fIReLab User Manual

v1.0 – Joe Simon Labs 2025

Table of Contents

EULA

END USER LICENSE AGREEMENT

IMPORTANT: Please read this End User License Agreement ("Agreement") carefully before installing or using the software ("fIReLab") provided by "Joe Simon Labs" ("Licensor," "we," "our," or "us"). By installing or using the software, you ("Licensee," "you," or "your") agree to be bound by the terms of this Agreement.

1. LICENSE GRANT

1.1 **Grant of License**: Joe Simon Labs grants you a non-exclusive, non-transferable, limited license to install and use "fIReLab" software on a single device, subject to the terms of this Agreement.

1.2 **Permitted Use**: You may use "fIReLab" for personal, commercial, or educational purposes as long as you comply with this Agreement and any applicable laws.

1.3 **Restrictions**: You may not:

- Reverse-engineer, decompile, disassemble, attempt to derive the source code of, or otherwise tamper with the software.
- Distribute, rent, lease, lend, sell, or sublicense the software to any third party.
- Modify or create derivative works based on the software without prior written consent from Joe Simon Labs.
- Use the software for unlawful purposes or in any manner that violates the terms of this Agreement.

2. COPYRIGHT AND OWNERSHIP

2.1 The software is licensed, not sold. Joe Simon Labs retains all ownership, copyright, and other intellectual property rights in the software, including any updates, enhancements, or modifications.

2.2 You acknowledge that the software is protected by copyright laws and international copyright treaties, as well as other intellectual property laws.

3. TERM AND TERMINATION

3.1 **Term**: This Agreement is effective until terminated by either party.

3.2 **Termination**: Joe Simon Labs may terminate this Agreement if you fail to comply with any term or condition of this Agreement. Upon termination, you must immediately uninstall and destroy all copies of the software.

4. WARRANTIES AND DISCLAIMERS

4.1 **No Warranty**: The software is provided "as is," and Joe Simon Labs makes no warranties, express or implied, regarding the software's performance, accuracy, or fitness for a particular purpose.

4.2 **Limitation of Liability**: In no event shall Joe Simon Labs be liable for any damages arising from the use or inability to use the software, including but not limited to direct, indirect, incidental, special, or consequential damages, even if Joe Simon Labs has been advised of the possibility of such damages.

5. PRIVACY AND DATA COLLECTION

This software collects no data.

6. GOVERNING LAW AND DISPUTES

6.1 **Governing Law**: This Agreement shall be governed by and construed in accordance with the laws of Denmark, without regard to its conflicts of laws principles.

6.2 **Dispute Resolution**: Any dispute arising out of or in connection with this Agreement shall be resolved through binding arbitration in Denmark, under the rules of the Danish Arbitration Institute, unless otherwise agreed by both parties.

7. MISCELLANEOUS

7.1 **Entire Agreement**: This Agreement constitutes the entire agreement between the parties with respect to the software and supersedes all prior or contemporaneous agreements or communications, whether written or oral.

7.2 **Severability**: If any provision of this Agreement is found to be invalid or unenforceable, the remainder of the Agreement will remain in full force and effect.

7.3 **Amendments**: Joe Simon Labs reserves the right to update or modify this Agreement at any time. You will be notified of any material changes, and continued use of the software after such changes constitutes acceptance of the new terms.

8. CONTACT INFORMATION

For any questions regarding this Agreement or the software, please contact Joe Simon Labs at: joesimonlabs@gmail.com

BY INSTALLING OR USING THIS SOFTWARE, YOU ACKNOWLEDGE THAT YOU HAVE READ, UNDERSTOOD, AND AGREED TO BE BOUND BY THE TERMS OF THIS EULA.

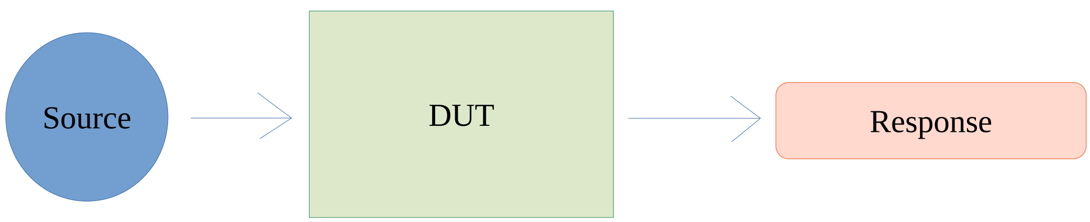
PortAudio License

This software (fIReLabs) uses the open source library PortAudio, which is included in the installer. The license of PortAudio follows:

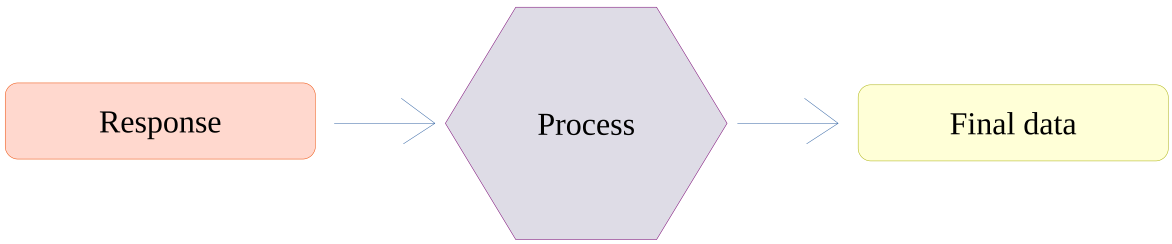
Introduction

fIReLab is a tool designed to ease some of the common tasks that arise when working with linear systems, such as performing measurements and post-processing them. Many of these have identical procedures, as they commonly involve a similar setup which consists of producing a known excitation signal and observing the response of the DUT (device under test). Then, when the response of the DUT has been observed, some tasks related to processing the measurement data are common. For these needs, fIReLab includes a “recorder” and a “processor”.

