mathbf{x}(t)$ and $\mathbf{y}(t)$ is represented by a simple function, namely the sum. A natural approach is to model the likelihood function based on such functional dependency. This is the approach taken by (Jayaram & Thickstun, 2020), where they use a Gaussian likelihood function:

$$p(\mathbf{y}(t) \mid \mathbf{x}(t)) = \mathcal{N}(\mathbf{y}(t) \mid \sum_{n=1}^{N} \mathbf{x}_n(t), \gamma^2(t)\mathbf{I}), \tag{7}$$

with the standard deviation given by a hyperparameter $\gamma(t)$. The authors argue that aligning the $\gamma(t)$ value to be proportionate to $\sigma(t)$ optimizes the outcomes of their NCSN-BASIS separator.

We present a novel approximation of the posterior score function in Eq. (6) by modeling $p(\mathbf{y}(t) \mid \mathbf{x}(t))$ as a Dirac delta function centered in $\sum_{n=1}^{N} \mathbf{x}_n(t)$:

$$p(\mathbf{y}(t) \mid \mathbf{x}(t)) = \mathbb{1}_{\mathbf{y}(t)=\sum_{n=1}^{N} \mathbf{x}_n(t)} \,. \tag{8}$$

The complete derivation can be found in Appendix A, and we present only the final formulation, which we call 'MSDM Dirac'. The method constrains a source, without loss of generality $\mathbf{x}_N$, by setting $\mathbf{x}_N(t) = \mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)$ and estimates:

$$\begin{aligned}\nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx\; & S_m^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t), \mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)), \sigma(t)) \\ & - S_N^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t), \mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)), \sigma(t)) \,,\end{aligned}$$

where $1 \leq m \leq N-1$ and S_m^θ, S_N^θ denote the entries of the score network corresponding to the m-th and N-th sources. Our approach models the limiting case wherein $\gamma(t) \to 0$ in the Gaussian likelihood function. This represents a scenario where the functional dependence between $\mathbf{x}(t)$ and $\mathbf{y}(t)$ becomes increasingly tight, thereby sharpening the conditioning on the given mixture during the generation process. The pseudo-code for the 'MSDM Dirac' source separation sampler, using the Euler ODE integrator of (Karras et al., 2022), is provided in Algorithm 1. The Euler ODE discretization logic uses the S_{churn} mechanism of (Karras et al., 2022) and optional correction steps (Song et al., 2021) (see Section C.3 for more details).

The separation procedure can be additionally employed in the weakly-supervised source separation scenario, typically encountered in generative source separation (Jayaram & Thickstun, 2020; Zhu et al., 2022; Postolache et al., 2023a). This scenario pertains to cases where we know that specific audio data belongs to a particular instrument class while not having access to sets of sources sharing a context. To adapt to this scenario, we assume independence between sources $p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \prod_{n=1}^{N} p_n(\mathbf{x}_n)$ and train a separate model for each source class. We call the resulting model 'Independent Source Diffusion Model with Dirac Likelihood' ('ISDM Dirac'). We derive its formula together with formulas for the Gaussian versions 'MSDM Gaussian' and 'ISDM Gaussian' in Appendix B.

5 Experimental Results

We experiment on Slakh2100 (Manilow et al., 2019), a standard dataset for music source separation. We chose Slakh2100 because it has a significantly larger quantity of data (145h) than other multi-source waveform datasets like MUSDB18-HQ (Rafii et al., 2019) (10h). The amount of data plays a decisive role in determining the quality of a generative model, making Slakh2100 a preferable choice. Nevertheless, in Appendix E we conduct a study of data efficiency on MUSDB18-HQ. Details on datasets, architecture, training, and sampling are provided in Appendix C.

5.1 Music Generation

The performance of MSDM on the generative tasks is tested through subjective and objective evaluation.

Table 1: Comparison between total generation capabilities of MSDM (Slakh2100) and an equivalent architecture trained on Slakh2100 mixtures. Both subjective (quality and coherence, higher is better) and objective (FAD, lower is better) evaluations are shown. The quality and coherence columns refer to the average scores of the listening tests, with respective variances.

Model	**FAD** $\downarrow$	**Quality** $\uparrow$	**Coherence** $\uparrow$
MSDM	6.55	6.51 ± 2.19	6.35 ± 2.36
Mixture Model	6.67	6.15 ± 2.47	5.67 ± 2.60

Table 2: Quantitative and qualitative results for the partial generation task on Slakh2100. We use both subjective (quality and density, higher is better) and objective (sub-FAD, lower is better) evaluation metrics. The sub-FAD metric is reported for all combinations of generated sources (**B**: Bass, **D**: Drums, **G**: Guitar, **P**: Piano). The quality and density columns refer to the average scores of the listening tests, with respective variances.

Slakh2100	**B**	**D**	**G**	**P**	**BD**	**BG**	**BP**	**DG**	**DP**	**GP**	**BDG**	**BDP**	**BGP**	**DGP**	**Quality** $\uparrow$	**Density** $\uparrow$
MSDM	0.45	1.09	0.11	0.76	2.09	1.00	2.32	1.45	1.82	1.65	2.93	3.30	4.90	3.10	6.3 ± 2.7	6.1 ± 2.6

Subjective evaluation is done through listening tests, whose form format is reported in Appendix F. Concisely, we produce two forms, one for total generation and one for partial generation. In the first, subjects are asked to rate, from 1 to 10, the *quality* and instrument *coherence* (i.e., how the instruments sound plausible together) of 30 generated chunks, of which 15 are generated by MSDM and 15 by a model trained on mixtures (using the same diffusion architecture as MSDM). In the second one, knowing the fixed instruments, subjects are asked to rate, from 1 to 10, the *quality* and the *density* of the generated accompaniment. Namely, 'quality' tests how the full chunk sounds plausible with respect to the ground truth data, and 'density' tests how much the generated instruments are present in the chunk. We also provide examples of music and accompaniment generation[1].

For the objective evaluation of the generative tasks, we generalize the FAD protocol in Donahue et al. (2023) to our total and partial generation tasks with more than one source. Given D_{real} a dataset of ground truth mixtures chunks and $\mathcal{I}$ a set indexing conditioning sources ($\emptyset$ for total generation), we build a dataset D_{gen} whose elements are the sum between conditioning sources (indexed by $\mathcal{I}$) and the respective generated sources. We define the *sub-FAD* as $FAD(D_{\text{real}}, D_{\text{gen}})$.

