Spectral envelope extraction by means of cepstrum analysis and filtering in Pure Data

José Henrique Padovani
Unicamp (Arts Institute), Campinas, Brazil
tel.: +55 31 92166546
zepadovani@gmail.com

ABSTRACT

This paper introduces a Pure Data abstraction that implements the method of *cepstral* deconvolution by means of filtering the low *quefrencies* of an audio signal. Two possible applications are described: 1) cross convolution between different audio signals and 2) spectral peak detection for formant estimation (this second application is currently being developed).

Keywords

Cepstrum, deconvolution, FFT, Pure Data.

1.AUDIO AS CONTROL DATA

One of the great improvements that interactive musical systems based on environments like Pd and Max have achieved in the last fifteen years is the possibility to operate real-time audio computations in personal computers [1]**.** In these environments, the computations are usually used to process or synthesize audio and/or video to obtain new sounds and images in the end of the DSP process.

Nevertheless, these fast computations are also useful tools to extract information from the great amount of data of audio signals and use them as parameters or variables to control other processes. In the Pd environment, there are some objects – like [fiddle~], [bonk~] and [sigmund~] – that reduce the amount of data received in a high audio rate, retrieving more selective information about the signal in a control rate [2].

This paper describes an abstraction in Pd that reduces the spectral information of a signal by means of its deconvolution through *cepstrum* analysis and filtering. After achieving the spectral envelope of the signal (by extracting the low *quefrencies* of its *cepstrum*) it is possible to apply the signal's frequency response in cross-convolutions with other signals or to analyze the envelope to find its peaks – which can provide important data about the most relevant formants.

The Pd abstraction described here is a translation and a development of a previous method implemented in Max/MSP that was not used to extract the peaks of the analyzed signal [3], but to synthesize jaw harp sounds through the cross convolution of vowel and string sounds generated through the Karplus-Strong method.

2.CEPSTRUM: A SHORT DESCRIPTION

The technique of *cepstrum analysis* (also called *cepstrum alanysis*) was introduced for the first time by Bogert, Healy and Tukey, in 1963 [4], and was strongly related to researches that had as its main objective to discriminate between earthquakes and nuclear explosions in seismologic signals [5]. In 1964, M. Noll publishes a work describing the application of *cepstrum* analysis to detect pitch and presence/absence of speech components in audio signals [6].

Tempelaars describes the *cepstrum analysis* as a "technique to deriving information about the input signal and/or the system from the output signal only"[7]. Having speech sounds, for example, we can presume that they were generated through the convolution of $x(t)$, "the excitation signal of the vocal chords" and $H(f)$, the "frequency response of the vocal tract conceived as a linear filter"[8].

Considering a single frame of the spectrum of an audio signal as a waveform, it is possible to analyze its periodicities in the same way that it is done when one applies Fourier transforms to time-domain signals. In this way, it is possible to determine the periodical components of the spectrum that are related to its smooth shape (frequency response) and those that are related to the excitation signal that is filtered by this spectral filter. The *cepstrum analysis* is capable to compute this by applying a second Fast Fourier Transform (FFT) over the logarithm of the band magnitudes of a FFT of the time-domain signal.

As stated by Tempelaars [9] and Brilliger [10], Tukey *et al*. have created a number of neologisms to distinguish between the numerical values and analysis elements of the second Fourier transform and those related to the first Fourier transform. As a result, from terms like *spectrum*, *analysis*, *magnitude*, *phase*, *filtering* and *frequency*, were derived the anagrams *cepstrum*, *alanysis*, *gamnitude, saphe, liftering* and *quefrency*, all them related to the FFT of the log of the FFT of the time-domain signal (in our case, the audio input).

Regarding the speech signal example, the "slow" *quefrencies* analyzed by this second FFT are the ones that describe the smooth shape of the spectral envelope while the "fast" ones describe the excitation signal that is convolved by the frequency response of the vocal tract.

PdCon09, July 19-26, 2009, São Paulo, SP, Brazil.

Thus, while in time domain the signal $y(t)$ can be described as the result of a convolution (*) of two signals

$$y(t) = x(t) * h(t) \tag{1}$$

and in the frequency domain the same can be described as a multiplication of two signals

$$Y(f) = X(f) \times H(f) \tag{2}$$

in the *quefrencies* domain of the *cepstrum* we *alanyze* the logarithm of the FFT bins' magnitudes that are related to the signal frequencies, so that it becomes possible to subtract one of the terms of the addition and obtain only the frequency response signal:

$$\log H(f) = \log Y(f) - \log X(f) \tag{3}$$

The whole process and the main objects and abstractions used to implement the *cepstrum* analysis and filtering in Pure Data to retrieve the spectral envelope of an audio signal are described in Fig.1.

