# High Fidelity Text-Guided Music Editing via Single-Stage Flow Matching

**Gael Le Lan Bowen Shi Zhaoheng Ni Sidd Srinivasan**
**Anurag Kumar Brian Ellis David Kant Varun Nagaraja Ernie Chang**
**Wei-Ning Hsu Yangyang Shi Vikas Chandra**
AI at Meta
{glelan,bshi}@meta.com

## Abstract

We introduce MELODYFLOW, an efficient text-controllable high-fidelity music generation and editing model. It operates on sequences of continuous latent representations from a low frame rate 48 kHz stereo variational auto encoder codec. Based on a diffusion transformer architecture trained on a flow-matching objective the model can edit diverse high quality stereo samples of variable duration, with simple text descriptions. We adapt the ReNoise latent inversion method to flow matching and compare it with the naive denoising diffusion implicit model (DDIM) inversion on a variety of music editing prompts. Our results indicate that the regularized latent inversion outperforms DDIM for zero-shot test-time text-guided editing on several objective metrics. Subjective evaluations exhibit comparable performance between both methods, showing a noticeable improvement over previous state of the art for music editing. Code and model weights will be publicly made available. Samples are available at https://melodyflow.github.io.

## 1 Introduction

Text-conditioned music generation has made tremendous progress in the past two years [Schneider et al., 2023, Huang et al., 2023, Agostinelli et al., 2023, Copet et al., 2024, Ziv et al., 2023, Liu et al., 2023, Li et al., 2023]. The dominant approach involves representing audio as a sequence of compressed discrete or continuous representations and training a generative model on top of it. Two dominant generative model architectures have emerged, one based on Language Models (LMs) [Agostinelli et al., 2023, Copet et al., 2024], the other on diffusion [Schneider et al., 2023, Huang et al., 2023, Liu et al., 2023, Li et al., 2023]. A third method sometimes referred to as discrete diffusion relies on non-autoregressive masked token prediction [Ziv et al., 2023, Garcia et al., 2023]. The target level of audio fidelity depends on the models and some have already successfully generated 44.1 kHz or high stereo signals [Schneider et al., 2023, Li et al., 2023, Evans et al., 2024a].

Due to the growing popularity of diffusion models in computer vision, a new area of research has emerged focusing on text-controlled audio editing [Wang et al., 2023, Lin et al., 2024, Garcia et al., 2023, Wu et al., 2023, Novack et al., 2024, Zhang et al., 2024, Manor and Michaeli, 2024]. The creative process for sound design typically involves multiple iterations and using efficient editing methods is an effective approach to achieving this. Music editing constitutes an open ended list of tasks such as inpainting/outpainting, looping, instrument or genre swapping, vocals removal, lyrics editing, tempo control or recording conditions modification (e.g. from studio quality to a concert setting). Some of those tasks have been addressed in recent works with specialised models [Wang et al., 2023, Garcia et al., 2023, Lin et al., 2024, Wu et al., 2023, Copet et al., 2024] or zero-shot editing methods from the computer vision domain that are exclusive to diffusion models [Novack et al., 2024, Zhang et al., 2024, Manor and Michaeli, 2024]. However, none of these have

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

demonstrated high fidelity generic style transfer capabilities, due to either lack of high quality data or foundational music generation model, or design choices that do not generalise to any kind of editing task. The question of inference speed is key for creatives and the music domain is particularly challenging due to the high fidelity (48 kHz stereo) requirement in the sound design process. Recently the Flow Matching (FM) generative modeling formulation has been introduced [Lipman et al., 2022], that consists in building optimal transport paths between data and noise samples. It is a more robust and stable alternative for training diffusion models with notably faster inference and was successfully applied to train foundational speech [Le et al., 2024] and audio [Vyas et al., 2023] generative models. Prajwal et al. [2024] employed a two-stage FM model for text-guided music generation, cascading semantic and acoustic features generation.

In this work we present MELODYFLOW, a single-stage text-conditioned FM model designed for instrumental music generation and editing. The model operates on continuous representations of a low frame rate Variational Audio Encoder (VAE) codec. Additionally, thanks to the versatility of FM, MELODYFLOW is compatible with any zero-shot test-time editing method such as DDIM inversion [Song et al., 2020]. We enhance the editability of the FM inversion process with a regularized latent inversion method inspired by that of [Garibi et al., 2024] for diffusion-based image editing. Both our objective and subjective evaluations on music editing indicate that MELODYFLOW can support a diversity of editing tasks on real songs without any finetuning, achieving fast music editing with remarkable consistency, text-adherence and minimal quality loss compared with original samples.

**Our contributions:** (i) The first of its kind single-stage text-to-music FM model to generate and edit 48 kHz stereo samples of up to 30 seconds, with enhancements in both the audio latent representation and generative model, striking a better balance between quality and efficiency. (ii) We explore a novel regularized FM inversion method capable of performing faithful zero-shot test-time text-guided editing on various axes while maintaining coherence with the original sample. (iii) We make the code and model weights publicly available to foster research on music editing.

## 2 Method

MELODYFLOW is a non-causal transformer-based FM model conditioned on T5 text representations. It operates on sequences of latent audio representations from a neural quantizer-free audio codec with a VAE bottleneck. Combined with latent inversion, the model is able to perform text-guided editing of real or generated audio samples. The overall architecture of the model is depicted in the Figure 1. Some of our design choices are critical in striking a good balance between quality and efficiency, which we highlight hereafter. The appendix A.1 shared additional details regarding the model key components (codec, generative model and latent inversion implementation).

### 2.1 Codec VAE bottleneck

Compared with Vyas et al. [2023] we swap the Residual Vector Quantization (RVQ) layer of the codec with a VAE bottleneck. Although FM with VAE have been explored for image generation Esser et al. [2024], our work is the first to study the influence of the VAE regulariser on music generation performance. Indeed Rombach et al. [2022] - a seminal work on VAE for image generation - note that *LDMs trained in VQ-regularized latent spaces achieve better sample quality* than KL-regularized ones. Our ablation in appendix A.3.3 demonstrates that this conclusion does not apply for audio. Using a KL-regularizer achieves better waveform reconstruction performance for a much lower frame rate, unlocking faster inference and scaling to high-fidelity stereo audio (shown in appendix A.3.4).

**Minibatch coupling** [Tong et al., 2023, Pooladian et al., 2023] expanded over prior work on FM modeling by sampling pairs $(\mathbf{x}, \epsilon)$ from the joint distribution given the by the optimal transport plan between the data $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^{B}$ and noise $\mathbf{E} = \{\epsilon^{(i)}\}_{i=1}^{B}$ samples within a batch of size $B$. Essentially this translates into running the Hungarian algorithm so as to find the permutation matrix $\mathbf{P}$ that minimizes $||\mathbf{X} - \mathbf{PE}||_2^2$. They demonstrate it results in straighter optimal transport paths during inference (that are closer to the theoretical linear mapping assumption between noise and data samples) and consequently offers better quality-efficiency trade offs. We shed light on the importance of mini-batch coupling in section 4.2 and we underline the overall benefit of the FM model design choices on its efficiency in appendix A.3.2.