Results for total and partial generations are reported in Tables 1 and 2 respectively, both for subjective and objective evaluations. Results in Table 1 show a minimal difference between the model trained on mixtures and MSDM. This suggests that, given the same dataset and architecture, the generative power of MSDM is the same as the model trained on mixtures while being able to perform separation and partial generation. Table 2 shows via the subjective results that the task of partial generation can be performed with non-trivial quality. Our method being the first able to generate any combination of partial sources, does not have a competitor baseline for the objective metrics. We thus report the sub-FAD results of our method as baseline metrics for future research.

5.2 Source Separation

In order to evaluate source separation, we use the scale-invariant SDR improvement (SI-SDR$_\text{I}$) metric (Roux et al., 2019). The SI-SDR between a ground-truth source $\mathbf{x}_n$ and an estimate $\hat{\mathbf{x}}_n$ is defined as:

$$\text{SI-SDR}(\mathbf{x}_n, \hat{\mathbf{x}}_n) = 10 \log_{10} \frac{\|\alpha \mathbf{x}_n\|^2 + \epsilon}{\|\alpha \mathbf{x}_n - \hat{\mathbf{x}}_n\|^2 + \epsilon},$$

where $\alpha = \frac{\mathbf{x}_n^\top \hat{\mathbf{x}}_n + \epsilon}{\|\mathbf{x}_n\|^2 + \epsilon}$ and $\epsilon = 10^{-8}$. The improvement with respect to the mixture baseline is defined as SI-SDR$_\text{I}$ = SI-SDR$(\mathbf{x}_n, \hat{\mathbf{x}}_n)$ − SI-SDR$(\mathbf{x}_n, \mathbf{y})$.

On Slakh, we compare our supervised MSDM and weakly-supervised MSDM with the 'Demucs' (Défossez et al., 2019) and 'Demucs + Gibbs (512 steps)' regressor baselines from (Manilow et al., 2022), the state-of-the-art for supervised music source separation on Slakh2100, aligning with the

[1] https://gladia-research-group.github.io/multi-source-diffusion-models/

Table 3: Quantitative results for source separation on the Slakh2100 test set. We use the SI-SDR$_I$ as our evaluation metric (dB – higher is better). We present both the supervised ('MSDM Dirac', 'MSDM Gaussian') and weakly-supervised ('ISDM Dirac', 'ISDM Gaussian') separators and specify if a correction step is used. 'All' reports the average over the four stems.

Model	**Bass**	**Drums**	**Guitar**	**Piano**	**All**
Demucs (Défossez et al., 2019; Manilow et al., 2022)	15.77	19.44	15.30	13.92	16.11
Demucs + Gibbs (512 steps) (Manilow et al., 2022)	17.16	19.61	**17.82**	**16.32**	**17.73**
Dirac Likelihood					
ISDM	18.44	20.19	13.34	13.25	16.30
ISDM (correction)	**19.36**	**20.90**	14.70	14.13	17.27
MSDM	16.21	17.47	12.71	13.29	14.92
MSDM (correction)	17.12	18.68	15.38	14.73	16.48
Gaussian Likelihood (Jayaram & Thickstun, 2020)					
ISDM	13.48	18.09	11.93	11.17	13.67
ISDM (correction)	14.27	19.10	12.74	12.20	14.58
MSDM	12.53	16.82	12.98	9.29	12.90
MSDM (correction)	13.93	17.92	14.19	12.11	14.54

evaluation procedure of (Manilow et al., 2022). We evaluate over the test set of Slakh2100, using chunks of 4 seconds in length (with an overlap of two seconds) and filtering out silent chunks and chunks consisting of only one source, given the poor performance of SI-SDR$_I$ on such segments. We report results comparing our Dirac score posterior with the Gaussian score posterior of (Jayaram & Thickstun, 2020), using the best parameters of the ablations in Appendix D and 150 inference steps.

Results are reported in Table 3 and show that: *(i)* The Dirac likelihood improves overall results, even outperforming the state of the art when applied to ISDM on Bass and Drums *(ii)* adding a correction step is beneficial *(iii)* MSDM with Dirac likelihood and one step of correction gives results comparable with the state of the art and superior to standard Demucs overall. We stress again that, while the baselines can perform the separation task alone, MSDM can also perform generative tasks.

6 Conclusions

We have presented a general method, based on denoising score-matching, for source separation, mixture generation, and accompaniment generation in the musical domain. Our approach utilizes a single neural network trained once, with tasks differentiated during inference. Moreover, we have defined a new sampling method for source separation. We quantitatively tested the model on source separation, obtaining results comparable to state-of-the-art regressor models. We qualitatively and quantitatively tested the model on total and partial generation. Our model's ability to handle both total and partial generation and source separation positions it as a significant step toward the development of general audio models. This flexibility paves the way for more advanced music composition tools, where users can easily control and manipulate individual sources within a mixture.

6.1 Limitations and Future Work

The amount of available contextual data constrains the performance of our model. To address this, pre-separating mixtures and training on the separations, as demonstrated in (Donahue et al., 2023), may prove beneficial. Additionally, it would be intriguing to explore the possibility of extending our method to situations where the sub-signals are not related by addition but rather by a known but different function. Finally, future work could adapt the model to jointly model MIDI information (for example, extracted from sources (Lin et al., 2021)) for further control.

ACKNOWLEDGEMENTS

The authors were partially supported by the ERC grant no. 802554 (SPECGEO), PRIN 2020 project no. 2020TA3K9N (LEGO.AI), PRIN 2022 project no. 2022AL45R2 (EYE-FI.AI, CUP H53D2300350-0001), and PNRR MUR project no. PE0000013-FAIR.

REFERENCES

Andrea Agostinelli, Timo I Denk, Zalán Borsos, Jesse Engel, Mauro Verzetti, Antoine Caillon, Qingqing Huang, Aren Jansen, Adam Roberts, Marco Tagliasacchi, et al. Musiclm: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=NsMLjcFaO8O.

Kin Wai Cheuk, Ryosuke Sawata, Toshimitsu Uesaka, Naoki Murata, Naoya Takahashi, Shusuke Takahashi, Dorien Herremans, and Yuki Mitsufuji. Diffroll: Diffusion-based generative music transcription with unsupervised pretraining capability. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Woosung Choi, Minseok Kim, Jaehwa Chung, and Soonyoung Jung. Lasaft: Latent source attentive frequency transformation for conditioned source separation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 171–175. IEEE, 2021.

Alexandre Défossez. Hybrid spectrogram and waveform source separation. In *Proceedings of the ISMIR 2021 Workshop on Music Source Separation*, 2021.

Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. Music source separation in the waveform domain. *arXiv preprint arXiv:1911.13254*, 2019.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *International Conference on Learning Representations*, 2019.

