Tahiroğlu, Koray; Kastemaa, Miranda; Koli, Oskar

GANSpaceSynth: A Hybrid Generative Adversarial Network Architecture for Organising the Latent Space using a Dimensionality Reduction for Real-Time Audio Synthesis

DOI:
10.5281/zenodo.5137902

Published: 19/07/2021

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Tahiroğlu, K., Kastemaa, M., & Koli, O. (2021). *GANSpaceSynth: A Hybrid Generative Adversarial Network Architecture for Organising the Latent Space using a Dimensionality Reduction for Real-Time Audio Synthesis*. 10. Paper presented at Conference on AI Music Creativity, Graz, Austria.
https://doi.org/10.5281/zenodo.5137902

GANSpaceSynth: A Hybrid Generative Adversarial Network Architecture for Organising the Latent Space using a Dimensionality Reduction for Real-Time Audio Synthesis

Koray Tahiroğlu[1], Miranda Kastemaa[1] and Oskar Koli[1]

Department of Media, Aalto University School of ARTS, FI 00076 AALTO Finland
firstname.surname@aalto.fi

Abstract. Generative models enable possibilities in audio domain to present timbre as vectors in a high-dimensional latent space with Generative Adversarial Networks (GANs). It is a common method in GAN models in which the musician's control over timbre is mostly limited to sampling random points from the space and interpolating between them. In this paper, we present a novel hybrid GAN architecture that allows musicians to explore the GAN latent space in a more controlled manner, identifying the audio features in the trained checkpoints and giving an opportunity to specify particular audio features to be present or absent in the generated audio samples. We extend the paper with the detailed description of our GANSpaceSynth and present the Hallu composition tool as an application of this hybrid method in computer music practices.

Keywords: AI-enhanced Music Creativity, Generative systems, AI and music, hybrid GAN, Composition systems, Audio hallucination.

1 Introduction

The attainment of building machines being able to synthesis and output on activities that people do is a big part of the AI space, and one of the big reasons that AI is getting all this interest. But for a variety of reason it might just not the most interesting problem to work on in a field with a growing demand for challenging human creativity. The most interesting thing to think about for the autonomous algorithms, that is now evolving in music-making, is going beyond the possibilities in exploring creativity in a mutual human-machine cooperation in which we are aware of what is happening musically and be able to control the development of musical creativity. Recent developments in autonomous and generative AI methods present ways and means of realisation of the of human-machine co-creativity in music (Carnovalini & RodÃ, 2020; Briot & Pachet, 2020). In our current research, we focus on generative systems that comprise the behaviour of generative AI models and ascribe it to the intrinsic features of digital musical instruments (Tahiroğlu, Kastemaa, & Koli, 2020, 2021).

In this paper, we address some issues associated with the generative models and present our hybrid generative adversarial network (GAN) architecture, GANSpaceSynth, in detail. GANSpaceSynth generates new audio samples using features learned from the original dataset and has the ability to specify particular audio features to be present or absent in the generated audio samples. In the following sections, we also present our Hallu composition tool as one of the application of GANSpaceSynth in co-creative computer music practices.

2 Deep Learning Models for Audio Synthesis

Recent advances in image generation have been driven by the development of generative adversarial networks (GANs)(Goodfellow et al., 2014) and the success of image GANs sparked interest in adapting the technique to audio generation. A simple way to do this is to represent audio as a spectrogram image and apply an existing image GAN model. Engel et al. refined the spectrogram approach in GANSynth, with improvements to the spectral representation including the use of instantaneous frequency spectra and mel frequency scale (Engel et al., 2019).

While powerful, such approaches do not make strong use of existing knowledge of signal processing and perception, hampering their efficiency. Engel et al. discussed these problems in DDSP: Differentiable Digital Signal Processing(Engel, Hantrakul, Gu, & Roberts, 2020), pointing out phase alignment and spectral leakage as challenges in the case of GANSynth. They examined an alternative type of model, known as vocoders or synthesisers, where the neural network is used to learn synthesis parameters for oscillators that generate the actual audio. This method was previously limited by the inability to integrate the synthesis elements into the neural network in order to allow end-to-end training on audio examples. The authors overcame this problem by introducing a set of differentiable signal processing components and demonstrated efficient and high-fidelity implementations of timbre transfer as well as extraction and transfer of room acoustics. Magenta and AIUX at Google recently published a web interface for the timbre transfer as Tone Transfer (Hantrakul et al., 2020).

