EVOLUTIONARY ALGORITHM FOR PROCEDURAL GENERATION OF AUDIO FROM TIME-FREQUENCY ANALYSIS OF DIGITAL SAMPLES

Alex Wang
EE230C Class Project
UCLA Winter 2018

ABSTRACT

This project details a method for generating new audio waveforms given a set of input audio samples. The magnitudes of the short-time Fourier transforms of the inputs are decomposed by non-negative matrix factorization, and the resulting components can be placed in linear combinations to generate spectrograms. A genetic algorithm is implemented to stochastically train the weights of these linear combinations, and output audio samples can be estimated from the generated spectrograms via the Griffin-Lim method.

1 INTRODUCTION

Modern music has been revolutionized by the advent of two electronic instruments: the sampler and the synthesizer. The synthesizer is able to create original musical timbres by manually combining and altering a set of base waveforms. For example, a popular sound in electronic music called Reese bass can be synthesized by layering a low-frequency sine wave with two sawtooth waves of slightly different frequencies. The destructive interference between the sawtooth waves give the sound a unique tremolo-like effect which was widely influential in the house music scenes of the UK. The sampler on the other hand replays stored sound, and cannot create new sounds on its own. New sounds can be made by using the sampler in tandem with other devices; one prominent example is a production style pioneered by recording artist Kanye West, in which hip hop instrumentals are created by looping pitched-up samples of soul music. However, in most cases the sampler is simply used to inject sounds directly drawn from other sources. In fact, many modern genres can be characterized by the nature of their respective samples. The drum pattern of any modern song that is not recorded acoustically is sequenced using sampled percussion, and these samples are frequently compiled into drum kits and shared online. Due to this high accessibility, a large majority of today's music production ends up sounding unoriginal.

The overuse of popular samples has resulted in musical saturation, and the most recognizable sample found in pop music today is the 808. This bass sound was originally a preset found on an analog synthesizer from the 1980s called the Roland TR-808. It has been used so frequently that producers have resorted to modifying the sample of the 808 to craft more unique sounding instrumentals rather than resynthesize new sounds. The primary motivation behind this project is to automate the creation of new samples as a means of dynamically expanding the possible timbres available for a producer to sample. Similar to how a synthesizer modifies a set of base waveforms to create new waveforms, this project provides a formulation for modifying a set of base samples to create new samples. A genetic algorithm is used to randomly train parameters, and this is simply to determine the baseline level of output quality that is feasible with this technique. Other autoencoding methods are described which can employ similar heuristics.

2 Griffin-Lim Method

Many applications require recovering a signal from a modified spectrogram. The spectrogram of a signal is obtained by taking the magnitude squared of the short-time Fourier transform (STFT) of the signal, and an algorithm to estimate the signal from a given spectrogram has been established by Griffin and Lim [1]. This algorithm works iteratively to minimize the mean squared error between the target STFT and STFT of the estimated signal. In this section we follow the derivations for the update equation of the Griffin-Lim method.

Let $x(n)$ and $X_w(mS,\omega)$ denote a real sequence and its STFT. The variable S is a positive integer representing the sampling period of $X_w(n,\omega)$, and the analysis window used in the STFT is denoted by $w(n)$ and is assumed to be real and nonzero for $0 \le n \le L-1$. From the definition of the STFT

$$X_w(mS,\omega) = F_l[x_w(mS,l)] = \sum_{l=-\infty}^{\infty} x_w(mS,l)e^{-j\omega l}$$

where

$$x_w(mS,l) = w(mS-l)x(l)$$

and $F_l[x_w(mS,l)]$ represents the Fourier transform of $x_w(mS,l)$ with respect to l. Let $Y_w(mS,\omega)$ denote the target STFT, and let $y_w(mS,l)$ be given by

$$y_w(mS,l) = \frac{1}{2\pi}\int_{\omega=-\pi}^{\pi} Y_w(mS,\omega)e^{j\omega l}d\omega.$$

In general, the target STFT may be modified and does not necessarily equal the STFT of any possible signal, but a sequence $x(n)$ can be found whose STFT is the closest to $Y_w(mS,\omega)$ using the following distance measure:

$$D[x(n),Y_w(mS,\omega)] = \sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int_{\omega=-\pi}^{\pi}|X_w(mS,\omega)-Y_w(mS,\omega)|^2 d\omega.$$