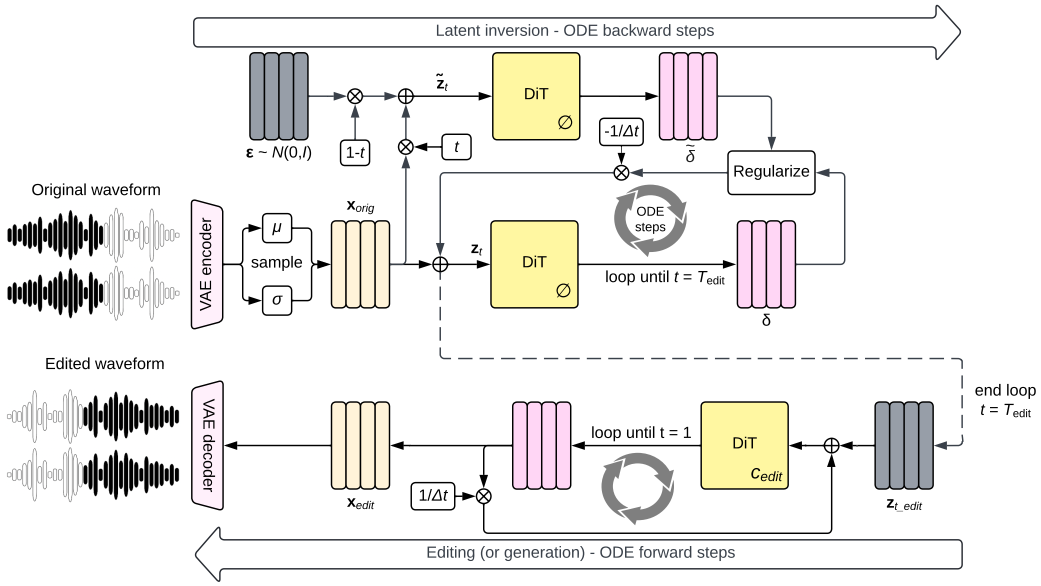


Figure 1: Overview of MelodyFlow. An source waveform is encoded into latent representation before being fed to the ODE solver. A diffusion transformer predicts the flow from data to noise step by step, while being regularized against an artificially constructed mixture of noise and data at each step. $T_{edit}$ is the target inversion timestep. Given $S$ ODE solver steps, the inversion step size is given by $\Delta t = (1 - T_{edit})/S$. The editing prompt $c_{edit}$ is not used during inversion.

**Regularized latent inversion** [Garibi et al., 2024] recently proposed ReNoise, an iterative renoising mechanism applied at each inversion sampling step of a diffusion model. It enables building reversible diffusion trajectories with a good reconstruction/editability balance thanks to a noise regularization applied at each inversion step. We follow a similar reasoning but adapted in the FM setting. In our case we only enforce the model prediction regularization, but neither de-correlation nor noise correction steps. More details about our adaptation of ReNoise for FM can be found in appendix A.1.4, along with an experimental comparison with the original implementation A.4.2. We provide a comparative analysis with DDIM inversion in sections 4.2 and A.4.1.

## 3 Experimental setup

**Model** MELODYFLOW is a diffusion transformer of sizes 400M (small) and 1B (medium) parameters with U-shaped skip connections Bao et al. [2023]. The model is conditioned via cross attention on a T5 representation [Raffel et al., 2020] computed from the text description of the music. The flow step is injected following [Hatamizadeh et al., 2023]. Minibatch coupling is computed with `torch-linear-assignement`[1]. MELODYFLOW-small (resp. MELODYFLOW-medium) is trained on latent representation sequences of 32 kHz mono (resp. 48 kHz stereo) segments of 10 (resp. 30) seconds, encoded at 20 Hz frame rate (resp. 25 Hz). From the codec perspective the only difference between encoding mono or stereo waveform is the number of input (resp. output) channels for the first (resp. last) convolution of the encoder (resp. decoder): 1 for mono and 2 for stereo. More details regarding audio representation and FM model training are provided in appendix A.2.

**Inference and editing** For text-to-music inference we use the `midpoint` ODE solver from `torchdiffeq` with a step size of $0.03125$. Classifier free guidance of $5.0$ is chosen after grid search. For music editing we run our regularized inversion method with the `euler` ODE solver until $T_{edit} = 0.04$ with the same classifier free guidance of $5.0$ applied in both ODE directions. Trajectory inversion estimation is run with $S = 25, K = 4, w_k = k - 1$ and $\lambda_{KL} = 0.2$.

[1] `https://github.com/ivan-chai/torch-linear-assignment`

Table 1: Comparison to baselines on text-guided high fidelity music editing of samples from the IN-DOMAIN test set, using LLM-assisted editing prompts.

| MODEL | METHOD | OVL. ↑ | REL. ↑ | CON. ↑ | AVG. ↑ |
|---|---|---|---|---|---|
| AUDIOLDM 2-music | DDPM inv. | 2.48±0.07 | 2.36±0.08 | **2.72**±0.09 | 2.52 |
| MUSICGEN-melody | Chroma cond. | 2.57±0.08 | 2.46±0.09 | 2.14±0.07 | 2.39 |
| MELODYFLOW-medium | Reg. inv. | **2.72**±0.08 | **2.72**±0.07 | 2.61±0.10 | **2.68** |

### 3.1 Datasets

**Training** Our training dataset is made of 10K high-quality internal music tracks and the ShutterStock and Pond5 music collections with respectively 25K and 365K instrument-only music tracks, totalling into 20k hours. All datasets consist of full-length music sampled at 48 kHz stereo with metadata composed of a textual description sometimes containing the genre, BPM and key. Descriptions are curated by removing frequent patterns that are unrelated to the music (such as URLs). For 32 kHz mono models the waveform is downsampled and the stereo channels are averaged.

**Evaluation** For the main text-to-music generation results we evaluate MELODYFLOW and prior work on the MusicCaps dataset [Agostinelli et al., 2023]. We compute objective metrics for MELODYFLOW and report those from previous literature. Subjective evaluations are conducted on a subset of 198 examples from the genre-balanced set. For ablations we rely on an in-domain held out evaluation set different from that of [Copet et al., 2024], made of 8377 tracks. The same in-domain tracks are used for objective editing evaluations. Subjective evaluations of edits are run on a subset of 181 higher fidelity samples from our in-domain test set with LLM-assisted designed prompts (A.2.3).

**Metrics** We evaluate MELODYFLOW using both objective and subjective metrics, following the evaluation protocol of [Kreuk et al., 2022, Copet et al., 2024] for generation. Reported objective metrics are the Fréchet Audio Distance (FAD) [Roblek et al., 2019] with VGGish embeddings [Hershey et al., 2017], the Kullback–Leibler divergence (KLD) with PASST audio encoder [Koutini et al., 2021] and CLAP[2] cosine similarity [Elizalde et al., 2023]. For music editing evaluations we compute the average L2 distance between the original and edited latent sequences (LPAPS [Iashin and Rahtu, 2021]), FAD$_{edit}$ between the distribution of source and edited samples and CLAP$_{edit}$ between the edited audio and the editing prompt. Subjective evaluations relate to (i) overall quality (OVL), and (ii) relevance to the text input (REL), both using a Likert scale (from 1 to 5). Additionally for music editing evaluations we report (iii) editing consistency (CON). Raters were recruited using the Amazon Mechanical Turk platform and all samples were normalized to -14dB LUFS [Series, 2011]. For stereo samples objective evaluation the signal is down mixed into mono prior to metrics computation. For subjective ratings we keep the original audio format generated by each model.