Chris Donahue, Antoine Caillon, Adam Roberts, Ethan Manilow, Philippe Esling, Andrea Agostinelli, Mauro Verzetti, Ian Simon, Olivier Pietquin, Neil Zeghidour, et al. Singsong: Generating musical accompaniments from singing. *arXiv preprint arXiv:2301.12662*, 2023. URL https://arxiv.org/abs/2301.12662.

Seth* Forsgren and Hayk* Martiros. Riffusion - Stable diffusion for real-time music generation, 2022. URL https://riffusion.com/about.

Enric Gusó, Jordi Pons, Santiago Pascual, and Joan Serrà. On loss functions and evaluation metrics for music source separation. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 306–310. IEEE, 2022.

Curtis Hawthorne, Ian Simon, Adam Roberts, Neil Zeghidour, Josh Gardner, Ethan Manilow, and Jesse Engel. Multi-instrument music synthesis with spectrogram diffusion. In *International Society for Music Information Retrieval Conference*, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pp. 6840–6851, 2020.

Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel P. W. Ellis. Mulan: A joint embedding of music audio and natural language. In *International Society for Music Information Retrieval Conference*, 2022.

Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(24):695–709, 2005. URL http://jmlr.org/papers/v6/hyvarinen05a.html.

Vivek Jayaram and John Thickstun. Source separation with deep generative priors. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4724–4735, 2020.

Vivek Jayaram and John Thickstun. Parallel and flexible sampling from autoregressive models via langevin dynamics. In *Proc. ICML*, pp. 4807–4818. PMLR, 2021.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Yitzhak Katznelson. *An Introduction to Harmonic Analysis*. Cambridge Mathematical Library. Cambridge University Press, 3 edition, 2004. doi: 10.1017/CBO9781139165372.

Ilya Kavalerov, Scott Wisdom, Hakan Erdogan, Brian Patton, Kevin Wilson, Jonathan Le Roux, and John R Hershey. Universal sound separation. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 175–179. IEEE, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Durk P Kingma and Yann LeCun. Regularized estimation of image statistics by score matching. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010.

Qiuqiang Kong, Yong Xu, Wenwu Wang, Philip J. B. Jackson, and Mark D. Plumbley. Single-channel signal separation and deconvolution with generative adversarial networks. In *Proc. IJCAI*, pp. 2747–2753. AAAI Press, 2019. ISBN 9780999241141.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=a-xFK8Ymz5J.

Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=CYK7RfcOzQ4.

Junhyeok Lee and Seungu Han. NU-Wave: A Diffusion Probabilistic Model for Neural Audio Upsampling. In *Proc. Interspeech 2021*, pp. 1634–1638, 2021. doi: 10.21437/Interspeech.2021-36.

Liwei Lin, Qiuqiang Kong, Junyan Jiang, and Gus Xia. A unified model for zero-shot music source separation, transcription and synthesis. In *Proceedings of 22st International Conference on Music Information Retrieval, ISMIR*, 2021.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. Audioldm: Text-to-audio generation with latent diffusion models. *arXiv preprint arXiv:2301.12503*, 2023.

Francesc Lluís, Jordi Pons, and Xavier Serra. End-to-end music source separation: Is it possible in the waveform domain? In *INTERSPEECH*, pp. 4619–4623, 2019. URL https://doi.org/10.21437/Interspeech.2019-1177.

Yen-Ju Lu, Yu Tsao, and Shinji Watanabe. A study on speech enhancement based on diffusion probabilistic model. In *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 659–666. IEEE, 2021.

Yi Luo and Nima Mesgarani. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing*, 27(8): 1256–1266, 2019.

Shahar Lutati, Eliya Nachmani, and Lior Wolf. Separate and diffuse: Using a pretrained diffusion model for improving source separation. *arXiv preprint arXiv:2301.10752*, 2023.

Ilaria Manco, Emmanouil Benetos, Elio Quinton, and György Fazekas. Learning music audio representations via weak language supervision. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 456–460. IEEE, 2022.

Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux. Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity. In *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019.

Ethan Manilow, Curtis Hawthorne, Cheng-Zhi Anna Huang, Bryan Pardo, and Jesse Engel. Improving source separation by explicitly modeling dependencies between sources. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 291–295. IEEE, 2022.

Giorgia Minello, Alessandro Bicciato, Luca Rossi, Andrea Torsello, and Luca Cosmo. Graph generation via spectral diffusion. *arXiv preprint arXiv:2402.18974*, 2024.

Gautam Mittal, Jesse Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*, 2021. URL https://archives.ismir.net/ismir2021/paper/000058.pdf.

Vivek Narayanaswamy, Jayaraman J. Thiagarajan, Rushil Anirudh, and Andreas Spanias. Unsupervised audio source separation using generative priors. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2020-October:2657–2661, 2020. doi: 10.21437/Interspeech.2020-3115.

OpenAI. Gpt-4 technical report, 2023.

Santiago Pascual, Gautam Bhattacharya, Chunghsin Yeh, Jordi Pons, and Joan Serrà. Full-band general audio synthesis with score-based diffusion. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

Genís Plaja-Roglans, Miron Marius, and Xavier Serra. A diffusion-inspired training strategy for singing voice extraction in the waveform domain. In *Proc. of the 23rd Int. Society for Music Information Retrieval*, 2022.

Emilian Postolache, Giorgio Mariani, Michele Mancusi, Andrea Santilli, Luca Cosmo, and Emanuele Rodolà. Latent autoregressive source separation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(8):9444–9452, Jun. 2023a. doi: 10.1609/aaai.v37i8.26131. URL https://ojs.aaai.org/index.php/AAAI/article/view/26131.

Emilian Postolache, Jordi Pons, Santiago Pascual, and Joan Serrà. Adversarial permutation invariant training for universal sound separation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023b.

Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. Musdb18-hq - an uncompressed version of musdb18, August 2019. URL https://doi.org/10.5281/zenodo.3338373.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241. Springer, 2015.

Simon Rouard and Gaëtan Hadjeres. CRASH: raw audio score-based generative modeling for controllable high-resolution drum sound synthesis. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference, ISMIR 2021*, pp. 579–585, 2021.

Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey. Sdr – half-baked or well done? In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 626–630, 2019. doi: 10.1109/ICASSP.2019.8683855.

Koichi Saito, Naoki Murata, Toshimitsu Uesaka, Chieh-Hsin Lai, Yuhta Takida, Takao Fukui, and Yuki Mitsufuji. Unsupervised vocal dereverberation with diffusion-based generative models. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

Andrea Santilli, Silvio Severino, Emilian Postolache, Valentino Maiorca, Michele Mancusi, Riccardo Marin, and Emanuele Rodola. Accelerating transformer inference for translation via parallel decoding. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12336–12355, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.689. URL https://aclanthology.org/2023.acl-long.689.

