Real-time Complex Cepstral Signal Processing for Musical Applications

Ted Apel
tapel@ucsd.edu
Center for Research in Computing and the Arts
University of California, San Diego

November 30, 2001

Abstract

The use of cepstral methods for speech analysis is widespread. However, cepstral methods have had more limited use with musical sound transformations. This paper presents a method of complex cepstral processing of audio signals for real-time musical applications. Methods for the necessary real-time phase unwrapping are presented, and cepstral domain processing methods are discussed. A Max/MSP implementation is presented which uses a new external phaseunwrap~ to perform the phase unwrapping. Although enhanced phase unwrapping algorithms would improve the quality of the system, it was found that the complex cepstral domain processing methods produced promising sonic transformations.

1 Introduction

Cepstral analysis is based on the idea of analyzing frequency spectra for periodicities[1]. Cepstral methods are relevant to musical signal processing because the harmonic structures of musical sounds are condensed into peaks in the cepstral domain and the spectral envelope of a sound can oftentimes be separated from its corresponding excitation. Real-time cepstral analysis and reconstruction provides a method of sound processing based on these properties.

Bogert et al. introduced cepstral processing as a new method of analyzing seismic data by Fourier transformation of a signals log spectrum[11]. Oppenheim generalized this method into an analysis-reconstruction system called homomorphic processing. One method of homomorphic processing is complex cepstral analysis, which is a modification of the cepstral analysis method to allow for retransformation of the signal back to the original domain. As shown below, this can be completed by calculating the real and imaginary components of the signal at each stage of the cepstral analysis.

Cepstral processing is used widely in speech analysis systems to separate the formant and excitation parts of a speech signal[10]. These applications take advantage of cepstral processing as a feature detector to separate different aspects of a speech signal. Less work, however, has been done to take advantage of the reversibility of the complex cepstrum to manipulate signals in the cepstral domain.

Several researchers have explored real-time spectral domain transformations using the Max/MSP signal processing language[9][13][14][16]. As suggested by Pabon[8], cepstral features can be filtered, combined, and shifted before retransformation to the time domain. The goal of the present work is to extend the real-time spectral domain transformations into the cepstral domain.

This paper presents the discrete-time complex cepstrum, two methods of unwrapping phase for real-time audio, the author's real-time implementation of a complex cepstrum analysis and reconstruction system in the Max/MSP environment, and two methods of sound transformation in the cepstral domain.

2 Calculation of the Complex Cepstrum

The cepstrum is defined as the inverse fourier transform of the log magnitude spectrum of a signal[1]. The complex cepstrum generalizes this idea to use the complex logarithm to calculate the inverse fourier transform. The complex cepstrum is defined as

$$\hat{x}(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} \log\left[X(\omega)\right] e^{j\omega n} d\omega, \tag{1}$$

where $\hat{x}(n)$ is the complex cepstrum and $X(\omega)$ is the discrete-time Fourier transform defined as

$$X(\omega) = \sum_{n=-\infty}^{\infty} x(n) e^{-j\omega n}. \tag{2}$$

Therefore the complex cepstrum of discrete-time signals can be calculated from

$$\hat{x}(n) = \mathrm{IDTFT}\Big[\log\Big(\mathrm{DTFT}[x(n)]\Big)\Big], \tag{3}$$

where DTFT is the discrete-time fourier transform, and IDTFT is its inverse. When computing the complex cepstrum, the more efficient FFT is typically employed in place of the DFTFT. First the FFT of a signal is calculated, second the complex logarithm is calculated from the results of the FFT, and third the reverse FFT is calculated from the complex logarithm. This process begins by calculating the FFT [6] of a digital signal with the following formula.