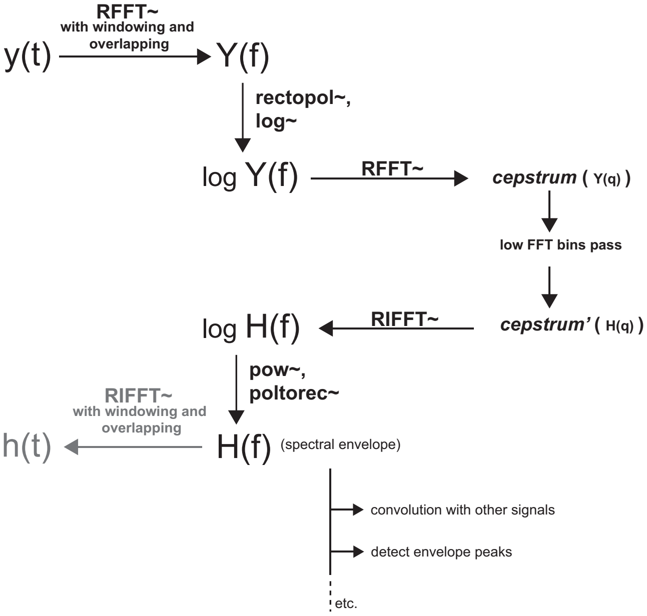


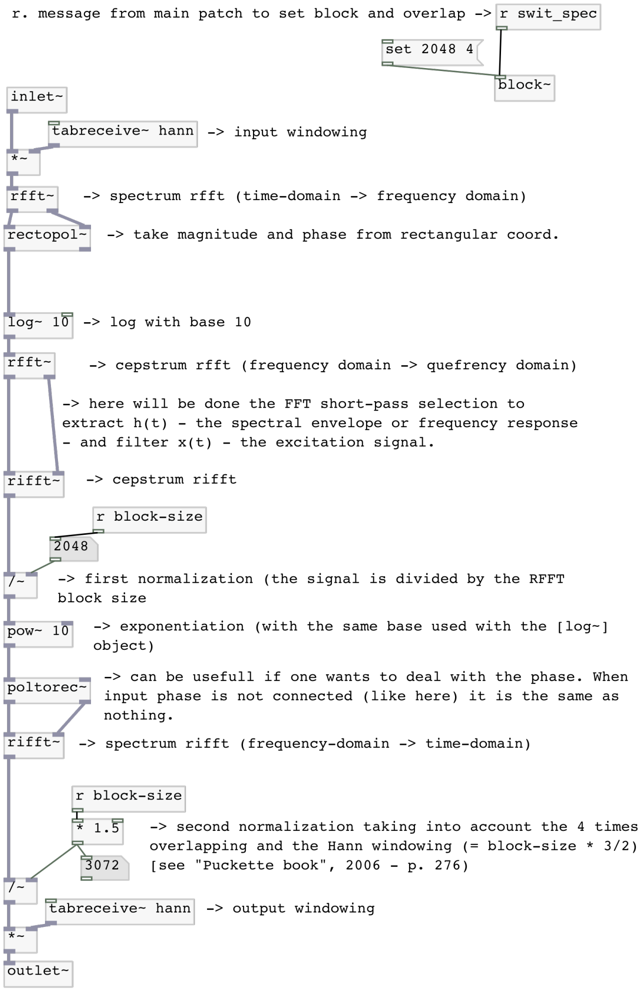
Figure 1 - *Cepstrum* analysis and filtering to extract the spectral envelope of an audio signal.

3.PURE DATA ABSTRACTIONS

To write the Pd abstraction of the *cepstrum* process, the first task was to create a standard sub-patch with windowing, overlapping and a bigger block-size to compute the real Fast Fourier Transform of the input audio (Fig. 2). [11]

To make the conversions between rectangular and polar coordinates from the [RFFT~] and the [RIFFT~] objects, two abstractions were created: [rectopol~] and [poltorec~] (although one could simply connect the outlet of [pow~] to the first inlet of [rifft~], once the phase is ignored in the present computation method). To take the logarithm and the exponentiation of the frequency domain signal, the objects [log~] and [pow~] were used (both objects are included in the cyclone library). One could also use the external [expr~] to compute the same operations.

