Dynamic Control of Frequency-domain-based Spectral Transformations

Introduction

While the use of the Fast Fourier Transform (FFT) for signal processing in music applications has been widespread, applications in real-time systems for dynamic spectral transformation has been quite limited. The limitations have been largely due to amount of computation required for the operations. With faster machines, and with suitable implementations for frequency-domain processing, real-time dynamic control of high-quality spectral processing can be accomplished with great efficiency and simple approach. This paper will focus on dynamic real-time control of frequency-domain-based signal processing, and will describe recent work (implementations) in this area. Since the implementation of the FFT/IFFT is central to the approach and methods discussed below, I will provide some background on the implementation and the development environment used in my work.

Frequency-domain signal processing operations and techniques

The standard operations which are used when processing audio signals in the frequency domain typically include: (1) windowing of the time-domain input signal, (2) transformation of the input signal into a frequency domain signal (spectrum) using the Fast Fourier Transform (FFT), (3) various frequency-domain operations such as complex multiplication for convolution, (4) transformation of the frequency-domain signals back into the time domain using the Inverse Fast Fourier Transform(IFFT), (5) and windowing of the time-domain output signal. This section of the paper will discuss some of the basic operations and techniques used in the applications discussed in this paper.

Development environment and Implementation

The development environment used by the author, Max / Max Signal Processing (MSP) [Zicarelli 1997], has evolved from the Max software, developed by Miller Puckette for the Ircam Signal Processing Workstation (ISPW) [Lindemann et al. 1991]. The environment provides for the development of real-time general purpose audio applications.

The FFT object provided in MSP is based on Miller Puckette's ISPW implementation [Puckette 1991] and stores time-domain signals as buffers of samples upon which the FFT analysis is done. For the purpose of discussion, the examples given in this paper make use of buffers of 1024 samples. Unlike time-domain signals, a frequency-domain signal is represented by a succession of spectral "frames". Like frames in a movie, the frames of FFT data represent a "snapshot" of a brief segment of an audio signal. A frame consists of a certain number of equally spaced frequency bands called "bins". The number of bins is equal to the size of the FFT buffer, thus the frames of FFT data have 1024 bins. Each bin describes the energy in a specific part of the audio signal's frequency range.

The FFT object outputs each frame, bin-by-bin, using three sample streams running at the sampling rate. Thus, each bin is represented by three samples consisting of "real" and "imaginary" values, and the bin number (index). At any given instant, each of the FFT's three signal outlets, shown below, produce a sample describing the *nth* bin of the current FFT frame. The IFFT is the complement of the FFT and expects, as input, real and imaginary values in the same format as FFT output.

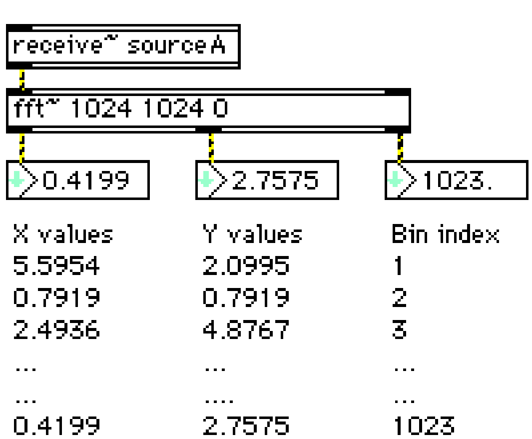


figure 1: sample-by-sample output of the FFT object

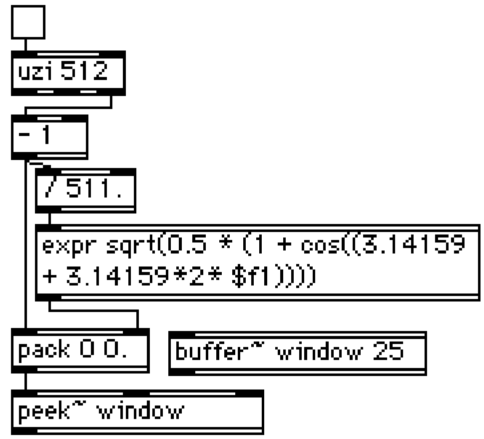


figure 2: windowing function generator

As seen in the figure above, the index values provides a synchronization signal, making it possible to identify bins within a frame, and recognize frame boundaries. The index values can be used to access bin-specific data for various operations, such as attenuation or spatialization, and to read lookup tables for windowing.

