EXPOCHIRP TOOLBOX: A PURE DATA IMPLEMENTATION OF EXPONENTIAL SWEEP SINE (ESS) IMPULSE RESPONSE MEASUREMENT

S Di Rosario Buro Happold Ltd, London, UK
K Vetter Independent PD programmer, Rotterdam, Netherlands

1 INTRODUCTION

The use of Exponential Sweep Sine techniques to measure impulse responses (IR) in enclosures or of sound reproduction systems has been investigated thoroughly in recent years and many commercially available software packages have been developed. The authors have found themselves performing Impulse Response measurements that needed an in-depth knowledge of the measurement process , hence the decision to program our own software that allows control over each step of the IR measurement.

Pure Data (PD) (an open source programming platform) has been adopted as our software development tool due to its modular work-flow and ease of use.

The ExpoChirp project aims to be a cross platform open source implementation of up to date IR measurement and analysis techniques. ExpoChirp modules show graphical data in all stages of the measurement procedures and this enables users to assess the working, specifications and quality of the implemented routines, which is useful for education purposes and for special cases in real life IR measurements, as already described.

The ExpoChirp project was initiated in 2010 at the Pure Data forum[1].

Only basic modules for test signal generation, test response recording, IR editing, spectral analysis, and convolution filter testing have been programmed using third party PD class objects together with self developed PD objects like the test signal generator that is described in this paper.

The ExpoChirp project has received the interest of potential co-developers, and future efforts will focus on sub-topics like RT-analysis, STI analysis, multi-channel room acoustics measurement, Ambisonic B-format conversion, non-linear system analysis and convolution.

Descriptions and tutorials of modules will be published and updated online [2].

This paper discusses principles of the ESS method and the way it is implemented in the toolbox.

2 EXPONENTIAL SINE SWEEP METHOD (ESS)

2.1 Regular ESS implementation

Our choice of the ESS method is related to our familiarity with the technique, being one that we use regularly in our work as acousticians for analysis of audio systems and measurement of room acoustics parameters.

In 2000, Prof. Angelo Farina proposed the use of a sine signal with exponentially varied-frequency (Exponential Sweep Sine (ESS)) as test signal to measure the transfer function of weakly non-linear and time invariant systems [3.]; since then, this method has gained wide popularity and has been implemented in most of the commercially available software for IR measurements of enclosures and audio systems.

A key characteristic of the ESS technique is the ability to isolate any harmonic distortion orders (non-linear components) from the linear impulse response. The method has also been demonstrated to be very robust to minor time-variance of the system under test and to mismatch between the sampling clock of the signal generation and recording. A practical example is represented by the possibility of measuring an IR of a big PA/VA system fed with a test signal not reproduced by the recording laptop.

The test signal in this method can be defined by the following formula:

$$x(n)=\sin\left(\frac{\omega_1 \cdot N}{\ln(\omega_2/\omega_1)}\cdot\left(e^{\frac{n}{N}*\ln(\omega_2/\omega_1)}-1\right)\right) \quad (2.1)$$

where:
ω_1 is the normalized start angular frequency
ω_2 is the normalized stop angular frequency
n is the sample index
N is the number of samples

Even though the exponential chirp has many qualities that make it an excellent test signal, it is not absolutely perfect.

If we convolve the ESS signal expressed in (2.1) with its inverse, as formulated in [3.], we obtain the theoretical self-IR of the test signal; this is expected to have a linear frequency spectrum as it has not be influenced by any system.

Instead, the spectrum is characterized by ripple in frequency domain at the extremes, and overshoot may exceed +5dB in certain cases.

The chirp start and stop boundaries represent undesired frequencies, causing partial phase cancellations and summations over the signal which has in itself a constant amplitude and a constant frequency rise.
The regular formula only takes care of the start phase, while the stop phase is a function of start frequency, stop frequency and chirp length.

Figure 1 shows spectrum plots of a deconvolved exponential chirp which happened to end in an unfavorable phase. The deconvolved chirp was generated by Angelo Farina's Aurora plug-ins [5.], and loaded in a PD patch for inspection. No windowing was done on the chirp.

