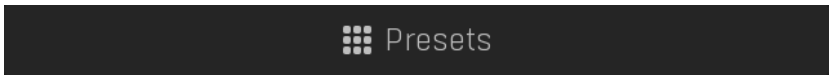
MTurboFilter



Presets

Presets button shows a window with all available presets. A preset can be loaded from the preset window by double-clicking on it, selecting via the buttons or by using your keyboard. You can also manage the directory structure, store new presets, replace existing ones etc. Presets are global, so a preset saved from one project, can easily be used in another. The arrow buttons next to the preset button can be used to switch between presets easily.

Holding **Ctrl** while pressing the button loads a random preset. There must be some presets for this feature to work of course.

Presets can be backed up by 3 different methods:
A) Using "Backup" and "Restore" buttons in each preset window, which produces a single archive of all presets on the computer.
B) Using "Export/Import" buttons, which export a single folder of presets for one plugin.
C) By saving the actual preset files, which are found in the following directories (not recommended):
Windows: C:\Users\{username}\AppData\Roaming\MeldaProduction
Mac OS X: /Library/Application support/MeldaProduction

Files are named based on the name of the plugin like this: "{pluginname}.presets", so for example MAutopan.presets or MDynamics.presets. If the directory cannot be found on your computer for some reason, you can just search for the particular file.

Please note that prior to version 16 a different format was used and the naming was "{pluginname}presets.xml". *The plugin also supports an online preset exchange. If the computer is connected to the internet, the plugin connects to our server once a week, submits your presets and downloads new ones if available. This feature is manually maintained in order to remove generally unusable presets, so it may take some time before any submitted presets become available. This feature relies on each user so we strongly advise that any submitted presets be named and organised in the same way as the factory presets, otherwise they will be removed.*

Left arrow

Left arrow button loads the previous preset.

Right arrow

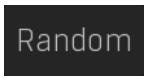
Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.



Randomize

Randomize button (with the text 'Random') generates random settings. Generally, randomization in plugins works by selecting random values for all parameters, but rarely achieves satisfactory results, as the more parameters that change the more likely one will cause an unwanted effect. Our plugins employ a smart randomization engine that learns which settings are suitable for randomization (using the existing presets) and so is much more likely to create successful changes.

In addition, there are some mouse modifiers that assist this process. The smart randomization engine is used by default if no modifier keys are held.

Holding **Ctrl** while clicking the button constrains the randomization engine so that parameters are only modified slightly rather than completely randomized. This is suitable to create small variations of existing interesting settings.

Holding **Alt** while clicking the button will force the engine to use full randomization, which sets random values for all reasonable automatable parameters. This can often result in "extreme" settings. Please note that some parameters cannot be randomized this way.

Panic

Panic button resets the plugin state. You can use it to force the plugin to report latency to the host again and to avoid any audio problems.
For example, some plugins, having a look-ahead feature, report the size of the look-ahead delay as latency, but it is inconvenient to do that every time the look-ahead changes as it usually causes the playback to stop. After you tweak the latency to the correct value, just click this button to sync the track in time with the others, minimizing phasing artifacts caused by the look-ahead delay mixing with undelayed audio signals in your host. It may also be necessary to restart playback in your host.
Another example is if some malfunctioning plugin generates extremely high values for the input of this plugin. A potential filter may start generating very high values as well and as a result the playback will stop. You can just click this button to reset the plugin and the playback will start again.

Settings

Settings button shows a menu with additional settings of the plugin. Here is a brief description of the separate items.

Licence manager lets you activate/deactivate the plugins and manage subscriptions. While you can simply drag & drop a licence file onto the plugin, in some cases there may be a faster way. For instance, you can enter your user account name and password and the plugin will do all the activating for you.

There are 4 groups of settings, each section has its own detailed help information: **GUI & Style** enables you to pick the GUI style for the plugin and the main colours used for the background, the title bars of the windows and panels, the text and graphs area and the highlighting (used for enabled buttons, sliders, knobs etc).

Advanced settings configures several processing options for the plugin.

Global system settings contains some settings for all MeldaProduction plugins. Once you change any of them, restart your DAW if needed, and it will affect all MeldaProduction plugins.

Dry/Wet affects determines, for Multiband plugins, which multiband parameters are affected by the Global dry/wet control.

Smart interpolation adjusts the interpolation algorithm used when changing parameter values; the higher the setting the higher the audio quality and the lower the chance of zippering noise, but more CPU will be used.

WWW

WWW button shows a menu with additional information about the plugin. You can check for updates, get easy access to support,

MeldaProduction web page, video tutorials, Facebook/Twitter/YouTube channels and more.

Sleep indicator

Sleep indicator informs whether the plugin is currently active or in sleep mode. The plugin can automatically switch itself off to save CPU, when there is no input signal and the plugin knows it cannot produce any signal on its own and it generally makes sense. You can disable this in Settings / **Intelligent sleep on silence** both for individual instances and globally for all plugins on the system.

Plugin toolbar

Plugin toolbar provides some global features, A-H presets and more.

Oversampling

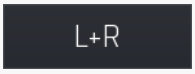
Oversampling can potentially improve sound quality by processing at a higher sample rate. Processors such as compressors, saturators, distortions etc., which employ nonlinear processing generate higher harmonics of the existing frequencies. If these frequencies exceed the Nyquist rate, which equals half of the sampling rate, they get mirrored back under the Nyquist rate. This is known as aliasing and is almost always considered an artifact. This is because the mirrored frequencies are no longer harmonic and sound as digital noise as this effect does not physically occur in nature. Oversampling reduces the problem by temporarily increasing the sampling rate. This moves the Nyquist frequency which in turn, diminishes the level of the aliased harmonics. Note that the point of oversampling is not to remove harmonics, we usually add them intentionally to make the signal richer, but to reduce or attenuate the harmonics with frequencies so high, that they just cannot be represented within the sampling rate.

To understand aliasing, try this experiment: Set the sampling rate in your host to 44100 Hz. Open MOscillator and select a "rectangle" or "full saw" waveform. These simple waveforms have lots of harmonics and without oversampling even they become highly aliased. Now select 16x oversampling and listen to the difference. If you again select 1x oversampling, you can hear that the audio signal gets extensively "dirty". If you use an analyzer (MAnalyzer or MEqualizer for example), you will clearly see how, without oversampling, the plugin generates lots of inharmonic frequencies, some of them which are even below the fundamental frequency. Here is another, very extreme example to demonstrate the result of aliasing. Choose a "sine" shape and activate 16x oversampling. Now use a distortion or some saturation to process the signal. It is very probable that you will be able to hear (or at least see in the analyzer) the aliased frequencies.

The plugin implements a high-quality oversampling algorithm, which essentially works like this: First the audio material is upsampled to a higher sampling rate using a very complicated filter. It is then processed by the plugin. Further filtering is performed in order to remove any frequencies above the Nyquist rate to prevent aliasing from occurring, and then the audio gets downsampled to the original sampling rate.

Oversampling also has several disadvantages of which you should be aware before you start using it. Firstly, upsampled processing induces latency (at least in high-quality mode, although you can select low-quality directly in this popup), which is not very usable in real time applications. Secondly, oversampling also takes much more CPU power, due to both the processing being performed at a higher sampling rate (for 16x oversampling at 44100 Hz, this equates to 706 kHz!), and the complex filtering. Finally, and most importantly, oversampling creates some artifacts of its own and for some algorithms processing at higher sampling rates can actually lower the audio quality, or at least change the sound character. Your ears should always be the final judge.

As always, use this feature ONLY if you can actually hear the difference. It is a common misconception that oversampling is a miraculous cure all that makes your audio sound better. That is absolutely not the case. Ideally, you should work in a higher sampling rate (96kHz is almost always enough), while limiting the use of oversampling to some heavily distorting processors.



Channel mode

Channel mode button shows the current processing channel mode, e.g. **Left+Right (L+R)** indicates the processing of left and right channels. This is the default mode for mono and stereo audio material and effectively processes the incoming signal

as expected. However the plugin also provides additional modes, of which you may take advantage as described below. Mastering this feature will give you unbelievable options for controlling the stereo field.

Note that this is not relevant for mono audio tracks, because the host supplies only one input and output channel.

Left (L) mode and Right (R) mode allow the plugin to process just one channel, only the left or only the right. This feature has a number of simple uses. Equalizing only one channel allows you to fix spectral inconsistencies, when mids are lower in one channel for example. A kind of stereo expander can be produced by equalizing each side differently. Stereo expansion could also be produced by using a modulation effect, such as a vibrato or flanger, on one of these channels. Note however that the results would not be fully mono compatible.

Left and right channels can be processed separately with different settings, by creating two instances of the plugin in series, one set to 'L' mode and the other to 'R' mode. The instance in 'L' mode will not touch the right channel and vice versa. This approach is perfectly safe and is even advantageous, as both sides can be configured completely independently with both settings visible next to each other.

Mid (M) mode allows the plugin to process the so-called mid (or mono) signal. Any stereo signal can be transformed from left and right, to mid and side, and back again, with minimal CPU usage and no loss of audio quality. The mid channel contains the mono sum (or centre), which is the signal present in both left and right channels (in phase). The side channel contains the difference between the left and right channels, which is the "stereo" part. In 'M mode' the plugin performs the conversion into mid and side channels, processes mid, leaves side intact and converts the results back into the left and right channels expected by the host.

To understand what a mid signal is, consider using a simple gain feature, available in many plugins. Setting the plugin to M mode and decreasing gain, will actually lower or attenuate the mono content and the signal will appear "wider". There must be some stereo content present, this will not work for monophonic audio material placed in stereo tracks of course. Similarly amplifying the mono content by increasing the gain, will make the mono content dominant and the stereo image will become "narrower".

As well as a simple gain control there are various creative uses for this channel mode.
Using a **compressor** on the mid channel can widen the stereo image, because in louder parts the mid part gets attenuated and the stereo becomes more prominent. This is a good trick to make the listener focus on an instrument whenever it is louder, because a wider stereo image makes the listener feel that the origin of the sound is closer to, or even around them.
A **reverb** on the mid part makes the room appear thin and distant. It is a good way to make the track wide due to the existing stereo content, yet spacey and centered at the same time. Note that since this effect does not occur naturally, the result may sound artificial on its own, however it may help you fit a dominant track into a mix.
An **equalizer** gives many possibilities - for example, the removal of frequencies that are colliding with those on another track. By processing only the mid channel you can keep the problematic frequencies in the stereo channel. This way it is possible to actually fit both tracks into the same part of the spectrum - one occupying the mid (centre) part of the signal, physically appearing further away from the listener, the other occupying the side part of the signal, appearing closer to the listener.
Using various **modulation effects** can vary the mid signal, to make the stereo signal less correlated. This creates a wider stereo image and makes the audio appear closer to the listener.

Side (S) mode is complementary to M mode, and allows processing of only the side (stereo) part of the signal leaving the mid intact. The same techniques as described for M mode can also be applied here, giving the opposite results.
Using a **gain** control with positive gain will increase the width of the stereo image.
A **compressor** can attenuate the side part in louder sections making it more monophonic and centered, placing the origin a little further away and in front of the listener.
A **reverb** may extend the stereo width and provide some natural space without affecting the mid content. This creates an interesting side-effect - the reverb gets completely cancelled out when played on a monophonic device (on a mono radio for example). With stereo processing you have much more space to place different sounds in the mix. However when the audio is played on a monophonic system it becomes too crowded, because what was originally in two channels is now in just one and mono has a very limited capability for 2D placement. Therefore getting rid of the reverb in mono may be advantageous, because it frees some space for other instruments.
An **equalizer** can amplify some frequencies in the stereo content making them more apparent and since they psycho acoustically become closer to the listener, the listener will be focused on them. Conversely, frequencies can be removed to free space for other instruments in stereo.
A **saturator / exciter** may make the stereo richer and more appealing by creating higher harmonics without affecting the mid channel, which could otherwise become crowded.
Modulation effects can achieve the same results as in mid mode, but this will vary a lot depending on the effect and the audio material. It can be used in a wide variety of creative ways.

Mid+Side (M+S) lets the plugin process both mid and side channels together using the same settings. In many cases there is no difference to L+R mode, but there are exceptions.
A **reverb** applied in M+S mode will result in minimal changes to the width of the stereo field (unless it is true-stereo, in which case mid will affect side and vice versa), it can be used therefore, to add depth without altering the width.
A **compressor** in M+S mode can be a little harder to understand. It basically stabilizes the levels of the mid and side channels. When channel linking is disabled in the compressor, you can expect some variations in the sound field, because the compressor will attenuate the louder channel (usually the mid), changing the stereo width depending on the audio level. When channel linking is enabled, a compressor will usually react similarly to the L+R channel mode.
Exciters or saturators are both nonlinear processors, their outputs depend on the level of the input, so the dominant

channel (usually mid) will be saturated more. This will usually make the stereo image slightly thinner and can be used as a creative effect.

How to modify mid and side with different settings? The answer is the same as for the L and R channels. Use two instances of the plugin one after another, one in M mode, the other in S mode. The instance in M mode will not change the side channel and vice versa.

Left+Right(neg) (L+R-) mode is the same as L+R mode, but the the right channel's phase will be inverted. This may come in handy if the L and R channels seem out of phase. When used on a normal track, it will force the channels out of phase. This may sound like an extreme stereo expansion, but is usually extremely fatiguing on the ears. It is also not mono compatible - on a mono device the track will probably become almost silent. Therefore be advised to use this only if the channels are actually out of phase or if you have some creative intent.

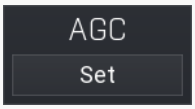
There are also 4 subsidiary modes: **Left & zero Right (L(R0))**, **Right & zero Left (R(L0))**, **Mid & zero Side (M(S0))** and **Side & zero Mid (S(M0))**. Each of these processes one channel and silences the other.

Surround mode is not related to stereo processing but lets the plugin process up to 8 channels, depending on how many the host supplies. For VST2 plugins you have to first activate surround processing using the **Activate surround** item in the bottom. This is a global switch for all MeldaProduction plugins, which configures them to report 8in-8out capabilities to the host, on loading. It is disabled by default, because some hosts have trouble dealing with such plugins. After activation, restart your host to start using the surround capabilities of the plugins. Deactivation is done in the same way. Please note that all input and output busses will be multi-channel, that includes side-chain for example. For VST3/AU/AAX plugins the activation is not necessary.

First place the plugin on a surround track - a track that has more than 2 channels. Then select **Surround** from the plugin's Channel Mode menu. The plugins will regard this mode as a natural extension of 2 channel processing. For example, a compressor will process each channel separately or measure the level by combining the levels of all of the inputs provided. Further surround processing properties, to enable/disable each channel or adjust its level, can be accessed via the **Surround settings** in the menu.

Ambisonics mode provides support for the modern 3D systems (mostly cinema and VR) with up to 64 channels (ambisonics 7th order). Support for this is still quite rare among the DAWs, so this needs to be activated in all DAWs using the **Activate ambisonics** item in the bottom. This is a global switch for all MeldaProduction plugins, which configures them to report 64in-64out capabilities to the host, on loading. After activation, restart your host to start using the ambisonics capabilities of the plugins. Deactivation is done in the same way. Please note that all input and output busses will be multi-channel, that includes side-chain for example.

First place the plugin on an ambisonics track, supported are all orders from 1st (4 channels) to 7th (64 channels). Then select **Ambisonics** from the plugin's Channel Mode menu. Finally select the **Ambisonics settings** in the menu and configure the Ambisonics order and other settings if needed. The plugins will regard this mode as a natural extension of 2 channel processing. For example, a compressor will process each channel separately or measure the level by combining the levels of all of the inputs provided.



AGC

AGC button enables or disables the automatic gain control - the automatic adjustment of the output volume such that it matches the input volume. Human hearing is very adaptable. In fact differences in loudness, for example when loading a preset, may go unnoticed and instead be perceived by the listener as "better sounding", leading to a misjudgement. This feature should prevent this effect, thus allowing the listener to focus on the sonic qualities only.

AGC works by measuring input and output loudness, and then compensating for the difference while also taking into account any induced latency. The loudness measurement follows the ITU and EBU specifications with an RMS of 400ms, meaning that the reaction time is 400ms. This is very important, as you should be aware that AGC needs time to properly adjust after any change of settings. Also note that this is a nonlinear operation. It may cause some distortion due to the long measurement time. It should be negligible though.

AGC makes sense in most applications including reverberation and equalization for example. However, in some cases it can work against the plugin. A simple example of this is a tremolo, where the plugin manipulates output volume. If the tremolo rate is slow enough, say 1Hz, it makes the period longer than the actual AGC measurement time. So whenever the tremolo changes audio level, the AGC starts compensating for it. This can of course be used creatively, since AGC will always be a little "late", but it is definitely not a desired outcome in normal use.

Another example of this is compression. When used with short attack and release times, AGC can effectively compensate for the attenuation of the compressor. However when the attack and release times are higher than 100ms, the compressor's reaction time becomes too slow, and in conjunction with AGC, severe pumping can occur.

As a general rule of thumb as for all audio processing tasks, use it only if you know you need it. AGC is a powerful tool that can make your workflow easier, but it can also be damaging.

Set

Set button uses the AGC (automatic gain compensation) processor to calculate the ideal output gain to ensure that the output audio loudness is equal to the input level. To use it, simply enable playback in your host and click the button. The plugin's output gain will be adjusted to match the input and output levels as closely as possible.
If the AGC is already enabled, the change will be instant and you can disable the AGC afterwards. Typically you will browse presets, generate random settings etc. During the entire time you will have AGC enabled to prevent you from experiencing different output loudness levels. When you find a sonically ideal setup, you simply click the Set button to set the output gain automatically and disable the AGC as you won't need it anymore.
If the AGC is not already enabled, clicking the Set button displays a window with progress bar for a few seconds, while the plugin temporarily enables AGC and analyses input and output of the plugin. After that the AGC is disabled again.

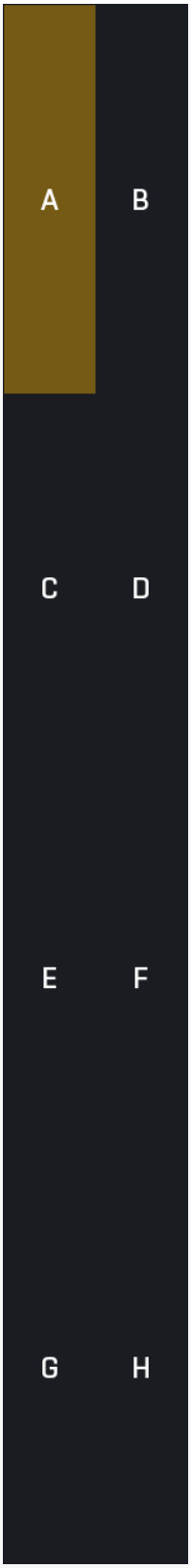
To get the best results, you should feed the plugin with some "universal" signal. If you are processing a specific instrument, play a typical part, a chorus in case of vocals for example. If you are creating presets designed for general use, white/pink noise may be the best signal to use.

Limiter

Limiter button enables or disables the safety limiter. Its purpose is to protect you from peaks above 0dB, which can have damaging effects to your processing chain, your monitors and even your hearing.

It is generally advised to keep your audio below 0dB at all times in all stages of your processing chain. However, several plugins may cause high level outputs with certain settings, often due to unprevented resonances with specific audio materials. The safety limiter prevents that.

Note that it is NOT wise to enable this "just in case". As with any processing, the limiter requires additional processing power and modifies the output signal. It is a transparent single-band brickwall limiter, but you still need to be careful when using it.



A-H presets selector

A-H presets selector controls the current A-H preset. This allows the plugin to store up to 8 sets of settings, including those parameters that cannot be automated or modulated. However it does not include channel mode, oversampling and potentially some other global controls available from the Settings/Settings menu.

For example, this feature can be used to keep multiple settings, when you are not sure about the ideal configuration When you change any parameter, only the currently selected preset is modified.

The four buttons below enable you to switch between the last 2 selected sets using the A/B button, morph between the first 4 sets using the morphing button and copy & paste settings from one preset to another (via the clipboard).

It is also possible to switch between the presets using MIDI program change messages sent from your host. The set selected depends on the Program Change number: 0 selects A, 7 selects H, 8 selects A, 15 selects H and so on.

A/B

A/B button switches between the active and previously active A-H preset (not necessarily the A and B presets themselves). To compare any 2 of the A-H presets, select one and then the other. Clicking this button will then switch between these two. You can do the same thing by clicking on the particular presets, but this makes it easier, letting you close your eyes and just listen.

Morph

Morph button lets you morph between the A, B, C and D settings. Morphing only affects those parameters that can be automated or modulated; that does include most of the parameters however. When you click this button, an X/Y graph is shown allowing you to drag the position indicator to any position between the letters A, B, C and D. The closer you drag the indicator to one of the letters, the closer the actual settings are to that preset.

Please note that this will overwrite and change the preset that is currently selected, so it is best to select a new preset e.g. 'E', then use the morphing method. This way you will define the settings for A, B,C and D, morph between them, and store the result in 'E' without any modification of the original A, B, C and D presets.

Please note that the ABCD morphing itself cannot be automated and that, while morphing, the changes to the underlying parameters are not notified to the host (there may be hundreds of change events).

Copy

Copy button copies the current settings to the system clipboard.
Hold **Ctrl** to save the settings as a file instead. That may be necessary for complex settings, which may be too long for system clipboard to handle. It may also be advantageous when you want to send the settings via email. You can load the settings by drag & dropping them to a plugin or holding **Ctrl** and clicking **Paste**.

Paste

Paste button pastes settings from the system clipboard into the current preset. Hold **Ctrl** to load the settings from a file instead. Hold **Shift** to paste the settings to all of the A-H slots at once.

Undo

Undo button reverts the last change. Only changes to automatable or modulatable parameters and global settings (load/randomize) are stored.

Redo

Redo button reverts the last undo operation.

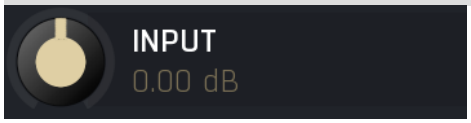
WAV

WAV button lets you process a file using the plugin with current settings. You can either click the button and select a file, or drag & drop the file (or multiple files) onto the button. If you let the plugin process WAV files, these will be saved with the original settings. If you use a different file type (such as MP3), the plugin will create WAV files with 32-bit bits-per-sample floating point.

Please note that the files will be overwritten, so make a copy first if you want to keep the original.

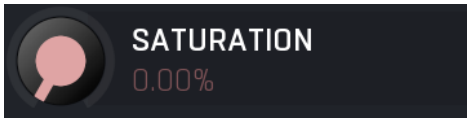
Collapse

Collapse button minimizes or enlarges the panel to release space for other editors.



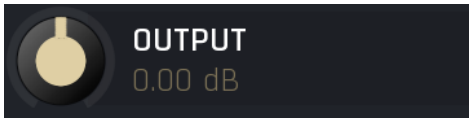
Input gain

Input gain defines input gain. It can be used to make the input level reaching say -20dBFS to 0dBFS, which is usually optimal for various 'nalog' style processing.
Range: -24.00 dB to +24.00 dB, default 0.00 dB



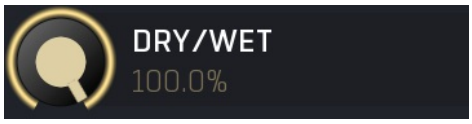
Saturation

Saturation controls the amount of saturation performed on the output audio.Saturation can smooth out the result adding some higher harmonics and can generally increase the overall loudness.
Range: 0.00% to 100.0%, default 0.00%



Output gain

Output gain defines output gain applied on the processed signal. It is useful to make the output loudness similar to the input loudness, so that **Dry/Wet** doesn't trick you with loudness differences. Note that after this output gain control there's a safety clipper at +20dB.
Range: -24.00 dB to +24.00 dB, default 0.00 dB



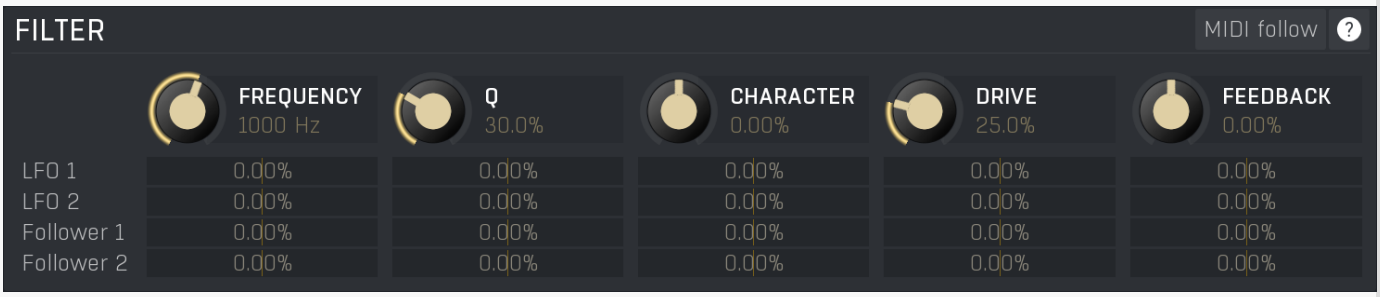
Dry/Wet

Dry/Wet defines ratio between dry and wet signals. 100% means fully processed, 0% means no processing at all. Please note that values in between may causes some phase shifting effects.
Range: 0.00% to 100.0%, default 100.0%

Tab selector

Tab selector switches between subsections.

Filter panel



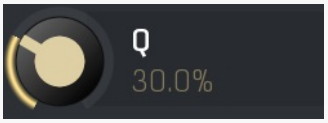
Filter panel contains the filter parameters. There are 4 main parameters, frequency (as F), Q, character (as C) and analog (as A). Selected filtering algorithm may or may not use all of them, so it is possible that some of them will not have any effect.

MIDI follow

MIDI follow button shows MIDI follow settings, which you can use to make the filter listen to input MIDI notes and adjust its frequency accordingly.

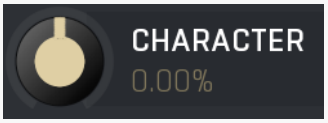
Frequency

Frequency defines filter frequency. For classic lowpass filter for example this controls the cutoff frequency, but for more complex filters the meaning can be completely different. Frequency should be used in all filter algorithms.
Range: 20.00 Hz to 20.0 kHz, default 1000 Hz



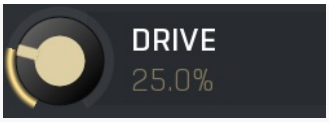
Q

Q defines resonance of the filter. Resonance should be used in most filter algorithms. By default it controls Q (translated into 0.05 to 100) of Pass/Shelf/Peak filters and dry/wet of comb filters.
Range: 0.00% to 100.0%, default 30.0%



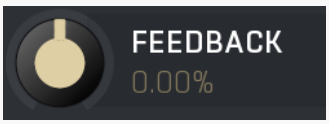
Character

Character defines an additional parameter used by the filter. Its meaning will differ depending on the filter algorithm and it may be ignored as well. By default it controls gain of Peak/Shelf filters and feedback of Comb filters.
Range: -100.0% to 100.0%, default 0.00%



Drive

Drive defines drive of the filter. By default it controls dry/wet ratio of saturators, but you can use it as additional parameter in any filter.
Range: 0.00% to 100.0%, default 25.0%



Feedback

Feedback controls the single-sample feedback used by the feedback module. Note that engaging it will considerably increase CPU consumption. Also note that in some cases feedback may sound different depending on the sampling rate.
Range: -100.0% to 100.0%, default 0.00%

Frequency

Frequency controls how much can the LFO change the filter frequency.
Range: -100.0% to 100.0%, default 0.00%

Q

Q controls how much can the LFO change the filter resonance.
Range: -100.0% to 100.0%, default 0.00%

Character

Character controls how much can the LFO change the filter character parameter.
Range: -100.0% to 100.0%, default 0.00%

Drive

Drive controls how much can the LFO change the filter drive parameter.
Range: -100.0% to 100.0%, default 0.00%

Feedback

Feedback controls how much can the LFO change the filter feedback parameter.
Range: -100.0% to 100.0%, default 0.00%

Frequency

Frequency controls how much can the follower change the filter frequency.
Range: -100.0% to 100.0%, default 0.00%

Q

Q controls how much can the follower change the filter resonance.
Range: -100.0% to 100.0%, default 0.00%

Character

Character controls how much can the follower change the filter character parameter.
Range: -100.0% to 100.0%, default 0.00%

Drive

Drive controls how much can the follower change the filter drive parameter.
Range: -100.0% to 100.0%, default 0.00%

Feedback

Feedback controls how much can the follower change the filter feedback parameter.
Range: -100.0% to 100.0%, default 0.00%

Tab selector

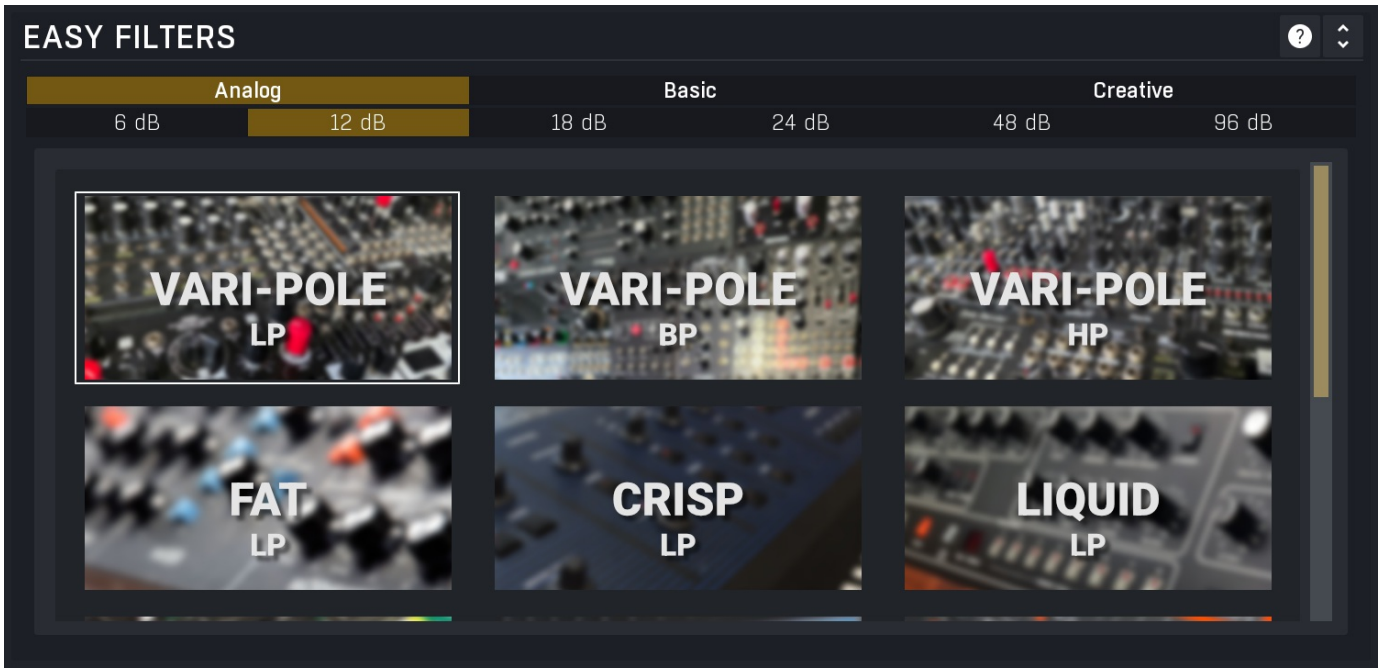
Tab selector switches between subsections.

Randomize

Randomize button generates random settings for the tab.

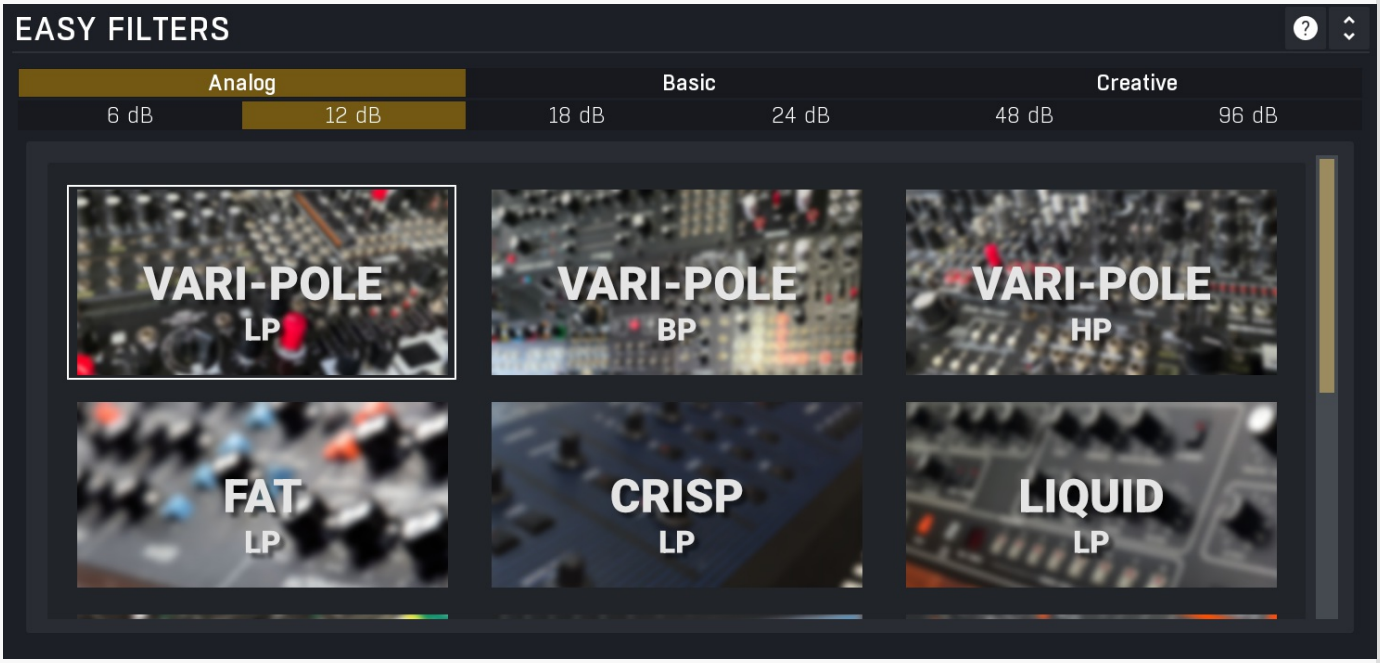
Presets

Presets button chooses a random preset for the tab.



Filters panel
Filters panel contains the predefined analog filters.

Frequency



ANALOG FILTERS

VARI-POLE LP, BP, and HP. A generic analog model with variable slope.

Frequency controls the filter cutoff.
Q controls the resonance.
Character controls the slope. -100% = 1 pole (6dB). 0% = 2 pole (12dB). +100% = 4 pole (24dB).
Drive controls the amount of saturation inside the filters.
Feedback controls the amount of positive and negative (phase inverted) feedback.

FAT, CRISP, LIQUID, SMOOTH, SOLID, BRIGHT, ACID, RAW, and BRUTAL. Virtual analog low-pass filters modelled on famous classic filters from vintage synthesisers.
Frequency controls the filter cutoff from approx. 20Hz to 20kHz.
Q controls the filter resonance.
Character controls the resonance compensation. Many classic filters lose low-end level when increasing resonance. With Character at -100 % you get the classic response; 0% has full level compensation, and +100% has increased bass response.
Drive controls the level going into the filter, resulting in overdrive distortion. It also often has the effect of smoothing out resonance. Gain staging plays an important role in analog synthesizers, as it can completely change the character of the filter. Using Input and Output can help here.
Feedback controls the amount of the filter's output that is fed back into its input. This can range from subtle fatness to full-on screaming. Negative values invert the phase of the feedback signal.