This distance measure is the squared error between $X_w(mS,\omega)$ and $Y_w(mS,\omega)$ integrated over all ω and summed over all m. This can be rewritten using Parseval's theorem as

$$D[x(n),Y_w(mS,\omega)] = \sum_{m=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}[x_w(mS,l)-y_w(mS,l)]^2$$

In order to minimize $D[x(n),Y_w(mS,\omega)]$ the gradient with respect to $x(n)$ is set to zero, which yields the following result:

$$x(n) = \frac{\sum_{m=-\infty}^{\infty} w(mS-n)y_w(mS,n)}{\sum_{m=-\infty}^{\infty} w^2(mS-n)}$$

An iterative procedure is used to estimate $x(n)$ from the magnitude of the target STFT. With every iteration of this algorithm, squared error between $|X_w(mS,\omega)|$ and $|Y_w(mS,\omega)|$ decreases. Let $x^i(n)$ denote the estimated $x(n)$ after the ith iteration. $x^{i+1}(n)$ is obtained by taking the STFT of $x^i(n)$, replacing $|X_w^i(mS,\omega)|$ with $|Y_w(mS,\omega)|$ and then finding the signal with STFT closest to this modified STFT. The iterative algorithm results in the following update equation:

$$x^{i+1}(n) = \frac{\sum_{m=-\infty}^{\infty} w(mS-n)\frac{1}{2\pi}\int_{\omega=-\pi}^{\pi} \hat{X}_w^i(mS,\omega)e^{j\omega n}d\omega}{\sum_{m=-\infty}^{\infty} w^2(mS-n)}$$

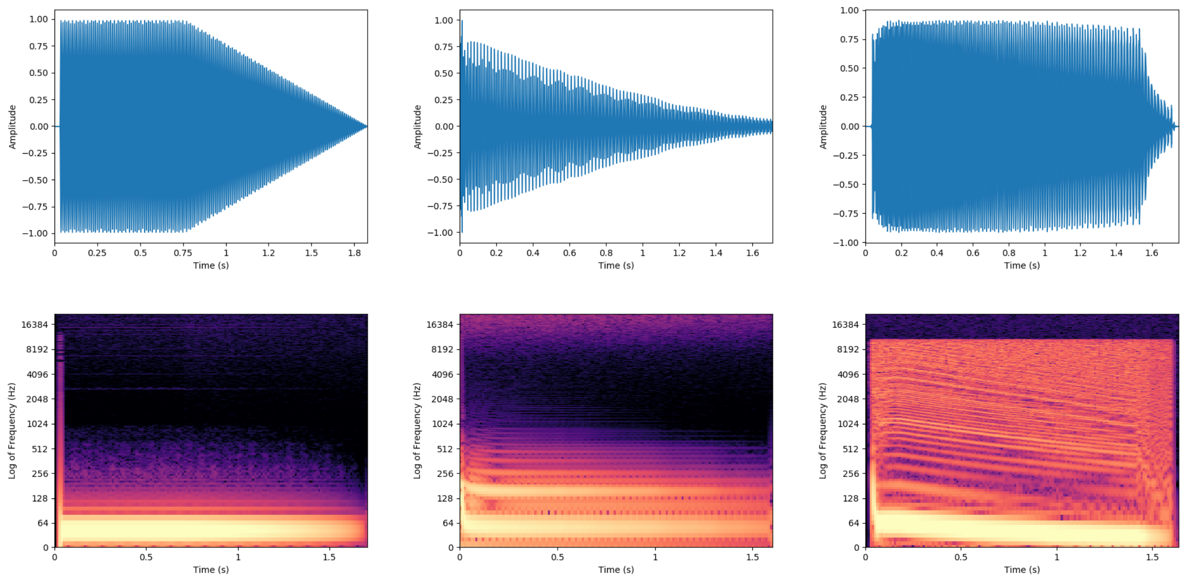
where

$$\hat{X}_w^i(mS,\omega) = |Y_w(mS,\omega)|\frac{X_w^i(mS,\omega)}{|X_w^i(mS,\omega)|}.$$

In the case where $\left|X_w^i(mS,\omega)\right| = 0$, $\hat{X}_w^i(mS,\omega)$ is set to $|Y_w(mS,\omega)|$. With each iteration, the distance measure

$$D_M[x(n),|Y_w(mS,\omega)|] = \sum_{m=-\infty}^{\infty}\frac{1}{2\pi}\int_{\omega=-\pi}^{\pi}[|X_w(mS,\omega)| - |Y_w(mS,\omega)|]^2 d\omega$$

is decreased, and the paper by Griffin and Lim goes on to prove that the algorithm always converges to a set consisting of the critical points of D_M as a function of $x(n)$.



