Zero-Shot Unsupervised and Text-Based Audio Editing Using DDPM Inversion

Hila Manor [1] **Tomer Michaeli** [1]

Abstract

Editing signals using large pre-trained models, in a zero-shot manner, has recently seen rapid advancements in the image domain. However, this wave has yet to reach the audio domain. In this paper, we explore two zero-shot editing techniques for audio signals, which use DDPM inversion with pre-trained diffusion models. The first, which we coin *ZEro-shot Text-based Audio (ZETA)* editing, is adopted from the image domain. The second, named *ZEro-shot UnSupervized (ZEUS)* editing, is a novel approach for discovering semantically meaningful editing directions without supervision. When applied to music signals, this method exposes a range of musically interesting modifications, from controlling the participation of specific instruments to improvisations on the melody. Samples, code, and the full version of this paper can be found on the project web page.

1. Introduction

Media creation and editing have seen a dramatic transformation with the recent advancements in text-based generative models, particularly those based on denoising diffusion models (DDMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020). In the audio domain, text-based editing has only very recently started gaining traction. Recent works adapted methods that were originally explored in the image domain (Brooks et al., 2023; Gal et al., 2022; Kim et al., 2022; Kawar et al., 2023; Ruiz et al., 2023; Zhang et al., 2023b). Those either train models from scratch for specific editing tasks (Copet et al., 2023; Han et al., 2023; Wang et al., 2023), or apply test-time optimization (Paissan et al., 2023; Plitsis et al., 2024). To date, zero-shot editing for audio signals has only been illustrated in the AudioLDM work (Liu et al., 2023a), using the naive SDEdit (Meng et al., 2021) approach.

In this paper we explore two approaches for zero-shot audio editing with pre-trained audio DDMs, one based on *text guidance* and the other based on semantic perturbations that are found in an *unsupervised* manner. Both approaches rely on the recently introduced edit-friendly DDPM inversion method (Huberman-Spiegelglas et al., 2024), which we use for extracting latent noise vectors corresponding to the source signal. To generate the edited signal, we use those noise vectors in a DDPM sampling process (Ho et al., 2020), while drifting the diffusion towards the desired edit. Our zero-shot text-guided audio (ZETA) editing technique allows a wide range of manipulations, from changing the style or genre of a musical piece to changing specific instruments in the arrangement (Fig. 1(c),(d)), all while maintaining high perceptual quality and semantic similarity to the source signal. Our zero-shot unsupervised (ZEUS) technique allows generating *e.g.*, interesting variations in melody that adhere to the original key, rhythm, and style, but are impossible to achieve through text guidance (Fig. 1(a),(b)).

We compare our methods to state-of-the-art as well as to zero-shot editing approaches for audio (Copet et al., 2023; Meng et al., 2021; Song et al., 2021; Dhariwal & Nichol, 2021), in conjunction with the AudioLDM2 model (Liu et al., 2023b). We show that our approaches outperforms these methods in terms of generating semantically meaningful modifications, while remaining faithful to the original signal's structure.

2. Method

2.1. DDPM Inversion

Denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020) generate samples through an iterative process, which starts with a Gaussian noise vector $\mathbf{x}_T \sim \mathcal{N}(0, \boldsymbol{I})$ and gradually denoises it in T steps as

$$\mathbf{x}_{t-1} = \boldsymbol{\mu}_t(\mathbf{x}_t) + \sigma_t \mathbf{z}_t, \quad t = T, \dots, 1. \tag{1}$$

Here, $\{\mathbf{z}_t\}$ are standard Gaussian vectors, $\{\sigma_t\}$ is an increasing sequence of noise levels, and $\boldsymbol{\mu}_t(\mathbf{x}_t)$ is a linear function of $\hat{\mathbf{x}}_{0|t}$, which is the MSE-optimal prediction of a clean signal $\mathbf{x}_0$ from its noisy version

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(0, \boldsymbol{I}). \tag{2}$$

The coefficients $\{\bar{\alpha}_t\}$ monotonically decrease from 1 to 0.

[1] Faculty of Electrical and Computer Engineering, Technion – Israel Institute of Technology, Haifa, Israel. Correspondence to: Hila Manor <hila.manor@campus.technion.ac.il>.

Accepted by the Structured Probabilistic Inference & Generative Modeling workshop of ICML 2024, Vienna, Austria. Copyright 2024 by the author(s).

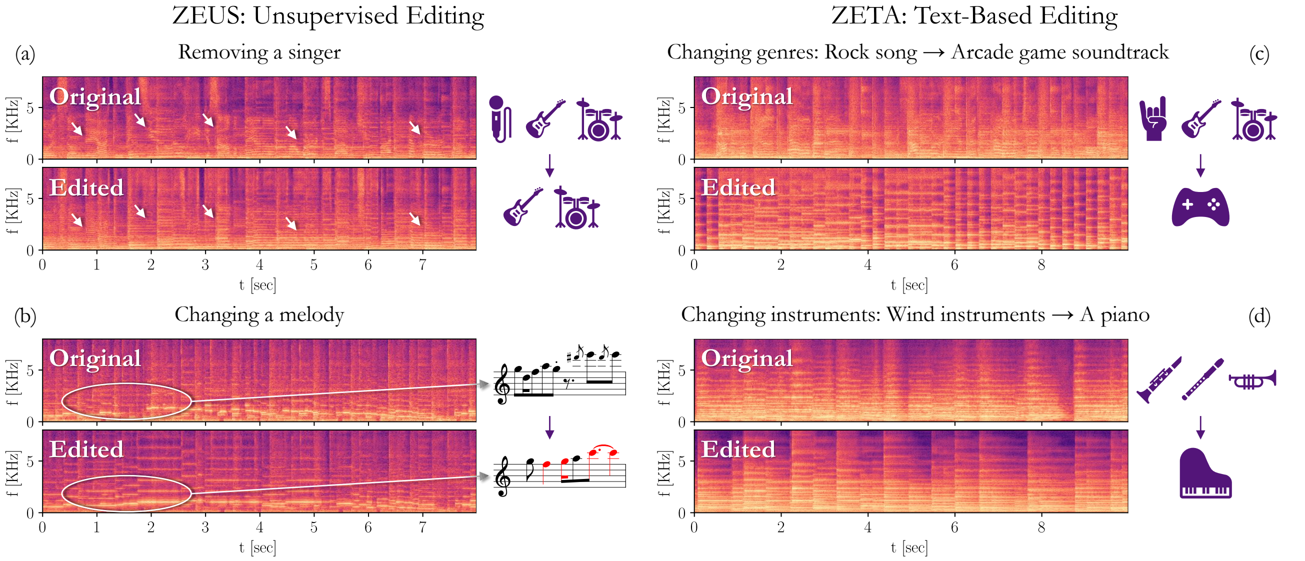


Figure 1. **Zero-shot audio editing.** We present two methods for editing audio signals using DDMs: ZEUS, a novel unsupervised approach (left), and ZETA, a text-based approach adopted from the image domain (right). Both methods can edit a variety of concepts from style to instrumentation. (a) The singer (curved pitches) is removed while the rest of the signal remains intact. (b) The melody notes change, reflected by a change in the dominant pitch. (c) The genre is changed, affecting the entire statistics of the spectrogram. (d) The instrumentation changes from a woodwind section to a piano, visible by the attack (abrupt starts) of the piano keys. All examples can be listened to in our examples page. For (c),(d), $T_{\text{start}} = 100, 70$, respectively (Sec. 2.2). For (a),(b), $T_{\text{start}} = 150, 200$, $t' = 115, 80$, $T_{\text{end}} = 1$, using the top 3 PCs (Sec. 2.3).

Rather than generating a synthetic signal, here we are interested in editing a real audio excerpt, $\mathbf{x}_0$. To do so, we start by extracting noise vectors $\{\mathbf{x}_T, \mathbf{z}_T, ..., \mathbf{z}_1\}$ that cause the sampling process (1) to generate the given signal $\mathbf{x}_0$ at $t = 0$. This is called *inversion*. We then use those noise vectors to sample a signal using (1) while steering the generation towards a desired edit effect, as we detail in Sections 2.2 and 2.3. To extract the noise vectors, we use the edit-friendly DDPM inversion method of Huberman-Spiegelglas et al. (2024). Please see App. C for more details.

We note that a diffusion process can be either performed in the raw waveform space or in some latent space (Rombach et al., 2022). In this work we utilize the pre-trained AudioLDM2 (Liu et al., 2023a;b) model, which works in a latent space. AudioLDM2 generates mel-spectograms conditioned on text. Those mel-spectograms are decoded into waveforms using HiFi-GAN (Kong et al., 2020).

2.2. ZETA: Text-Based Audio Editing

The first editing approach we consider uses text guidance. In this setting, our goal is to edit a real audio signal $\mathbf{x}_0$ by using a text prompt p_{edit} describing the desired result. Optionally, a user may also want to describe the original signal with some text prompt, p_{src}, so as to achieve a more fine-grained modification.

To achieve this goal, we adopt the method of Huberman-Spiegelglas et al. (2024), which has been only explored in the image domain. Here, we explore this approach in the context of text-to-audio models, and demonstrate its transferability to a new domain. Specifically, we start by inverting the signal $\mathbf{x}_0$ using DDPM Inversion. We do this while injecting to the denoiser network the prompt describing the source, p_{src}. This is illustrated in the left pane of Fig. 2. We then run the generative process (1) with the extracted noise vectors, while injecting the prompt p_{edit} describing the desired output (top-right pane of Fig. 2). In both directions, we use classifier-free guidance (Ho & Salimans, 2021) (CFG) for the text guidance. The noise vectors extracted from the source signal ensure that the generated signal has the same "coarse structure" as the source, while the change in the text conditioning affects more fine-grained features, and leads to the editing effect.

The balance between adhering to the target text and remaining loyal to the original signal can be controlled using two parameters. The first is the CFG strength factor. Increasing this parameter steers the generation more strongly towards the desired text at the expense of departing from the original signal. The second parameter is the timestep T_{start} from which we begin the generation process. The smaller it is, the more the edited signal remains consistent with the source signal, and the less it adheres to the target text. We explore its effect in more detail in App. I. In App. J we extend this editing technique to support user-provided target segments

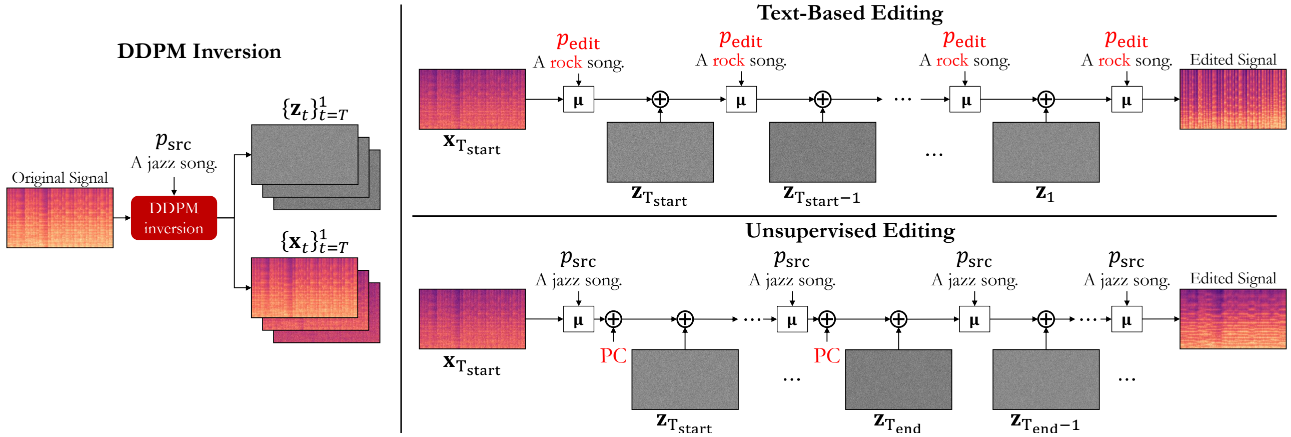


Figure 2. **Overview of our text-based and unsupervised editing methods.** We start by extracting the noise vectors corresponding to an input signal using DDPM inversion, optionally conditioned on a text prompt p_{src}. For the text-based editing approach, we then continue the reverse process with a different text prompt. For the unsupervised approach, we continue the reverse process when applying PCs calculated on the forward process. Red color shows what changed in the reverse process.

for editing. This allows more control over the resulting edited signal, which can be particularly important for editing lengthy audio data.

2.3. ZEUS: Unsupervised Editing

Editing using text guidance is limited by the expressiveness of the text prompt and by the model's language understanding. This is arguably very significant in music, where a user may want to generate *e.g.*, variations, improvisations, or modifications to the arrangement of the piece, which are virtually impossible to precisely describe by text. To support these kinds of edits, here we pursuit a different approach, which extracts in an unsupervised manner semantically meaningful editing directions in the noise space of the diffusion model. As we show, these directions can be used to perturb the generation process in multiple ways, enabling controllable semantic modifications to the signal.

As in Sec. 2.2, we start by performing edit-friendly DDPM inversion to extract noise vectors corresponding to $\mathbf{x}_0$, optionally using a text-prompt describing the signal, p_{src}. We then use those vectors in the sampling process (1), but with perturbations. Specifically, recall from Sec. 2.1 that each timestep t involves $\hat{\mathbf{x}}_{0|t}$, the MSE-optimal prediction of $\mathbf{x}_0$ from $\mathbf{x}_t$. This prediction, obtained from the denoiser, corresponds to the posterior mean $\mathbb{E}[\mathbf{x}_0|\mathbf{x}_t]$. Our approach is to perturb this posterior mean along the top principal components (PCs) of the posterior, *i.e.*, the top eigenvectors of the posterior covariance $\text{Cov}[\mathbf{x}_0|\mathbf{x}_t]$.

To compute the posterior PCs, we adapt the method of Manor & Michaeli (2024) for the generative pipeline. This work showed that the posterior covariance in Gaussian denoising is proportional to the Jacobian of the MSE-optimal denoiser. It further showed that extracting the top eigenvectors and eigenvalues of this Jacobian can be done using the subspace iteration method (Saad, 2011), where each iteration can be approximated using a single forward pass through the denoiser network. See App. D for an algorithm.

Having computed the posterior PCs $\{\mathbf{v}_{i|t'}\}$ and their corresponding eigenvectors $\{\lambda_{i|t'}\}$ at some timestep t', in App. D we show we can add or subtract each of them to the denoised signal $\hat{\mathbf{x}}_{0|t}$ at every timestep $t \in [T_{\text{start}}, \dots, T_{\text{end}}]$, multiplied by some factor, c_t, by simply modifying the generation process from Eq. (1) into

$$\mathbf{x}_{t-1} = \boldsymbol{\mu}_t(\mathbf{x}_t) + \gamma c_t \lambda_{i|t}^{1/2} \mathbf{v}_{i|t'} + \sigma_t \mathbf{z}_t, \quad t = T, \dots, 1, \tag{3}$$

where γ is a user-chosen parameter controlling the strength of the modification. We additionally explore in App. D two different approaches for choosing which PCs to add. Note that instead of adding a single PC, $\mathbf{v}_{i|t'}$, we can add a linear combination of PCs, thereby creating a new direction that changes multiple semantic elements at once.