A note on analog filter emulation: While every effort has been made to meticulously model filter characteristics, it is worth considering that they will never be an exact match to anything. Synthesizers changed the type of filter chips in various revisions, and even matching chips could sound different from each other. Then there is the rest of the synthesizer that will play a huge role in the sound character of the filter, such as the oscillators, mixer gain, VCA, etc. Therefore, rather than trying to faithfully recreate any specific synthesizer's filter, we advise simply choosing the character that sounds best for your sound. For this reason, we added the ability to quickly switch between the models to audition them in context.

BASIC FILTERS

All filters in the Basic section are available in 6dB (1 pole), 12dB (2 pole), 18dB (3 pole), 24dB (4 pole), 48dB (8 pole), and 96dB (16 pole) slope variations.

VARI-RES LP. A low-pass with adjustable resonance shape.

Frequency controls the filter cutoff.
Q controls the amount of resonance.
Character controls the shape of the resonant peak.
Drive controls the amount of pre-filter saturation.
Feedback controls a frequency offset of the resonant peak.

VARI-NOTCH LP. A low-pass with an adjustable notch.

Frequency controls the filter cutoff and notch frequency.
Q controls the amount of resonance.
Character controls a frequency offset of the notch.
Drive controls the amount of pre-filter saturation.
Feedback controls the resonance of the notch.

GATE LP. A "West Coast" style filter (LPG) often used for percussive sounds. The VCA is linked to the filter cutoff, resulting in a natural decay.

Frequency controls the filter cutoff and amplitude level (20Hz = silence).
Q controls the amount of resonance.
Character controls the transfer curve for the level.
Drive controls the amount of pre-filter saturation.
Feedback controls the amount of positive and negative (phase inverted) feedback.

TWIN-PEAKS LP, BP, and HP. Dual parallel filters with adjustable frequency offset.

Frequency controls the filter cutoff for both peaks.
Q controls the amount of resonance for both peaks.
Character controls a frequency offset of the second peak.
Drive controls the amount of pre-filter saturation.
Feedback controls a tilt balance between the two peaks.

MORPH LP-BP-HP. A multi-mode filter that morphs between states.

Frequency controls the filter cutoff.
Q controls the amount of resonance.
Character controls the morph between LP, BP, and HP.
Drive controls the amount of pre-filter saturation.
Feedback controls a tilt balance between the low and high frequencies.

X-FADE LP-BP-HP. A multi-mode filter that crossfades between states.

Frequency controls the filter cutoff.
Q controls the amount of resonance.
Character controls the crossfade between LP, BP, and HP.
Drive controls the amount of pre-filter saturation.
Feedback controls the amount of positive and negative (phase inverted) feedback.

VARI-WIDTH BP. A band-pass filter with adjustable width.

Frequency controls the filter cutoff.
Q controls the amount of resonance.
Character controls the band-width.
Drive controls the amount of pre-filter saturation.
Feedback controls the amount of positive and negative (phase inverted) feedback.

CREATIVE FILTERS

LO-FI LP. A low-pass filter with sample rate and bit reduction.

Frequency controls the frequency of the low-pass filter and sample rate.
Q controls the amount of resonance.
Character controls a frequency offset of the sample rate.
Drive controls the amount of bit-crushing.
Feedback controls the pre/post filter balance of the bit-crusher.
VOWEL A-E-I-O-U. A multi-peak filter that mimics vocal vowel sounds.

Frequency controls the frequency of the peaks.
Q controls the resonance of the peaks.
Character controls the vowel morphing.
Drive controls the level of the peaks.
Feedback controls a comb filter, morphing from human to robot.

PHASE LP. A low-pass filter with a series of all-pass filters.

Frequency controls the frequency of all filters.
Q controls the resonance of the low-pass filter.
Character controls a frequency offset of the all-pass filters.
Drive controls the resonance of the all-pass filters.
Feedback controls the number of all-pass filters, up to 16.

SELF-MOD LP. A low-pass filter with filter FM.

Frequency controls the frequency of the filter and modulator.
Q controls the resonance of the low-pass filter.
Character controls a frequency offset of the modulator.
Drive controls the modulation depth.
Feedback controls the amount of positive and negative (phase inverted) feedback.

DISTO LP. A heavily distorted low-pass filter with wave rectification.

Frequency controls the frequency of the low-pass filter.
Q controls the resonance of the low-pass filter.
Character controls the waveshaper.
Drive controls the amount of distortion.
Feedback controls the amount of positive and negative (phase inverted) feedback.

COMB LP. A low-pass filter with a comb filter.

Frequency controls the frequency of the low-pass and comb filters.
Q controls the resonance of the low-pass and comb filters.
Character controls a frequency offset of the comb filter.
Drive controls the mix of a second offset comb filter.
Feedback controls the feedback of the comb filter.

RIPPLE LP. A low-pass filter with a series of peak filters.

Frequency controls the frequency of all filters.

Q controls the resonance of the low-pass and peak filters.
Character controls a frequency offset of the peak filters.
Drive controls the amount of pre-filter saturation.
Feedback controls inverting of odd- or even-numbered peaks.

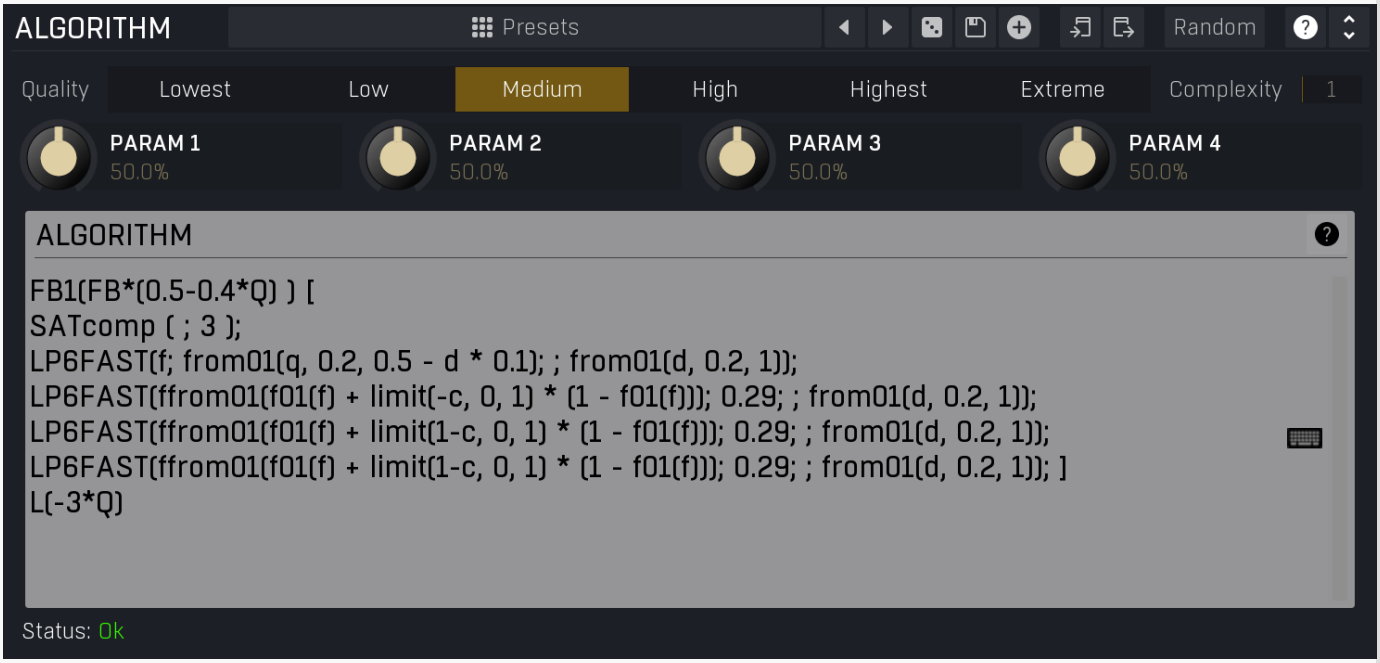
SHIFT-UP LP. A low-pass filter with a positive frequency shifter.

Frequency controls the frequency of the low-pass filter and shifter.
Q controls the resonance of the low-pass filter.
Character controls a frequency offset of the shifter.
Drive controls the mix of the shifter.
Feedback controls the feedback for the shifter.

SHIFT-UP LP. A low-pass filter with a negative frequency shifter.

Frequency controls the frequency of the low-pass filter and shifter.
Q controls the resonance of the low-pass filter.
Character controls a frequency offset of the shifter.
Drive controls the mix of the shifter.
Feedback controls the feedback for the shifter.

Algorithm panel



Algorithm panel controls the actual filtering algorithm and controls how the plugin actually sounds. Note that for safety reasons the plugin performs clipping at +20dB located after the output gain.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.

Random

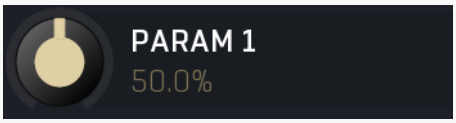
Random button generates random settings using the existing presets.

Quality

Quality controls how often the filter coefficients are recalculated. Since the plugin is not supposed to be a static equalizer, the filter parameters are supposed to change in time, depending on the LFOs and followers for example. However in many cases updating the filter coefficients is very expensive in terms of CPU, so this parameters lets the filters recalculate every 'n' samples. **Extreme** mode makes the filters initialized every single sample and therefore it requires the most CPU power.

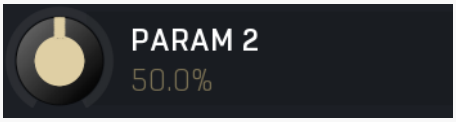
Complexity

Complexity controls the order of the filter and may or may not be used in the filter using the # term.
Range: 1 to 10, default 1



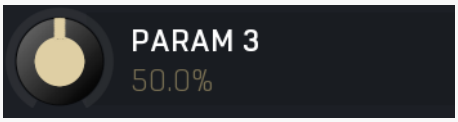
Param 1

Param 1 controls a custom parameter useful inside the algorithm.
Range: 0.00% to 100.0%, default 50.0%



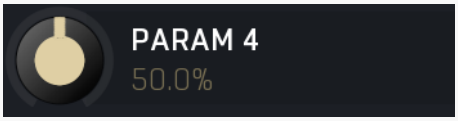
Param 2

Param 2 controls a custom parameter useful inside the algorithm.
Range: 0.00% to 100.0%, default 50.0%



Param 3

Param 3 controls a custom parameter useful inside the algorithm.
Range: 0.00% to 100.0%, default 50.0%



Param 4

Param 4 controls a custom parameter useful inside the algorithm.
Range: 0.00% to 100.0%, default 50.0%

ALGORITHM

```
FB1(FB*(0.5-0.4*Q) ) [
SATcomp ( ; 3 );
LP6FAST(f; from01(q, 0.2, 0.5 - d * 0.1); ; from01(d, 0.2, 1));
LP6FAST(ffrom01(f01(f) + limit(-c, 0, 1) * (1 - f01(f))); 0.29; ; from01(d, 0.2, 1));
LP6FAST(ffrom01(f01(f) + limit(1-c, 0, 1) * (1 - f01(f))); 0.29; ; from01(d, 0.2, 1));
LP6FAST(ffrom01(f01(f) + limit(1-c, 0, 1) * (1 - f01(f))); 0.29; ; from01(d, 0.2, 1)); ]
L(-3*Q)
```

Algorithm

Algorithm controls how the filter actually works and you can program it completely using a single formula exploiting the many modules that the software provides. If you have reached this stage, it probably means that you are ready to get seriously creative and this is exactly what it is all about.

There are pretty much infinite ways to build a filter. But most filters can be (and usually are) defined by a set of smaller filters with variously changing parameters. If you take a simple lowpass filter, there are many ways this thing can work. It can change Q based on frequency for example. With multiple stages of such a basic lowpass filter things can get very complex.

While providing a completely visual interface for this modularity may be easier to understand, the complexity may actually be worse. So we came up with a simple single-statement programming language the algorithm is entirely constructed from. To make things easier to use, the algorithm can contain multiple lines, but newlines have no meaning.

The algorithm controls what modules are used and how they are connected and the parameters for each module are generated using the filter parameters and can be redefined using standard mathematical formulas. All the hardcore mathematics is hidden inside the engine and now the only thing for you to do is to experiment, use your ears, and be creative.

Introduction

Before you start programming your own filter algorithms it is highly recommended to check out the presets. You will learn a lot from a close inspection of them. Also you should enable the safety limiter button (on the right-hand side of the plugin), since infinite feedback is quite common when dealing resonant filters. In most cases the engine will compensate for everything, but sometimes it just cannot be done.

First let's simplify the reading by defining that "f" is the filter Frequency parameter, "q" is the filter Q parameter, "c" is the filter Character parameter, "d" is filter Drive parameter and "fb" is the filter Feedback parameter.

Now a little example, so that you can see how easy it actually is:

*FB1[PARALLEL[LOWPASS;COMB] ; HIGHPASS(f*0.5;q*0.5)]* = a 12dB/oct lowpass filter in parallel with a comb filter, both with frequency defined by "f" and Q by "q". After that, the output is processed using a high-pass filter with frequency "f/2" and Q "q/2", so both are halved. In the formula "*0.5" was used instead of the division, because generally division is far more CPU intensive. The whole thing is in a feedback loop with 1 sample feedback controlled by the "fb" parameter.

Syntax

The syntax defining the topology consists of several modules, placed one by one and separated by a semicolon ';'. Most modules can have parameters, placed in round brackets '(parameter1;parameter2;...)', and some may have submodules, placed in square brackets '[submodule1;submodule2;...]'.

In front of each module you can specify the number of occurrences, 10 is the maximum for safety reasons. The number only serves as a shortcut, so that you don't need to specify the same module multiple times, for example 3lowpass is exactly the same thing as LOWPASS;LOWPASS;LOWPASS. Instead of a number you can use a hashtag '#', which will be replaced by the value of the **Complexity** parameter. This way the user can change the filter algorithm without editing the algorithm formula itself at all, by changing the **Complexity** value.

You can insert spaces and newlines anywhere; these have no meaning, but they can make the algorithm easier to look at. You can also mix uppercase and lowercase letters. Most modules have a short name and a long name, it doesn't matter which you use.

To summarise, the syntax looks like this (the elements in italics are optional):
count/# **MODULE** *(parameter1;parameter2;...) [submodule1;submodule2;...] ;*

For complex algorithms you may use C++ style comments : any text within */* comment */* will be ignored.

User parameters and expressions

While combining modules can lead to various results, the basic filters would all follow the **f** (frequency), **q** (resonance), **c** (character) and **d** (drive) parameters, which means that for example all lowpasses would be the same. The actual fun part is to combine and control their parameters as well. Additionally following parameters can be used:
time = time in seconds from the beginning of the processing. This can be used to create custom modulations.
channel = audio channel, 0 = left, 1 = right... This can be used to create stereo filter algorithms.

channels = total number of audio channels.
samplerate = current sampling rate.
p1, p2, p3, p4 = custom parameters of whatever meaning.

Each parameter of each module has a default, which will be used when you do not specify that parameter. So for example you can write:

LOWPASS(;q+0.1)

Here the first parameter (frequency) is not specified, so it defaults to "f". Q is specified and is the "Q" plus 0.1. In the documentation of each module below there is a "=" present for each parameter defining its default value.

Expression evaluator uses traditional C/C++ style formating, which is natural for most people. It provides arithmetics, logical and conditional operators. Following terms are supported:
Constants: **pi**, **e**, **sqrt2**, **ln2**

Arithmetic operators:
-a inverts the sign, e.g. "-x" produces +2 for x=-2
a+b = addition
a-b = subtraction
a*b = multiplication
a/b = division
a%b = modulo, remainder after division
a^b = power, e.g. "2^3" produces 2*2*2 = 8

Arithmetic functions:
min(a,b) = minimum of both values
max(a,b) = maximum of both values
limit(a,min,max) = a limited into the interval min..max
to01(a,min,max) = converts "a" as min..max to 0..1
from01(a,min,max) = converts "a" as 0..1 to min..max
tom11(a,min,max) = converts "a" as min..max to -1..1
fromm11(a,min,max) = converts "a" as -1..1 to min..max

Basic mathematic functions: **abs(x)** = absolute value, e.g. abs(-3) = 3 **sqr(x)** = x*x **sqrt(x)** = square root **exp(x)** = natural exponential e^x **ln(x)** = natural logarithm **log10(x)** = logarithm with base 10 **log(x, base)** = logarithm with specified base **inv(x)** = 1/x **sgn(x)** = sign of x, -1 or 0 or +1 depending on x*x **round(x)** = rounding to the nearest value **floor(x)** = rounding to the nearest lower value, e.g. floor(-2.3) = -3 **ceil(x)** = rounding to the nearest higher value, e.g. ceil(-2.3) = -2 **rand(x)** = random value from 0 to x

Functions for specific units:
f01(a) = converts "a" as frequency from 20..20000 into log scale 0..1
ffrom01(a) = converts "a" as 0..1 (log scale) to frequency from 20..20000
ffrom01limit(a) = same, but makes sure the output is in 20..20000 range
f01range(a,min,max) = converts "a" as frequency from min to max into log scale 0..1
ffrom01range(a,min,max) = converts "a" as 0..1 (log scale) to frequency from min to max
ffrom01rangelimit(a,min,max) = same, but makes sure the output is in min..max range
todb(a) = converts "a" as multiplier to dB value by calculating "20*log10(a)"
fromdb(a) = converts "a" as dB value to multiplier by calculating "10^(a/20)"

Trigonometric functions: **sin(x)**, **asin(x)**, **cos(x)**, **acos(x)**, **tan(x)**, **atan(x)**, **sinh(x)**, **cosh(x)**, **tanh(x)**

Logical operators:
a==b = comparison producing 1 if "a" and "b" are equal, 0 otherwise
a!=b = comparison producing 1 if "a" and "b" are NOT equal, 0 otherwise
a<b = comparison producing 1 if "a" is lower than "b", 0 otherwise
a<=b = comparison producing 1 if "a" is lower or equal to "b", 0 otherwise
a>b = comparison producing 1 if "a" is greater than "b", 0 otherwise
a>=b = comparison producing 1 if "a" is greater or equal to "b", 0 otherwise
!a = logical negation, 0 produces 1, 0 otherwise
a&&b = logical AND, produces 1 if both "a" and "b" are nonzero
a||b = logical OR, produces 1 if any of "a" and "b" are nonzero
a^^b = logical XOR, produces 1 if "a" and "b" are logically different
a ? b : c = if a is nonzero, then the result is b, otherwise it is c}}}

Time for a few examples of some handy expressions:
*f * (2^c)* = frequency increased/decreased based on the character parameter by an amount of 1 octave.
1 - f01(f) = 1 for 20Hz, then decreasing to 0 for 20kHz. f01 is useful to change other parameters with frequency.
to01(f, 100, 1000) = frequency converted to 0 for frequencies 100Hz and below, and 1 for 1kHz and above.
limit(f, 100, 1000) = frequency trimmed to 100Hz .. 1kHz.
ffrom01(q) = Q interpretted as frequency from 20Hz to 20kHz.

*q + c * sin(time*2*pi)* = Q modulated by 1Hz sine wave, modulation amount controlled by character.

Helper modules

The first set of modules doesn't do anything directly to the sound, but these are necessary for managing other modules.

S / **SERIAL** [submodule1;submodule2;...]
Executes submodule1, submodule2 etc. in series, one by one. Since this is a basic module it is the default and you can omit its name. Thus (*[4lowpass]* is the same as *s[4lowpass]*).

Example: *[2LOWPASS;COMB]* = 2 lowpasses followed by the comb, so the output = lp1->lp2->comb

P / **PARALLEL** [submodule1;submodule2;...]
Executes submodule1, submodule2 etc. in parallel and mixes their outputs together.

Example: *P[2LOWPASS;COMB]* = 2 lowpasses followed by the comb, each receiving the dry audio signal, so the output = lp1+lp2+comb

L / **LEVEL** *(gain in dB = 0)*
Level module performs a simple gain. When there is no gain specified, it does nothing, literally. That way it can be used as a dummy module which is sometimes useful for more complex algorithms.

Example: *P[L;4COMB]* = a level module and 4 comb filters in parallel, each receiving the incoming signal itself, and mixed together.

Example: *L(-3)* = a simple -3dB gain.

LC / **LEVELC** *(gain as multiplier = 1)*
LevelC does exactly the same job as Level, but its parameter is specified as a multiplier instead of a gain in decibels. So the module simply multiplies the input signal by the specified value.

CLIP
A simple clipper, which doesn't let anything above 0dBFS pass. That is often quite useful, especially when working with feedback.

Example: *LOWPASS; CLIP* = A low-pass filter followed by a clipper.

Basic filters

These are the basic building blocks of all filter design.

LP / **LOWPASS** *(frequency = f; q = q; gain = c*20; analog = 0; feedback = 0) [feedback, fb1, fb2]*
LP6 / **LOWPASS6** *...*
LP6b / **LOWPASS6B** *...*
HP / **HIGHPASS** *...*
HP6 / **HIGHPASS6** *...*
HP6b / **HIGHPASS6B** *...*
LS / **LOWSHELF** *...*
HS / **HIGHSHELF** *...*
BP / **BANDPASS** *...*
AP / **ALLPASS** *...*
AP6 / **ALLPASS6** *...*
NOTCH *...*
PEAK *...*
PEAKANALOG *...*
Basic filters with self explanatory meanings. Postfix "6" means 6dB/oct slope, 12dB/oct otherwise. Analog parameter controls the amount of analog-style feedback saturation. LP6 and HP6 are bi-linear transform filters. LP6B and HP6B are basic filters with 6dB/oct slope.
Q (resonance) parameter uses range 0..1, but it is translated to range 0.05 to 100 of the classic scientific Q definition. Q is not used by 6dB/oct filters.
Gain controls the amplification and attenuation in decibels and defaults to c*20, making it go from -20dB to +20dB controlled by the character parameter. Gain is not used by lowpass/highpass/allpass filters.
The feedback parameter controls the feedback from the output to input, providing a near-zero-latency feedback feature. The signal can then be postprocessed using feedback module. Note that using different sampling rate will produce different sound. The filters are implemented as classic biquad sections, which have 2 feedback delays. The module lets you process each feedback with other modules (fb1, fb2) creating various complex scenarios, such as a filter inside a filter etc.

Please be very careful when using feedbacks, as these can easily translate to an infinite feedback. That's why if any feedback or module is used, output is clipped to -20dB..+20dB range. Also note that using feedback processing may require much more CPU power.

Example: *LP(f*2) ; HP(f*0.5) ; HIGHSHELF(;;c*10)* = a lowpass an octave above, followed by a highpass octave below, followed by a high-shelf at the center frequency with gain range -10dB to +10dB.

Example: *LOWPASS(;;;; c)[SAT]* = a lowpass with feedback controlled by character and the feedback signal processed using saturation.

Example: *LP [COMPRESSOR; [COMPRESSOR; SATURATION]]* = a lowpass with custom feedback processing - first feedback is processed using a limiter, and then next one with a limiter followed by a saturator.

LPFAST / **LOWPASSFAST** *(frequency = f; q = q; gain = c*20; analog = 0; feedback = 0) [feedback, fb1]*
HPFAST / **HIGHPASSFAST** *...*
LP6FAST / **LOWPASS6FAST** *...*
HP6FAST / **HIGHPASS6FAST** *...*
BPFAST / **BANDPASSFAST** *...*
NOTCHFAST *...*
PEAKFAST *...*
The previous filters struggle when modulated extremely quickly. In these cases these became unstable and can produce extreme gains for example. The fast filters come to rescue. The work very differently (implementing the so-called SVF filtering), but the parameters are actually the same, and when modulated slowly they even behave pretty much the same way. The module syntax is nearly identical to previous biquad filters, but there's only one internal feedback.

Example: *LPFAST(f*2) ; HPFAST(f*0.5) ; PEAKFAST(;;c*10)* = a lowpass an octave above, followed by a highpass octave below, followed by a peak filter at the center frequency with gain range -10dB to +10dB.

Example: *LPFAST(;;;; c)[SAT]* = a lowpass with feedback controlled by character and the feedback signal processed using saturation.

Example: *LPFAST [[COMPRESSOR; SATURATION]]* = a lowpass with custom feedback processing limiter followed by a saturator.

C / **COMB** *(feedback = c; analog = 0; minfrequency = 20; freq1 = f; drywet1 = q; freq2 = f; drywet2 = q; ...)*
A comb filter with as many taps as you need. minfrequency is the minimum frequency each tap can be tuned to, and its total minimum 0.01. freq1 is the frequency of the first tap, drywet1 is the dry/wet of the first tap etc. All taps are executed in parallel and since they share a single delay line, it is more efficient than using a parallel module with multiple combs, but shares feedback value for all taps.

Example: *COMB(1; ; 1; f; ; f*0.8; ; f*0.6)* = A comb filter with feedback always +1 (hence +100%) and 3 taps.

D / **DELAY** *(time = 0.1)*
A single tap delay without feedback with time specified in seconds.

Example: *D(100 + c * 100)* = A delay with time controlled by character going from 0 to 200ms.

RATIO (drywet = 1; ratio = d; mode = 2) [submodule1;submodule2;...]
A Ratio is parallel processing module, which can crossfade between other inserted modules according to set ratio and used crossfade mode. This processor behaves in similar way as MRatio plugin which mixes two audio inputs. The main difference here is, it is crossfading between other submodules and you can add as many submodules you want, not just two. If you insert less than two modules, no fading is performed as it has no sense.

You can control the dry/wet ratio in 0 to 1, crossfade ratio from 0 to 1 and crossfade mode from 0 to 7. Valid modes are the same as in MRatio: 0 = Squared, 1 = Linear, 2 = Constant power, 3 = Transition, 4 = Fade, 5 = Fade smooth, 6 = Cut, 7 = Cut smooth.

Example: *RATIO(1;d;)[LP; BP; HP]* = A Ratio module which has inside three filters (lowpass, bandpass and highpass), ratio can be changed via filter d parameter (DryWet is 1 in this case) and crossfade mode is default Constant power.

Dynamics/complex/nonlinear processing

Following modules are useful for some more creative algorithms.

COMPRESSOR *(threshold = 0; ratio = inf; attack = 0; release = 0.01)*
A compressor with threshold an dB in -200 to +200, ratio in 1 to 10^10, and attack & release in seconds in 0 to 100. Default values specify a brickwall limiter with attack 0ms and release 10ms.

Example: *LOWPASS[COMPRESSOR(-20;1 + c*4;0.001)] ; compressor* = A low-pass filter with a feedback compressor with -20dB threshold, ratio controller by Character parameter from 1:1 to 4:1, and 1ms attack time. The output is then processed using a normal limiter.

SAT / **SATURATION** *(drywet = d; mode = 0; evenharmonics = 0; threshold = silence; inputgain = 0; outputgain = 0)*
A saturation unit based on MSaturator. You can control the dry/wet ratio in 0 to 1, mode from 0 to 8 and even harmonics from 0 to 10, where 1 means +100%. Threshold can bet set from silence(default) up to 0 dB, input and output gain from -48dB to 48dB. Please do NOT use even harmonics in a feedback processing, that would make any filter unstable.

Example: *LOWPASS[; SATURATION(c)]* = A low-pass filter with a secondary feedback line processed by a saturator controlled by the Character parameter.

SATCOMP / **SATURATIONCOMPENSATED** *(drywet = d; mode = 0; evenharmonics = 0)*
Similar to SAT, but it compensates for the loss of volume, similarly to what MSaturator does. Do NOT use this one inside any feedback paths, or you will get an immediate infinite feedback.

COMPANDER *(drywet = 1; output = 0; threshold = -40, ratio = inf., attack = 0.01, release = 0.1)*
[submodule1;submodule2;...]
A compander is a combination of compressor and expander module and it is based on integrated compander in MSaturator. This processor can do custom serial processing (see helper module **Serial**) of sub-modules placed in between compressor and expander. This results first in compression of dry input signal, which maximizes the SNR (signal to noise ratio), then is made processing of inserted sub-modules and finally expansion which reduces the "noise" (can be i.e. noise of analog tape). Resulting wet output signal is then "quieter", but you can compensate it with output parameter. Essentially there are main 2 approaches to compression - A) set the threshold high, so that it affects everything above it, B) set the threshold low and use dry/wet to actually lower the effect of compression, which provides an easy way to control the amount of compression without too much editing of the more advanced parameters. Please note that lowering ratio does NOT have the same effect as lowering dry/wet in most cases.
You can control the dry/wet ratio in 0 to 1, output gain from -20dB to +20dB, threshold from -80dB to 0dB, ratio from 1 to infinity, attack from 0 to 1 sec, release from mode from 0.001 to 10 sec.

Example: *COMPANDER[SAT; LP]* = A compander which has inside in series saturation module followed by low-pass filter, all modules in default settings.

Example: *CDER(;10;-60;;c*2;c*10)[SAT]* = A compander which has inside just suration module in default settings. Compander dry/wet in control by drive parameter, output gain set to +10dB, threshold -50dB, ratio to infinity, attack and release in control by multiplied filter "c" parameter.

CHANNEL [submodule1;submodule2;...]
A channel module is simple multichannel processing module, which lets you run different algorithms in different channels. It lets you do various stereo (or surround...) expansions. First channel is processed with submodule1, 2nd with submodule 2 etc. When it runs out of submodules, it starts repeating the sequence. So for example if you specify 2 modules in surround 7.1 mode, 1st is processed using 1, 2nd with 2, 3rd with 1, 4th with 2 etc.

Example: *CHANNEL[LP; HP]* = Processes the left channel with low pass and right channel with high pass filters.

FB1 / **FEEDBACK1** *(feedback = fb)* [submodule1;submodule2;...]
A 1-sample feedback module. The shortest feedback possible with such generalized modular system. Note that using different sampling rate will produce different sound.

Feedback parameter controls how much of the output signal is fed back to the delay line. If feedback is 0, no signal is sent back and the module becomes identical to **serial**. The amount of feedback can be set in range from -1 (-100%, negative feedback) to +1 (+100%, positive feedback). Engaging this module requires all submodules to switch to 1-sample processing, which requires much more CPU. The feedback is always clipped to 0dBFS for safety reasons.

Example: *FB1[SAT]* = An feedback with saturation in the feedback loop.

FB / **FEEDBACK** *(feedback = fb; delay = 0.1; delaymax = 1; clipper = 1)* [submodule1;submodule2;...]
This module produces a positive audio feedback. Such situation may occur when an acoustic path exists between an audio input (i.e. microphone) and an audio output (i.e. loudspeaker). In this example, a signal received by the microphone is amplified and passed out of the loudspeaker. The sound from the loudspeaker can then be received by the microphone again, amplified further, and then passed out through the loudspeaker again. This creates, usually non wanted audio feedback. Feedback processor can do custom serial processing (see helper module **Serial**) of sub-modules placed inside this effect, which means feedback from all inserted modules is sent back to the input.

Feedback parameter controls how much of the output signal is fed back to the delay line. If feedback is 0, no signal is sent back. The amount of feedback can be set in range from -1 (-100%, negative feedback) to +1 (+100%, positive feedback). Delay parameter controls the actual length of delay. Delay can be set from 0 to 10 seconds (minimum is 1 sample, maximum is DelayMAX). Nice effects are made by very short delays and feedback above 50%.
DelayMAX parameter controls the maximum length of the variable delay line. Valid range is from 0.1 to 10sec. This parameter CANNOT be modulated, it is set only once when setting up the filter processor and later modulations are ignored. You can of

course modulate the delay parameter as you like in range from 0 to MAXDelay.
Clipper parameter is used to limit (clip) the maximal signal output level into valid range, which may be very handy if bigger feedback loop is generated. This paramater also CANNOT be modulated. Possible values are 0 (clipper OFF) or default 1 (ON).

Example: *FEEDBACK[SAT]* = An feedback with saturation in the feedback loop.

Example: *FB(0.5;0.005)[SAT;HYST]* = An interesting effect, where feedback parameter is set to 0.5 (50%), delay time to 5ms. Feedback module has inside in series saturation module followed by hysteresis module, both in default settings. Processes the left channel with low pass and right channel with high pass filters.

FBCOMB / **FEEDBACKCOMB** *(feedback = fb; frequency = f; delaymax = 1; clipper = 1)* [submodule1;submodule2;...]
This module is similar to the previous **feedback** module, but the delay is defined as frequency instead of time. This way it is technically comb filter.

Example: *FBCOMB[SAT]* = A comb filter with saturation in the feedback loop.

EZSHAPER *(drywet = d; interpolation = 0; mode = 0; symmetry = 0; y1 = 0; y2 = 1; y3 = 2...)*
An EZShaper is a waveshaper module allowing you to specify any number of points, the Y (output) values only, the X (input) values are equally spread along the X axis. It is faster and simpler version of shaper, which you can find further in this help. Unlike say MWaveShaper, Y values are not limited to 0..1, which is especially important when using asymmetric wave-shaping.
DryWet controls the module dry/wet ratio in 0 to 1.
Interpolation controls how the values in between the points are interpolated. 0 means linear interpolation, 1 means cubic interpolation. Values in between are fading between these two.
Mode controls the shape of the entire graph and is identical to the mode in the MWaveShaper plugin. You can think of it as another waveshaper on top of your waveshaper. Values as follows: 0 = linear, 1 = square, 2 = cubic, 3 = logarithmic. Usually the lower the number, the lower the CPU consumption is.
Symmetry defines if the graph is symmetric (0) and you are specifying points with X from 0 to +1 and X from 0 to -1 is a mirror image of the same points. Or if it is asymmetric (1) and the points are spread equally over the whole input range from -1 to +1, so the first point is output for input -1 and the last one for input +1. If this is the case, you need to make sure the middle point will be 0, otherwise the shaper will change silence (input 0) into something "not silent". Asymmetric processing can generate even harmonics; hence it is useful to simulate behaviour of analog devices.

Examples:
EZSHAPER(;0;0;0; 0.0; 1.0) - default settings
EZSHAPER(;;;1; -1; c; 1) - asymmetric processing with points -1, c and +1, where "character" controls the middle point being at 0.
EZSHAPER(;q;;; 0;c;1;c;1) - symmetric modulation with 5 points, where "q" controls the linear/cubic ratio and "character" controls the 2 points of the shape.

SHAPER *(drywet = d; mode = 0; symmetry = 0; p1x = 0; p1y = 0; p1type = 1; p1curvature = 0; p2x = 1; p2y = 1; ...)*
Shaper is a waveshaper module implementing the same system as in MWaveShaper plugin. Here you can modulate each point with complex formulas leading to pretty much no limits for your imagination. The number of points not limited and you can control X and Y coordinates, type and curvature. Unlike MWaveShaper, Y values are not limited to 0..1, which is especially important when using asymmetric wave-shaping.
DryWet controls the module dry/wet ratio in 0 to 1.
Mode controls the shape of the entire graph and is identical to the mode in the MWaveShaper plugin. You can think of it as another waveshaper on top of your waveshaper. Values as follows: 0 = linear, 1 = square, 2 = cubic, 3 = logarithmic. Usually the lower the number, the lower the CPU consumption is.
Symmetry defines if the graph is symmetric (0) and you are specifying points with X from 0 to +1 and X from 0 to -1 is a mirror image of the same points. Or if it is asymmetric (1) and you are specifying the whole input range from -1 to +1, which is mapped to X from 0 to +1: X = 0 is input -1, X = 0.5 is input 0 and X = 1 is input +1. Y (output) is not mapped, so it ranges from -1 to +1 as usual. If this is the case, you need to make sure the graph will be 0 at X = 0.5, otherwise the shaper will change silence (input 0) into something "not silent". Asymmetric processing can generate even harmonics; hence it is useful to simulate behaviour of analog devices.
Point settings control the actual points. If you specify less than 2 points, the 1 basic points [0;0] and [1;1] will be added automatically. For each point, X coordinate is supposed to fall into 0..1 for both symmetric and asymmetric processing (see Symmetry above), but it's only a recommendation. However, it is highly recommended for each point to have the X coordinate at least as high as the previous point, otherwise the previous point's X will be used instead. Without this constraint, bad things could happen. Y value has no real limits. Curvature is closely tight to the type and ranges from -1 to +1. Type controls the type of the segment after this point and is again identical to MWaveShaper's engine, so it is highly recommended to check it out. Type values are as follows: 0 = Mode 1, 1 = Mode 2, 2 = Mode 3, 3 = Free 1, 4 = Double 1, 5 = Double 2, 6 = Smooth 1, 7 = Smooth 2, 8 = Stairs, 9 = Pulses, 10 = Saw, 11 = Triangles, 12 = Sines

Examples:
SHAPER(;;; 0;0;9;0.3;) - simple but interesting shaper creating just basic 2 points at [0;0] and [1;1] (auto added) using pulses mode.
*SHAPER(;3;; 0;0;q * 12;c)* - another simple but interesting and brutal shaper with the same points, this time in logarithmic mode, but jumping between all the types and curvatures using "q" and "character" controls, providing opportunities for complex modulation.
SHAPER(;;1; 0;1;;; 1;-1) - simple phase inversion using asymmetric mode.
SHAPER(;;; 0;0;;; 0.2;q;;; 0.4;c;;; 0.6;c-q;;; 0.8;q-c;;; 1;1) - 6 point shaper exploiting "q" and "character" to create more

complex modulations.