GANSynth learns to represent timbre as vectors in a high-dimensional latent space. Navigating this space along humanly meaningful directions is difficult, and the musician's control over timbre is mostly limited to sampling random points from the space and interpolating between them. Addressing this issue, alternative ways to interact with the latent space with generative melody models have been demonstrated (Vogl & Knees, 2016; Roberts, Engel, Raffel, Hawthorne, & Eck, 2018; Zhou, Koyama, Goto, & Igarashi, 2020). Härkönen et al. faced analogous problems with image GANs and came up with a simple but powerful solution in GANSpace with Principal Component Analysis (PCA) (Härkönen, Hertzmann, Lehtinen, & Paris, 2020). PCA can be applied to the neuron activations in an existing GAN model in order to identify significant directions of change, which can be turned into editable parameters. They found many of the discovered edits to be semantically meaningful, improving the interpretability of the models used.

For example, in face-generating models, edits were found for controlling image features like head rotation and hair color.

There are other dimensionality reduction techniques used for similar purposes, such as UMAP (McInnes, Healy, & Melville, 2020); however, we have been motivated by the simplicity of the PCA implementation in our approach to apply GANSpace to GANSynth. Similarly, related approaches to organizing the latent space of generative audio models have been proposed using Variational Autoencoders (VAEs) (Kingma & Welling, 2019). Esling et al. used perceptual ratings from timbre studies to regularize VAE latent spaces in order to construct generative timbre spaces, allowing interpolation and extrapolation of instrument timbres as well as descriptor-based synthesis (Esling, Chemla-Romeu-Santos, & Bitton, 2018). Tatar et al. proposed another VAE architecture and a graphical interface for latent timbre synthesis, aiming for reconstruction of arbitrary audio excerpts rather than just instrument timbres (Tatar, Bisig, & Pasquier, 2020). Our decision to work with GANSynth is motivated by its fast generation performance, which makes it appealing for real-time applications.

3 GANSpaceSynth Hybrid Architecture Properties

The hybrid architecture is based on the idea that we could simply apply the GANSpace method to another GAN model (GANSynth). Despite significant differences between the image and audio domains, we have been motivated by the generality of the GANSpace technique, and the success of this method applied to Progressive GAN on which GANSynth is based.

GANSynth is implemented in the Python programming language using the TensorFlow (Abadi et al., 2016) open-source machine learning library. TensorFlow provides tools for e.g. defining the model's structure, feeding data into the model as well as training and generating on both CPUs and GPUs. Like all GANs, GANSynth is composed of a generator and a discriminator network. The generator takes as input a 256-dimensional vector $\mathbf{z}$ and a number denoting pitch. These are concatenated to form the input layer, which then feeds into a stack of upsampling 2D convolutions that generate spectrograms at progressively higher resolutions. The generator's output is a spectrogram of size 128 frames $\times$ 1024 frequency bins $\times$ 2 channels (amplitude and phase). The discriminator runs such spectrograms through a stack of downsampling 2D convolutions mirroring the generator, and produces as output a divergence estimate between real and generated data. In addition, a pitch classifier tries to predict the pitch label. The divergence estimate and pitch prediction error are combined to form the network's loss function, which is optimized during training (Engel et al., 2019).

The other part of the hybrid architecture, GANSpace, presents a method for finding significant latent space directions in a trained GAN model (Härkönen et al., 2020). This is done by sampling a large number of random input vectors and performing a dimensionality reduction, specifically Principal Component Analysis (PCA), on the activation space of an early layer of the network. In the image domain, the directions can map to a variety of semantic image features, such as

viewpoint, aging and lighting of generated faces. The features can be controlled by moving along these direction vectors. GANSpace uses NetDissect(Bau, Zhou, Khosla, Oliva, & Torralba, 2017) to hook into the early layers of the network. NetDissect supports the deep learning frameworks Caffe and PyTorch, but not TensorFlow, which GANSynth is implemented in. In GANSpaceSynth, we instead created a simple hook mechanism by exposing inner layers to the outside and adding TensorFlow placeholders to allow editing the layers' activations. Figure 1 shows the components diagram of the hybrid architecture.