## 4 Results

### 4.1 Text-guided music editing

We compare MELODYFLOW-medium with existing open source music editing implementations, namely MUSICGEN-melody and AUDIOLDM 2 with DDPM inversion (following [Manor and Michaeli, 2024]). The table 1 presents the main music editing results. MELODYFLOW outperforms both baselines on the quality and text-fidelity axes. Consistency-wise MELODYFLOW lags slightly behind AUDIOLDM 2. Indeed during our listening tests we observe that AUDIOLDM 2 with DDPM inversion sometimes only generates a distorted version of the original track. Averaging on the three axes MELODYFLOW sets a new baseline for zero-shot music editing at test-time.

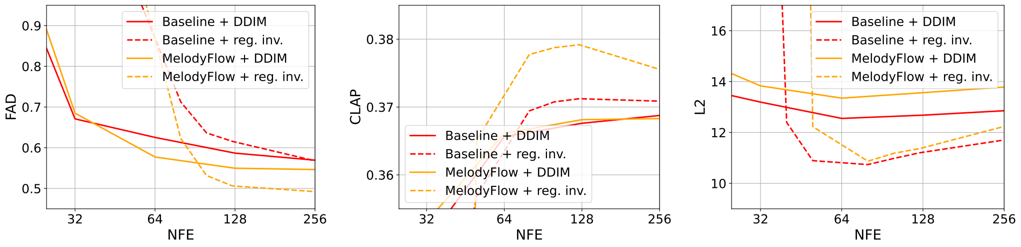
### 4.2 Ablations

To shed light on the benefit of our design choices, we compare MELODYFLOW with the baseline FM implementation of [Le et al., 2024], both being trained on the same latent representation.

[2] `https://github.com/LAION-AI/CLAP`

Table 2: FM model design ablation. FAD (resp. MSE) is reported on the IN-DOMAIN test (resp. validation) set. Baseline is adapted from [Le et al., 2024] but retrained on our music latents.

| ABLATION | HEADS | LAYERS | INFILL | SAMPLING | OT-FM | $\text{FAD}_{vgg} \downarrow$ | $\text{MSE}_{loss} \downarrow$ |
|---|---|---|---|---|---|---|---|
| baseline | 16 | 24 | ✓ | uniform | ✗ | .53 | - |
| − infilling | 16 | 24 | ✗ | uniform | ✗ | .50 | .8596 |
| + sampling | 16 | 24 | ✗ | logit-normal | ✗ | .44 | .8484 |
| + batch coupling | 16 | 24 | ✗ | logit-normal | ✓ | .42 | .8322 |
| + wider model | 18 | 18 | ✗ | logit-normal | ✓ | .39 | .8310 |



(a) $\text{FAD}_{edit}$ vs. NFE.

(b) $\text{CLAP}_{edit}$ vs. NFE.

(c) LPAPS vs. NFE.

Figure 2: Efficiency-quality trade offs of MELODYFLOW in the text-guided music editing setting, measured using objective metrics. Objective metrics ($\text{FAD}_{edit}$ in the Figure 2a, $\text{CLAP}_{edit}$ in the Figure 2b and LPAPS in the Figure 2c) indicate a sweet spot around 128 NFE.

**FM design choices** Table 2 presents the impact of the key design choices in the FM model training. Considering our baseline implementation was initially designed for text-to-speech [Le et al., 2024], it included an infilling objective that was meant to help the model handle variable length sequences that are inherent to the speech domain. We report the in-domain test FAD and the last validation $\text{MSE}_{loss}$ computed from the EMA checkpoint. No loss value is report for the baseline as the infilling objective facilitates the task. For the second line of the table the reported loss is weighted by the logit-normal probability density function at each sampled flow step to be comparable with logit-normal sampling. With all methods combined the in-domain FAD is reduced to 0.39 from 0.53 and consistent with the loss decrease, which validates our design choices.

**Music editing efficiency** We compare DDIM with MELODYFLOW inversion using a target inversion timestep $T_{edit} = 0$. $\text{FAD}_{edit}$ (Figure 2a), $\text{CLAP}_{edit}$ (Figure 2b) and LPAPS (Figure 2c) are plotted as a function of the total number of DiT forward passes (NFE). The combination of our FM implementation and proposed inversion method consistently outperform the baseline.

## 5 Discussion

In this work we presented MELODYFLOW, the first non-autoregressive model tailored for zero-shot test-time text-guided editing of high-fidelity stereo music. In the text-to-music setting the model offers competitive performance thanks to a low frame rate VAE codec and FM model featuring logit-normal flow step sampling, optimal-transport minibatch coupling and L-shaped attention mask. Combined with our proposed regularized latent inversion method, MELODYFLOW outperforms previous zero-shot test-time methods by a large margin. The model achieves remarkable efficiency that is key for the sound design creative process and supports variable duration samples. Our extensive evaluation, that includes objective metrics and human studies, highlights MELODYFLOW promise for efficient music editing with remarkable consistency, text-adherence and minimal quality degradation compared with the original, while remaining competitive on the task of text-to-music generation. For future work we intend to explore how to accurately evaluate specific editing axes and how such a model could help design metrics that better correlate with human preference.

## 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.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22669–22679, June 2023.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation. *Advances in Neural Information Processing Systems*, 36, 2024.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*, 2022.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. *arXiv preprint arXiv:2403.03206*, 2024.

Zach Evans, CJ Carr, Josiah Taylor, Scott H Hawley, and Jordi Pons. Fast timing-conditioned latent audio diffusion. *arXiv preprint arXiv:2402.04825*, 2024a.

Zach Evans, Julian D. Parker, CJ Carr, Zack Zukowski, Josiah Taylor, and Jordi Pons. Long-form music generation with latent diffusion, 2024b.

Hugo Flores Garcia, Prem Seetharaman, Rithesh Kumar, and Bryan Pardo. Vampnet: Music generation via masked acoustic token modeling. *arXiv preprint arXiv:2307.04686*, 2023.

Daniel Garibi, Or Patashnik, Andrey Voynov, Hadar Averbuch-Elor, and Daniel Cohen-Or. Renoise: Real image inversion through iterative noising. *arXiv preprint arXiv:2403.14602*, 2024.

Deepanway Ghosal, Navonil Majumder, Ambuj Mehrish, and Soujanya Poria. Text-to-audio generation using instruction-tuned llm and latent diffusion model. *arXiv preprint arXiv:2304.13731*, 2023.

Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. *arXiv preprint arXiv:2312.02139*, 2023.

Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. *Advances in Neural Information Processing Systems*, 35:28708–28720, 2022a.

Qingqing Huang, Aren Jansen, Joonseok Lee, Ravi Ganti, Judith Yue Li, and Daniel PW Ellis. Mulan: A joint embedding of music audio and natural language. *arXiv preprint arXiv:2208.12415*, 2022b.

