

Bach and Fourier: The Odd Couple?

By W.R. Winfrey and Russell P. Stewart

Mathematics Department, Concord University, Athens, West Virginia, USA

Abstract

Small recording studios face a variety of problems that could be addressed with the ability to alter the key of recorded music. Because clients of small studios are not typically professional performers, the music produced must often be re-recorded multiple times to correct bad notes, fix an out of tune performance, or simply to determine the correct key in which to record. This is a costly process in terms of both time and money. The research reported herein aims to develop software that can change the key of a piece or an individual note, allowing a competent engineer the means of turning an amateur performance into a professional product with minimal time and effort.

Current technologies exist to address these common problems, but the solutions are limited in strength and sonic clarity. Most of the current algorithms use time domain manipulation and overlapping windows to create an illusion of time-consistency, even though only short segments are being altered. The low-frequency oscillations introduced by these algorithms ultimately cause artefacts, echoes, and amplitude inconsistencies. These problems are magnified with larger shifts in key.

This paper reports progress in developing a new method to transform the key of music by direct manipulation of the amplitude and phase of the Discrete Fourier Transform that will change the note (including both the base harmonic and higher harmonics) to a different frequency while preserving the duration and structure of the note, thus eliminating the problems associated with currently available algorithms. The analysis is based on properties of the Classical Fourier Transform. The process uses the Discrete Fourier Transform to decouple the two components of the phase that control the time placement of the note and the structure of the note, transform the decoupled note in the frequency domain, and then recombine the components to obtain a new note with a new frequency.

1. Introduction

A piece of music is played in a certain key. That is, if the key frequency is f_0 each note has a base frequency f that is related to the key frequency by $f = \alpha^k f_0$ where k is an integer and $\alpha = 2^{1/12}$. The note also contains some number of harmonics, which occur at frequencies $2f, 3f, \dots, K = 2\alpha^k f_0, 3\alpha^k f_0, \dots, K$. Changing the key means that the notes are built upon new frequency f'_0 . The base frequency of the note is now $f' = \alpha^k f'_0$ and the harmonics are $2\alpha^k f'_0, 3\alpha^k f'_0, \dots, K$. The two key frequencies are related by $f'_0 = s f_0$ where s is typically a power of $\alpha = 2^{1/12}$. So the frequencies $\alpha^k f_0, 2\alpha^k f_0, 3\alpha^k f_0, \dots, K$ in the piece of music are replaced by the frequencies $s\alpha^k f_0, s2\alpha^k f_0, s3\alpha^k f_0, \dots, K$

It would seem initially that it would be simple to make the key change by using the classical Fourier Transform $F[f(t), \nu] = F(\nu) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi\nu t} dt$, $f(t) = \int_{-\infty}^{\infty} F(\nu) e^{i2\pi\nu t} d\nu$.

Specifically, if $f(t)$ is the piece of music, compute its Fourier Transform $F(\nu)$ and replace ν with $s\nu$ to get $F(s\nu)$ then invert the transform to get the piece of music in the new key.

However, if $f(t)$ and $F(\nu)$ are Fourier Transform pairs one can easily prove that $\frac{1}{s} f\left(\frac{t}{s}\right)$

and $F(s\nu)$ are Fourier Transform pairs. That is, if all the frequencies are doubled, the music plays in half the time it did before. So, it seems that the problem cannot be solved in the Fourier context. However, this is not the case.

2. Preliminaries

The Fourier Transform can be written as $F(\nu) = |F(\nu)| e^{i\phi(\nu)}$ where $|F(\nu)|$ is the amplitude and $\phi(\nu)$ is the phase. The amplitude tells how much of a particular frequency is present in the signal and the phase specifies how the signal is assembled from the individual frequencies, i.e. the phase controls the structure of the signal and the temporal placement of its parts. Most digital signal processing algorithms focus on phase-invariant methods because this assembly information is encoded in the phase in a complicated manner and it is best just to leave it alone. In the context of a piece of music, the phase controls the temporal placement of the note and how it is assembled from its component frequencies. Two simple models will show the fundamental method of key change and the critical observation that makes it work.