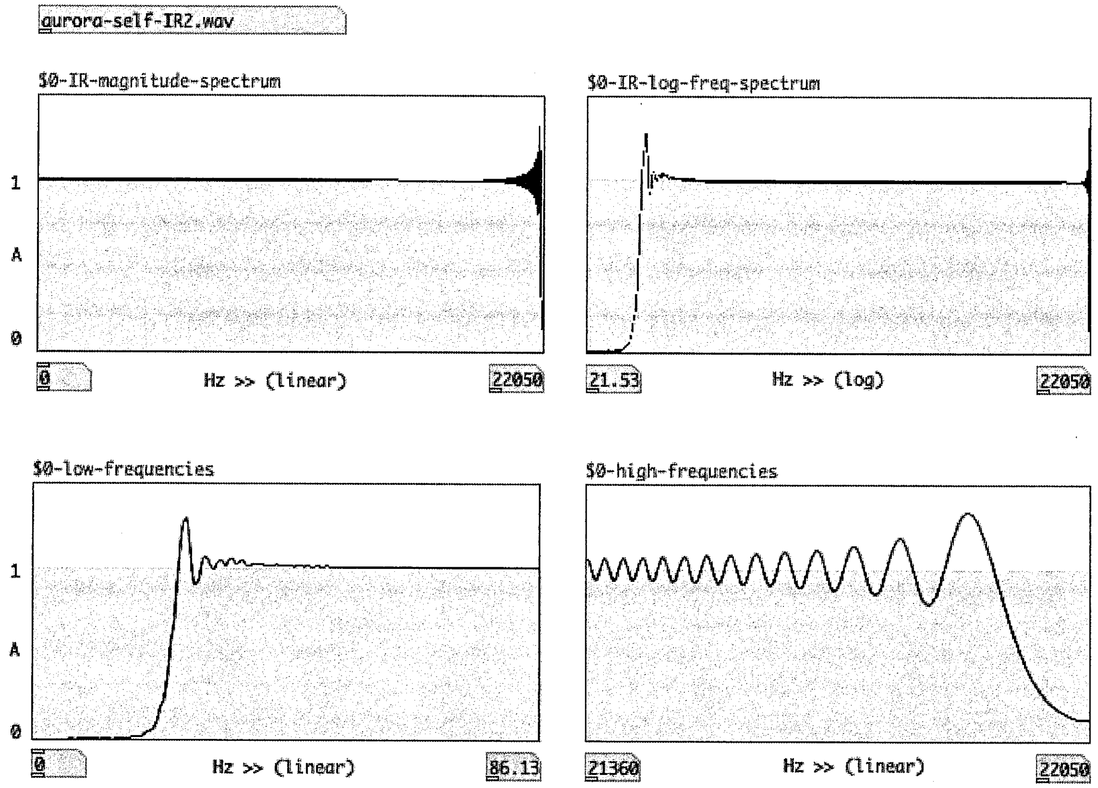

Figure 1: *magnitude spectrum of deconvolved exponential chirp*

Farina has discussed this problem of exciting the sound system with a step function when the chirp ends in an unfortunate phase [6]. A fade-out window can be applied to force a smooth decay of the high spectrum end but such a fade-out is represented in the deconvolved chirp by substantial variation of IR values around the central pulse, which should be a unit impulse ideally.

To avoid both these effects, step function and time domain artefact, Farina proposes to manually cut the chirp at the latest zero-crossing before its termination. With an alternative chirp definition however, the problem can be solved in a more systematical way, as will be shown in the next section 2.2.

A properly designed exponential chirp will even arrange harmonic distortion responses in such a way that each higher order response can be quantified separately, and the non-linearity components of the IR clearly identified.

2.2 Designing a phase-controlled exponential chirp in Pure Data

As shown in figure 1, ripples are only present at the low and high frequency extremes, which are not always of concern, but we would rather like to provide a reliable test signal for all possible measurement situations.

For the purpose of this discussion, the traditional formula as presented in equation (2.1) is repeated here, and components of it are distinguished and labeled.

$$x(n)=\sin\left(\frac{\omega_1\cdot N}{\ln(\omega_2/\omega_1)}\cdot\left(e^{\frac{n}{N}\cdot\ln(\omega_2/w_1)}-1\right)\right) \quad (2.1)$$

$$\frac{\omega_1\cdot N}{\ln(w_2/\omega_1)} \text{ Scaling factor } (2.2)$$

$$e^{\frac{n}{N}\cdot\ln(\omega_2/\omega_1)} \text{ Exponential curve } (2.3)$$

$$-1 \text{ Constant } (2.4)$$

The exponential curve has remarkable values at indexes n=0 and n=N:

$$e^{\frac{n}{N}\cdot\ln(w_2/\omega_1)}=e^0=1 \; \underline{\textit{and}} \; e^{\frac{n}{N}\cdot\ln(\omega_2/\omega_1)}=e^{\ln(\omega_2/\omega_1)}=\omega_2/\omega_1$$

Part of the solution was found in an alternative chirp formulation, which makes the chirp end in sine phase at Nyquist by definition, thus minimizing high frequency overshoot. This can also be seen as a limitation for the toolbox use (i.e. it would not be possible to perform frequency limited measurements at low frequency only) and we are aiming to address this in the next revision of the toolbox.

We observed that it would be advantageous if (ω_2/ω_1) were an integer, and the scaling factor an integer multiple of (2π). The chirp will then start and stop in the same phase (phase zero) if we drop the constant -1.

The most obvious way to make (ω_2/ω_1) an integer, is by creating chirps with an integer number of octaves P. Start frequency ω_1 becomes ($\omega_2/2^P$), and (ω_2/ω_1) reduces to 2^P.
Since a good approximation of a unit impulse (after deconvolution) is obtained with the normalized stop frequency at π radian (the Nyquist frequency), we use π as the default for ω_2.

The chirp definition in [expochirp~] is given as an equation set (2.5).

$$x(n)=\sin\left(\frac{(\pi/2^P)\cdot L}{\ln(2^P)}\cdot e^{\frac{n}{N}\cdot\ln(2^P)}\right) \text{ and } \frac{(\pi/2^P)\cdot L}{\ln(2^P)}=M\cdot\pi\cdot 2 \quad (2.5)$$

where:
P = an integer number of octaves
L = ideal chirp length (floating point value)
N = chirp length (= L rounded to integer)
M = a positive non-zero integer

The above is implemented in ExpoChirp Toolbox by letting the user select the desired number of octaves and a maximum chirp length first, from where L and N are computed. With the chirp ending in sine phase, overshoot and ringing at the high spectrum side are systematically kept at a minimum, well under +1dB (see Figure 4 in section 2.6).

Therefore a new Pd class [expochirp~] was written in C, calculating with double precision internally, while rendering the chirp and it's inverse as 32 bit audio signals.

2.3 Instantaneous frequency curve

Because of the rounding of L to N in the phase-controlled exponential chirp, there will be a small deviance in the resulting frequency range, as compared to the desired range. The instantaneous frequency curve must be inspected to see how important the error is. The frequencies are represented by the first derivative of the phase curve:

$$x'(n)=\frac{(\pi/2^P)*L}{\ln(2^P)}*\frac{\ln(2^P)}{N}*e^{\frac{n}{N}*\ln(2^P)}=(\frac{\pi}{2^P})*\frac{L}{N}*e^{\frac{n}{N}*\ln(2^P)} \quad (2.6)$$

This shows that every frequency is multiplied by L/N , where it would normally be $N/N = 1$.
The frequency deviance is not a frequency shift but a frequency stretch. Chirp length N is very large in real test conditions, typically 100,000 to 1,000,000 samples. The frequency deviance then, is in the order of a factor 0.00001. The advantage of a phase-controlled chirp outweighs this minor frequency stretch for practical purposes, but for cases with extremely short chirps the importance of the impurity should be considered.

With the phase-controlled chirp comprising an integer number of octaves P , the chirp has P equally-sized segments which all start and end in sine phase. As a consequence, frequencies with octave intervals or other harmonic intervals will appear with identical phase (more on this in section 2.4). In Figure 2, the first two octaves of such a chirp are shown. Each subsequent octave segment has a doubled number of cycles.