$$\mathrm{FFT}[x(n)] = X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n) e^{-j\omega nk}, \quad 0 \geq k \geq N-1 \tag{4}$$

where $x(n)$ is a digital signal, $X(k)$ is the complex valued spectrum, N is the number of samples in the transformed signal, and $\omega = 2\pi/N$. Before the complex logarithm of $X(k)$ can be computed the magnitude spectrum and *unwrapped* phase spectrum must be calculated. The magnitude of $X(k)$ is calculated by

$$|X(k)| = \sqrt{a_k^2 + b_k^2}, \tag{5}$$

where $X(k) = (a_k + jb_k)$. The phase of $X(k)$ is calculated by

$$\mathrm{ARG}[X(k)] = \arctan\left(\frac{b_k}{a_k}\right), \tag{6}$$

where $\mathrm{ARG}[X(k)]$ are the principle values of the phase and are bounded by π and $-\pi$. This bounding however, produces discontinuities in the phase spectrum which are not suitable for complex logarithmic processing. The following section discusses unwrapping these phase values to produce a continuous phase spectrum. Using the definition of a complex logarithm we see that

$$\hat{X}(k) = \log\left[X(k)\right] = \left(\log|X(k)| + j\arg\left[X(k)\right]\right) \tag{7}$$

is the complex logarithm of the complex spectra. Finally, the complex cepstrum is computed by calculating the reverse FFT of $\hat{X}(k)$,

$$\hat{x}(n) = \sum_{k=0}^{N-1} \hat{X}(k) e^{j\omega kn}, \quad 0 \geq n \geq N-1. \tag{8}$$

Modifications of $\hat{x}(n)$ are known as complex cepstral processing. In Section 5 two types of complex cepstral processing are presented. After processing, the transformation back to the time domain signal is accomplished by reversing this process as seen in Section 4.

3 Phase Unwrapping

As mentioned in Section 2, phase values produced from the arctangent function are bounded by $-\pi$ and π. While these principal values of phase are sufficient for calculating the reverse Fourier transform, jumps of 2π radians will be present in the phase spectrum. Since the phase spectrum will be analyzed as a continuous function during cepstral transformation the phase spectrum must be *unwrapped* to produce a continuous function. This is done by adding or subtracting the appropriate multiple of 2π radians to the principal valued phase. So,

$$\arg\left[X(k)\right] = \mathrm{ARG}[X(k)] + 2\pi r(k), \tag{9}$$

where $\arg\left[X(k)\right]$ is the unwrapped phase, and $r(k)$ is the integer multiple of 2π radians to add to the principal phase value.

There are many nearly exact methods of computing the unwrapped phase such as factorization[15], and Strum sequences[5][4]. Unfortunately, the complexity of these algorithms are too intensive for real-time computation[18]. We must rely on more efficient computational methods which approximate the unwrapped phase from the principal phase values. Two candidate methods are examined here, and their real-time implementations are discussed in Section 4.

3.1 Schafer's algorithm

A simple method of approximating the unwrapped phase from principle phase angles is Schafer's algorithm[7]. This algorithm constructs the unwrapped phase by adding and subtracting 2π when the difference between adjacent phase spectrum values increases beyond a given threshold. Each $r(k)$ is calculated using the following algorithm:

$$\begin{aligned}
\text{If } \Big[\Delta(\mathrm{ARG}[X(k)]) > (2\pi - \epsilon)\Big] \quad \text{then}, r(k) &= \Big(r(k-1) - 1\Big) \\
\text{If } \Big[\Delta(\mathrm{ARG}[X(k)]) < -(2\pi - \epsilon)\Big] \quad \text{then}, r(k) &= \Big(r(k-1) + 1\Big) \\
\text{else}, r(k) &= r(k-1)
\end{aligned} \tag{10}$$

where $\Delta(\mathrm{ARG}[X(k)])$ is the principal valued phase difference,

$$\Delta(\mathrm{ARG}[X(k)]) = \left(\mathrm{ARG}[X(k)] - \mathrm{ARG}[X(k-1)]\right), \tag{11}$$

and ϵ is a variable threshold indicating the the amount of discontinuity required to increment or decrement $r(k)$. Schafer's algorithm produces accurate results when the frequency sampling rate is high enough that the phase difference is always less than the threshold value. This shortcoming of this algorithm is discussed further in Section 4.

