Audio Engineering Society

# Convention Paper

Presented at the 124th Convention
2008 May 17–20 Amsterdam, The Netherlands



# A Framework for Automatic Mixing Using Timbral Similarity Measures and Genetic Optimization

Bennett A. Kolasinski

Music Technology, New York University

New York, NY 10003, USA
bak282@nyu.edu

## ABSTRACT

A novel method is introduced for automatic mix recreation using timbral classification techniques and an optimization algorithm. This approach uses the Euclidean distance between modified Spectral Histograms to calculate the distance between a mix and a target sound and uses a genetic optimization algorithm to figure out the best coefficients for that mix. The implementation has been shown to successfully recreate multitrack mixes accurately and may pave the way towards the automatic mixing of novel multitrack sessions based on a desired target sound.

## 1. INTRODUCTION

The line between content producers and consumers is blurring more rapidly by the day. This can be attributed to two major factors--the ease of access to information via the internet and the ever-lowering bar of entry. Not even a decade ago, basic audio editing and composition was relegated to top-of-the-line computers with expensive outboard gear or dedicated hardware; today, such features are

available even on entry-level computers in ever-shrinking sizes and price points.

Conversely, computers are capable of handling more and more audiovisual information than ever before. Audio editing software on higher-end computers today can handle the recording and playback of dozens or even hundreds of audio tracks with effects on each of them with ease.

Whether you are a novice user trying to make edits and mix music into your latest podcast or you are an expert music editor trying to manage a hundred tracks in a professional recording, you will at some point be faced with dealing with a daunting amount of data. The field of music information retrieval (MIR) is developing tools that may help with this. A system that extracts relevant features from the incoming audio signal and adjusts its presets based on those features would represent a dramatic leap over the static presets commonly found in recording software. Years ago, the notion of a computer making decisions based on such things as recording style and genre seemed far-fetched; now, thanks to great leaps made in timbral similarity and music identification and classification tasks, this does not seem so far off.

## 2. FRAMEWORK

### 2.1. Overview

This paper presents a novel approach to automatically mixing a multitrack recording session.

The approach taken in this paper is unique in that it does not directly analyze the features of each track in a recording at the mixing stage; rather, it mixes by calculating the similarity of the mixes it creates to a target sound. This approach is elegant in that it doesn't require knowledge of the different types of tracks that may be encountered in the mixing task. This means that a completely unknown set of sounds may be mixed using only the target sound.

The mixer implementation applies a gain coefficient to each track and then sums each track together:

$$\sum_{i=1}^{n} track_i * \alpha_i \qquad (1)$$

where $n$ is the total number of tracks in the mix, $track_i$ is the current track, and $\alpha_i$ is the gain to be applied to the current track.

### 2.2. Timbral Similarity

Timbral similarity is used in everything from music recommendation engines to music identification systems. One measure of timbral similarity, the Spectral Histogram (SH), is a histogram of the number of times loudness levels have been exceeded across frequency bands [1]. The frequency bands are defined by the critical bands of the Bark scale and the loudness levels are measured in sones, both of which are psychoacoustic measures: the Bark scale is split up into frequency bands that correspond to the critical bands of human hearing, and the sone is a measure of perceived loudness.

A typical Spectral Histogram is shown in Figure 1, with the Bark bands across the vertical axis and the loudness levels on the horizontal axis.

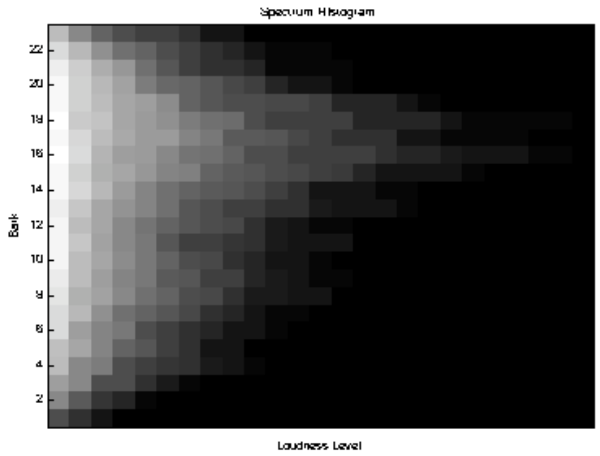


**Fig 1. Spectral Histogram**

The timbral similarity of two sounds can be classified by calculating the distance between two SHs. The Euclidean distance is taken between the Spectral Histograms of a mix and the target mix:

$$\sqrt{SH(mix)^2 - SH(target)^2} \quad (2)$$

This distance metric is used as the genetic algorithm's cost function to figure out the best gain coefficients for each track so that the mix sounds as close to the target mix as possible.