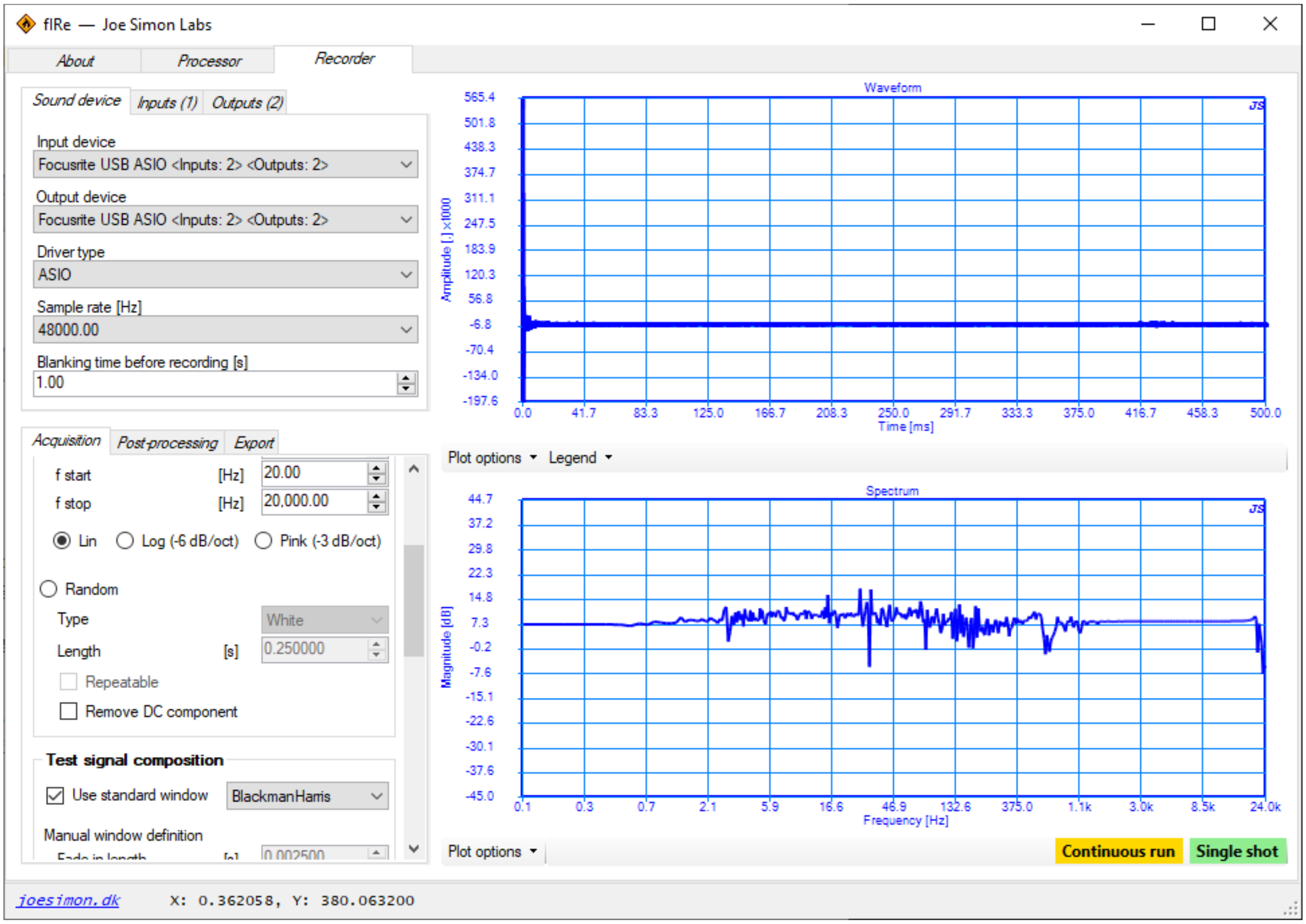
Typical use of the recorder:



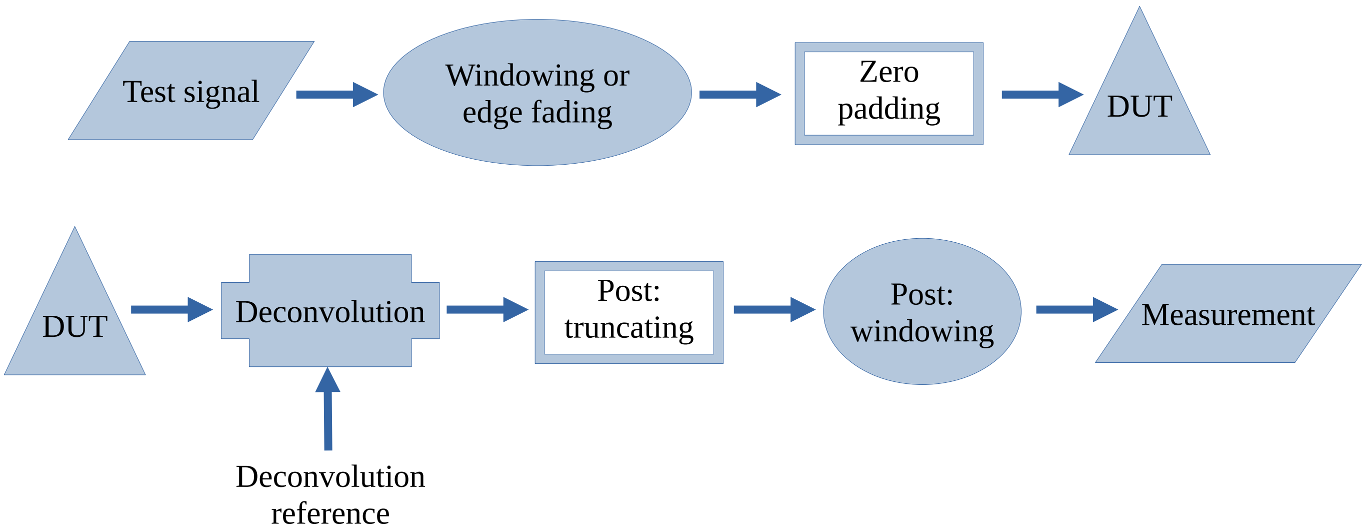
Typical use of the processor:



Recorder


fIRe — Joe Simon Labs
About
Processor
Recorder
Sound device
Inputs (1)
Outputs (2)
Input device
Focusrite USB ASIO <Inputs: 2> <Outputs: 2>
Output device
Focusrite USB ASIO <Inputs: 2> <Outputs: 2>
Driver type
ASIO
Sample rate [Hz]
48000.00
Blanking time before recording [s]
1.00
Acquisition
Post-processing
Export
f start [Hz] 20.00
f stop [Hz] 20,000.00
Lin
Log (-6 dB/oct)
Pink (-3 dB/oct)
Random
Type White
Length [s] 0.250000
Repeatable
Remove DC component
Test signal composition
Use standard window BlackmanHarris
Manual window definition
Waveform
Amplitude [-] x1000
Time [ms]
Plot options
Legend
Spectrum
Magnitude [dB]
Frequency [Hz]
Continuous run
Single shot
joesimon.dk
X: 0.362058, Y: 380.063200

Block diagram

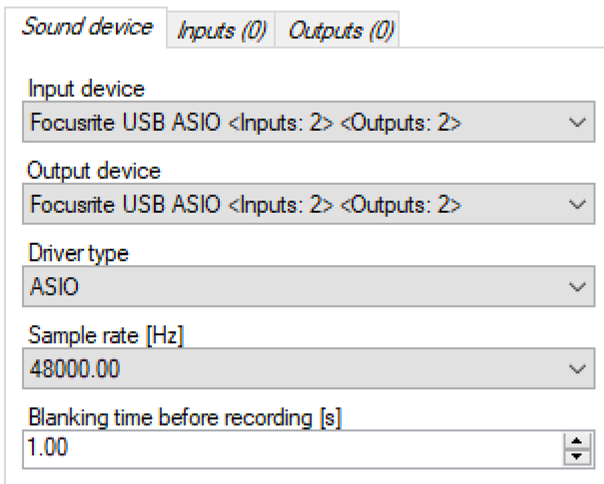


The test signal (from the different source types available) with optional pre-processing is sent to the DUT through the chosen hardware output(s) of the audio interface. The output from the DUT is received at the hardware input(s) of the audio interface and the optional post-processing is applied before being displayed.

The recorder can be used to produce output signals without the need for recording any of the inputs, working in that case as a playback system of the composed excitation signal.

Audio interface

Most audio interfaces should operate normally thanks to the use of the PortAudio library. For best results in terms of low-latency, consistent latency and glitch-free operation, the use of a device with ASIO drivers is recommended.



Usage:

1. Select the desired driver type (ASIO recommended if available).
2. Select the input and output devices (the same if possible).
3. Select which outputs and which inputs are to be used by checking. Using inputs is optional (it will then be a test signal generator). Using at least one output is necessary as the duration of the playback and recording is defined by the test signal generation characteristics. If no output signal is desired, a zero-amplitude test signal can be generated.