HYST / **HYSTERESIS** *(amount = d; ratio = 0; drywet = d; mode = 0; shape1 = 0; shape2 = 6; ratio12 = 0; attack = 0.1; release = 1; rmslength = 0.1; peakhold = 0; 2nd = 0; 2ndbylevel = 0; 2ndratio = 0; 3rd = 0; 3rdbylevel = 0; 3rdratio = 0; 4th = 0; 4thbylevel = 0; 4thratio = 0; 5th = 0; 5thbylevel = 0; 5thratio = 0)*
Hysteresis is another exciting non-linear processing saturator, which produces hysteresis distortion driven by a level follower and your settings. Processor varies its parameters according to the detected level in time and the saturation behaves as "magnetized" to certain input levels. This processor consist of three main modules, Hysteresis, Saturator and Harmonics generator all connected in series and you can enable them individually.

Hysteresis module has following two parameters: amount of hysteresis (you can set any value) and ratio with possible values from -1 to 1. Amount means how much should hysteresis affect input signal. Positive value affects negative phase of the signal as it is typical for analog components. Negative value affects only positive phase. Ratio controls the effect of the level follower - by default the level linearily controls the amount of processing. By using a higher or lower value you change the response to the level - tendency to make either more or less processing. In general however, the higher the level, the higher the impact of the hysteresis will be.
Saturator module blends between 2 saturation curves you know from MSaturator plugin. The higher the level, the more the second curve wins over the first. Dry/Wet ratio controls the amount of processing from 0 to 1 and by default is mapped to filter drive parameter. Mode refers to similar pre/post processing as from the shaper modules : 0 = linear, 1 = square, 2 = cubic, 3 = logarithmic. Shape1 and Shape2 define the two processing shapes ranging from 0 to 8, same as in the **sat** module. Ratio12 works the same as in Hysteresis part and controls the tendency to switch to shape2 based on the detected level.
Attack, release, rmslength and peak hold define the behaviour of the level follower. All are specified in seconds. Attack & Release ranges from 0 to 100 seconds, RMS length and peak hold from 0 to 10 seconds and Peak hold from 0 to 10sec.
Harmonic generator is the last module. It is used for generation of additional 4 harmonics (from 2nd to 5th), which behaves the same way as in MSaturator. Each harmonic has it's static level amount, dynamic one (based on the level) and the ratio controlling the reponse to the level detector, same as before.

Examples:
*HYST(d*10;c;0)* - hysteresis module only, with amount driven by multiplied filter drive parameter, ratio neutral, level follower in default and all other modules OFF.
HYST(0;; d;3;0;6;c) - saturator module only, dry/wet set to filter drive parameter, logarithmic mode, module is fading between Soft1 and Clip3 shapes, with ratio12 controlled by filter "c" parameter. Level follower settings again in default.
HYST(0;; 0;;;;; 0.1;1;0.1;0; -c;1;; c;1;; -c;1;; c;1;) - harmonics generator module only, generation of all harmonics from 2nd to 5th, with values of even harmonics controlled by "-c" and odd values controlled by "c" parameter. Impact of level follower set to 1 (maximum) and harmonic ratios in default.
HYST(1;0; d;0;0;6;0; 0.1;1;0.1;0; 1;0;0; 1;0;0; 1;0;0; 1;0;0) - all three modules enabled with explicitly written default values. Only harmonics generator levels are set to 1, instead of default 0 in order to enable this module too.

BITREDU / **BITREDUCTION** *(drywet = d; bits = 4, width = 0)*
A bit reduction is distortion processor, which can reduce the number of processing bits. The number of bits determines the maximum possible signal-to-noise ratio. This is because the minimum possible noise level is the error caused by the quantization of the signal, usually called quantization noise. This noise level is non-linear and signal-dependent. This processor can therefore generate distortion caused by digital signal quantization.
You can control the dry/wet ratio in 0 to 1 (mapped by default to filter drive parameter), number of bits from 1 to 16, width from -4 to +4.
Number of bits controls the number of bits in each sample, the lower the number, the lower the accuracy and more lo-fi the sound becomes.
Width lets you apply different bit reduction to different channels, producing a stereo expansion effect.

Example: *BITREDU(;q*8;c*2)* = A bit reducer with drywet parameter controlled by filter drive parameter, bits by multiplied filter "q" parameter to generate high quantization noise and width driven by multiplied filter "c" parameter.

SRREDU / **SRREDUCTION** *(drywet = d; sr = f, width = 0)*
A sample rate reduction is distortion processor, which can reduce the audio sampling rate. Sampling is the reduction or conversion of a continuous-time audio signal to a discrete-time (digital) signal. The higher the sample rate, the higher the sampled sound quality. Maximum sampled frequency is equal to 1/2 of sample rate and it is called Nyquist frequency. This processor can reduce the sampling rate of audio input and generate distorting caused by this phenomenon.
You can control the dry/wet ratio in 0 to 1 (mapped by default to filter drive parameter), sample rate (mapped by default to filter frequency parameter) from 20Hz to your current sample rate (i.e 48 000Hz), width from -1 (-100%) to +1 (+100%).
Sample rate (sr) controls the sampling rate to which the audio is converted. The lower the number, the lower the accuracy and more lo-fi the sound becomes.
Width lets you apply different reduction to different channels, producing a stereo expansion effect.

Example: *SRREDU(;;c)* = A sample rate reducer with drywet parameter controlled by filter drive parameter, sample rate (sr) by filter "f" parameter and width driven by filter "c" parameter.

FREQSHIFT *(shift = 100)*
A frequency shifter module with shift specified in Hz ranging from up to -20kHz .. +20kHz.

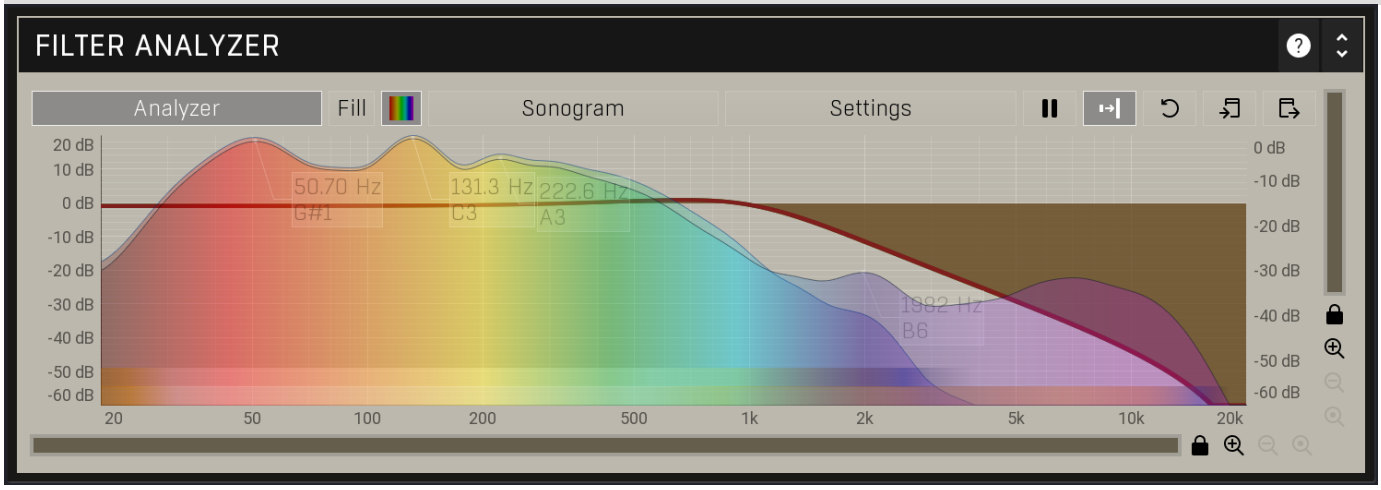
Example: *FREQSHIFT(c * 100)* = Frequency shifter with shift ranging from -100Hz to +100Hz controlled by filter character parameter.

EXPR / **EXPRESSION** *(drywet = 1; expr = x)*
Expression module lets you perform processing based on mathematical expressions. It has the same syntax as you know from expression panel in plugin MUtility and in the expressions described above. The main difference here is, it contains another set of parameters, which are useful for creative filter design, signal generators etc.
Expression defines the mathematical expression to use. The expression is calculated on every sample and the output is limited to a range of -1 ... +1 to avoid problems with invalid formulas. Some hosts do not let keyboard events pass in which case you may need to edit the formula in a text editor and then copy & paste it into the plugin. The available parameters in this module are:

x equals the current input sample.
x1, x2, x3, x4 are the previous input samples.
y1, y2, y3, y4 are the previous output samples.
sr equals the current sampling rate.
n equals the index of current sample starting from the initialization.
f, q, c, d, fb are the filter parameters described above.

Please note that this processing is much slower than using standard components as everything is performed "sample by sample" and also the audio quality may be worse. However it may be useful for specific purposes or testing new creative ideas.

Examples:
Example: *EXPR(; sin(f * 2 * pi * n / sr))* = sine wave generator with target frequency controlled by filter "f" parameter. Dry/wet is by default equal to 1.
Example: *EXPR(; y1 + f / sr * (x - y1))* = simple low pass filter. Dry/wet is by default equal to 1.
Example: *EXPR(d; tanh(x))* = classic input signal saturation. Dry/wet is mapped to filter drive parameter.
Example: *EXPR(; 0.3*x + 0.2*x1 - 0.3*x2 - 0.1*y1 + 0.4*y2)* = special case of biquad filter. Dry/wet is by default equal to 1.



Analyzer view
Analyzer view contains the different forms of analysis that you can display at once. This mainly consists of a powerful spectrum analyzer and sonogram.

Analyzer
Analyzer button enables or disables the spectrum analyzer, which shows the levels of individual frequencies. In most practical cases it is more convenient to use the sonogram, which shows the frequencies in time, but provides a lower level resolution as the levels are differentiated by color. The spectrum analyzer also provides a micro-sonogram (shown in the bottom of the panel) which uses the same color-based view as the sonogram.

Fill
Fill button enables or disables the full-sized analyzer micro-sonogram. This means that the micro-sonogram at the bottom of the equalizer graph will fill the whole analyzer view. Color differentiation is often easier to understand than the classical spectrum analyzer, so this might help you better understand the spectrum of your audio material.

An alternative is to use the spectrum sonogram.

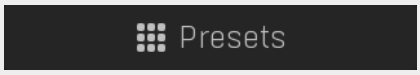
Analyzer Rainbow Colors
Analyzer Rainbow Colors lets you see the analyzed sound spectrum in beautiful colors, following the same style as visible light. It ranges from infra-red colors for the lowest frequencies to ultra-violet colors for the highest frequencies in the analyzed audio. If rainbow colors are disabled, the analyzer and graph will be single-colored, following the setup from Settings/Graphs.

Sonogram

Sonogram button enables or disables the spectrum sonogram, which shows levels of individual frequencies in time. Levels are differentiated by color, so the accuracy is not as good as when using the spectrum analyzer. However, the time axis improves the visual orientation in the spectrum for typical audio signals. In contrast, the spectrum analyzer is more of a scientific tool.

Settings

Settings button shows the settings of the spectrum analyzer and the spectrum sonogram.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

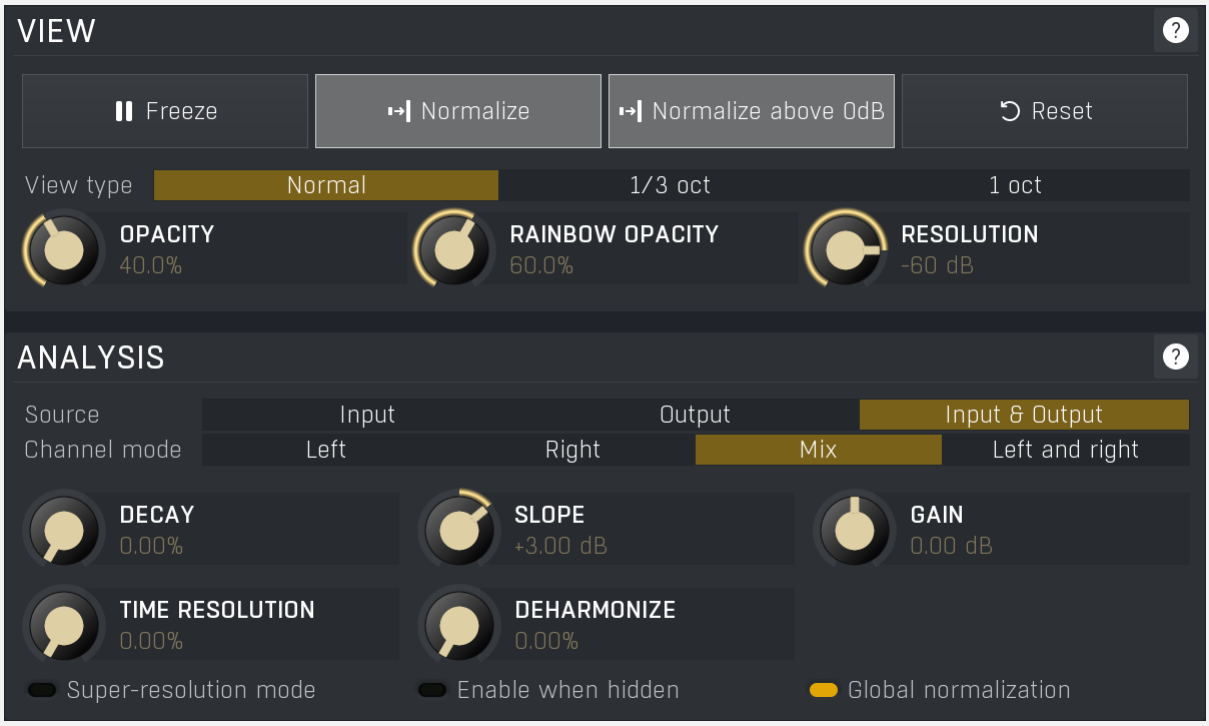
Paste

Paste button loads the settings from the system clipboard.

Tab selector

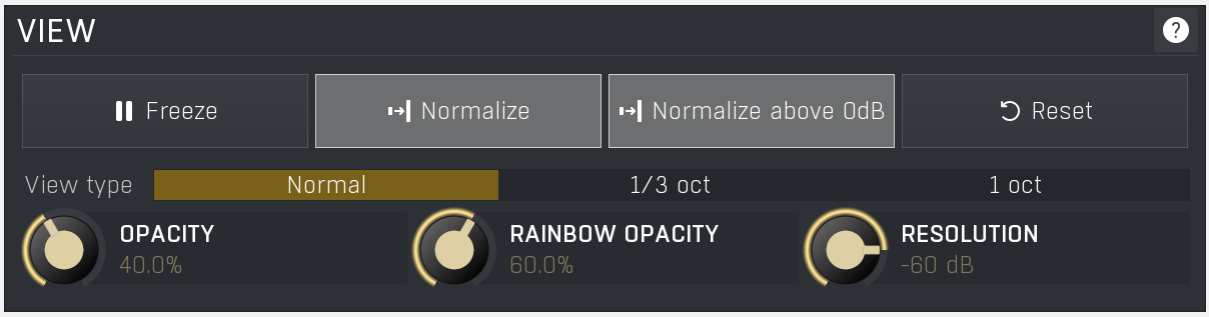
Tab selector switches between subsections.

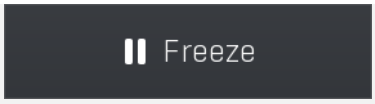
Main settings panel



Main settings panel contains the most useful settings controlling the analyzer behaviour and view.

View



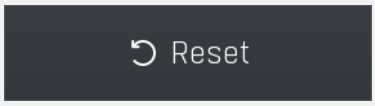


Freeze

Freeze button stops processing temporarily.

Normalize

Normalize button enables or disables the visual normalization, which makes the loudest frequency be displayed at the top of the analyser area (0dB); it does not normalise the sound. This is very useful for comparing frequency levels, however it does hide the actual level.
When comparing 2 spectrums you are usually interested mainly in the frequency level differences. In most cases both audio materials will have different overall levels, which would mean that one of the graphs would be "lower" than the other, making the comparison quite difficult. Normalize fixes this and makes the most prominent frequencies of the spectrum reach the top of the analyzer area (or have the most highlighted color in case of sonogram).

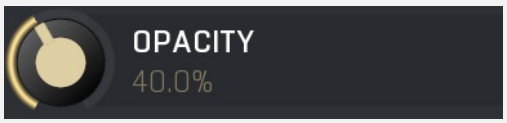


Reset

Reset button resets the analyzer state. This is particularly useful when analyzing infinite average and maximum values.

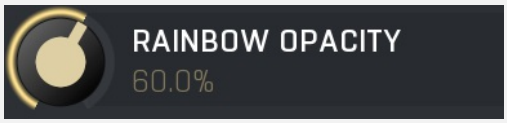
View type

View type controls the way the spectrum is displayed. By default a smooth curve is presented. This view provides the best resolution and detail, but other modes (1/3 octave, 1 octave) may be easier to read.



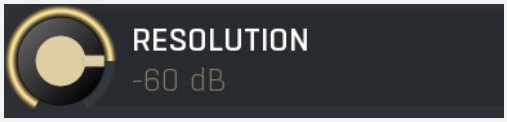
Opacity

Opacity controls the opacity of all analyzer graphs.



Rainbow opacity

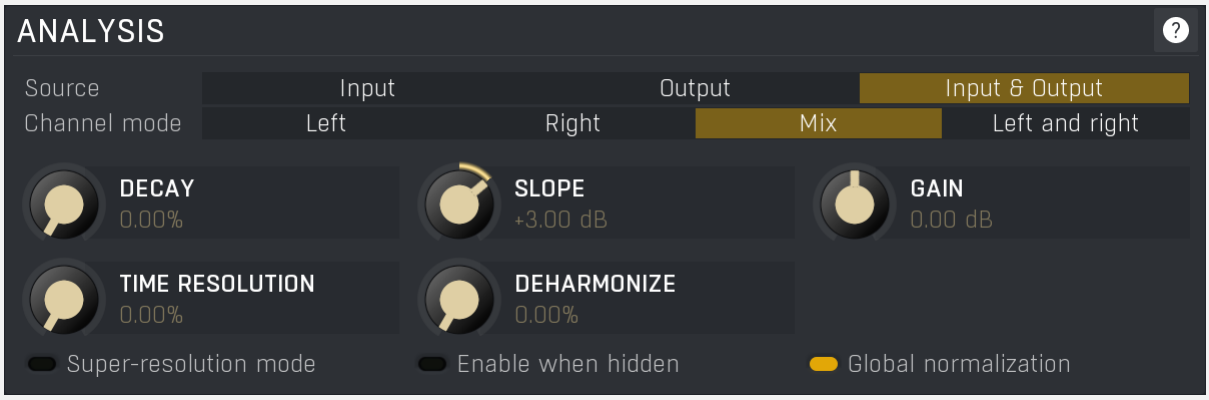
Rainbow opacity controls the opacity of the rainbow graph, if enabled.



Resolution

Resolution defines the vertical range on the display. The human auditory system has a resolution of about 90dB and the relevant range is usually less than 60dB. However you may want to use a higher resolution to check for technical problems - aliasing, distortion etc.

Analysis



Source mode

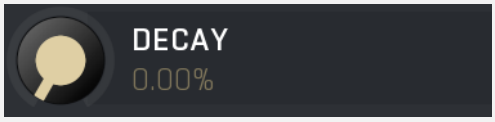
Source mode defines which audio stages are to be analyzed. By default both input & output are selected and analyzed. However you may want to analyze only the input, or the output (or the external side-chain, where available, on its own or with the input or output).

Channel mode

Channel mode defines which channels are to be analyzed. By default all channels are merged into a mono sum (Mix mode), which is then analyzed. However you may want to analyze separate channels or display both the left and right channels separately. Please note that if two channels (for example: input & output, or input & side-chain) are displayed at the same time then mix mode is used instead of left & right mode. Similarly, when the plugin is in Surround mode then Mix mode is used.

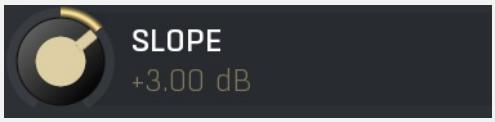
Also please note that when the plugin is in one of the Mid / Side modes of operation, then you should read 'Left' as 'Mid' and 'Right' as 'Side'.

Different analyser combinations can, of course, be saved as different named presets.



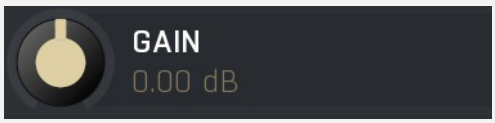
Decay

Decay controls the speed at which the magnitudes return to the minimum value (silence). It is an alternative to averaging, which affects the speed that the frequencies both gain and lose their magnitudes. With a decay of 0% the magnitude goes to the minimum immediately. With 100% it stays the same forever, so it makes it display the maximum.



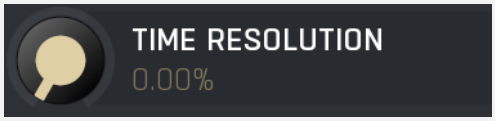
Slope

Slope makes the analyser increase the magnitude of higher frequencies, since they are typically lower in energy. 3dB per octave is a typical value, which makes pink noise horizontal as pink noise contains equal energy in each octave. Therefore if you set slope to 3dB, the response would be the same for the FFT and 1/3 octave graphs.



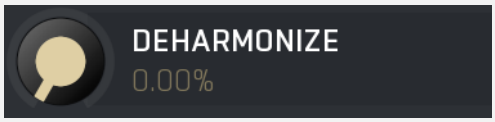
Gain

Gain makes all frequencies change magnitude by the specified amount. This has no meaning when normalization is enabled.



Time resolution

Time resolution improves the time resolution, but lowers the spectral resolution. This is typically useful for more scientific analyses, where the signal is moving quickly and you need to follow its movements quickly. This is often advantageous for sonograms with very high FFT sizes.



Deharmonize

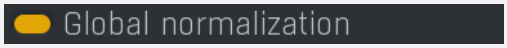
Deharmonize tries to remove harmonics in the content and leave only fundamentals. This may help you find the dominant frequencies in the signal.

Super-resolution mode

Super-resolution mode activates a special processing algorithm, which provides high resolution even in the low frequency spectrum. Using standard FFT algorithms you can increase the FFT size to get better bass resolution, but this also slows down the response. Super-resolution mode keeps the quick response in high frequencies as they are naturally quicker, but also highly enhances the bass spectrum resolution. It requires additional CPU power.

Enable when hidden

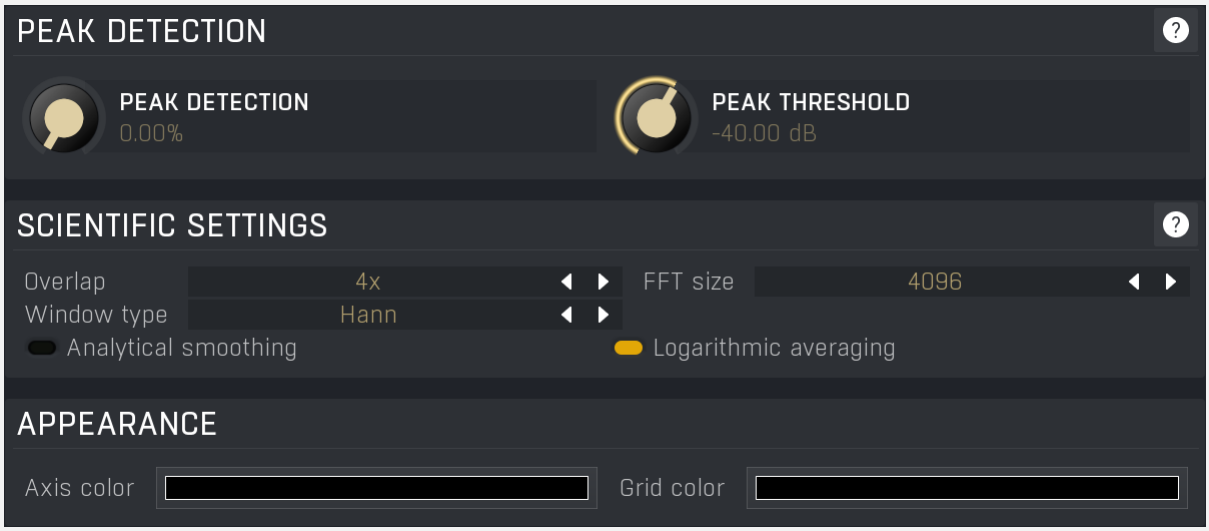
Enable when hidden causes the analysis engine to continue processing the signal even when the GUI is hidden. Otherwise the sonogram is stopped, therefore will not be immediately available when the GUI is shown again.



Global normalization

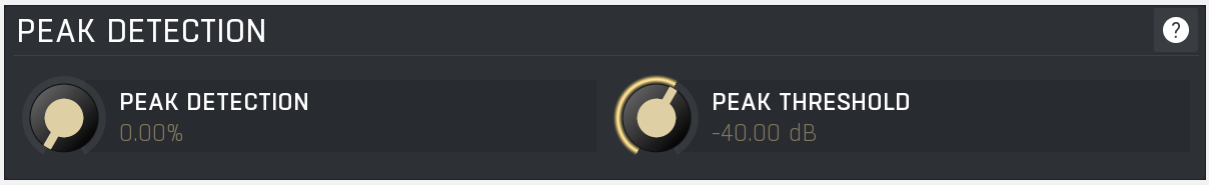
Global normalization makes the **normalization** work based on the maximum of all graphs visible at the time. This means that the levels between the graphs will stay the same, but the maximum level will be 0dB. This is useful for comparing relative levels. If you disable this, all graphs will be normalized separately and will touch 0dB unless they are silent; and this is useful for comparing spectra.

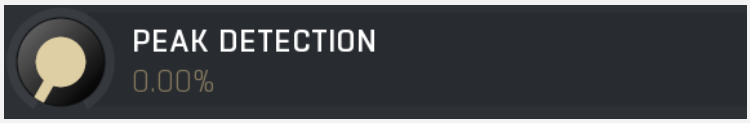
Advanced panel



Advanced panel contains more advanced settings controlling the scientific parameters of the audio analysis.

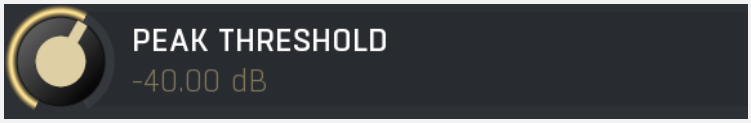
Peak detection





Peak detection

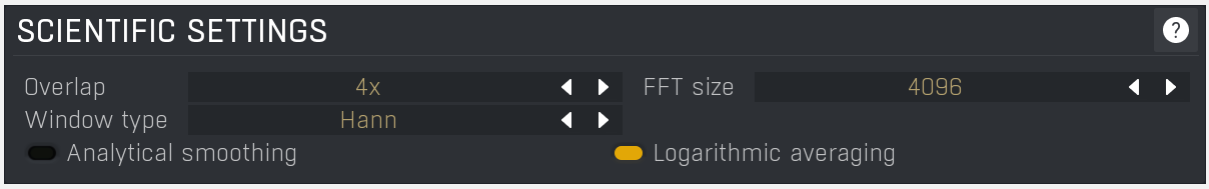
Peak detection tries to the remove skirts of separate sinusoids letting you view the frequencies contained in your audio material. This may be handy when performing more scientific analyses.

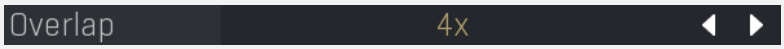


Peak threshold

Peak threshold defines the level below the maximum which is used for peak detection. You can use this to control which peaks get through and to get rid of small insignificant ones.

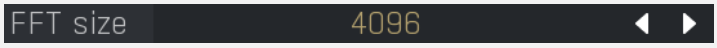
Scientific settings





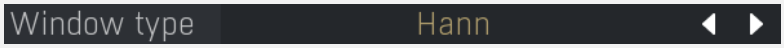
Overlapping

Overlapping makes the analyser perform multiple FFT processing on the same data which results in better precision at the cost of higher CPU impact. With higher overlapping the response also speeds up.



FFT size

FFT size defines FFT processing block size. It basically controls the resolution. However for higher resolution in bass content it is recommended to use super-resolution mode instead as it keeps the quick response in higher frequencies.

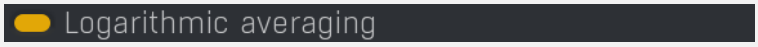


Window type

Window type defines the type of window used to pre-process the source samples. This has several consequences for the frequency response, but it is a little scientific parameter. If you do not have specific requirements you can just leave this set to its default.

Analytical smoothing

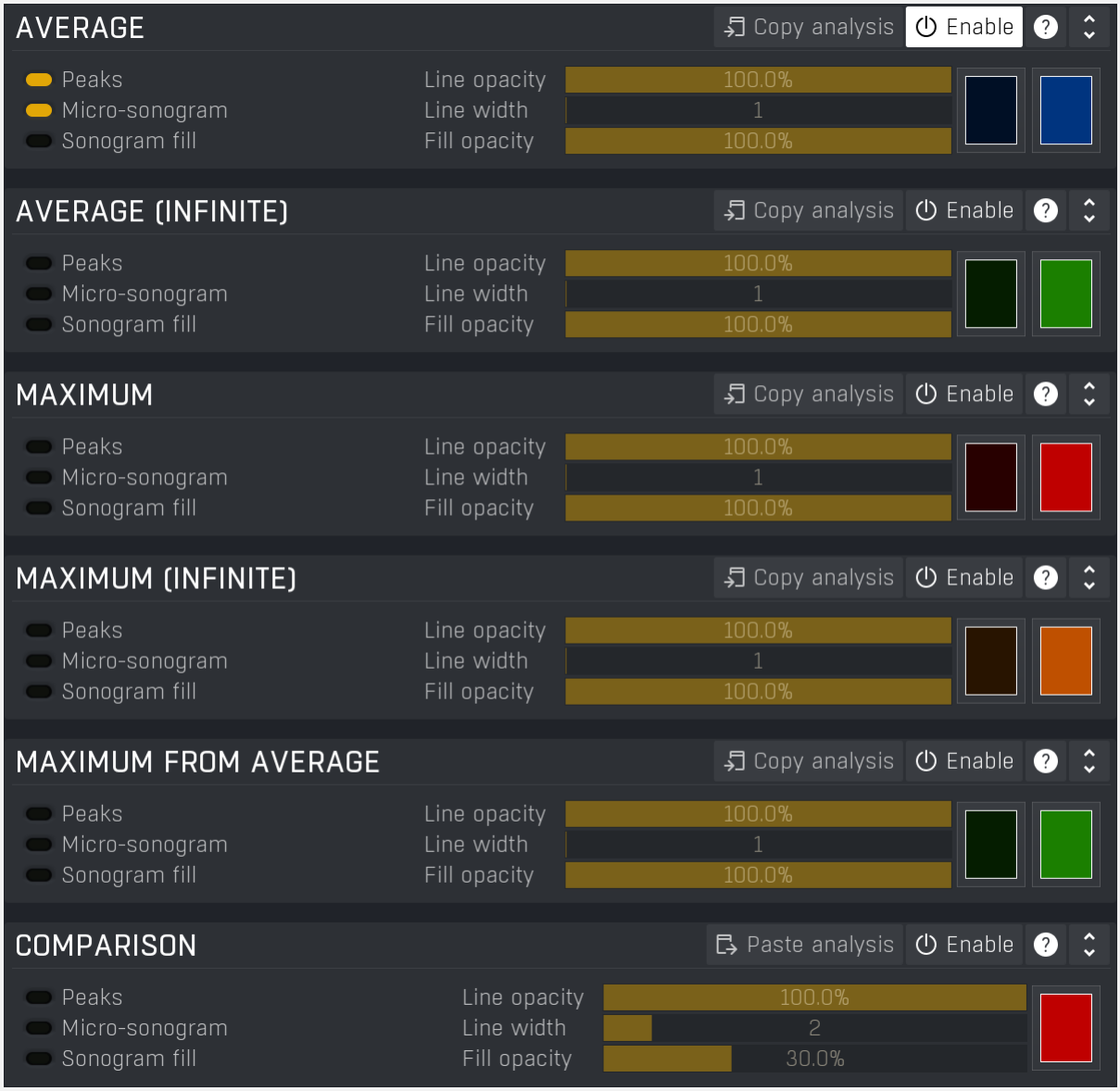
Analytical smoothing switch activates a more complicated smoothing algorithm, which provides more accurate results, however it may require much more CPU power. Unlike normal smoothing this method doesn't change the proportions of frequencies with higher magnitudes. It is useful mostly for technical analysis and for most musical signals it is often better to use the default smoothing method.



Logarithmic averaging

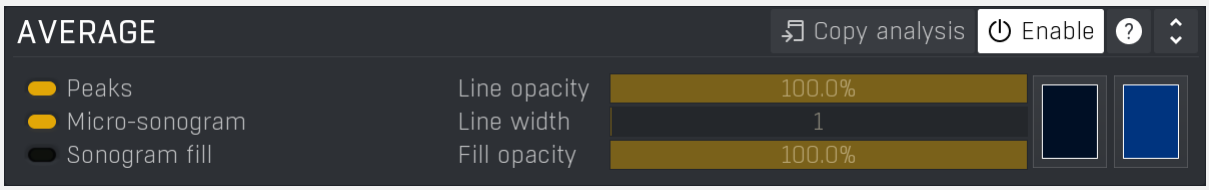
Logarithmic averaging switch activates averaging in logarithmic mode, hence decibels. If you disable it, linear averaging will be used.

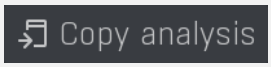
Graphs panel



Graphs panel contains visual settings for the different graphs that you can show in the analyzer.

Average





Copy analysis

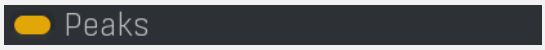
Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.

Analyzer Fill and Line Color - All Channels

Analyzer Fill and Line Color - All Channels defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in all channel modes - see channel mode help in the Main Settings tab.

Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main Settings tab.



Peaks

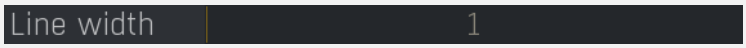
Peaks enables detection of frequencies with the highest magnitudes. Frequencies which are at most 20dB lower than the maximum are displayed, and there may be at most 8 of them. Please note that this feature requires additional CPU power.

Line opacity

Line opacity controls the opacity of the graph outline.

Micro-sonogram

Micro-sonogram displays a small single-state sonogram at the bottom of the graph. This may help you compare relevant frequencies, because it is usually easier to compare colors than graph values.



Line width

Line width controls the width of the graph online.