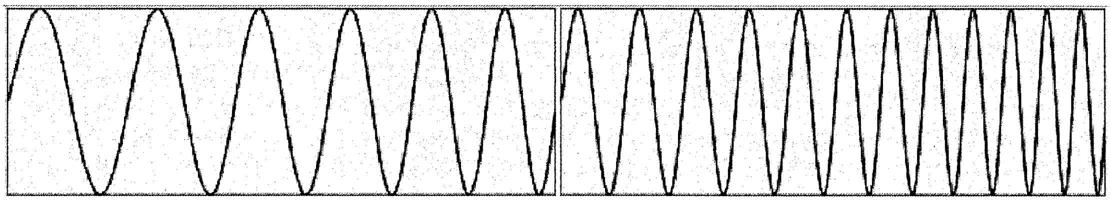

Figure 2: *first two octaves of a phase-controlled exponential chirp*

2.4 Inverse of the exponential chirp

In order to calculate the impulse response, the recorded test signal should be deconvolved with its time reversal that would also have its amplitude inverted as described in [3.] and [4.] , while taking proper amplitude scaling into account.

If each subsequent equally-sized time interval in the chirp has a doubled number of cycles, then it must be the case that there is an amplitude decay from the low to high frequencies, not in time domain but in the spectrum.

For any frequency, the second harmonic has half the amplitude of the fundamental, a 6 dB per octave decay. The general definition of a 6 dB per octave decay is 2^{-p} where p is the octave index (p varies between 0 and P). This is a curve starting with value 1 at index 0, and applying the curve to the inverse chirp causes an amplitude loss respective to the original chirp, which must be compensated. An exact amplitude compensation factor **C** is presented in (2.7).

$$C=P\cdot(\frac{\ln(2)}{1-2^{-P}}) \quad (2.7)$$

Figure 3 below show a graphical representation of the 6 dB decay the integral of the decay curve and the compensation factor.

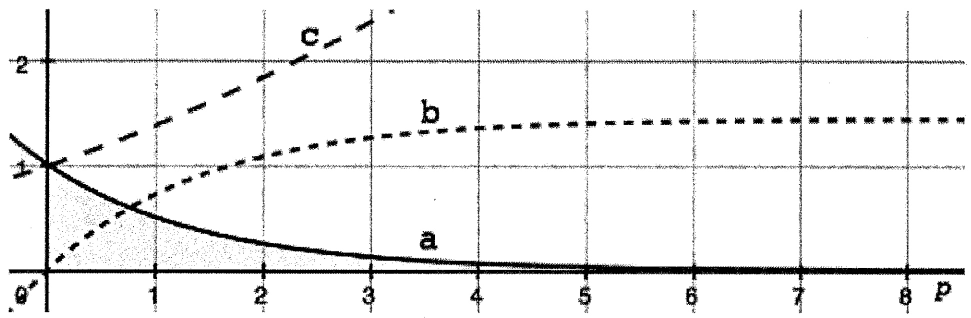

Figure 3: *curve (a) 2-p decay, (b) integral of the decay curve (eq. 2.7), (c) amplitude compensation factor (eq. 2.8)*

The time-reversed chirp, amplitude decay and amplitude compensation factor together form the inverse chirp definition as formulated in equation (2.9).

$$x^{-1}[n]=x[N-n]\cdot(2^{\frac{P}{N}})^{-n}\cdot(\frac{P\cdot\ln(2)}{1-2^{-P}}) \quad (2.9)$$

This calculates an inverse chirp with average amplitude equal to the forward chirp, being $\frac{1}{\sqrt{2}}$.

When convolving the chirp with its inverse, the result is an approximate pulse with height N/2. A 2/N normalization factor is applied after convolution, in order to get an IR with the unit impulse as a reference.
This normalization factor can be used as a reference when calibrating files for auralisation purposes[9].

2.5 Non-linear system aspects and phase-controlled chirp

Antonin Novak introduced the use of a phase-synchronized exponential swept-sine signal for the purpose of non-linear system identification [7].

If higher order impulse responses (products of harmonic distortions) are to be accurately traced, harmonics in the chirp must be phase-aligned, otherwise phase cancellations and summations may corrupt the analysis result.

Comparison of Novak's chirp definition and the method proposed in this paper reveals that in both cases the calculation of an ideal chirp length (given the other parameters) produces the phase alignment of harmonics.

At the time of writing, we have not yet worked on routines for non-linear system identification in the ExpoChirp toolbox. However, with future extensions of the toolbox in mind, it is a fortunate coincidence that [expochirp~] generates a chirp suitable for this purpose.

2.6 Further ripple reduction: fade-in window

Though the phase-controlled chirp definition minimizes frequency ripple at the high spectrum end of the chirp, this intervention did not do anything for the low frequency ripple.

Even when the signal starts and stops in sine phase, the overshoot at the start frequency is around +3dB while at the stop frequency it is below +1dB, in the cases we tested. A fade-in window can help to produce a smoother spectrum. Remarkably, a fade-in window does not raise coefficient values in the deconvolved chirp, in contrast with the case of a fade-out window as it was mentioned in the previous paragraphs.

Figure 4 below shows the magnitude spectrum plots of a deconvolved exponential chirp phase controlled and with the fade-in window applied.