3. Experiments

To evaluate our editing methods we used AudioLDM2 (Liu et al., 2023b) as the pre-trained model. In our text-based editing experiments we compare to MusicGen (Copet et al., 2023) conditioned on melody using their medium checkpoint, and to DDIM inversion (Song et al., 2021; Dhariwal & Nichol, 2021) and SDEdit (Meng et al., 2021) using the same AudioLDM2 checkpoint as we use. DDIM inversion is typically applied for the entire diffusion process, *i.e.*, when T_{start} is set to T. In App. E we include a comparison to *Partial DDIM Inversion*, a version that applies the inversion only up to T_{start}. To evaluate our unsupervised editing method, we compare it to SDEdit, where we supply

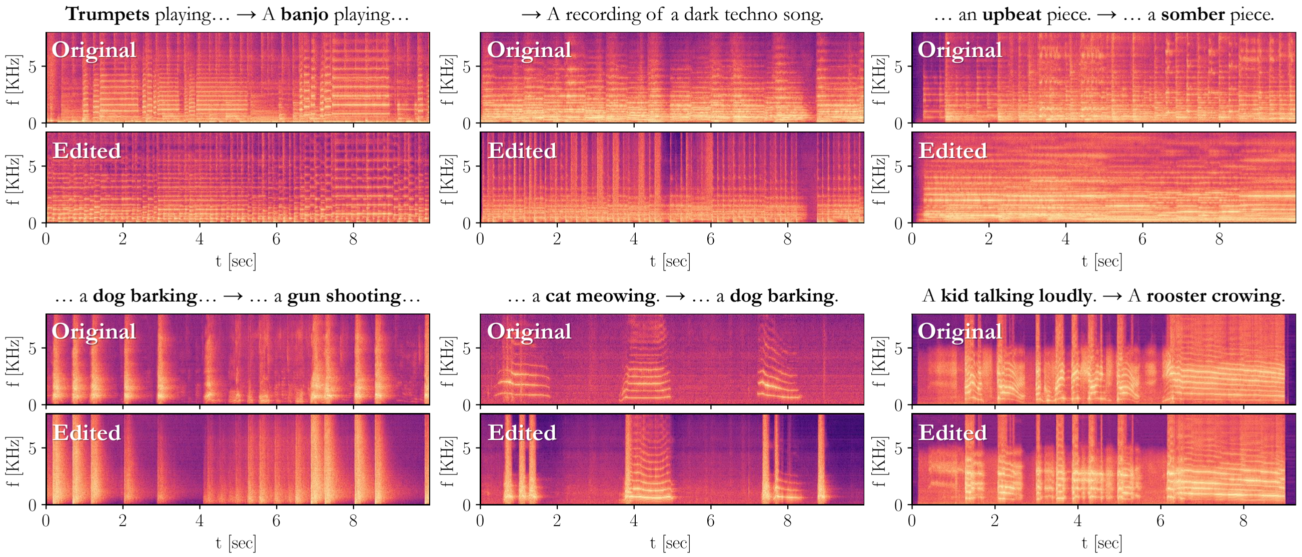


Figure 3. **Text-based zero-shot editing.** ZETA editing enables changing a plethora of elements in a signal, from the genre of a song, to the objects heard in a recording. All examples can be listened to in Sec. 1.1 of our examples page. The source and target text prompts are shown above the spectrograms, where bold marks changed text. The parameter T_{start} (Sec. 2.2), from top to bottom and left to right, is 90, 90, 80, 100, 150, 110. The first row of original-edited pairs used the music checkpoint of AudioLDM2, while the second used the large general checkpoint.

it with a prompt describing the source signal (rather than the desired edited signal). This baseline performs uncontrolled modifications to the signal, whose strength we choose using the starting timestep.

We do not compare to AUDIT (Wang et al., 2023) and InstructME (Han et al., 2023), which train a model specifically for editing purposes, as they did not share their code and trained checkpoints. Additionally, we do not compare to DreamBooth and Textual Inversion as demonstrated on audio by Plitsis et al. (2024), since they solve a different task – that of personalization. This task aims at learning a concept from a reference audio, rather than consistently modifying the input itself. Thus, personalization may allow *e.g.*, changing a genre, but cannot be used for fine-grained edits such as changing a specific instrument into a different one. In fact, as Plitsis et al. (2024) show, both methods have difficulty preserving the key and dynamics of the original piece, where textual inversion fails to even retain the same tempo. We encourage the reader to listen to our results and qualitative comparisons on our examples page.

Both of our methods incur a negligible memory overhead w.r.t. to the memory consumed by the diffusion model, and are comparable in inference speed to all evaluated competing methods. A detailed analysis can be found in App. G.

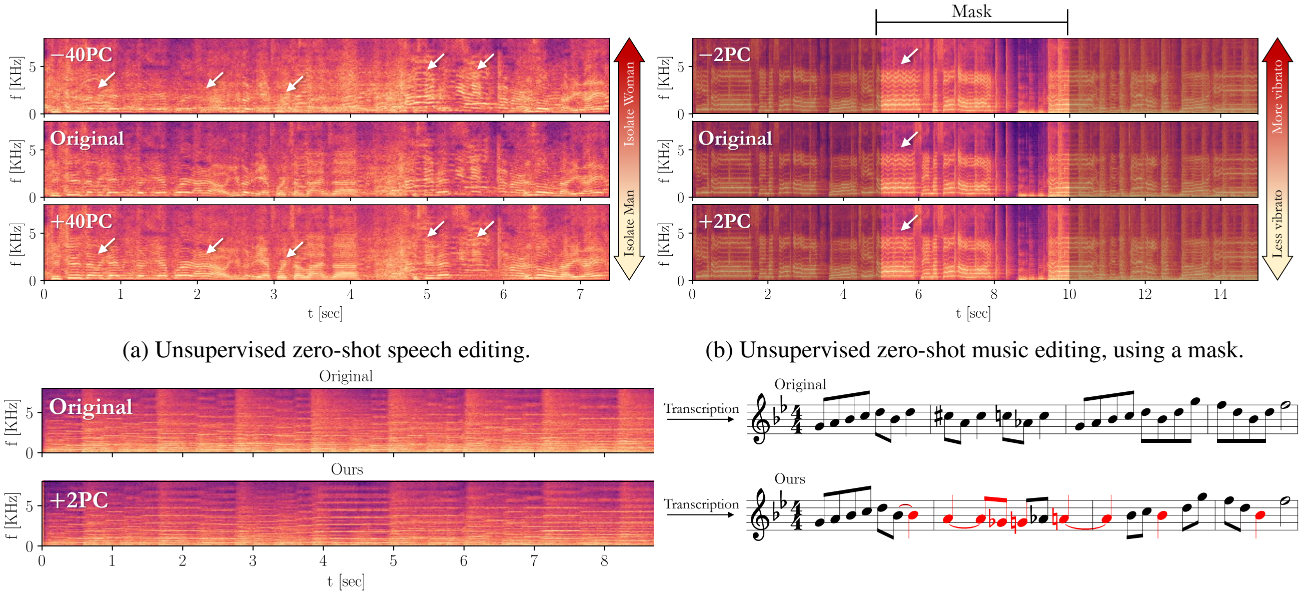
For evaluation we use the MusicDelta subset of the MedleyDB dataset (Bittner et al., 2014), comprised of 34 musical excerpts in varying styles and in lengths, and create and release a corresponding small but varied dataset of manual text-prompts, named *MedleyMDPrompts*, sub-sampled randomly for evaluation. More information can be found in App. F. Audio signals are taken from AudioSet (Gemmeke et al., 2017).

We quantitatively evaluate the results using three types of metrics; a CLAP (Wu et al., 2023; Chen et al., 2022) based score to measure the adherence of the result to the target-prompt (higher is better); LPAPS (Iashin & Rahtu, 2021; Paissan et al., 2023), an audio LPIPS (Zhang et al., 2018) measure to quantify the consistency of the edited audio relative to the source audio (lower is better); and FAD (Kilgour et al., 2019), an audio FID (Heusel et al., 2017) metric to measure the distance between two distributions of audio signals. More details, as well as hyper-parameters configurations, can be found in App. F. A discussion on the limitations of our methods is presented in App. B.

3.1. Text-Based Editing.

Results for different effects achieved with ZETA editing are shown in Fig. 1(c),(d) and Fig. 3. Text-based editing allows changing the global semantics of the signal, *e.g.*, by changing one instrument to another or by changing the genre of the song. All examples and more can be listened to in Sec. 1.1 of our examples page.

To quantitatively measure the adherence of the edited signals to the target prompt and their fidelity to the source excerpts, we plot the CLAP-LPAPS results for all methods in Fig. 5. For SDEdit and for our method we plot results for multiple



(a) Unsupervised zero-shot speech editing.

(b) Unsupervised zero-shot music editing, using a mask.

(c) Unsupervised melody improvisation.

Figure 4. **Unsupervised zero-shot editing.** Our ZEUS editing directions carry semantic meanings, ranging from separation of the persons in a conversation (a) to the vibrato of a singer (b) or a change in melody (c), all while retaining high semantic similarity to the source. Directions can be easily calculated and applied on a segment of the signal using a mask (c) (see App. J). All examples can be listened to in Sec. 1.2 of our examples page. In (a) we use $t' = t$, $T_{\text{start}} = 115$, $T_{\text{end}} = 95$, in (b) we fix $t' = 120$ and use $T_{\text{start}} = 150$, $T_{\text{end}} = 1$, and in (c) $t' = 95$, $T_{\text{start}} = 150$, $T_{\text{end}} = 50$. The PCs shown are the 1st ,3rd, and a combination of the first 3 PCs, respectively. The speech example uses the large checkpoint of AudioLDM2, without a source prompt, and the rest use the music checkpoint, with a source prompt randomly chosen from our prompts dataset.

T_{start} values. It is evident that MusicGen, which is trained to be conditioned on a chromagram of the input signal, does not enable transferring a concept in the same way as the zero-shot editing methods we explore. As can be seen, our method outperforms all other methods under any desired balance between fidelity and text-adherence. Qualitative comparisons can be listened to in Sec. 2.1 of our examples page. A visual comparison can be found in App. H.

User study. We additionally evaluated our method via a user study conducted through Amazon Mechanical Turk (AMT). AMT users were asked to answer a sequence of 16 questions, after completing a single practice question with provided feedback. In each question, first the original music signal was played to the user, following the reveal of the target text-prompt for editing. Next, our edited result and one edit from a random competing method were played in a random order. Users were instructed to select the edit that better matches the text prompt while keeping the rest of the essence of the original signal. The data used for evaluation consisted of 8-second long random segments from each of the songs in the MusicDelta subset, edited with a randomly selected target text-prompt from our MedleyMDPPrompts dataset. To allow fair comparison to MusicGen, we removed prompts that referred to singing. For our method and SDEdit we arbitrarily chose $T_{\text{start}} = 100$. Each of the competing methods was compared using 50 workers over two batches. Each sequence of 16 questions included a vigilance question, during which one of the compared signals was random noise. Out of 300 participants only 277 passed the vigilance test (*i.e.*, did not choose the random noise), and their results for all methods are reported in Fig. 6. As can be seen, our method was clearly preferred over all competing methods. More details on the user study can be found in App. M.

3.2. Unsupervised Editing

Next, we perform experiments using ZEUS editing. The strength of the modification is controlled by the parameter γ. As can be seen in Figs. 1(a),(b), and 4, and can be listened to in Sec. 1.2 of our examples page, the modifications can range from effects like voice emphasis, to more stylistic changes *e.g.*, in a singer's vibrato. An interesting change is more apparent in music, where the semantic editing directions can take the form of an improvisation on the original piece, obtained by changing the melody, as shown in Fig. 4c.

The PCs of the posterior covariance convey the uncertainty of the denoising model at the current timestep. The synthesis process is inherently more uncertain at earlier timesteps in the sampling process (*i.e.*, at larger t). Therefore, the

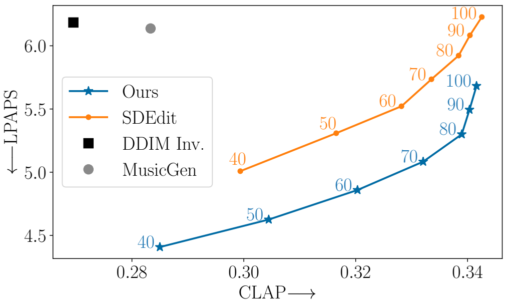


Figure 5. **Adherence to target prompt vs. fidelity to the original signal.** The plot compares MusicGen (Copet et al., 2023), SDEdit (Meng et al., 2021), DDIM Inversion (Song et al., 2021; Dhariwal & Nichol, 2021) and our method over the MusicDelta subset in MedleyDB (Bittner et al., 2014), using our prompts dataset. Our method and SDEdit are shown with T_{start} values ranging between 40 and 100. Our results achieve lower (better) LPAPS for any level of CLAP (higher is better), indicating a good balance between text adherence and signal fidelity.

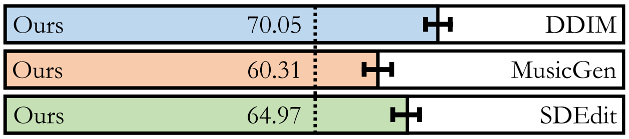


Figure 6. **User study.** We report the percentage of users who preferred our method over DDIM inversion (Song et al., 2021; Dhariwal & Nichol, 2021), MusicGen (Copet et al., 2023) and SDEdit (Meng et al., 2021), when answering which edited signal better matches the text prompt, while keeping the rest of the essence of the original signal. The error bars indicate the calculated Wilson score interval.

extracted directions $\{\mathbf{v}_{i|t}\}$ generally exhibit more global changes spread over larger segments of the samples for earlier timesteps, and more local changes for later timesteps (see App. K for further discussion).

Qualitative comparisons can be listened to in Sec. 2.2 of our examples page. Here for a quantitative analysis we measure the FAD to two different datasets. The first is the original Music Delta subset. This measures the strength of the edit, as it quantifies the deviation from the original distribution. The second is the FMA-pop dataset, a subset of FMA (Defferrard et al., 2017) proposed by Gui et al. (2024). This FAD gives an idea about the musical quality of the edited output on its own. Please see App. F for more information on the choice of metrics. We plot the two metrics in Fig. 7, using different T_{start} configurations for SDEdit and different t and t' values for our approach. Our method achieves a higher quality (lower FAD to general music) for any desired edit strength. See App. H for additional results, App. L for validation of the semantics of our directions, and App. N for their applicability in other domains, *e.g.*, images.

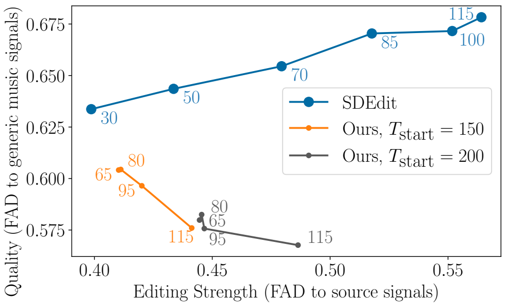


Figure 7. **Unsupervised editing strength vs. quality.** We compare SDEdit (Meng et al., 2021) to our method over the MusicDelta subset of MedleyDB (Bittner et al., 2014), using our prompts dataset. SDEdit is shown with T_{start} values ranging between 30 and 115, and our method is shown for different t' values (indicated on the plot) and T_{start} values (indicated in the legend). We use $T_{\text{end}} = 1$. For any level of perceptual deviation from the original signal, our method retains a higher quality (w.r.t. FMA-pop (Gui et al., 2024; Defferrard et al., 2017)).

4. Conclusion

We presented two methods for zero-shot editing of audio signals using pre-trained diffusion models. To the best of our knowledge, this is the first attempt to fully explore zero-shot editing in the audio domain. In addition to a text-based method, which we adopted from the image domain, we proposed a novel unsupervised method for discovering editing directions. We demonstrated both qualitatively and quantitatively that our methods outperform other methods for text-based editing, and illustrated that our unsupervised method is able to create semantically meaningful modifications and improvisations to a source signal.