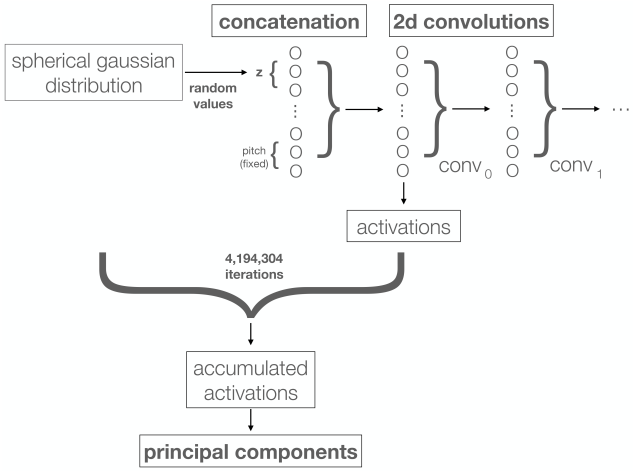

Fig. 1. The block diagram of the GANSpace method applied to GANSynth model for generating the principal components list.

Given a trained GANSynth model, we sample random input vectors and feed them into the network. Using the estimators code from GANSpace, we compute a PCA of the aggregated activations on an early convolutional layer. To avoid running out of memory, we split the computation into batches and use the incremental PCA estimator from Scikit-learn (Pedregosa et al., 2011). The output of this computation includes the principal components (i.e. a list of orthogonal directions in the latent space, sorted from most to least significant) as well as a global mean and standard deviation for each component.

3.1 Organising the Latent Space Using a Dimensionality Reduction

GANSynth's input vector **z** is sampled from a spherical Gaussian distribution and constitutes a point in the model's latent space as shown in Figure 1. The latent space essentially functions as a compact representation of timbres learned during training. The problem of GAN interpretability comes from the difficulty of understanding the structure of this latent space. With the PCA output from

GANSpace, it becomes possible to navigate the latent space of a GANSynth model in a more controlled manner. We form a linear combination of the principal components, scaled by coefficients given by the musician (defaulting to zero) and multiplied by the corresponding standard deviations. We then add this to the global mean to obtain a point to synthesise from. Essentially, we treat the global mean vector as the center of the space and allow the musician to specify how far to move from the center along each principal component direction. Because this computation takes place on a hidden layer of the network, we bypass the network's input layer completely, meaning $\mathbf{z}$ no longer has any influence on the sound. Since the input layer also contains the labels for pitch conditioning, a side effect is that we lose the ability to control pitch. This is a drawback compared to VAE-based approaches such as (Esling et al., 2018). It may be possible to design a way to retain pitch control in the GAN setting, however this has not been a priority in our research. We are mainly interested in using GANSynth unsupervised, with datasets of arbitrary samples rather than single notes with a defined pitch, allowing us to skip laborious processes of labeling.

Table 1. Authors' Interpretation of Perceived Audio Characteristics from PCA Components

$(-1, -1)$ muffled texture	$(0, -1)$ bright texture	$(1, -1)$ airy texture
$(-1, 0)$ quiet muffled texture	$(0, 0)$ quiet bright texture	$(1, 0)$ quiet airy texture with whistling
$(-1, 1)$ very quiet muffled texture	$(0, 1)$ very quiet bright texture	$(1, 1)$ near silence with quiet whistling

In initial experiments with Magenta's `all_instruments` model, on the first convolutional layer and with 4,194,304 samples, we found that the PCA indeed does reveal some significant directions in the model's latent space. However, ascribing semantic meaning to these is difficult, therefore we also experimented with other models. The model `ct-conversations` was trained on recordings of "KET conversations" performed by Thomas Bjelkeborn and Koray Tahiroğlu, with all pitch labels set to the same value. The model mostly generates inharmonic ambiences and textures. The table 1 lists perceived characteristics of these sounds from exploring the first two PCA components at specific points on the plane spanned by the top two principal components.