Ryosuke Sawata, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Takashi Shibuya, Shusuke Takahashi, and Yuki Mitsufuji. Diffiner: A Versatile Diffusion-based Generative Refiner for Speech Enhancement. In *Proc. INTERSPEECH 2023*, pp. 3824–3828, 2023. doi: 10.21437/Interspeech.2023-1547.

Robin Scheibler, Youna Ji, Soo-Whan Chung, Jaeuk Byun, Soyeon Choe, and Min-Seok Choi. Diffusion-based generative speech source separation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

Flavio Schneider, Zhijing Jin, and Bernhard Schölkopf. Moûsai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023. URL https://arxiv.org/abs/2301.11757.

Joan Serrà, Santiago Pascual, Jordi Pons, R Oguz Araz, and Davide Scaini. Universal speech enhancement with score-based diffusion. *arXiv preprint arXiv:2206.03065*, 2022.

Jung-Eun Shin, Adam J Riesselman, Aaron W Kollasch, Conor McMahon, Elana Simon, Chris Sander, Aashish Manglik, Andrew C Kruse, and Debora S Marks. Protein design and variant prediction using autoregressive generative models. *Nature communications*, 12(1):2403, 2021. URL https://proceedings.neurips.cc/paper/2019/hash/3001ef257407d5a371a96dcd947c7d93-Abstract.html.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis R. Bach and David M. Blei (eds.), *Proceedings ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pp. 2256–2265. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/sohl-dickstein15.html.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 11895–11907, 2019.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

Y Cem Subakan and Paris Smaragdis. Generative adversarial source separation. In *Proc. ICASSP*, pp. 26–30. IEEE, 2018.

Naoya Takahashi, Nabarun Goswami, and Yuki Mitsufuji. Mmdenselstm: An efficient combination of convolutional and recurrent neural networks for audio source separation. In *Proc. IWAENC*, pp. 106–110, 2018. doi: 10.1109/IWAENC.2018.8521383.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Aäron 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. In *Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9)*, pp. 125, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

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

Tomer Weiss, Eduardo Mayo Yanes, Sabyasachi Chakraborty, Luca Cosmo, Alex M. Bronstein, and Renana Gershoni-Poranne. Guided diffusion for inverse molecular design. *Nature Computational Science*, 3(10):873–882, Oct 2023. ISSN 2662-8457. doi: 10.1038/s43588-023-00532-0. URL https://doi.org/10.1038/s43588-023-00532-0.

Scott Wisdom, Efthymios Tzinis, Hakan Erdogan, Ron Weiss, Kevin Wilson, and John Hershey. Unsupervised sound separation using mixture invariant training. *Advances in Neural Information Processing Systems*, 33:3846–3857, 2020.

Dongchao Yang, Jianwei Yu, Helin Wang, Wen Wang, Chao Weng, Yuexian Zou, and Dong Yu. Diffsound: Discrete diffusion model for text-to-sound generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

Chin-Yun Yu, Sung-Lin Yeh, György Fazekas, and Hao Tang. Conditioning and sampling in variational diffusion models for speech super-resolution. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2021.

Ge Zhu, Jordan Darefsky, Fei Jiang, Anton Selitskiy, and Zhiyao Duan. Music source separation with generative flow. *IEEE Signal Processing Letters*, 29:2288–2292, 2022. doi: 10.1109/LSP.2022.3219355.

A Derivation of MSDM Dirac Posterior Score for Source Separation

We prove the main result of Section 4.2.3. We condition the generative model over the mixture $\mathbf{y}(0) = \mathbf{y}$. As such, we compute the posterior:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) = \int_{\mathbf{y}(t)} p(\mathbf{x}(t), \mathbf{y}(t) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{y}(t) = \int_{\mathbf{y}(t)} p(\mathbf{x}(t) \mid \mathbf{y}(t), \mathbf{y}(0)) p(\mathbf{y}(t) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{y}(t)\,.$$

The first equality is given by marginalizing over $\mathbf{y}(t)$ and the second by the chain rule. Following Eq. (50) in Song et al. (2021), we can eliminate the dependency on $\mathbf{y}(0)$ from the first term, obtaining the approximation:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx \int_{\mathbf{y}(t)} p(\mathbf{x}(t) \mid \mathbf{y}(t)) p(\mathbf{y}(t) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{y}(t)\,. \tag{9}$$

We compute $p(\mathbf{y}(t) \mid \mathbf{y}(0))$, using the chain rule after marginalizing over $\mathbf{x}(0)$ and $\mathbf{x}(t)$:

$$p(\mathbf{y}(t) \mid \mathbf{y}(0)) = \int_{\mathbf{x}(0), \mathbf{x}(t)} p(\mathbf{y}(t), \mathbf{x}(t), \mathbf{x}(0) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{x}(0) \mathrm{d}\mathbf{x}(t)$$

$$= \int_{\mathbf{x}(0), \mathbf{x}(t)} p(\mathbf{y}(t) \mid \mathbf{x}(t), \mathbf{x}(0), \mathbf{y}(0)) p(\mathbf{x}(t) \mid \mathbf{x}(0), \mathbf{y}(0)) p(\mathbf{x}(0) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{x}(0) \mathrm{d}\mathbf{x}(t)\,.$$

By the Markov property of the forward diffusion process, $\mathbf{y}(t)$ is conditionally independent from $\mathbf{x}(0)$ given $\mathbf{x}(t)$ and we drop again the conditioning on $\mathbf{y}(0)$ from the first two terms, following Eq. (50) in Song et al. (2021). As such, we have:

$$p(\mathbf{y}(t) \mid \mathbf{y}(0)) \approx \int_{\mathbf{x}(0), \mathbf{x}(t)} p(\mathbf{x}(0) \mid \mathbf{y}(0)) p(\mathbf{x}(t) \mid \mathbf{x}(0)) p(\mathbf{y}(t) \mid \mathbf{x}(t)) \mathrm{d}\mathbf{x}(0) \mathrm{d}\mathbf{x}(t)\,. \tag{10}$$

We model the likelihood function $p(\mathbf{y}(t) \mid \mathbf{x}(t))$ with the Dirac delta function in Eq. (8). The posterior $p(\mathbf{x}(0) \mid \mathbf{y}(0))$ is obtained via Bayes theorem substituting the likelihood:

$$p(\mathbf{x}(0) \mid \mathbf{y}(0)) = \frac{p(\mathbf{x}(0)) \mathbb{1}_{\mathbf{y}(0) = \sum_{n=1}^{N} \mathbf{x}_n(0)}}{p(\mathbf{y}(0))} = \begin{cases} \frac{p(\mathbf{x}(0))}{p(\mathbf{y}(0))} & \text{if } \sum_{n=1}^{N} \mathbf{x}_n(0) = \mathbf{y}(0) \\ 0 & \text{otherwise} \end{cases}$$