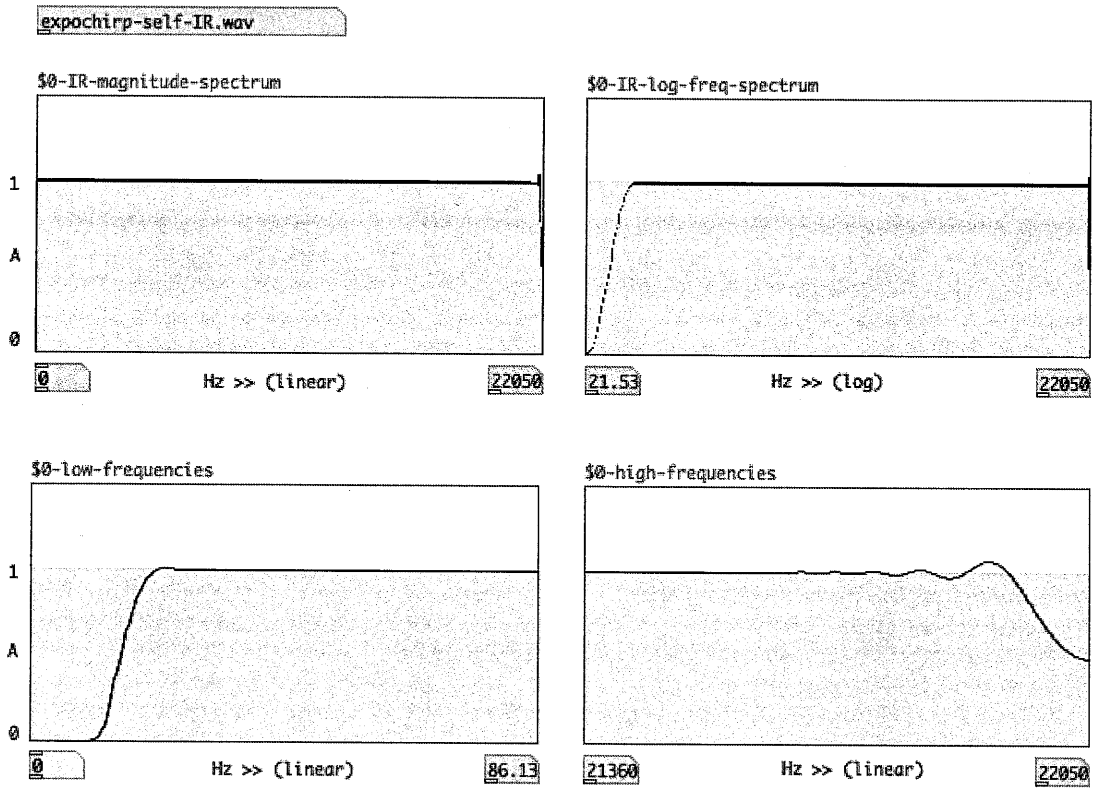

Figure 4: *magnitude spectrum plots of deconvolved fade-in-windowed and phase-controlled exponential chirp*

Experiments with a sine-shaped fade-in shown that the window must be fairly long to be sufficiently effective, in the order of a second.

At the low frequency side, this is not problematic because the chirp can be easily extended downwards below 20 Hz.

We decided to implement a optional fade-in window stretching over exactly the first octave in the chirp. For test signals with ten or more seconds length, this has a very beneficial effect.

Figure 4 shows spectrum plots of a phase-controlled chirp with the one octave long fade-in.

The chirp was calculated to include 11 octaves, of which the first one is modulated by the window. The transition band is less steep and the flat spectrum region starts exactly 10 octaves below Nyquist. With this systematic window definition, the exact start point of the flat spectrum range is always known.

2.7 Software

This section is intended to describe the modular structure of ExpoChirp and its functionality.

As described in the previous sections, ExpoChirp software needs Pure Data programming environment to be installed in its extended version [8].

When opening the ExpoChirp software, the first module we would find is the Toolbox general module which shows all the 4 tools modules that perform different tasks for IR measurements as shown in Figure 5 below.

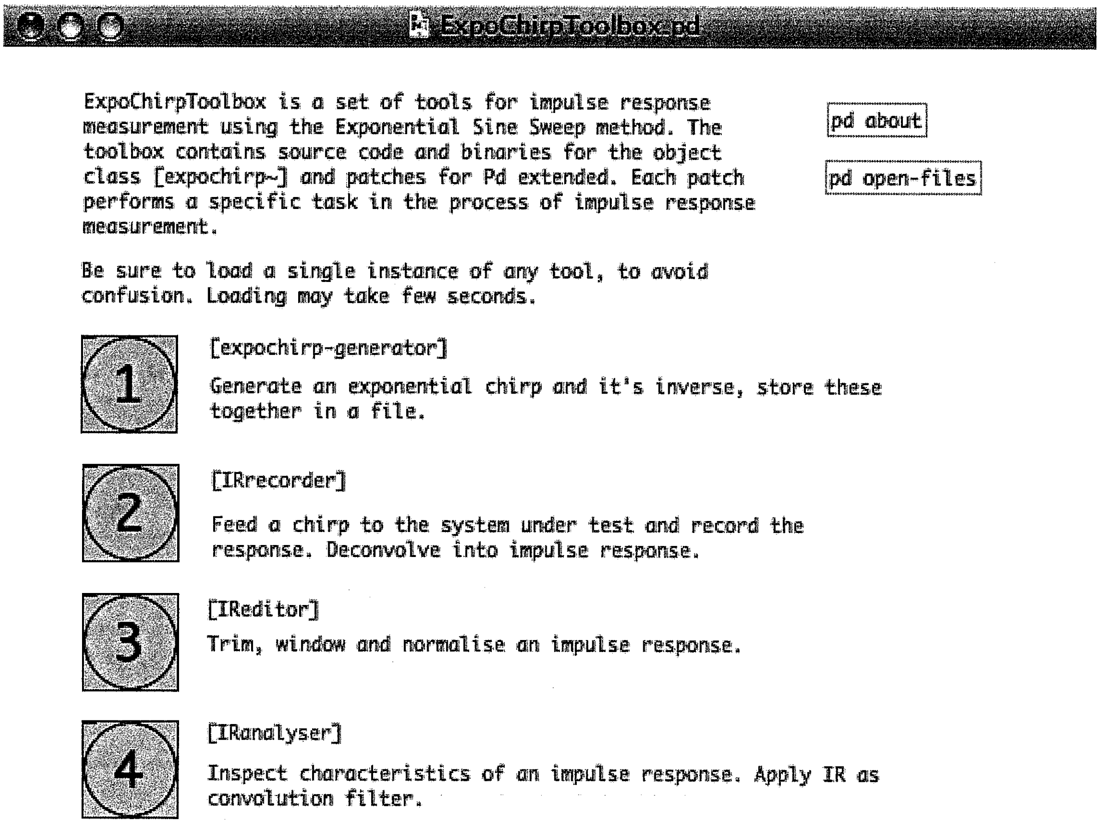

Figure 5: *Toolbox general module*

These tools modules are:

1. **[expochirp-generator]**: this is the module where the test signal and its inverse is generated while storing both in a file readable by the other modules; in this module the self-IR can be calculated as well to understand the quality of the test signal.
2. **[IRrecorder]**: this is the module where the actual IR measurement can be performed and the IR calculated including all the non-linear linear components; the test signal to be used is generated in the [expochirp-generator] module but, as said previously, an external

source can be used providing it is reproducing the same audio file generated by the software module.