Qingqing Huang, Daniel S Park, Tao Wang, Timo I Denk, Andy Ly, Nanxin Chen, Zhengdong Zhang, Zhishuai Zhang, Jiahui Yu, Christian Frank, et al. Noise2music: Text-conditioned music generation with diffusion models. *arXiv preprint arXiv:2302.03917*, 2023.

Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly ddpm noise space: Inversion and manipulations. *arXiv preprint arXiv:2304.06140*, 2023.

Vladimir Iashin and Esa Rahtu. Taming visually guided sound generation. *arXiv preprint arXiv:2110.08791*, 2021.

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

Diederik Kingma and Ruiqi Gao. Understanding diffusion objectives as the elbo with simple data augmentation. *Advances in Neural Information Processing Systems*, 36, 2024.

Khaled Koutini, Jan Schlüter, Hamid Eghbal-Zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. *arXiv preprint arXiv:2110.05069*, 2021.

Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. Audiogen: Textually guided audio generation. *arXiv preprint arXiv:2209.15352*, 2022.

Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. High-fidelity audio compression with improved rvqgan. *Advances in Neural Information Processing Systems*, 36, 2024.

Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, et al. Voicebox: Text-guided multilingual universal speech generation at scale. *Advances in neural information processing systems*, 36, 2024.

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)*, pages 626–630. IEEE, 2019.

Peike Li, Boyu Chen, Yao Yao, Yikai Wang, Allen Wang, and Alex Wang. Jen-1: Text-guided universal music generation with omnidirectional diffusion models. *arXiv preprint arXiv:2308.04729*, 2023.

Liwei Lin, Gus Xia, Yixiao Zhang, and Junyan Jiang. Arrange, inpaint, and refine: Steerable long-term music audio generation and editing via content-based controls. *arXiv preprint arXiv:2402.09508*, 2024.

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

Haohe Liu, Qiao Tian, Yi Yuan, Xubo Liu, Xinhao Mei, Qiuqiang Kong, Yuping Wang, Wenwu Wang, Yuxuan Wang, and Mark D Plumbley. Audioldm 2: Learning holistic audio generation with self-supervised pretraining. *arXiv preprint arXiv:2308.05734*, 2023.

Hila Manor and Tomer Michaeli. Zero-shot unsupervised and text-based audio editing using ddpm inversion. *arXiv preprint arXiv:2402.10009*, 2024.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023.

Zachary Novack, Julian McAuley, Taylor Berg-Kirkpatrick, and Nicholas J Bryan. Ditto: Diffusion inference-time t-optimization for music generation. *arXiv preprint arXiv:2401.12179*, 2024.

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, and Ricky TQ Chen. Multisample flow matching: Straightening flows with minibatch couplings. *arXiv preprint arXiv:2304.14772*, 2023.

KR Prajwal, Bowen Shi, Matthew Le, Apoorv Vyas, Andros Tjandra, Mahi Luthra, Baishan Guo, Huiyu Wang, Triantafyllos Afouras, David Kant, et al. Musicflow: Cascaded flow matching for text guided music generation. In *Forty-first International Conference on Machine Learning*, 2024.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL `http://jmlr.org/papers/v21/20-074.html`.

Dominik Roblek, Kevin Kilgour, Matt Sharifi, and Mauricio Zuluaga. Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms. In *Proc. Interspeech 2019*, pages 2350–2354, 2019.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Flavio Schneider, Ojasv Kamal, Zhijing Jin, and Bernhard Schölkopf. Mo\^ usai: Text-to-music generation with long-context latent diffusion. *arXiv preprint arXiv:2301.11757*, 2023.

BS Series. Algorithms to measure audio programme loudness and true-peak audio level. In *International Telecommunication Union Radiocommunication Assembly*, 2011.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Christian J. Steinmetz, Jordi Pons, Santiago Pascual, and Joan Serrà. Automatic multitrack mixing with a differentiable mixing console of neural audio effects, 2020.

Alexander Tong, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Kilian FATRAS, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. In *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

Apoorv Vyas, Bowen Shi, Matthew Le, Andros Tjandra, Yi-Chiao Wu, Baishan Guo, Jiemin Zhang, Xinyue Zhang, Robert Adkins, William Ngan, et al. Audiobox: Unified audio generation with natural language prompts. *arXiv preprint arXiv:2312.15821*, 2023.

Yuancheng Wang, Zeqian Ju, Xu Tan, Lei He, Zhizheng Wu, Jiang Bian, et al. Audit: Audio editing by following instructions with latent diffusion models. *Advances in Neural Information Processing Systems*, 36:71340–71357, 2023.

Alec Wright and Vesa Välimäki. Perceptual loss function for neural modelling of audio systems, 2019.

Shih-Lun Wu, Chris Donahue, Shinji Watanabe, and Nicholas J Bryan. Music controlnet: Multiple time-varying controls for music generation. *arXiv preprint arXiv:2311.07069*, 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.

Yixiao Zhang, Yukara Ikemiya, Gus Xia, Naoki Murata, Marco Martínez, Wei-Hsiang Liao, Yuki Mitsufuji, and Simon Dixon. Musicmagus: Zero-shot text-to-music editing via diffusion models. *arXiv preprint arXiv:2402.06178*, 2024.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

Alon Ziv, Itai Gat, Gael Le Lan, Tal Remez, Felix Kreuk, Jade Copet, Alexandre Défossez, Gabriel Synnaeve, and Yossi Adi. Masked audio generative modeling. In *The Twelfth International Conference on Learning Representations*, 2023.

# A Appendix

## A.1 MELODYFLOW key components

### A.1.1 Latent audio representation

Our audio codec derives from EnCodec [Défossez et al., 2022] with additional features from the Descript Audio Codec (DAC) [Kumar et al., 2024] (snake activations, band-wise STFT discriminators) and [Evans et al., 2024a] (VAE bottleneck, perceptual weighting). A convolutional auto-encoder encodes the raw waveform into a sequence of latent bottleneck representations, its frame rate function of the convolution strides. Audio fidelity is enforced by multi-scale STFT reconstruction losses (both genuine and adversarial). Although [Vyas et al., 2023] have successfully trained a flow matching model on continuous latent representations extracted from the encoder of a discrete audio codec (before the RVQ layer), we argue that the latent space of such a codec is not the best suited for continuous modeling. We show in section A.3.3 that a VAE quantizer-free codec can operate at a much lower frame rate than its discrete equivalent for a better level of fidelity, reducing the cost of generation model training and inference by essentially the same factor.