With this sub-patch it possible to test if the normalization, the overlapping, the Hann window and the second-level Fourier Transform are working, although the process makes no substantial change in the analyzed signal.



```
r. message from main patch to set block and overlap -> [r swit_spec]
[set 2048 4(
[block~]
[inlet~]
[tabreceive~ hann] -> input windowing
[*~]
[rfft~]   -> spectrum rfft (time-domain -> frequency domain)
[rectopol~]  -> take magnitude and phase from rectangular coord.
[log~ 10] -> log with base 10
[rfft~]    -> cepstrum rfft (frequency domain -> quefrency domain)
        -> here will be done the FFT short-pass selection to
        extract h(t) - the spectral envelope or frequency response
        - and filter x(t) - the excitation signal.
[rifft~]   -> cepstrum rifft
[r block-size]
[2048]
[/~]    -> first normalization (the signal is divided by the RFFT
        block size
[pow~ 10]  -> exponentiation (with the same base used with the [log~]
           object)
[poltorec~]  -> can be usefull if one wants to deal with the phase. When
             input phase is not connected (like here) it is the same as
             nothing.
[rifft~]  -> spectrum rifft (frequency-domain -> time-domain)
[r block-size]
[* 1.5]    -> second normalization taking into account the 4 times
           overlapping and the Hann windowing (= block-size * 3/2)
[3072]     [see "Puckette book", 2006 - p. 276)
[/~]
[tabreceive~ hann] -> output windowing
[*~]
[outlet~]
```

Figure 2 - The basic abstraction structure to compute the *cepstrum* of an audio signal. No transformation or filtering is done here in the *quefrency* domain.

The second step in the *cepstrum* abstraction is the short-pass filter. To synchronize the process that receives the bin number related to the cut *quefrency* value*,* it is necessary to create a conditional rule – constructed with Yadegari's external [fexpr~][12] and with a [phasor~] object –, that returns a ramp between 0 and N–1 bins in each [block~] cycle. (Fig.3)

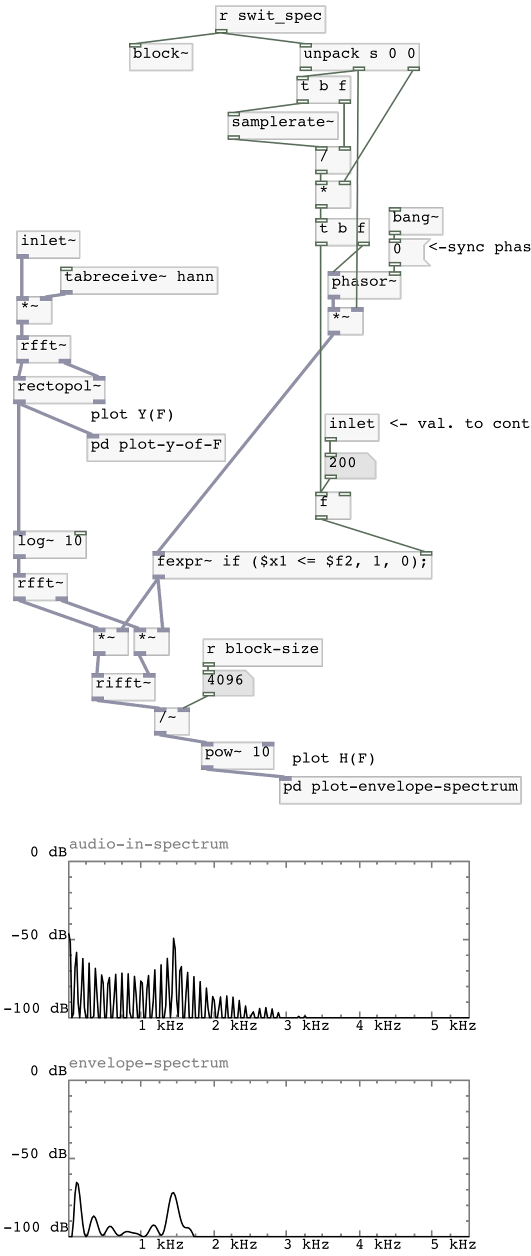


Figure 3 - The sub-patch [pd cepstrum], extracting the spectral envelope (array "envelope-spectrum") of the filtered signal (see Fig.4).