3. **[IReditor]**: this is the module where the calculated IR can be trimmed, deleting the non-linear parts if only the linear IR is needed; the module automatically calculates the true "zero time" of the IR allowing the exact evaluation of the time of flight of the direct sound.
4. **[IRanalyser]**: this is the module where the calculated and trimmed IR can be inspected (its magnitude and frequency response evaluated) and can also be used as a convolution filter; this is the less developed module which is under discussion for future developments.

2.7.1 [expochirp-generator]

Figure 6 below shows the first window that appears when accessing the [expochirp-generator] module presenting the four different operations that could be performed:

1. generate the test signal and its inverse
2. Compute self-IR (if needed and interested)
3. Check spectral data
4. Save self IR as wav or aiff file

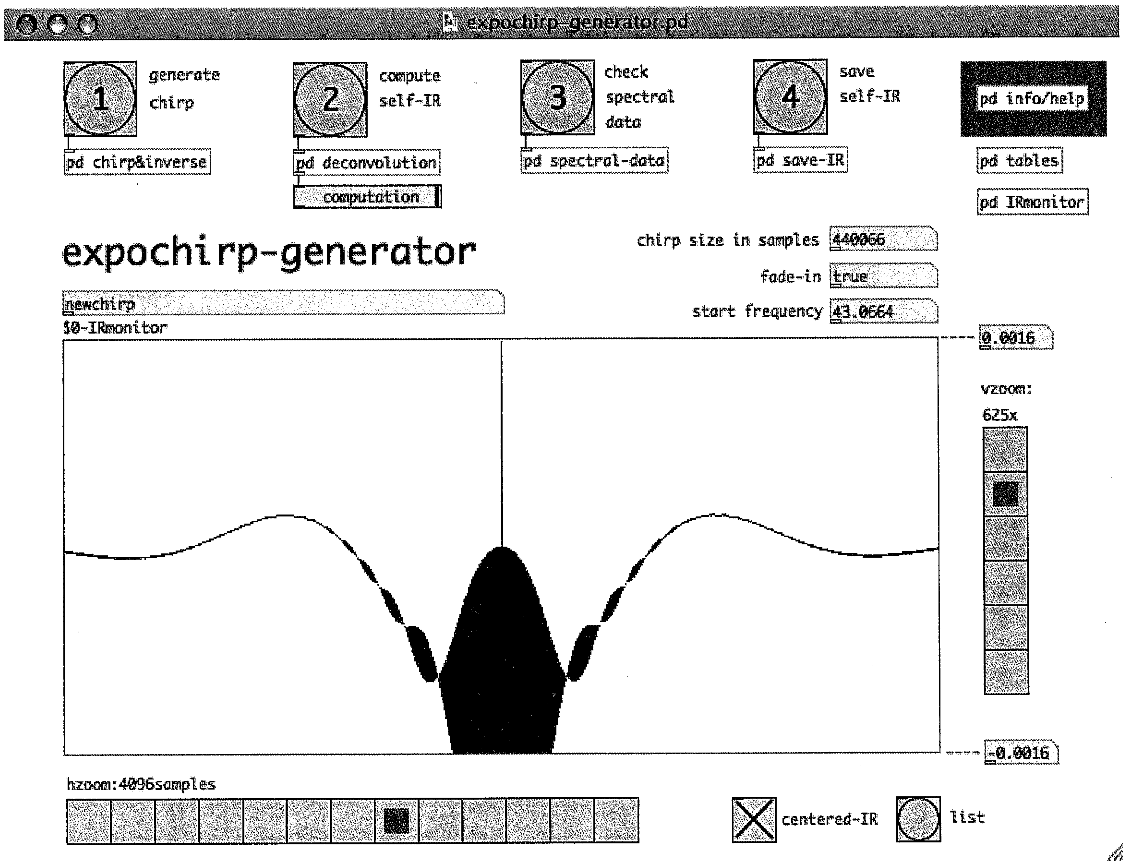

Figure 5: *[expochirp-generator] module*

The most important operation we would look at is the chirp generator; a screen shot is shown in Figure 6 below.