(a) (b) (c)

Figure 1: Three 808 audio samples are used as test input for the implementation of the Griffin-Lim method. The first sample is a sub-bass sound (a) that has frequencies mostly under 100 Hz. The second sample is the classic tuned 808 (b) that has a range of higher frequencies. The third sample is a distorted 808 (c) with a descending chirp-like feature. The samples sound crisp and are indistinguishable from synthesized sounds.

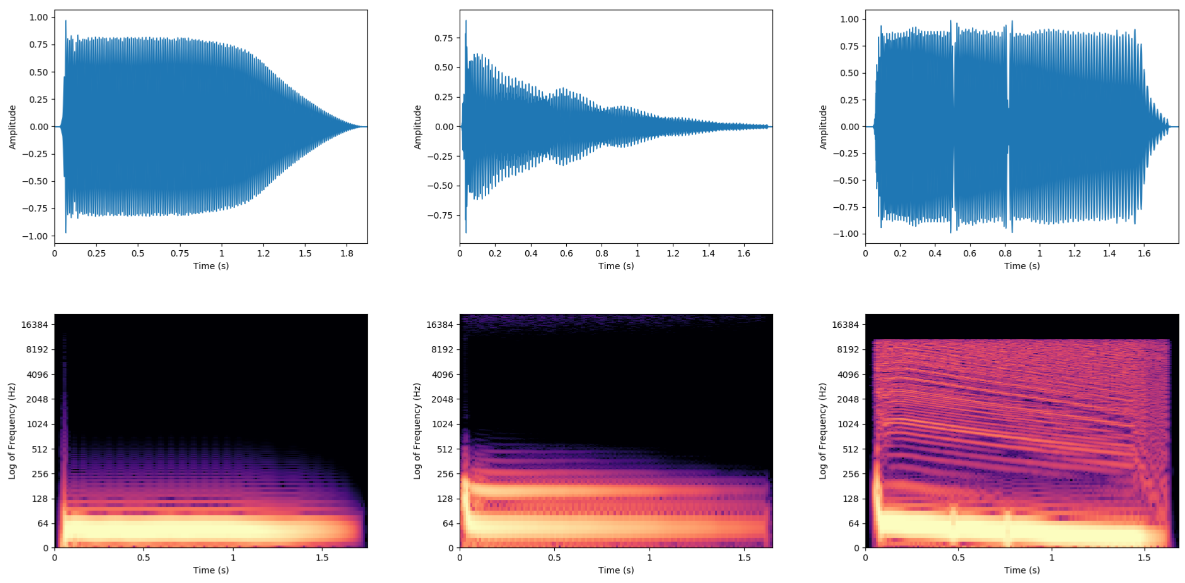


Figure 2: The output of the Griffin-Lim algorithm when ran for 50 iterations on the three samples from Figure 1. The spectrograms appear to be precisely replicated with a reduction in background noise, but the waveforms show apparent differences in their respective amplitude envelopes. Listening to the audio files reveals that the replicated samples sound muddier and have lower volumes in comparison to the inputs, but the timbre is faithfully recreated.

3 Non-negative Matrix Factorization

A widely used tool for the analysis of higher-dimensional data, non-negative matrix factorization (NMF) is able to automatically extract sparse and meaningful features from a set of non-negative data vectors [2]. NMF has demonstrated success in efficiently separating multiple audio sources within a single channel [3], and it is also effective in the automatic classification of musical instrument sounds [4]. However, there has been no previous work done on exploring the reconstruction of the features obtained from NMF in order to create new timbres. In this section we will present the multiplicative update rules needed to implement NMF.

Given a set of multivariate n-dimensional data vectors, the vectors are placed in columns of an $n \times m$ matrix V where m is the number of examples in the data set. This matrix is then approximately factorized into an $n \times r$ matrix W and an $r \times m$ matrix H. Each data vector v is approximated by a linear combination of the columns of W weighted by the components of h, where v and h are the corresponding columns of V and H.