Windowing

It is necessary when modifying spectral data to apply an envelope (window) to the time-domain input/output of an FFT/IFFT pair, and to overlap multiple frames [Rabiner & Gold, 1975]. For simplicity's sake, the windowing operation shown below corresponds to a two-overlap implementation (two overlapping FFT/IFFT pairs); it is easily expanded for use in a four or eight-overlap implementation. Because of the flexibility of MAX and MSP, arbitrary windowing functions can be conveniently generated (see figure 2). In figure 3, note the use of the FFT frame index to read the lookup table-based windowing function in synchronization with the frame. The frame index is scaled between 0 and 1 in order to read the windowing function stored in an oscillator.

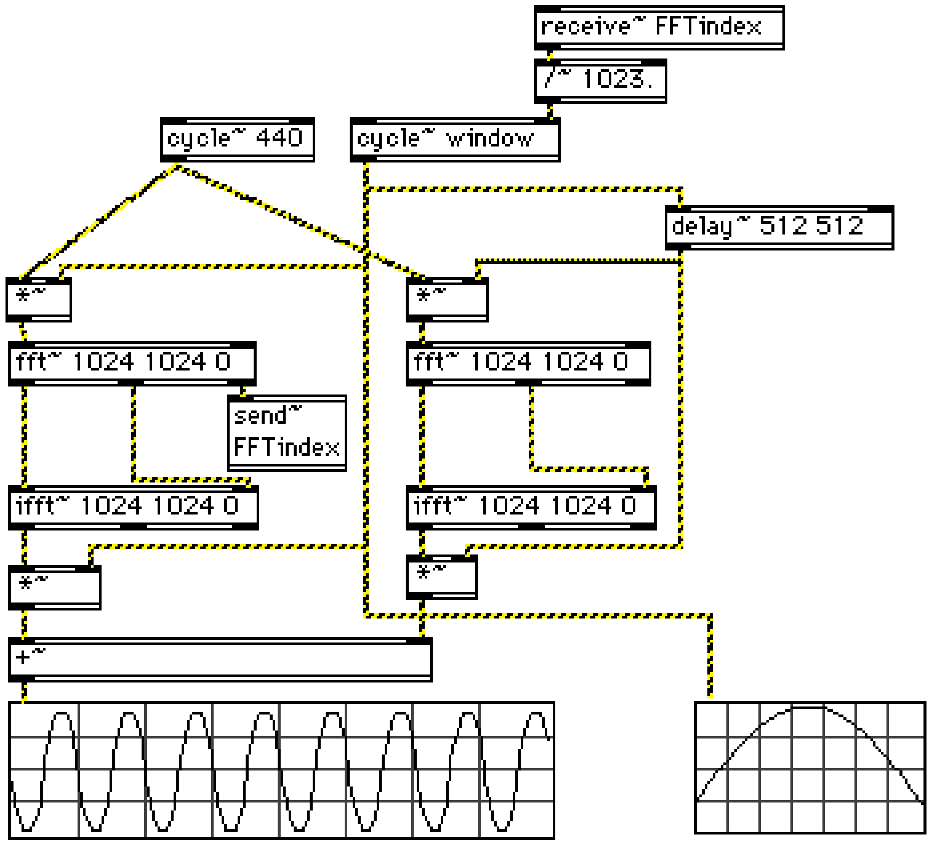


figure 3. typical two-overlap windowing of the input and output signals

Audio-rate control of FFT-based processing

The Max/MSP environment has two run-time schedulers: the Max "control" scheduler, which is timed on the basis of milliseconds, and the MSP "signal" scheduler, which is timed at the audio sampling rate [Puckette, 1991]. In FFT-based processing applications, where changes to the resulting spectrum are infrequent, MSP's control objects may be used to provide control parameters for the processing. This is both precise and economical, but limited in terms of spectral rate of change. For example, our implementation of FFT-based narrow-band "graphic" equalization uses a lookup table describing a filtering function, which is updated at the control rate; a given (frequency-domain) input signal is convolved with this function and spectral processing is accomplished. However, updating lookup tables at the control rate has bandwidth limitations. The rapidity with which a lookup table can be altered is limited by the bandwidth of the control system, giving the filtering certain static characteristics. Using 512 sliders to control individual FFT bins, drawing a filter shape for a lookup table with the mouse, or changing the lookup table data algorithmically provides only limited time-varying control of the filter shape. In addition, the amount of control data represented in a lookup table is large and cumbersome. Significant and continuous modification of a spectrum, as in the case of a sweeping band-pass filter, is not possible using MSP's control objects, since they can not keep up with the task of providing 1024 parameter changes at the FFT frame rate of 43 times a second (at the audio sampling rate of 44,100 samples per second). *see sound example 1*