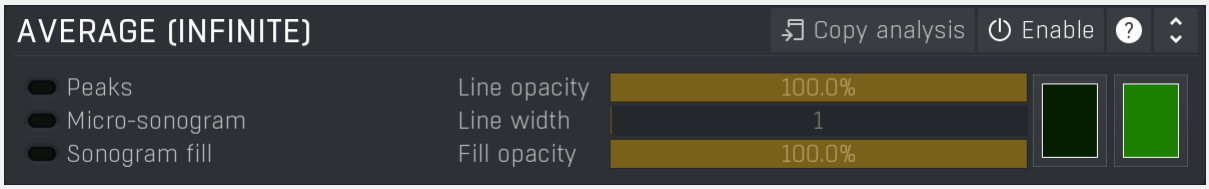
Fill

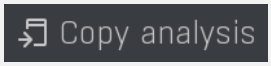
Fill makes the sonogram (enabled by **Show sonogram**) fill the whole area.

Fill opacity

Fill opacity controls the opacity of the graph interior fill.

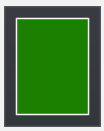
Average (infinite)





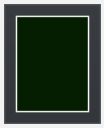
Copy analysis

Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.



Analyzer Fill and Line Color - All Channels

Analyzer Fill and Line Color - All Channels defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in all channel modes - see channel mode help in the Main Settings tab.

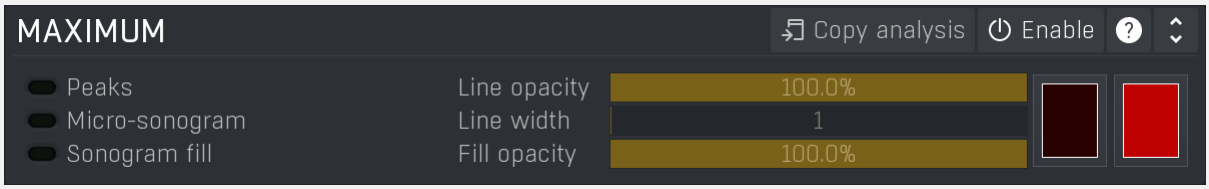


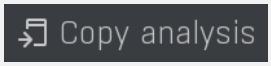
Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main

Settings tab.

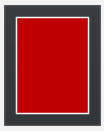
Maximum





Copy analysis

Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.



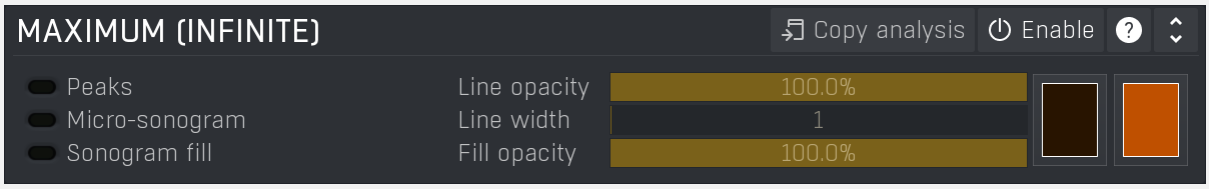
Analyzer Fill and Line Color - All Channels

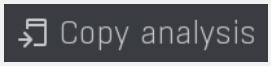
Analyzer Fill and Line Color - All Channels defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in all channel modes - see channel mode help in the Main Settings tab.

Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main Settings tab.

Maximum (infinite)



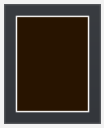


Copy analysis

Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.

Analyzer Fill and Line Color - All Channels

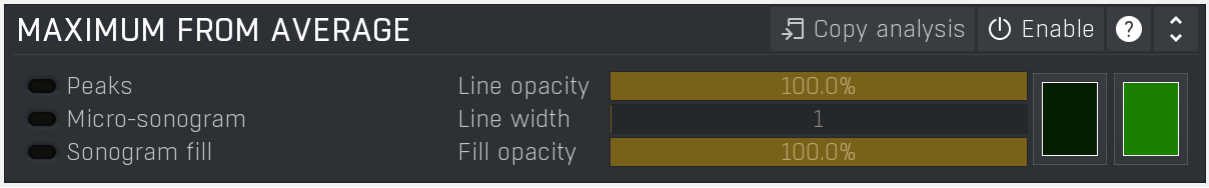
Analyzer Fill and Line Color - All Channels defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in all channel modes - see channel mode help in the Main Settings tab.

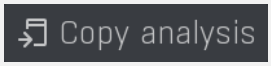


Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main Settings tab.

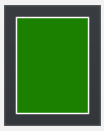
Maximum from average





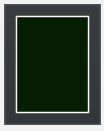
Copy analysis

Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.



Analyzer Fill and Line Color - All Channels

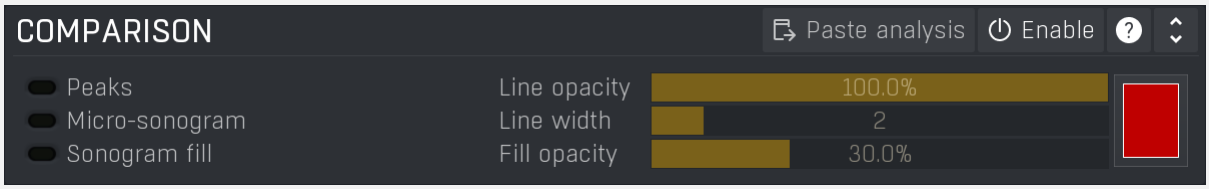
Analyzer Fill and Line Color - All Channels defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in all channel modes - see channel mode help in the Main Settings tab.



Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main Settings tab.

Comparison



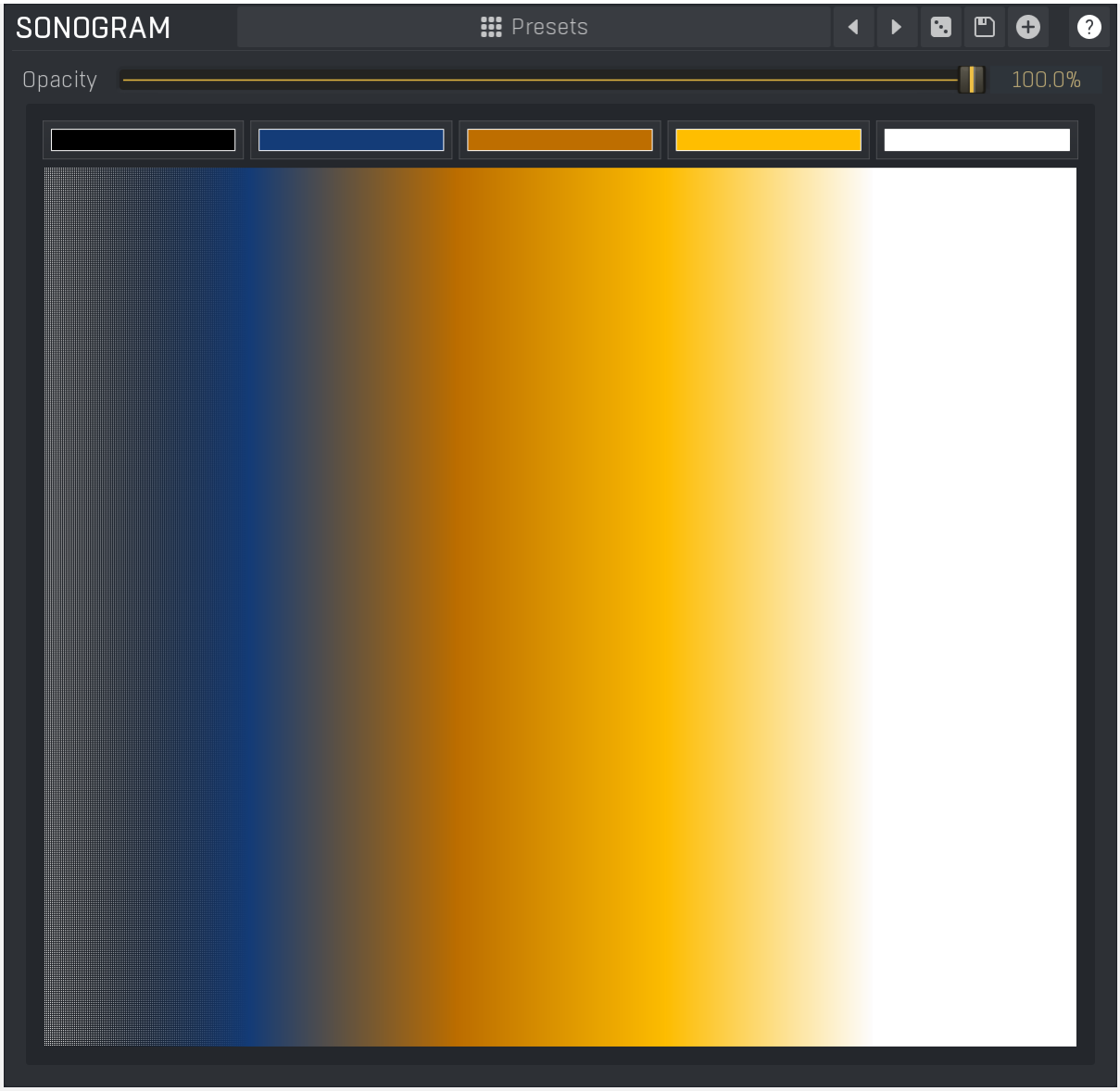
Paste analysis

Paste analysis button pastes an analysis from the system clipboard and displays it as a comparison. This way you can compare your analysis to any other analysis from MeldaProduction plugins.

Analyzer Fill and Line Color - Left Channel

Analyzer Fill and Line Color - Left Channel defines the fill and line colors of the analyzer graph when Rainbow analyzer colors are disabled. This setting is used to set colors in the left channel mode only - see channel mode help in the Main Settings tab.

Sonogram panel



Sonogram panel contains visual settings of the sonogram, mainly the sonogram colors. A sonogram uses a set of colors. When the particular frequency's level is at the minimum, the first color is used. When it is at the maximum, the last color is used. Otherwise it interpolates the colors in-between.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

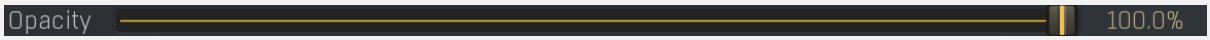
Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

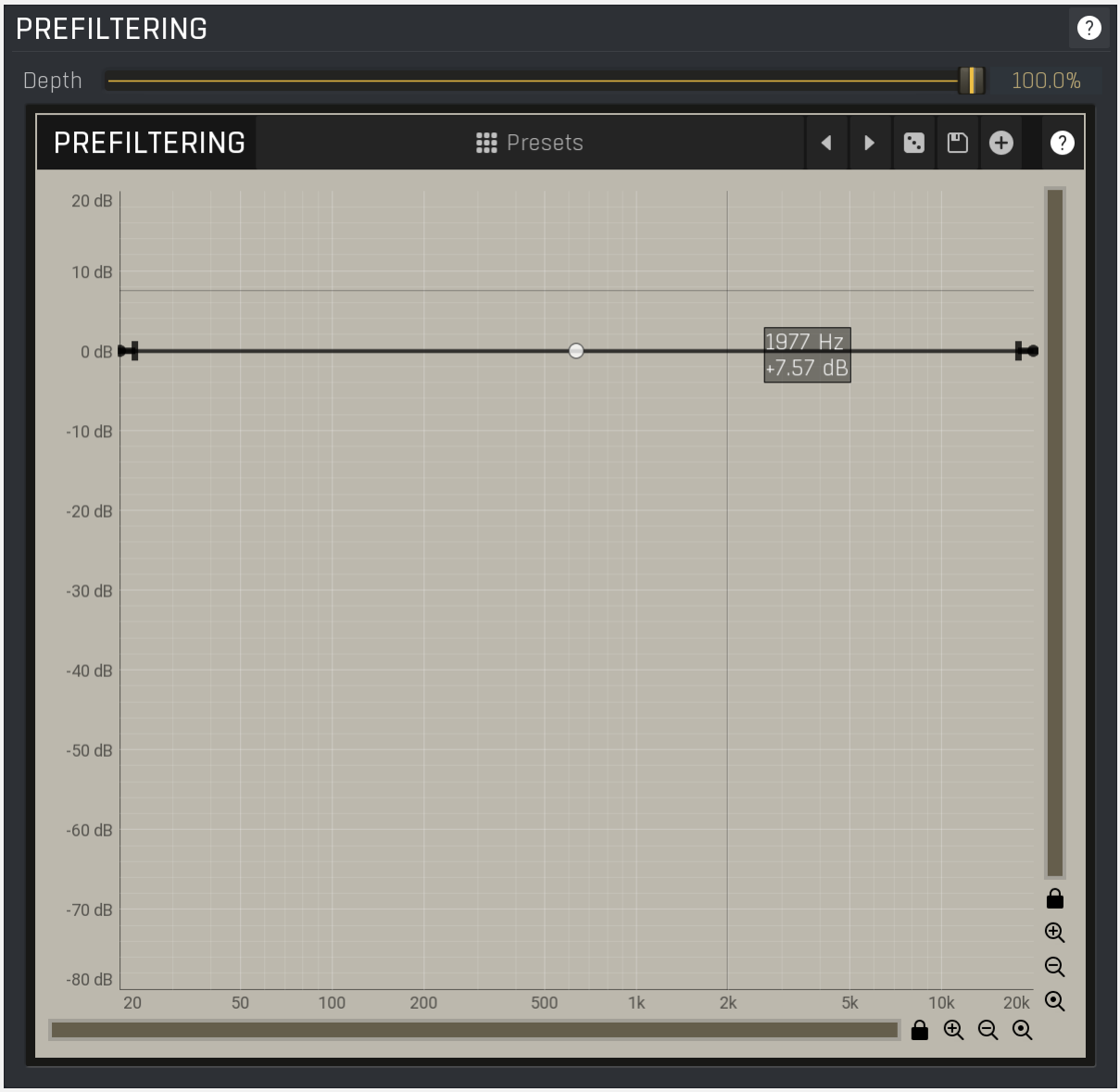
Save button replaces the current preset.



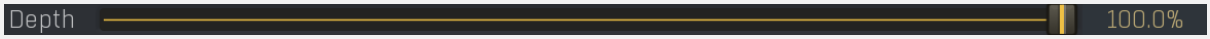
Opacity

Opacity controls the opacity of the sonogram.

Prefiltering panel

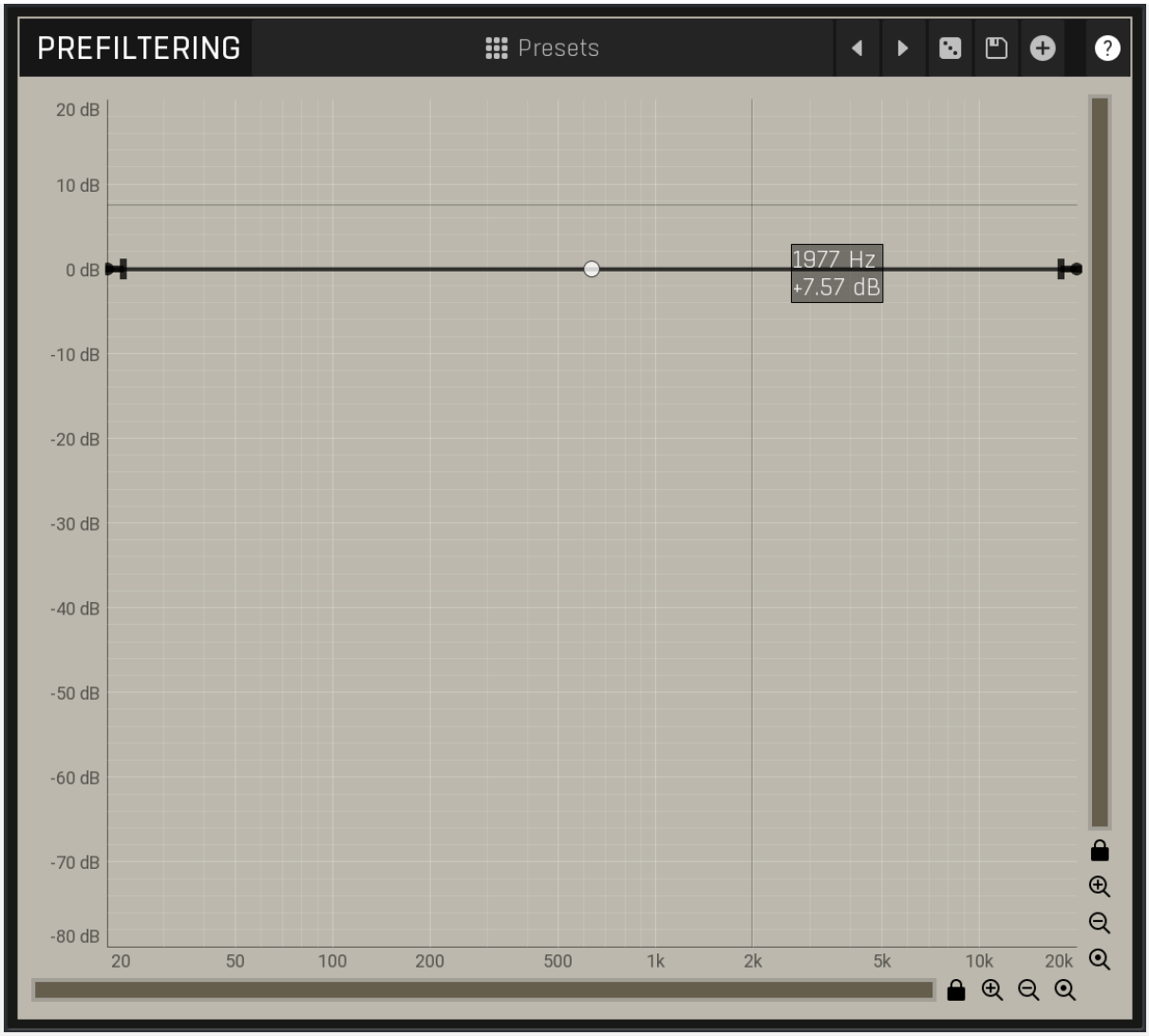


Prefiltering panel provides the optional prefiltering, which means that level of each frequency is either increased or decreased before analysis. Normally the analyzer shows scientific levels of each frequency. However you can for example use the predefined loudness curves, which makes the analyzer show how the human auditory system responds to the frequencies, so it in fact provides more accurate analysis taking into account the fact that human hearing is more complicated than the mathematical model.



Depth

Depth controls the amount of prefiltering. 100% makes the analyzer follow the prefiltering graph precisely, 0% essentially disables this feature.

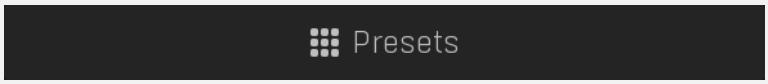


Prefiltering

Envelope graph

Envelope graph provides an extremely advanced way to edit any kind of shape that you can imagine. An envelope has a potentially unlimited number of points, connected by several types of curves with adjustable curvature (drag the dot in the middle of each arc) and the surroundings of each point can also be automatically smoothed using the smoothness (horizontal pull rod) control. You can also literally draw the shape in drawing mode (available via the main context menu).

- **Left mouse button** can be used to select points. If there is a *point*, you can move it (or the entire selection) by dragging it. If there is a *curvature circle*, you can set up its tension by dragging it. If there is a *line*, you can drag both edge points of it. If there is a *smoothing controller*, you can drag its size. Hold **Shift** to drag more precisely. Hold **Ctrl** to create a new point and to remove any points above or below.
- **Left mouse button double click** can be used to create a new point. If there is a *point*, it will be removed instead. If there is a *curvature circle*, zero tension will be set. If there is a *smoothing controller*, zero size will be set.
- **Right mouse button** shows a context menu relevant to the object under the cursor or to the entire selection. Hold **Ctrl** to create or remove any points above or below.
- **Middle mouse button** drag creates a new point and removes any points above or below. It is the same as holding Ctrl and dragging using left mouse button.
- **Mouse wheel** over a point modifies its smoothing controller. If no point is selected, then all points are modified.
- **Ctrl+A** selects all points. **Delete** deletes all selected points.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

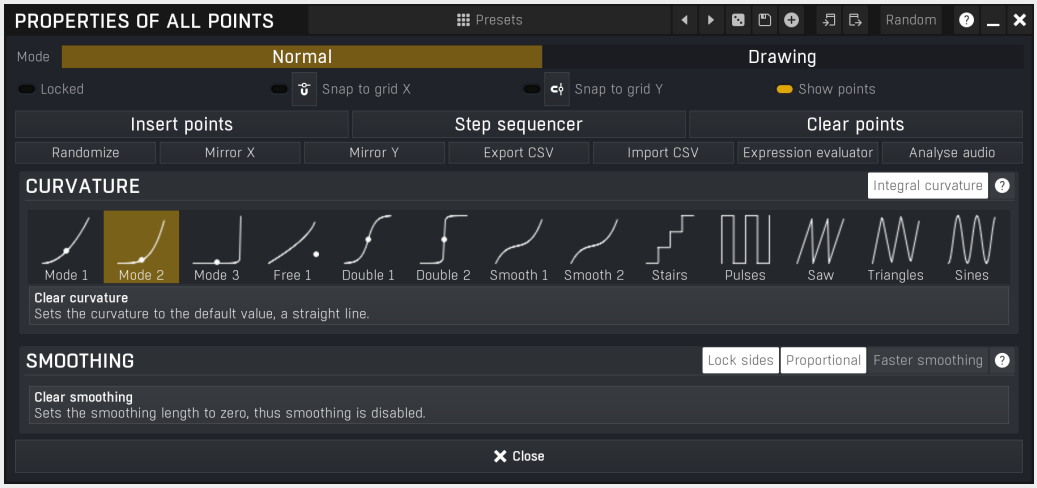
Randomize

Randomize button loads a random preset.

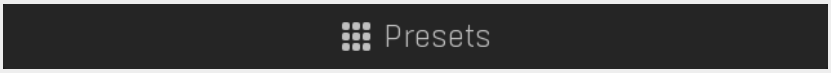
Save

Save button replaces the current preset.

Envelope graph menu



Envelope graph menu provides additional features which are used to edit the graph. Open the menu using right mouse button in the graph. Please note that if you select some points in the graph, or click on a point for example, the menu will be different and will cover only those features related to the selected set of points.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

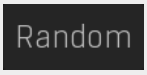
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

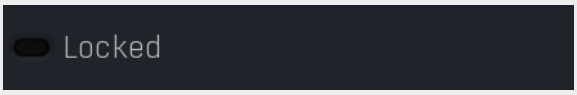
Paste

Paste button loads the settings from the system clipboard.



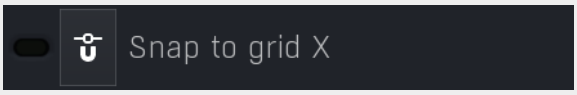
Random

Random button generates random settings using the existing presets.



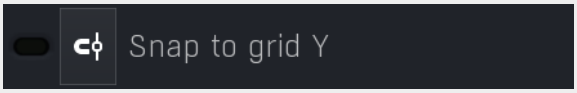
Locked

Locked switch makes the selected/all points locked, so they cannot be moved or changed.



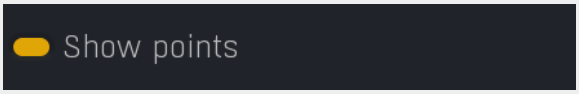
Snap to grid X

Snap to grid X activates the snap to grid feature. Alternatively you can press **Alt** while dragging a point or selection.



Snap

Snap button activates the snap to grid feature. Alternatively you can press **Alt** while dragging a point or selection.



Show points

Show points enables the display of envelope points on the graph.

Insert point

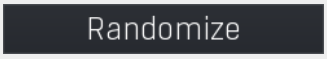
Insert point button creates a point at mouse position.

Step sequencer

Step sequencer button generates the envelope from step sequencer.

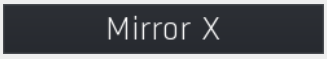
Clear points

Clear points button deletes all points.



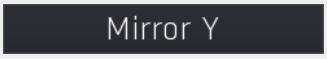
Randomize

Randomize button slightly modifies the Y coordinates.



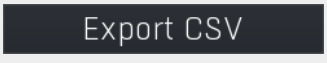
Mirror X

Mirror X button inverts the X coordinates of all points.



Mirror Y

Mirror Y button inverts the Y coordinates of all points.



Export CSV

Export CSV feature lets you export the graph to a CSV file. CSV file is a simple text format, which has multiple lines with X and Y coordinates delimited by ';'. For example:
0.275;0.2
0.438;0.5
0.775;0.67

Import CSV

Import CSV feature lets you select a CSV file and imports the graph points from it. CSV file is a simple text format, which has multiple lines with X and Y coordinates delimited by ';'. For example:
0.275;0.2
0.438;0.5
0.775;0.67

Expression evaluator

Expression evaluator lets you generate points based on a mathematic formula. The only input variable is 'x', so as an example you may write 'ln(x^3 + 1) - sin(x*x)'.

Expression evaluator uses traditional C/C++ style formating, which is natural for most people. It provides arithmetics, logical and conditional operators. Following terms are supported:
Constants: **pi**, **e**, **sqrt2**, **ln2**

Arithmetic operators:
-a inverts the sign, e.g. "-x" produces +2 for x=-2
a+b = addition
a-b = subtraction
a*b = multiplication
a/b = division
a%b = modulo, remainder after division
a^b = power, e.g. "2^3" produces 2*2*2 = 8

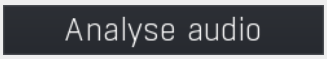
Arithmetic functions:
min(a,b) = minimum of both values
max(a,b) = maximum of both values
limit(a,min,max) = a limited into the interval min..max
to01(a,min,max) = converts "a" as min..max to 0..1
from01(a,min,max) = converts "a" as 0..1 to min..max
tom11(a,min,max) = converts "a" as min..max to -1..1
fromm11(a,min,max) = converts "a" as -1..1 to min..max

Basic mathematic functions: **abs(x)** = absolute value, e.g. abs(-3) = 3 **sqr(x)** = x*x **sqrt(x)** = square root **exp(x)** = natural exponential e^x **ln(x)** = natural logarithm **log10(x)** = logarithm with base 10 **log(x, base)** = logarithm with specified base **inv(x)** = 1/x **sgn(x)** = sign of x, -1 or 0 or +1 depending on x*x **round(x)** = rounding to the nearest value **floor(x)** = rounding to the nearest lower value, e.g. floor(-2.3) = -3 **ceil(x)** = rounding to the nearest higher value, e.g. ceil(-2.3) = -2 **rand(x)** = random value from 0 to x

Functions for specific units:
f01(a) = converts "a" as frequency from 20..20000 into log scale 0..1
ffrom01(a) = converts "a" as 0..1 (log scale) to frequency from 20..20000
ffrom01limit(a) = same, but makes sure the output is in 20..20000 range
f01range(a,min,max) = converts "a" as frequency from min to max into log scale 0..1
ffrom01range(a,min,max) = converts "a" as 0..1 (log scale) to frequency from min to max
ffrom01rangelimit(a,min,max) = same, but makes sure the output is in min..max range
todb(a) = converts "a" as multiplier to dB value by calculating "20*log10(a)"
fromdb(a) = converts "a" as dB value to multiplier by calculating "10^(a/20)"

Trigonometric functions: **sin(x)**, **asin(x)**, **cos(x)**, **acos(x)**, **tan(x)**, **atan(x)**, **sinh(x)**, **cosh(x)**, **tanh(x)**

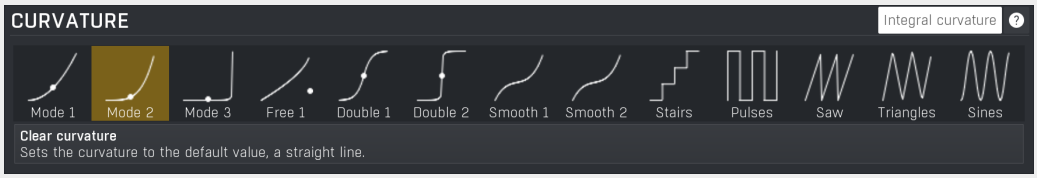
Logical operators:
a==b = comparison producing 1 if "a" and "b" are equal, 0 otherwise
a!=b = comparison producing 1 if "a" and "b" are NOT equal, 0 otherwise
a<b = comparison producing 1 if "a" is lower than "b", 0 otherwise
a<=b = comparison producing 1 if "a" is lower or equal to "b", 0 otherwise
a>b = comparison producing 1 if "a" is greater than "b", 0 otherwise
a>=b = comparison producing 1 if "a" is greater or equal to "b", 0 otherwise
!a = logical negation, 0 produces 1, 0 otherwise
a&&b = logical AND, produces 1 if both "a" and "b" are nonzero
a||b = logical OR, produces 1 if any of "a" and "b" are nonzero
a^^b = logical XOR, produces 1 if "a" and "b" are logically different
a ? b : c = if a is nonzero, then the result is b, otherwise it is c



Analyse audio

Analyse audio lets you analyse a portion of an audio file at specified intervals, extract its level envelope and use those levels to construct the graph's curve.

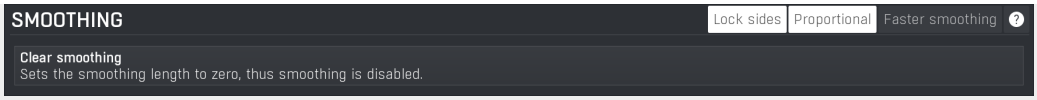
Curvature



Integral curvature

Integral curvature makes the multi-curvature modes such as rectangles always have an integral number of items, e.g. 1, 2, 3, ... rectangles. If you disable this, it will be also possible to have for example 2.3 rectangles, which will however cause a discontinuity.

Smoothing



Lock sides **Lock sides**

Lock sides makes the smoothing factor equal on both sides.

Proportional **Proportional**

Proportional makes the smoothing area size defined by the smaller side.

Faster smoothing **Faster smoothing**

Faster smoothing enables slightly faster algorithm, which can however often cause unnecessary curving.

Pause

Pause button stops the analyzer temporarily.

Normalize

Normalize button enables or disables the visual normalization, which makes the loudest frequency be displayed at the top of the analyser area (0dB); it does not normalise the sound. This is very useful for comparing frequency levels, however it does hide the actual level.
When comparing 2 spectrums you are usually interested mainly in the frequency level differences. In most cases both audio materials will have different overall levels, which would mean that one of the graphs would be "lower" than the other, making the comparison quite difficult. Normalize fixes this and makes the most prominent frequencies of the spectrum reach the top of the analyzer area (or have the most highlighted color in case of sonogram).

Reset

Reset button resets analyzer graphs. This is particularly useful when analyzing infinite average and maximum values.

Copy analysis

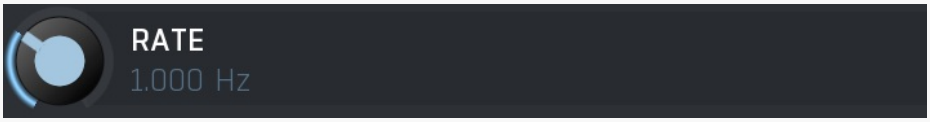
Copy analysis button copies the current state of the analysis into the system clipboard so that you can paste it into another analyzer for comparison. Hold **ctrl** to export the analysis into a CSV file.

Paste

Paste button pastes the analysis from the system clipboard and displays it as the comparison in the graph.

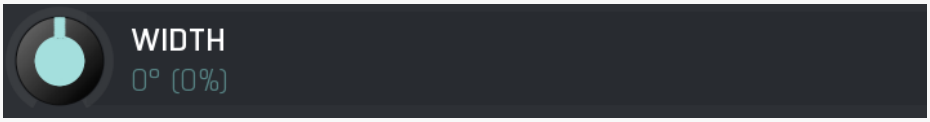
LFO panel

LFO panel contains parameters of the LFO.



Rate

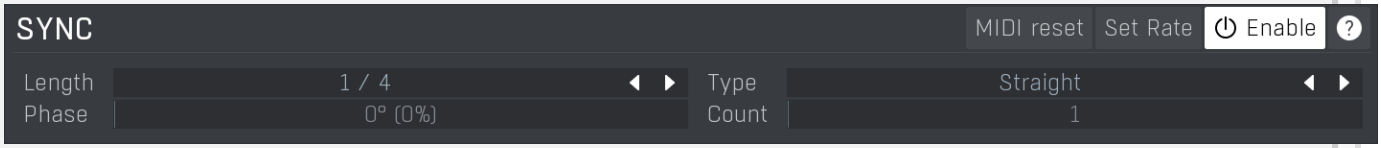
Rate defines the speed of the low frequency oscillator. This is available only when synchronization to host is disabled.
Range: 0.0100 Hz to 20.0 kHz, default 1.000 Hz



Width

Width defines the phase difference between particular channels. This is very simple and often practical way to accomplish a kind of stereo expansion.
Range: -360Â° (-100.0%) to 360Â° (100.0%), default 0Â° (0%)

Synchronization panel



Synchronization panel contains parameters for the to-host synchronization.

MIDI reset

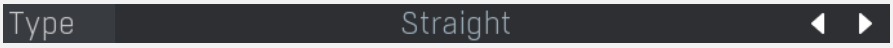
MIDI reset button displays the settings for the MIDI reset feature, which can reset the LFO based on incoming MIDI notes.

Set Rate

Set Rate button sets the **Rate** parameter used when sync is disabled according to current sync speed. This is useful when you want to leave the oscillator unsynchronized, however you want to start with the current synced speed.

Length

Length defines the note length to be used.



Type

Type defines the note type, such as straight notes or triplets, to be used. Together the **Length** and **Type** determine the actual time/delay.
Example: '1/4 Straight' at 120 bpm = a delay of 500 ms, '1/4 Triplet' at 160 bpm = a delay of 281.25 ms.