### A.1.2 Conditional flow matching model

Given an audio sample $\mathbf{a} \in \mathbb{R}^{D \times f_s}$, a sequence $\mathbf{x} \in \mathbb{R}^{L \times d}$ of latent representations is extracted by the neural codec. Flow matching models optimal transport paths that map a sequence $\epsilon \in \mathbb{R}^{L \times d} \sim \mathcal{N}(0, I)$ to $\mathbf{x}$ trough a linear transformation - function of the flow step $t$ - following equation 1.

$$\mathbf{z}_t = t\mathbf{x} + (1 - t)\epsilon, t \in [0, 1] \tag{1}$$

During training, $t$ is randomly sampled and the neural network $\Theta$ is trained to estimate the derivative $d\mathbf{z}_t/dt$ conditioned on $t$ and a text description $c$.

$$d\mathbf{z}_t/dt = v_\Theta(\mathbf{z}_t, t, c) = \mathbf{x} - \epsilon \tag{2}$$

By design, after training, the model can be used with any ODE solver to estimate $\mathbf{x} = \mathbf{z_1}$ given $\epsilon = \mathbf{z_0}$ (and vice versa), and a text description. The text-to-music inference happens as such: starting from a random noise vector $\epsilon \in \mathbb{R}^{L \times d} \sim \mathcal{N}(0, I)$ and a text description $c$ of the expected audio the ODE solver is run from $t = 0$ to $t = 1$ to estimate the most likely sequence of latents $\mathbf{x}_{generated}$.

$$\mathbf{x}_{generated} = \mathbf{ODE}_{0 \to 1}(\epsilon, c) \tag{3}$$

After the latents have been estimated they are fed to the codec decoder to materialize the waveform. It was shown in [Kingma and Gao, 2024] that the flow step sampling density during training plays an important role in model performance. In our implementation we chose to sample $t$ from a logit-normal distribution. Logit-normal sampling was indeed originally proposed in [Karras et al., 2022] for v-prediction and also showed promising results in [Esser et al., 2024], both for the same task of text-to-image generation.

### A.1.3 Text-guided editing through latent inversion

Due to the bijective nature of the FM formulation (where given a text condition, each latent sequence is mapped to a single noise vector), the model is compatible with existing latent inversion methods such as DDIM inversion [Song et al., 2020]. Given the latent representation $\mathbf{x}_{orig}$ of an existing audio with an optional accompanying caption $c \in \{\varnothing, c_{orig}\}$, the model can serve to estimate its corresponding noise (or intermediate) representation $\mathbf{z}_{t_{edit}} = \mathbf{ODE}_{t_{edit} \leftarrow 1}(\mathbf{x}_{orig}, c)$ by running the ODE solver in the backward direction until an intermediary time step $t_{edit}$. Given the intermediary representation $\mathbf{z}_{t_{edit}}$, the ODE forward process can be conditioned on a new text description $c_{edit}$ that materialises the editing prompt: $\mathbf{x}_{edit} = \mathbf{ODE}_{t_{edit} \to 1}(\mathbf{z}_{t_{edit}}, c_{edit})$. A good inversion process should accurately reconstruct the input when $c_{edit} = c_{orig}$, as shown in equation 4.

$$\mathbf{x}_{edit} = \mathbf{ODE}_{t_{edit} \to 1}(\mathbf{ODE}_{t_{edit} \leftarrow 1}(\mathbf{x}_{orig}, c \in \{\varnothing, c_{orig}\}), c_{orig}) \approx \mathbf{x}_{orig} \tag{4}$$

In such case when swapping $c_{orig}$ for $c_{edit}$ in the $t_{edit} \to 1$ forward direction, the expectation is for the generated audio to preserve some consistency with the original while being faithful to the editing prompt. However in practice it was observed by [Mokady et al., 2023] that DDIM inversion suffers from poor editability due to the classifier free guidance.

### A.1.4 Adapting ReNoise for Flow Matching

The theoretical grounds of Flow Matching are about estimating straight trajectories between noise and data samples, which helps in achieving efficient inference. However in practice those trajectories are never completely straight and naive DDIM inversion suffers from two problems.

1. When running the ODE solver, any pair of successive $(\mathbf{z}_{t_1}, \mathbf{z}_{t_2})$ along the inversion path usually has estimated velocities $v_\Theta(\mathbf{z}_{t_1}, t_1, c) \neq v_\Theta(\mathbf{z}_{t_2}, t_2, c)$. Building a fully reversible inversion path requires estimating $\mathbf{z'}_{t_2}$ such that $v_\Theta(\mathbf{z}_{t_1}, t_1, c) \approx v_\Theta(\mathbf{z'}_{t_2}, t_2, c)$, for example using the convergence property demonstrated by [Garibi et al., 2024].

2. The distribution of predicted velocities tends to shift away from that of training, partially due to the classifier free guidance. One way for overcome this is to regularize them via gradient descent. Given a manually constructed $\tilde{\mathbf{z}}_t = \mathbf{x}t + \epsilon(1-t)$ and a real $\mathbf{z}_t$ along the inversion trajectory, and corresponding predictions $v_\Theta(\mathbf{z}_t, t, c)$ and $\tilde{v}_\Theta(\tilde{\mathbf{z}}_t, t, c)$. We arrange model predictions in 4x4 patches and compute the average Kullback-Leibler (KL) divergence $\mathcal{L}_{patchKL}$ between corresponding patches.

Our adaptation of the ReNoise inversion algorithm for flow matching is detailed in algorithm 1.

**Algorithm 1** Proposed regularized FM inversion

```
Input: Sequence of audio latents $\mathbf{x}$. Number of ODE backward steps $S$. Original text description
    $c \in \{\varnothing, c_{orig}\}$. ReNoise iteration steps $K$ with weights $\{w_k\}_{k=1}^K$, KL regularization weight $\lambda_{KL}$.
Output: A noisy latent $\mathbf{z}_{T_{edit}}$ such that $\mathbf{ODE}_{T_{edit} \to 1}(\mathbf{z}_{T_{edit}}, c_{orig}) \approx \mathbf{x}$.
  $\Delta t \leftarrow (1 - T_{edit})/S$
  for $t = 1, 1 - \Delta t, \ldots, T_{edit} + \Delta t$ do
      $\mathbf{z}_{t-\Delta t}^{(0)} \leftarrow \mathbf{z}_t$
      for $k = 1, \ldots, K$ do
          $\delta \leftarrow v_\Theta(\mathbf{z}_{t-\Delta t}^{(k-1)}, t - \Delta t, c)$
          if $w_k > 0$ then
              sample $\epsilon \sim \mathcal{N}(0, I)$
              $\tilde{\mathbf{z}}_{t-\Delta t}^{(k-1)} \leftarrow \mathbf{x}t + \epsilon(1-t)$
              $\tilde{\delta} \leftarrow v_\Theta(\tilde{\mathbf{z}}_{t-\Delta t}^{(k-1)}, t - \Delta t, c)$
              $\delta \leftarrow \delta - \lambda_{KL} \nabla_\delta \mathcal{L}_{patchKL}(\delta, \tilde{\delta})$
          end if
          $\mathbf{z}_{t-\Delta t}^{(k)} \leftarrow \mathbf{z}_t - \delta \Delta t$
      end for
      $\mathbf{z}_{t-\Delta t} \leftarrow \frac{\sum_{k=1}^K w_k \mathbf{z}_{t-\Delta t}^{(k)}}{\sum_{k=1}^K w_k}$
  end for
  return $\mathbf{z}_{T_{edit}}$
```

## A.2 Experimental setup

### A.2.1 Audio latent representation

Our compression model implementation is that of Copet et al. [2024] enhanced by band-wise discriminators and snake activations from [Kumar et al., 2024], perceptual weighting [Wright and Välimäki, 2019], VAE bottleneck and multi resolution STFT reconstruction loss from [Evans et al., 2024a]. We train a mono 32 kHz codec at 20 Hz frame rate and another one supporting stereo 48 kHz audio at 25 Hz. The bottleneck dimension is of 128. Both are trained on one-second random audio crops for 200K steps, with a constant learning rate of 0.0003, AdamW optimizer and loss balancer of [Défossez et al., 2022]. Stereo codecs are trained with sum and difference loss [Steinmetz et al., 2020]. The bottleneck layer statistics are tracked during training (dimension-wise) for normalization prior to FM model training.