Acknowledgements

This research was partially supported by the Israel Science Foundation (grant no. 2318/22), by a gift from Elbit Systems, and by the Ollendorff Minerva Center, ECE faculty, Technion.

Impact Statement

The purpose of this paper is to advance the field of Machine Learning and in particular zero-shot editing of audio signals. We feel that there are many potential societal consequences of our work, but the predominant one relates to the ability of using our method for copyright infringement. In this work we worked on audio licensed under Creative Commons Attribution, and as this is an academic work it is in fair use. However, users might use our methods to modify existing copyrighted musical pieces without sufficient permission of the copyright holder, and this might not fall under fair use under different circumstances. We believe it is important to develop methods for automatically detecting whether AI-based methods have been applied to audio signals.

References

Agostinelli, A., Denk, T. I., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., Huang, Q., Jansen, A., Roberts, A., Tagliasacchi, M., et al. MusicLM: Generating music from text. *arXiv preprint arXiv:2301.11325*, 2023. 10, 15

Bittner, R., Salamon, J., Tierney, M., Mauch, M., Cannam, C., and Bello, J. MedleyDB: A multitrack dataset for annotation-intensive mir research. In *15th International Society for Music Information Retrieval Conference, ISMIR 2014*, 2014. 4, 6, 15, 16, 17

Brooks, T., Holynski, A., and Efros, A. A. InstructPix2Pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402, June 2023. 1, 10

Cao, M., Wang, X., Qi, Z., Shan, Y., Qie, X., and Zheng, Y. MasaCtrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22560–22570, 2023. 10

Chen, H., Xie, W., Vedaldi, A., and Zisserman, A. VGGSound: A large-scale audio-visual dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 721–725. IEEE, 2020. 15

Chen, K., Du, X., Zhu, B., Ma, Z., Berg-Kirkpatrick, T., and Dubnov, S. HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 646–650. IEEE, 2022. 4, 16

Copet, J., Kreuk, F., Gat, I., Remez, T., Kant, D., Synnaeve, G., Adi, Y., and Défossez, A. Simple and controllable music generation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 3, 6, 10, 15, 16, 18

Defferrard, M., Benzi, K., Vandergheynst, P., and Bresson, X. FMA: A dataset for music analysis. In *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017. 6, 16

Dhariwal, P. and Nichol, A. Diffusion models beat GANs on image synthesis. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 8780–8794. Curran Associates, Inc., 2021. 1, 3, 6, 10, 13, 15, 16, 17

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. In *The Eleventh International Conference on Learning Representations*, 2022. 1, 10

Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Moore, R. C., Plakal, M., and Ritter, M. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 776–780, New Orleans, LA, 2017. IEEE. 4

Gui, A., Gamper, H., Braun, S., and Emmanouilidou, D. Adapting Frechet audio distance for generative music evaluation. In *Proc. IEEE ICASSP 2024*, 2024. 6, 15, 16

Haas, R., Huberman-Spiegelglas, I., Mulayoff, R., and Michaeli, T. Discovering interpretable directions in the semantic latent space of diffusion models. *arXiv preprint arXiv:2303.11073*, 2023. 10, 12

Han, B., Dai, J., Song, X., Hao, W., He, X., Guo, D., Chen, J., Wang, Y., and Qian, Y. InstructME: An instruction guided music edit and remix framework with latent diffusion models. *arXiv preprint arXiv:2308.14360*, 2023. 1, 4, 10

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., et al. CNN architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pp. 131–135. IEEE, 2017. 15

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-or, D. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2022. 10

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 4

Ho, J. and Salimans, T. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 2

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. 1

Huberman-Spiegelglas, I., Kulikov, V., and Michaeli, T. An edit friendly DDPM noise space: Inversion and manipulations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 2, 10, 11

Iashin, V. and Rahtu, E. Taming visually guided sound generation. In *British Machine Vision Conference (BMVC)*, 2021. 4

Jeong, J., Kwon, M., and Uh, Y. Training-free content injection using h-space in diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5151–5161, January 2024. 10

Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., and Irani, M. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6007–6017, 2023. 1, 10

Kilgour, K., Zuluaga, M., Roblek, D., and Sharifi, M. Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms. In *Proc. Interspeech 2019*, pp. 2350–2354, 2019. doi: 10.21437/Interspeech.2019-2219. 4

Kim, G., Kwon, T., and Ye, J. C. DiffusionCLIP: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2426–2435, 2022. 1

Kong, J., Kim, J., and Bae, J. HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17022–17033. Curran Associates, Inc., 2020. 2

Kwon, M., Jeong, J., and Uh, Y. Diffusion models already have a semantic latent space. In *The Eleventh International Conference on Learning Representations*, 2022. 10

Liu, H., Chen, Z., Yuan, Y., Mei, X., Liu, X., Mandic, D., Wang, W., and Plumbley, M. D. AudioLDM: Text-to-audio generation with latent diffusion models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 21450–21474. PMLR, 23–29 Jul 2023a. 1, 2, 10, 16

Liu, H., Tian, Q., Yuan, Y., Liu, X., Mei, X., Kong, Q., Wang, Y., Wang, W., Wang, Y., and Plumbley, M. D. AudioLDM 2: Learning holistic audio generation with self-supervised pretraining. *arXiv preprint arXiv:2308.05734*, 2023b. 1, 2, 3, 13

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021. 16

Manor, H. and Michaeli, T. On the posterior distribution in denoising: Application to uncertainty quantification. In *The Twelfth International Conference on Learning Representations*, 2024. 3, 10, 11, 13

Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2021. 1, 3, 6, 10, 15, 16, 17, 18, 21, 26, 27

Mokady, R., Hertz, A., Aberman, K., Pritch, Y., and Cohen-Or, D. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6038–6047, 2023. 13

Paissan, F., Wang, Z., Ravanelli, M., Smaragdis, P., and Subakan, C. Audio editing with non-rigid text prompts. *arXiv preprint arXiv:2310.12858*, 2023. 1, 4, 10

Park, Y.-H., Kwon, M., Jo, J., and Uh, Y. Unsupervised discovery of semantic latent directions in diffusion models. *arXiv preprint arXiv:2302.12469*, 2023. 10

Plitsis, M., Kouzelis, T., Paraskevopoulos, G., Katsouros, V., and Panagakis, Y. Investigating personalization methods in text to music generation. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1081–1085. IEEE, 2024. 1, 4, 10

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022. 2, 21, 26, 27

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. DreamBooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22500–22510, 2023. 1, 10

Saad, Y. *Numerical methods for large eigenvalue problems: revised edition*. SIAM, 2011. 3, 11

Shen, Y. and Zhou, B. Closed-form factorization of latent semantics in GANs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1532–1540, June 2021. 10

Shen, Y., Gu, J., Tang, X., and Zhou, B. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9243–9252, 2020. 10

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015. 1

Song, J., Meng, C., and Ermon, S. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 1, 3, 6, 10, 13, 15, 16, 17

Spingarn, N., Banner, R., and Michaeli, T. GAN "steerability" without optimization. In *International Conference on Learning Representations*, 2020. 10

Tumanyan, N., Geyer, M., Bagon, S., and Dekel, T. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1921–1930, 2023. 10, 21, 26, 27

Wang, Y., Ju, Z., Tan, X., He, L., Wu, Z., Bian, J., and sheng zhao. AUDIT: Audio editing by following instructions with latent diffusion models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 4, 10

Wu, C. H. and De la Torre, F. A latent space of stochastic diffusion models for zero-shot image editing and guidance. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7378–7387, 2023. 10, 11

Wu, Y., Chen, K., Zhang, T., Hui, Y., Berg-Kirkpatrick, T., and Dubnov, S. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023. 4, 15, 16

Wu, Z., Lischinski, D., and Shechtman, E. StyleSpace analysis: Disentangled controls for stylegan image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12863–12872, 2021. 10

Zhang, C., Ren, Y., Zhang, K., and Yan, S. SDMuse: Stochastic differential music editing and generation via hybrid representation. *IEEE Transactions on Multimedia*, pp. 1–9, 2023a. 10

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018. 4

Zhang, Y., Ikemiya, Y., Xia, G., Murata, N., Martínez, M., Liao, W.-H., Mitsufuji, Y., and Dixon, S. MusicMagus: Zero-shot text-to-music editing via diffusion models. *arXiv preprint arXiv:2402.06178*, 2024. 10

Zhang, Z., Han, L., Ghosh, A., Metaxas, D. N., and Ren, J. SINE: Single image editing with text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6027–6037, 2023b. 1

A. Related Work

Specialized audio editing models. The most common approach for editing audio is to train specialized models for this particular task. MusicGen (Copet et al., 2023) and MusicLM (Agostinelli et al., 2023) train language based models for generating audio conditioned on text, and optionally also on a given melody. Editing a music excerpt with MusicGen is achieved by conditioning the generation on the excerpt's chromagram while supplying a text prompt describing the desired edit. However, because of its reliance on chromagrams, it typically fails in editing polyphonic music. MusicLM conditioning is built on a novel proprietary joint music-text embedding space, named MuLan, built to encode monophonic melodies. A different approach, borrowed from the image domain (Brooks et al., 2023), is to train an instruction based diffusion model for editing. This has been done for general audio (Wang et al., 2023) as well as specifically for music (Han et al., 2023). These methods are limited to a small set of modifications (*e.g.*, "Add", "Remove", "Replace") and require training on a large dataset of triplets (text prompt, input audio, and output audio). Our methods require no training and are not limited to a fixed set of instructions.

Test-time optimization. Instead of training a model from scratch, some works leverage large pre-trained models for editing. Paissan et al. (2023) and Plitsis et al. (2024) demonstrated the effectiveness of test-time optimization methods, adopted from the image domain (Gal et al., 2022; Kawar et al., 2023; Ruiz et al., 2023), to editing of audio signals. These methods either fine-tune the diffusion model to reconstruct the given signal (Ruiz et al., 2023), optimize the text-embedding to reconstruct the signal (Gal et al., 2022), or use some combination of these approaches (Kawar et al., 2023). However, performing optimization for each new signal at test-time is computationally intensive. Moreover, these methods struggle with changing specific concepts, *e.g.*, replacing only the piano in a music piece with a banjo. Our techniques avoid test-time optimization, and can achieve focused edits.

Zero-shot editing. Some works focus on zero-shot editing using pre-trained diffusion models. Perhaps the simplest approach is SDEdit (Meng et al., 2021), which adds noise to the signal and then runs it through the reverse diffusion process with a different text prompt. SDEdit was recently used for audio (Liu et al., 2023a) as well as for piano-roll music (Zhang et al., 2023a). However it suffers from a severe tradeoff between adherence to the text and adherence to the original signal. Another direction, which has become popular in the image domain, is to use inversion techniques that extract the diffusion noise vectors corresponding to the source signal. One method for doing so is DDIM inversion (Song et al., 2021; Dhariwal & Nichol, 2021). This method was found suboptimal for editing images on its own, and is therefore typically accompanied by intervention in the attention maps during the diffusion process (Hertz et al., 2022; Cao et al., 2023; Tumanyan et al., 2023). A concurrent work by Zhang et al. (2024) recently proposed to edit audio by using DDIM inversion combined with latent space manipulations. However, inline with our observations about DDIM inversion, their method's applicability to real audio signals is limited. Another approach is DDPM inversion (Huberman-Spiegelglas et al., 2024; Wu & De la Torre, 2023), which is conceptually similar, but applies to the DDPM sampling scheme. Here we adopt the DDPM inversion method of Huberman-Spiegelglas et al. (2024), which has shown state-of-the-art results in the image domain.

Unsupervised editing. Finding semantic editing directions in an unsupervised manner, without any guidance or reference samples, has been exhaustively studied in the context of GANs (Spingarn et al., 2020; Shen et al., 2020; Shen & Zhou, 2021; Wu et al., 2021). Recently, several works proposed ways for finding editing directions in the bottleneck features (h-space) (Kwon et al., 2022) of a diffusion model (Haas et al., 2023; Park et al., 2023; Jeong et al., 2024) in an unsupervised manner. The unsupervised method we explore in this paper finds editing directions in the noise space of the diffusion model. This is done through adaptation of the method of Manor & Michaeli (2024), which quantifies uncertainty in Gaussian denoising.

B. Limitations

As in all zero-shot editing methods, the quality of the edited outputs are limited by the quality of the pre-trained audio model. This is most noticeable in text-based editing, as lack of proficiency in the textual information given in the target prompt can limit the edited output.

In unsupervised editing this concern is mitigated since no prompt is needed. However, as in all unsupervised methods, there rises the drawback of uncontrollability of the extracted PCs. A user cannot know in advance his PC of interest, and finding a PC for a user's liking is not guaranteed.

C. DDPM Inversion Details

As detailed in Sec. 2.1, to edit a real signal we follow the general approach of Huberman-Spiegelglas et al. (2024) and Wu & De la Torre (2023) for applying inversion. This process extracts noise vectors $\{\mathbf{x}_T, \mathbf{z}_T, ..., \mathbf{z}_1\}$ that cause the sampling process (1) to generate the given signal $\mathbf{x}_0$ at $t = 0$. We can then use those noise vectors to sample a signal using (1) while steering the generation towards a desired edit effect, as we detail in Sections 2.2 and 2.3.

To extract the noise vectors, we use the edit-friendly DDPM inversion method of Huberman-Spiegelglas et al. (2024). This method accepts as input the source signal $\mathbf{x}_0$ and generates from it an auxiliary sequence of vectors

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\tilde{\boldsymbol{\epsilon}}_t, \quad t = 1, \dots, T, \tag{4}$$

where $\tilde{\boldsymbol{\epsilon}}_t$ are sampled independently from $\mathcal{N}(0, \boldsymbol{I})$. It then extracts the noise vectors by isolating them from (1) as

$$\mathbf{z}_t = (\mathbf{x}_{t-1} - \boldsymbol{\mu}_t(\mathbf{x}_t))/\sigma_t, \quad t = T, \dots, 1. \tag{5}$$

While the noise vectors constructed this way have a different distribution than those participating in the original generative process (1), they have been shown to encode the global structure of the signal $\mathbf{x}_0$ more strongly, making them particularly suitable for editing tasks.

D. Unsupervised Editing Implementation Details

Consider the multivariate denoising problem observation $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where $\mathbf{x}$ is a random vector and the noise $\mathbf{n} \sim \mathcal{N}(0, \sigma^2)$ is statistically independent of $\mathbf{x}$. Then, Manor & Michaeli (2024) show that the posterior covariance relates to the Jacobian of an MSE-optimal denoiser,

$$\mathrm{Cov}[\mathbf{x}|\mathbf{y} = \boldsymbol{y}] = \sigma^2 \frac{\partial \mathbb{E}[\mathbf{x}|\mathbf{y} = \boldsymbol{y}]}{\partial \boldsymbol{y}}, \tag{6}$$

where $\boldsymbol{y}$ is the noisy sample. The work further showed that extracting the top eigenvectors and eigenvalues of the posterior covariance can be done using the subspace iteration method (Saad, 2011), where each iteration can be approximated using a single forward pass through the denoiser network.