3.2 Itoh's algorithm

Itoh's phase unwrapping algorithm was proposed for use with digital imaging[3][2]. Itoh's algorithm produces the exact unwrapped phase if the phase spectrum differences are less then π radians. Unlike other potential methods, Itoh's method is well suited to real-time implementation because it is not conditional or recursive. Tribolet's adaptive integration algorithm[17], for example, requires an iterative process that has a variable amount of computation time for each unwrapped phase estimate. Itoh's algorithm is a three step process requiring only the principle valued phase as an input. First, as in the Schafer algorithm, the principal valued phase differences $\Delta(\text{ARG}[X(k)])$ are computed from equation (11). Second, these changes in phase are wrapped between $-\pi$ and π:

$$\Phi\Big(\Delta(\text{ARG}[X(k)])\Big) = \arctan\left(\frac{\sin\Delta(\text{ARG}[X(k)])}{\cos\Delta(\text{ARG}[X(k)])}\right) \tag{12}$$

where Φ is the wrapping operator. And third, the wrapped phase differences are summed to estimate each unwrapped phase value:

$$\arg\left[X(k)\right] = \text{ARG}[X(0)] + \sum_{k=0}^{m-1} \Phi\Big(\Delta(\text{ARG}[X(k)])\Big). \tag{13}$$

This method produces unwrapped phase values that are accurate when the phase does not change more than 2π between phase bins. That is, if

$$-\pi \leq \Big(\Delta(\text{ARG}[X(k)])\Big) < \pi, \tag{14}$$

then the Itoh method produces the correct unwrapped phase. Thus, the Itoh method improves upon the Schafer method by eliminating the need to select a threshold value. Both the Schafer and Itoh algorithms are implemented in Section 4.

3.3 Removing Linear Phase Component

After implementing the Schafer and Itoh algorithms, it was found that they each produce a linear component to their unwrapped phase. This linear component produces a discontinuity at the 0 and N values of the phase spectrum. In order to make this unwrapped phase a differentiable even function, a method of ensuring that the function is differentiable at 0 and N was devised. Using the Itoh algorithm, phase was unwrapped for the phase spectrum from 0 through $(N-1)/2$ and the inverse was copied to $N/2$ through N to complete the phase spectrum. This method guarantees that the resultant unwrapped phase is a differentiable function at 0 and N.

4 Implementation

The Max/MSP real-time audio processing environment[19] facilitates constructing a platform for manipulating sound in the complex cepstral domain. The environment provides many time and frequency domain primitives and a method of adding primitives with the C language.

In order to produce a complex cepstrum analysis and reconstruction system, the author implemented both the Schafer and Itoh phase unwrapping algorithms in Max/MSP. The Schafer algorithm is implemented as one external object written in C called phaseunwrap_schafer~, whereas the Itoh algorithm consists of an abstraction composed of three objects: phasedifference~, phasewrap~, and phaseadd~. The objects phasedifference~ and phaseadd~ were implemented as externals while

phasewrap∼ is a Max/MSP primitive. The object phasedifference∼ calculates the principal valued phase differences from an input vector consisting of the wrapped phase spectrum, and phaseadd∼ sums the wrapped phase differences to produce the unwrapped phase. The three objects that compose phaseunwrap_itoh∼ are shown in Figure 1. With the phase unwrapping objects completed

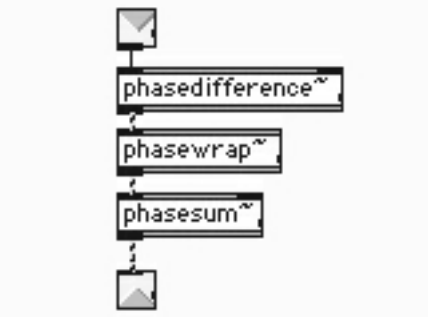

Figure 1: Three components of Itoh's phase unwrapping algorithm.

we can construct our complex cepstrum analysis system. Figure 2 shows a Max/MSP patch which computes the complex cepstrum using the phaseunwrap_itoh∼ abstraction, and Figure 3 shows the corresponding reconstructing system which converts the signal back into the time domain. Our system used an FFT size of 1024 bins and a hop overlap of 1/4. Using this framework, the quality

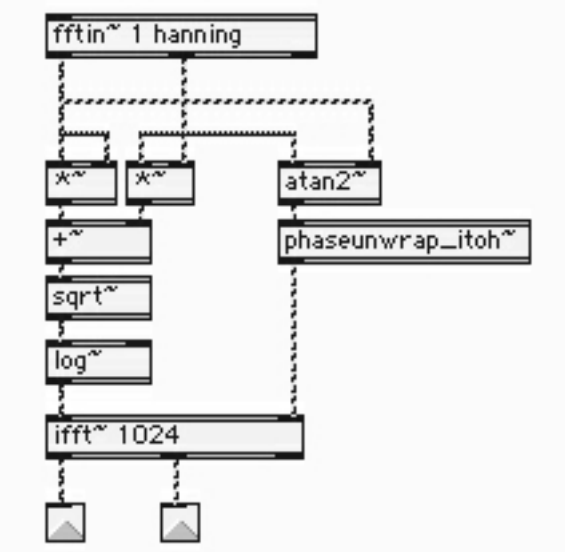

Figure 2: Max/MSP complex cepstrum analysis.