### A.2.2 Flow matching model

MELODYFLOW is a diffusion transformer that follows Esser et al. [2024] configurations where each head dimension is of 64 and the model has the same number of heads and layers (either 18 or 24). Model implementation is that of `audiocraft`[3] but adapted for FM following [Vyas et al., 2023]: U-shaped skip connections are added along with linear projections applied after concatenation with each transformer block output Bao et al. [2023]. The model is conditioned via cross attention on a T5 representation [Raffel et al., 2020] computed from the text description of the audio, using 20% dropout rate during training in anticipation for the classifier free guidance applied at inference. Cross attention masking is used to properly adapt to the text conditioning sequence length of each sample within a batch and we use zero attention for the model to handle unconditional generation transparently. No prepossessing is applied on text data and we only rely on the descriptions (additional annotations tags such as musical key, tempo, type of instruments, etc. are discarded, although they also sometimes appear in the text description). The flow step is injected following [Hatamizadeh et al., 2023]. Minibatch coupling is computed with `torch-linear-assignement`[4]. MELODYFLOW-small (resp. MELODYFLOW-medium) is trained on latent representation sequences of 32 kHz mono (resp. 48 kHz stereo) segments of 10 (resp. 30) seconds, encoded at 20 Hz frame rate (resp. 25 Hz). MELODYFLOW-small (resp. MELODYFLOW-medium) is trained for 240k (resp. 120k) steps with AdamW optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay of 0.1 and gradient clipping at 0.2), a batch size of 576 and a cosine learning rate schedule with 4000 warmup steps. Additionally, we update an exponential moving average of the model weights ever 10 steps with a decay of 0.99. Each model is trained on 8 H100 96GB GPUs with `bfloat16` mixed precision and FSDP [Zhao et al., 2023]. MELODYFLOW-small requires 3 days and MELODYFLOW-medium 6 days of training.

### A.2.3 LLM-assisted editing prompt generation

For editing prompts design we prompted the LLama-3 large language model Dubey et al. [2024] to modify the original descriptions by targeting genre swapping. Edited descriptions were then manually verified to ensure their plausibility and coherence. As an example, given the original description *This is a lush indie-folk song featuring soaring harmony interplay and haunting reverb-y harmonica*, the resulting editing prompt is *This is a lush Indian classical-inspired song featuring soaring harmony interplay and haunting reverb-y bansuri flute*.

### A.2.4 Subjective evaluation form

A screenshot of the music subjective evaluation form is shown in the Figure 3.

### A.3 Text-to-music generation experiments

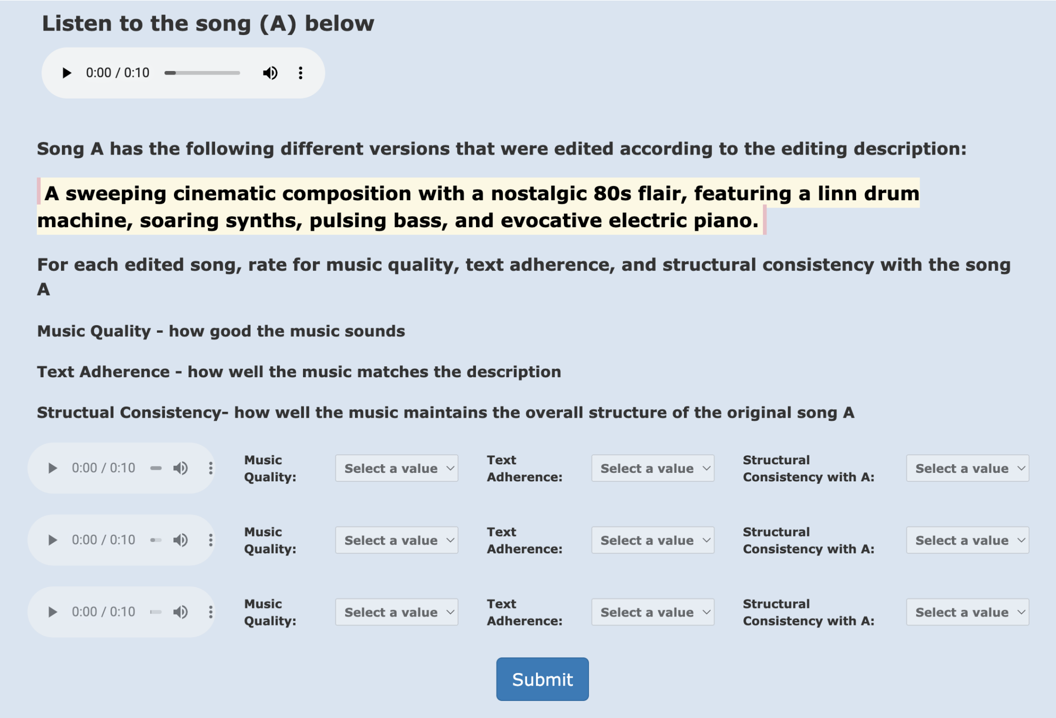
Text-to-music generation performance is reported in table 3. For text-to-music qualitative evaluations we compare MELODYFLOW to three baselines: MUSICGEN, AUDIOLDM 2 and STABLE-AUDIO. For MUSICGEN, AUDIOLDM 2 we use the available open source implementations and for STABLE-AUDIO we use the public API (as of Wed. May 14 2024, AudioSpark 2.0 model version). MELODYFLOW achieves comparable performance with MUSICGEN, both lagging slightly behind STABLE-AUDIO in terms of human preference. We do not report objective metrics on STABLE-AUDIO as none were reported on the full MusicCaps benchmark Evans et al. [2024a]. MELODYFLOW achieves remarkable efficiency with only 64 inference steps.

### A.3.1 Classifier-free guidance

In the Figure 4a we report the in-domain evaluation FAD as a function of the classifier-free guidance factor in the text-to-music generation setting. Throughout the paper we use a classifier-free guidance factor of 5.0.

[3] `https://github.com/facebookresearch/audiocraft`

[4] `https://github.com/ivan-chai/torch-linear-assignment`



Listen to the song (A) below

Song A has the following different versions that were edited according to the editing description:

**A sweeping cinematic composition with a nostalgic 80s flair, featuring a linn drum machine, soaring synths, pulsing bass, and evocative electric piano.**

For each edited song, rate for music quality, text adherence, and structural consistency with the song A

Music Quality - how good the music sounds

Text Adherence - how well the music matches the description

Structual Consistency- how well the music maintains the overall structure of the original song A

Figure 3: Music editing subjective evaluation form. Given the original song A, raters are asked to evaluate three different edits of A, on the three following axes: quality, text adherence, consistency.