In DDMs, Eq. (2) can be rearranged to fit the aforementioned observation model:

$$\frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}} = \mathbf{x}_0 + \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}}\boldsymbol{\epsilon}_t, \quad t = 1, \dots, T. \tag{7}$$

Therefore, $\mathbf{x} = \mathbf{x}_0$, $\mathbf{y} = \mathbf{x}_t/\sqrt{\bar{\alpha}_t}$, and $\mathbf{n} \sim \mathcal{N}(0, (1-\bar{\alpha}_t)/\bar{\alpha}_t\boldsymbol{I})$. Under this point of view, each timestep t in the diffusion process is a Gaussian denoising problem, and we get

$$\begin{aligned}\mathrm{Cov}\left[\mathbf{x}_0 \middle| \frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}}\right] &= \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} \cdot \frac{\partial \mathbb{E}\left[\mathbf{x}_0 \middle| \frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}}\right]}{\partial \frac{\mathbf{x}_t}{\sqrt{\bar{\alpha}_t}}} \\ \mathrm{Cov}\left[\mathbf{x}_0|\mathbf{x}_t\right] &= \frac{1-\bar{\alpha}_t}{\bar{\alpha}_t} \cdot \frac{\partial \mathbb{E}\left[\mathbf{x}_0|\mathbf{x}_t\right]}{\partial \mathbf{x}_t}.\end{aligned} \tag{8}$$

This allows for using the algorithm proposed by Manor & Michaeli (2024) at every desired timestep t' of the diffusion model, to extract semantic directions $\{\mathbf{v}_{i|t'}\}$ and their corresponding factors $\{\lambda_{i|t'}\}$.

These directions are PCs of the posterior covariance, and as such they need to be added to the primal space, and in particular to $\hat{\mathbf{x}}_{0|t'}$. As described in Sec. 2.1, the reverse process of a diffusion model is written in Eq. (1) as $\mathbf{x}_{t-1} = \boldsymbol{\mu}_t(\mathbf{x}_t) + \sigma_t\mathbf{z}_t$, where $\{\mathbf{z}_t\}_{t=1}^T \sim \mathcal{N}(0, \boldsymbol{I})$, $\mathbf{z}_0 = 0$, $\{\sigma_t\}$ is an increasing sequence of noise levels, and $\boldsymbol{\mu}_t(\mathbf{x}_t)$ is a function of a neural network trained to predict $\boldsymbol{\epsilon}_t$ from $\mathbf{x}_t$.

This function can be expressed as the sum of two elements:

$$\boldsymbol{\mu}_t = \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big) + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big). \tag{9}$$

Here, $\mathbf{f}_t(\mathbf{x}_t)$ is the aforementioned neural network trained to predict $\boldsymbol{\epsilon}_t$ from $\mathbf{x}_t$. $\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big)$ is given by

$$\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big) = \big(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)\big)/\sqrt{\bar{\alpha}_t}, \tag{10}$$

and is the predicted $\mathbf{x}_0$ at timestep t, noted as $\hat{\mathbf{x}}_{0|t}$. $\boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big)$ is the direction pointing to $\mathbf{x}_t$, given by

$$\boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big) = \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\mathbf{f}_t(\mathbf{x}_t). \tag{11}$$

Note that $\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big)$ gives a simple connection between $\hat{\mathbf{x}}_{0|t}$ and $\mathbf{f}_t(\mathbf{x}_t)$, the neural network, and therefore changing one is equivalent to changing another. Specifically, suppose we apply an edit to $\boldsymbol{\mu}_t$ by adding $\gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}$ to $\hat{\mathbf{x}}_{0|t}$ in Eq. (9). We denote this edited $\boldsymbol{\mu}_t$ as $\boldsymbol{\mu}_t^{\text{edit}}$:

$$\boldsymbol{\mu}_t^{\text{edit}}(\mathbf{x}_t) = \sqrt{\bar{\alpha}_{t-1}}\Big(\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big) + \gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big). \tag{12}$$

By substituting Eq. (10) into Eq. (12) we get

$$\begin{aligned}
\boldsymbol{\mu}_t^{\text{edit}}(\mathbf{x}_t) &= \sqrt{\bar{\alpha}_{t-1}}\Big(\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}} + \gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big) \\
&= \sqrt{\bar{\alpha}_{t-1}}\cdot\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)+\gamma\sqrt{\bar{\alpha}_t}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}}{\sqrt{\bar{\alpha}_t}} + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big) \\
&= \sqrt{\bar{\alpha}_{t-1}}\cdot\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\big(\mathbf{f}_t(\mathbf{x}_t)-\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\big)}{\sqrt{\bar{\alpha}_t}} + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big) \\
&= \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{P}\Big(\mathbf{f}_t(\mathbf{x}_t)-\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big).
\end{aligned} \tag{13}$$

This is an asymmetric reverse process formulation (Asyrp), since the two functions $\boldsymbol{P}$ and $\boldsymbol{D}$ that compose $\boldsymbol{\mu}_t^{\text{edit}}$ are operating on different variables. A symmetric reverse process is then given by also changing $\boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big)$ accordingly,

$$\boldsymbol{\mu}_t^{\text{edit}}(\mathbf{x}_t) = \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{P}\Big(\mathbf{f}_t(\mathbf{x}_t)-\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) + \boldsymbol{D}\Big(\mathbf{f}_t(\mathbf{x}_t)-\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big). \tag{14}$$

This means effectively subtracting $\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}$ from the noise prediction network output. Similar to Haas et al. (2023), we empirically find that the difference between the two formulations only changes the amplification of the editing effect, and therefore opt to use a symmetric reverse process for simplicity.

Finally, we can write $\boldsymbol{\mu}_t^{\text{edit}}$ explicitly by using both Eq. (10) and Eq. (11):

$$\begin{aligned}
\boldsymbol{\mu}_t^{\text{edit}}(\mathbf{x}_t) &= \sqrt{\bar{\alpha}_{t-1}}\Big(\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}} + \gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\Big(\mathbf{f}_t(\mathbf{x}_t)-\gamma\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big) \\
&= \sqrt{\bar{\alpha}_{t-1}}\Big(\frac{\mathbf{x}_t-\sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)}{\sqrt{\bar{\alpha}_t}}\Big) + \gamma\sqrt{\bar{\alpha}_{t-1}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'} \\
&\qquad + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\mathbf{f}_t(\mathbf{x}_t) - \gamma\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'} \\
&= \sqrt{\bar{\alpha}_{t-1}}\boldsymbol{P}\big(\mathbf{f}_t(\mathbf{x}_t)\big) + \boldsymbol{D}\big(\mathbf{f}_t(\mathbf{x}_t)\big) + \gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\Big(\sqrt{\bar{\alpha}_{t-1}} - \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\Big) \\
&= \boldsymbol{\mu}_t(\mathbf{x}_t) + \gamma\sqrt{\lambda_{i|t}}\mathbf{v}_{i|t'}\cdot\Big(\sqrt{\bar{\alpha}_{t-1}} - \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\Big).
\end{aligned} \tag{15}$$

Then, by setting

$$c_t = \sqrt{\bar{\alpha}_{t-1}} - \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}, \tag{16}$$

we get Eq. (3):

$$\mathbf{x}_{t-1} = \boldsymbol{\mu}_t(\mathbf{x}_t) + \gamma c_t \lambda_{i|t}^{1/2}\mathbf{v}_{i|t'} + \sigma_t\mathbf{z}_t, \quad t = T,\dots,1.$$

The addition of PCs can be done in two different ways. The first is to use $t' = t$ in (3), so that each denoising step is perturbed with its own PCs. The second way involves extracting a direction from some specific timestep t', and adding it at all timesteps $t \in [T_{\text{start}}, \ldots, T_{\text{end}}]$, where t' need not necessarily be a member of this set. In this case, it is important to use the factor $\lambda_{i|t}$ that matches the timestep t to which the perturbation is added, and not the factor $\lambda_{i|t'}$ corresponding to the timestep at which the PC was computed. The role of this factor is to match the applied change to the uncertainty level at the current timestep.

Adding the same direction at all timesteps usually strongly modifies a specific element, *e.g.*, a single note or the strength of a vibrato effect. Adding to each timestep its own PC can lead to a larger deviation from the original signal, *e.g.*, emphasizing a singer or changing a melody. This is because the extracted PCs can wildly differ in semantics and locality across a large range of timesteps. Here, as in Sec. 2.2, the editing directions can be confined to a user-chosen segment by applying a mask during the computation of the posterior PCs.

PCs computed using subspace iterations for each timestep separately are calculated independently of one another. As such, PCs for adjacent timesteps might be highly correlated. This is because in adjacent timesteps the noise level and uncertainty level are similar. Specifically, PCs from adjacent timesteps might be highly negatively correlated, as the positive directions are independently chosen at each timesteps. As explained in Sec. 2.3, one way of editing using Eq. (3) involves setting $t' = t$, so that each denoising step is perturbed with its own PCs. Therefore, it is possible that when perturbing the signal using PCs from neighboring timesteps they will cancel each other, thereby lessening the editing effect. To that end, at the end of the PCs computation for each timestep we compare the current PCs, $\mathbf{v}_{i|t}$, to those calculated during the previous timestep, $\mathbf{v}_{i|t+1}$. When the PCs correlation is lower than some threshold $\rho < 0$, we swap the direction of the current PCs, $\mathbf{v}_{i|t}$.

We empirically find that for each PC index i, the values of $\{\lambda_{i|t}\}_{t=1}^{T}$ are similar across signals and AudioLDM2 checkpoints. We therefore compute their average value over a dataset once, and use these average values for all signals.

In addition to publishing the code repository, we provide in Alg. 1 the complete algorithm for the unsupervised PC computation described here and in Sec. 2.3 for reference.

E. Partial DDIM Inversion

DDIM inversion (Song et al., 2021; Dhariwal & Nichol, 2021) is usually used only when T_{start} is set to T. The intuition behind this plain manner is that in DDIM sampling, the denoising process is deterministic given the noise map at $t = T$. The DDIM inversion process extracts this single noise map at $t = T$, and is usually not stopped at some mid-way timestep. The disadvantage of this approach is that it accumulates error which leads to poor reconstruction even when using the same prompt for the generation stage (Mokady et al., 2023). Contrary to that, DDPM sampling is a stochastic process, and therefore stopping DDPM inversion at any t is both theoretically motivated and can provide different edited results for the same text-prompt, using different inverted noise maps. The choice of t for DDPM inversion is completely up to the user and allows for determining how much the edited signal should diverge from the original signal.

Nevertheless, for the sake of completeness we choose to also compare here to a version of DDIM inversion that does stop mid-way, hence the name *Partial DDIM Inversion*. We use the same CFG parameters used for the plain DDIM Inversion approach, and plot the CLAP-LPAPS results for our method, the competing methods, and partial DDIM inversion for multiple T_{start} values, in the same manner as in Sec. 3.1. For each method the T_{start} values range is shown up to its turning point, that is due to the divergence between the training dataset distributions of CLAP and the text-encoder of AudioLDM2. The results appear in Fig. 8. Interestingly, this uncommon yet simple approach achieves relatively good results, however it is a bit lacking in text-adherence compared to other methods.

F. Experimental Details

In all experiments using AudioLDM2, we use 200 inference steps as recommended by the authors (Liu et al., 2023b). In all of our unsupervised editing experiments, we run 50 subspace iterations for extracting PCs, and set $C = 10^{-3}$ as the small approximation constant as described by Manor & Michaeli (2024). We use MusicGen with their default parameters provided in their official implementation demo. Additionally, we opt to not use negative prompts in all experiments.

F.1. MedleyMDPrompts

To enable a systematic analysis and quantitative comparison to other editing methods, we create and release with our code base a corresponding small dataset of prompts, named *MedleyMDPrompts*. This prompts dataset includes 3-4 source

Algorithm 1 Unsupervised PCs Computation

```
Inputs:
    Timesteps to extract PCs for $\{T_{\text{start}}, \ldots, T_{\text{end}}\}$,
    Inverted noise vectors $\{\mathbf{x}_{T_{\text{start}}}, \mathbf{z}_{T_{\text{start}}}, \ldots, \mathbf{z}_{T_{\text{end}}}\}$,
    Number of PCs $N$,
    DDPM Denoiser $\mathbf{f}_t(\cdot)$,
    coefficients $\{\bar{\alpha}_{T_{\text{start}}}, \ldots, \bar{\alpha}_{T_{\text{end}}}\}$,
    noise-levels $\{\sigma_{T_{\text{start}}}, \ldots, \sigma_{T_{\text{end}}}\}$,
    Threshold for correlation swap $\rho$,
    Approximation constant $C \ll 1$,
    Iterations amount $K$

Initialize $\mathbf{x}_t \leftarrow \mathbf{x}_{T_{\text{start}}}$
for $t \leftarrow T_{\text{start}}$ to $T_{\text{end}}$ do
    $\mathbf{x}_{0|t} \leftarrow \left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t)\right)/\sqrt{\bar{\alpha}_t}$    {Run a normal reverse pass.}
    $\boldsymbol{\mu}_t \leftarrow \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_{0|t} + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\mathbf{f}_t(\mathbf{x}_t)$
    $\mathbf{x}_{t-1} \leftarrow \boldsymbol{\mu}_t + \sigma_t \mathbf{z}_t$
    $\{\mathbf{v}_{i|t}^{(0)}\}_{i=1}^{N} \leftarrow \mathcal{N}(0, \boldsymbol{I})$    {Extract PCs using $K$ subspace iterations over $\mathbf{x}_{0|t}$.}
    for $k \leftarrow 1$ to $K$ do
        for $i \leftarrow 1$ to $N$ do
            $\mathbf{x}_t^{\text{shifted}} \leftarrow \mathbf{x}_t + C\sqrt{\bar{\alpha}_t}\mathbf{v}_{i|t}^{(k-1)}$
            $\mathbf{x}_{0|t}^{\text{shifted}} \leftarrow \left(\mathbf{x}_t^{\text{shifted}} - \sqrt{1-\bar{\alpha}_t}\mathbf{f}_t(\mathbf{x}_t^{\text{shifted}})\right)/\sqrt{\bar{\alpha}_t}$
            $\mathbf{v}_{i|t}^{(k)} \leftarrow \frac{1}{C}\left(\mathbf{x}_{0|t}^{\text{shifted}} - \mathbf{x}_{0|t}\right)$
        end for
        $\mathbf{Q}, \mathbf{R} \leftarrow$ QR_Decomposition($[\mathbf{v}_{1|t}^{(k)} \cdots \mathbf{v}_{N|t}^{(k)}]$)
        $[\mathbf{v}_{1|t}^{(k)} \cdots \mathbf{v}_{N|t}^{(k)}] \leftarrow \mathbf{Q}$
    end for
    $\mathbf{v}_{i|t} \leftarrow \mathbf{v}_{i|t}^{(K)}$    {Save the computed PCs and EVs for timestep $t$.}
    $\lambda_{i|t} \leftarrow \frac{1/\bar{\alpha}_t - 1}{C}\left\|\mathbf{x}_{0|t}^{\text{shifted}} - \mathbf{x}_{0|t}\right\|$
    for $i \leftarrow 1$ to $N$ do
        if $\mathbf{v}_{i|t} \cdot \mathbf{v}_{i|t+1} < \rho$ then
            $\mathbf{v}_{i|t} \leftarrow -\mathbf{v}_{i|t}$    {Swap the PCs direction if it is highly negatively correlated with the previous PC.}
        end if
    end for
    $\mathbf{x}_t \leftarrow \mathbf{x}_{t-1}$
end for
```