To find an approximate factorization $V \approx WH$, we can employ two possible cost functions to measure the distance between two non-negative matrices. One such measure is the square of the Euclidean distance between the two matrices:

$$\|A - B\|^2 = \sum_{ij} \left(A_{ij} - B_{ij}\right)^2.$$

Another useful measure is

$$D(A\|B) = \sum_{ij}\left(A_{ij}\log\frac{A_{ij}}{B_{ij}} - A_{ij} + B_{ij}\right)$$

which reduces to the Kullback-Leibler divergence, or relative entropy, when $\sum_{ij} A_{ij}$ and $\sum_{ij} B_{ij}$ are both equal to one. These two measures are both lower bounded by zero and vanish if and only if $A = B$. Although the functions $\|V - WH\|^2$ and $D(V\|WH)$ are convex in W only or H only, they are not convex in both variables together, so it is difficult to find a global minima. The following multiplicative update rules provide convergence towards local minima:

$$H_{a\mu} \leftarrow H_{a\mu}\frac{(W^TV)_{a\mu}}{(W^TWH)_{a\mu}} \qquad W_{ia} \leftarrow W_{ia}\frac{(VH^T)_{ia}}{(WHH^T)_{ia}}$$

This ensures the Euclidean distance $\|V - WH\|$ is non-increasing.

$$H_{a\mu} \leftarrow H_{a\mu}\frac{\sum_i W_{ia}V_{i\mu}/(WH)_{i\mu}}{\sum_k W_{ka}} \qquad W_{ia} \leftarrow W_{ia}\frac{\sum_\mu H_{a\mu}V_{i\mu}/(WH)_{i\mu}}{\sum_\nu H_{a\nu}}$$

This ensures the relative entropy $D(V\|WH)$ is non-increasing. Each update consists of multiplication by a factor, and the factor is unity when $V = WH$ so that perfect reconstruction is necessarily a fixed point of the update rules.

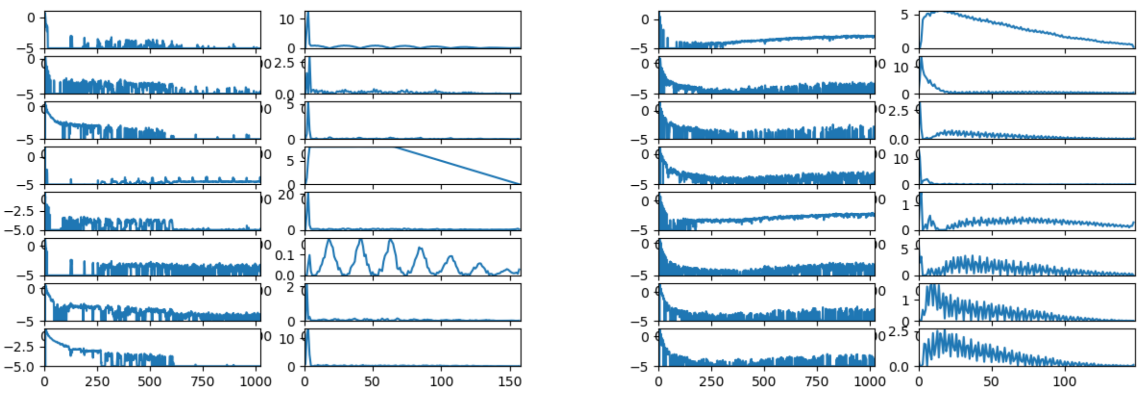


Figure 3. The first and second columns of plots on the left show the NMF decomposition of the spectrogram from Figure 1a. The first column shows various frequency spectra, which are column vectors of the matrix W. The second column shows the temporal activations, which are column vectors of the matrix H. The third and fourth columns on the right show the NMF decomposition for the spectrogram from Figure 1b. In order to reconstruct the original sample's spectrogram from these components, outer products are taken between each frequency spectrum and its corresponding temporal activation, and the resulting matrices are summed. Source separation is achieved by only summing together certain components. For our single-source audio samples, we can consider these components to be subdivisions of timbre.

4 Application of a Genetic Algorithm

A single audio sample can be decomposed using NMF into an assortment of frequency spectra, each with a corresponding temporal activation. This provides a basis for musical timbres that can be used to create new timbres. In this section, a genetic algorithm is proposed which draws genes from this basis in order to generate and evolve new audio samples.