Keeping in mind that the input signal to be modified is FFT data, a more dynamic approach to filtering is to update lookup tables containing a filter function at the signal rate (the audio sampling rate). The term "Spectral Processing Function" (SPF) will be used frequently in this text and refers to a lookup table-based function (actually a signal), whose length is that of the FFT. For each window of input signal (FFT data) we receive in real-time, we generate a corresponding SPF with which the input may be convolved. Dynamically, the SPF can describe a particular sequence of forms (or spectral envelopes), which determine the time-varying intensity of spectral processing by frequency component via convolution; a "form" describes the action of the spectral processor. Thus, our approach to dynamic processing spectra focuses on efficiently generating forms with a potentially high degree of detail or complexity, whose descriptions are simple (generated using few parameters, for low dimensional control) and intuitive [Settel,Lippe 1998].

In this paper, we will explore two methods for generating SPFs: the first involves the generation of forms using the spectral envelope of a signal (via FFT), while the second makes use of table-based waveform generation techniques. In each case, low-dimensional parametric control of complex forms is achieved. In both cases, a time-varying SPF is generated for, and convolved with, each window of input signal to perform filtering or band-limited panning, as shown below:

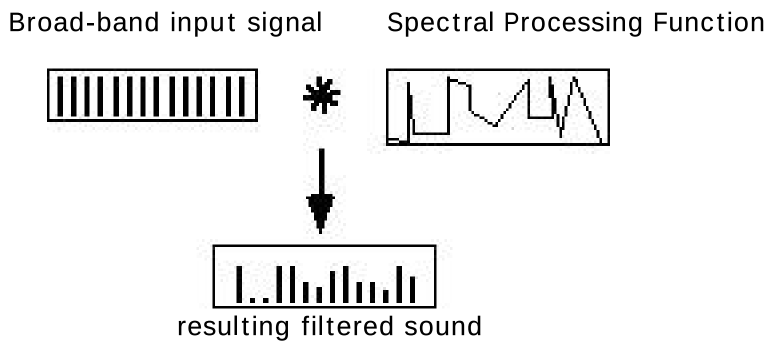


figure 4: a signal is filtered (via convolution) using a Spectral Processing Function (SPF)

Generating complex forms via Spectral Generation

When hunting around for a suitable method for generating complex "forms" (or spectral envelopes) using simple techniques, the classic FM pair algorithm (generating complex spectra using frequency modulation), with its simple structure and inherent potential for generating highly detailed and complex spectra, leaps to mind [Settel,Lippe 1994,1995]. The SPF is simply the spectral envelope of the signal generated via FM , it provides a rich source of possible forms, whose shapes and complexity are determined using the FM algorithm's few parameters (carrier FQ, carrier:modulator ratio, modulation index). The intuitive mapping of the FM parameter values to the resulting spectral form make this method extremely easy to use. In our implementation, a particular set of static or time-varying parameters is specified for the FM algorithm; the parameters determine the shape of the resulting SPF. With each window of input signal to be processed, the algorithm's output is transformed via an FFT into a SPF (spectral envelope), which is then applied to the corresponding window of input signal via convolution. Thus, the maximum rate of change for the SPF (signal processing parameters) is given by the frame rate (function of window length) of the FFTs; a 1024 point FFT, with a sampling rate of 44.1 khz, would translate to a spectral update rate of approximately 43hz.