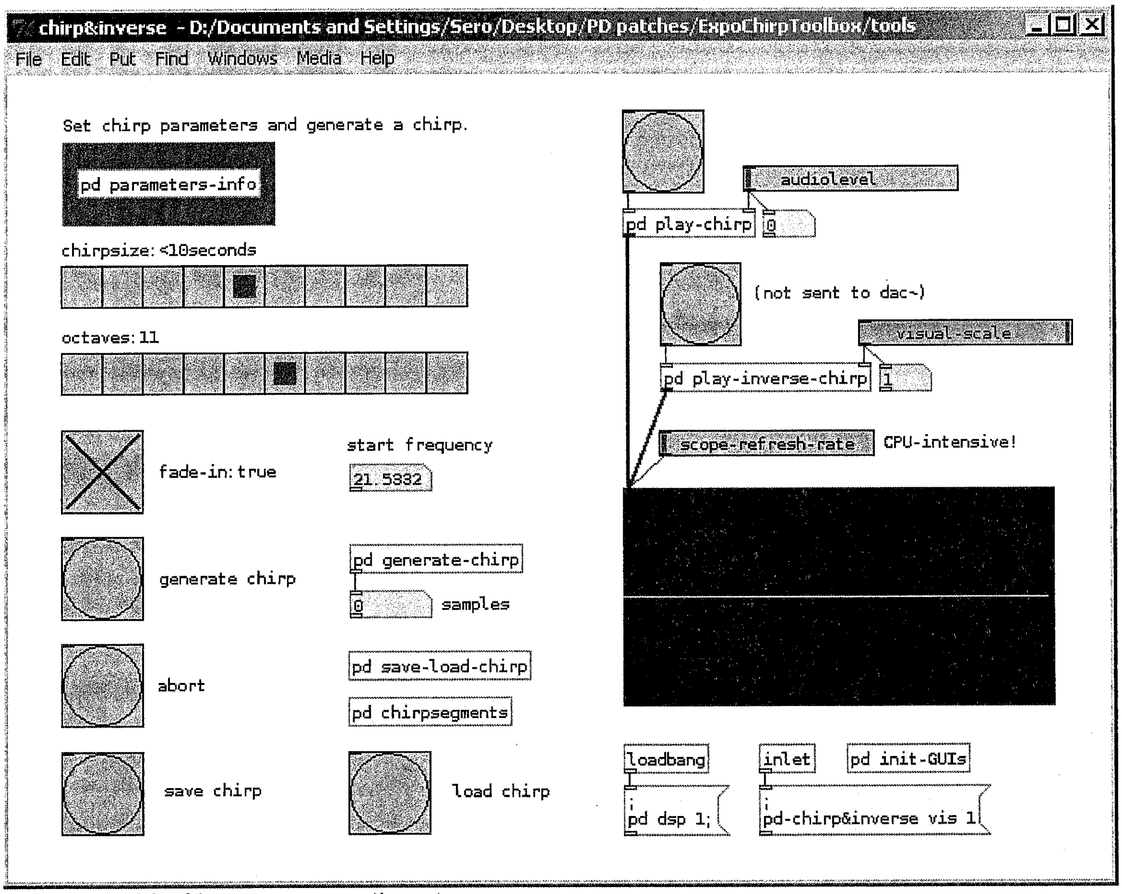

Figure 6: *chirp&inverse generation step*

The chirp&inverse tool allows creation of the test signal (ESS) and its inverse by defining its main parameters:

- Chirp size in seconds

- Octaves (note: this will always create a signal that will end at Nyquist frequency, only the start frequency will vary)
- Fade-in-window (on by default)

It is also able to play the test signal through the reproduction system for a quick check on audio quality (it will play the chirp when pushing the generate chirp button).
The intention with this module is that the user would create his/her personal set of test signal with different characteristics based on his/her experience and would not access this module frequently.

Test signals are stored as proprietary files (.xchirp) in the location of choice but the file can be read also with a text editor.

2.7.2 [IRrecorder]

This is the module where the IR measurements can be performed and the impulse response computed by deconvolution of the recorded signal with its inverse. A screenshot is shown in Figure 7 below.

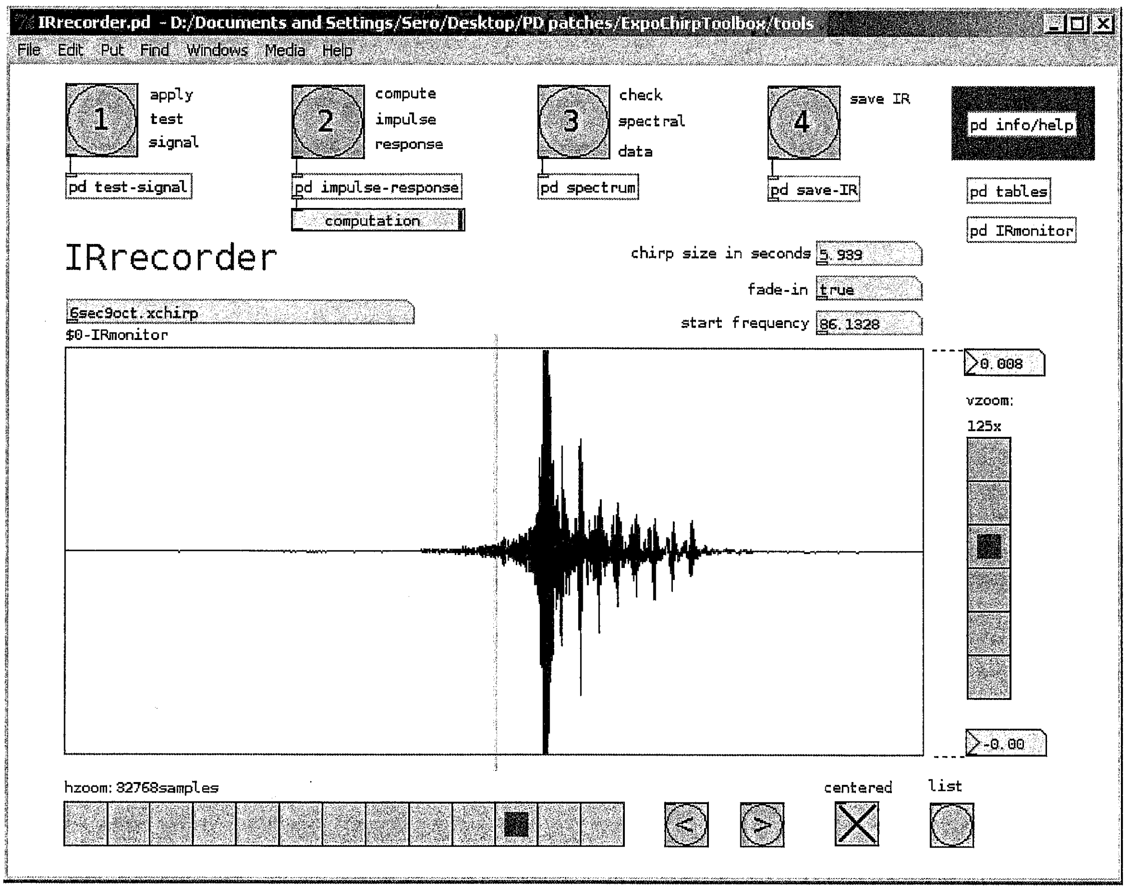

Figure 7: *[IRrecorder] module*

The module presents 4 different operations that can be performed for the purpose of calculating the IR that are listed below:

1. **Apply test signal:** this is the operation that permits to reproduce and record the test signal;

2. **Compute impulse response:** it performs the deconvolution of the recorded test signal with its inverse that has been loaded in the software when loading the .xchirp file above;
3. **Check spectral data:** it allows an initial check of the magnitude and phase spectrum of the IR;
4. **Save IR:** it saves the IR in a mono wave file (32 bit floating point)

The most important of the four operations above is in our view the first one: apply test signal, which allow to perform the physical IR measurement.

A screen shot of the operation is presented in Figure 8 below; it shows all the functions needed in performing an IR measurement.

The most important aspects of this module are, in our view, the possibility of accessing the computer audio setting easily, the possibility of control input and output gain by reproducing a pure sine test signal and the input of the expected reverberant tail that will increase the time the software will record sound in order to catch the high frequencies reverberant tail as well.