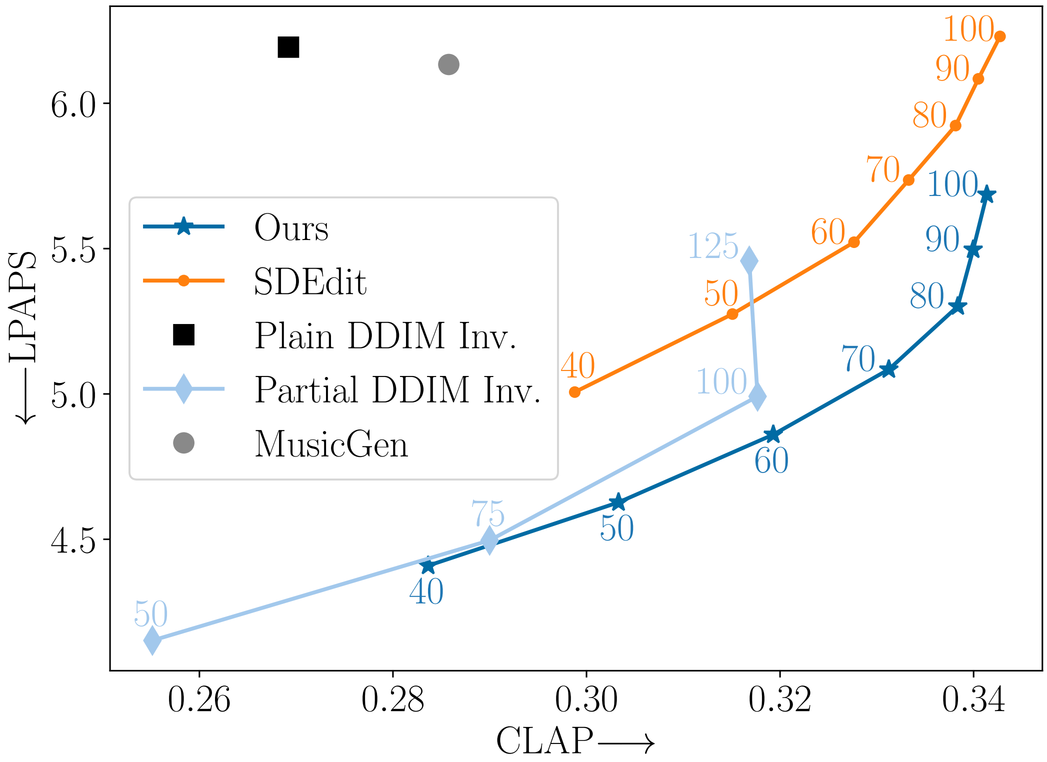


Figure 8. **Adherence to target prompt vs. fidelity to the original signal.** The plot compares MusicGen (Copet et al., 2023), SDEdit (Meng et al., 2021), DDIM Inversion (Song et al., 2021; Dhariwal & Nichol, 2021) and our method over the MusicDelta subset in MedleyDB (Bittner et al., 2014), using our prompts dataset. Our method and SDEdit are shown with T_{start} values ranging between 40 and 100. DDIM Inversion is shown both in its plain version, and its partial version when stopped mid-way using T_{start} values ranging between 50 and 125. Our results achieve lower (better) LPAPS for any level of CLAP (higher is better), indicating a good balance between text adherence and signal fidelity.

prompts for each signal, and 3-12 editing target prompts for each of the source prompts, totalling 107 source prompts and 696 target prompts, all labeled manually by the authors. We design the prompts to complement each other, *e.g.*, if the source prompt mentions a saxophone is playing, its target prompts may swap only the word “saxophone” with “guitar” or “piano”. We additionally design some of the target prompts such that they do not require a complementary source prompt, and should provide enough information to edit a signal on their own (*e.g.*, for genre change). In our supervised text-guided experiments we randomly sub-sample a third of the source-target prompts pairs for each signal (where we include additional pairs with an empty source prompt where applicable). Thus, we evaluate our supervised prompt-based method on 324 signal-text pairs. In our unsupervised uncertainty-based experiments we randomly sub-sample one of the source-prompts per audio signal.

We remark that some works use MusicCaps (Agostinelli et al., 2023) to quantitatively evaluate synthesized samples. However, this dataset contains only 10-second long music excerpts, while real music pieces can vary wildly over longer segments, changing instruments, genre or key completely. This aspect is important in the context of text-based editing, where the signal may be a minute long, and the edit should remain consistent across the entire piece (*e.g.*, when changing one instrument into another).

F.2. Metrics

FAD has been used in the past with deep features of VGGish (Hershey et al., 2017) or other convolutional neural networks (CNNs) trained on VGGSound (Chen et al., 2020). However, Gui et al. (2024) have shown that using such methods as a perceptual metric for music signals is sub-optimal, and so we follow their suggestion by using instead a trained large CLAP model (Wu et al., 2023) for the deep features of the FAD calculation. LPAPS has also been used in the past using CNNs trained on VGGSound, nevertheless we continue with the same reasoning and use intermediate features from the same CLAP model as LPAPS’ backbone in our evaluations. In particular, we use the output layers of the four intermediate

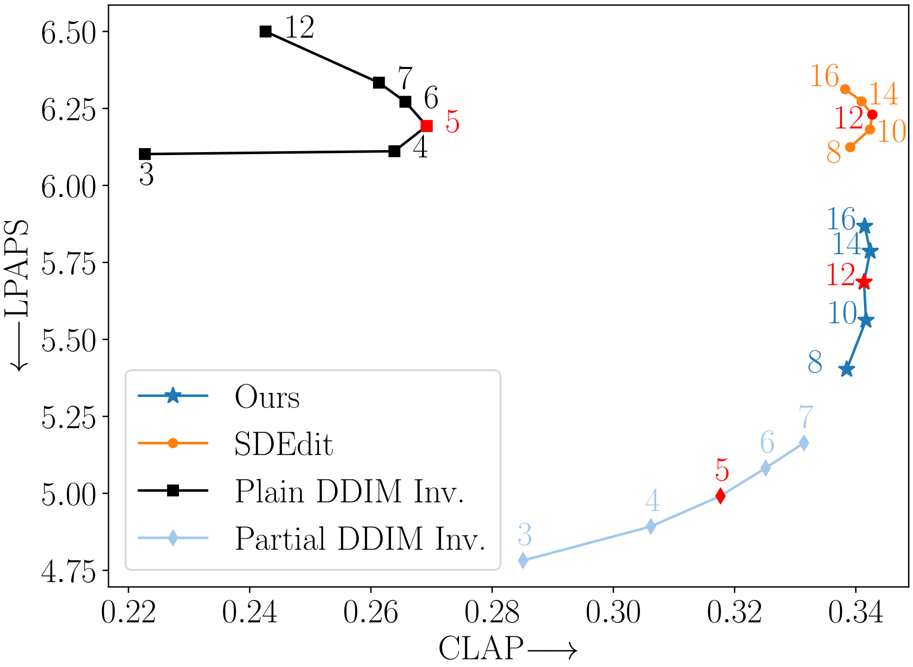


Figure 9. **Comparison of different target classifier-free guidance strengths used in the text-based editing process.** The plot compares SDEdit (Meng et al., 2021), DDIM Inversion (Song et al., 2021; Dhariwal & Nichol, 2021) and our method over the MusicDelta subset in MedleyDB (Bittner et al., 2014), using our prompts dataset. Our method and SDEdit are shown with target-prompt CFG strengths ranging between 8 and 16, when T_{start} is set to 100. DDIM Inversion is shown both in its plain version ($T_{\text{start}} = 200$), with target-prompt CFG strengths ranging between 3 and 12, and in its partial version when T_{start} is set to 100, with target-prompt CFG strengths ranging between 3 and 7. The floating numbers indicate the target-prompt CFG strength for each method. The chosen strength for SDEdit (12) and DDIM Inversion (5) are the best considering the trade-off between text-adherence and fidelity to the original signal. Our method can achieve even slightly better LPAPS scores while retaining the same CLAP score when lowering the strength below the value of 12, which we used in our experiments.

Swin-transformer blocks (Liu et al., 2021) of the CLAP model as LPAPS' features. For the CLAP model used in the CLAP, LPAPS, and FAD metrics calculation, we follow Gui et al. (2024) and MusicGen (Copet et al., 2023), and use the "music_audioset_epoch_15_esc_90.14.pt" checkpoint of LAION-AI (Chen et al., 2022; Wu et al., 2023). Since this checkpoint was trained for 10-second long segments, to calculate the score of a signal we first split the signal into overlapping 10-second long segments, calculate the score of each segment separately, and take the mean of the scores as the score of the entire signal. In all evaluations and for both metrics we use an overlap of one second.

Quantifying unsupervised edits in music can be challenging as the editable aspects of music vary widely, from rhythm, melody, instrumentation, mood and genre. Therefore, for this task, LPAPS is less fitting for measuring the "strength" of an edit. Specifically, a semantically small change like a slight shift in rhythm that occurs across the entire signal can throw off LPAPS completely, even though it is barely perceived by humans. Similarly, a short melodic change will rank low on LPAPS distance, but can shift the perceived mood of the piece. Therefore, instead of LPAPS, in our unsupervised editing experiments we measure the FAD to two different datasets. The first is the original Music Delta subset. This measures the strength of the edit, as it quantifies the deviation from the original distribution. The second is the FMA-pop dataset, a subset of FMA (Defferrard et al., 2017) proposed by Gui et al. (2024). This subset contains the 30 most popular songs for each of the 163 genres in the FMA dataset, and as such contains a large variety of genres and styles. This FAD gives an idea about the musical quality of the edited output on its own. This is because the songs in FMA-pop corresponds to the top listens and thus represent "likeability".

F.3. Classifier-Free Guidance Strength.

The classifier-free guidance strength for the source parameter was set to 3, as recommended by Liu et al. (2023a). In the unsupervised editing approach, this is the only classifier-free guidance strength hyper-parameter, as the prompt (if used) and

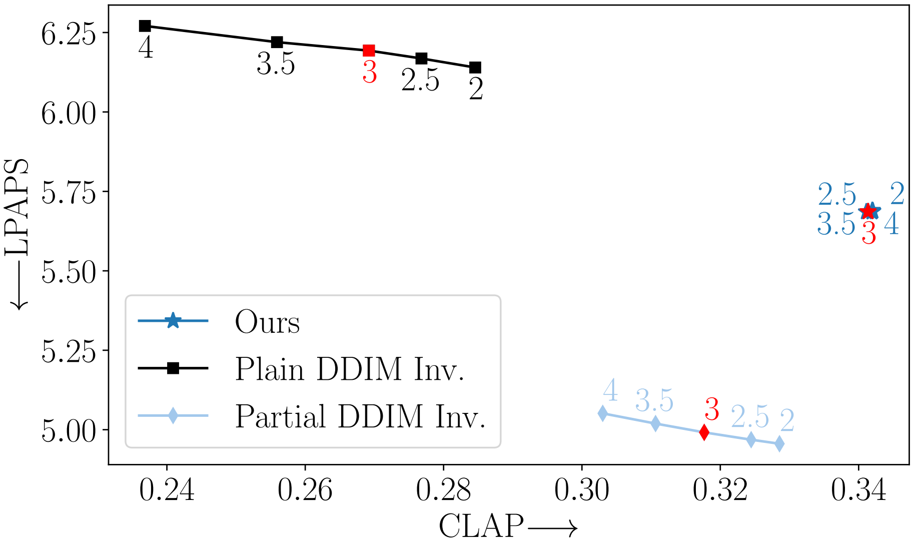


Figure 10. **Comparison of different source classifier-free guidance strengths used in the text-based editing process.** The plot compares DDIM Inversion (Song et al., 2021; Dhariwal & Nichol, 2021) and our method over the MusicDelta subset in MedleyDB (Bittner et al., 2014), using our prompts dataset. All methods are shown with source-prompt CFG strengths ranging between 2 and 4. DDIM version is shown both in its plain version ($T_{\text{start}} = 200$, and in its partial version. For both our method and partial DDIM inversion T_{start} is set to 100. The floating numbers indicate the source-prompt CFG strength for each method. The chosen strength for DDIM inversion (3) is a balanced choice, whereas for our method different strengths achieve largely similar scores.

its strength is not changed during the editing process. In the text-based editing approach, the strength used for the target prompt was chosen such that the resulting edits achieve a good balance between their CLAP and their LPAPS scores. To verify the chosen classifier-free guidance strengths for the different methods, we conduct two ablations, and display the results in Fig. 9 and Fig. 10. We include in the ablation the results for the partial DDIM inversion approach discussed in App. E, and set its T_{start} value to 100. We set the same T_{start} value for our method for and SDEdit (Meng et al., 2021).

For the target CFG ablation, we keep the source prompt guidance strength set to 3 for all methods. The results depicted in Fig. 9 show that the chosen strength for SDEdit (12) and DDIM Inversion (Song et al., 2021; Dhariwal & Nichol, 2021) (5) are the best considering the trade-off between text-adherence, measured with CLAP, and fidelity to the original signal, measured with LPAPS. Our method can achieve even slightly better LPAPS scores when retaining the same CLAP score when lowering the strength below the value of 12, which we used in our experiments. In the source CFG ablation, we keep the chosen target prompt guidance strengths, *i.e.*, 5 for DDIM inversion and 12 for our method. As can be seen in Fig. 10, the chosen hyper-parameter is a balanced choice for DDIM inversion, whereas for our method the different strengths achieve simliar CLAP and LPAPS scores.

G. Memory Requirements and Inference Speed

Let T be the total number of sampling steps (*e.g.*, in all our experiments we used $T = 200$), and S a Boolean indicating whether a source prompt is used. Whenever any prompt is used by applying classifier-free guidance, this involves two passes through the diffusion model's UNet, and therefore two neural function evaluations (NFEs).

We will start by addressing the text-based editing approach. Our method and DDIM Inversion require $(3 + S)T_{start}$ neural function evaluations (NFEs). $(1+S)T_{start}$ NFEs account for the inversion process (which runs from $t = 0$ up to $t = T_{start}$), and $2T_{start}$ NFEs account for the prompt-accompanied generation (which runs from $t = T_{start}$ to $t = 0$). For DDIM inversion, T_{start} is usually always taken to be T, as discussed in App. F. We also compare to a DDIM inversion process which stops mid-way, so this DDIM inversion version takes the same amount of NFEs as our method. SDEdit adds noise to the signal and then generates the edited output using $2T_{start}$ NFEs. Note, however, that SDEdit does not allow using a source prompt to help guide the editing process. MusicGen is a non-diffusion based model, trained specifically for editing, and is therefore not comparable using NFEs. From our experiments, for a 30 seconds long audio signal, a single NFE takes an average 123 milliseconds to complete. In a realistic editing scenario, without a source prompt (*i.e.*, setting $S = 0$), and