The frequency of the [phasor~] object is set to be equal to the sample rate divided by the block-size and multiplied by the overlap factor. The phase is adjusted to 0 in every DSP cycle (by using a [bang~] object). Finally, the [phasor~] signal is multiplied by the block-size, allowing the comparison between the control number (i.e. the cut bin value) and the current bin of *cepstrum* analysis.

In Fig. 3 and 4 we have, in that order, the sub-patch [pd cepstrum] and the band-pass filtered signal used to test the Pd implementation of "*cepstrum liftering*".

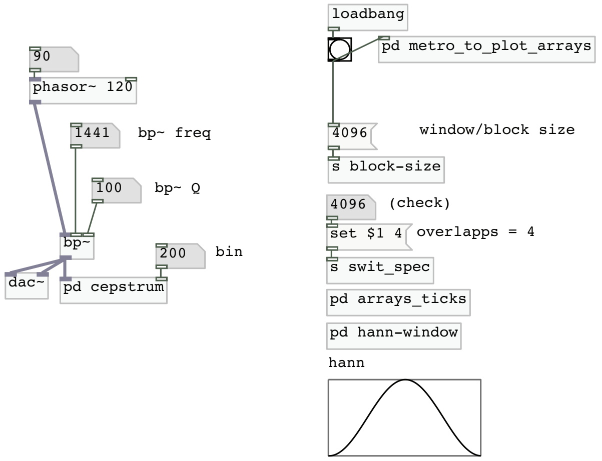


Figure 4 - The main patch, sending the band-pass filtered [phasor~] signal to test the cepstrum process.

4.CROSS-CONVOLUTION WITH THE SPECTRAL ENVELOPE

Perhaps the most practical application of extracting a spectral envelope with the *cepstrum* analysis is to use its shape to filter a second signal through spectral convolution. After the exponentiation that brings the envelope signal back to the frequency domain, its value can be used to multiply the real and imaginary pairs obtained by applying the FFT analysis to a second signal.

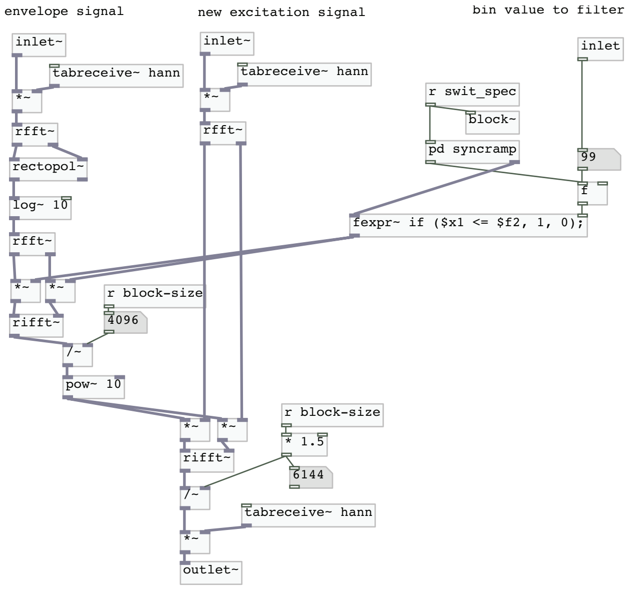


Figure 5 - Abstraction illustrating the use of the extracted envelope to convolve a new "excitation signal".

In a real-time DSP context, this process can be used to create numerous sound effects. A good example is the convolution between the frequency response of a speech signal and a white noise. By changing the value that controls the short-pass filter (i.e., the maximum bin allowed to pass the *cepstrum* filtering), it is possible to produce a very whispered voice, discarding almost totally the periodic components of the original signal.

In any case, it is important to have a second signal with some temporal homogeneity in spectral distribution otherwise the spectral envelope will be silenced when multiplied by very low magnitude values. Using reverbs or adding low amplitude noise to the signal to be filtered may be satisfactory strategies in such cases. The same can be said about the spectral envelope, which can be smoothed to have a more gradual change over time with the use of delays, with objects like [max~] or by arithmetical operations (to take the mean value of consecutive frames, for example).