Notes:

- While it is possible to mix and match different devices for input and output operation, this is not recommended as instability or unexpected behavior may occur.
- Some audio interfaces may produce input and or output clicks or other artifacts when a streaming event starts (a new recording is started by the PortAudio library). For such eventuality, the "*Blanking time before recording*" feature will allow for a certain amount of time to pass before any signals are sent or recorded to and from the DUT, thus allowing any possible remnants of the artifacts to disappear.

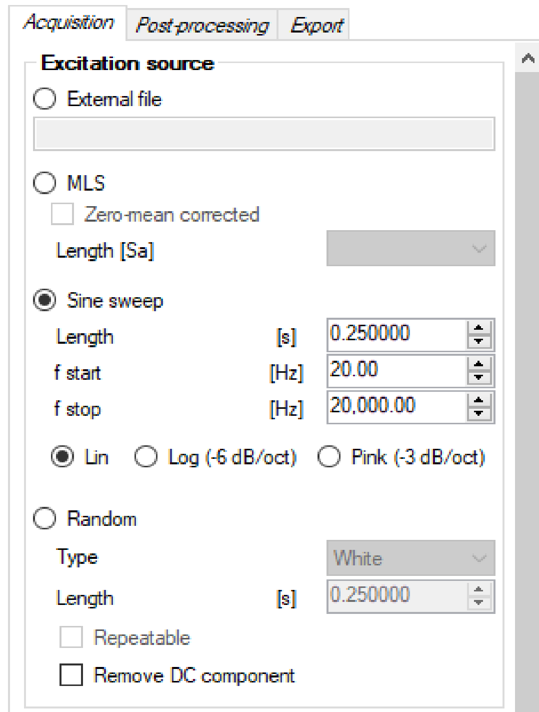
- Some interfaces offer a different channel count depending on the configured sample rate. In that case, fIReLab will not discover the changed channel count until the app is restarted. This limitation is imposed by the PortAudio library.
- fIReLab will not detect new audio interfaces after it has started. This limitation is imposed by the PortAudio library.

Generation of test signals

Signal source

The test signal originate from four difference sources:

1. An external file (*see valid formats*)
2. MLS (maximum-length sequence) generator
3. Sine sweep generator
4. Random generator



1) External file

Either drop a valid audio file on the text box below it or type the path to one. If an audio file is dropped, it will be checked for compatibility.

2) MLS

A standard MLS generator that can be set to produce sequences of the following lengths: 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768.

- If the "Zero-mean corrected" option is checked, the mean of the generated signal is subtracted before being sent to the hardware output.

3) Sine sweep

A sine signal with a frequency in the range given by "*f start*" and "*f stop*" with a total duration of "*Length*" seconds will be generated. The progression of frequency in the range can be:

- Linear (all frequencies are found with the same energy).
- Logarithmic (frequency density proportional to $1/f^2$).
- Pink (frequency density proportional to 1/f).

Choose the shape of the progression according to the needs and limitations of your test setup application. For audio applications, logarithmic or pink are typical choices.

Note: this generator can also be used to produce a single frequency sine setting "*f start*" and "*f stop*" to the same value.

4) Random generator

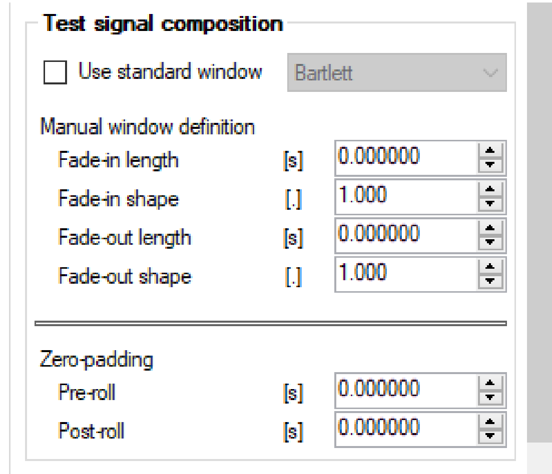
A pseudo-random generator is used to produce the following types of noise:

- White (uniform distribution)
- Pink (-3 dB / octave)
- Red (also known as brown) (-6 dB / octave)
- Violet (+6 dB / octave)

Notes:

- If the “Repeatable” option is checked, the generator is always restarted with the same seed, such that repeatable measurements can be performed. Keep in mind that this will make the generated signal predictable, but the generator remains pseudo-random, so there is no guarantee that any certain frequencies will be present in any specific manner.
- If the “Remove DC component” option is checked, the mean of the generated signal is subtracted before being sent to the hardware output.

Test signal composition



The test signal is a composition of the source signal with two possible additions (in this order):

- The application of a window (standard or manually defined by a fade-in and fade-out)
- The length extension with a pre and post zero-padding.

Standard window

- The source signal is amplitude modulated by one of the standard window shapes.
- The following window types are available: Bartlett, Hanning, Hamming, Blackman, Blackman-Harris, Gaussian, Tukey.

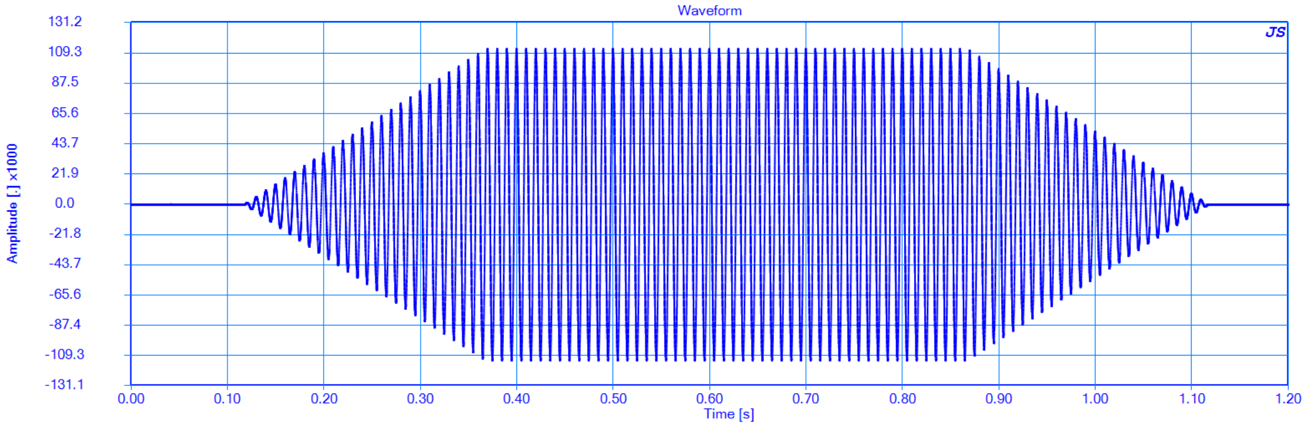
Manual window definition

A simple fade-in and / or fade-out can be applied to the source signal, with a chosen duration and shape. The “shape” parameter corresponds to the exponent of the gain factor applied at the beginning / ending of the source signal. The gain factor will be in the range [0, 1], thus between completely faded and completely present.