Given a set of input audio samples, a "gene pool" can be created by taking each input's spectrogram and decomposing it using NMF. An "organism" is created by randomly selecting "genes" from the gene pool. In order for each organism to have a proper set of basis vectors for recombination into an output spectrogram, the genes cannot be selected completely at random. For this simple proof-of-concept, we divide the gene pool into "temporal genes" and "spectral genes," and we limit the organisms to a set of "species." The set of possible species is defined by the set of input audio samples. If two organisms are members of the same species, then they have the exact same temporal genes, which are the temporal activations of the input sample that is the representative of said species. This is to ensure that each output waveform has a similar amplitude envelope compared to the input waveforms. Many of the temporal activations resulting from the NMF decomposition oscillate rapidly, and recreating audio from these components result in a grainy and stuttered sound rather than a single continuous sound. On the other hand, spectral genes are not confined to species and can be freely superposed in order to generate unique timbres.

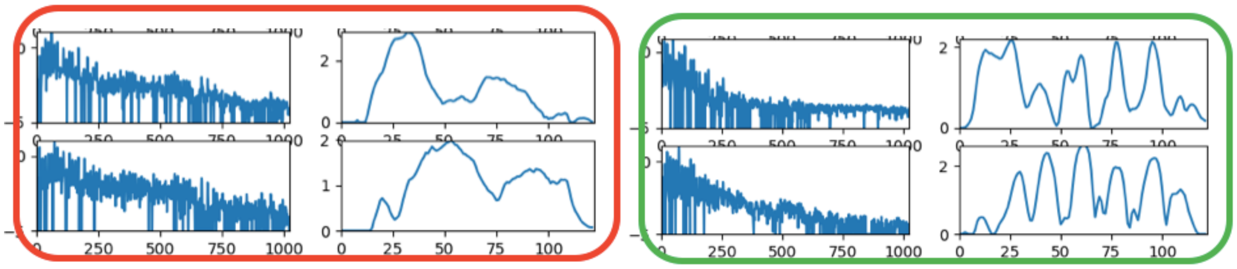


Figure 4. The red outline contains genes for an input clarinet sample, with spectral genes on the left and temporal genes on the right. The green outline contains genes for an input flute sample.

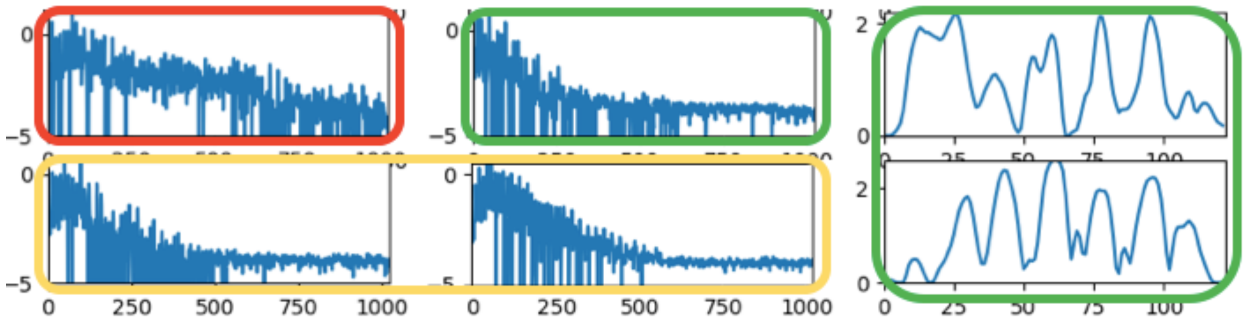


Figure 5. This shows an example organism which is a member of the flute species. The temporal genes on the right are from the flute, but the spectral genes come from the clarinet, the flute, and a random third species outlined in yellow. In order to reconstruct the spectrogram, the red and green genes are linearly combined into a single spectrum, and the yellow genes are linearly combined into another spectrum. For each row, the outer product is taken between the combined spectrum and the corresponding temporal activation, and the resulting matrices are summed.

Conventional methods of crossover and mutation are used to provide genetic variation. Each organism is initialized by first randomly selecting a species, and then for each temporal activation of the species, random spectral genes are selected with random weights. The genetic algorithm then iterates through "generations," where only a fraction of the organisms survives after each generation. From the survivors, new organisms are spawned which are genetic variations of the survivors. For this simulation, three mutation possibilities are implemented. There is a chance for crossover, where genes are swapped from other surviving organisms. There is also a chance for an organism to gain a completely new spectral gene from the gene pool. Finally, the last mutation will always happen but with a random amplitude: the weights corresponding to the spectral genes are slightly redistributed. The weights are for when the spectral genes are linearly combined. If a weight is no longer positive, the corresponding spectral gene is removed. In order to prevent the output audio from becoming too loud, weights cannot increase past one.