Needless to say, other sources for generating complex spectra, such as amplitude modulation, additive synthesis, or waveshaping, may be used in the implementation above. However, the clear advantage of using FM lies in the simple control of the highly complex spectra it offers. For example:

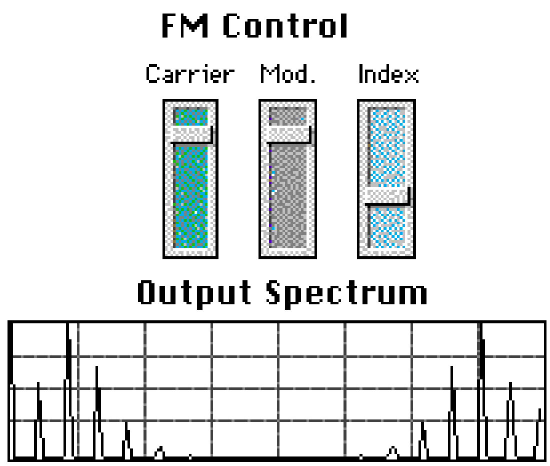
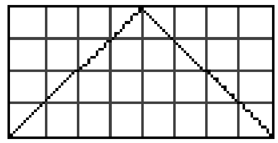


figure 5. a complex spectral form depending on only three parameters

Generating complex forms using waveform generators
The use of wave tables and basic table lookup operations provides a general and flexible approach to the well-known waveform generation and synthesis techniques we use to generate our Spectral Processing Functions (SPF). Techniques such as FM, AM, waveshaping, phase modulation, pulse-width modulation, all have the potential to provide complex, evolving waveforms, which can be used as SPFs to provide a high level of flexibility and detail for spectral processing techniques such as filtering or spectral panning. The use of table lookup operations such as inversion, scaling, offsetting, wrapping, and nonlinear distortion (waveshaping), provides powerful means for dynamically modifying these SPFs. For example, nonlinear indexing of lookup tables can be employed to provide dynamic control for a constant-Q bandpass filter, where control on a nonlinear frequency scale is required. Most important, the parameters of these waveform-based techniques and are few, familiar and easy to understand.

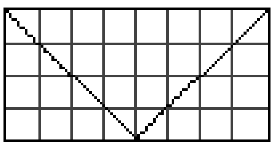
Let's now focus on the lookup table operations underlying these techniques for the generation of a Spectral Processing Function. As discussed above, the FFT used in our implementation provides us with a phasor that we use as an index for table lookup. Note that the index is mirrored around the sampling rate/2, following from the symmetrical (real) spectrum output by the FFT.



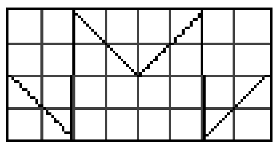
0hz sr/2 0hz

Phasor from the FFT's output

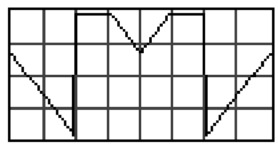
The following sequence of operations is performed on the phasor; note how its shape (form) is changed by each operation.



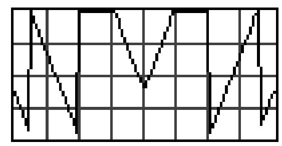
1. Inversion
-1



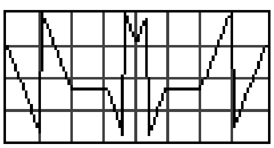
2. Phase Rotation
50%



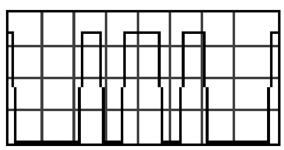
3. Scale Y (w/ clip)
1.36



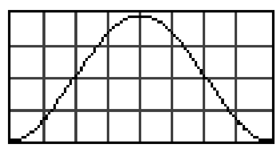
4. Period Multiply
2



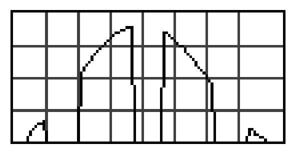
5. Phase Rotation
40%



6. Lookup Pulse wave
duty cycle = 50%



7. Amplitude Mod.



Resulting SPF

figure 6. lookup table operations

Note that steps 3, 6, and 7 entail loss of information; their placement in the sequence of operations can not be arbitrary.