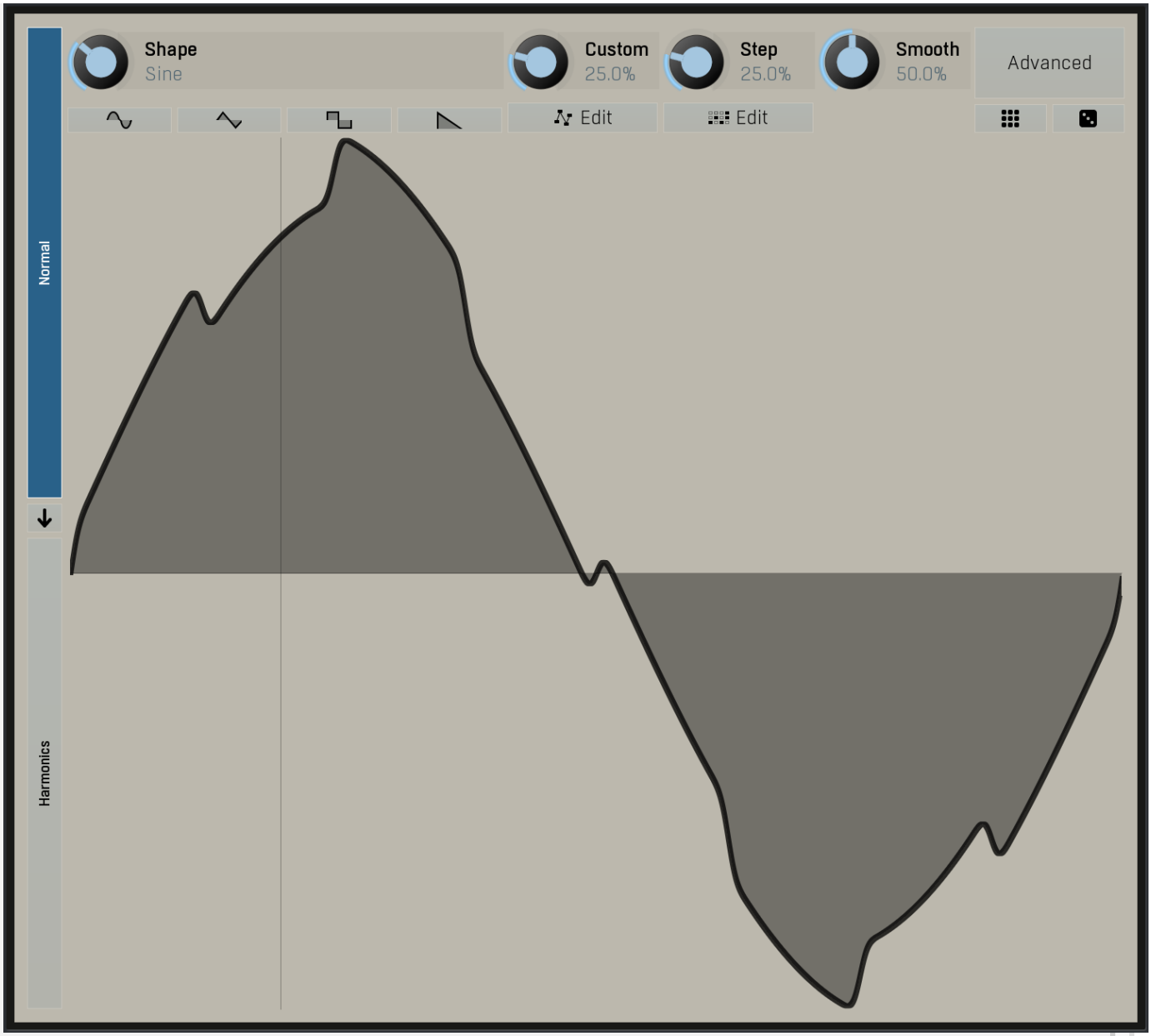
Phase

Phase defines the phase offset of the to-host synchronization.
Range: 0Â° (0%) to 360Â° (100.0%), default 0Â° (0%)

Count

Count defines the number of the units, hence multiplies of the sync length.
Range: 1 to 64, default 1

Signal graph



Signal graph defines the low frequency oscillator shape.Signal-generator is an incredibly versatile generator of low & high frequency signals. It offers 2 distinct modes - Normal and Harmonics.
Normal mode is appropriate for low-frequency oscillators, where the graphical shape is relevant and is used to drive some form of modulation. For example, a tremolo uses this modulation to change the actual signal level in time. Frequencies for such oscillators usually do not exceed 20Hz as this is a sort of limit above which the frequencies become audible.
Harmonics mode is designed for high-frequency oscillators, where the actual shape is not as important as the harmonic content of the resulting signal, hence it is especially useful for actual audio signals. Please note that since a shape can contain more harmonics than those available from the harmonic generator, the results may not be exactly the same. As an example, a rectangular wave in normal mode may sound fuller than when converted to the harmonic mode.

Use the arrow-down button to switch from normal mode to harmonics mode or click the **Normal** and **Harmonics** buttons

Normal mode

The generator first uses a set of predefined signal shapes (sine, triangle, rectangle...), which you can select directly by right-clicking on the editor and choosing the requested shape from the menu. This menu also provides a link to the modulator shapes preset manager, normalization and randomization. You can also use the **Main shape** parameter, which generates a combination of adjacent signals to provide a nearly inexhaustible number of basic shapes.

The engine then combines the predefined shape with a **Custom shape**, which may be anything you can draw using the advanced envelope engine, depending on the level set by the **Custom shape** control. Use the **Edit** button to edit the custom shape.

You can also combine those results with a fully featured step sequencer, with variable number of steps and several shapes for each of them, depending on the level set by the **Step sequencer** control. Use the lower **Edit** button to edit

the step sequence.

Those results may be mixed with a custom sample, which is available from the advanced settings, accessed by clicking the **Advanced** button.

Smoothness softens any abrupt edges, generated by the step sequencer for example.

Finally there are **Advanced** features providing more complex transformations, adding harmonics etc. or you can click the **Randomize** button in the top-left corner to generate a random, but reasonable, modulator shape.

Harmonics mode

Harmonics mode represents the signal as a series of harmonics (that is, multiples of the base frequency). For example, when your oscillator has a frequency of 2Hz (set in the **Rate** panel), then the harmonics are 2Hz, 4Hz, 6Hz, 8Hz etc. In theory, any signal can be created by mixing a potentially infinite number of these harmonics.

The harmonics mode lets you control the levels and phases of each harmonic. The top graph controls the levels of individual harmonics, while the bottom one controls their phases. Use the left-mouse button to change the values in each graph, the right-mouse button sets the default for the harmonics - 0% level and 0% phase. In both graphs the harmonics of power 2 (that is octaves) are highlighted. Other harmonics may actually sound disharmonic, despite their names.

For example, if you reset all harmonics to the defaults and increase only the first one, you will get a simple sine wave. By adding further harmonics you make the output signal more complex.

Harmonics controls the number of generated harmonics. The higher the number is, the richer the output signal is (unless the levels are 0% of course). This is useful to make the sound cleaner. For example, if you transform a saw-tooth wave to harmonics, it would not sound like a typical saw-tooth wave anymore, but more like a low-passed version of one. The more harmonics you use, the closer you get to the original saw-tooth wave.

Generator is a powerful tool for generating the harmonics, which are otherwise rather clumsy to edit. The generator provides several parameters based upon which it creates the entire series of harmonic levels and phases. These parameters are usually easier to understand than the harmonics themselves. Part of the generator is the randomizer available via the **Random seed** button, which smartly generates random settings for the generator. This makes the process of getting new sounds as simple as possible.

Signal generation fundamentals

The signal generator produces a periodic signal with specified wave shape. This means that the signal is repeating over and over again. As a result it can only contain multiples of the fundamental frequency. For example, if the generator is producing 100Hz signal, then it can contain 100Hz (fundamental or 1st harmonic), 200Hz (2nd harmonic), 300Hz (3rd harmonic), 400Hz (4th harmonic) etc. However, it can never produce 110Hz. You can then control the level of each harmonic and their relative phases. It does not matter whether you use the normal mode using oscillator shapes, or harmonics mode where you can control the harmonics directly. If both modes result in the same wave shape (such as sine wave vs. 1st harmonic only), then the result is exactly the same.

Sine wave is the simplest of all as it contains the fundamental frequency only. The "sharper" the signal shape is, the more harmonics it contains. The biggest source of higher harmonics is a "discontinuity", which you can see in both rectangle and saw waves. In theory, these signals have an infinite number of harmonics. However since our hearing is highly limited to less than 20kHz, the number of harmonics which are relevant is actually pretty small. If you generate a 50Hz signal, which is very low, and assuming that you have extremely good ears and you actually hear 20kHz, then the number of harmonics audible for you is 20000 / 50 = 400.

What happens above 20kHz?

Consider the example above again, what happens with harmonics above 400? These either stay there and simply are not audible, disappear if anti-aliasing is used, or get aliased back under 20kHz in which case you get the typical digital dirt.

When you convert a rectangle wave to harmonics mode, only the first 256 harmonics are used, so it basically works like an infinitely steep low-pass filter. What is the limit then? 50 Hz * 256 = 12.8kHz. The harmonic mode will not produce anything above this limit if you are generating a 50Hz signal. Most people do not hear anything above 15kHz, so this is usually enough, but if not, you may need to use the normal mode where you get the "infinite" number of harmonics.

What you see is not always what you get!

Say you want a rectangle wave and play a 440Hz tone(A4). You would expect the output signal to be a really quick rectangle wave, right? Wrong! If you would do that, and actually most synthesizers on the market do that, you would get the infinite number of harmonics. And, since you are working in say 48kHz sampling rate, the maximum frequency that can actually exist in your signal is 24kHz. So everything above it would get aliased below 24kHz, and there would be a lot

of aliased dirt.

The "good" synthesizers perform a so-called anti-aliasing. There are several methods, most of them require quite a lot of CPU or have other limitations. The goal is to remove all frequencies above the 24kHz in our case or in reality, it is more about removing all aliased frequencies above 20kHz - this means, that we do not care about frequencies above 20kHz, because we do not hear them anyway. But we will keep it simple. Let's say we remove everything above 20kHz. You already know that the rectangle wave can be created using an infinite number of harmonics or sine waves. We removed everything above the 45th harmonic (20000 / 440) so our rectangle wave is trying to be formed using just 45 harmonics, so it will not really look like a rectangle wave.

After some additional filtering (like DC removal), the rectangle wave may look completely different than a true rectangle wave, yet it would sound the same! Does it matter? Not really. You simply edit the shape as a rectangle wave and let the synthesizer do the ugly stuff for you. But do not check the output, because it may be very different than what you would expect ;).

How can I generate non-harmonic frequencies?

Ok, so now you are playing a 440Hz (A4) saw wave, it contains 440Hz, 880Hz, 1320Hz etc. Anything generated using the signal generator can contain only these frequencies, the only difference is the levels and phases of each of them. What if you want to make the signal dirty by adding say 500Hz? Well, that is not that simple! Here we are getting into audio synthesizer stuff, so let us just give you a few hints.

The traditional way is to use modulation. One particular method is called frequency modulation (FM). Instead of generating a 440Hz saw wave with your generator, you change the pitch, up and down. You are modulating the frequency, that's why FM. It is basically a vibrato, but as you increase the speed of the vibrato, it gets so quick that you stop noticing the pitch changes (that's very simplified but it serves the purpose) and instead it starts producing a very complex spectrum. Will the 500Hz be there? Well, if setup correctly, yes, but there will also be lots of other non-harmonic frequencies.

Another way is possible without any other tools. Let's say you do not want 440Hz, but 660Hz. Then you may generate 220Hz instead of 440Hz (which is one octave below it) and voila, 660Hz is the 3rd harmonic (3 x 220 is 660)! But you need to shift the saw wave one octave above. Fortunately it is not that hard here - go to the normal mode, select saw tooth, click advanced, and use the harmonics panel to remove the fundamental and leave just the 2nd harmonic, then convert it to harmonic mode. Well, it's not that hard, but it's not exactly simple either...

The only way is, of course, additive synthesis. In that case you do not use one oscillator, but many of them. It lets you generate just about anything. But there is a catch, actually many of them. First, you need to say "ok I want this frequency and that frequency...", the setup is actually infinitely hard as there may be an infinite number of frequencies :). And the second is, of course, CPU requirements.

So is there some ultimate solution? Nope, sorry. The good thing is, you will not probably need it, because while what you see is not always what you get, also what you want is often not what you really want to hear :).

Normal

Normal button switches the generator into the normal mode, which lets you edit the shape of the oscillator. This is especially advantageous for low-frequency oscillators, where the shape matters even though it doesn't have any physical

meaning.

Convert

Convert button converts the current shape into harmonic-based representation. Please note that since the number of harmonics is limited, the result will not perfectly resemble the original shape.

Harmonics

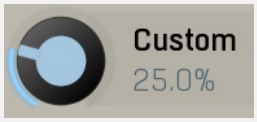
Harmonics button switches the generator into the harmonics mode, which lets you edit the levels and phases of individual harmonics. This is especially advantageous for high-frequency oscillators, hence sound generators.

Signal generator in Normal mode

Signal generator in Normal mode works by generating the oscillator shape using a combination of several curves - a predefined set of standard curves, custom shape, step sequencer and custom sample. It also post-processes the shape using several filters including smoothing to custom transformations. This is especially useful when using the oscillator as an LFO (low-frequency-oscillator), where the harmonic contents does not really matter, but the shape does.

Shape

Shape controls the main shape used by the signal generator. There are several predefined shapes: exponential, triangle, sine power 8, sine power 4, sine square, sine, harmonics, more harmonics, disharmonics, sine square root, sine 4 root, rectangle, rect-saw, saw, noise and mess. You can choose any of them or interpolate between any 2 adjacent shapes using this control.



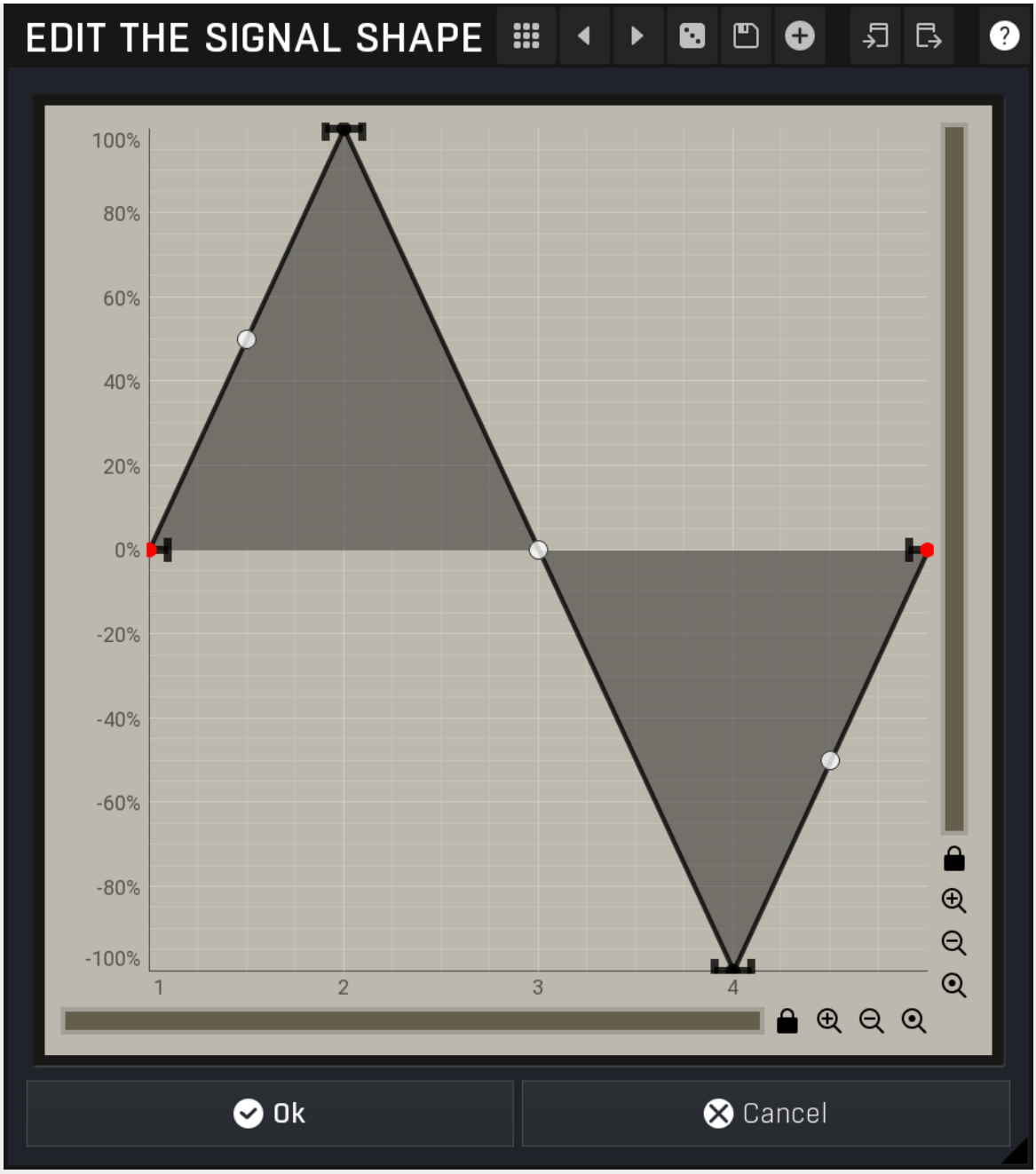
Custom

Custom controls the amount of the custom shape that is blended into the main shape.

Edit

Edit button shows the custom shape editor.

Signal generator custom shape editor



Signal generator custom shape editor controls the custom shape. You can edit virtually any shape that you can imagine and then blend it with the standard shapes, the step sequencer etc.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

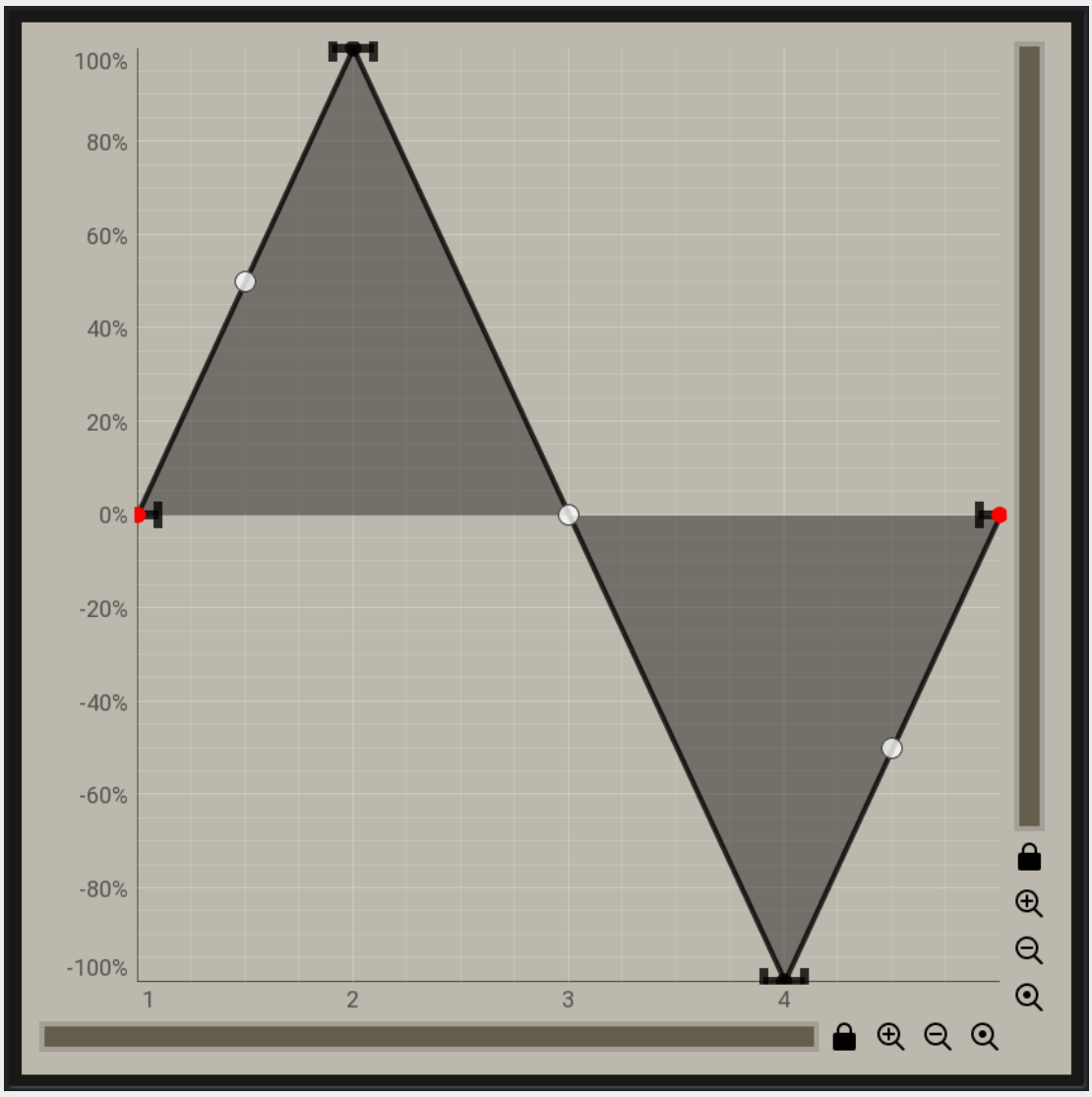
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.



Graph editor

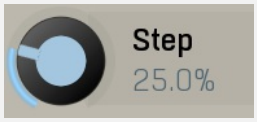
Graph editor lets you edit the envelope graph.

Envelope graph

Envelope graph provides an extremely advanced way to edit any kind of shape that you can imagine. An envelope has a potentially unlimited number of points, connected by several types of curves with adjustable curvature (drag the dot in the middle of each arc) and the surroundings of each point can also be automatically smoothed using the smoothness (horizontal pull rod) control. You can also literally draw the shape in drawing mode (available via the main context menu).

- **Left mouse button** can be used to select points. If there is a *point*, you can move it (or the entire selection) by dragging it. If there is a *curvature circle*, you can set up its tension by dragging it. If there is a *line*, you can drag both edge points of it. If there is a *smoothing controller*, you can drag its size. Hold **Shift** to drag more precisely. Hold **Ctrl** to create a new point and to remove any points above or below.
- **Left mouse button double click** can be used to create a new point. If there is a *point*, it will be removed instead. If there is a *curvature circle*, zero tension will be set. If there is a *smoothing controller*, zero size will be set.
- **Right mouse button** shows a context menu relevant to the object under the cursor or to the entire selection. Hold **Ctrl** to create or remove any points above or below.
- **Middle mouse button** drag creates a new point and removes any points above or below. It is the same as holding Ctrl and dragging using left mouse button.
- **Mouse wheel** over a point modifies its smoothing controller. If no point is selected, then all points are modified.

- **Ctrl+A** selects all points. **Delete** deletes all selected points.



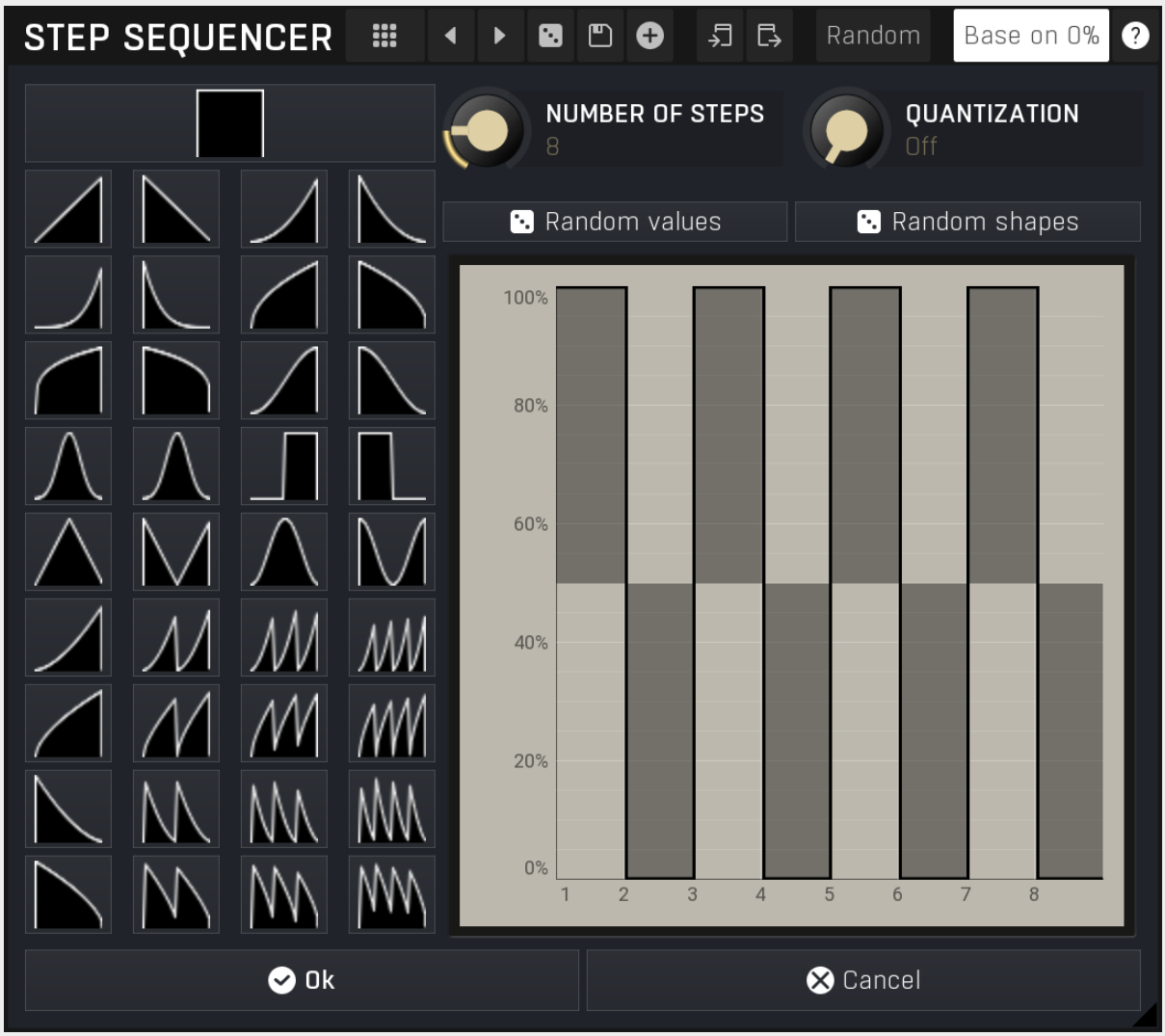
Step

Step controls the amount of the step sequencer shape that is blended into the main shape (which has already been blended with the custom shape).

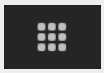
Edit

Edit button shows the step sequencer editor.

Signal generator step sequencer editor



Signal generator step sequencer editor controls the step sequencer shape. You can have various numbers of steps each with a different value and shape. Note that for classic rectangular shapes the output can be very rough, hence it may be worth considering using **Smoothness** parameter to smooth out the resulting shape. This will use additional CPU power of course, but that should be negligible unless you modulate any of the signal generator parameters.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

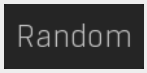
Right arrow
Right arrow button loads the next preset.

Randomize
Randomize button loads a random preset.

Save
Save button replaces the current preset.

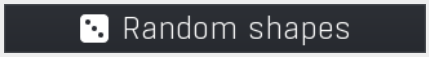
Copy
Copy button copies the settings onto the system clipboard.

Paste
Paste button loads the settings from the system clipboard.

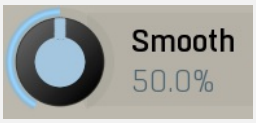


Random
Random button generates random settings using the existing presets.

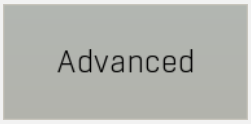
Random values
Random values button generates random sequence of values, but keeps the shape of each step.



Random shapes
Random shapes button generates random sequence of shapes, but keeps the values of each step.

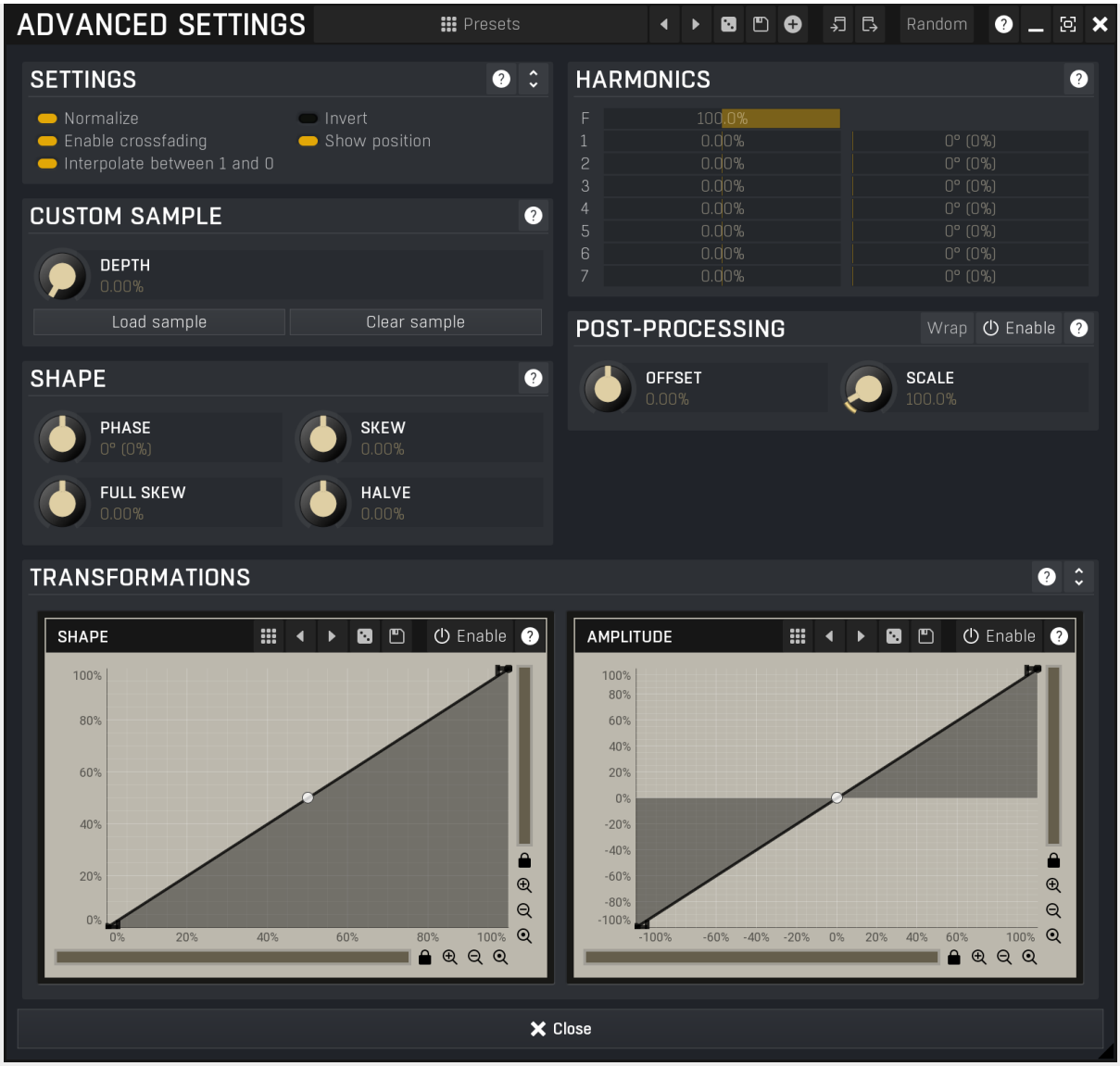


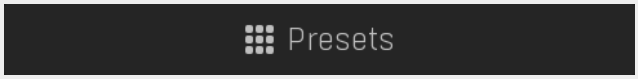
Smooth
Smooth controls the amount of smoothing. Many shapes, especially those produced by the step sequencer, have rough jagged edges, which may be advantageous, but when used to modulate certain parameters, the output may be clicking or causing other artifacts. Smoothness helps it by smoothing the whole signal shape out and removing these rough edges.



Advanced
Advanced button displays an additional window with more advanced settings for post-processing the signal shape, such as harmonics or custom transformations.

Advanced settings





Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

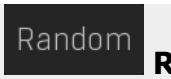
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

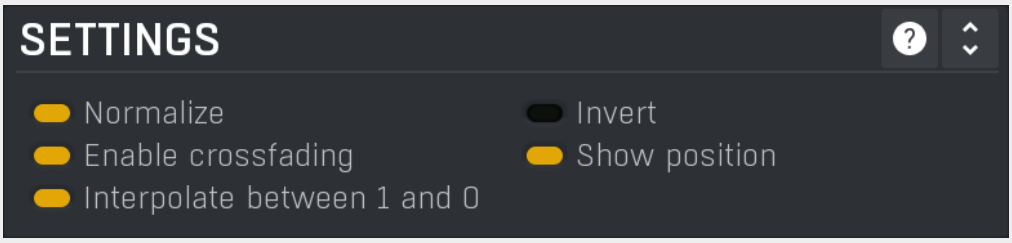
Paste button loads the settings from the system clipboard.



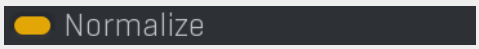
Random

Random button generates random settings using the existing presets.

Settings panel

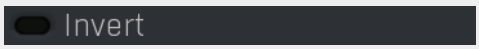


Settings panel contains some global settings of the oscillator.



Normalize

Normalize switch enables normalization to -1..+1. It is generally desirable since even if you draw a custom shape, you usually want it to have the full range. You may want to disable it if you want to create some custom shapes, where the level actually matters.



Invert

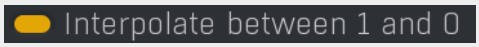
Invert switch simply inverts the output shape vertically.

Enable crossfading

Enable crossfading enables interpolation between shapes when the shape is changing. This requires more CPU, but can avoid zipper noise when the shape is being modulated for example.

Show position

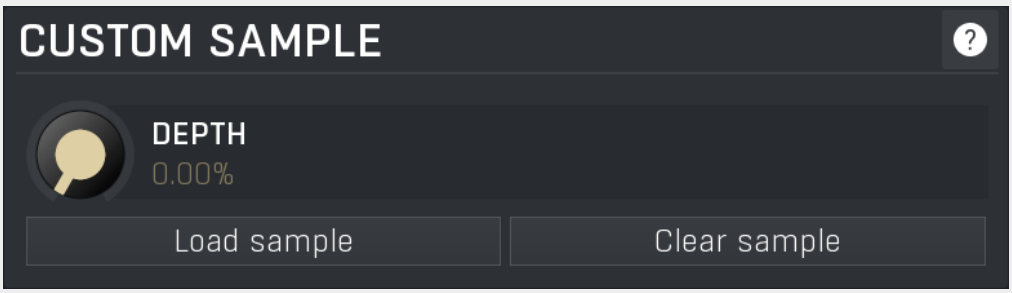
Show position makes the editor display a position indicator.



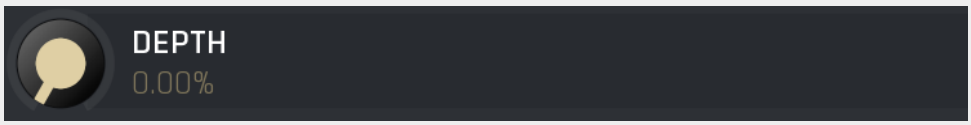
Interpolate between 1 and 0

Interpolate between 1 and 0 smoothens the discontinuity between 1 and 0 values, which is inevitable for shapes such as saw or rect for example. However when this is a high frequency oscillator (HFO), this discontinuity is what creates the highest frequencies, so it is actually desirable. When using it as an LFO, you may also want the discontinuity in some extreme cases.

Custom sample panel

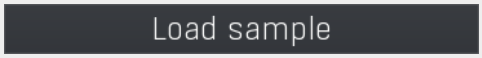


Custom sample panel contains parameters of the custom sample that you can load and mix with the other sources. Do NOT confuse this with a sampler, the custom sample is taken as one period of the waveform. It can be used for creative effects and it can be used to import a custom waveform. The custom sample is then stored with limited precision within the settings, so the sample does not need to be kept on the system, but note that these settings may be quite large. To limit the space required by the settings, the sample is stored only if the depth is not 0%, meaning only if the sample is actually used.



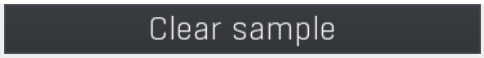
Depth

Depth controls the amount of custom sample mix. 0% means the sample is not used even if there actually is one loaded. 100% means the sample completely overrides the basic shape, custom shape, step sequencer... However, transformations are still performed on the sample.



Load sample

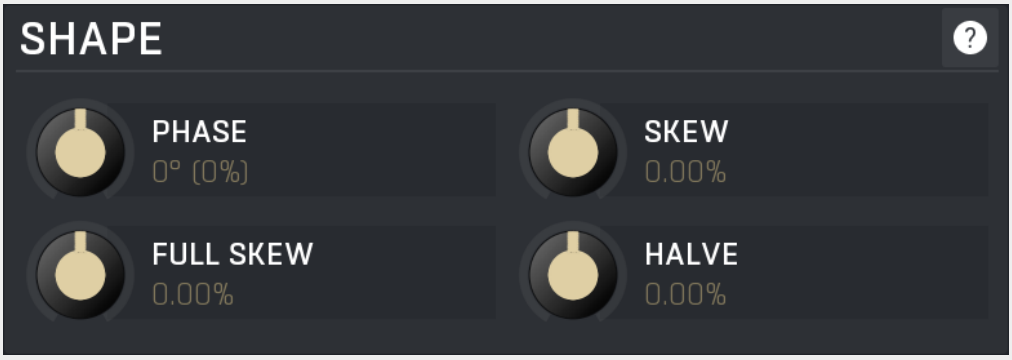
Load sample button displays a file selection window, which lets you select the custom sample file.