Table 3: Comparison to text-to-music baselines. We report the original objective metrics for AudioLDM 2 and MusicGen. For subjective evaluations we report mean and CI95.

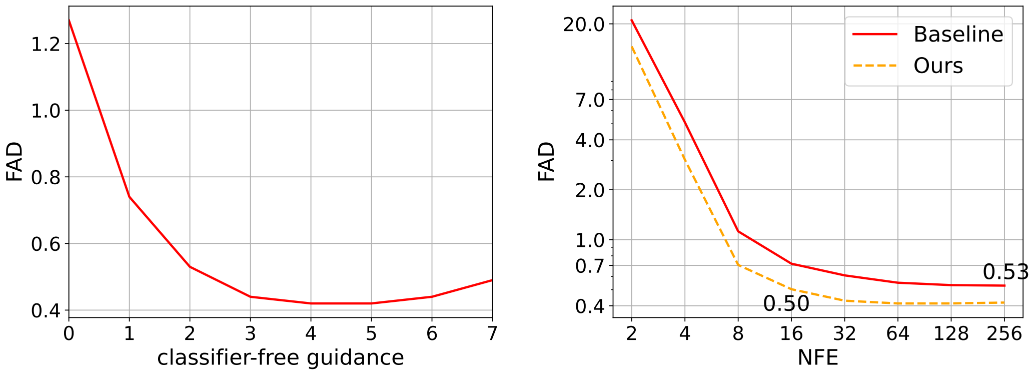
| Model | $\text{FAD}_{\text{vgg}}$ ↓ | KL ↓ | $\text{CLAP}_{\text{sim}}$ ↑ | Ovl. ↑ | Rel. ↑ | # Steps |
|---|---|---|---|---|---|---|
| Reference | - | - | - | 3.67±0.10 | 4.04±0.10 | - |
| AudioLDM 2 | 3.1 | 1.20 | 0.31 | 2.79±0.08 | 3.40±0.08 | 208 |
| MusicGen-small | 3.1 | 1.29 | 0.31 | - | - | 1500 |
| MusicGen-medium | 3.4 | 1.23 | 0.32 | 3.40±0.08 | 3.79±0.07 | 1500 |
| Stable-Audio | - | - | - | 3.67±0.08 | 3.89±0.07 | 100 |
| MelodyFlow-small | 2.8 | 1.27 | 0.33 | 3.27±0.08 | 3.83±0.08 | 64 |
| MelodyFlow-medium | 3.5 | 1.30 | 0.31 | 3.41±0.08 | 3.77±0.07 | 64 |

### A.3.2 Inference efficiency

In the Figure 4b we present the measured FAD as a function of inference steps for both the baseline and final version of MelodyFlow. Not only is MelodyFlow achieving better performance overall but it is able to outperform the baseline with 16 times fewer NFE (e.g. inference steps).

### A.3.3 Codec bottleneck and framerate

We ablate on the bottleneck choice for a fixed frame rate of 50Hz by comparing RVQ ([Copet et al., 2024] setting), VAE and identity in Table 4. From the codec perspective our results indicate optimal reconstruction performance with the identity or VAE bottlenecks followed by RVQ. The same ranking applies for SI-SDR [Le Roux et al., 2019]. Regarding FM model performance we report a decreasing in-domain FAD in the favor the VAE. Ablating on the VAE codec frame rate shows that 5 Hz is comparable to 50 Hz codec with RVQ regarding codec performance. FM model performance is closer to that of 50 Hz codec with identity bottleneck. For MelodyFlow-small we chose to use 20 Hz.



(a) Classifier-free guidance.

(b) Generation FAD vs. NFE.

Figure 4: Text-to-music generation quality (FAD) as a function of the classifier-free guidance factor (Figure 4a) and inference steps (Figure 4b). The baseline of the Figure 4b is the FM model architecture of [Le et al., 2024] but retrained on our music latents. The combination of our flow matching design choices enable faster generation for a given efficiency budget or better overall quality.

Table 4: Codec bottleneck and framerate ablation for 32 kHz mono audio. Both compression and generative model performances are reported on the IN-DOMAIN test set.

| BOTTLENECK | FRAME RATE (HZ) | $\text{STFT}_{loss}\downarrow$ | SI-SDR$\uparrow$ | $\text{FAD}_{vgg}\downarrow$ |
|---|---|---|---|---|
| ∅ | 50 | 0.35 | 18.5 | 0.68 |
| RVQ | 50 | 0.55 | 4.4 | 0.55 |
| VAE | 50 | 0.34 | 18.1 | 0.48 |
|  | 20 | 0.44 | 12.9 | 0.47 |
|  | 5 | 0.53 | 3.5 | 0.67 |

Table 5: Ablation on L-mask and stereo for MELODYFLOW-large. Each variant is trained on 30s audio segments encoded with a 25 Hz frame rate codec trained on 48 kHz audio.

| CHANNELS | $\text{STFT}_{loss}\downarrow$ | SI-SDR$\uparrow$ | L-MASK | $\text{FAD}_{10s}\downarrow$ | $\text{FAD}_{30s}\downarrow$ |
|---|---|---|---|---|---|
| 2 | 0.40 | 12.48 | ✓ | 0.59 | 0.65 |
|  |  |  | ✗ | 1.48 | 0.65 |
| 1 | 0.39 | 13.34 | ✓ | 0.49 | 0.59 |

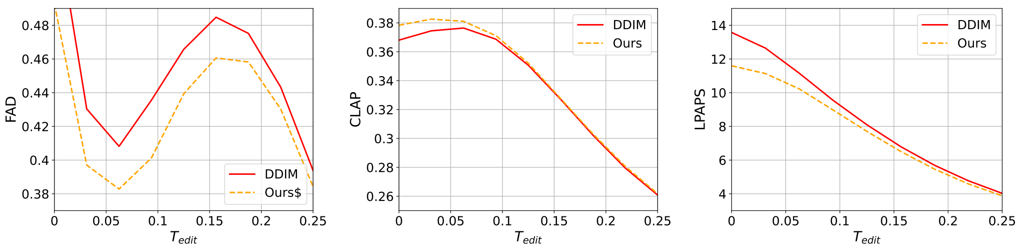
#### A.3.4 Scaling for high-fidelity and duration versatility.

The table 5 reports the impact of moving from mono to stereo with the same MELODYFLOW-medium model size (1B parameters). We also report the effect of applying a L-shaped attention mask during model training to support durations shorter than 30 seconds during inference. For each sequence of length L, we randomly select a segment boundary within the range $[0, L]$. Positions before the boundary can only attend to themselves in the self-attention, while positions after it attend to the entire sequence. The in-domain FAD is reported for 10s and 30s generated segments. Our results indicate that the L-mask helps supporting versatile duration with no penalty on full-length segments, unlocking faster inference for shorter segments. However moving from mono to stereo only slightly affects the generative model performance, likely due to our mono-based evaluation.