The genetic algorithm selects survivors using two fitness metrics. The first metric is the integral of the mean squared error of the power spectral density,

$$fit_A(org) = -\int \frac{1}{N}\sum_{i}^{N}\left(S_i(f) - S_{org}(f)\right)^2 df$$

where $S_i(f)$ is the power spectral density of one of the input audio samples and $S_{org}(f)$ is the power spectral density of the organism whose fitness is being determined. The negative sign is so that lower mean squared error gives higher fitness. The second metric is the integral of the average coherence,

$$fit_B(org) = \int \frac{1}{N}\sum_{i}^{N}\frac{\left|S_{i,org}(f)\right|^2}{S_i(f)S_{org}(f)}df$$

where $S_{i,org}(f)$ is the cross-spectral density between the organism's waveform and one of the input audio samples. The actual implementations sum over frequency bins rather than integrating. Simulations using the fitness metric based on coherence proved to be more efficient, so in each generation the organisms are sorted by $fit_B(org)$ and $fit_A(org)$ is used as a tie-breaker. After many generations of selective pressure, the resulting organisms end up with high fitness according to both metrics, whereas sorting by fit_A first does not lead to organisms with high fit_B. Because of this, a hierarchical fitness metric is used rather than a weighted sum of the two metrics.

Once organisms have been selected, their genes can be recombined into spectrograms, and the Griffin-Lim method can then be used to construct audio samples from said spectrograms.

5 RESULTS

The genetic algorithm is executed on four sets of data. The set of three 808 samples from Figure 1 yields good results due to the inherently similar timbres between the inputs. The generated samples sound unique compared to the inputs, but are still recognizable as 808 bass sounds.

A set of synth samples from the Moog Opus-3 synthesizer also creates interesting timbres, even though the inputs have more variation. Nevertheless, the sounds aren't as crisp as actual synths.

Using a set of acoustic tom-tom drum samples produces interesting distorted percussive sounds, but does not create any timbres that sound significantly different from the inputs. The input timbres are all very similar, so deviations may tend to be selected against.

A set of orchestral brass, woodwind, and string instrument samples also does not create any timbres that are distinguishable from the inputs. This may be due to the fact that the only organisms that survive are ones that sound very similar to the inputs.

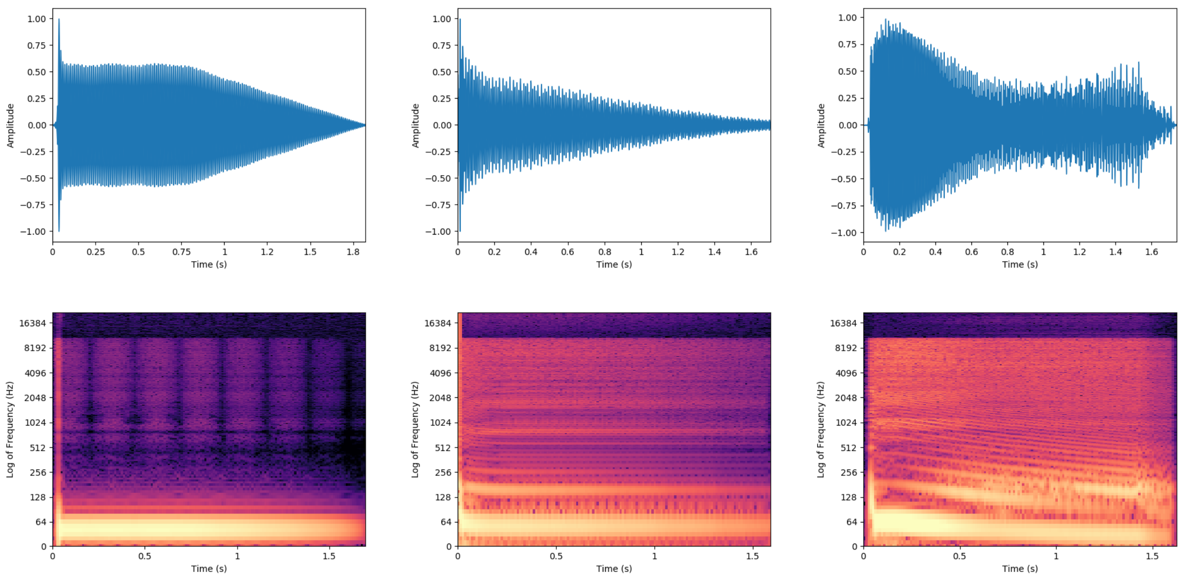


Figure 6. Three procedurally generated 808 samples, one for each species corresponding to the input samples from Figure 1.