Clear sample

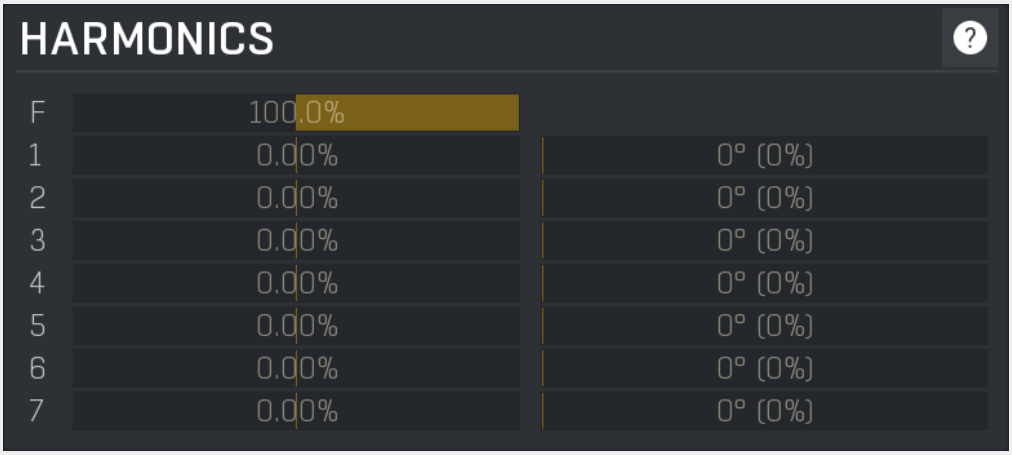
Clear sample button removes the custom sample if it has been loaded.

Shape panel



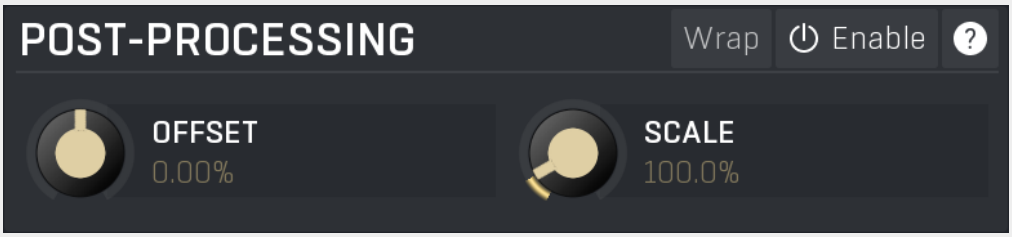
Shape panel contains parameters performing various transformations on the signal shape. Please note that most transformation require a significant amount of CPU resources, so you should not automate or modulate the signal shape if you are using them.

Harmonics panel



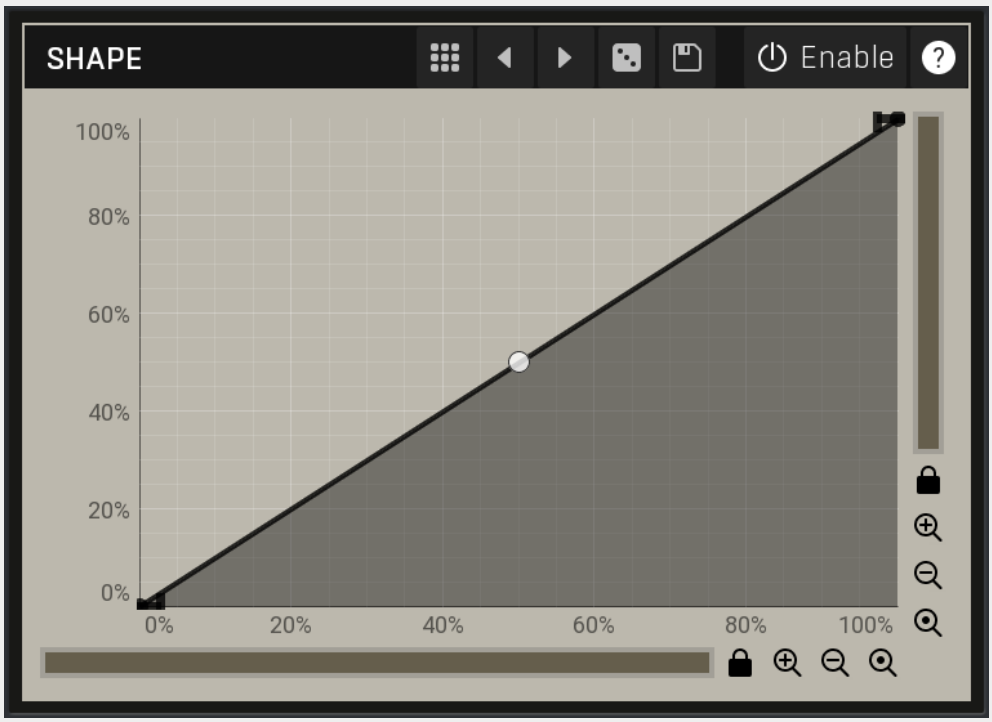
Harmonics panel lets you add separate harmonics of the original signal. The first one named **F** is the fundamental, the original signal. You can use the main **Phase** parameter to shift that.

Post-processing panel



Post-processing panel lets you post-process the shape after all the previous generator items.

Transformations



Shape transformation graph

Shape transformation graph lets you perform arbitrary modification of the graph shape. Basically this graph lets you modify the shape "in time". The Y axis represents the position in the source signal related to the position in the target signal. The best way to check what it does is simply to try it.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

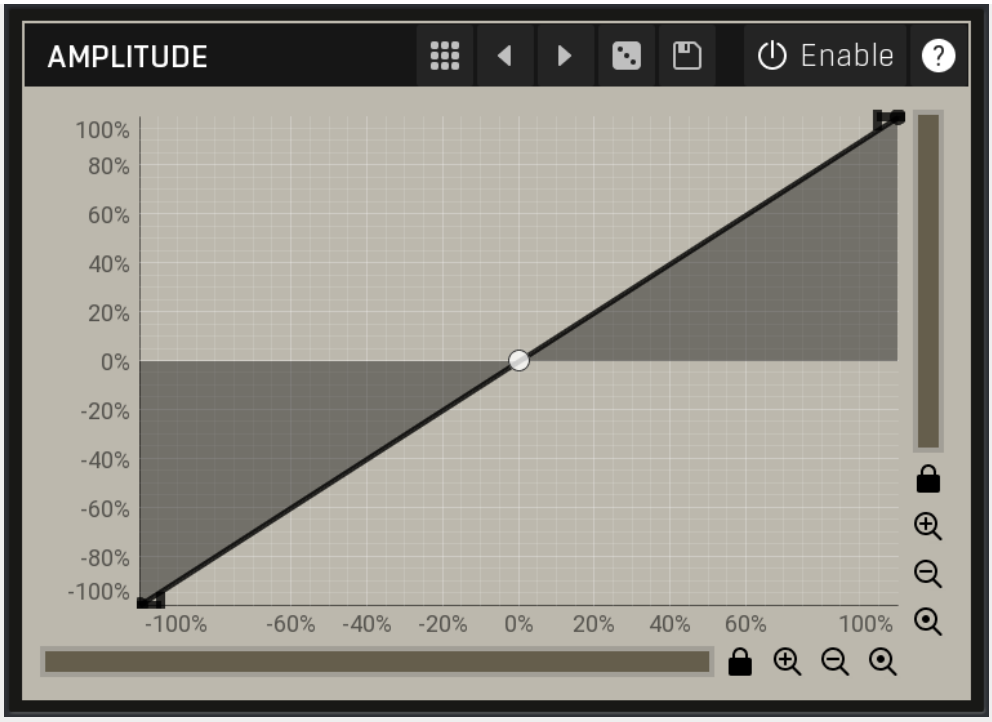
Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.



Amplitude transformation graph

Amplitude transformation graph lets you perform arbitrary modification of the graph amplitude. Basically this graph lets you modify the shape's level, vertical axis. The X axis represents the original values, the Y axis defines the resulting values. The best way to check what it does is simply to try it.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

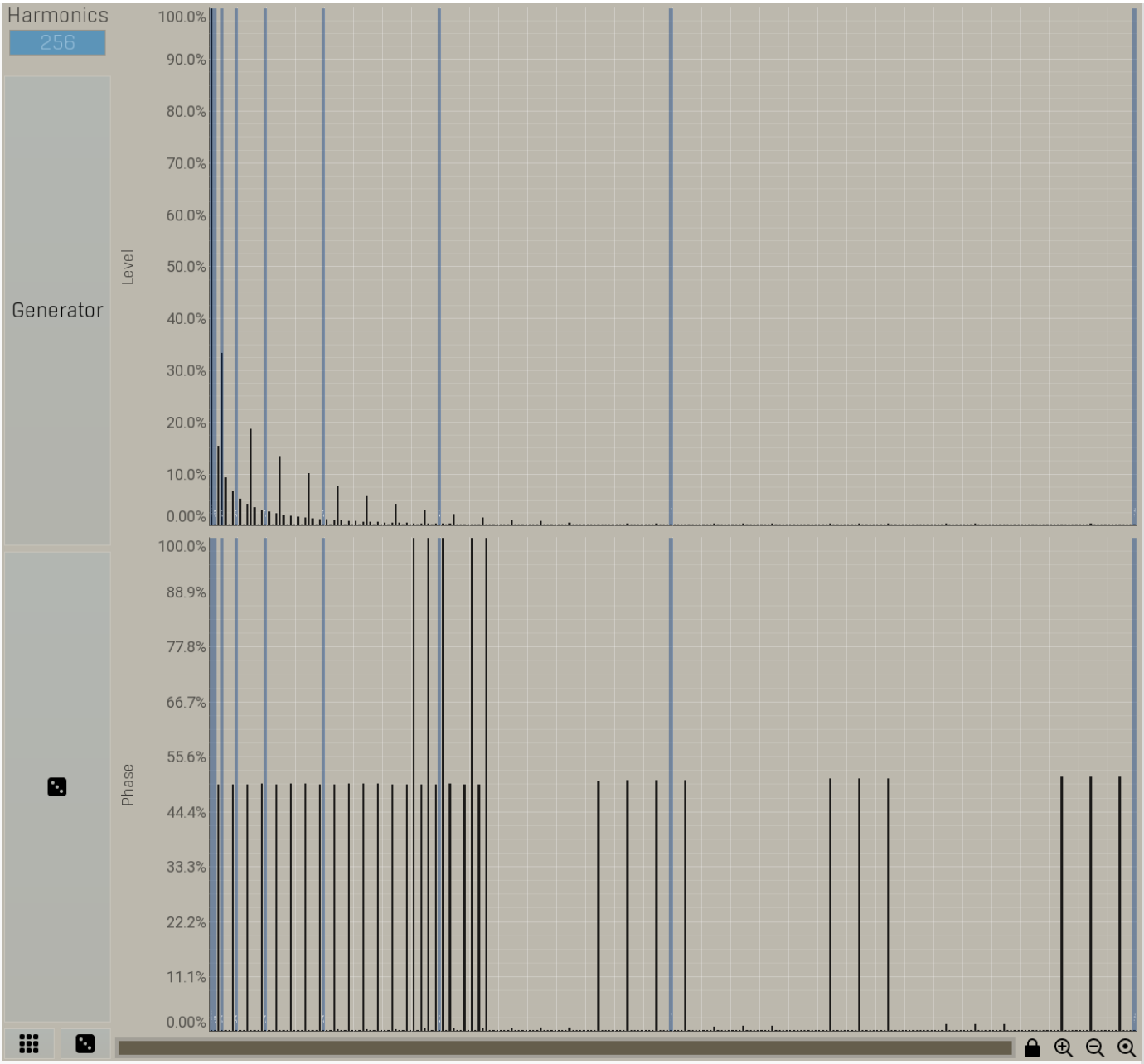
Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Signal generator in Harmonics mode



Signal generator in Harmonics mode works by generating the oscillator shape using individual harmonics. Essentially a harmonic is a sine wave. The first harmonic, known as the fundamental, fits once in the oscillator time period, hence it is the same as selecting sine wave in the **Normal mode**. The second harmonic fits twice, the third three times etc. In theory, any shape you create in normal mode can be converted into harmonics. However, this approach to signal generation needs an enormous number of harmonics, which is both inefficient to calculate and mostly hard to edit. Therefore, the harmonic mode can process up to 256 harmonics, which is enough for very complex spectrums, however it is still not enough to generate an accurate square wave for example. If your goal is to create basic shapes, it is better to use the normal mode.

It is nearly impossible to say how a particular curve will sound when used as a high-frequency oscillator in a synthesizer, just by looking at its shape. Harmonics mode, on the other hand, is directly related to human hearing and makes this process very simple. In general, the more harmonics you add, the richer the sound will be. The higher the harmonic, the higher the tone. Usually, one leaves the first harmonic enabled too, as this is the fundamental tone, however you may experiment with more dissonant sounds without it.

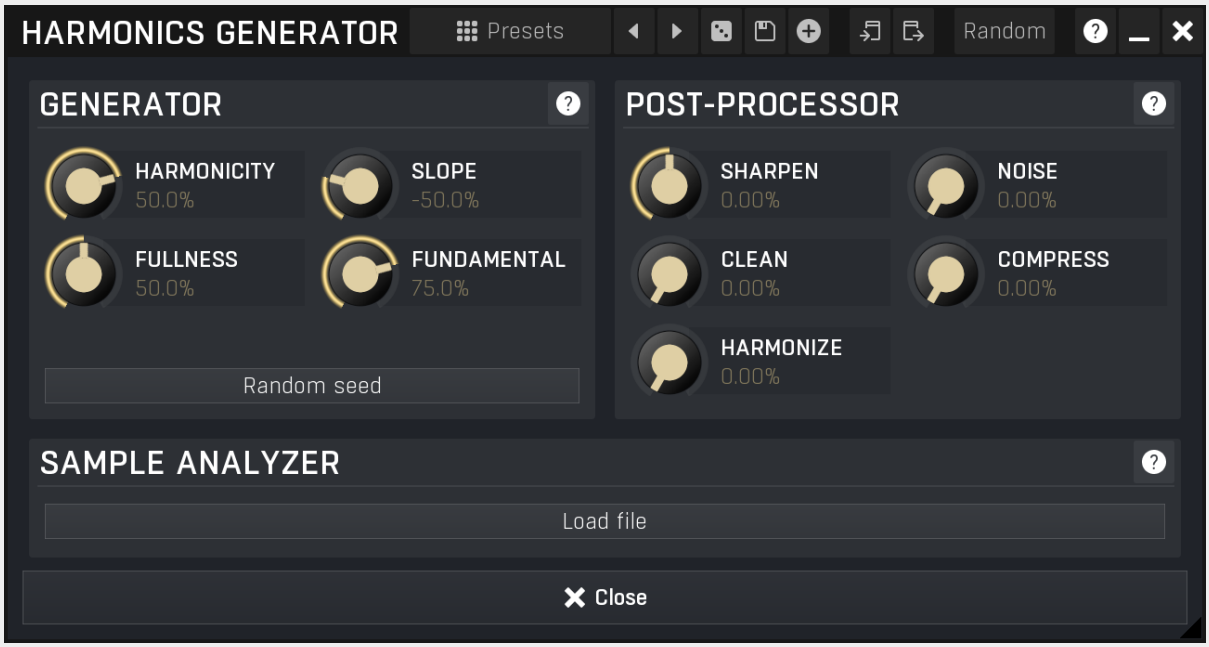
Editing harmonics can be time consuming unless you hear what you want, so a signal generator is also available. This great tool lets you generate a random spectrum by a single click. You can also open the **Generator** settings and edit its parameters, which basically control the audio properties in a more natural way - using parameters such as complexity, harmonicity etc.

Generator

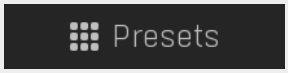
Generator

Generator button shows a powerful harmonics generator, which can create unlimited number of various timbres and even analyze a sample and extract harmonics from it.

Harmonics generator



Harmonics generator is a powerful tool, that can generate various harmonics-based timbres and even analyze a sample file and extract harmonics from it.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

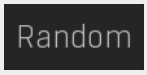
Right arrow button loads the next preset.

Randomize
Randomize button loads a random preset.

Save
Save button replaces the current preset.

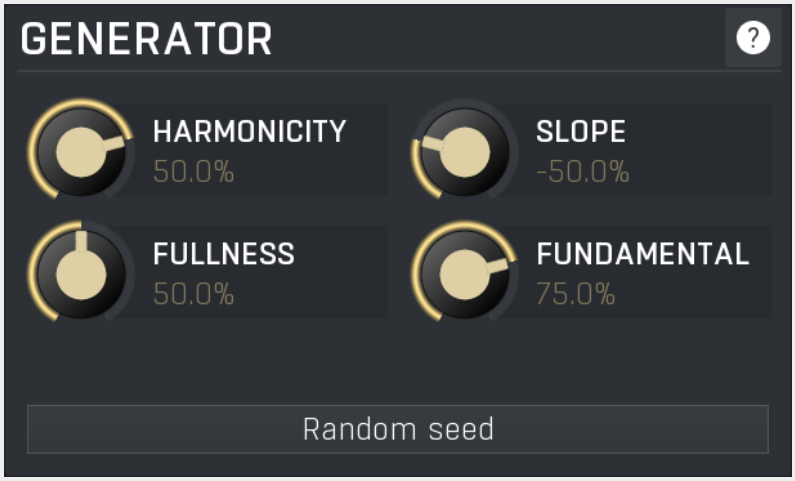
Copy
Copy button copies the settings onto the system clipboard.

Paste
Paste button loads the settings from the system clipboard.



Random
Random button generates random settings using the existing presets.

Generator panel



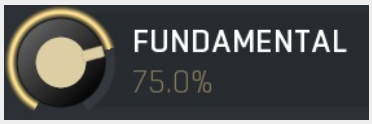
Generator panel contains parameters of the harmonics generator. By changing any of the parameters, the harmonics are changed, however only **Random seed** button changes the structure completely. The other parameters can be used to tweak the results.

Harmonicity
Harmonicity controls the ratio between natural harmonics and those which sound disharmonic (despite the title "harmonics"). Assuming that the 1st harmonic is the fundamental, 2nd harmonic is 1 octave above, 4th is 2 octaves above, both can be considered very natural. 3rd harmonic is 1 octave and a 5th above the fundamental, and is still pretty harmonic, but less than the octaves. 5th harmonic is 2 octaves and a major 3rd above the fundamental. Such a tone may sound very disharmonic, in minor scales for example. Higher harmonics are often very disharmonic and produce typical ringing timbres.
When harmonicity parameter is set to 100%, only octaves are allowed. By lowering the value more and more disharmonics are created and with 0% all frequencies are allowed. For values below 0% disharmonics are preferred, hence you can expect more ringing timbres.

Slope
Slope defines the amount of higher harmonics compared to lower ones. When 0%, the higher harmonics have the same levels as lower ones. Typically you use values below 0%, which attenuates the higher harmonics making the resulting sound darker. Similarly values above 0% make the sound brighter.

Fullness
Fullness controls the number of generated harmonics. With values around 0% the resulting timbers will contain only a

few harmonics making the sound clear. Higher values increase number of harmonics making the timbre rich.



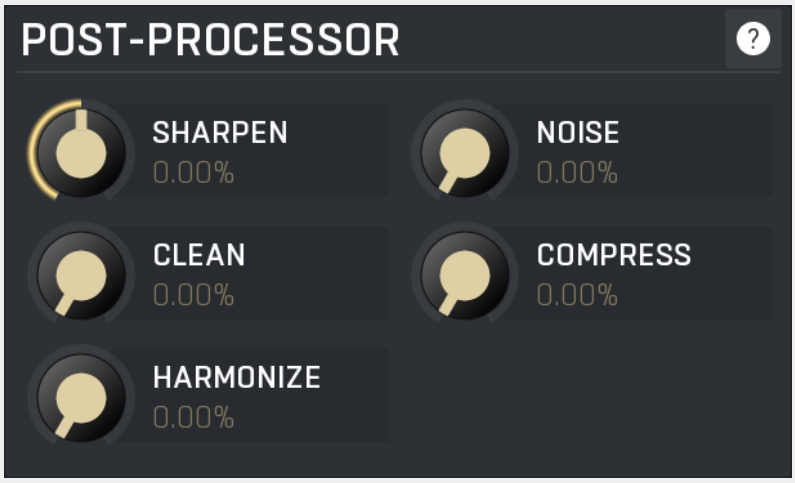
Fundamental

Fundamental controls the minimum level of the fundamental (the 1st harmonic). Most sounds have a very strong fundamental as it carries the pitch.

Random seed

Random seed button generates a new series of harmonics. Pressing this button will create a whole new timbre.

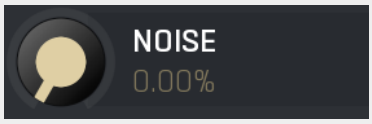
Post-processor panel



Post-processor panel contains parameters of the harmonics post-processor. The generator and sample analyzer first create a series of harmonics, the timbre. These harmonics are mixed depending on the **Sample ratio** parameter. After that the post-processor is engaged, which can further transform the harmonics in several ways.

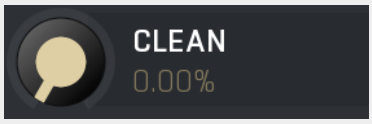
Sharpen

Sharpen is a sort of soft compression/expanding unit. Values below 0% decrease the level of quiet harmonics, while values above 0% increase their level.



Noise

Noise defines amount of noise added to the timbre. Noise can make the results dirty providing much richer timbres.

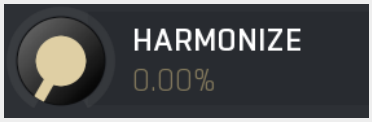


Clean

Clean controls the threshold of a gate. It basically attenuates or removes harmonics below this level making the output cleaner.

Compress

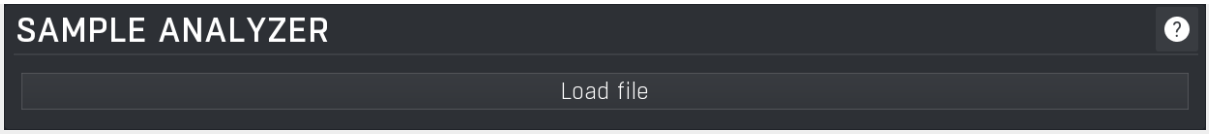
Compress reduces the dynamic range of the harmonics by increasing levels of the quiet ones, but keeping the levels of the loud ones.



Harmonize

Harmonize creates additional higher harmonics from existing ones. This is especially useful to transform rich dirty disharmonic timbres into similarly rich but more harmonic timbres.

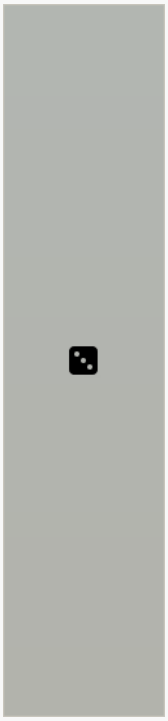
Sample analyzer panel



Sample analyzer panel contains parameters of the sample analyzer. If there is no sample loaded, the sample analyzer is turned off. The analyzer takes the selected sample and a position within it, analyses one period of the signal waveform and produces the output set of harmonics. You can then combine these harmonics with the output of the generator using **Sample ratio** parameter.
The sample itself is not store with the plugin settings. Instead the path to the target sample file is stored along with the analyzed harmonics. If the sample file is not available, you cannot modify the analysis parameters and the last analyzed harmonics are used. This means that you actually don't need to have the sample file available on the computer on which you are using the settings.

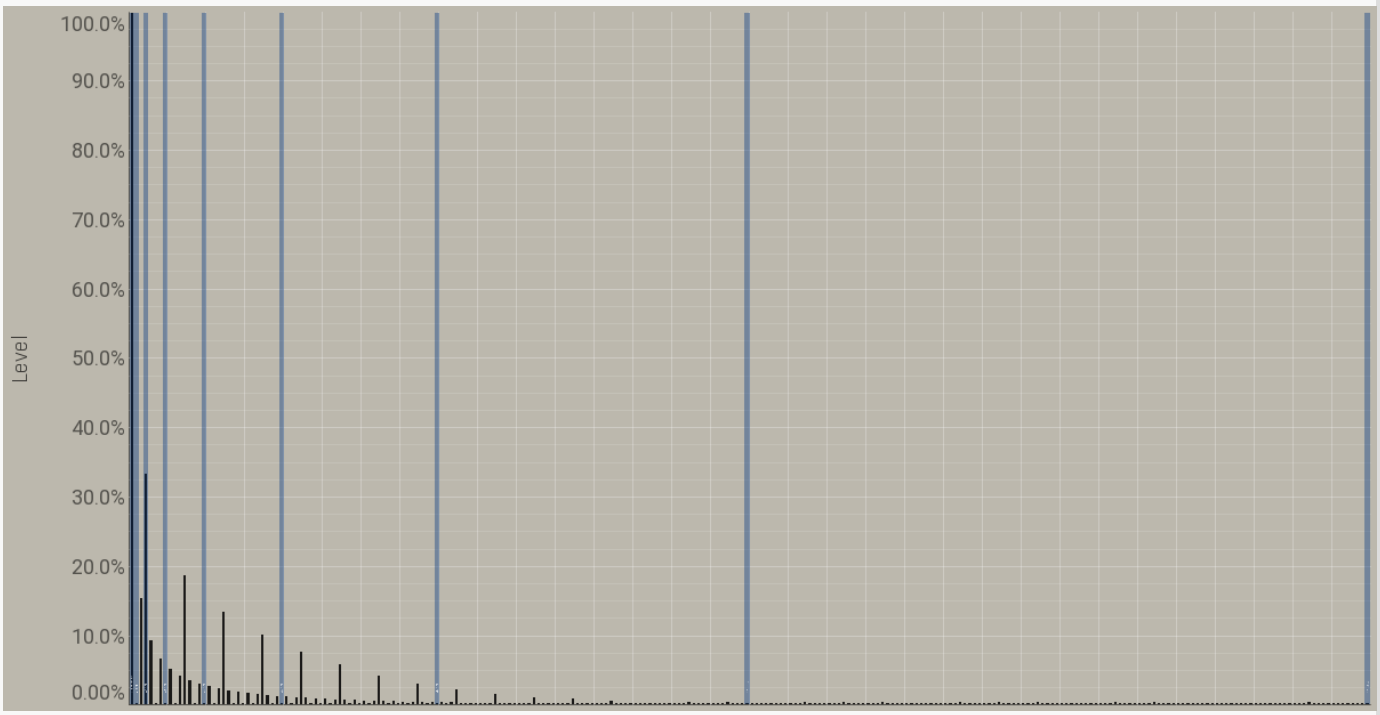
Load file

Load file button lets you select a sample file to analyse.



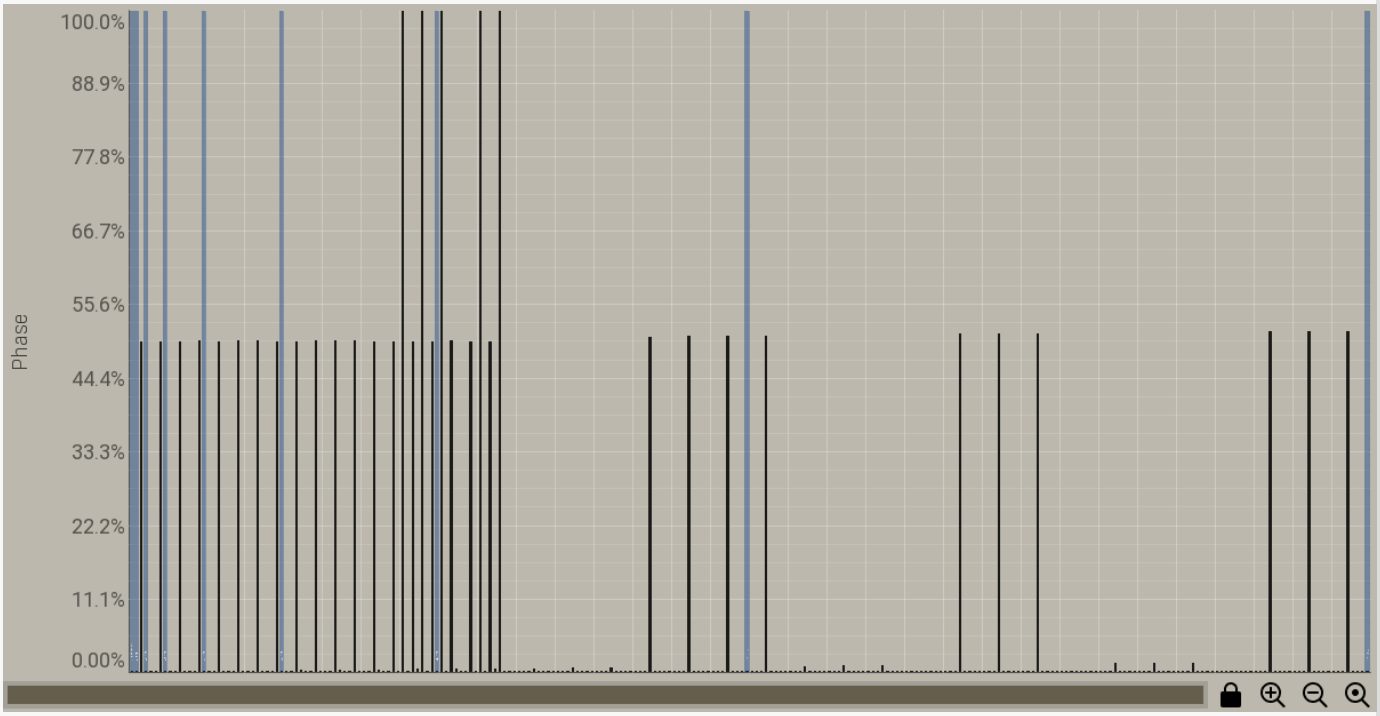
Randomize

Randomize button selects random parameters for the harmonics generator, so you can use it to get a random sound character instantly. Hold **Ctrl** to slightly modify existing generator settings instead of completely changing them.



Magnitudes graph

Magnitudes graph contains the levels of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.



Phases graph

Phases graph contains the phases of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.

Follower panel

Follower panel contains the parameters defining how the plugin determines the level of the source signal. Each follower operates on the original input or side-chain.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

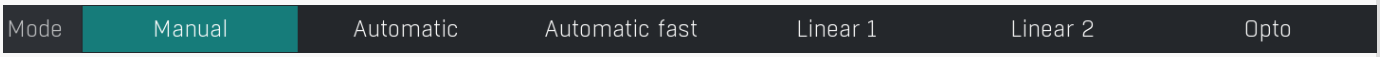
Save
Save button replaces the current preset.

Copy
Copy button copies the settings onto the system clipboard.

Paste
Paste button loads the settings from the system clipboard.

Random
Random button generates random settings using the existing presets.

Side-chain
Side-chain button activates the side-chain input as the source for level measurement.



Release mode

Release mode defines how the plugin performs when decreasing level. In **manual mode** this is based only on the **release time**, which is suitable for most cases when the signal has constant characteristics. Automatic release modes can adapt to signals with unstable characteristics.

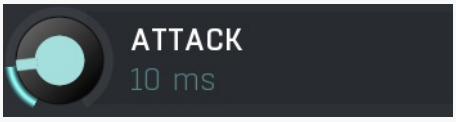
Automatic and **Automatic fast** modes: the longer the level stays above the threshold, the longer the release time will be and thus, the longer it will take to move below the threshold and end the release stage. The idea is that if the input is loud for some time, it will most likely stay that way for some more time, hence it should be stabilized to avoid unnecessary temporary fluctuations, which could result in pumping.
Both automatic modes increase the release time when the input signal is above the threshold and vice versa. The speed of the increase depends on the **Auto speed** parameter. Automatic fast mode uses full speed immediately after crossing the threshold, automatic mode varies the speed according to the current signal level.

For example, when a guitarist plays softly, the level is low and fluctuates around the threshold and the release time gets slower. So the processor quickly responds to sudden changes. However, when the guitarist starts playing a solo, the level rises and, the longer the solo is, the longer the release time becomes, hence the response becomes slower avoiding unnecessary fluctuations (pumping) when the solo contains small silent sections.

Linear 1 and **Linear 2** modes: the higher the level is, the longer the release. The idea is that if the input is very loud, it will probably stay that way for some time, so it is wise to keep the levels up too. This is similar to the automatic modes, however the main factor is not how long the level is high, but how high it is.
Below the threshold the release time is the same as the attack time, above the threshold the release time rises from the attack time up to the specified release time parameter. Linear 1 mode usually provides higher release times than does Linear 2.

Opto mode: the higher the level is, the shorter the release. So this is kind of the opposite of linear modes. The idea is, that you are expecting short transients, which you wish to deal with. Normally the higher the level would get in such a transient, the longer it would take to get the level below the threshold, so, when used in a compressor for example, these transients would cause unnecessary compression in the sustain stage. The opto detector lowers the level quickly, minimizing the amount of compression in the sustain stage.

*For example, let's say you are compressing a full drumset, but there is a very dominant sharp and short hi-hat sound, so it is appropriate to have short release times. You would use **Opto** mode. But the rest of the drumset deserves a softer treatment, so you want to keep longer release times. Use one of the other modes.*



Attack
Attack defines the attack time, that is how quickly the level detector increases the measured input level. When the input peak level is higher than the current level measured by the detector, the detector moves into the attack mode, in which the measured level is increased depending on the input signal. The higher the input signal, or the shorter the attack time, the faster the measured level rises. Once the measured level exceeds the **Threshold** then the dynamics processing (compression, limiting, gating) will start.

There must be a reasonable balance between attack and **release** times. If the attack is too long compared to the release, the detector will tend to keep the measured level low, because the release would cause that level to fall too quickly. In most cases you may expect the attack time to be shorter than the release time.

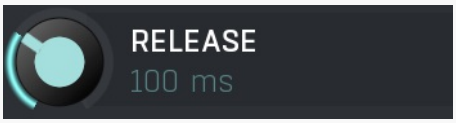
To understand the working of a level detector, it is best to cover the typical cases:

*In a **compressor** the attack time controls how quickly the measured level moves above the threshold and the processor begins compressing. As a result, a very short attack time will compress even the beginning transient of a snare drum for example, hence it would remove the punch. With a very long attack time the measured level may not even reach the threshold, so the compressor may not do anything.*

*In a **limiter** the attack becomes a very sensitive control, defining how much of the signal is limited and how much of it becomes saturated/clipped. If the attack time is very short, limiting starts very quickly and the limiter catches most peaks itself and reduces them, providing lower distortion, but can cause pumping. On the other hand, a higher attack setting (typically above 1ms) will let most peaks through the limiter to the subsequent in-built clipper or saturator, which causes more distortion of the initial transient, but less pumping.*

*In a **gate** the situation is similar to a compressor - the attack time controls how quickly the measured level can rise above the threshold at which point the gate opens. In this case you will usually need very low attack times, so that the gate reacts quickly enough. The inevitable distortion can then be avoided using look-ahead and hold parameters.*

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level rising, use a shorter attack time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 10000 ms, default 10 ms



Release

Release defines the release time, that is how quickly the level detector decreases the measured input level. The shorter the release time, the faster the response is. Once the attack stage has been completed, when the input peak level is lower than the current level measured by the detector, the detector moves into the release mode, in which the measured level is decreased depending on the input signal. The lower the input signal, or the shorter the release time, the faster the measured level drops. Once the measured level falls under the **Threshold** then the dynamics processing (compression, limiting, gating) will stop.

There must be a reasonable balance between **attack** and release times. If the attack is too long compared to release, the detector would tend to keep the level low, because release would cause the level to fall too quickly. Hence in most cases you may expect the attack time to be shorter than the release time.