With the first component, we see an influence on the generated sounds' brightness and tone, and the second component mainly affects loudness. The third component has almost no noticeable effect. Significant entanglement does occur in all the models we have tested and even small movements along a single

axis can produce more variation than suggested in table 1. We invite readers to listen to the generated samples[1]. The entanglement may be due to factors such as dataset quality, the choice of GAN layer, the number of samples used for PCA, and the choice of PCA for dimensionality reduction.

3.2 Real Time Audio Synthesis

While Python scripts can be used to generate audio with GANSpaceSynth, our goal is to bring this work closer to musicians and instrument builders by integrating it with the Pure Data (Pd) visual programming language for audio processing. Using the Pyext external, which allows writing Pd objects in Python, we implemented a Pd interface for generating sounds with GANSpaceSynth.

Pyext previously only had support for the now-obsolete Python 2, so we had to modify it to support Python 3. This was a major undertaking as the Python APIs changed significantly between these versions. At the time of writing, our fork of Pyext can be built on macOS and Linux, but we hope to add Windows support in the future. Our Pd object allows loading a trained model and PCA components from files, setting coefficients for the PCA components and generating. Generated audio is written to standard Pd arrays.

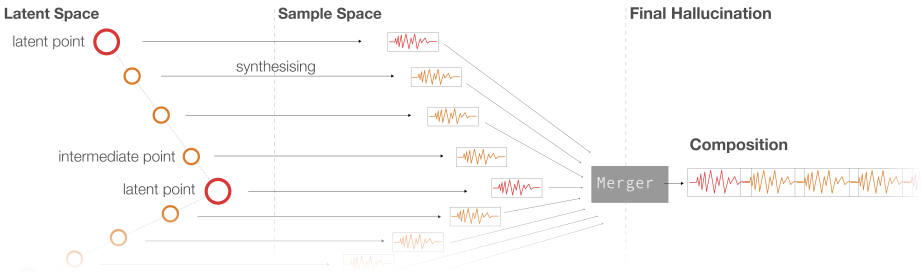

Fig. 2. Hallu provides a controlled structure to interpolate between points in latent space through intermediate points to generate audio samples for the final composition.

4 Hallu Composition Tool using GANSpaceSynth

In addition to the developments of the autonomous AI-terity instrument (Tahiroğlu et al., 2021), one of the outcomes of this research has been to build a composition tool around audio deep neural networks. We implemented GANSpaceSynth to generate hallucinatory audio effects similar to what has been made with other GAN models in the image domain.

The term "hallucination" has been used in a few different ways in relation to machine learning. Face hallucination has been introduced as a term

[1] `https://tinyurl.com/jemrejrv` (Sound samples & their spectrogram images)

to describe techniques for super-resolution and sketch-photo synthesis of faces (Wang, Tao, Gao, Li, & Li, 2013). On the other hand, images generated by Deep Dream showed similarities to biological visual hallucinations (Suzuki, Roseboom, Schwartzman, & Seth, 2017). Data augmentation using generative models has been referred to as data hallucination as well (Lin, Wang, Lei, & Chen, 2019). Common to these definitions is the use of generative models to synthesize images that were not in the training dataset. While these definitions are closely linked the term hallucination we use in Hallu composition tool, at the same time we refer hallucination to the act of exploration of the latent space.

Since GANSynth generates four-second clips of audio, our approach to composing longer pieces involves generating many such clips and stitching them together. As the figure 2 shows, we pick points in the latent space, interpolate between them to compute some number of intermediate points, and for each point we synthesise the audio corresponding to that point. Our first implementation was relatively simple, because the GANSynth repository already had examples of how to randomly sample the latent space and how to interpolate between these points. The generated hallucinations were essentially random walks in the latent space. GANSpaceSynth opened up the possibility of more controlled hallucinations to be part of the composition tool.

To hallucinate with GANSpaceSynth, we now use the PCA components to specify points in the latent space. The Hallu tool[2] consists of a Pure Data patch interfacing with GANSpaceSynth. First, the musician loads a trained model and an associated PCA result file. They are then able to add a desired number of steps. Each step can be edited individually by adjusting sliders corresponding to the PCA components. These steps constitute the principal points through which the hallucination will travel. The musician can listen to the sound generated at the selected point by clicking the preview button. A global settings section allows specifying the number of interpolation steps between each principal step, as well as parameters for stitching together the generated samples. These parameters include sample spacing (the time between playback start of two consecutive samples), start trim (how much to trim from the start of generated samples) as well as an amplitude envelope with attack, sustain and release durations for each sample. Having specified the parameters, the musician can start the generation process and listen to the generated hallucination once generation is complete.