We substitute it in Eq. (10), together with Eq. (1) and Eq. (8), obtaining:

$$\int_{\mathbf{x}(0): \sum_{n=1}^{N} \mathbf{x}(0) = \mathbf{y}(0)} \frac{p(\mathbf{x}(0))}{p(\mathbf{y}(0))} \int_{\mathbf{x}(t)} \mathcal{N}(\mathbf{x}(t); \mathbf{x}(0), \sigma^2(t)\mathbf{I}) \mathbb{1}_{\mathbf{y}(t) = \sum_{n=1}^{N} \mathbf{x}_n(t)} \mathrm{d}\mathbf{x}(t) \mathrm{d}\mathbf{x}(0)\,. \tag{11}$$

We sum over the first $N-1$ sources in the inner integral, setting $\mathbf{x}_N(t) = \mathbf{y}(t) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)$:

$$\int_{\mathbf{x}_{1:N-1}(t)} \mathcal{N}(\mathbf{x}_{1:N-1}(t), \mathbf{y}(t) - \sum_{n=1}^{N-1} \mathbf{x}_n(t); \mathbf{x}(0), \sigma^2(t)\mathbf{I}) \mathrm{d}\mathbf{x}_{1:N-1}(t) \tag{12}$$

$$= \int_{\mathbf{x}_{1:N-1}(t)} \prod_{n=1}^{N-1} \mathcal{N}(\mathbf{x}_n(t); \mathbf{x}_n(0), \sigma^2(t)\mathbf{I}) \mathcal{N}(\mathbf{y}(t) - \sum_{n=1}^{N-1} \mathbf{x}_n(t); \mathbf{x}_N(0), \sigma^2(t)\mathbf{I}) \mathrm{d}\mathbf{x}_{1:N-1}(t)$$

$$= \mathcal{N}(\mathbf{y}(t); \sum_{n=1}^{N} \mathbf{x}_n(0), N\sigma^2(t)\mathbf{I})\,. \tag{13}$$

The second equality is obtained by factorizing the Gaussian, which has diagonal covariance matrix, while the last equality is obtained by iterative application of the convolution theorem Katznelson (2004). We substitute Eq. (13) in Eq. (11), obtaining:

$$\begin{aligned} p(\mathbf{y}(t) \mid \mathbf{y}(0)) &\approx \int_{\mathbf{x}(0):\sum_{n=1}^{N}\mathbf{x}_n(0)=\mathbf{y}(0)} \frac{p(\mathbf{x}(0))}{p(\mathbf{y}(0))}\mathcal{N}(\mathbf{y}(t);\sum_{n=1}^{N}\mathbf{x}_n(0), N\sigma^2(t)\mathbf{I})\mathrm{d}\mathbf{x}(0) \\ &= \mathcal{N}(\mathbf{y}(t);\mathbf{y}(0), N\sigma^2(t)\mathbf{I}) \int_{\mathbf{x}(0):\sum_{n=1}^{N}\mathbf{x}_n(0)=\mathbf{y}(0)} \frac{p(\mathbf{x}(0))}{p(\mathbf{y}(0))}\mathrm{d}\mathbf{x}(0) \\ &= \mathcal{N}(\mathbf{y}(t);\mathbf{y}(0), N\sigma^2(t)\mathbf{I})\,. \end{aligned} \tag{14}$$

At this point, we apply Bayes theorem in Eq. (9), substituting the Dirac likelihood:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx \int_{\mathbf{y}(t)} \frac{p(\mathbf{x}(t))p(\mathbf{y}(t) \mid \mathbf{x}(t))}{p(\mathbf{y}(t))} p(\mathbf{y}(t) \mid \mathbf{y}(0))\mathrm{d}\mathbf{y}(t) \tag{15}$$

$$= \int_{\mathbf{y}(t)} \frac{p(\mathbf{x}(t))\mathbb{1}_{\mathbf{y}(t)=\sum_{n=1}^{N}\mathbf{x}_n(t)}}{p(\mathbf{y}(t))} p(\mathbf{y}(t) \mid \mathbf{y}(0))\mathrm{d}\mathbf{y}(t) \tag{16}$$

$$= \frac{p(\mathbf{x}(t))}{p(\sum_{n=1}^{N}\mathbf{x}_n(t))} p(\sum_{n=1}^{N}\mathbf{x}_n(t) \mid \mathbf{y}(0))\,. \tag{17}$$

Estimating Eq. (17), however, requires knowledge of the mixture density $p(\sum_{n=1}^{N}\mathbf{x}_n(t))$, which we do not acknowledge. As such, we approximate Eq. (16) with Monte Carlo, using the mean of $p(\mathbf{y}(t) \mid \mathbf{y}(0))$, namely $\mathbf{y}(0)$ (see Eq. (14)), obtaining:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx \frac{p(\mathbf{x}(t))\mathbb{1}_{\mathbf{y}(0)=\sum_{n=1}^{N}\mathbf{x}_n(t)}}{p(\mathbf{y}(0))} = \begin{cases} \frac{p(\mathbf{x}(t))}{p(\mathbf{y}(0))} & \text{if } \sum_{n=1}^{N}\mathbf{x}_n(t) = \mathbf{y}(0) \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

Similar to how we constrained the integral in Eq. (12), we parameterize the posterior, without loss of generality, using the first $N-1$ sources $\tilde{\mathbf{x}}(t) = (\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t))$. The last source is constrained setting $\mathbf{x}_N(t) = \mathbf{y}(0) - \sum_{n=1}^{N-1}\mathbf{x}_n(t)$ and the parameterization is defined as:

$$F(\tilde{\mathbf{x}}(t)) = F(\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t)) = (\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t), \mathbf{y}(0) - \sum_{n=1}^{N-1}\mathbf{x}_n(t))\,. \tag{19}$$

Plugging Eq. (19) in Eq. (18) we obtain the parameterized posterior:

$$p(F(\tilde{\mathbf{x}}(t)) \mid \mathbf{y}(0)) \approx \frac{p(F(\tilde{\mathbf{x}}(t)))}{p(\mathbf{y}(0))} \tag{20}$$

At this point, we compute the gradient of the logarithm of Eq. (20) with respect to $\tilde{\mathbf{x}}(t)$:

$$\begin{aligned} \nabla_{\tilde{\mathbf{x}}(t)} \log p(F(\tilde{\mathbf{x}}(t)) \mid \mathbf{y}(0)) &\approx \nabla_{\tilde{\mathbf{x}}(t)} \log \frac{p(F(\tilde{\mathbf{x}}(t)))}{p(\mathbf{y}(0))} \\ &= \nabla_{\tilde{\mathbf{x}}(t)} \log p(F(\tilde{\mathbf{x}}(t))) - \nabla_{\tilde{\mathbf{x}}(t)} \log p(\mathbf{y}(0)) \\ &= \nabla_{\tilde{\mathbf{x}}(t)} \log p(F(\tilde{\mathbf{x}}(t)))\,. \end{aligned} \tag{21}$$