6 Discussion

The genetic algorithm proposed in this project yields original audio samples that can reasonably be used in music production. The reconstruction of audio samples from a given set of temporal activations and a random linear combination of frequency spectra does create new timbres. There are still many areas with room for improvement.

While training does increase the fitness of the organisms, after many generations a single species always begins to dominate. Even after adjusting the mutation rates and the proportion of survivors, the final generation is usually all one species. One simple remedy is to simply have a low number of generations and a high number of organisms per generation, and this is effective because the output quality does not benefit from a large number of generations, and only new timbres are desired rather than optimal ones. Another solution is to change the fitness metric, because the metrics proposed tend to favor certain species depending on the power spectral densities of the input samples.

Another major flaw is the speciation constraint. Further work can be done by allowing random combinations of temporal genes, but techniques must be implemented for ensuring the proper selection of these temporal activations. The final output waveform should have a similar envelope as the input waveforms, but using the same temporal activations result in too much similarity and not enough variation. Temporal genes could be selected based on relative time position so that a final waveform can be more fleshed out and not have any gaps. Perhaps the gene pool resulting from the NMF decomposition can be filtered so that any random selection of genes can result in a waveform with reasonable amplitude envelope.

Additional mutations can involve altering the spectral genes themselves. Allowing modifications to the frequency spectra other than a constant multiplicative factor provides more degrees of freedom for a wider variety of output timbres.

Moving away from the evolutionary architecture, there is a lot of potential for future developments with other machine learning algorithms. A neural network can be constructed for the weights of the spectral genes, and with the right implementation and cost function, gradient descent can be used. The most promising method is progressive growth of generative adversarial networks (GAN). Recently GANs have demonstrated the ability to learn from photographs of faces and create realistic images of faces of people that don't exist [5]. This application would be perfect for exploring new musical timbres. GANs work by simultaneously growing two neural networks, one as a generator to create new data and the other as a discriminator which attempts to distinguish generated data from given sample data. This architecture can be implemented in the time domain rather than working with spectrograms, and other work also points in this direction as well. Recently, high quality waveforms have been generated by WaveNet, a deep neural network which learns from dilated causal convolutions on the time-domain sequences of audio signals [6]. WaveNet has been shown to greatly outperform the Griffin-Lim method in replicating and modifying audio signals [7].

7 Acknowledgements

I would like to thank Professor Yao. Prior to this class I had no experience in signals and systems, and his description of the musical score as the oldest form of time-frequency representation inspired me to dive into this fascinating niche of signal processing for music production.

References

[1] D. W. Griffin and J. S. Lim (1984). *"Signal Estimation from Modified Short-Time Fourier Transform," IEEE Trans. on Acoust., Speech, Signal Processing* **32**, pp. 236-243.

[2] D. D. Lee and H. S. Seung (2001). *"Algorithms for Non-negative Matrix Factorization," Advances in Neural Information Processing (NIPS '01)* **13**, pp. 556–562.

[3] C. Demir, M. U. Doğan, A. T. Cemgil and M. Saraçlar (2011). *"Single-channel speech-music separation using NMF for automatic speech recognition," IEEE 19th Signal Processing and Communications Applications Conference (SIU)*, Antalya, pp. 486-489.

[4] E. Benetos, M. Kotti and C. Kotropoulos (2006). *"Musical instrument classification using non-negative matrix factorization algorithms," IEEE International Symposium on Circuits and Systems*, Island of Kos, pp. 4.

[5] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen (2017). *Progressive growing of GANs for improved quality, stability, and variation*, arXiv:1710.10196.

[6] A. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu (2016). *Wavenet: A generative model for raw audio*, arXiv:1609.03499.

[7] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, M. Norouzi, (2017). *Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders*, arxiv.org/abs/1704.01279.