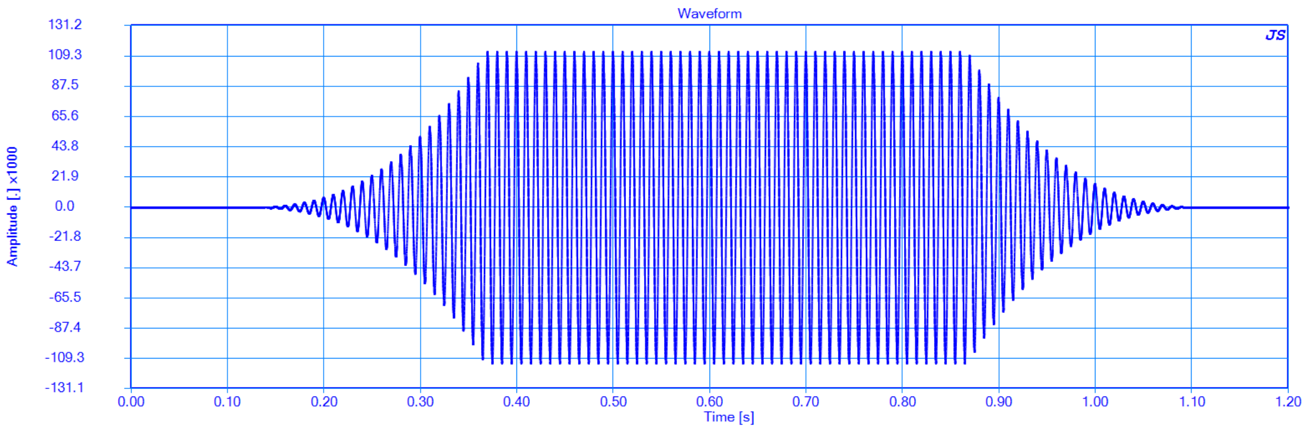
$$G=\left(\frac{t}{T}\right)^{S}$$

t *is the time of the sample relative to the fade length,* ***T*** *is the fade length*

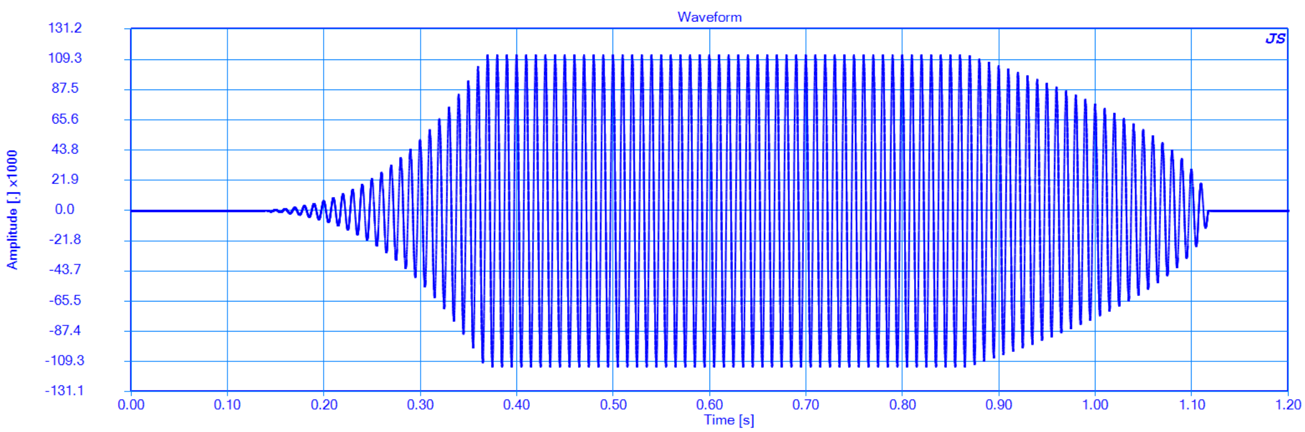
- A “shape” of 1.0 will produce a linear fade.
- A “shape” of > 1.0 will produce a more “audio taper” fade (slowly builds up at first, then faster towards the end of the fade). A value of 2.5 approximately corresponds to a typical analog audio potentiometer.
- A “shape” of < 1.0 will produce a more “reverse audio taper” fade.



Example of linear (shape = 1.0) fade-in and fade-out.



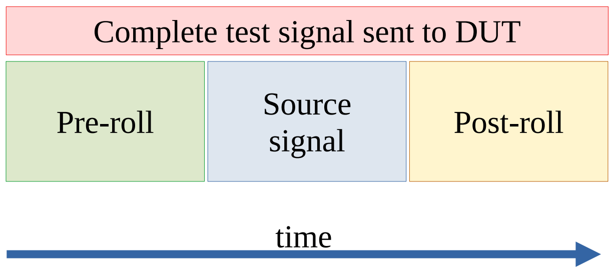
Example of "audio" (shape = 2.5) fade-in and fade-out.



Example of "audio" (shape = 2.5) fade-in and "reverse audio" (shape 0.5) fade-out.

Zero-padding

The "pre-roll" is the length of the extra time (silence) added at the beginning, before the source signal. The "post-roll" is the length of the extra time (silence) added at the end, after the pre-roll and source signal have taken place. These can be used for different purposes, such as giving the DUT time to reach stationary state (pre-roll) and accounting for the latency in the system such that the entire source signal is played through the DUT before the recording ends (post-roll).

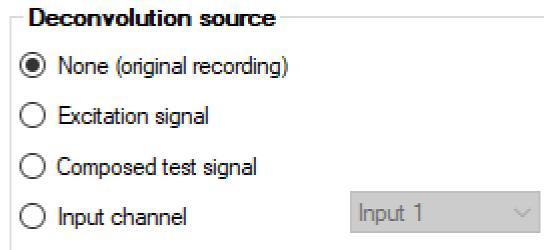


Note: in the majority of cases, some amount of post-roll will be necessary if the recording must observe the entirety of the source signal, since effectively all hardware devices will introduce some amount of latency due to the hardware and software buffers. The latency may not be constant (thus varying for each recording taken) so a generous post-roll time is sometimes required depending on the specific audio interface used.

Deconvolution

The signal produced by the DUT is recorded by the hardware input of the selected audio interface. Before it is averaged (if averaging is enabled) and displayed, it can be deconvolved with one of the following signals as reference:

- Excitation signal (source direct + windowing)
- Composed test signal (excitation signal + pre- and post-roll)
- A new recording taken from another physical hardware input of the audio interface.



Typically, the deconvolution is used when the objective of the measurement is to display the response of the DUT compared to a known reference, such as when observing its frequency response. If no deconvolution is applied to the recording, the measurement will simply display the captured audio signal (with the optional post-processing when selected).

Hardware response and latency removal use

Possibly the most typical use of the deconvolution feature is to minimize the contribution of the (*linear*) response of the audio interface to the measurement. Evidently, when measuring a DUT with any audio interface, the obtained measurement will be the accumulation of:

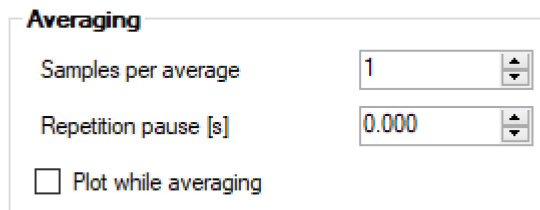
- response of DAC of audio interface
- response of the DUT
- response of the ADC of the audio interfaces

In order to minimize the contribution of the DAC and ADC of the audio interface to the overall measurement, an additional input of the interface is used to record the DAC output directly (without going through the DUT). Assuming that both ADCs and front-end analog circuitry in the audio interface are very similar, the obtained measurement after deconvolution will be almost entirely due to the DUT (and the circuitry of the analog output, typically negligible for a good quality interface). In addition, this will also eliminate any latency introduced by the hardware and software buffers, since it will be present equally on all recorded signals.