Using the chain-rule for differentiation on Eq. (21) we have:

$$\nabla_{\tilde{\mathbf{x}}(t)} \log p(F(\tilde{\mathbf{x}}(t)) \mid \mathbf{y}(0)) \approx \nabla_{F(\tilde{\mathbf{x}}(t))} \log p(F(\tilde{\mathbf{x}}(t)))J_F(\tilde{\mathbf{x}}(t)), \tag{22}$$

where $J_F(\tilde{\mathbf{x}}(t)) \in \mathbb{R}^{(N\times D)\times((N-1)\times D)}$ is the Jacobian of F computed in $\tilde{\mathbf{x}}(t)$, equal to:

$$J_F(\tilde{\mathbf{x}}(t)) = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \ldots & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \ldots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \ldots & \mathbf{I} \\ -\mathbf{I} & -\mathbf{I} & \ldots & -\mathbf{I} \end{bmatrix}$$

The gradient with respect to a source $\mathbf{x}_m(t)$ with $1 \leq m \leq N-1$ in Eq. (22) is thus equal to:

$$\nabla_{\mathbf{x}_m(t)} \log p(F(\tilde{\mathbf{x}}(t)) \mid \mathbf{y}(0)) \approx [\nabla_{F(\tilde{\mathbf{x}}(t))} \log p(F(\tilde{\mathbf{x}}(t))]_m \\ - [\nabla_{F(\tilde{\mathbf{x}}(t))} \log p(F(\tilde{\mathbf{x}}(t))]_N \,,$$

where we index the components of the m-th and N-th sources in $\nabla_{F(\tilde{\mathbf{x}}(t))} \log p(F(\tilde{\mathbf{x}}(t))$. Finally, we replace the gradients with the score networks:

$$\nabla_{\mathbf{x}_m(t)} \log p(F(\tilde{\mathbf{x}}(t))|\mathbf{y}(0)) \approx S_m^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t), \mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)), \sigma(t)) \\ - S_N^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_{N-1}(t), \mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t)), \sigma(t)) \,, \tag{23}$$

where S_m^θ and S_N^θ are the entries of the score network corresponding to the m-th and N-th sources.

B Derivation of Gaussian and Weakly-Supervised Posterior Scores for Source Separation

In this Section we derive the formulas for 'MSDM Gaussian', 'ISDM Dirac' and 'ISDM Gaussian'. We first adapt the Gaussian posterior introduced in Jayaram & Thickstun (2020) to continuous-time score-based diffusion models Karras et al. (2022). We plug the Gaussian likelihood function (Eq. (7)) into Eq. (15), obtaining:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx \int_{\mathbf{y}(t)} \frac{p(\mathbf{x}(t))\mathcal{N}(\mathbf{y}(t); \sum_{n=1}^N \mathbf{x}_n(t), \gamma^2(t)\mathbf{I})}{p(\mathbf{y}(t))} p(\mathbf{y}(t) \mid \mathbf{y}(0)) \mathrm{d}\mathbf{y}(t) \tag{24}$$

Following Jayaram & Thickstun (2020), $\mathbf{y}(t)$ is not re-sampled during inference and is always set to $\mathbf{y}(0)$. As such, we perform Monte Carlo in Eq. (24) with $\mathbf{y}(0)$, the mean of $p(\mathbf{y}(t) \mid \mathbf{y}(0)$ (see Eq. (14)), obtaining:

$$p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx \frac{p(\mathbf{x}(t))\mathcal{N}(\mathbf{y}(0); \sum_{n=1}^N \mathbf{x}_n(t), \gamma^2(t)\mathbf{I})}{p(\mathbf{y}(0))} \,. \tag{25}$$

At this point, we compute the gradient of the logarithm of Eq. (25) with respect to $\mathbf{x}_m(t)$:

$$\begin{aligned} \nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t) \mid \mathbf{y}(0)) &\approx \nabla_{\mathbf{x}_m(t)} \log \frac{p(\mathbf{x}(t))\mathcal{N}(\mathbf{y}(0); \sum_{n=1}^N \mathbf{x}_n(t), \gamma^2(t)\mathbf{I})}{p(\mathbf{y}(0))} \\ &= \nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t)) + \nabla_{\mathbf{x}_m(t)} \log \mathcal{N}(\mathbf{y}(0); \sum_{n=1}^N \mathbf{x}_n(t), \gamma^2(t)\mathbf{I}) \\ &= \nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t)) \; - \frac{1}{2\gamma^2(t)} \nabla_{\mathbf{x}_m(t)} \|\mathbf{y}(0) - \sum_{n=1}^N \mathbf{x}_n(t)\|_2^2 \\ &= \nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t)) \; - \frac{1}{\gamma^2(t)} (\mathbf{y}(0) - \sum_{n=1}^N \mathbf{x}_n(t)) \,. \end{aligned} \tag{26}$$

We obtain the 'MSDM Gaussian' posterior score by replacing the contextual prior with the score network:

$$\nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx S_m^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_N(t)), \sigma(t)) - \frac{1}{\gamma^2(t)} (\mathbf{y}(0) - \sum_{n=1}^N \mathbf{x}_n(t)) \,. \tag{27}$$

The weakly-supervised posterior scores are obtained by approximating:

$$p(\mathbf{x}_1(t), \ldots, \mathbf{x}_N(t)) \approx \prod_{n=1}^N p_n(\mathbf{x}_n(t)) \,,$$

Model	Inference Time (s)	# of parameters
Demucs	$0.111_{\pm 0.071}$	40M
Demucs + Gibbs (512 steps)	$0.111_{\pm 0.071} \times 512 = 56.832_{\pm 36.352}$	$\sim$ 40M
Demucs + Gibbs (256 steps)	$0.111_{\pm 0.071} \times 256 = 28.416_{\pm 18.176}$	$\sim$ 40M
ISDM (correction)	$4.6_{\pm 0.345} \times 4 = 18.4_{\pm 1.38}$	405M $\times 4$
MSDM (correction)	$4.6_{\pm 0.345}$	405M

Table 4: Inference times for a single 12s long separation and number of parameters for each model in Table 3. We report Demucs + Gibbs (256 steps) since the minimum number of steps that makes the SI-SDR$_{\text{I}}$ over all instruments (17.59 dB) greater than ISDM. While ISDM and MSDM are not time-competitive to Demucs, they are more time-efficient than Demucs + Gibbs (256 and 512 steps).