### A.4 Additional music editing ablations

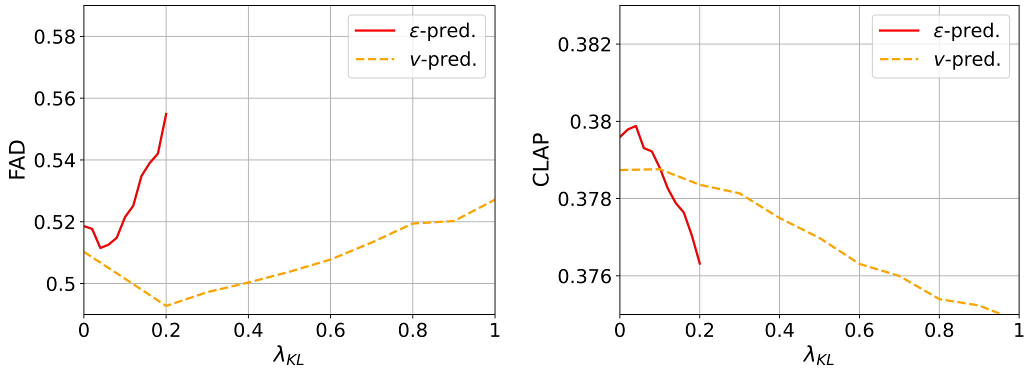
#### A.4.1 Flow matching target inversion timestep

In the Figures 5a, 5b and 5c we report music editing objective metrics as a function of $T_{edit}$, comparing naive DDIM inversion with MELODYFLOW. For MELODYFLOW inversion we ablate



(a) Editing FAD$_{edit}$ vs. $T_{edit}$.
(b) Editing CLAP$_{edit}$ vs. $T_{edit}$.
(c) Editing LPAPS vs. $T_{edit}$.

Figure 5: Music editing quality as a function of the target inversion step $T_{edit}$. We report FAD$_{edit}$ (Figure 5a), CLAP$_{edit}$ (Figure 5b) and LPAPS (Figure 5c) objective metrics.



(a) FAD$_{edit}$ as a function of $\lambda_{KL}$.
(b) CLAP$_{edit}$ as a function of $\lambda_{KL}$.

Figure 6: Effect of the loss regularization weight $\lambda_{KL}$ on the quality (Figure 6a) and text-adherence (Figure 6b) of music editing. Noise and velocity prediction are compared, with or without the original text description $c_{orig}$.

on the original text description used in the backward ODE process ($c_{T_{edit} \longleftarrow 1} \in \{\varnothing, c_{orig}\}$). DDIM with text conditioning is not shown due to its instability.We observe that our proposed regularized inversion outperforms DDIM inversion, irrespective of the use of the text conditioning in the inversion process. Using $c_{orig}$ to condition the inversion shows better text adherence (higher CLAP$_{edit}$) at the detriment of quality (higher FAD$_{edit}$). Consistency with the original song is generally much higher (lower LPAPS) than when using DDIM inversion, which correlates with our listening tests. The S-shaped FAD curves of the Figure 5a indicate an inversion target optimum around $T_{edit} = 0.06$.

### A.4.2 Regularized inversion ablation

During latent inversion the FM model predicts the velocity $v_\Theta(\mathbf{z}_t, t, c) = \mathbf{x} - \epsilon$. The predictions are regularized using the Algorithm 1 using a weighted KL patch-wise divergence loss $\mathcal{L}_{patchKL}$. In the Figures 6a and 6b we ablate on the divergence loss weight $\lambda_{KL}$ for $T_{edit} = 0.04$, using $S = 25, K = 4, w_k = k - 1$. Since during latent inversion we know the original latent $\mathbf{x}_{orig}$, the model prediction can be rewritten as $\epsilon$-prediction by computing $\epsilon = v_\Theta(\mathbf{z}_t, t, c) - \mathbf{x}_{orig}$. In that scenario we can apply the exact ReNoise inversion method of [Garibi et al., 2024], using 10 iterations of noise de-correlation (shown as $\epsilon$-prediction in the Figures). Whether the model prediction is expressed as noise or velocity, the Figures show that an optimum can be achieved around $\lambda_{KL} = 0.15$ for velocity prediction and around $0.05$ for noise prediction. Overall the quality is better (lower FAD$_{edit}$ in the Figure 6a) when directly regularizing the velocity prediction. In both cases we observe a higher CLAP$_{edit}$ in the Figure 6b when the original text description $c_{orig}$ conditions the inversion process, confirming better text-adherence. This happens at the expense of a higher FAD$_{edit}$ compared with unconditional inversion.

### A.5 Related work

**Audio representation** Recent advancements in neural codecs have seen the application of VQ-VAE on raw waveforms, incorporating a RVQ bottleneck as demonstrated in Zeghidour et al. [2021], Défossez et al. [2022], later refined as per Kumar et al. [2024]. [Evans et al., 2024a] proposed a modification to this approach by replacing the RVQ with a VAE bottleneck to enhance the modeling of continuous representations. In addition, several recent audio generative models have adopted Mel-Spectrogram latent representations, coupled with a vocoder for reconstruction, as shown in the works of [Ghosal et al., 2023, Liu et al., 2023, Le et al., 2024]. Furthermore, Défossez et al. [2022] introduced an additional layer of complexity by incorporating quantization to support discrete representation on top of the continuous representation.

**Text-to-music generation** Models that operate on discrete representation are presented in the works of [Agostinelli et al., 2023, Copet et al., 2024, Ziv et al., 2023]. Agostinelli et al. [2023] proposed a representation of music using multiple streams of tokens, which are modeled by a cascade of transformer decoders conditioned on a joint textual-music representation [Huang et al., 2022b]. Copet et al. [2024] introduced a single-stage language model that operates on streams of discrete audio representations, supporting both 32 kHz mono and stereo. Ziv et al. [2023] replaced the language model with a masked generative single-stage non-autoregressive transformer. Schneider et al. [2023], Huang et al. [2023], Liu et al. [2023] use diffusion models. Schneider et al. [2023] utilized diffusion for both the generation model and the audio representation auto-encoder. Liu et al. [2023] trained a foundational audio generation model that supports music with latent diffusion, conditioned on autoregressively generated AudioMAE features [Huang et al., 2022a]. Evans et al. [2024a,b] proposed an efficient long-form stereo audio generation model based on the latent diffusion of VAE latent representations. This model introduced timing embeddings conditioning as a method to better control the content and length of the generated music.

**Music editing** Lin et al. [2024] proposed a parameter-efficient fine-tuning method for autoregressive language models to support music inpainting tasks. Garcia et al. [2023] developed a masked acoustic modeling approach for music inpainting, outpainting, continuation and vamping. Wu et al. [2023] fine-tuned a diffusion-based music generation model with melody, dynamics and rhythm conditioning. Novack et al. [2024] is a fine-tuning free framework for controlling pre-trained text-to-music diffusion models at inference-time via initial noise latent optimization. Zhang et al. [2024] investigated zero-shot text-guided music editing with conditional latent space and cross attention maps manipulation. Manor and Michaeli [2024] employs DDPM inversion [Huberman-Spiegelglas et al., 2023] for zero-shot unsupervised and text-guided audio editing.