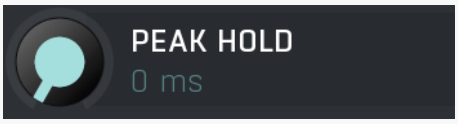
To understand the working of a level detector, it is best to cover the typical cases:

*In a **compressor** the release time controls how quickly the measured level falls below the threshold and the compression stops. As a result a very short release time makes the compressor stop quickly, for example, leaving the sustain of a snare drum intact. On the other hand, a very long release keeps the compression working longer, hence it is useful to stabilize the levels.*

*In a **limiter** the release time keeps the measured level above the limiter threshold causing the gain reduction. Having a very long release time in this case doesn't make sense as the limiter would be working continuously and the effect would be more or less the same as simply decreasing the input gain manually. However too short a release time lets the limiter stop too quickly, which usually causes distortion as the peaks through the limiter to the subsequent in-built clipper or saturator. Hence release time is used to avoid distortion at the expense of decreasing the output level.*

*In a **gate** the situation is similar to a compressor - the release time controls how quickly the measured level can fall below the threshold at which point the gate closes. Having a longer release time in a gate is a perfectly acceptable option. The release time will basically control how much of the sound's sustain will pass.*

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level falling, use a shorter release time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 10000 ms, default 100 ms



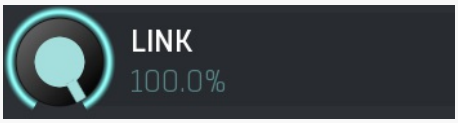
Peak hold

Peak hold defines the time that signal level detector holds its maximum before the release stage is allowed to start. As an example, you can imagine that when an attack stage ends there can be an additional peak hold stage and the level is not yet falling, before the release stage starts. This is true only when **true peak** mode is enabled (check the advanced detector settings if available).

It is often used in **gates** to avoid the gated level falling below the threshold too quickly, while having short release times. If you want the gate to close quickly, you need a short release time. But in that case the ending may be too abrupt and even

cause some distortion. So you use the peak hold to delay the release stage.

It is also used along with **look-ahead** to avoid distortion in **limiters and compressors**. If you need a very short attack, the attack stage may be too quick and cause distortions. In limiters this attack time is often 0ms, in which case it becomes a clipper. Setting look-ahead and peak hold to the same value will make the detector move ahead in time, so that it can react to attack stages before they actually occur and yet hold the levels for the actual signal to come.
Range: 0 ms to 10000 ms, default 0 ms



Link channels

Link channels controls how much the signal level for each channel is controlled by the other channels. With 0% the link is disabled and each channel is not affected by the other channels at all. This is suitable to balance stereo channels, for example. With 100% the link is enabled and all channels are controlled by levels of all channels equally (that is the average level of those channels), therefore the processor will apply the same amount of processing on all channels. This is the default in most cases as it preserves relative levels between the channels.
Range: 0.00% to 100.0%, default 100.0%

RMS length

RMS length smoothes out the values of the input levels (not the input itself), such that the level detector receives the pre-processed signal without so many fluctuations. When set to its minimum value the detector becomes a so-called "peak detector", otherwise it is an "RMS detector".

When you look at a typical waveform in any editor, you can see that the signal is constantly changing and contains various transient bursts and separate peaks. This is especially noticeable with rhythmical signals, such as drums. Trying to imagine how a typical attack/release detector works with such a wild signal may be complex, at least. RMS essentially takes the surrounding samples and averages them. The result is a much smoother signal with fewer individual peaks and short noise bursts.

RMS length controls how many samples are taken to calculate the average. It stabilizes the levels, but it also causes a slower response time. As such it is great for mastering, when you want to lower the dynamic range in a very subtle way without any instabilities. However, it is not really desirable for processing drums, for example, where the transient bursts may actually be individual drum hits, hence it is usually recommended to use peak detectors for percussive instruments.

Note that the RMS detector has 2 modes - a simplified approximation is used by default, and a true RMS is processor can be enabled from the advanced settings (if provided). Both respond differently, neither of them is better than the other, they are simply different.
Range: Peak to 10000 ms, default Peak

Release min

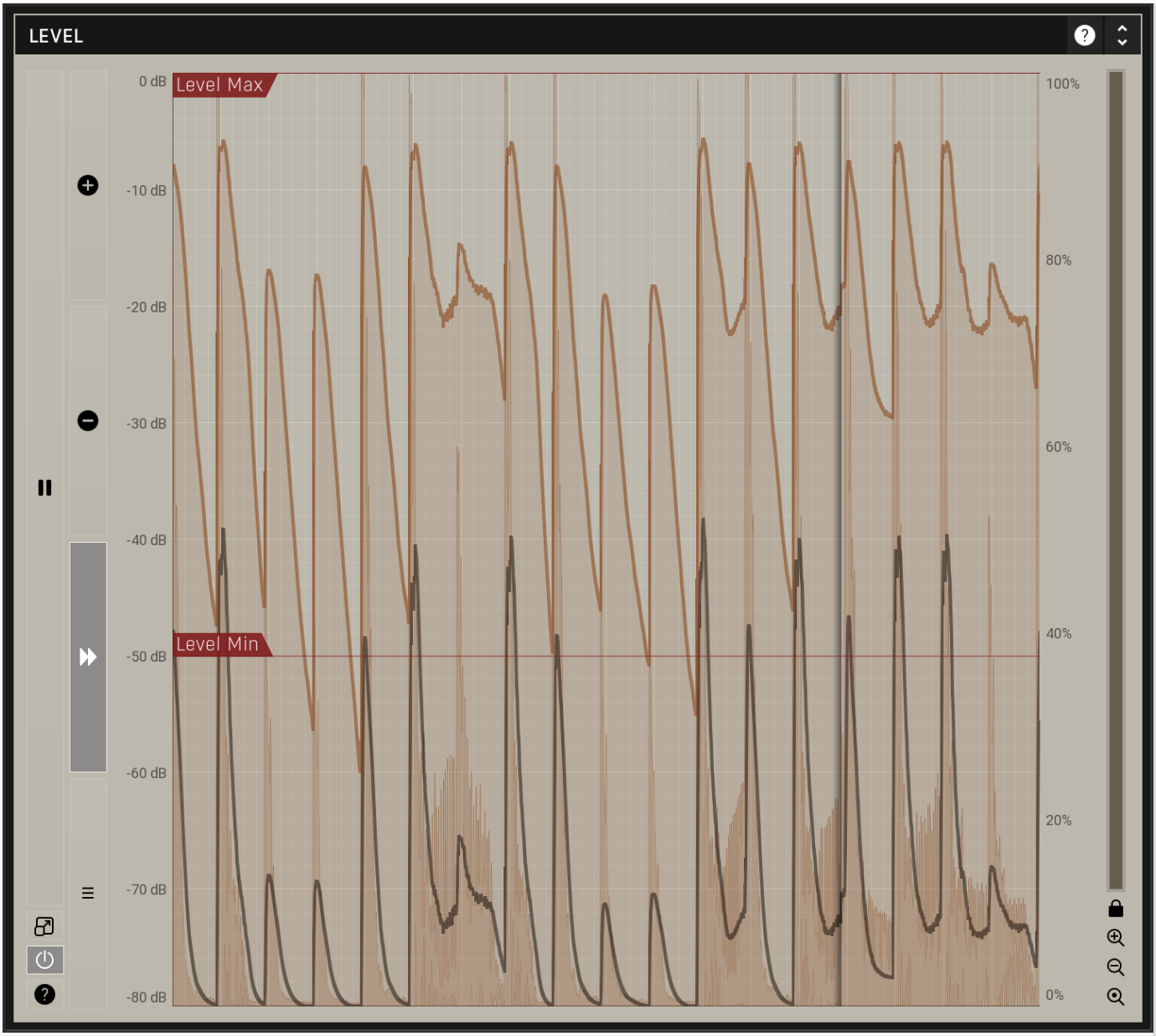
Release min defines the minimum release time and is used with automatic release modes. Note that release min may actually be set higher than **Release**, in which case the behaviour depends on the selected release mode. If it is set at the original 0ms, the actual value equals to **Attack** parameter and it is at least 1ms to avoid unnecessary distortion.
Range: 0 ms to 10000 ms, default 0 ms

Auto speed

Auto speed defines how quickly the automatic release works. Specifically how much the release time increases/decreases per second. It is relevant only in **automatic** release modes. *For example if you set it to 5000ms, the release time will be able to increase by 1000ms in 5000ms, when incoming signal exceeds the lowest threshold.*
Range: 1.0 ms to 10000 ms, default 1000 ms

Delay

Delay defines how much the follower output should be delayed. It is a powerful way to keep attacks intact or to combine multiple followers to get more complex responses for example.
Range: 0 ms to 10000 ms, default 0 ms



Level panel

Level panel contains the metering system showing the follower level. It is indispensable when setting up the follower.
The **orange graph** displays the measured level, which depends on the detector parameters, such as **Attack** or **Release**. In most cases you will want to set the follower so that it responds well to the full range of the audio material. After the detector parameters the level range is the next most important and is available via **Level min** and **Level max** parameters(these can be also adjusted directly from the analyser). In most cases you will want the minimum level to lie just below the lowest signal peaks and maximum level just above the highest peaks.
The **white graph** displays the output modulator values. It includes all the processing that affects the modulator including **LFO modulation** and **Project** features.

Pause

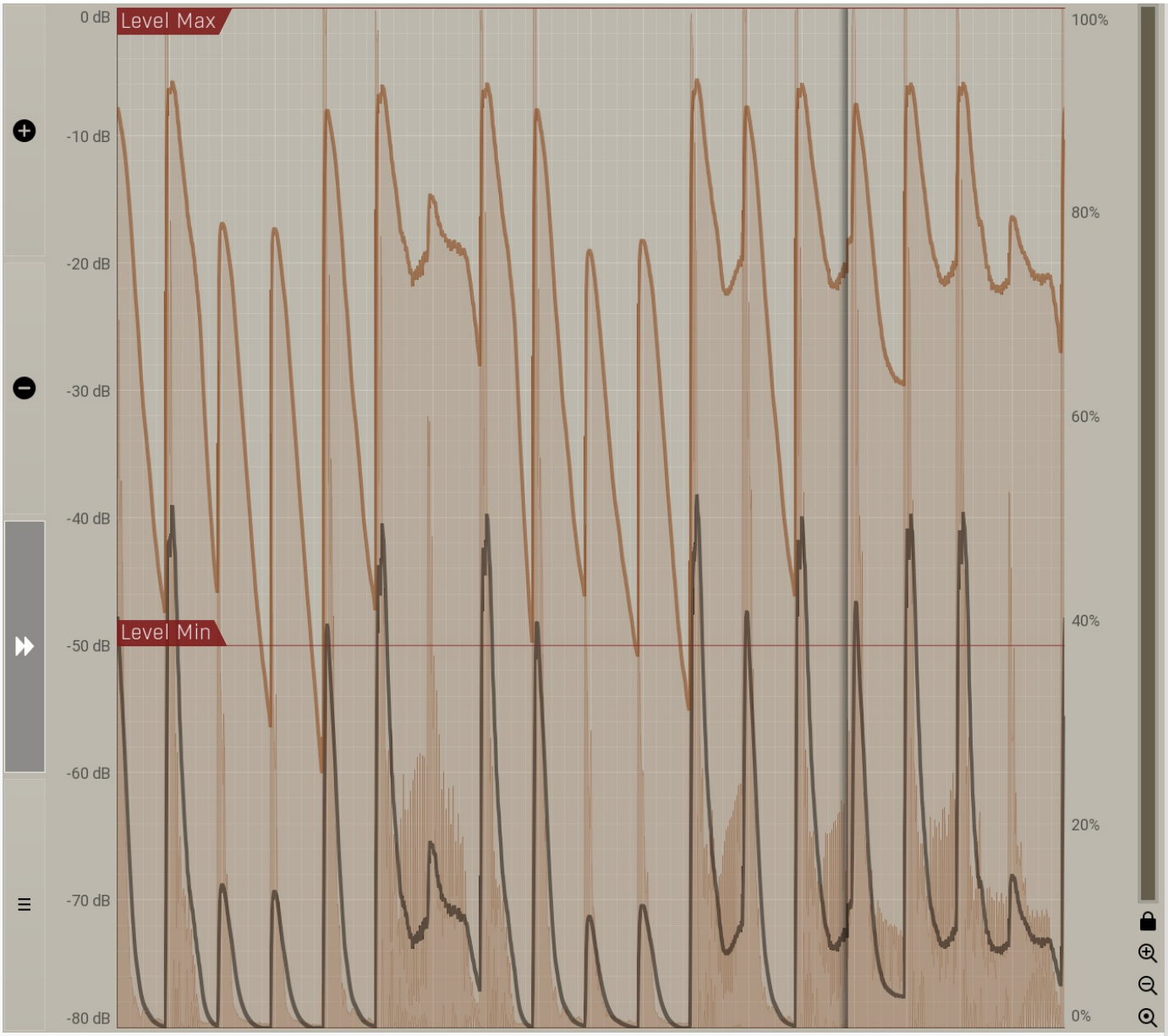
Pause button pauses the processing.

Popup

Popup button shows a pop-up window and moves the whole metering / time-graph system into it. This is especially useful in cases where you cannot enlarge the meters within the main window or such a task is too complicated. The pop-up window can be arbitrarily resized. In metering mode it is useful for easier reading from a distance for example. In time-graph mode it is useful for getting higher accuracy and a longer time perspective.

Enable

Enable button enables or disables the metering system. You can disable it to save system resources.



Time-graph view

Time-graph view shows the measurements over a period of time.

Plus

Plus button increases the time-graph speed (reduces the period that is displayed).

Minus

Minus button decreases the time-graph speed (increases the period that is displayed).

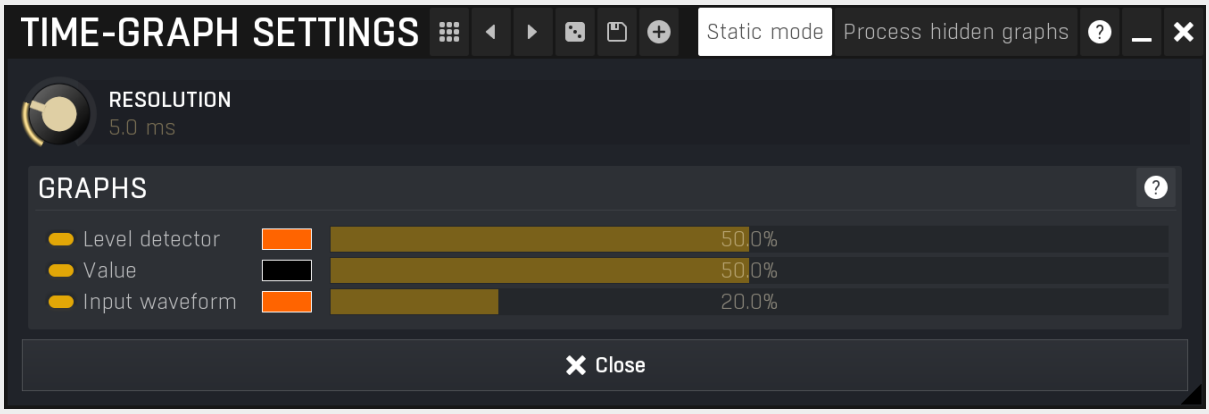
Rewind

Rewind button enables or disables the time-graph static mode. In static mode the graphs are fixed and the current position cycles from left to right; otherwise the graphs move from right to left and the current position is fixed (at the right-hand side).

Menu

Menu button displays the time-graph settings. In this window you can control which graphs are displayed, the speed and other relevant parameters.

Time-graph settings



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

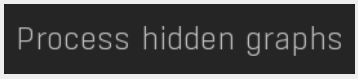
Save

Save button replaces the current preset.

Static mode

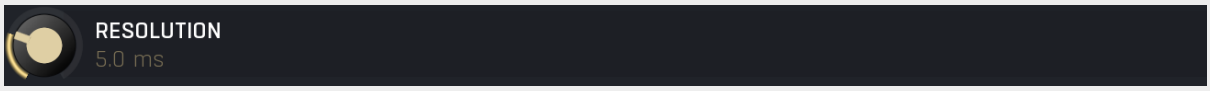
Static mode stops the graph from scrolling to the left and makes the graph refresh from left to right instead.

When this is disabled, the entire graph is moving from right to left as the incoming audio is processed. This may make it hard to spot the actual details, which is where the static mode comes to the rescue. Static mode is the default state and in most cases is more practical.



Process hidden graphs

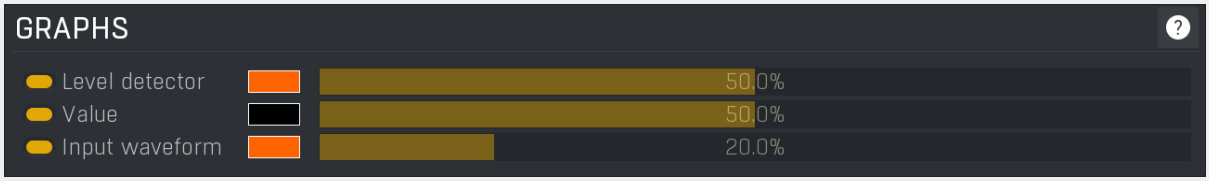
Process hidden graphs enables measurement of graphs which are actually disabled in the view. This may come handy if you need to repeatedly show and hide several graphs. With this mode disabled, which it is by default, the processor saves CPU resources by computing only those measurements that are actually visible. However, when you show a currently hidden graph, no measurements are available, so you will need to wait for the graph to be generated from the incoming signal. If you enable this option, the graph will be available immediately after you make it visible.



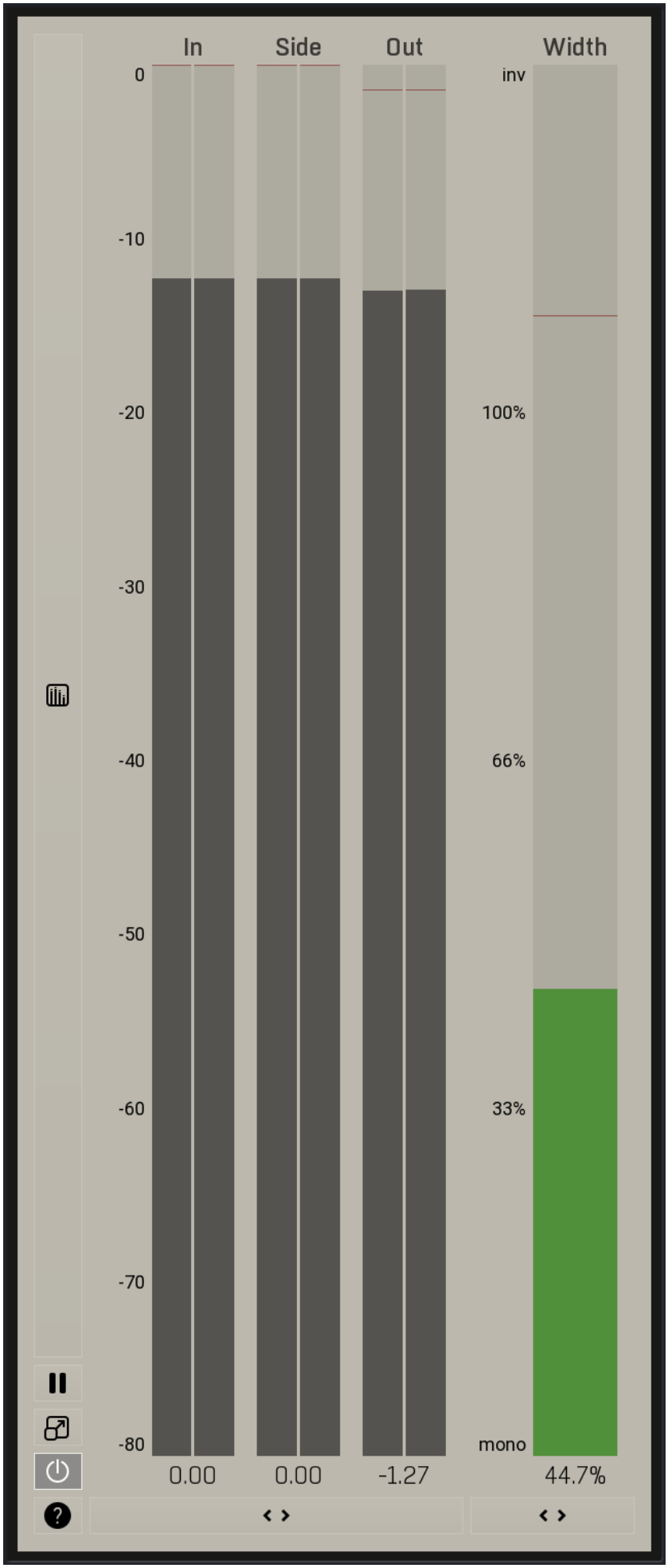
Resolution

Resolution controls the time it takes for the graph to move one pixel. Therefore this actually controls the display speed.

Graphs panel



Graphs panel contains all available graphs and lets you show or hide each of them, and change their visual properties.



Global meter view

Global meter view provides a powerful metering system. If you do not see it in the plugin, click the **Meters** or **Meters & Utilities** button to the right of the main controls. The display can work as either a classical level indicator or, in time graph mode, show one or more values in time. Use the first button to the left of the display to switch between the 2 modes and to control additional settings, including pause, disable and pop up the display into a floating window. The meter always shows the actual channels

being processed, thus in M/S mode, it shows mid and side channels.
In the classical level indicators mode each of the meters also shows the recent maximum value. Click on any one of these values boxes to reset them all.

In meter indicates the total input level. The input meter shows the audio level before any specific processing (except potential oversampling and other pre-processing). It is always recommended to keep the input level under 0dB. You may need to adjust the previous processing plugins, track levels or gain stages to ensure that it is achieved.

As the levels approach 0dB, that part of the meters is displayed with **red** bars. And recent peak levels are indicated by single bars.

Out meter indicates the total output level. The output meter is the last item in the processing chain (except potential downsampling and other post-processing). It is always recommended to keep the output under 0dB.

As the levels approach 0dB, that part of the meters is displayed with **red** bars. And recent peak levels are indicated by single bars.

Width meter shows the stereo width at the output stage. This meter requires at least 2 channels and therefore does not work in mono mode. Stereo width meter basically shows the difference between the mid and side channels.
When the value is **0%**, the output is monophonic. From 0% to 66% there is a green range, where most audio materials should remain.
From 66% to 100% the audio is very stereophonic and the phase coherence may start causing problems. This range is colored blue. You may still want to use this range for wide materials, such as background pads. It is pretty common for mastered tracks to lie on the edge of green and blue zones.
Above 100% the side signal exceeds the mid signal, therefore it is too monophonic or the signal is out of phase. This is marked using red color. In this case you should consider rotating the phase of the left or right channels or lowering the side signal, otherwise the audio will be highly mono-incompatible and can cause fatigue even when played back in stereo.
For most audio sources the width is fluctuating quickly, so the meter shows a 400ms average. It also shows the temporary maximum above it as a single coloured bar.
If you right click on the meter, you can enable/disable loudness pre-filtering, which uses EBU standard filters to simulate human perception. This may be useful to get a more realistic idea about stereo width. However, since humans perceive the bass spectrum as lower than the treble, this may hide phase problems in that bass spectrum.

Time graph

Time graph button switches between the metering view and the time-graphs. The metering view provides an immediate view of the current values including a text representation. The time-graphs provide the same information over a period of time. Since different time-graphs often need different units, only the most important units are provided.

Pause

Pause button pauses the processing.

Popup

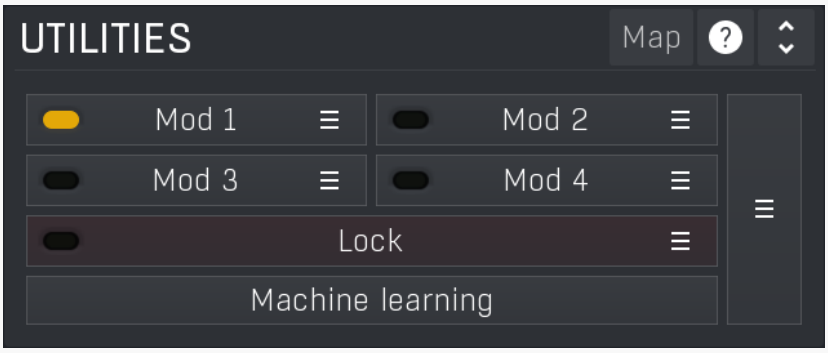
Popup button shows a pop-up window and moves the whole metering / time-graph system into it. This is especially useful in cases where you cannot enlarge the meters within the main window or such a task is too complicated. The pop-up window can be arbitrarily resized. In metering mode it is useful for easier reading from a distance for example. In time-graph mode it is useful for getting higher accuracy and a longer time perspective.

Enable
Enable button enables or disables the metering system. You can disable it to save system resources.

Collapse
Collapse button minimizes or enlarges the panel to release space for other editors.

Collapse
Collapse button minimizes or enlarges the panel to release space for other editors.

Utilities



Map
Map button displays all current mappings of modulators, multiparameters and MIDI (whichever subsystems the plugin provides).

Modulator
Modulator button displays settings of the modulator. It also contains a checkbox, to the left, which you can use to enable or disable the modulator. Click on it using your right mouse button or use the **menu button** to display an additional menu with learning capabilities - as described below.

Menu
Menu button shows the **smart learn** menu. You can also use the right mouse button anywhere on the modulator button.

Learn activates the learning mode and displays "REC" on the button as a reminder, **Clear & Learn** deletes all parameters currently associated with the modulator, then activates the learning mode as above. After that every parameter you touch will be associated to the modulator along with the range that the parameter was changed. Learning mode is ended by clicking the button again.

In smart learn mode the modulator does not operate but rather records your actions. You can still adjust every automatable parameter and use it normally. When you change a parameter, the plugin associates that parameter with the modulator and also records the range of values that you set.

For example, to associate a frequency slider and make a modulator control it from 100Hz to 1KHz, just enable the smart learn mode, click the slider then move it from 100Hz to 1KHz (you can also edit the range later in the modulator window too). Then disable the learning mode by clicking on the button.

Menu
Menu button displays additional menu containing features for modulator presets and randomization.

Lock
Lock button displays the settings of the global parameter lock. Click on it using your left mouse button to open the Global Parameter Lock window, listing all those parameters that are currently able to be locked.

Click on it using your right mouse button or use the **menu button** to display the menu with learning capabilities - **Learn** activates the learning mode, **Clear & Learn** deletes all currently-lockable parameters and then activates the learning mode. After that, every parameter you touch will be added to the lock. Learning mode is ended by clicking the button again.
The On/Off button built into the Lock button enables or disables the active locks.

Machine learning

Machine learning is a very advanced feature, which lets you set the plugin parameters to make it produce specific results. It is designed ONLY for very advanced users!

First you need to select the original audio file. Then you select another audio file, which is the desired output. Finally you select a list of multiparameters (MPs) to set. The machine learning system will adopt all your CPU cores and try to find the most matching settings.

Machine learning is always a very alchemistic thing. You need to find proper settings for "comparing" the audio, that may be much more difficult, than it may seem, since for example comparing waveforms may be often unusable if phase changes are involved. Also the number of MPs to use is an important factor - the more MPs you use, the more combinations there are, so the plugin may have more versatility at its disposal, but it may also require significantly more CPU power to find a reasonable solution. The test signal you are processing is another important factor you may need to experiment with. The final solution may also be quite cryptic, it may be difficult to understand how it is possible that these settngs lead to the required results. In general you may expect to spend a huge number of time settings this up and it is no exception that you may need to run the learning for days on many cores with the hopes to find a good solution and even that is no guarantee of success.

Collapse

Collapse button minimizes or enlarges the panel to release space for other editors.

Multiparameter

Multiparameter button displays settings of the multiparameter. The multiparameter value can be adjusted by dragging it or by pressing Shift and clicking it to enter a new value from the virtual keyboard or from your computer keyboard.

Click on the button using your left mouse button to open the **Multiparameter** window where all the details of the multiparameter can be set. Click on it using your right mouse button or click on the **menu button** to the right to display an additional menu with learning capabilities - as described below.

Menu

Menu button shows the **smart learn** menu. You can also use the right mouse button anywhere on the multiparameter button.

Learn attaches any parameters, including ranges. Click this, then move any parameters through the ranges that you want and click the multiparameter button again to finish. While learning is active, "REC" is displayed on the multiparameter button and learning mode is ended by clicking the button again.

Clear & Learn clears any parameters currently in the list then attaches any parameters, including ranges. Click this, then move any parameters through the ranges that you want and click the multiparameter button again to finish. While learning is active, "REC" is displayed on the multiparameter button and learning mode is ended by clicking the button again.

Reset resets all multiparameter settings to defaults.

Quick Learn clears any parameters currently in the list, attaches one parameter, including its range and assigns its name to the multiparameter. Click this, then move one parameter through the range that you want.

Attach MIDI Controller opens the MIDI Settings window, selects a unused parameter and activates MIDI learn. Click this then move the MIDI controller that you want to assign.

Reorder to ... lets you change the order of the multiparameters. This can be useful when creating active-presets. Please note that this feature can cause problems when one multiparameter controls other multiparameters, as these associations will not be preserved and they will need to be rebuilt.

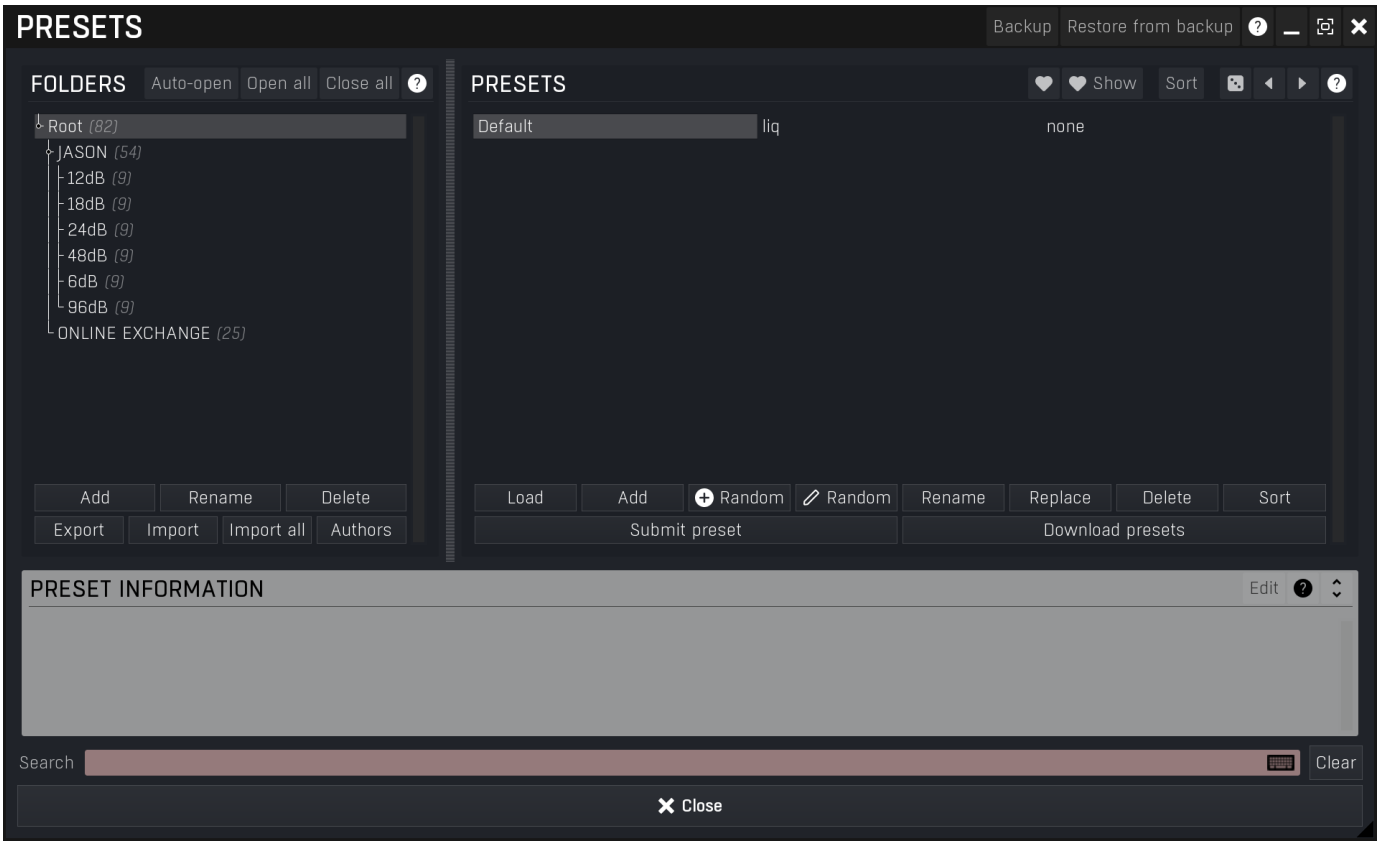
In learning mode the multiparameter does not operate but rather records your actions. You can still adjust every automatable parameter and use it normally. When you change a parameter, the plugin associates that parameter with the multiparameter and also records the range of values that you set.

For example, to associate a frequency slider and make a multiparameter control it from 100Hz to 1KHz, just enable the smart learn mode, click the slider then move it from 100Hz to 1KHz (you can also edit the range later in the Multiparameter window too). Then disable the learning mode by clicking on the button.

Collapse

Collapse button minimizes or enlarges the panel to release space for other editors.

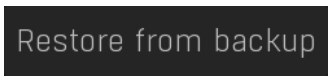
Preset selector



Preset management window provides management for your presets.

Backup

Backup button lets you backup presets for all MeldaProduction software into a single file, so you can transfer it to a different machine and restore the presets there for example.



Restore from backup

Restore from backup button lets you restore presets for all MeldaProduction software from a single file created by the **Backup** button.

Folders tree

Folders tree lets you organize your presets into any number of folders. Use the buttons at the bottom of the window to create, rename or delete sub-folders. Note that these are not actual files & folders on disk, but are records in the preset database.

Auto-open **Auto-open**

Auto-open switch makes the tree automatically open selected items, so that all sub-folders are visible, whenever you select one. This makes it easier to browse through large structures containing many folders. The switch also makes the browser show all presets available in the selected folder including all sub-folders (except when you select the root folder).

Open all **Open all**

Open all button expands the whole tree, so you can see all of the folders. This may be handy when editing large preset structures.

Close all **Close all**

Close all button collapses the whole tree except for the root folder. This may be handy when editing large preset structures.

Add **Add**

Add button creates a new folder in the tree

Rename **Rename**

Rename button lets you rename the selected folder.

Delete **Delete**

Delete button deletes the folder including all the presets and subfolders in it.

Export **Export**

Export button lets you export the selected folder including all presets and sub-folders into a file, which you can then transfer to any computer. Or just use as a back-up.

Import **Import**

Import button lets you import a file containing presets and sub-folders and add it to the selected folder. The importer will ask you whether to destroy the original contents, so that the new presets replace previous ones, or to keep both.

Import all **Import all**

Import all button lets you import all files from the specified directory and add them to the selected folder.

Authors

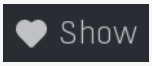
Authors displays a report of the authors creating the presets.

Presets list

Presets list contains all presets available in the selected folder. **Double-click** on a preset or use **Load** button to load a preset. Use the buttons at the bottom of the list to perform additional changes. Please note that these are not actual files & folders on disk, but are records in the preset database.

Favourite

Favourite button toggles the 'favourite' indicator for the selected preset.



Show

Show button shows only the favourite presets and hides the others.

Sort

Sort button shows the presets sorted alphabetically.

Random

Random button selects and loads a random preset from the current folder. This way you can quickly browse the presets in the folder in a completely random order.

Previous

Previous button selects and loads the previous preset from the current folder.

Next

Next button selects and loads the next preset from the current folder.

Submit preset

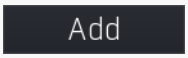
Submit preset button submits the selected preset to the online exchange servers and retrieves all the presets currently in the database. This feature serves as an online database of presets available for all the user community. Please do not submit garbage presets.

Download presets

Download presets button retrieves all the presets currently in the database. This feature serves as an online database of presets available for all the user community. Please consider participating by submitting your presets as well.