While the above operations and parameter settings will produce a static SPF, it is important to note that the "phase rotation" operation also accepts a frequency parameter (LFO); a non-zero value will cause the phase rotation value to change constantly. Thus, the resulting SPF will vary periodically due to the constant change of its phase rotation parameter. This "LFO-based" technique provides control for periodic spectral modulation, and is useful in implementations such as swept band-pass filters (or spectral panners), comb filters, or "phase shifters".

To generate a more complex SPF, we start the same phasor above, then apply the following operations:

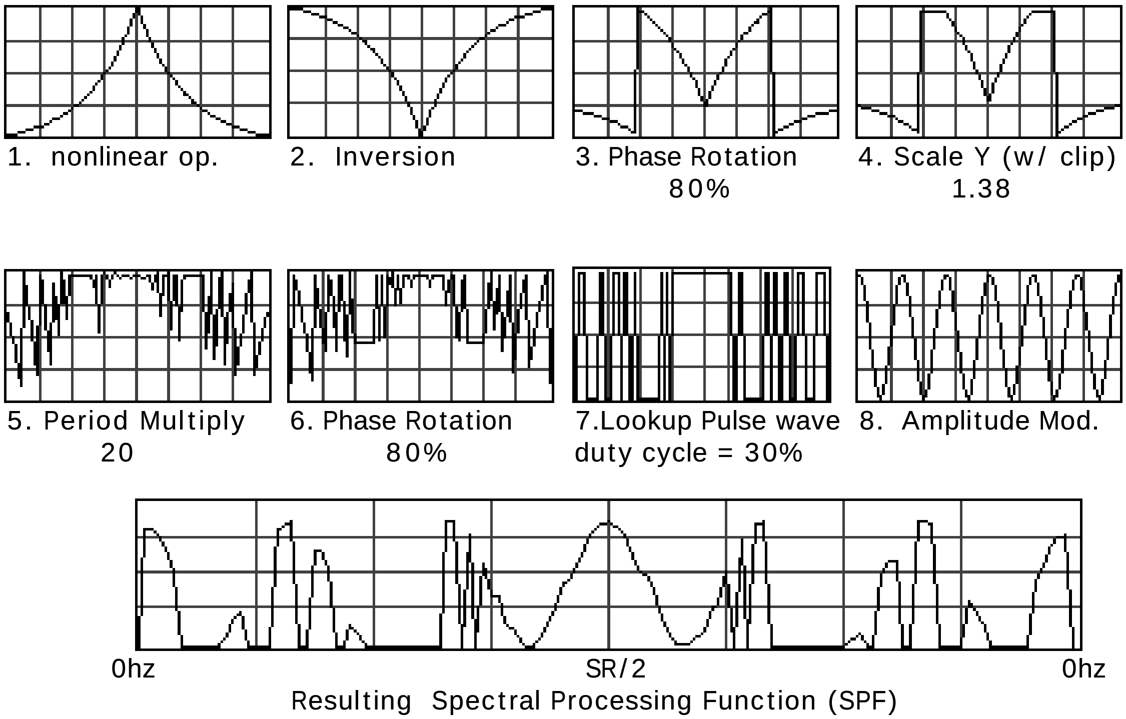


figure 7. generating a complex SPF

Applying the techniques

The above techniques for dynamic spectral processing lend particularly well to applications such as high-resolution dynamic filtering, and spectral panning. The author has implemented such applications, and has experimented with spectral resolution, making use of 1024 to 4096 point FFTs, twice or four times overlapped. Given the implicit tradeoff between time and frequency resolution, and considerations of computational cost, the use of two overlapped 1024 point FFTs is preferable for real-time performance situations when audio signal latency (delay from input to output) is an issue. However, the author found that a longer FFT of 4096 points is generally preferable, despite the relatively low spectral update rate of 10hz (10 FFT frames per second), since a long window allows for a (correspondingly long) highly detailed spectral processing function (SPF). For example, the "period multiply" operation shown above requires a long SPF when a greater number of periods is specified.