of the analysis and reconstruction system were sonically evaluated. It was found that neither the Schafer nor the Itoh algorithm was completely satisfactory at calculating the unwrapped phase because both algorithms produce results which are distorted variations of the original audio signal. It was found that the forced evenness in our implementation was a marked improvement over the unmodified Itoh algorithm or the Schafer algorithm. For this reason, all the sonic transformations in the following section were completed with the modified Itoh algorithm.

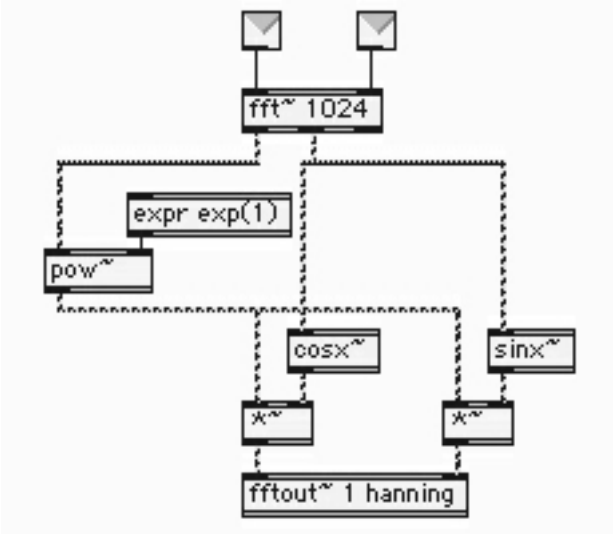

Figure 3: Max/MSP complex cepstrum reconstruction.

5 Complex Cepstral Processing

Two types of Max/MSP patches were created which process signals in the complex cepstral domain. Cepstral filtering or *liftering* patches and cepstral cross synthesis patches were created. Each will be discussed below.

5.1 Complex Cepstral Filtering

Following the naming convention of reversing the first letters of the corresponding time domain word, Bogert named filtering in the cepstral domain *liftering*. Several musical signals were high-pass, low-pass, and band-pass liftered. Low-pass liftering results in the high frequency periodicities being removed from the signals corresponding frequency spectrum. This technique is common in speech signal processing applications in order to isolate the formants of a speech sound. Our implementation proved successful in isolating the time varying spectral envelope of musical sounds. By changing the cutoff frequency (*quefrency*) location, real-time control over the smoothness of the spectral envelope was achieved. This technique extends to the complex cepstrum the results reported by Todoroff[16] in which the real cepstrum was low-pass liftered[16].

High-pass liftering results in the higher periodicities of a signal being isolated from its spectral envelope. Listening to the results of any high-pass liftering without further processing is unsatisfactory for two reasons. The removal of the spectral envelope results in a spectral *whitening* that lacks temporal development, and the limitations of our phase unwrapping algorithm are exemplified by high frequency gurgling associated with incorrect phase unwrapping.

Band-pass liftering of peaks in the cepstral domain produced particularly interesting results. It was found that by band-rejecting the highest peak in the cepstral domain, the fundamental frequency and its harmonics are removed from the resulting sound. This effect produced sounds which clearly displayed the temporal qualities, spectral envelope, and noise characteristics of the original musical instrument sample without its corresponding harmonic structure.

It was found that the real-time control of the lifter characteristics made it simple to experiment with various lifter settings and that both low-pass and band-reject filters produced interesting sonic

results.

5.2 Complex Cepstral Cross-Synthesis

A method of sonic cross-synthesis is made uncomplicated in the complex cepstral domain. The technique uses two sounds in the complex cepstral domain. One is low-pass liftered at a given cutoff quefrency while the other is high-pass liftered at the same cutoff quefrency. The two signals are then added in the cepstral domain. This results in the low frequency periodicities of one signal being combined with the high frequency periodicities of another. This can be thought of as the spectral envelope of one signal being combined with the excitation of another.