Given n principal steps and m interpolation steps, Hallu generates $n + (n - 1)m$ distinct points in latent space. We use simple linear interpolation to obtain the intermediate steps, taking m samples at even spacings. How good the generated hallucinations sound depends greatly on the data the GANSynth model was trained on. Using the NSynth dataset resulted in hallucinations which aren't very interesting, mainly because the dataset consists of single notes. We trained a number of GANSynth models on different custom datasets to find out what kind of models could generate recognisable variations in hallucinations. The initial problem we faced was that it takes a significant effort to create a good quality

[2] Hallu is an open source composition tool `https://github.com/SopiMlab/GANSpaceSynth/`

dataset to train GANSynth on. We ended up creating a Python script that takes longer audio tracks and splits them into 4 second samples, which allowed us to prepare a diverse sounding datasets. In some trained models, hallucination could result in diverse sounds in comparison to the original dataset, on the other hand with other models, generated hallucinations sounded very similar to the original tracks while still being original in our subjective opinion [3].

5 Conclusions

In this paper, we presented the GANSpaceSynth hybrid method, which contributes to the work in generative systems, in audio domain, that makes it possible exploring GAN latent space with more awareness of what is happening musically and having the opportunity to control the development of musical creativity in a human-musician and AI cooperation. GANSpaceSynth hybrid method is a novel approach to achieve an unconditional generative model based on organising the relationship between points in the latent space (audio features) and the generated audio samples, rather than in random order.

Compared to GANSpace in the image domain, the semantic editing capabilities demonstrated by GANSpaceSynth so far are more limited. In the future, we intend to refine our approach by further investigating the interaction between choices of dataset, dimensionality reduction method, layer and sample count. Nevertheless, the edits we find do give us more control over exploring the range of sounds generated by GANSynth models.

The approach has now been applied to produce music based on generative hallucination method as well as to build the AI-terity musical instrument with autonomous features. We propose Hallu as a platform for experiencing co-creativity, and in future work we are aiming to reflect other musicians' experiences with using it. It is in our research interest to explore further creative strategies for GANSpaceSynth in the use of building autonomous musical instruments performed by two musicians sharing the same GAN space, which may open up new variety of musical demands other than we only had imagined was possible. We have limited our discussions to the controlled interaction in GAN latent space in this paper. An interesting and exciting future may bring in another focus in our research, towards more structural use of *neural network pruning* in GANSpaceSynth to improve the performance of the audio sample generation on CPU. The open source code of the project is available at `https://github.com/SopiMlab/GANSpaceSynth/`.

6 Acknowledgement

This work was supported by the Academy of Finland (project 316549) and A!OLE funding. We acknowledge the third reviewer for his/her constructive comments and effort to improve our paper.

[3] Examples of Hallu compositions are a available at `https://tinyurl.com/cpy4urh6`

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... others (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Bau, D., Zhou, B., Khosla, A., Oliva, A., & Torralba, A. (2017). Network dissection: Quantifying interpretability of deep visual representations. In *Computer vision and pattern recognition.*

Briot, J.-P., & Pachet, F. (2020). Deep learning for music generation: challenges and directions. *Neural Computing and Applications*, *32*(4), 981–993. Retrieved from `https://doi.org/10.1007/s00521-018-3813-6` doi: 10.1007/s00521-018-3813-6

Carnovalini, F., & RodÃ, A. (2020). Computational creativity and music generation systems: An introduction to the state of the art. *Frontiers in Artificial Intelligence*, *3*, 14. Retrieved from `https://www.frontiersin.org/article/10.3389/frai.2020.00014` doi: 10.3389/frai.2020.00014

Engel, J., Agrawal, K. K., Chen, S., Gulrajani, I., Donahue, C., & Roberts, A. (2019). GANSynth: Adversarial neural audio synthesis. In *International conference on learning representations.* Retrieved from `https://openreview.net/forum?id=H1xQVn09FX`