Simple Model #1. The first simple model is of a piece of music played by an idealized instrument. The notes played by the instrument will have base frequencies $f = \alpha^k f_0$ for some integer k and harmonics that are integer multiples of this base frequency. Assume each note that is played is the same mathematical function of the base frequency and has the same damping factors on the base frequency and each of its harmonics. Specifically, let $N(t, f) = N(t, \alpha^k f_0)$ be a note played at time $t = 0$ having a base frequency of $f = \alpha^k f_0$. Then all notes of this base frequency are temporal translations of this one. A note having a different base frequency $f^* = \alpha^l f_0$ played at time $t = 0$ is given by $N(t, f^*)$.

Since $N(t, f) \equiv 0$ for $t < 0$ and $N(t, f) \rightarrow 0$ rapidly as $t \rightarrow \infty$, the classical Fourier Transform $F[N(t, f)]$ exists.

In the context of this model, a tune has the notes $N(t, \alpha^k f_0)$, for a base frequency associated with a particular k played at the n_k times $t = t_{k,1}, t_{k,2}, \dots, t_{k,n_k}$. The part of the tune at this base frequency is $N(t - t_{k,1}, \alpha^k f_0) + N(t - t_{k,2}, \alpha^k f_0) + \dots + N(t - t_{k,n_k}, \alpha^k f_0)$. The entire tune is

$tune = \sum_k N(t - t_{k,1}, \alpha^k f_0) + N(t - t_{k,2}, \alpha^k f_0) + \dots + N(t - t_{k,n_k}, \alpha^k f_0)$ where the sum is over all base frequencies. By the basic properties of the Fourier Transform,

$$F[N(t - t_{k,l}, \alpha^k f_0)] = \exp(-i2\pi\nu t_{k,l}) F[N(t, \alpha^k f_0)]$$

$$\text{then } F[tune] = \sum_k \left(\left(\sum_{l=1}^{n_k} e^{-i2\pi\nu l_k} \right) F \left[N(t, \alpha^k f_0) \right] \right) = \sum_k \left(T_k(\nu) F \left[N(t, \alpha^k f_0) \right] \right).$$

The interpretation of this is the following. The term $T_k(\nu)$ controls the temporal placement of the simple notes and $F \left[N(t, \alpha^k f_0) \right]$ is the transform of an individual note. The phase of the note is split into two parts: the part that controls the temporal placement and the part that remains with the note and determines its structure.

Conceptually the process is the following. For each k divide the transform by $T_k(\nu)$ to get $F \left[N(t, \alpha^k f_0) \right]$, change f_0 , then multiply the result by $T_k(\nu)$. If the frequencies in $F \left[N(t, \alpha^k f_0) \right]$ are simply scaled by s , the reconstructed tune has the notes played at the proper times, but the length of the note is too long or short by a factor of $1/s$, so only half the problem is solved.

Simple Model #2. The second simple model suggests a solution to the problem of the note length. Consider the two idealized purely sinusoidal notes (no damping and no harmonics) of unit amplitude, one having a frequency of four Hz and one having a frequency of eight Hz. The duration of the notes is 0.25 seconds, followed by 0.75 seconds of silence.

Figure 1 shows the amplitude and phase of the Discrete Fourier Transform (DFT) of the two notes. Note that the amplitudes and phases of the two notes are similar. The points labelled with an "x" in the transform of the 4 Hz note correspond to those similarly labelled in the transform of the 8 Hz note in the sense that they occur at corresponding points on the amplitude curves. Comparing the phases at the corresponding points, we see that they are equal except at points where the amplitude is basically 0 and the phase does not matter. The critical observation is that the major peak and two lesser peaks to the right in the 8 Hz transform, amplitude with phase, can be obtained by translating the corresponding peaks, amplitude with phase, in the 4 Hz transform from 4 Hz to 8 Hz. Thus the outstanding problem in Simple Model #1, that of changing the key of a single note from f_0 to $f'_0 = s f_0$, can be solved approximately by finding the major peaks (and side lobes) in the DFT of the note that correspond to $\alpha^k f_0, 2\alpha^k f_0, 3\alpha^k f_0, \dots, K$ and shifting them to $s\alpha^k f_0, s2\alpha^k f_0, s3\alpha^k f_0, \dots, K$.