where p_n are estimated with independent score functions S_n^θ. In the contextual samplers in Eq. (23) ('MSDM Dirac') and Eq. (27) ('MSDM Gaussian'), $S_n^\theta((\mathbf{x}_1(t), \ldots, \mathbf{x}_N(t)), \sigma(t))$ refers to a slice of the full score network on the components of the $n-$th source. In the weakly-supervised cases, S_n^θ is an individual function. To obtain the 'ISDM Dirac' posterior score, we factorize the prior in Eq. (21), then use the chain rule of differentiation, as in Appendix A, to obtain:

$$\nabla_{\mathbf{x}_m(t)} \log p(F(\tilde{\mathbf{x}}(t)) \mid \mathbf{y}(0)) \approx \nabla_{\mathbf{x}_m(t)} \log p_m(\mathbf{x}_m(t)) + \nabla_{\mathbf{x}_m(t)} \log p_N(\mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t))$$

$$\approx S_m^\theta(\mathbf{x}_m(t), \sigma(t)) - S_N^\theta(\mathbf{y}(0) - \sum_{n=1}^{N-1} \mathbf{x}_n(t), \sigma(t)) \,.$$

We obtain the 'ISDM Gaussian' posterior score by factorizing the joint prior in Eq. (26):

$$\nabla_{\mathbf{x}_m(t)} \log p(\mathbf{x}(t) \mid \mathbf{y}(0)) \approx S_m^\theta(\mathbf{x}_m(t), \sigma(t)) - \frac{1}{\gamma^2(t)}(\mathbf{y}(0) - \sum_{n=1}^{N} \mathbf{x}_n(t)) \,.$$

C Experimental Setup

C.1 Dataset

We perform experiments mainly on Slakh2100 (Manilow et al., 2019), a standard dataset for music source separation. Slakh2100 is a collection of multi-track waveform music data synthesized from MIDI files using virtual instruments of professional quality. The dataset comprises 2100 tracks, with a distribution of 1500 tracks for training, 375 for validation, and 225 for testing. Each track is accompanied by its stems, which belong to 31 instrumental classes. For a fair comparison, we only used the four most abundant classes as in (Manilow et al., 2022), namely Bass, Drums, Guitar, and Piano; these instruments are present in the majority of the songs: 94.7% (Bass), 99.3% (Drums), 100.0% (Guitar), and 99.3% (Piano).

In Appendix E, we experiment on MUSDB18-HQ Rafii et al. (2019), a benchmark dataset for the music source separation task. It contains 150 tracks, with 100 allocated for training and 50 for testing, amounting to roughly 10 hours of professional-grade audio. Each piece within the dataset is separated into the stems: Bass, Drums, Vocals, and Other, with the latter encompassing any elements not included in the categories above.

C.2 Architecture and Training

The implementation of the score network is based on a time domain (non-latent) unconditional version of Moûsai (Schneider et al., 2023).

We used the publicly available repository `audio-diffusion-pytorch/v0.0.432`[2]. The score network is a U-Net Ronneberger et al. (2015) comprised of encoder, bottleneck, and de-

[2] `https://github.com/archinetai/audio-diffusion-pytorch/tree/v0.0.43`

Table 5: Hyperparameter search for source separation. We use 'MSDM Dirac' (top-left), 'ISDM Dirac' (bottom-left), 'MSDM Gaussian' (top-right) and 'ISDM Gaussian' (bottom-right) posteriors. We report the SI-SDR$_\text{I}$ values in dB (higher is better) averaged over all instruments (Bass, Drums, Piano, Guitar).

		Dirac Likelihood				**Gaussian Likelihood**						
	S_churn	Constrained Source				$\gamma(t)$						
		Bass	Drums	Guitar	Piano	$0.25\sigma(t)$	$0.5\sigma(t)$	$0.75\sigma(t)$	$1\sigma(t)$	$1.25\sigma(t)$	$1.5\sigma(t)$	$2\sigma(t)$
MSDM	0	4.41	5.05	3.28	2.87	-41.54	6.37	6.05	5.67	5.729	5.13	4.33
	1	7.90	8.18	7.03	7.05	-47.24	6.79	6.51	6.15	6.19	5.66	4.45
	20	**14.29**	12.99	12.19	11.69	-47.17	11.07	10.51	9.43	10.19	9.18	7.58
	40	14.28	13.02	5.51	4.78	-47.17	-36.92	**12.48**	11.25	11.87	10.80	9.03
ISDM	0	5.05	3.69	-2.50	6.93	-45.46	7.12	6.50	5.78	5.02	4.49	3.69
	1	9.23	8.57	7.28	9.20	-47.54	7.57	7.20	6.32	5.35	4.82	3.83
	20	15.35	15.08	13.20	15.36	-46.86	12.89	12.21	10.87	9.32	8.32	6.47
	40	**17.26**	15.77	15.30	14.98	-46.86	-35.97	**14.09**	12.82	10.85	10.02	8.26
	60	16.21	15.57	15.51	14.20	-46.80	-46.85	14.06	12.57	11.83	10.81	9.24

coder with skip connections between the encoder and the decoder. The encoder has six layers comprising two convolutional ResNet blocks, followed by multi-head attention in the final three layers. The signal sequence is downsampled in each layer by a factor of 4. The number of channels in the encoder layers is [256, 512, 1024, 1024, 1024, 1024]. The bottleneck consists of a ResNet block, followed by self-attention, and another ResNet block (all with 1024 channel layers). The decoder follows a reverse symmetric structure with respect to the encoder. We employ `audio-diffusion-pytorch-trainer`[3] for training. We downsample data to 22kHz and train the score network with four stacked mono channels for MSDM (i.e., one for each stem) and one mono channel for each model in ISDM, using a context length of ~ 12 seconds. All our models were trained until convergence on an NVIDIA RTX A6000 GPU with 24 GB of VRAM. We trained all our models using Adam Kingma & Ba (2015), with a learning rate of 10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.99$ and a batch size of 16.

We report inference times and number of parameters of the various models in Table 4.

C.3 The Sampler

We use a first-order ODE integrator based on the Euler method and introduce stochasticity following (Karras et al., 2022). The amount of stochasticity is controlled by the parameter S_churn. As shown in Appendix D and explained in detail in (Karras et al., 2022), stochasticity significantly improves sample quality. We implemented a correction mechanism (Song et al., 2021; Jayaram & Thickstun, 2020) iterating for R steps after each prediction step i, adding additional noise and re-optimizing with the score network fixed at σ_i. As per Karras et al. (2022), we adopt a non-linear schedule for time discretization that gives more importance to lower noise levels. It is defined as:

$$t_i = \sigma_i = \sigma_\text{max}^{\frac{1}{\rho}} + \frac{i}{I-1}(\sigma_\text{min}^{\frac{1}{\rho}} - \sigma_\text{max}^{\frac{1}{\rho}})^\rho \,,$$

where $0 \leq i < I$, with I the number of discretization steps. We set $\sigma_\text{min} = 10^{-4}$, $\sigma_\text{max} = 1, \rho = 7$.