In the case of high resolution filtering applications, the SPF specifies the degree of attenuation to be applied to each component (band-limited region or bin) of the input spectrum (*see sound example 8).* The input signal's spectrum is convolved with the SPF and filtering is accomplished. The nature of the filter is determined by the form of the SPF. However, in the case of an application for spectral panning (band-limited panning), the SPF is used to specify the degree of phase rotation (via convolution with a complex sinusoid) to be applied to the components of the input spectrum. The operation results in a change to a given component's energy distribution in its real and imaginary parts. The real and imaginary outputs of the IFFT are mapped to a corresponding stereo output. Dynamic spectral panning occurs when the phase of the input signal's spectral components is rotated by a changing amount. *see sound example 9*

The SPF is applied in each application as shown below:

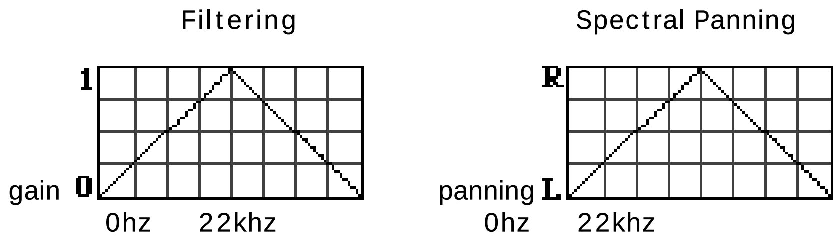


figure 8. applying SPFs for filtering and spectral panning

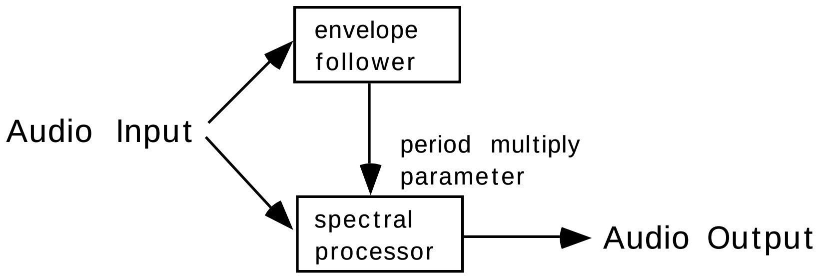
The filter above defines a lowpass. The spectral panning distributes (pans) the energy of the input signal's components continuously from left to right, based on the component's frequency; for example: components near 0hz are panned Left, components near 12khz are panned L and R, and components near 22khz are panned right.

Dynamic control of processing parameters via input signal analysis

Compressor/limiters are examples of signal processors that analyze the input signal to determine how it is in turn to be modified. A signal passing through such a processor can be thought of as a self-modifying signal. When an analysis stage (such as an envelope or pitch follower) is added to our spectral processing implementation, the resulting input analysis information may be used to dynamically control one or more parameters of the spectral processing. Since our interface for spectral processing requires only a small number of parameters, and since each parameter can significantly alter the shape of the SPF (offering a wide potential range of transformational possibilities), the mapping of few (even just one) input-derived control streams to processing parameters can provide a very high degree of signal self-modification.

Musically, the ability to control the degree and quality of a signal's spectral transformation via aspects of that signal's dynamics or inflection can be compelling; in live performance applications, this mapping of musical gesture to resulting timbre can be quite tangible and inspiring to the performer.