Elias Pampalk's MA Toolbox was used in the calculation of the Spectral Histogram [2]. The SH calculation provided by the MA Toolbox was modified by removing the normalization and scaling the calculation by the overall mean loudness of the mix. This allowed the Euclidean distance between the SHs to accurately locate parameters used in two-track mixing experiments that were used to determine an ideal cost function.

## 2.3. Genetic Optimization

### 2.3.1. Overview

Genetic algorithms (GAs) model their systems just like genes in a biological system, and the genes undergo a number of transformations like mutation and crossover just as they do through sexual reproduction in the living world. Genetic optimization has been shown to be very successful at navigating unwieldy search spaces such as search spaces with many local minima that would confuse normal hill-climbing optimization algorithms.

A toolbox for genetic optimization was used in this implementation. The "Genetic Algorithm Optimization Toolbox" (GAOT) implements a genetic optimization function as well as a number of other functions used by the genetic algorithm for mutation, crossover, and selection. It also provides a framework for customizing each of these functions as needed [3]. Each component of the system is described below.

### 2.3.2. Similar Work

Similar work to the automatic mixing problem has been performed on estimating the parameters for FM synthesis. Andrew Horner et al use a genetic algorithm to adjust the parameters of FM synthesis to closely match the characteristics of an acoustic signal [4]. Tan and Lim also successfully found parameters for double FM synthesis using genetic annealing, a variant on genetic optimization [5]. In both groups' experiments, they compared the harmonics of the acoustic sounds that they were trying to model to the harmonics generated by the FM synthesis as the cost function. This is analogous to the automatic mixing task in that both papers use a

GA to optimize the distance between a system's output and a known target sound.

### 2.3.3. Representation

Just like how the base pairs in a sequence of DNA make up genes that define an organism's attributes, a means of encoding information about a system is needed to serve as the genetic algorithm's 'DNA'.

Since this mixing task's goal is to set the proper gain values for each track, a mix is represented as the sequence of gains to be applied to each track. The gain was defined in voltage change that would be applied to each track; i.e. a gain value of 1 would indicate no gain change and a gain of 0.5 would indicate a -6dB gain applied to the track.

For every iteration of the genetic algorithm, each mix in a population is calculated by mixing the tracks as described in equation (1). The cost function is then utilized to calculate the distance between the resulting mix and the target mix; this cost is recorded and used by the GA's selection function.

### 2.3.4. Initialization

Initialization in a GA typically assigns all the individuals of a population to random values within a known range. This is partly responsible for the ability of the GA to search out large and uneven spaces for the globally optimal solution. In the case of the automatic mixing project, the bounds of the variables define the dynamic range of the signal; for example, they could be defined as [0.001 1.5] to allow for a -60dB to +3.5dB range. The gains are randomly assigned over this range but a future improvement to this algorithm could analyze each track in the mix to provide a more educated guess at initialization.

### 2.3.5. Evolution

An initial population is created of a predetermined size. The genetic optimization function then takes that population and performs a number of operations on it to produce new generations of viable populations until a termination criterion is reached. Since the maximum cost achieved by the GA is dependent on a wide array of factors, in particular the number of tracks being mixed, an ideal maximum cost varies from task to task and as such a fixed number of generations was used as the termination criterion.

### 2.3.6. Reproduction Functions

For each generation, the GA applies crossover and mutation operators to members of the population at a specified rate. In biology, crossover and mutation are responsible for introducing variations in offspring.

**Crossover**

Crossover provides for variation in offspring by transferring portions of one parent's genetic material to the other parent's. The crossover function used utilizes information about each parent's fitness as a heuristic for determining which parent provides the genetic material for crossover and how much material should be used in the crossover.

**Mutation**

Mutation will randomly alter one individual's genes to be within specified bounds. The mutation function used selects an amount of mutation from a probability distribution. This exploits the robust global search features of a typical genetic algorithm; as the number of generations reaches a maximum number of generations, the amount of change introduced

by the mutation can decrease so a search for a local minimum rather than the global minimum is performed.