Several different musical samples were combined using this technique. It was found that the real-time control of the crossover quefrency allowed for precise selection of the best cross-synthesis effect. It was necessary to find the optimal crossover quefrency for the cross-synthesis to be effective. Unfortunately changing this parameter to achieve a time varying cross-synthesis was not feasible. However, the technique showed promising results using sounds that had distinct spectral envelopes and excitation characteristics.

6 Conclusions

The use of complex cepstral analysis and reconstruction systems for real-time transformation of musical sound was examined. Although the Itoh algorithm is more computationally intensive, the estimate of the unwrapped phase shows significantly better sonic results than the Schafer algorithm. However, it was found that the calculation of the unwrapped phase necessary for computation of the complex cepstrum needs more accuracy than the Schafer or Itoh algorithm provides. Furthermore, it was found that both complex cepstral filtering and cross-synthesis are promising techniques for the transformation of sound in the complex cepstral domain. Future research could focus on improving the phase unwrapping algorithm and experimenting with different types of complex cepstral processing.

7 Acknowledgements

The author would like to thank The Center for Research in Computing in the Arts (CRCA), Anthony Burr, F. Richard Moore, Janice Neri, Miller Puckette, John Puterbaugh, and Shahrokh Yadegari for their support of this work.

References

[1] Deller, J. R., J. H. L. Hansen, et al. (2000). Discrete-time processing of speech signals. New York, Institute of Electrical and Electronics Engineers.

[2] Ghiglia, D. C. and M. D. Pritt (1998). Two-dimensional phase unwrapping : theory, algorithms, and software. New York, Wiley.

[3] Itoh, K. (1982). "Analysis of the phase unwrapping algorithm." Applied Optics 21(14): 2470.

[4] Long, D. G. (1988). "Exact computation of the unwrapped phase of finite-length time series." IEEE Transactions on Acoustics, Speech and Signal Processing 36(11): 1787-90.

[5] McGowan, R. and R. Kuc (1982). "A direct relation between a signal time series and its unwrapped phase." IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-30(5): 719-26.

[6] Moore, F. R. (1990). Elements of computer music. Englewood Cliffs, N.J., Prentice Hall.

[7] Oppenheim, A. V. and R. W. Schafer (1989). Discrete-time signal processing. Englewood Cliffs, N.J., Prentice Hall.

[8] Pabon, P. (1994). Real-time spectrum/cepstrum games. International Computer Music Conference, Denmark, ICMA.

[9] Penrose, C. (2001). Frequency Shaping of Audio Signals. International Computer Music Conference, Havana, Cuba.

[10] Rabiner, L. R. and R. W. Schafer (1978). Digital processing of speech signals. Englewood Cliffs, N.J., Prentice-Hall.

[11] Rosenblatt, M., Brown University, et al. (1963). Proceedings. New York,, Wiley.

[12] Scott, J. B. (1984). "Improving confidence in the phase unwrapping algorithm." IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-32(6): 1254-5.

[13] Settle, Z. a. C. L. (1995). Real-time Musical Applications using Frequency Domain Signal Processing. IEEE ASSP Workshop Proceedings, Mohonk, New York.

[14] Settle, Z. a. C. L. (1998). Real-time Frequency-Domain Digital Signal Processing on the Desktop. International Computer Music Conference. Hong Kong, China.

[15] Steiglitz, K. and B. Dickinson (1982). "Phase unwrapping by factorization." IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-30(6): 984-91.

[16] Todoroff, T. (1996). A real-Time Analysis and Resynthesis Instrument for Transformation of Sounds in the Frequency Domain. International Computer Music Conference. Hong Kong, China.

[17] Tribolet, J. M. (1977). "A new phase unwrapping algorithm." IEEE Transactions on Acoustics, Speech and Signal Processing ASSP-25(2): 170-7.

[18] Zazula, D. and M. L. Gyergyek (1992). "Complexity in signal processing using cepstral approach." Elektrotehniski Vestnik 59(3-4): 165-70.

[19] Zicarelli, D. (1998). "An extensible real-time Signal Processing Environment for Max." International Computer Music Conference. Ann Arbor Michigan.