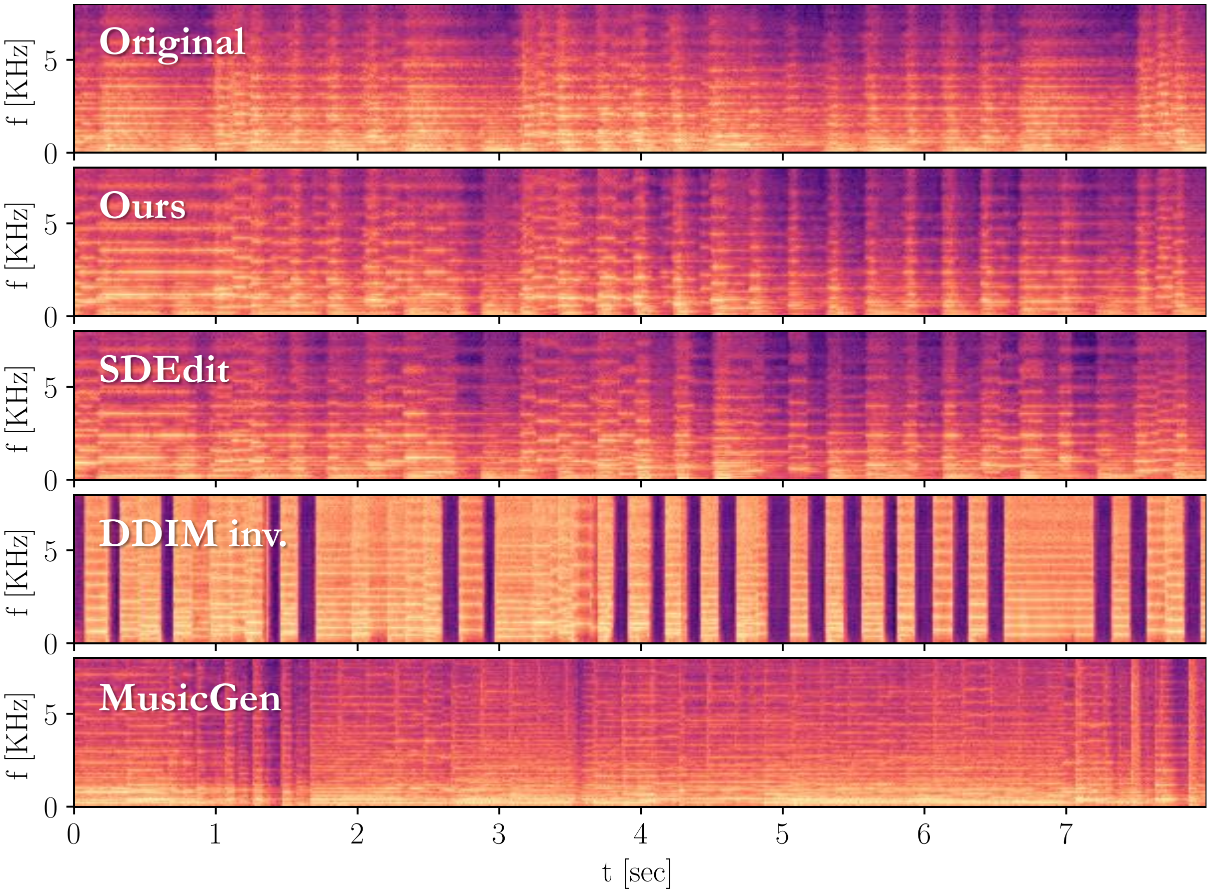


Figure 11. **Comparison of methods for text-based editing.** We compare our method, SDEdit (Meng et al., 2021), DDIM Inversion, and MusicGen (Copet et al., 2023) for editing of the same signal. All of the results achieve a CLAP score of ~ 0.34, however the LPAPS values are $4.29, 4.87, 6.26, 5.74$, respectively. This means that our method is most loyal to the original structure. Here for our method we use $T_{\text{start}} = 80$ and for SDEdit $T_{\text{start}} = 70$.

setting $D = 200, T_{start} = 100$, our method takes 37 seconds to complete, while MusicGen takes 35.5 seconds. All methods except MusicGen can easily set a lower D to shorten the sampling time substantially.

In the unsupervised editing approach, we will assume for this calculation no prompts. When a prompt is used to accompany the editing process, the results are just multiplied by a factor of two. Here we have an additional overhead in the calculation of the PC, which requires running K subspace iterations (per PC), meaning K NFEs. In our experiments we set $K = 50$, resulting in a 6-seconds long overhead for a 30-seconds long signal. We note that empirically we saw that a smaller K can suffice as well. In the unsupervised case, SDEdit still takes T_{start} NFEs. For our method, when t'
leT_{start}, we first invert the signal up to T_{start}, then compute the PCs, and finally run the generation, totaling in $2T_{start} + K$ NFEs. When $t' > T_{start}$, we extend the inversion process up to t', yielding $t' + T_{start} + K$ NFEs.

Finally, we consider the memory consumption of our methods. In all methods, both the used diffusion model and the noise tensors must be held in the memory of the GPU. We find that in the average case, the memory overhead of the noise tensors are negligible w.r.t. to the memory consumed by the diffusion model itself. For example, for a 30-seconds long signal, 200 noise tensors occupy 0.65GB, compared to 5.6GB occupied by the diffusion model.

H. More Results

We include a visual comparison of results using the compared text-based editing methods in Fig. 11. Fig. 12 shows the role different PCs can take in unsupervised editing.

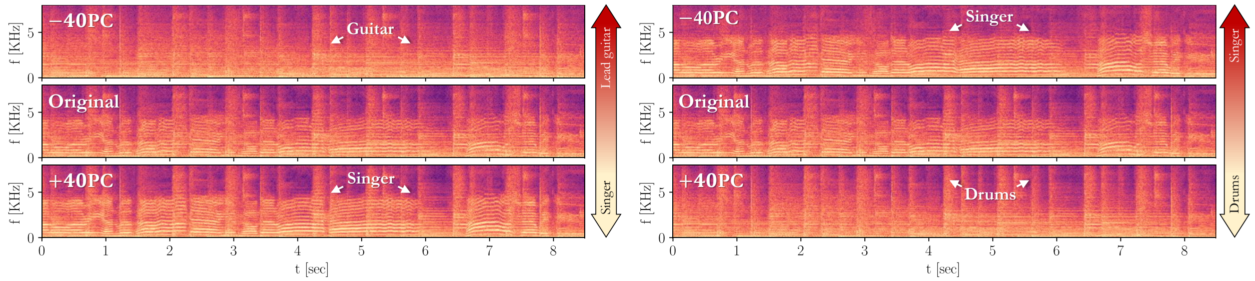


Figure 12. **The role of different PCs.** Our unsupervised editing method allows extracting different semantic editing directions for the same signal. These enable editing different semantic concepts that can complement each other. For example, the first PC shown here controls how much the singer is heard, at the expense of the lead guitar, while the second PC controls the whether the drums are dominant on the expense of the singer. Here $t' = t$, $T_{\text{start}} = 115$, and $T_{\text{end}} = 80$. This example can be listened to in Sec. 1.2.2 of our examples page.

I. The Effect of T_{start} on Text-Based Editing

The balance between adhering to the target text and remaining loyal to the original signal can be controlled using two parameters. The first is the strength factor of the classifier free guidance. Increasing this parameter steers the generation more strongly towards the desired text at the expense of departing from the original signal. The second parameter is the timestep T_{start} from which we begin the generation process. This timestep can generally be smaller than T, and the smaller it is, the more the edited signal remains consistent with the source signal.

Fig. 13 shows the effect of T_{start}. This parameter controls the trade-off between the adherence of the edited signal to the target prompt p_{edit}, and its fidelity to the original signal. This effect can also be listened to in Sec. 1.1.2 of our supplemental examples page.

J. Editing Over a User-Chosen Segment

Editing can be confined to a user-chosen segment, rather than the whole signal, by using a mask during the generative process. When doing so, at each timestep t, after computing $\mathbf{x}_{t-1}$ using (1) or (3), for text-based editing and unsupervised editing respectively, we enforce the parts of the signal outside the mask to shift back closer to the original signal at that timestep. We do this by setting

$$\mathbf{x}_{t-1} \leftarrow M \odot \mathbf{x}_{t-1} + (1 - M) \odot (\delta \mathbf{x}_{t-1}^{\text{orig}} + (1 - \delta)\mathbf{x}_{t-1}), \tag{17}$$

where M is the mask, and δ is some small constant which we fix to 0.025 in all experiments.

This shift is necessary due to "data leakage" caused by the architecture of the diffusion model. The term data leakage refers to the phenomenon where a localized edit, *e.g.*, a PC calculated for a specific segment of the signal using a mask, unintentionally affects the rest of the signal. This is commonly caused by the use of a UNet and attention modules as backbones for the diffusion model. The effect of the data leakage can be viewed in Fig. 14. The extracted PC at this timestep, $t = 80$, affects the snare of the drums beats. Without shifting the parts outside the mask back to the original signal, the snare changes across the entire signal. By setting $\delta > 0$ the change is localized to the masked region.

We notice empirically that the strength of the data leakage depends on the type of edit, however, generally it is not known a priori. Additionally, we would like to note that setting $\delta > 0$ is application specific and its effect is subjective. On the one hand, using a mask implies that changes outside the mask region are unwanted. However, allowing the edit to have a more global influence across the entire signal, could result in a more consistent result (*e.g.*, it might make more sense that the drum beats' snare changes across an entire musical piece).

K. The Effect of Using PCs From Different Timesteps $\{t'\}$

As mentioned in Sec. 3.2, modifications resulting from different choices of $\{t'\}$ vary in their extent of global impact. This can be measured quantitatively by calculating the entropy of the PC when summing over the different channels and height of the tensors. Fig. 15 displays the average entropy results for the first 3 PCs over the MusicDelta subset dataset used in the paper. Global changes that spread across large segments of the signal are characterized by a higher entropy, whereas lower

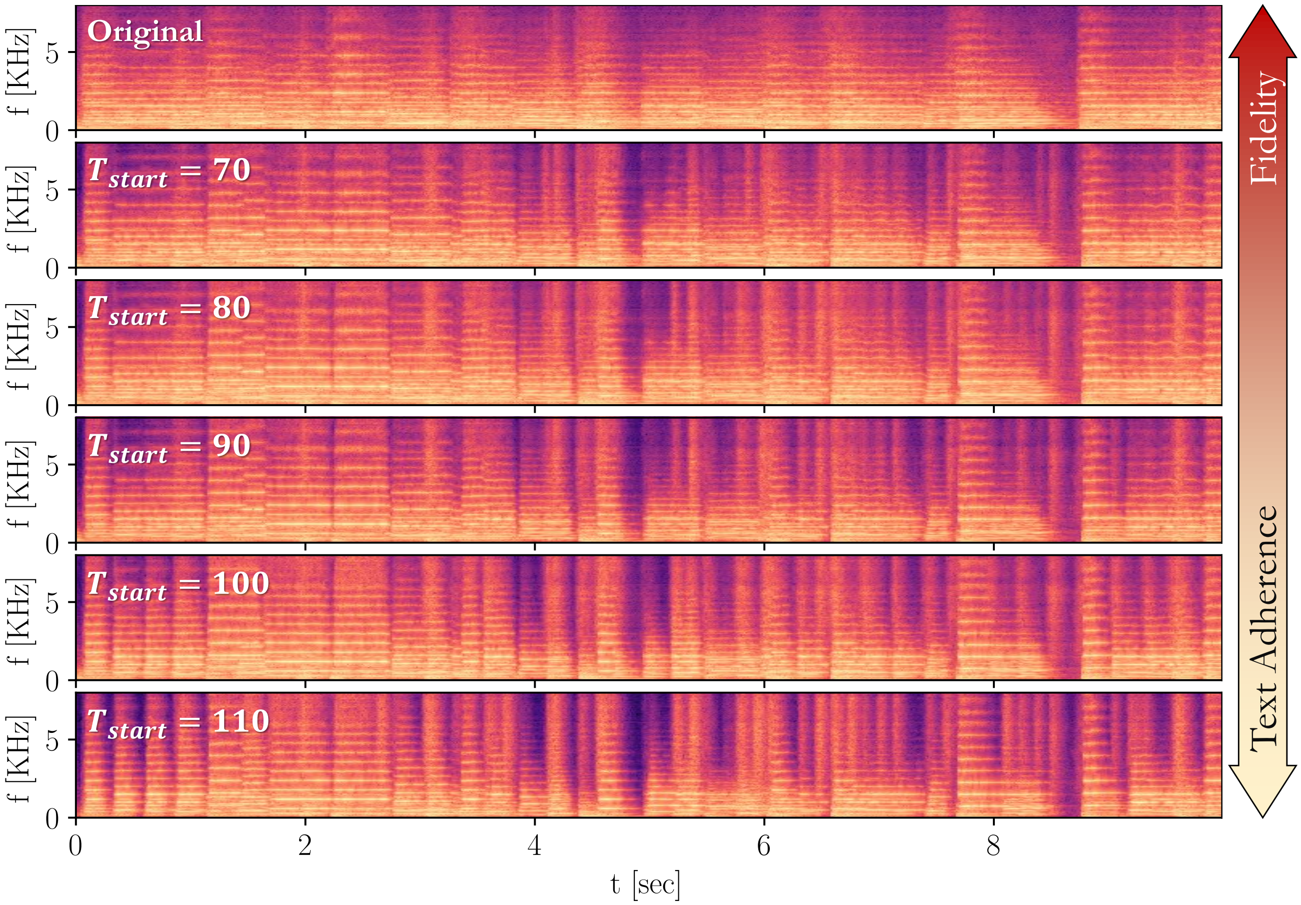


Figure 13. **The effect of T_{start} in text-based editing.** Here, an orchestral piece is edited by using only a target prompt, p_{edit}="A recording of a funky jazz song.". The signal retains more elements of the original signal as editing starts at a later timestep, denoted by T_{start}. This comes at a cost of adherence to the desired text description, *e.g*., less coherent beats as expected by a funky jazz song, and more elongated notes as in the original orchestral piece. This example can be listened to in Sec. 1.1.2 of our supplemental examples page.

entropy is an indicator for localized modifications. As can be seen, the entropy decays for smaller timesteps (later timesteps in the reverse diffusion process).

Empirically we see that the ability of the subspace iteration method to converge at large timesteps is hampered. This is also visible in Fig. 15, where the entropy is constant in the earliest timesteps. The uncertainty at the start of the diffusion process is very large, which coincides with the existence of multiple PCs with similar strength, *i.e*., there are no dominant directions. The subspace iteration method performs worse in such cases and is slower, and as such the extracted directions at earlier timesteps are not very interesting. We also note that timesteps so early in the diffusion process are responsible for very global semantics, therefore editing them will result in a very large deviation from the original signal. Effectively, editing using those timesteps is equivalent to synthesizing a signal almost from scratch instead of editing it, which is not the desired task. As the reverse process continues the uncertainty decreases, the signal's general structure is set, and the PCs display more fine-grained directions of editing.

We additionnally notice empirically that above a certain timestep the extracted directions are not interesting. We therefore restrict ourselves here to $t \leq 135$.

L. Comparison of Unsupervised Editing Directions With Random Directions

To ensure our computed PCs are significant and do indeed carry semantic meaning, we compare them to random directions in Fig. 16, Fig. 17 and Fig. 18. Specifically, we compare for multiple signals the 1^{st} PC computed using our method, or

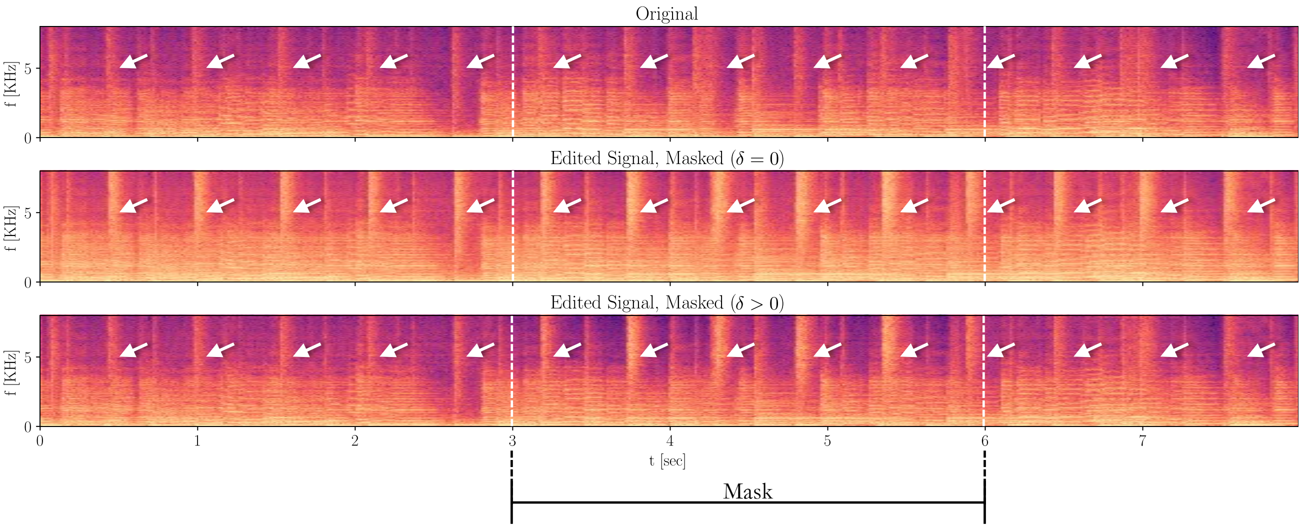


Figure 14. **Example of data leakage in diffusion models editing.** The extracted PC affects the snare of the drums beats, marked with arrows. By shifting the parts of the signal outside the mask closer to the original signal at each timestep, the snare only changes in the masked region. By not shifting the signal, the snare changes along the entire signal.

a combination of the first 3 PCs, using both editing ways: (i) Applying a specific timestep t' and adding it to a range of timesteps $[T_{\text{start}}, \ldots, T_{\text{end}}]$, and (ii) setting $t' = t$ when adding the PCs to a range of timesteps. When using a specific timestep, t', across a range of timesteps, $[T_{\text{start}}, \ldots, T_{\text{end}}]$, we sample a random direction from an isotropic Gaussian distribution. When settings $t' = t$, we randomly sample a direction for each timestep in the range $[T_{\text{start}}, \ldots, T_{\text{end}}]$, sampled i.i.d from an isotropic Gaussian distribution. In both cases, we normalize the randomly sampled directions to share the same norm as our directions, the unit-norm, and use the same computed eigenvalues $\lambda_{i|t}^{1/2}$ in Eq. (3).