A particularly effective mapping of this sort can be produced using the Waveform Generation technique discussed earlier, where the control derived from the input intensity is mapped to the "Period Multiply" parameter of the processing shown below:



Which, as a function of input signal intensity, can produce a range of SPFs such as:

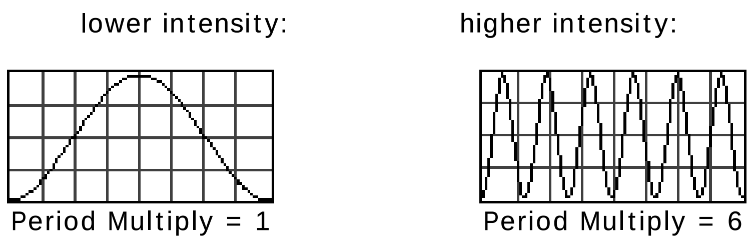
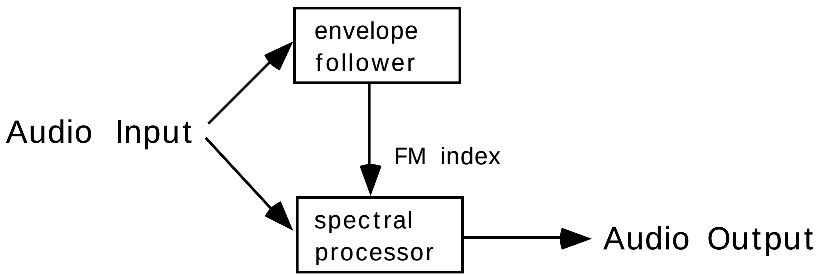


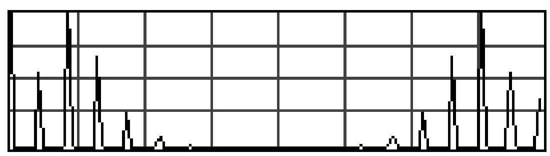
figure 9. self-modifying signal via the waveform generation technique

Thus, the more intense the input signal, the greater number of peaks and notches in the SPF. In a filtering application, the effect is that increasing input dynamic level translates to increasing separation (as opposed to fusion) of spectral components into discretely perceived sinusoids. In a spectral panning application, increasing dynamic level translates to increasing stereo separation of neighboring components, enhancing the perception of sound source location. In practice, the value of the period multiply parameter may reach 40 or 50, allowing for extremely fine spectral processing.

Alternatively, a similar result can be achieved using the Spectral Generation technique, discussed earlier in this paper. In this case, a Frequency Modulation pair is used to generate the SPF (see fig. 5 above), where the control derived from the input intensity is mapped to the "modulation index" parameter of the processing shown below.



The more intense the input signal, the greater the index of modulation, thus, the greater the number of peaks and notches in the SPF.



SPF generated from the spectrum of an FM pair with moderate index value

figure 10. self-modifying signal via the spectral generation technique

Needless to say, there are many other possible mappings of input features to spectral processing parameters. In each case, a given aspect of the input signal will potentially cause the spectrum of that same signal to be modified in a significant way. The challenge is, of course, to recognize the practical/musical sense of certain choices (mappings), which in turn, can serve a musical purpose in performance and/or composition. The approach of playing an instrument while processing its sound (as described above) has proven to be a very effective way to evaluate the mappings and discover their particular musical tendencies. As with dynamics processors, the choices of input signal, analysis mapping and processing technique produce results that cover a wide range of timbral possibilities. Future work will be concerned with discovering and establishing musically effective combinations of these choices.

Conclusion

The use of spectral generation and waveform-based techniques for low-dimensional audio-rate control of FFT-based processing has great potential: the parameters are few, familiar and easy to control, and direct mappings of real-time audio input from musicians to the control of FFT-based DSP is made possible. While the applications in this paper has been limited to filtering and spectral panning, audio-rate control of FFT-based processing applies equally well to any FFT-based applications where a high degree of processing control is required at the frame rate. The author has also implemented the above mentioned techniques in applications for denoising and dynamics processing. I believe that these techniques hold great promise for control-intensive FFT-based applications, especially for live-performance DSP, where a new level of interactivity for a relatively new set of frequency-domain DSP techniques, is now possible.