When a physical input cannot be left dedicated as a reference channel, there is still the option to deconvolve with the playback signal, with the obvious disadvantages of not being able to minimize the contributions of the ADC and analog front-end, as well as the latency. This may render averaging impossible if the latency is not constant in the system used.

Note: if the deconvolution source is another physical input channel, that channel becomes reserved for that purpose and is not available as a measurement input. This reference channel does not need to be checked as an input in the hardware configuration tab, as it is automatically included by fIReLab as needed.

Time-domain averaging



The measurement can be conducted multiple times and then averaged to, for example:

- increase the overall SNR
- observe the average behavior of a system that is not time-invariant
- obtain an average of different responses (such as when conducting room acoustics measurements on different locations)

The averaging takes place in the time domain, after the deconvolution has taken place (if enabled) and before the post-processing is applied.

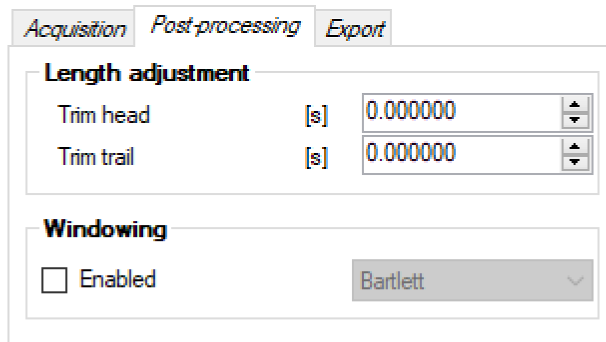
If the parameter "samples per average" is higher than 1, the overall measurement will be conducted said number of times, with a pause between shots corresponding to the parameter "repetition pause". The pause between averages may be needed to, for example, the DUT to reach stationary state after each shot.

If the option "Plot while averaging" is checked, fIReLab will pause briefly between averages to do the necessary post-processing, compute the frequency-domain response and plot the signals before moving on to the next average iteration. This may have a significant computation cost (specially noticeable if the recording is many seconds long) and will make the overall measurement process longer. It can be very helpful in some cases, such as to determine how many averages are needed to observe the desired improvement in SNR.

Post-processing

The measurement data can be post-processed before being displayed. This is often used to, for example:

- Reduce the length of the displayed data
- Improve the time-frequency display



The order of these two optional processing tools are:

1. Length adjustment
2. Windowing

Length adjustment

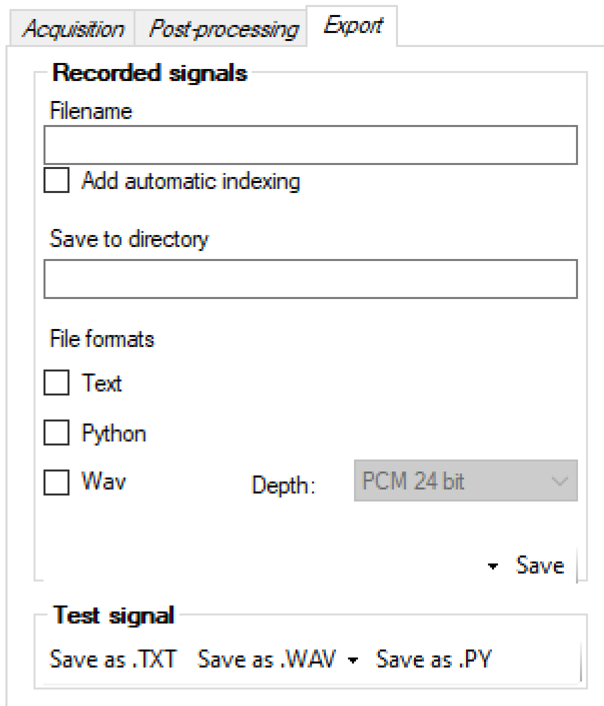
The length adjustment allows for some amount of signal to be removed from either end of the recording. The value entered for each end is in read in seconds and counted from the edge of the complete recorded signal (after deconvolution if enabled).

Standard window application

The same types of standard window found in the generator are available here: Bartlett, Hanning, Hamming, Blackman, Blackman-Harris, Gaussian, Tukey.

After this optional post-processing, all the signals are displayed in time and frequency domains.

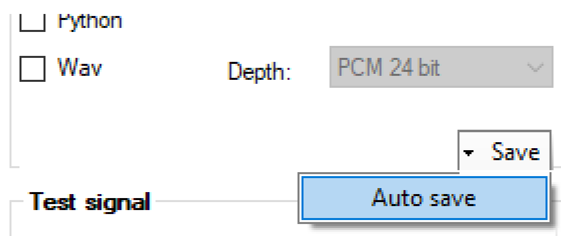
Exporting recorded data



The generated and recorded signals can be exported to data files for external analysis.

Exporting recorded signals

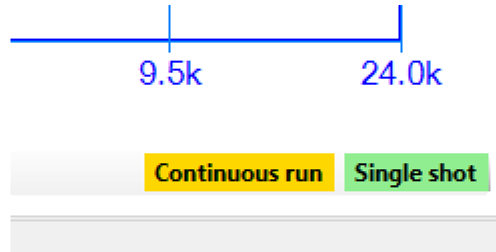
- Type in the name to be used under "Filename". It will be appended with the channel number in case of multiple channels being recorded simultaneously.
- Check "Add automatic indexing" to avoid overwriting existing files, such as when taking different measurements of the same setup which need to be stored separately.
- Type or drag and drop the path to which the recorded signals are to be saved.
- Check the desired format(s) the file should be saved in.
- Click the "Save" button to save the currently captured signals (displayed in the time and frequency domain plots of the recorder).
- **Optional:** enable "Auto save" to automatically perform a new save operation every time the recorder runs.



Exporting the test signal

The signal sent to the hardware outputs of the audio interface can also be saved for reference or separate usage using the buttons in the "Test signal" section.

Modes of operation



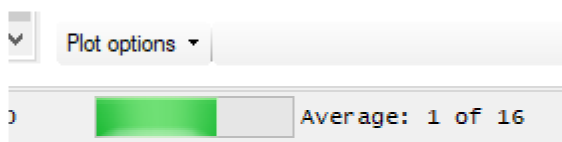
The recorder can operate in either of two modes:

- Continuous run
- Single shot

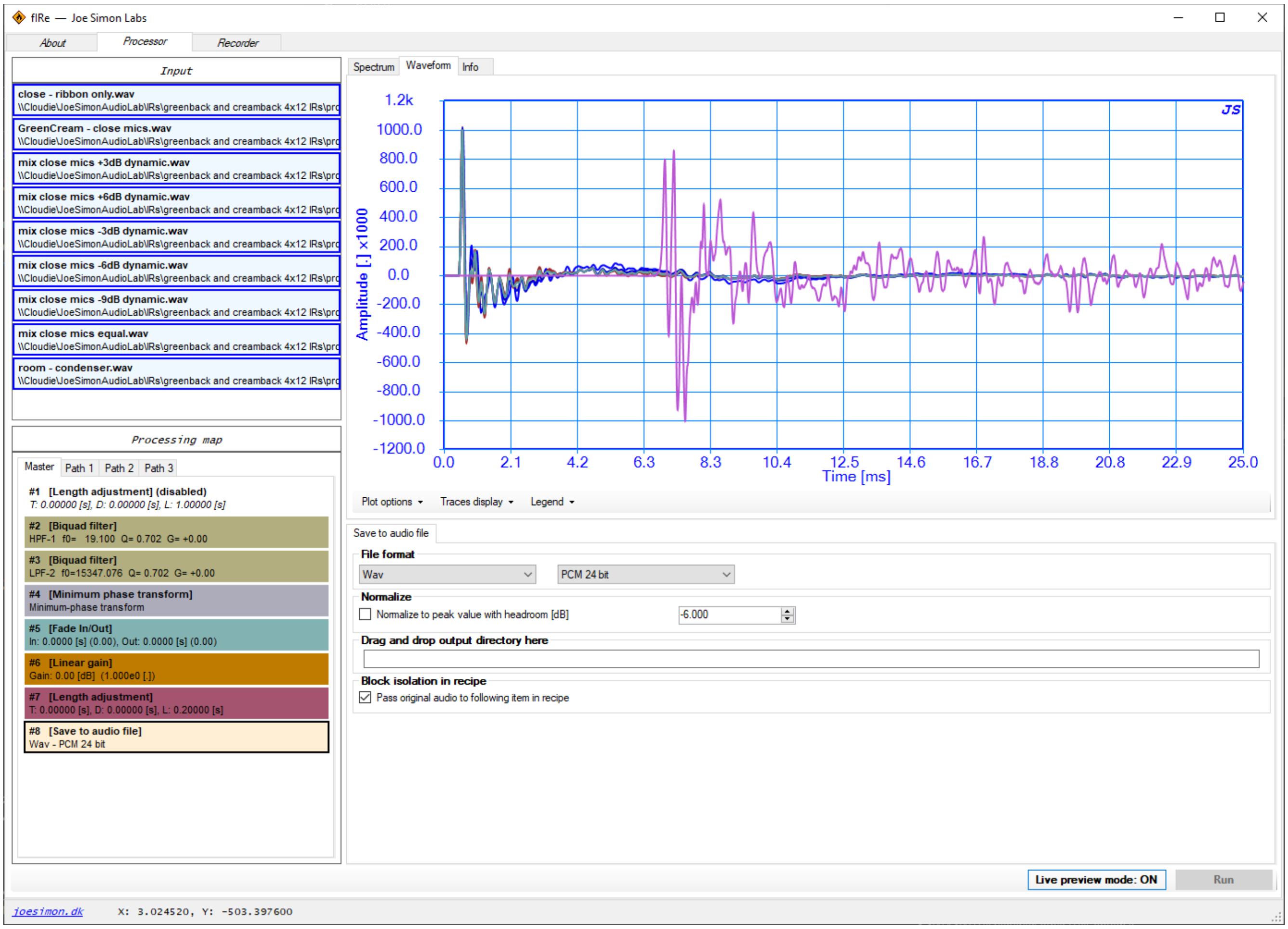
Much like an oscilloscope, the single shot mode will run the experiment immediately (as soon as the button is pressed), proceed to the record the amount of samples for the desired average count, and display the results after applying the necessary post-processing. The continuous mode will do the same but automatically trigger another shot after the previous one is finished.

The continuous mode can be stopped at any time by pressing the same button again (and the current shot is allowed finished). This is useful to, for example, observe the response of a DUT when some of its parameters are being changed.

During operation, the progress bar will display the current shot and overall progress.



Processor



Usage

The typical usage of the processor is quite simple:

- Load some signal files, typically containing impulse response data but not necessarily
- Apply a certain amount of processing blocks in a single- or multi-path fashion
- Display and or export the resulting signals

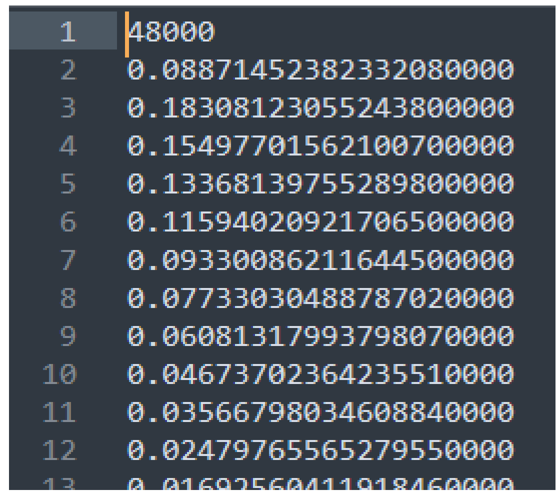
Supported file formats

Two types of input files are supported:

- Plain text
- WAV (formats accepted: 8 bit, 16 bit and 32 bit, fixed-point)

The plain text format must consist of:

- The sample rate (first line of the file)
- Each data sample in a new line. The samples may be in decimal (1.2345) or engineering notation (4.56e-3).

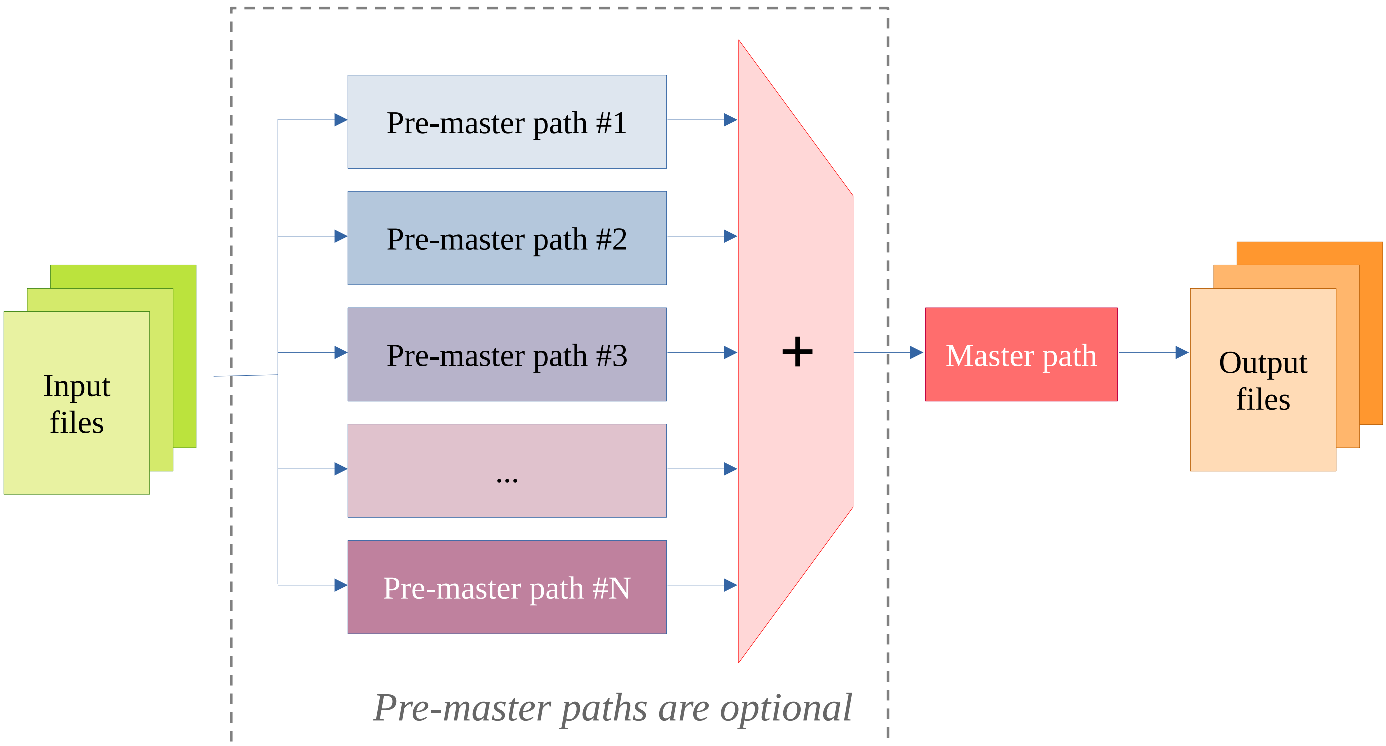


To load files into the processor, simply drag and drop them onto the "Input" box:

DROP
FILES
HERE

Multi-path processing

The input files can be processed through either a single or a two-step path system, as described below:

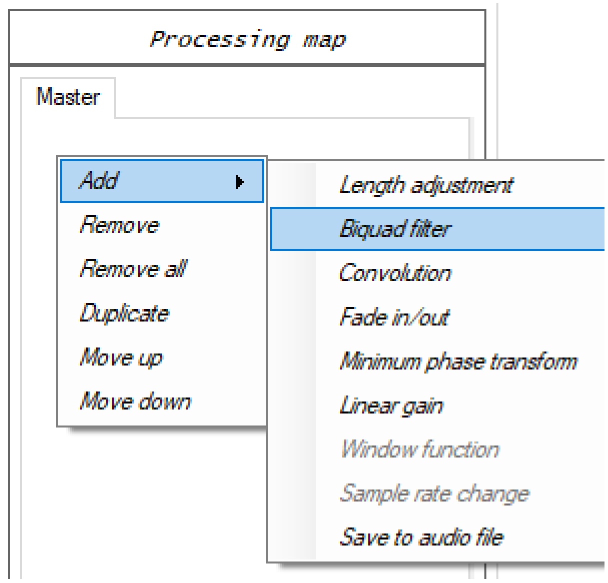


An arbitrary number of pre-master paths can be created (in principle limited only by the system memory available). These paths are then combined before passing through the master path. This process is performed individually for each of the input files to produce the corresponding output files.

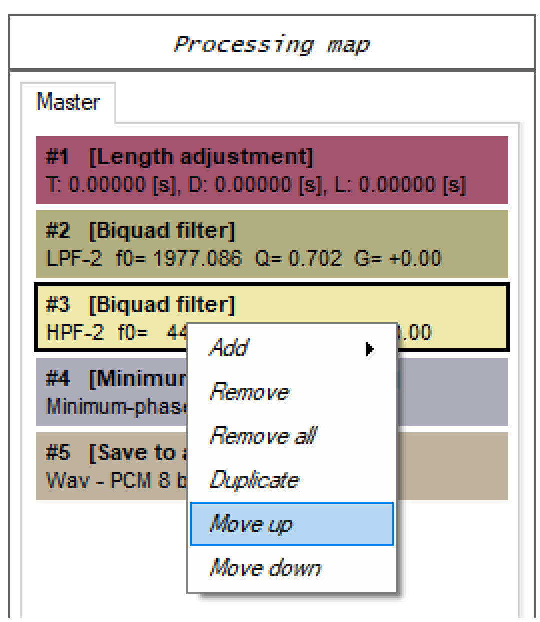
Each path (pre-master or master) contains its own set of processing blocks, which means that any arbitrary combination of processing blocks can be achieved as long as it is not deeper than two levels.

Paths and blocks creation and edition

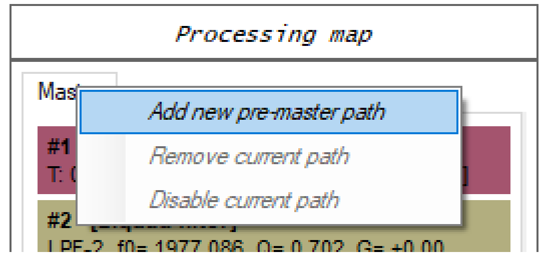
To create blocks in any path, right-click on the path box and choose any of the block types inside the sub-menu “Add”.



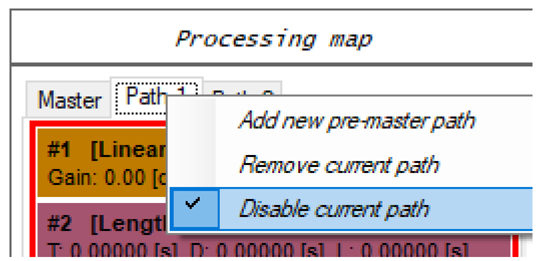
The blocks can be removed and moved up or down simply by selecting them and using the context menu that appears upon a right-click. Double-clicking a block toggles its enabled / bypassed state.

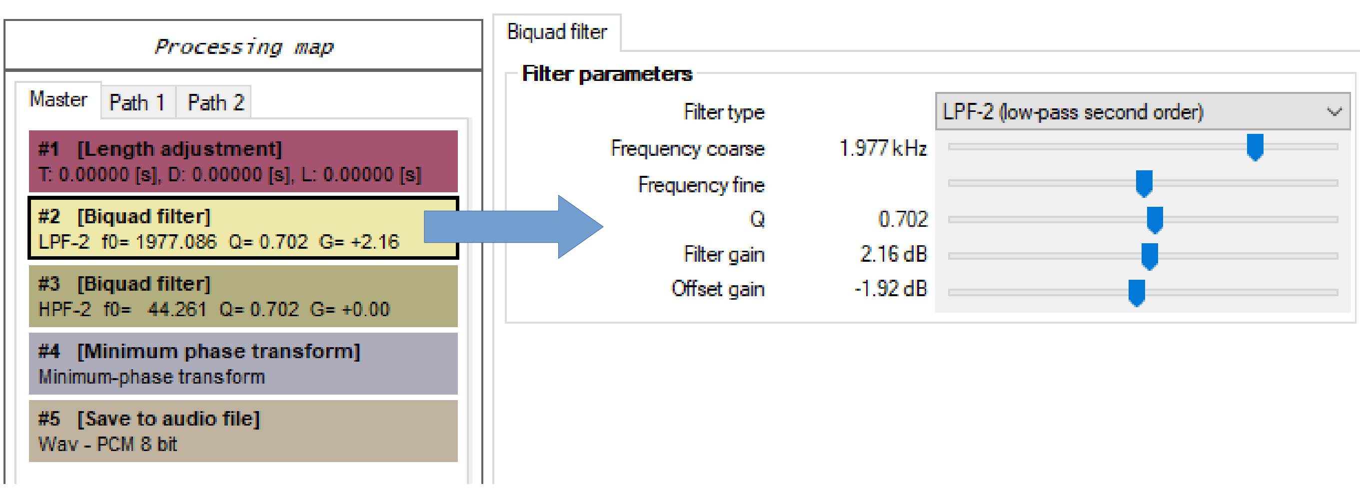


New paths can be creating by using the context menu that appears upon a right-click on any of the path tabs. A blank path will still pass the input signal directly to the master path.



Any pre-master path can be temporarily disabled. When disabled, no signal is passing through that path.





When a block is selected, its parameters will be displayed on the right side.