5.UNDER DEVELOPMENT: ENVELOPE PEAK DETECTION FOR FORMANT DETERMINATION

The extraction of the frequency response of a signal can provide important data about it. An interesting possibility would be to create a DSP system in Pd capable to detect the main formants of speech or instrumental sounds, providing the approximate values for frequency, amplitude and bandwidth of these formants.

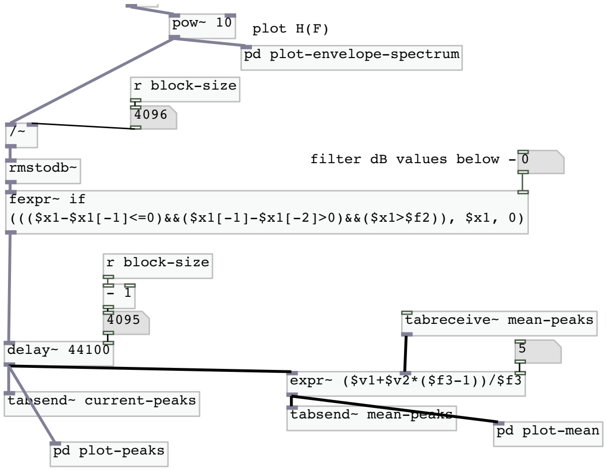


Figure 6 - Envelope peak detection mechanism.

My first attempt to do something similar to this was to add a peak detector to the *cepstrum* patch, filtering the peak values with very low dB amplitude. The idea was to use an array to store the mean value of the current bins of the detected peaks summed with the past values of the same array, creating thus a smoother changing of the peaks' values.

Nevertheless, the current peak detection abstraction needs be improved so that the peaks can be better detected and other important data can be extracted from the spectral envelope (such as the formant bandwidth, for example).

As stated by Tempelaars, [13] to find the spectral envelope peaks is a good start to formant determination, but does not fulfill the requirements of such an analysis. In fact, the *cepstrum* analysis can retrieve peaks that are very close to each other, falling in the same formant region of the spectrum. Another important fact is that the bin filtering cut value must be well tuned so that there are not too many or too few details in the spectral shape. Indeed, if the cut value is too high, a differentiation analysis will find too many peaks and will not be able to determine which are the most important ones. On the other hand, if it is set too low, the peaks' frequencies will be deviated from its actual values once the envelope shape will have been constructed by very few cepstral components.

6.REFERENCES

[1] Puckette, M. Max at seventeen. *Computer Music Journal 26/4* (2002), 31-43.

[2] Puckette, M.; Apel, T. Real-time audio analysis tools for Pd and MSP. *Proceedings, International Computer Music Conference*. (San Francisco: International Computer Music Association, 1998), 109-112.

[3] Padovani, J. H.; Freire, S. Explorando envoltórias espectrais em sistemas musicais interativos. *Anais do XVI Congresso da Associação Nacional de Pesquisa e Pós-Graduação em Música.* (Brasília: ANPPOM, 2006). (http://www.padovani.googlepages.com/explorandoenvoltorias-anppom2006.pdf)

[4] Bogert, B.; Healy, M.; Tukey, J. W. (1963). The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum, and saphe-cracking. In *Proc. Symposium on Time Series Analysis (M. Rosenblatt, ed.)* 209--243. Wiley, New York. [Reprinted version in The *Collected Works of John W. Tukey I. Time Series: 1949--1964* (D. Brillinger, ed.) 455--493. Wadsworth, Monterey, CA (1984).]

[5] Brillinger, D. John W. Tukey's Work on Time Series and Spectrum Analysis. *The Annals of Statistics*. 30/2002, 1595-1618.

[6] Noll, M. Short-time Spectrum and 'Cepstrum' Techniques for Vocal-Pitch Detection. *The Journal of American Society of Acoustic. V.36 n.2*, 1964, 296-302.

[7] Tempelaars, S. Signal processing, speech, and music. Lisse: Swets & Zeitlinger. 1996. pp. 267.

[8] *idem.*

[9] *ibidem*. pp. 268.

[10] Brillinger, op. cit., pp. 1612.

[11] Puckette, M. *The Theory and Technique of Electronic Music*, 2006 (http://crca.ucsd.edu/~msp/techniques.htm, see section 9.4)

[12] http://www.crca.ucsd.edu/~yadegari/expr.html

[13] Tempelaars, *op. cit.*, pp. 271.