D Hyperparameter Search for Source Separation

We conduct a hyperparameter search over S_churn to evaluate the importance of stochasticity in source separation over a fixed subset of 100 chunks of the Slakh2100 test set, each spanning 12 seconds (selected randomly). To provide a fair comparison between the Dirac ('MSDM Dirac', 'ISDM Dirac') and Gaussian ('MSDM Gaussian', 'ISDM Gaussian') posterior scores, we execute a search over their specific hyperparameters, namely the constrained source for the Dirac separators and the

[3] `https://github.com/archinetai/audio-diffusion-pytorch-trainer/tree/79229912`

Table 6: Comparison of results of MSDM and Demucs v2 (Défossez et al., 2019). We report the SI-SDR$_\text{I}$ values in dB (higher is better). The network is the same as the one trained on Slakh2100, except that the sampling rate is 44kHz and is trained on 6s long chunks.

	Tested on MUSDB			Finetuned on MUSDB			Trained on MUSDB				
Model	Bass	Drums	All	Bass	Drums	All	Bass	Drums	Other	Vocals	All
Demucs v2	-	-	-	-	-	-	13.28	11.53	8.59	16.80	12.55
MSDM	-0.83	-0.94	-0.88	3.46	5.03	4.25	4.87	3.28	1.97	6.83	4.24

$\gamma(t)$ coefficient for the Gaussian separators. Results are illustrated in Table 5. We observe that: *(i)* stochasticity proves beneficial for all separators, given that the highest values of SI-SDR$_\text{I}$ are achieved with $S_{\text{churn}} = 20$ and $S_{\text{churn}} = 40$, *(ii)* using the Dirac likelihood we obtain higher values of SI-SDR$_\text{I}$ with respect to the Gaussian likelihood, both with the MSDM and ISDM separators, and *(iii)* the ISDM separators perform better than the contextual MSDM separators (at the expense of not being able to perform total and partial generation).

E Data Efficiency Study: Results on MUSDB

We report in Table 6 the results of MSDM and Demucs v2 (Défossez et al., 2019) on the MUSDB18-HQ test set Rafii et al. (2019). We try three different strategies, we first check the out-of-distribution ability of the model trained on Slakh2100 by testing directly on MUSDB18-HQ. Then, we try finetuning the model trained on Slakh2100 on MUSDB18-HQ, and finally, we train directly on MUSDB18-HQ. Since the only stems shared between MUSDB18-HQ and Slakh2100 are 'Bass' and 'Drums', the first and second strategies can be tested only on these two stems.

F Subjective Evaluation

The details of the listening test are explained in Figure 3.

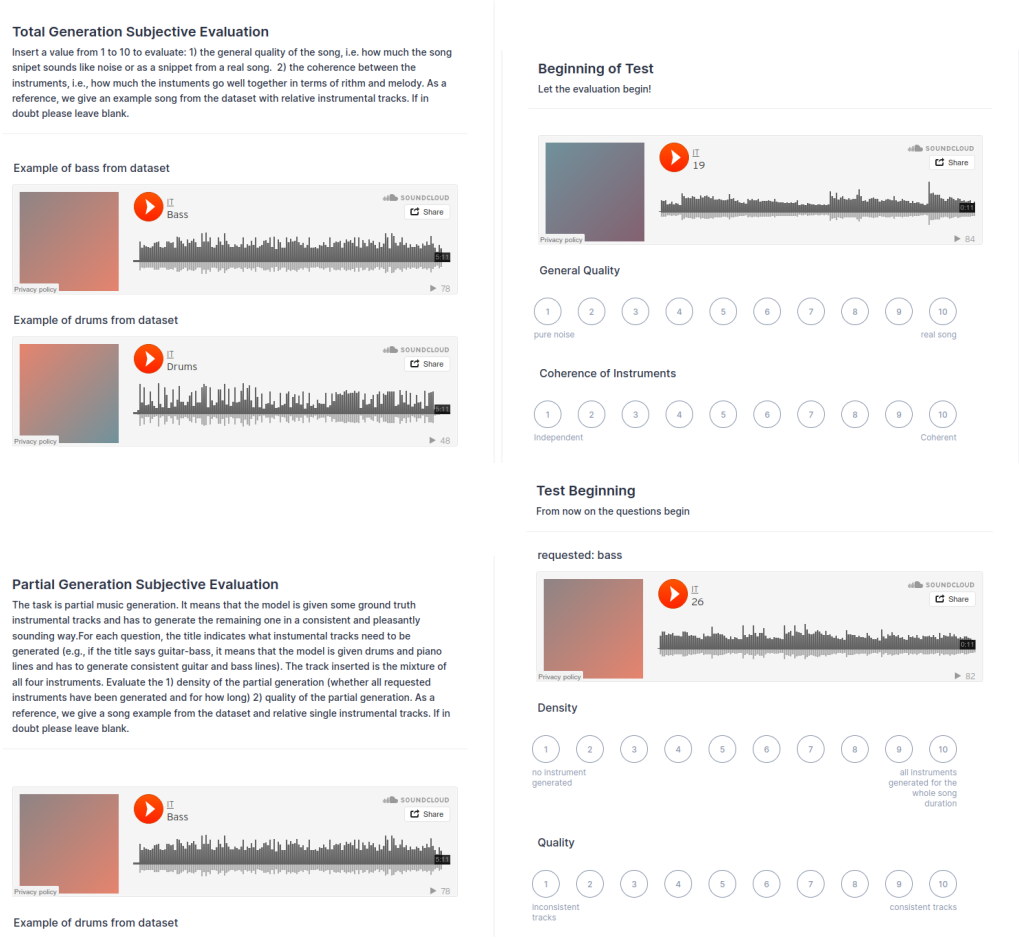

Figure 3: Snippets from the subjective evaluation form. The first row is relative to total generation, where people were asked to evaluate 30 songs, 15 of which were generated by the mixture model and 15 by MSDM. Thirty-two people answered the survey. The second row is relative to partial generation. Subjects were asked to evaluate 15 songs. A random subset of sources is fixed for each song, and MSDM generates the other. The requested sources are explicitly stated above the song (e.g., in the snippet, the model has generated only the Bass stem). Twenty-one subjects answered.