3. Implementation

In practice, the implementation is somewhat simpler than suggested in the discussion of Simple Model #1. Instead of explicitly computing the temporal placement portion of the phase, $T_k(\nu)$, it is sufficient to use a triangular window translated by half its width. Specifically, the signal is multiplied by the translated triangular window to extract a piece of the signal. Then the DFT of the extracted piece is computed. Next the major peaks with side lobes are extracted and shifted to new locations to obtain the DFT of the rekeyed piece of the signal. Then the inverse DFT is computed and added to the output signal.

4. Results

The algorithm was implemented in MATLAB and tested on a number of different pieces of music. The key often can be changed by as much as an octave in either direction (doubling or

halving the key frequency) without introducing any artefacts that are noticeable to the trained ear.

The method of key change seems to work for both monophonic and polyphonic sources. The musical content may be a single sound source (with harmonics of the root frequency) or several notes played at once. The algorithm sees polyphony as an extension of harmonics to a root.

5. Applications

Some possible applications of this algorithm include monophonic and polyphonic key adjusters, monophonic pitch correctors, and harmony processors. Pitch and key adjustment are simply direct applications of the algorithm. Monophonic pitch correction involves simple extension: identify the lowest major frequency peak in the Fourier spectrum and label it as the root, then identify where the root needs to move simply by comparing it to an agreed upon intonation map. Finally, choose the closest node and set the difference as the scale and implement the algorithm as above.

Harmony processing is another application of this method. Identify the root and chose a “harmony interval” such as a major third or a perfect fifth (in equally tempered tuning, these are the $1/3$ and $7/12$ powers of 2, respectively). Apply the algorithm to the frequency map. In the construction of the new data, instead of creating a new data set, simply add this shifted pitch to the original signal. Now there are two notes in the sound moment, with the new data being a harmony note of the original.

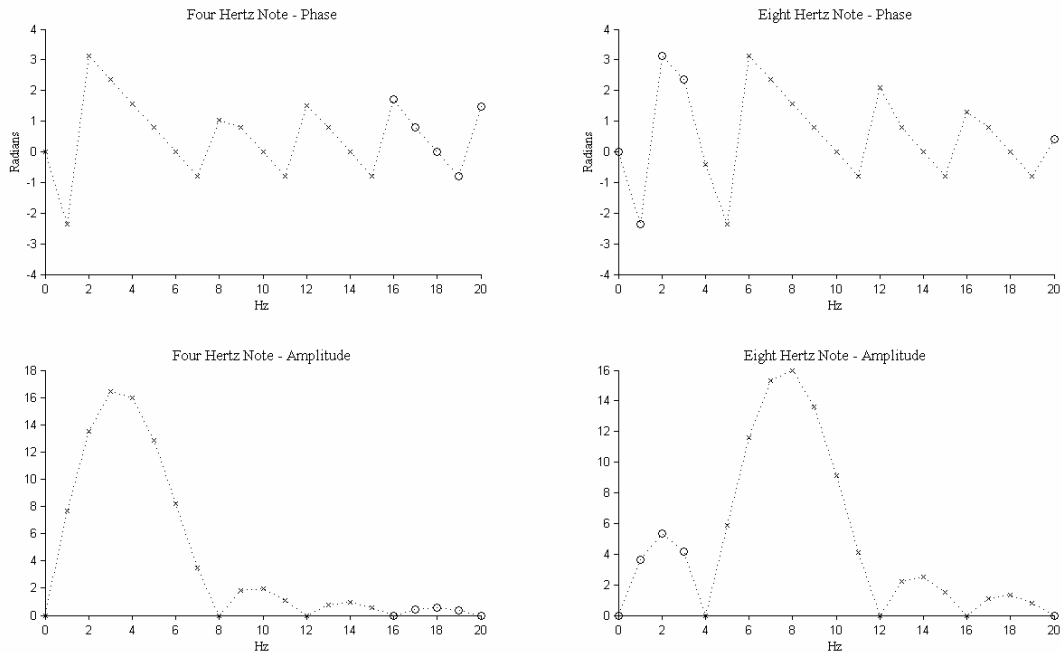


FIGURE 1. Details of parts of the Discrete Fourier Transforms of 4 Hz and 8 Hz notes