Types of processing blocks

The following types of blocks exist in the processor:

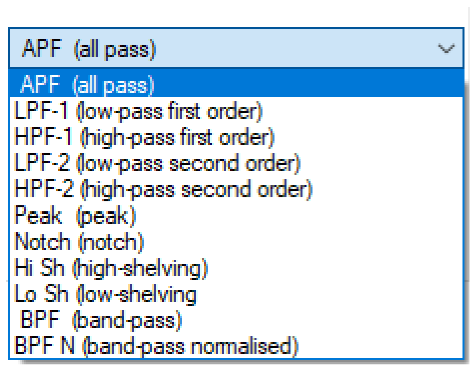
- **Length adjustment**

  Shortens (truncate) or extends (zero-padding) the input signal.

- **Biquad filter**

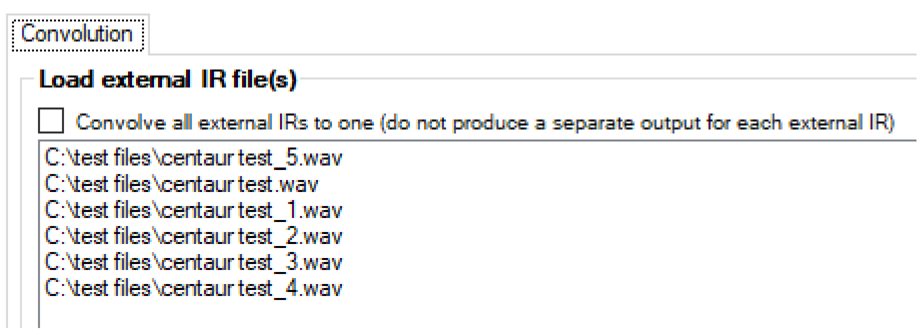
  Applies a first- or second-order filter of the standard design types:

  - All-pass
  - Low-pass first-order
  - High-pass first-order
  - Low-pass second-order
  - High-pass second-order
  - Peak
  - Notch
  - High-shelving second-order
  - Low-shelving second-order
  - Band-pass
  - Band-pass normalized



- **Convolution**

  Convolves the input signal with one or more external files, which can be loaded by dragging and dropping them on the block box. It can be configured to produce a new output per external file (which will multiply the total amount of outputs) or to pre-convolve all external files into one.



- **Fade-in / out**

  Applies a fade-in and or fade-out to the signal, designed in the same manner as the manual windowing of the recorder. This block has additional options to introduce the fade length in samples, in addition to doing so in seconds. See section "*Test signal composition*".

- **Minimum-phase transform**

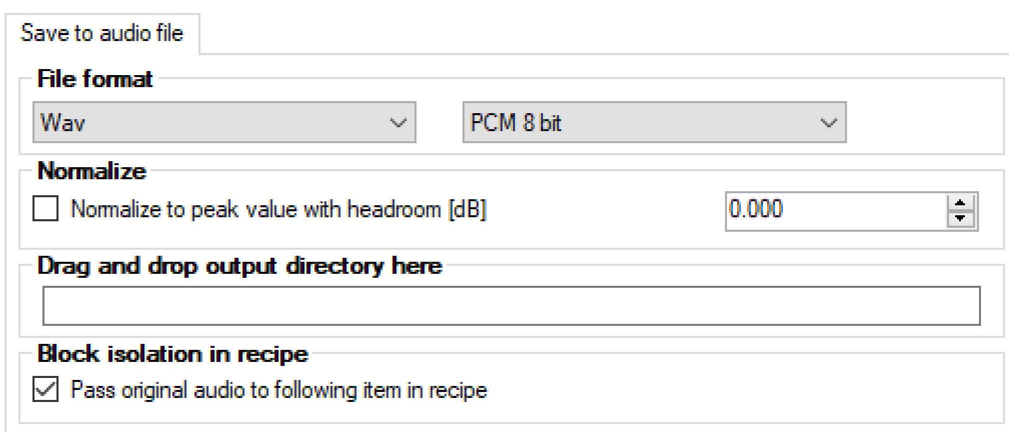
  Applies a minimum-phase transform to the signal, which results in the loss of the original phase information. There are no parameters for this block.

- **Linear gain**

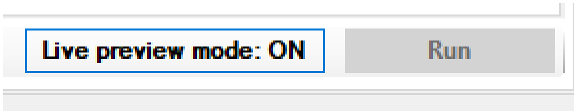
  Applies a gain to the signal, with a value set in decibels, in the range [-100, +100].

- **Save to audio file**

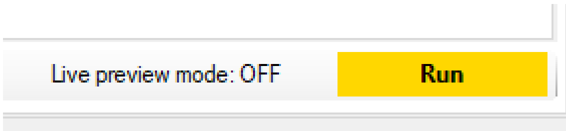
  Saves the incoming signal(s) to audio files, in either of the supported formats (plain text or WAV). It allows for the normalization of the signal before being stored, which can be needed when exporting to WAV since these files are generated with fixed-point sample format. To preview the effect of the data conversion, the option "Pass original audio to following item in recipe" can be unchecked.



Live mode

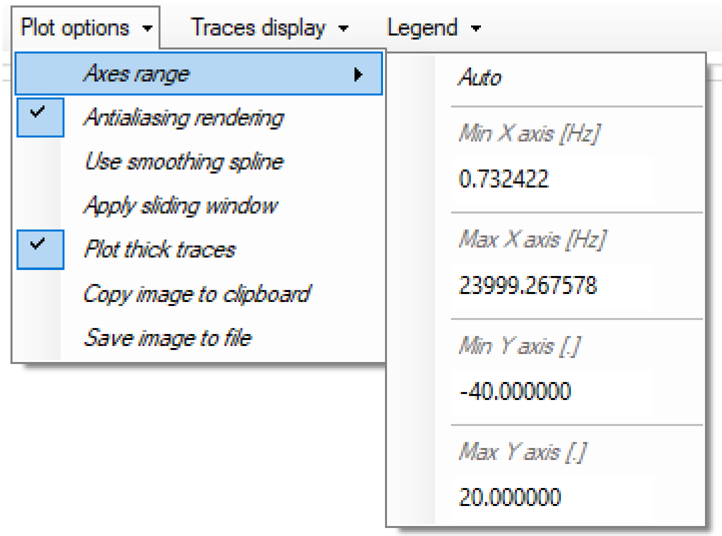


By default, the processor is set to operate in “Live mode”, which means that any changes to the processing chain result in the automatic and immediate application to all selected input files.



When processing a large number of files, this can become very slow, since for any change, fIReLab must re-process all selected input files. In such case, select only a few files which can be used as reference, design the needed processing, then disable the “Live mode”, select all input files and click “Run”.

Plots options



Most of the options in the plot menus are self explainatory. Still, there are a few items that deserve additional information.

- **Antialias rendering**

  Enables a softening of the otherwise pixel-based drawing, for a smoother visualization. This does not distort the displayed plot traces.

- **Use smoothing spline**

  Applies a polynomial spline to “fill-in the gaps”, in the style of a simple interpolator. It can distort the displayed plot traces in some cases, so use with caution. This is useful, for example, when visualizing a signal with a frequency that approaches the nyquist limit.

- **Apply sliding window** (frequency domain only)

  Passes the trace through a sliding window with a progressively larger size. Specially effective to see the overall trend of a spectrum reducing the local ripple. This will always distort the displayed plot traces.

- **Plot thick traces**

  Uses a thicker pen to draw the trace, which besides making it larger, will exagerate any small imperfections or noise that is otherwise harder to observe.