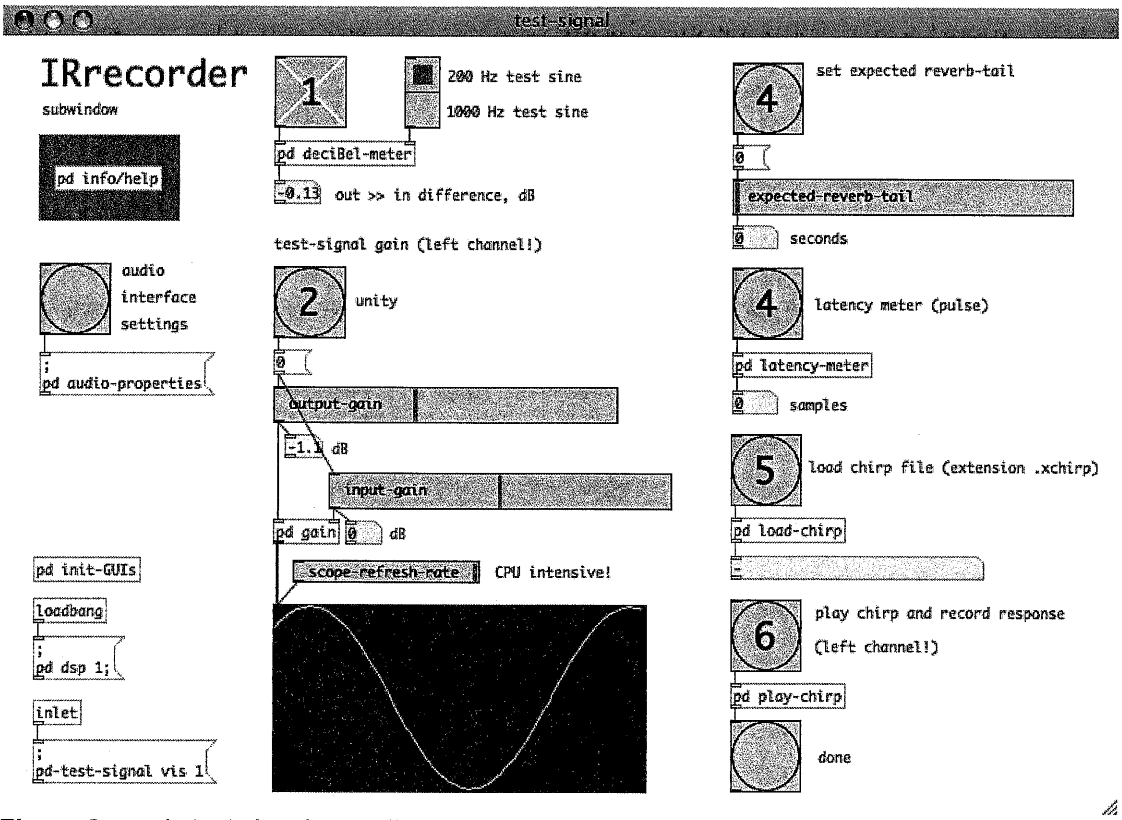

Figure 8: *apply test signal operation*

We also included a latency meter checker for loopback measurements, which will automatically calculate the latency of the system.

Of course, the loaded chirp file will load the test signal and its inverse in the software buffer.

2.7.3 [IReditor]

[IReditor] module has been implemented with the purpose of providing an audio file editor tool to the ExpoChirp toolbox which contains some advantages to perform the same operation with traditional audio editor tools.

The module allows an impulse response to be loaded in wav or aiff format and trimming of the file in order to delete the non-linear components of the measurements if a linear IR is needed; it also allows the application of a time window (recommended only if the IR would be used as a convolution filter) and to apply the normalization factor, described in the previous section 2.4, in order to obtain an IR with the unit impulse as a reference.

If the IR has been generated with the [IRrecorder] module, the editor will show the true zero time point of the IR in the center of the graph as cue point.

Figure 9 below shows a screenshot of the module.

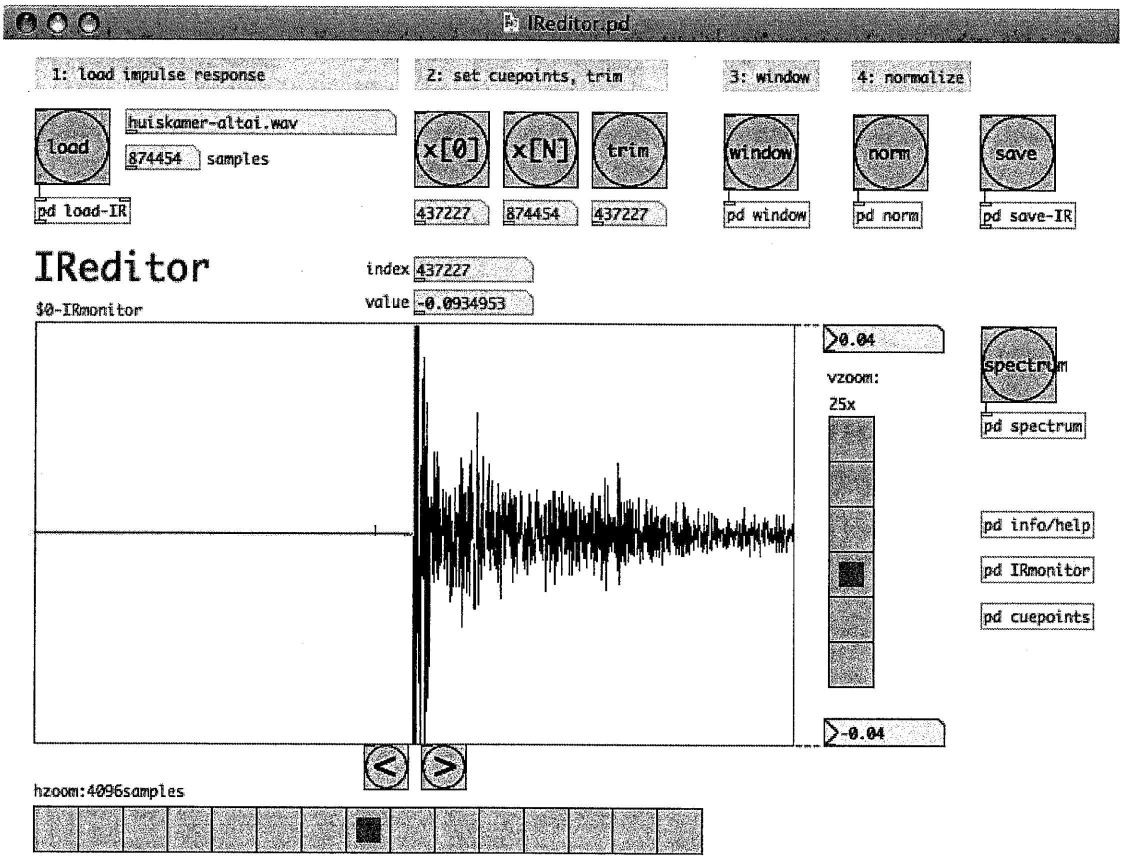

Figure 9: *[IReditor] module*

2.7.4 [IRanalyzer]