### 2.3.7. Selection

Crossover and mutation result in offspring with variable viability; it is the job of the selection function to choose which offspring survive until the next generation. The selection function ranks the individuals in a population based on their fitness level, which is calculated by the cost function. It then decides which members survive until the next generation.

## 3. TESTS

### 3.1. Test Corpus

A major difficulty in this project was procuring an adequate corpus of data for testing and development. The problem is twofold: one issue is the lack of freely available multitrack recording sessions, and the other is exchanging data between proprietary recording software and MATLAB. A personal collection of recordings was primarily used for the project's development, and each session had to be prepared track-by-track into continuous WAV files so they could be read into MATLAB.

Obviously this field of research could benefit greatly from an open repository of recordings as well as cross-platform libraries for interpreting those files. Hopefully this dearth of data will change as more people become aware of this effort and collaborate on sharing recordings as well as developing interoperable file formats.

### 3.2. Methodology

A set of tests was created to determine the effect of number of tracks and total number of generations on the mixing algorithm's efficacy.

A number of tracks was randomly selected from a 10-track multitrack rock recording. Twenty random mixes containing four randomly selected tracks and twenty random mixes containing eight randomly selected tracks were produced. The gains used to generate each mix were recorded and stored as the target gains. Each of these mixes was then used as the target for the automatic mixing system. The same combination of four or eight tracks as the target was fed into the system and allowed to run for a number of generations.

## 4. RESULTS

As expected, adding more tracks to a mix degrades the performance of the mixing algorithm. Furthermore, increasing the maximum number of generations that the mixing algorithm is allowed to run increases its performance.

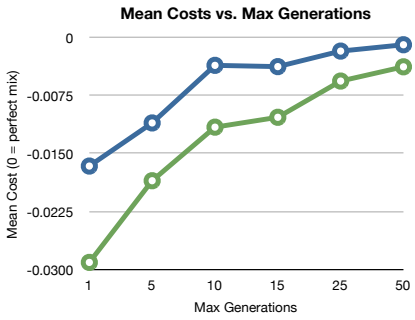


**Fig 2. Automatic mixing performance of 4 track (top) and 8 track (bottom) versus maximum generations**

As the number of generations increases, the disparity in performance between the smaller set of tracks (4) and the larger set of tracks (8) decreases. The performance of the GA levels off as the number of generations increases; at this point, the region of the global minimum in the search space has been found and the GA is exploring that area to find the optimal mix.

When run for fewer generations, the quality of the results from the automatic mixing varies widely. This can be attributed to the random initialization of the gains for each track; sometimes, the initial population generates an individual that is very close to the target, and other times it takes a number of generations to get close to the global minimum of the search space. A more refined initialization system could potentially help minimize this disparity.

The behavior of the mixing function as well as the narrowing down of the global search space to a localized search is apparent when watching the evolution of the mix as a bar graph, as you could watch the faders on an automated mixing console. The search starts out wide as a number of combinations are explored for a global minimum; then, as the maximum number of generations is reached, the search moves in smaller increments. The best-performing situation tested (4 tracks, 50 generations) comes quite close to hitting its target, indicated by the solid black lines in the figures.

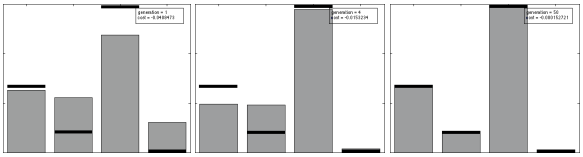

**Fig 3. Automatic mixing of 4 tracks. From left to right: 1 generation, 4 generations, and 50 generations**

Performance of the mixing algorithm with an eight track recording was not quite as good as with the four track recording, but it is still apparent that the mixes are approaching the target. As with the four track recording, the search for the eight track mix started off on a random course. By the end, it had recreated a reasonably close approximation to the actual mix, with one notable miss:

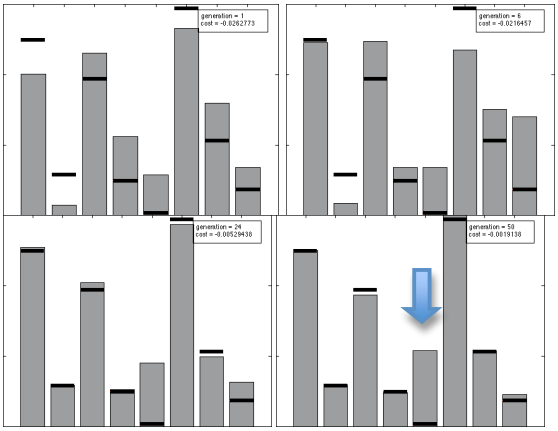

**Fig 3. Automatic mixing of 8 tracks. From top left to bottom right: 1 generation, 6 generations, 24 generations, 50 generations**

This was a fairly common occurrence when dealing with larger numbers of tracks-- the mixing algorithm appeared to 'lock on' to most but not all of the tracks and was unable to figure out others. Track 5, identified by the arrow in Figure 3, is a snare drum track. The target gain for this track was randomly generated to be .0067, or -43 dB. The final effect of the snare drum track at -43 dB on the target mix may have been too quiet to be perceptible by the cost function.

These results may also highlight a critical shortcoming in the approach of automatic mixing by evaluating the entire mix rather than on a track-by-track basis. The fact that track 5's gain is set much higher than the target value yet changing its value doesn't appear to

have much of an effect on the overall mix's cost could be because of the percussive nature of the track. The vast majority of the energy of the snare track comes from each hit, with the energy rapidly decaying after the hit. The hits occur relatively infrequently when compared to a more continuous sound such as the strumming of a guitar or a vocal melody. Properly handling a sparse track such as this in an automatic mixing task may require a track-by-track evaluation and initialization before the mixing takes place.

## 5. FUTURE WORK

The field of Music Information Retrieval has developed a number of techniques that could vastly improve the intuitiveness, efficiency, and utility of recording software. A framework for one application of this was presented here. This system for the automatic mixing of a multitrack recording combines a MIR-derived timbral classification system with a machine learning algorithm. The automatic mixing system has been shown to successfully recreate existing mixes under a number of situations with varying degrees of success; it remains to be seen what it will take to extend this system to creating novel mixes based on unknown targets.

This paper sought to shed light on the promising integration of MIR and the recording studio. It is the author's hope that this will lead to a more widespread effort towards achieving a 'smarter' recording studio. The development of an open file format for recording session exchange and libraries to support such a format across a number of platforms would be helpful to both the research community as well as software companies that create professional recording software. Furthermore, a collaborative repository of multitrack recording sessions would be of great use to all those who are pursuing this line of research.

## 6. RESOURCES

All code developed for this research thus far has been developed in MATLAB and is available to download on the NYU Music Technology Research website:

http://www.nyu.edu/projects/mtr/

The code used in this project requires both Elias Pampalk's MA Toolbox and Chris Houck's GAOT toolbox, both of which are freely available for download online.

MA Toolbox: http://www.pampalk.at/ma/

GAOT: http://www.ise.ncsu.edu/mirage/GAToolBox/gaot/

## 7. ACKNOWLEDGEMENTS

The author would like to thank Dr. Juan Pablo Bello for his immense help and guidance throughout this project. The author would also like to express his gratitude to Elias Pampalk and Chris Houck for making their code available and Ernest Li and Andy Saroff for their ideas and support. This work was supported by a Steinhardt Technology Award from New York University.

## 8. REFERENCES

[1] Pampalk, Elias et al. "On the Evaluation of Perceptual Similarity Measures for Music." Proceedings of the 6th International Conference on Digital Audio Effects (DAFx'03) 8 (2003).

[2] Pampalk, Elias. "A Matlab Toolbox to Compute Music Similarity from Audio." (2004).

[3] Houck, Christopher et al. "A Genetic Algorithm for Function Optimization: A Matlab Implementation." (1996).

[4] Horner, Andrew et al. "Machine Tongues XVI: Genetic Algorithms and Their Application to FM Matching Synthesis." Computer Music Journal 17.4 (2003): 17-29.

[5] Tan, B.G. and Lim, S.M. "Automated Parameter Optimization for Double Frequency Modulation Synthesis Using the Genetic Annealing Algorithm". Journal of the Audio Engineering Society 44.1/2 (1996): 3-15.