Our computed PCs display semantically meaningful editing directions, while using random directions over the strength γ introduces almost imperceptible changes. Using a large γ factor introduces random changes that rapidly degrade the quality of the modification. All examples can be listened to in Sec. 3 of our supplemental examples page.

M. User Study

We include screenshots from our user study interface in Fig. 19. For fairness, all samples are normalized at the same loudness equal to -19.11 dBFS.

N. Unsupervised Editing in Images

We demonstrated our novel unsupervised editing approach on audio signals, since when applied on music it exposes a range of musically interesting modifications, some of which are virtually impossible to describe by text precisely. However, this approach is not limited to the audio domain. As a preliminary demonstration of its generalization ability, we leverage Stable Diffusion (Rombach et al., 2022) to demonstrate in Figs. 20, 21 and Fig. 22 semantic editing directions extracted for images from “modified ImageNet-R-TI2I” (Tumanyan et al., 2023), and contrast the results with both the well known SDEdit (Meng et al., 2021) and our previously proposed random baseline (See App. L). In all of the examples, no source prompts were used. Our editing directions encode meaningful direction that change semantic elements while keeping the rest of the images, both its essence (*e.g.*, a sketch of a cat) and its structure intact.

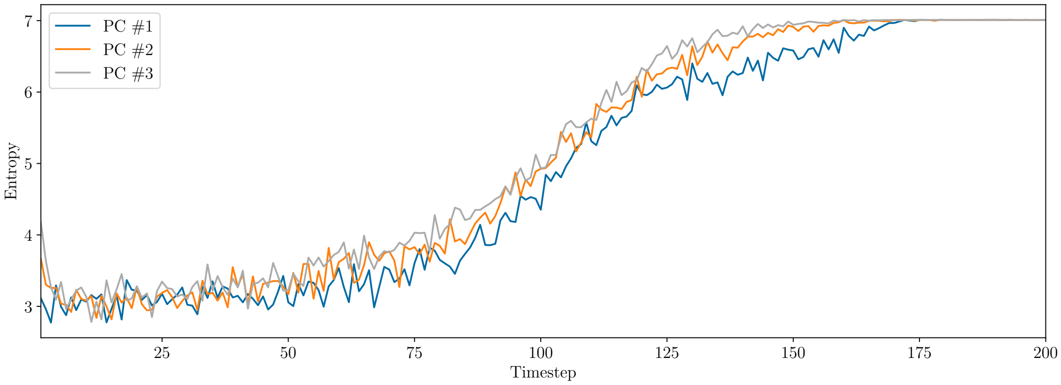
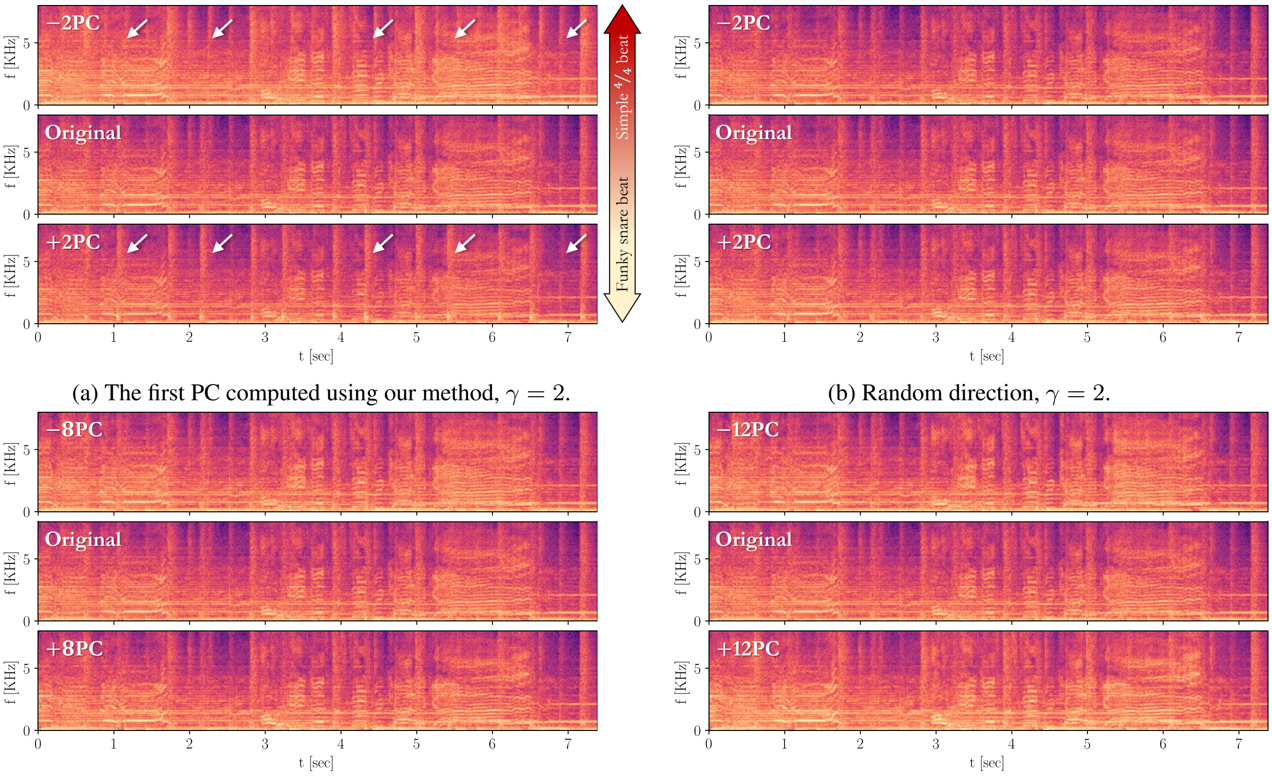


Figure 15. **Average entropy of the extracted PCs, across different timesteps.** Higher entropy is an indicator for global PCs, that change large segments of the signal. As the reverse process continues, the uncertainty in $\hat{\mathbf{x}}_{0|t}$ decreases, and the extracted PCs affect more localized areas, and measure lower in entropy.



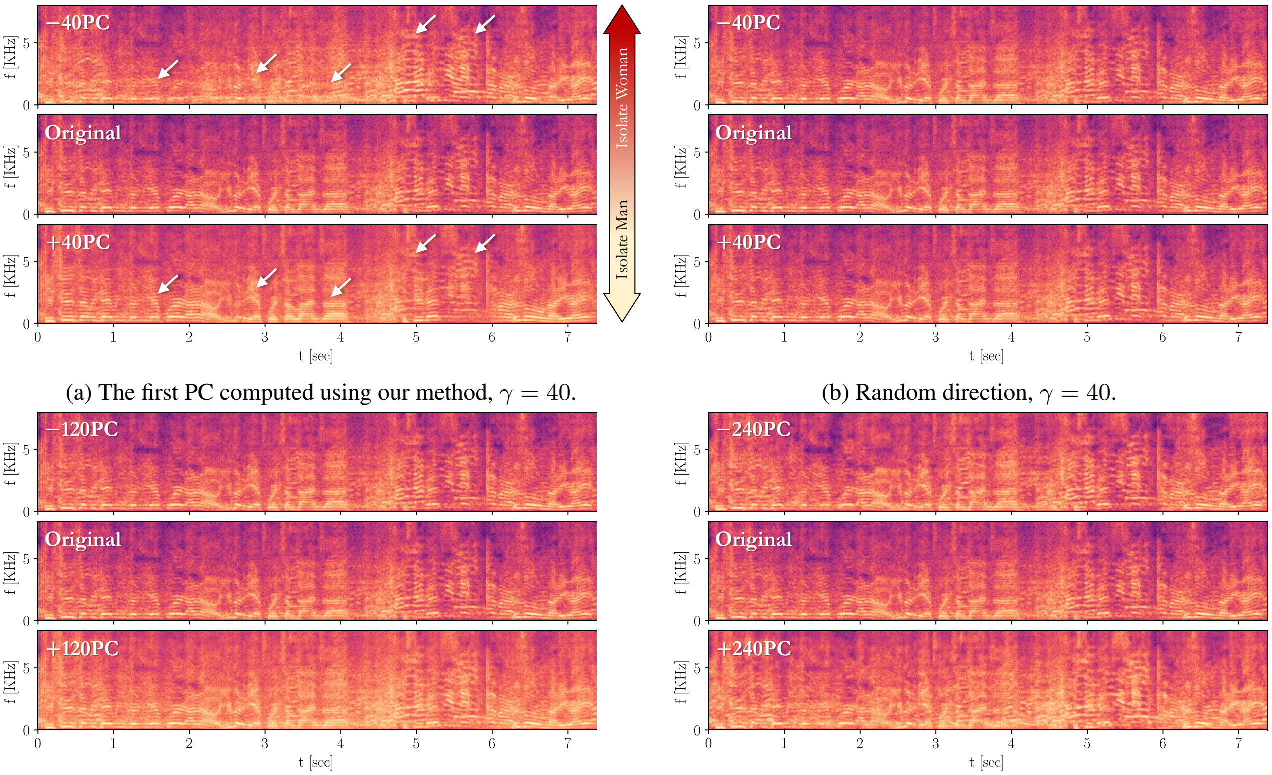
(a) The first PC computed using our method, $\gamma = 2$.

(b) Random direction, $\gamma = 2$.

(c) Random direction, $\gamma = 8$.

(d) Random direction, $\gamma = 12$.

Figure 16. **Our 1^{st} PC vs. a random direction.** While our method extract PCs with a clear semantic meaning (16a), *e.g.*, changing the drums beat style from a simple 4/4 beat to a syncopated (off-beat) and snare-heavy beat, the effect of using random directions (16b),(16c),(16d) varies from producing unnoticeable changes with small γ factor to degrading the signal when large γ values are used. This result can be listened to in Sec. 3 of our examples page. We fix here $t' = 80$ for our method, and apply our or the random directions for $T_{\text{start}} = 200$, $T_{\text{end}} = 1$. Both samples were generated using the music checkpoint of AudioLDM2, with the following source prompt randomly chosen from our prompts dataset: "A high quality recording of a man singing with a rock band accompaniment."



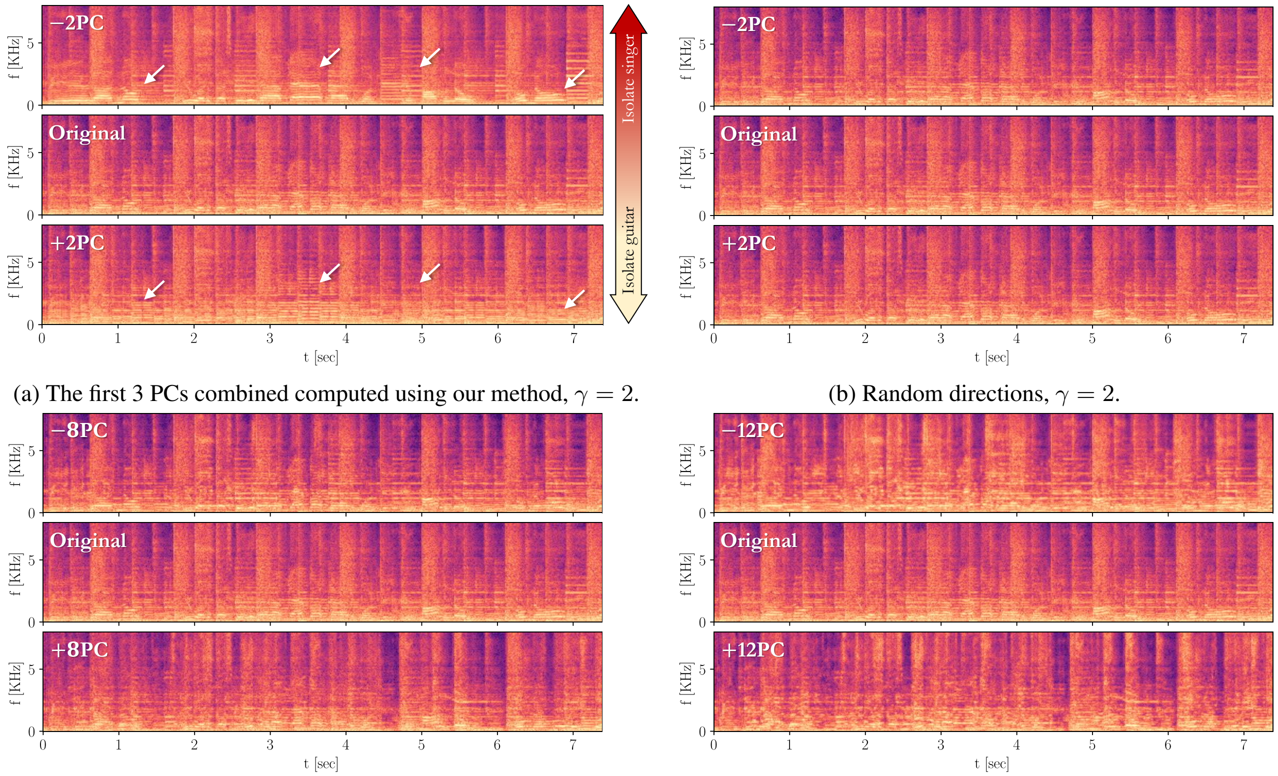
(a) The first PC computed using our method, $\gamma = 40$.

(b) Random direction, $\gamma = 40$.

(c) Random direction, $\gamma = 120$.

(d) Random direction, $\gamma = 240$.

Figure 17. **Our 1^{st} PC vs. a random direction.** While our method extract PCs with a clear semantic meaning (17a), *e.g.*, isolating a man or a woman speaking in a given signal, the effect of using random directions (17b),(17c),(17d) varies from producing unnoticeable changes with small γ factor to degrading the signal when large γ values are used. This result can be listened to in Sec. 3 of our examples page. We fix $t' = t$, and apply our or the random directions for $T_{\text{start}} = 115$, $T_{\text{end}} = 95$. Both samples were generated using the large checkpoint of AudioLDM2, without a source prompt.