Acknowledgments
The author would like to thank Cort Lippe, Miller Puckette, David Zicarelli and Philippe Depalle for their input towards the work in this paper.

References

Lindemann, E., Starkier, M., and Dechelle, F. 1991. "The Architecture of the IRCAM Music Workstation." Computer Music Journal 15(3), pp. 41-49.

Puckette, M., 1991. "Combining Event and Signal Processing in the Max Graphical Programming Environment." Computer Music Journal 15(3), pp. 68-77.

Puckette, M., 1991. "FTS: A Real-time Monitor for Multiprocessor Music Synthesis." Music Conference. San Francisco: Computer Music Association, pp. 420-429.

Rabiner and Gold, 1975, "Theory and Application of Digital Signal Processing ", Prentice-Hall

Settel, Z., and Lippe, C., 1994. "Real-time Timbral Transformation: FFT-based Resynthesis", Contemporary Music Review Vol. 10, United Kingdom, Harwood Academic Publishers, pp. 171-179.

Settel, Z., and Lippe, C., 1995. "Real-time Musical Applications using Frequency-domain Signal Processing" IEEE ASSP Workshop Proceedings, Mohonk New York.

Settel, Z., and Lippe, C., 1998. "Real-time Frequency-Domain Digital Signal Processing on the Desktop" International Computer Music Conference Proceedings, Ann Arbor, MI.

Zicarelli D., 1997. "Cycling74 Website", www.cycling74.com.

Sound Examples Catalogue

Dec. 1998

The following examples are provided on the CD accompanying the paper. Note that unless stated otherwise, all the examples were produced via processing implementations using two overlapped 4096 point FFT/IFFTs.

Track — Description

1). 59" Static Graphic EQ implementation. Modifications to the spectrum is made using Max control objects updated by mouse events. The rate of spectral change is relatively limited.

2). 15" Source sound A for cross synthesis: Spoken text

3). 12" Source sound B for cross synthesis: Clarinet sound

4). 57" Static Graphic EQ implementation. Same configuration as in example 1, but using different source material.

5). 22" Cross Synthesis of amp spectrum of Source sound A with Source sound B (see examples 3, 4).

6). 1'02" Cross Synthesis of amplitude spectrum of Source sound A with Source sound B (see examples 3, 4). The degree of spectral intersection is boosted using spectral smoothing via a lowpass on spectrum.

7). 1'02" Cross Synthesis of amp spectrum of Source sound A with Source sound B (see examples 3, 4). The degree of spectral intersection is increased using a technique which forces the energy of all components in Source sound B to approach 1 (unity). There are two parts to this example. During each, the degree of intersection is increased gradually. The second part covers a more extreme range than the first.

8.) 1'23" Dynamic control of spectral filtering via Waveform Generation Techniques. One or two control parameters are varied manually over time. Towards the end of the example, the original (unfiltered sound is presented).

9) 1'22" Dynamic control of spectral panning via Waveform Generation Techniques. One or two control parameters are varied manually over time. Towards the end of the example, the original (unfiltered sound is presented). Headphones recommended.

10). 30" Dynamic control of spectral filtering via Waveform Generation Techniques. A LFO is used to constantly vary the spectral processing. The saxophone sound in sound example 11 is the source. Note two overlapped 1024 point FFT/IFFTs are used in this example.

11). 25" Original (unfiltered) saxophone recording.

12). 36" Dynamic control of spectral filtering via Waveform Generation Techniques. The saxophone sound in sound example 11 is the source. This example includes dynamic control of processing parameters via envelope follower analysis of the saxophone (input) sound. Note two overlapped 1024 point FFT/IFFTs are used in this example.. Also note that reverb has been added to this example.

13). 1'07" Dynamic control of filtering and spectral panning via Waveform Generation Techniques. One or two control parameters are varied manually over time. Name the source.

14). 9) 23" Dynamic control of filtering and spectral panning via Waveform Generation Techniques. Control parameters are varied via LFO. Towards the end of the example, the same original is used here as in example 9.