This module is only at the initial phase of development at the time of this paper and we aim to include a more comprehensive analysis of the IR and the inclusion of all the acoustic parameters derivation as described in ISO 3382-1:2009 and ISO 3382-2:2009.

Current development of the module only include a magnitude spectrum and a frequency spectrum analysis in dBFS as shown in Figure 10 below.

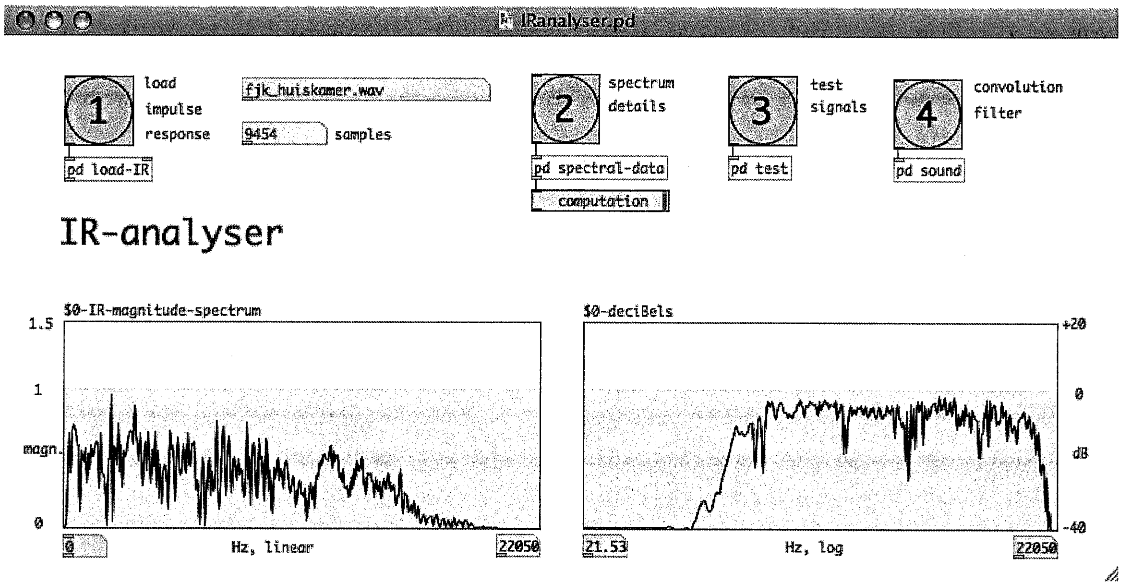

Figure 10: *[IRanalyser] module*

An additional element is represented by a simple convolution engine that can be used just to check the audio quality of the IR.

3 FURTHER DEVELOPMENT

The current state of the ExpoChirp toolbox allows IR measurements to be performed easily with a single microphone which is the standard method of capturing average acoustic characteristics of enclosures. It still relies on external software for the analysis of IRs and the derivation of their acoustical parameters for room acoustic analysis.

Our aim is to further develop this tool to incorporate the following:

- ISO 3382 part 1 &2:2009 room acoustic parameters
- Multichannel IR measurements (e.g. A-format) and included A-Format to B-Format conversion and Ambisonics reproduction modules
- Non-linear convolution (e.g. Volterra Kernel) and analysis for non-linear systems (e.g. audio outboard analysis)

The final purpose would be to develop a comprehensive IR measurement tool with live convolution functionality.

4 CONCLUSIONS

The design of ExpoChirp toolbox impulse response measurement tool, described in this paper, shows the possibility of achieving a professional quality tool developed on a free software platform (Pure Data).

At the moment, this tool does not offer the same operational speed and extensive functionality of commercially available software but resolves some operational issues for real-life measurements that have not been addressed yet in these.

We aim to present an extensive measurements validation during the presentation at the 27th Reproduced sound conference 2011.

5 ACKNOWLEDGEMENT

The ExpoChirp project would have not been possible without the amazing work from Miller Pluckett and the Pure Data community. We are also indebted to Angelo Farina for his extensive publications on ESS method and for his comments on Expo Chirp project.

This work is an amalgamation of professional acoustics and extensive DSP programming and would never be possible without the help of Pure Data web community.

Special thanks to Alistair Meachin for his help in checking the work throughout its development and for his vision on future use of this software.

6 REFERENCES

1. http://puredata.hurleur.com/index.php

2. www.katjaas.nl/expochirp/expochirp.html

3. Angelo Farina, "Simultaneous Measurement of Impulse Response and Distortion with a Swept-Sine Technique", *pre-print of the 108th AES Convention, 19-22 February 2000.*

4. Martin Holters, Tobias Corbach, Udo Zölzer, "Impulse response measurement techniques and their applicability in the real world", *proceedings of the 12th Int. Conference on Digital Audio Effects (DAFx-09), Como, Italy, September 1-4, 2009.*

5. http://www.aurora-plugins.com/Aurora_XP/index.htm

6. Angelo Farina, "Impulse Response Measurements", *paper for the 23rd Nordic Sound Symposium, 27-30 September 2007.*

7. Antonin Novak, Laurent Simon, Frantisec Kadlec and Pierrick Lotton: "Non-linear system identification using Exponential Swept-Sine signal", *IEEE Transactions on Instrumentation and Measurement, 2009-10-30.*

8. http://puredata.info/community/projects/software

9. S Di Rosario, A Meachin, "Auralisation Level Calibration", Proceedings of the Institute of Acoustics, Vol.30 Pt 6. 2008