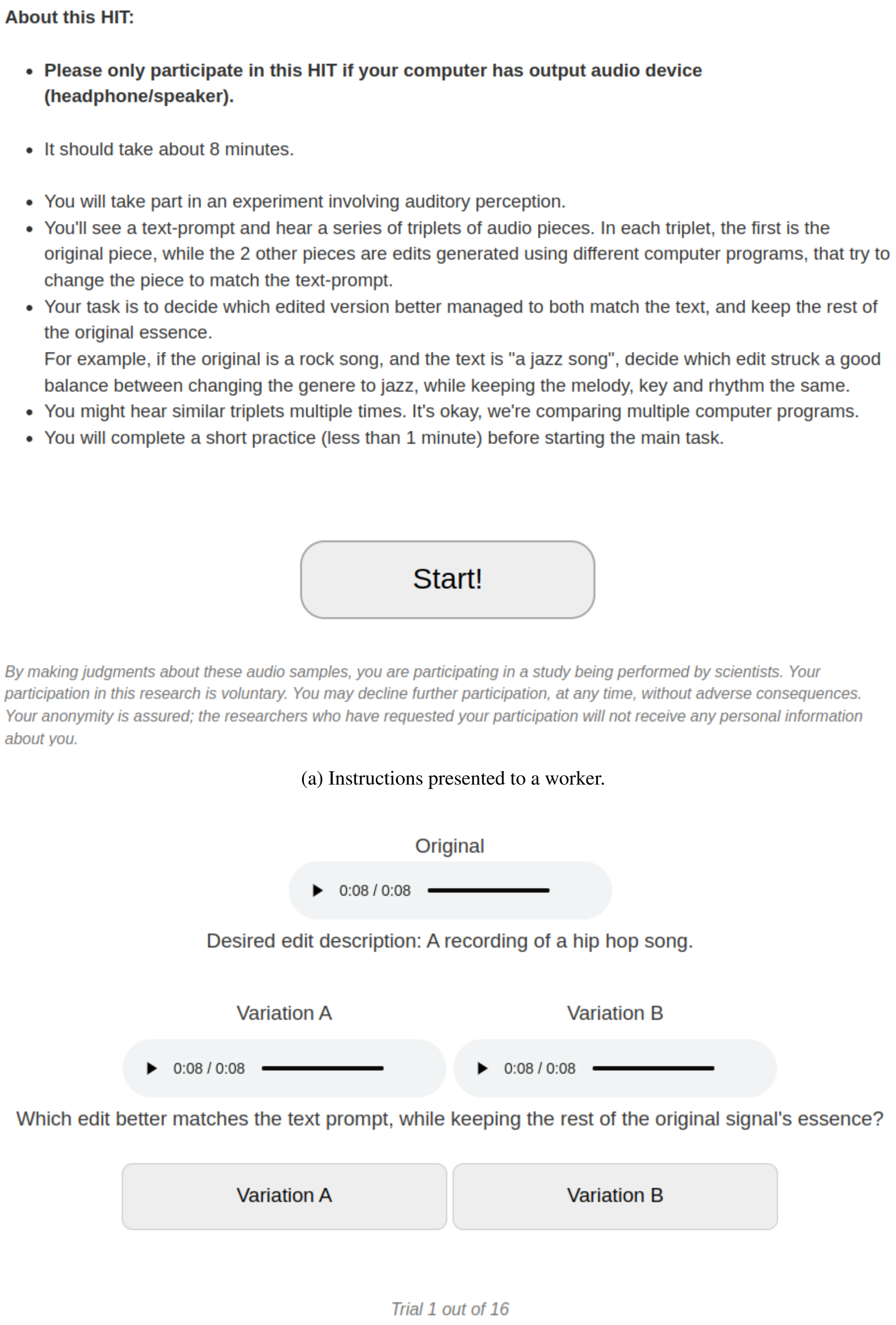
(a) The first 3 PCs combined computed using our method, $\gamma = 2$.

(b) Random directions, $\gamma = 2$.

(c) Random directions, $\gamma = 8$.

(d) Random directions, $\gamma = 12$.

Figure 18. **Our PCs vs. random directions.** While our method extract PCs with a clear semantic meaning (18a), *e.g.*, isolating a singer or a guitar in a given signal, the effect of using random directions (18b),(18c),(18d) varies from producing unnoticeable changes with small γ factor to degrading the signal when large γ values are used. This result can be listened to in Sec. 3 of our examples page. We fix $t' = 65$ for our method, and apply our or the random directions for $T_{\text{start}} = 200$, $T_{\text{end}} = 1$. Both samples were generated using the music checkpoint of AudioLDM2, with the following source prompt randomly chosen from our prompts dataset: "A recording of an old timey rock song from the sixties."



About this HIT:

- **Please only participate in this HIT if your computer has output audio device (headphone/speaker).**

- It should take about 8 minutes.

- You will take part in an experiment involving auditory perception.
- You'll see a text-prompt and hear a series of triplets of audio pieces. In each triplet, the first is the original piece, while the 2 other pieces are edits generated using different computer programs, that try to change the piece to match the text-prompt.
- Your task is to decide which edited version better managed to both match the text, and keep the rest of the original essence.
  For example, if the original is a rock song, and the text is "a jazz song", decide which edit struck a good balance between changing the genere to jazz, while keeping the melody, key and rhythm the same.
- You might hear similar triplets multiple times. It's okay, we're comparing multiple computer programs.
- You will complete a short practice (less than 1 minute) before starting the main task.

By making judgments about these audio samples, you are participating in a study being performed by scientists. Your participation in this research is voluntary. You may decline further participation, at any time, without adverse consequences. Your anonymity is assured; the researchers who have requested your participation will not receive any personal information about you.

(a) Instructions presented to a worker.

Desired edit description: A recording of a hip hop song.

Which edit better matches the text prompt, while keeping the rest of the original signal's essence?

Trial 1 out of 16

(b) Example of a question screen.

Figure 19. **Screenshots from our user study.** After reading the instructions (upper), workers were shown 16 question screens (lower) to choose their preferred edit.

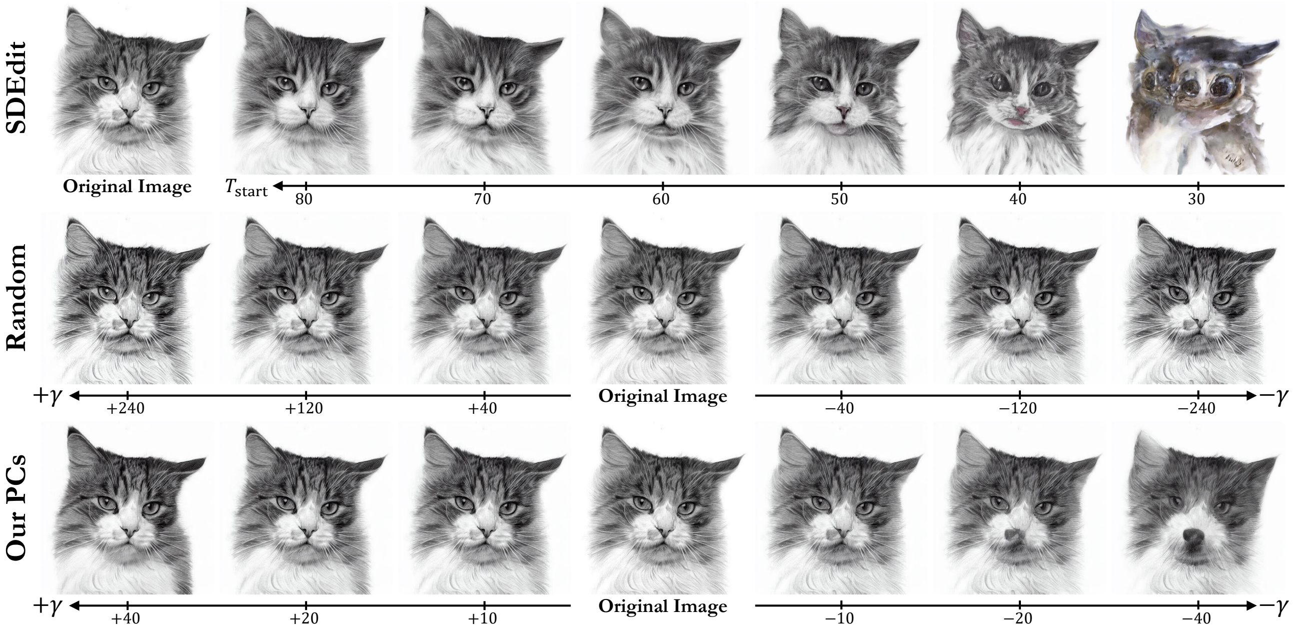


Figure 20. **Unsupervised zero-shot editing of images.** We demonstrate the applicability of our unsupervised approach for extract editing directions using Stable Diffusion (Rombach et al., 2022) on images taken from "modified ImageNet-R-TI2I" (Tumanyan et al., 2023), contrasted with SDEdit (Meng et al., 2021) and our previously proposed random baselines (See App. L. Our method extracts editing directions that carry a semantic meaning, *e.g.*, a direction for changing the species of the cat or making it a more distinct cat-breed, while retaining the original essence of the image.

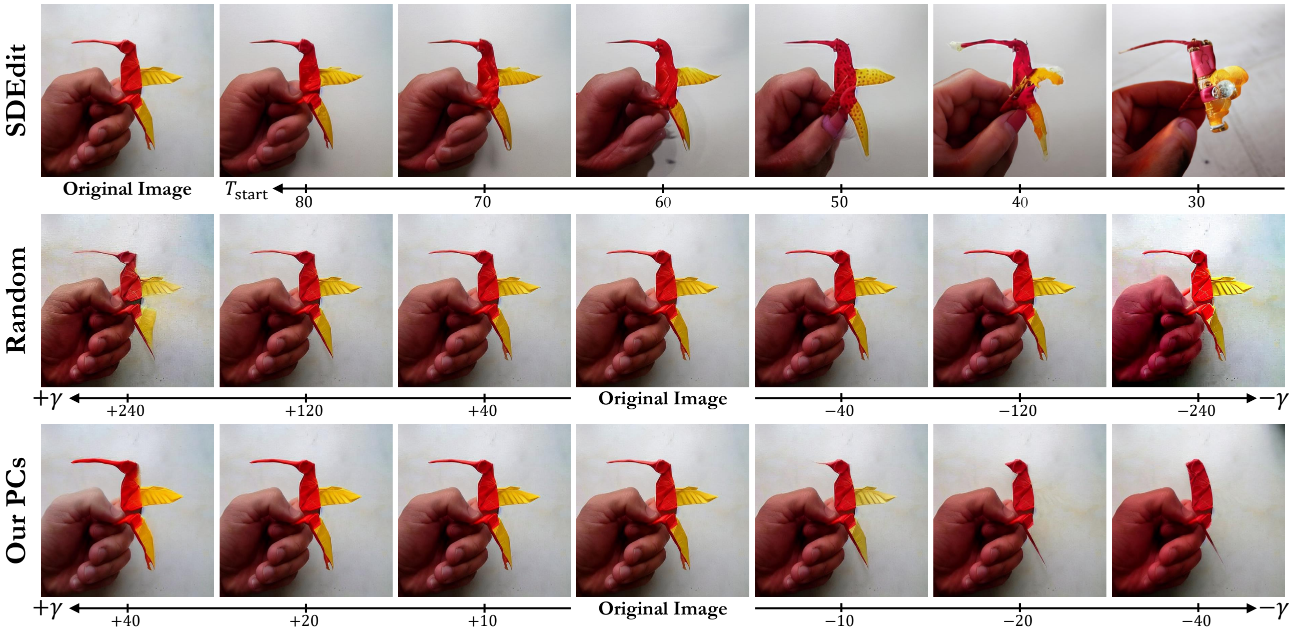


Figure 21. **Unsupervised zero-shot editing of images.** We demonstrate the applicability of our unsupervised approach for extract editing directions using Stable Diffusion (Rombach et al., 2022) on images taken from "modified ImageNet-R-TI2I" (Tumanyan et al., 2023), contrasted with SDEdit (Meng et al., 2021) and our previously proposed random baselines (See App. L. Our method extracts editing directions that carry a semantic meaning, *e.g.*, thickening the beak of a bird or shortening it until it is no longer a bird and the wings dissapear, while retaining the essence and structure of the image, *e.g.*, not degrading the hand.

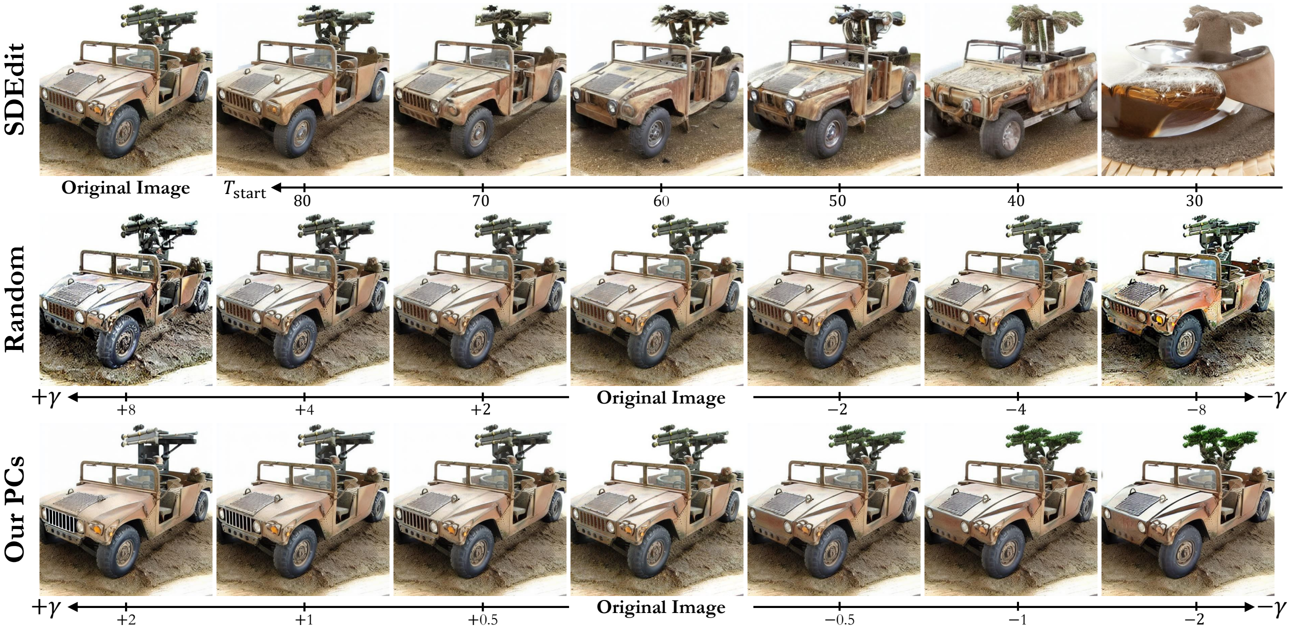


Figure 22. **Unsupervised zero-shot editing of images.** We demonstrate the applicability of our unsupervised approach for extract editing directions using Stable Diffusion (Rombach et al., 2022) on images taken from “modified ImageNet-R-TI2I” (Tumanyan et al., 2023), contrasted with SDEdit (Meng et al., 2021) and our previously proposed random baselines (See App. L. Our method extracts editing directions that carry a semantic meaning, *e.g.*, turning a gun to a bonsai tree, while retaining the essence and structure of the image, *e.g.*, keeping the jeep toy car relatively the same.