Engel, J., Hantrakul, L., Gu, C., & Roberts, A. (2020). *Ddsp: Differentiable digital signal processing.*

Esling, P., Chemla-Romeu-Santos, A., & Bitton, A. (2018). *Generative timbre spaces: regularizing variational auto-encoders with perceptual metrics.*

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). *Generative adversarial networks.*

Hantrakul, H., Zada, N., Carney, M., Bowers, M., Li, C., Toh, E., ... Engel, J. (2020). *Tone transfer.* `https://sites.research.google/tonetransfer`. (Accessed: 2020-10-26)

Härkönen, E., Hertzmann, A., Lehtinen, J., & Paris, S. (2020). *Ganspace: Discovering interpretable gan controls.*

Kingma, D. P., & Welling, M. (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, *12*(4), 307–392. Retrieved from `http://dx.doi.org/10.1561/2200000056` doi: 10.1561/2200000056

Lin, C.-C., Wang, Y.-C. F., Lei, C.-L., & Chen, K.-T. (2019). Semantics-guided data hallucination for few-shot visual classification. In *2019 ieee international conference on image processing (icip)* (p. 3302-3306). doi: 10.1109/ICIP.2019.8803420

McInnes, L., Healy, J., & Melville, J. (2020). *Umap: Uniform manifold approximation and projection for dimension reduction.*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, *12*, 2825–2830.

Roberts, A., Engel, J., Raffel, C., Hawthorne, C., & Eck, D. (2018, 10–15 Jul). A hierarchical latent vector model for learning long-term structure in music.

In J. Dy & A. Krause (Eds.), *Proceedings of the 35th international conference on machine learning* (Vol. 80, pp. 4364–4373). PMLR. Retrieved from http://proceedings.mlr.press/v80/roberts18a.html

Suzuki, K., Roseboom, W., Schwartzman, D. J., & Seth, A. K. (2017, November). A deep-dream virtual reality platform for studying altered perceptual phenomenology. *Scientific Reports*, *7*(1). Retrieved from https://doi.org/10.1038/s41598-017-16316-2 doi: 10.1038/s41598-017-16316-2

Tahiroğlu, K., Kastemaa, M., & Koli, O. (2020, July). Al-terity: Non-rigid musical instrument with artificial intelligence applied to real-time audio synthesis. In R. Michon & F. Schroeder (Eds.), *Proceedings of the international conference on new interfaces for musical expression* (pp. 337–342). Birmingham, UK: Birmingham City University. Retrieved from https://www.nime.org/proceedings/2020/nime2020_paper65.pdf

Tahiroğlu, K., Kastemaa, M., & Koli, O. (2021, 6 15). Ai-terity 2.0: An autonomous nime featuring ganspacesynth deep learning model. *International Conference on New Interfaces for Musical Expression*. Retrieved from https://nime.pubpub.org/pub/9zu49nu5 (https://nime.pubpub.org/pub/9zu49nu5)

Tatar, K., Bisig, D., & Pasquier, P. (2020, October). Latent timbre synthesis. *Neural Computing and Applications*, *33*(1), 67–84. Retrieved from https://doi.org/10.1007/s00521-020-05424-2 doi: 10.1007/s00521-020-05424-2

Vogl, R., & Knees, P. (2016). An intelligent musical rhythm variation interface. In *Companion publication of the 21st international conference on intelligent user interfaces* (p. 88–91). New York, NY, USA: Association for Computing Machinery. Retrieved from https://doi.org/10.1145/2876456.2879471 doi: 10.1145/2876456.2879471

Wang, N., Tao, D., Gao, X., Li, X., & Li, J. (2013, August). A comprehensive survey to face hallucination. *International Journal of Computer Vision*, *106*(1), 9–30. Retrieved from https://doi.org/10.1007/s11263-013-0645-9 doi: 10.1007/s11263-013-0645-9

Zhou, Y., Koyama, Y., Goto, M., & Igarashi, T. (2020, October). Generative Melody Composition with Human-in-the- Loop Bayesian Optimization. In *Proceedings of the 1st Joint Conference on AI Music Creativity* (p. 10). Stockholm, Sweden: AIMC. Retrieved from https://doi.org/10.5281/zenodo.4285364 doi: 10.5281/zenodo.4285364