Load

Load button loads the specified preset. Please note that you can do the same thing by double-clicking the preset itself or pressing the Enter key.



Add

Add button creates a new preset using the current settings.

Add

Add button saves a new preset with a random name.

Random rename

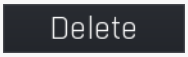
Random rename button changes the name of the preset to a random name.

Rename

Rename button lets you rename the selected preset.

Replace

Replace button replaces the selected preset by one with current settings.



Delete

Delete button deletes the selected preset.

Sort

Sort button sorts the presets by name. Please note that this change is permanent.

Search

Search filters the list of available presets to those containing the keywords in name or information.

Clear

Clear button deletes all text in the search field.

Preset information

Preset information field contains optional information about the preset, which you can edit when creating or renaming the preset.

Plugin settings

SETTINGS

Licence manager

GUI & STYLE

Style

Random style | Default style

Select current style as default

GPU acceleration: Enabled

Frames per second: 100

- Enable high DPI support
- Enable colorization
- Enable colorization for panels
- Allow default colors by plugin type
- Allow style changes if the editor is too big

Clear window settings cache

PLUGIN SETTINGS

- Intelligent sleep on silence
- Randomizer loudness compensation
- Smart bypass
- MIDI thru
- Sample-accurate event processing
- Latency reporting

Latency: 0 samples, 0.0 ms

GLOBAL SYSTEM SETTINGS

- Intelligent sleep on silence (global)
- Right click sets default value
- Tablet mode
- Enable keyboard input
- Enable scrolling without ctrl/cmd
- Collapse plugin toolbar

Set default settings | Reset default settings

ADVANCED GLOBAL SETTINGS

- Saturation antialiasing
- Forward unused keyboard input to DAW
- Silence when busy
- Store resampled files
- Use performance cores
- Show confirmations for destructive actions
- Online check for updates and tutorials
- Anonymous online platform reporting

Realtime processing threads: 14

CPU benchmark | System info

COMPATIBILITY SETTINGS

- Storage compatibility mode for V15
- Automation compatibility mode for V10

SMART INTERPOLATION

Normal	For rendering
Minimal	Minimal
Normal	Normal
High *(default)*	High *(default)*
Very high	Very high
Extreme	Extreme

Close

Plugin settings window offers more advanced settings and is available via the Settings button.

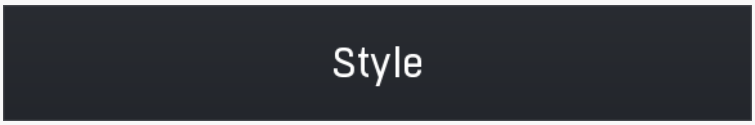
Licence panel

Licence panel lets you manage licences on this computer.

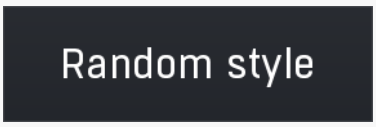
Activate
Activate button lets you activate your licence for the plugin on this computer.

GUI & Style panel

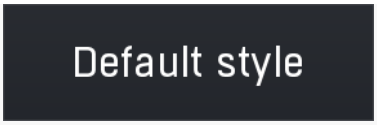
GUI & Style panel lets you configure the plugin's style (and potentially styles of other plugins) and other GUI properties.



Style
Style button lets you change the style for this particular plugin.

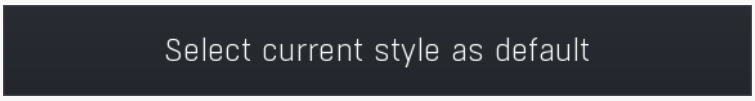


Random style
Random style button selects a random style with random editor mode.



Default style

Default style button reverts to the default style and default size of the GUI. Hold the Ctrl key while clicking to revert all MeldaProduction software products, not just the current plugin.



Select current style as default

Select current style as default button stores the current style as the default for all MeldaProduction software. This is used for the other plugins that are currently using the default style; that is, those plugins for which you have NOT selected a specific style. Please note that if you have already selected a specific style for a particular plugin, then it won't be changed until you use the **Default style button**.

To set a default style for all MeldaProduction plugins, follow these steps:

1. Open any MeldaProduction plugin, select a style you prefer. You can also customize it to suit your needs. Then, click the **Select Current Style as Default button**.
2. While holding the Ctrl/Cmd key, click the **Default Style button**. This action will delete the style settings in all MeldaProduction plugins. All plugins will adopt your new default style upon startup.

Note: Step 2 is not necessary, unless you set some other default style from each plugin. Step 2 opearation is destructive, as it removes all custom style settings from all plugins, so proceed with caution!

GPU acceleration

GPU acceleration controls how much the GPU is used for visual rendering to save CPU power.
Enabled mode provides maximum speed and lets the GPU perform as many drawing operations as possible.
Compatibility mode uses the GPU for drawing, but doesn't use modern technologies for maximum performance. Use it if you experience occasional problems with drawing, the usual case for older ATI graphics cards. With Pro Tools on OSX this mode is always used instead of Enabled mode due to compatibility problems with this host.
Disabled mode disables GPU acceleration completely, drawing is then performed by the CPU. Use only if you experience technical difficulties.
A known problem may occur when using multiple displays with multiple graphical interfaces. When moving the plugin window from one display to another, it may stop displaying correctly until you move it back to the original display.

Frames per second

Frames per second controls the refresh rate of the visual engine. The higher the number is the smoother everything is, but the more CPU it requires. You might want to lower this value if your computer is running out of CPU power.

Enable high DPI / retina support

Enable high DPI / retina support enables the plugin to use the high resolution on high DPI (Windows) and retina (OSX) devices. It is enabled by default and detected automatically, if the host allows it. If you run into any problems, you can disable it using this option. It may be desired if you use multiple displays where only some of them feature the high resolution making the image on the low resolution ones look ugly.

If you disable this option, on Windows the high DPI device detection will be ignored and the plugin will probably appear very small. You can manually compensate for it by using a bigger style. On OSX disabling this option will disable the high DPI rendering, resulting in the classic blurry look of non-compliant applications. Changes take effect after you restart the host.

Enable colorization

Enable colorization enables the plugin to change the colors of certain elements overriding your style settings. Plugins use that to highlight different parts of the graphics interface for easier workflow. You may want to disable it if you just feel it's not for you. This particular option is relevant only for controls - knobs, sliders, checkboxes etc.

Enable colorization for panels

Enable colorization for panels enables the plugin to change the colors of certain elements overriding your style settings. Plugins use that to highlight different parts of the graphics interface for easier workflow. You may want to disable it if you just feel it's not for you. This particular option is relevant only for containers - panels, graphs etc.

Allow default colors by plugin type

Allow default colors by plugin type is on by default and makes the plugin select its default colors depending on the type of the Plugin. Hence for instance equalizer will always be green. This is done by selecting one of the first 8 color presets for the current style, so the actual colors depend on selected style and its presets. You may want to disable this if you for example want all plugins to look the same including the style and colors. It is necessary to restart your host for a change to this option to take effect.

Allow style changes if the editor is

too big

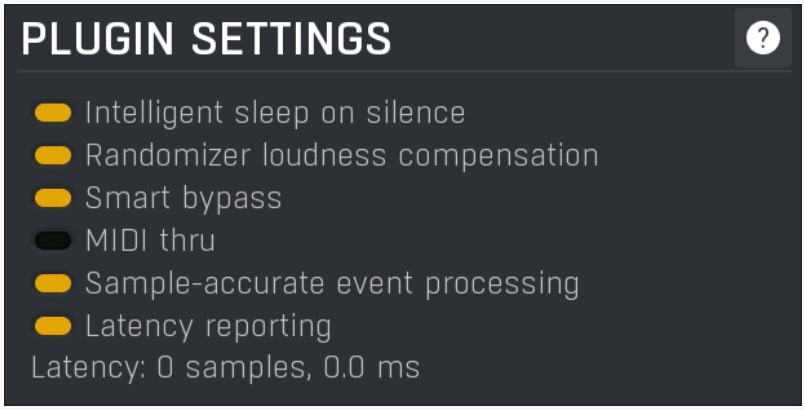
Allow style changes if the editor is too big is on by default and makes the plugin change its style, editor mode and other settings if it finds out it is too big to fit the current screen resolution.

Clear window settings cache

Clear window settings cache

Clear window settings cache button deletes stored states of all popup windows on all MeldaProduction software. The window settings mostly contain positions and sizes, but in some cases also the data inside the popup windows. You can use this feature if something goes wrong, a window doesn't appear at all, problems like that. While this shouldn't happen and it's generally better to contract our support, this button provides a potential quick fix.

Plugin settings panel



Plugin settings panel contains settings that control the behaviour of this plugin instance. These are properties that rarely need to be changed, so they have been moved here.

Intelligent sleep on silence

Intelligent sleep on silence

Intelligent sleep on silence option provides a huge CPU saver by automatically disabling the plugin processing if the input is silent and if the plugin doesn't generate some signal on its own. This makes the plugins take virtually no CPU if there is no need for them to actually process anything. Disable this if you run into any problems with them.

Randomizer loudness compensation

Randomizer loudness compensation

Randomizer loudness compensation enables the automatic detection of loudness after new settings have been generated using the main **Random** button and using the output gain of the plugin to get some predefined level. This is useful in most cases since normally randomized settings can produce various output levels, so this can mitigate the problem.

Smart bypass

Smart bypass

Smart bypass enables the high quality crossfading bypass system, which ensures a smooth transition between the processed and dry signals. You may want to disable it if you are using settings with latency on a plugin, which demands lots of CPU power, which would otherwise need to perform processing even when bypassed, which is pretty much the only downside of the smart bypassing algorithm.

MIDI thru

MIDI thru

MIDI thru makes the plugin pass all input MIDI through to its MIDI output. That is often advantageous in DAWs such as Reaper, which naturally pass MIDI from one plugin to the next.

Sample-accurate event processing

Sample-accurate event processing

Sample-accurate event processing makes the plugin schedule every event such as MIDI or automation to their accurate locations with sample accuracy, if the host allows it.

For example, if the block size in your host's audio settings is 1024 samples, this means the plugin is probably processing blocks of 1024 samples, in 44100 Hz sampling rate it is about 23ms. If this setting is disabled, any change in automation, MIDI, modulation etc. may then be granularized to 23ms (once per block), which means that you will not be able to recognize events that occur say 10ms apart from each other. When this setting is enabled however, the plugin divides processing blocks to sub-blocks and processes the events at their correct positions. This may, of course, require more CPU power.

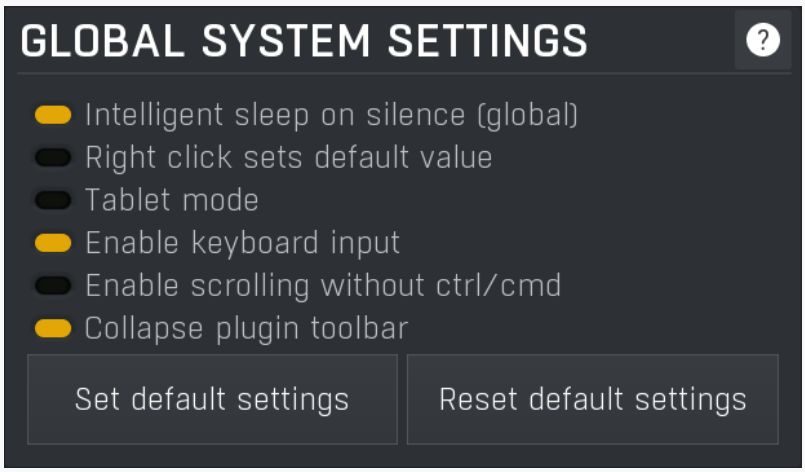
Latency reporting

Latency reporting

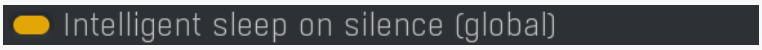
Latency reporting makes the plugin report latency to the DAW, if any. Normally this is enabled, but in certain live situations

you may want to disable this, so that the DAW stops compensating the latency on other tracks. It has no effect if the plugin is placed on master track.

Global system settings panel



Global system settings panel contains settings which are applied to all plugins on this computer.



Intelligent sleep on silence (global)

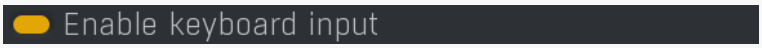
Intelligent sleep on silence (global) is a global switch, which disables the **Auto disable on silence** feature in all plugins on the system. It is provided "just in case" something goes wrong.

Right click sets default value

Right click sets default value makes the engine set default value to a parameter when you right click on it. By default, a menu is displayed instead, with an option to set the default value, but potentially with more features. When this is disabled, you can still set a default value by holding ctrl/cmd when right clicking the control.

Tablet mode

Tablet mode enables better support for tablets at the expense of the mouse. Enable this if you are using a tablet to control the plugins and it is behaving incorrectly.



Enable keyboard input

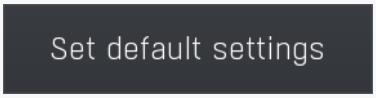
Enable keyboard input enables the keyboard input for the main plugin window. You may want to disable if the plugin intercepts spacebar key (often used by the host for playback enable/disable and your host doesn't allow for the problem itself.

Enable scrolling without ctrl/cmd

Enable scrolling without ctrl/cmd allows you to scroll anywhere in scrollable fields using mouse wheel. By default this is off, so that you have to hold ctrl/cmd while using the mouse wheel. That protects you from accidentally changing parameters.

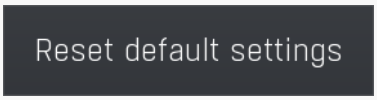
Collapse plugin toolbar

Collapse plugin toolbar makes all plugins collapse the plugin toolbar containing more advanced features such as channel modes, A-H presets, oversampling, safety limiter etc. It is enabled by default to make the user interfaces cleaner and easier to grasp for beginners.



Set default settings

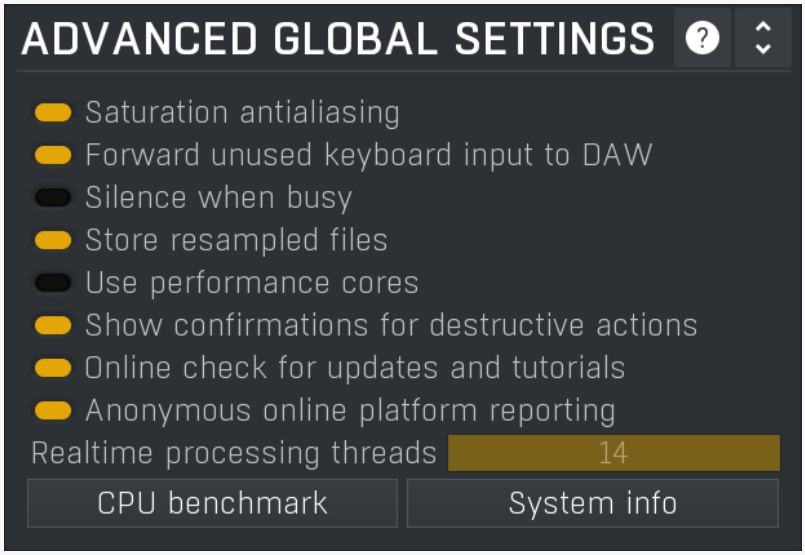
Set default settings button stores the current plugin settings as the defaults, so that when you open a new instance of the plugin, these settings will be loaded automatically.



Reset default settings

Reset default settings button removes the defaults that you set using **Set default settings** button, so that when you open a new instance of the plugin, the factory defaults will be loaded.

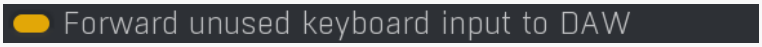
Advanced global settings panel



Advanced global settings panel contains advanced settings which are applied to all plugins on this computer.

Saturation antialiasing

Saturation antialiasing enables a global support for antialiasing in saturation algorithms available in many of the plugins. These require additional CPU processing, however significantly reduce aliasing artifacts without a need for oversampling.



Forward unused keyboard input to DAW

Forward unused keyboard input to DAW makes the plugin forward unused keyboard events to the DAW from its popups. If this is disabled, pressing say spacebar commonly used to start/stop playback won't work if a popup window is active. Enabling this makes this work and it is optional just in case your DAW does something unexpected.

Silence when busy

Silence when busy makes all plugins silence the output when something time consuming is being performed in background and the plugin needs to wait for it. For instance, in modular plugins such as MXXX, adding a module requires lots of changes in the entire engine, so it is performed in background and while the plugin is inconsistent state, it is temporarily bypassed. Sometimes however, when performing live, bypassing makes the dry signal go through and that may not be wanted. So you can enable this option, and the plugin will silence the output instead.

Store resampled files

Store resampled files allows the plugins create audio files for sampling rates being used if they differ from the original file sampling rate. It is used only by a few plugins, but it can improve the loading performance a lot at the cost of some additional storage on the hard drive. Disable this option if you are short on free space.

Use performance cores

Use performance cores instructs the plugin to use performance cores for audio processing. On systems with both efficiency and performance cores, this provides more efficient CPU utilization. Disabling this option may help when running modern operating systems on legacy hardware.
Note: Changing this option requires restarting the DAW!

Show confirmations for destructive actions

Show confirmations for destructive actions makes the plugin display a confirmation window whenever you are going to change the plugin settings irreversibly when using a feature, for example: when resetting your settings.

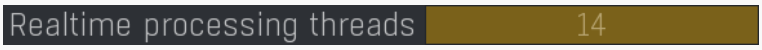
Online check for updates and tutorials

Online check for updates and tutorials lets the plugin ask about once a week if there is a new version or tutorial available so you can be easily kept up to date.

Anonymous online platform reporting

Anonymous online platform reporting helps us maximize compatibility with your operating system and host. If enabled, our plugins will send information about the system and host that you are using. We can use this information to find out which

plugins and platforms are used the most and maximize testing and support there. Platform reporting is completely anonymous and requires only minimal internet connection time (a few kB once a week).



Realtime processing threads

Realtime processing threads controls the number of threads that can perform voice processing.

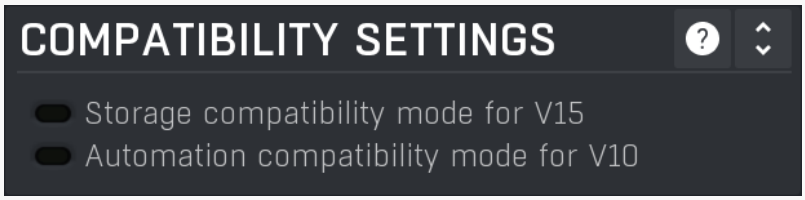
CPU benchmark

CPU benchmark button calculates the performance of the plugin with the current settings.

System info

System info button displays some technical information about the build and the machine.

Compatibility settings panel



Compatibility settings panel contains advanced settings you rarely need unless you run into some problems when using multiple versions or old projects.

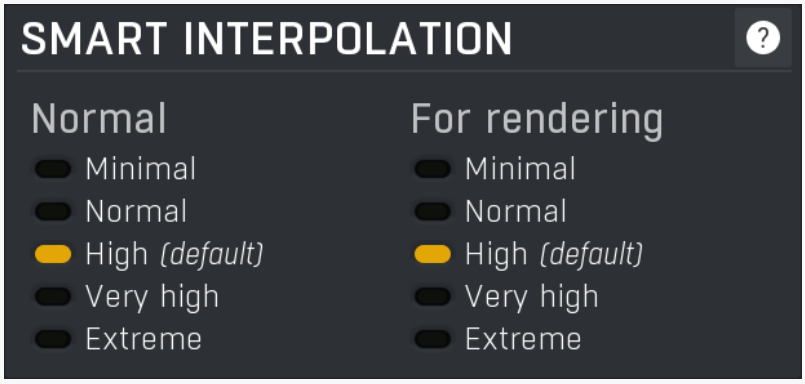
Storage compatibility mode for V15

Storage compatibility mode for V15 reverts to the older and much slower storage system used by version 15 and older. Use this if you want to open your projects or presets on older version of MeldaProduction plugins.

Automation compatibility mode for V10

Automation compatibility mode for V10 reverts the set of automation parameters back to version 10 and earlier. Use this if you need the plugins to work with projects, which contain autmation, made using version 10 or older. In version 11 the list of automatable parameters have been highly simplified and reorganized and multiparameters are provided for the vast number of hidden parameters. This should speed up loading, improve workflow with the plugins and improve compatibility with various hosts.

Smart interpolation panel



Smart interpolation panel controls the depth of the smart interpolation algorithm, which controls the parameters in order to provide maximum audio quality and lower the chance of zipper noise. Smart interpolation is engaged whenever you change any parameter via the GUI, modulators, multiparameters, MIDI or automation.

Many parameters can be automated easily and the plugin responds with sample-accurate results. However, several parameters need exhaustive pre-processing when changed. In these cases, the parameters are not updated every sample, but, for example, once every 32 samples. This highly reduces CPU usage, but affects the output quality.

With modulators the situation is more complicated. Besides the updating issue, the modulator itself can perform some pretty advanced processing, hence it is better to perform the processing in blocks. However, the bigger the block, the less often the modulator updates those parameters associated with it and the resulting modulation is less accurate. In a way you can say that the modulator is slower and lazier. This may actually be wanted, so when it comes to modulators it is not true that a better mode always means better output quality.

The smart interpolation mode controls the maximum number of samples being processed before the parameters are updated. **Minimal mode** uses 2048 samples and rarely will do anything unless processing offline. **Normal mode** uses 256 samples and usually is enough to achieve good quality results. **High mode** uses 32 samples and provides perfect quality for most cases. It is also a good compromise between CPU usage and audio quality, so it is the default. **Very high mode** uses 4 samples and you will rarely need it. **Extreme mode** uses 1 sample, which means that everything is updated after every single sample. This provides the highest possible accuracy and quality you can ever achieve, however it requires lots of CPU and it is very unlikely that you will ever need it. If you use this mode and still hear audio artifacts, then either what you are hearing is actually CPU overload, or you are doing something that is not physically possible.

The higher the mode, the quicker the parameter updates, but the more the CPU load.

Please note that modulating certain parameters without artifacts is impossible. For example, when modulating a delay very quickly, the physics of such a process just cannot occur in the natural world and the results are appropriately unnatural. These physically impossible processes usually manifest themselves as distortion or zipper noise.

Modulator editor

Modulator is an extremely advanced feature, which lets you change parameters automatically depending on various inputs. You can use this to add movement to your sound, respond to some plugins differently for louder sections, or even follow the pitch of the input.

The modulator edit window has two parts: on the left side you can configure the mode of the modulator (the way the modulator works) and on the right side there is a list of parameters to modulate. A modulator can control all automatable parameters (and often more than that) including the parameters of other modulators. Each modulator can control as many parameters as is needed and each of the parameters has its own range and transformation shape. The values and ranges of the first 4 parameters associated with the other modulators can also be modulated/automated. The following modulator modes are available:

Normal mode makes the modulator behave like an ordinary low-frequency oscillator (LFO). There are various ways to control its shape as with all oscillators in our plugins. Each modulator can synchronize to the host in the **Synchronization** panel. Modulators can also synchronize with each other using the **Sync groups**. Using **MIDI reset** you can reset the oscillator to any phase using MIDI notes, but obviously to-host synchronization must be disabled in order for this to work.

Note that the settings in this mode are used even if the modulator is actually in a different mode by using "LFO modulation". This basically blends between the actual mode, which may for example detect the input signal level, and give it some additional movement using the LFO depending on the LFO modulation parameter available for each of the remaining modes.

Follower mode makes the modulator detect the input signal level. It contains an extremely advanced and accurate level detector taken from our MDynamics plugin. The level follower is an immensely useful feature, yet it may be a little difficult for beginners to comprehend, so we will cover it here in more detail.

It is often necessary to adjust the follower slightly for new material. First, it has the standard parameters - attack, release, hold and RMS length. These are fairly standard features and help is available for each of them. **Level min and max** controls the range of input levels. When the input level is equal to or below the min level, the modulated parameters' values will be minimal. Similarly, when it reaches the max level, the modulated parameters' values will be at their maximum. This allows for adjustments to the range of input levels, which are certainly different for any audio material and settings. It can be used creatively too - for example, by using very low values for both limits we can differentiate between silent and non-silent parts, similar to the way a gate effect works.

Advanced detector settings provide some extraordinary features, such as psycho-acoustic pre-filtering, which forces the modulator to detect loudness instead of raw input levels, custom input signal pre-filtering using a fully featured 6-band equalizer, and custom attack and release shapes. **Band-pass panel** pre-filters the level detection signal using a band-pass filter, so this is like a very simplified version of the equalizer from the advanced detector settings. **Side-chain** makes the modulator measure side-chain input if the plugin has one. For modular plugins the modulator can also be driven by a feedback signal. The **advanced panel** provides some further level processing features that you can take advantage of creatively or to further adjust to your actual audio material.

Project onto LFO shape is a more advanced concept, which is available for other modulator modes too. You can easily imagine, that the modulator in any mode generates values for each parameter, we can say it is between 0 and 1, where 0 sets minimum parameter value, and 1 sets the maximum. Project onto LFO shape forces the modulator to use this range in the oscillator shape, which can then be configured in normal mode. The value is basically transformed by the oscillator shape, where the values generated by the modulator are on the horizontal axis (phase) and the output is the actual oscillator value. This feature has no physical meaning and can only be used creatively - to transform the more or less linear results of the level follower into a much more complicated curve.

Let us demonstrate the follower mode with an example - the idea is to apply a delay to a snare drum within a previously mixed drumset. This is commonly used on reggae/dub rhythms for example, however in these cases the snare track is usually available separately. Using the modulators you can get somewhat interesting results even with an already mixed drumset. The idea is to increase the input gain whenever the snare is playing, so that only the snare drum (and potentially other instruments playing at the same moment) are passed into the delay. So first teach the modulator to control input gain parameter of the delay and set it to follower mode, potentially configure some of the parameters to get the desired response. Now the louder the input is, the more delay you get. To make it respond only to snare drum, enable the band-pass and set the filter limits accordingly, e.g. 500Hz to 1k. This makes the input gain increased depending on the input level in this part of the spectrum, which contains the snare drum.

Envelope mode causes the modulator to generate an arbitrary envelope, similar to those from synthesizers. It can either follow MIDI - the envelope starts when a key is pressed, goes though the attack and decay stages, then holds in sustain stage until the key is released when the release stage begins, or it can follow audio - when the audio level exceeds **Threshold on** it behaves the same way as when a note is pressed in MIDI mode, and then when the input level drops below **Threshold off** it behaves like a key release. As with most modes there is LFO modulation and LFO projection and the input level can be driven by the side-chain or feedback if available. The envelope shape can be adjusted using several controls (lengths of each stage etc.) and you can even draw your own shape.

Random mode is a smooth random generator. It is very handy if you want some parameters to change over time, but do not actually want them to be periodic like LFOs. A modulator in random mode does not actually generate random values, the results will always be the same at each position in your arrangement in the host. This allows a pseudo synchronization with the host and ensures a "what you hear is what you get" performance. **Speed** parameter controls the speed of change and any slight change to this parameter will change the whole stream.

Pitch detects the pitch of the input signal assuming it is not polyphonic (here it can work too and will probably detect the lowest note, however it is definitely not suitable for percussive signals, which do not have a pitch). It is very useful, enabling you to tune an oscillator to follow your singing, or allow an equalizer to control separate harmonics of a vocal, use a distortion to get more drive for higher notes in a guitar solo and much more. The pitch detection may be a little tricky to understand, so we will discuss it in more detail.

A pitch detector takes the input signal and tries to approximate the pitch of the fundamental frequency in it. It is physically impossible to detect pitch instantly, as an extreme example, 20Hz takes 50ms for the signal to evolve enough to detect that there is actually a 20Hz frequency in the signal. For this and many other reasons any pitch detector employs several limitations. These are available in the **Detector panel**. The defaults will work well for most audio material, however, it is useful to understand the parameters, so that you can let the detector adapt better to your particular audio materials if necessary, and also in order to be more creative.

Min and max frequency parameters in the Detector panel control the limits of the frequencies you expect in the input. For example, a female voice is unlikely to sing below 100Hz, so it is customary to set the minimum frequency to 100Hz or even higher. Voice signals contain several artifacts, blows and pops, all of which can temporarily create frequencies below the actual pitch of the voice, so setting these limits is preferable to avoid "jumps" to incorrect pitches. **Stabilization** and **Speed** also prevent these jumps by restricting how quickly the pitch can change. These can also be used creatively. **Threshold** controls the minimum level of the input signal to be considered "not-silent and probably having pitch". This acts as a form of gate, which prevents the detector from analyzing irrelevant rumble in between actual performances. **Shift panel** allows the detected pitch to be shifted up or down and **Auto-tune panel** moves it to the closest note - similar to the automatic pitch changing function from MAutoPitch, except no pitch shifting is actually done and the results are used purely to control some parameters.

Min and max frequency parameters in the top of the editor have a very different meaning than the parameters of the same name in the detector panel. From now on we will assume that the pitch has been detected successfully and are now considering what to do with the results. Again, we may assume the modulator generates values from 0 to 1, where at 0 the modulated parameters' values become minimal and reach maximum at 1. When the input pitch is equal or below the min frequency parameter, the modulator's value is 0, hence modulated parameters will have a minimal value as well. Similarly when the pitch reaches max frequency, the modulated parameters will get to the maximum.

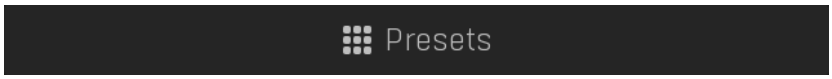
Now you may say this makes no sense, because the detected pitch cannot exceed the limits specified in the **Detector panel** anyway. The reason for this is that most "frequency" parameters of all plugins are limited from 20Hz to 20kHz, whether it is the

frequency of a band in an equalizer, or a high-pass frequency in a phaser for example. It is a reasonable solution since physiologically speaking these figures are on or around the range of our hearing limits.

Let us explain the concept with an example. We want to modulate a band of an equalizer, so that it always follows the fundamental frequency, the pitch, of our audio material. All we need to do is to switch the modulator to pitch mode, allow it to control the band frequency parameter and set the range for this parameter to the full range, from 20Hz to 20kHz. The pitch detector may then detect frequencies from 50Hz to 2kHz, but the modulator takes it that the actual limits (converted to 0..1) are 20Hz to 20kHz and that exactly the same range is configured for the band frequency parameter, so you could say that "they understand each other". We did not need to touch the min and max frequency parameters at all.

Here is one more example, where we would actually want to adjust the min and max frequency parameters. We want to control a drive parameter of a distortion for a guitar so that the higher the guitarist plays the more distortion he gets. Again, we teach a modulator to control the drive parameter, for any range we want, and switch the modulator to pitch mode. Now the modulator will move the drive parameter, but only slightly, because it assumes the pitch can vary from 20Hz to 20kHz, but the guitar may actually only play from about 100Hz to 1kHz. So we can use the min and max frequency parameters to say "what is high and what is low", to limit the frequency range. There are no general rules here, you have to experiment, because every instrument and parameter is different.

To sum things up, the difference between controlling a frequency parameter and a drive parameter is simply the fact that a frequency parameter is compatible with the pitch. After all, pitch is nothing more than a frequency (strictly speaking it is a logarithmic representation of frequency).



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

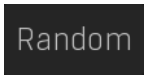
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.



Random

Random button generates random settings. Note that unlike copy & paste, presets & randomization do NOT affect the set of parameters being modified, hence it serves to optimize adjustment of the modulator behaviour assuming that you already specified the set of parameters to control.
If you hold **Shift**, the plugin will undo previous randomization.

R

R button enables automation read. This way you can actually automate the modulation value. First you use **W button** to record the modulator values over time. After that you can modify it in some way and enable automation read to override the normal modulator behaviour. Note that the results may be different when automation is used with potentially lower audio quality and slower response.

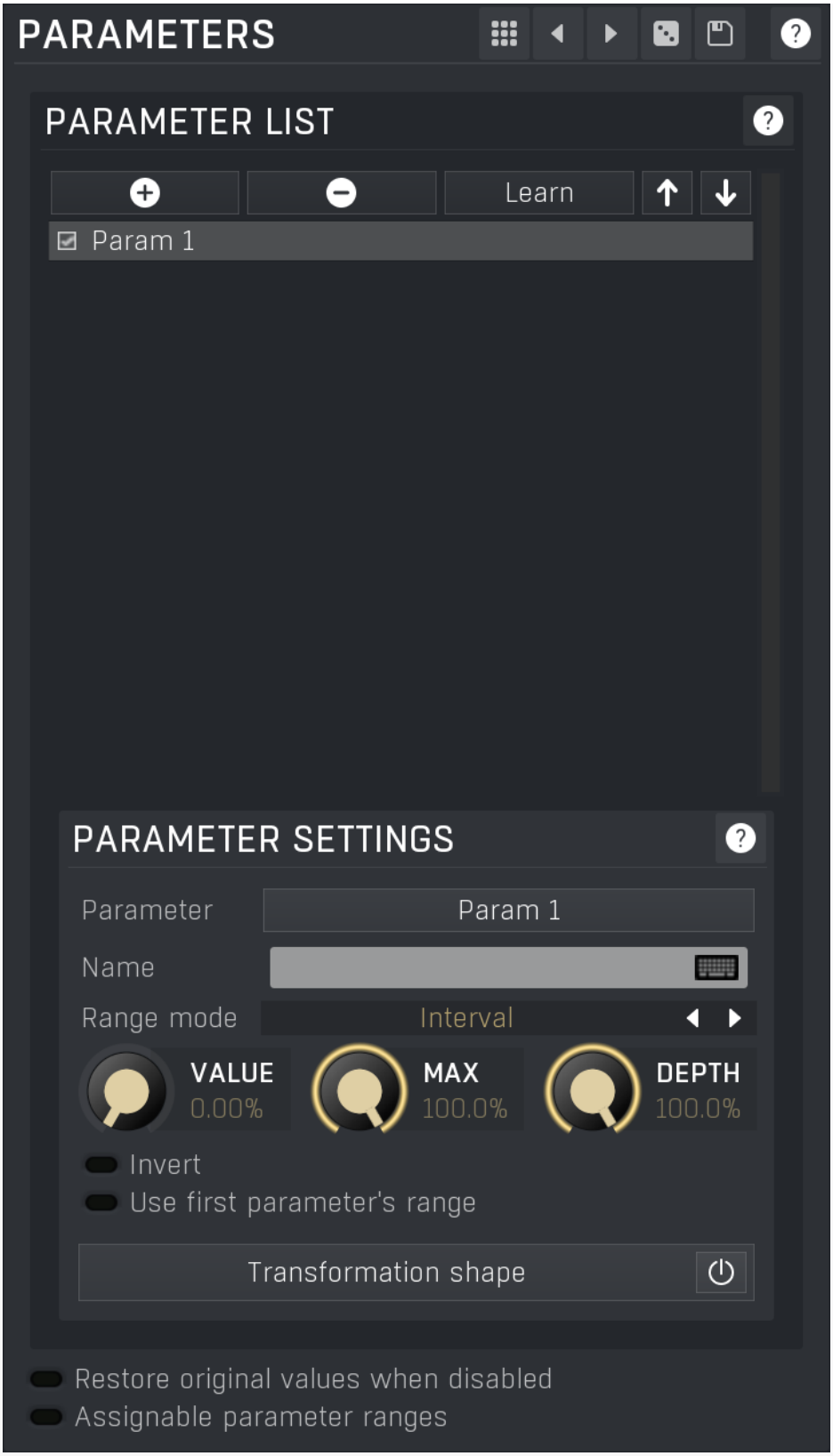
W

W button enables automation write. This way you can actually automate the modulation value. Use the button to record the modulator values over time. After that you can modify it in some way and enable automation read to override the normal modulator behaviour. Note that the results may be different when automation is used with potentially lower audio quality and slower response.

Map

Map button displays all current mappings of modulators, multiparameters and MIDI (whichever subsystems the plugin provides).

Parameters panel



Parameters panel contains the list of the parameters that the modulator is controlling, their ranges etc.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

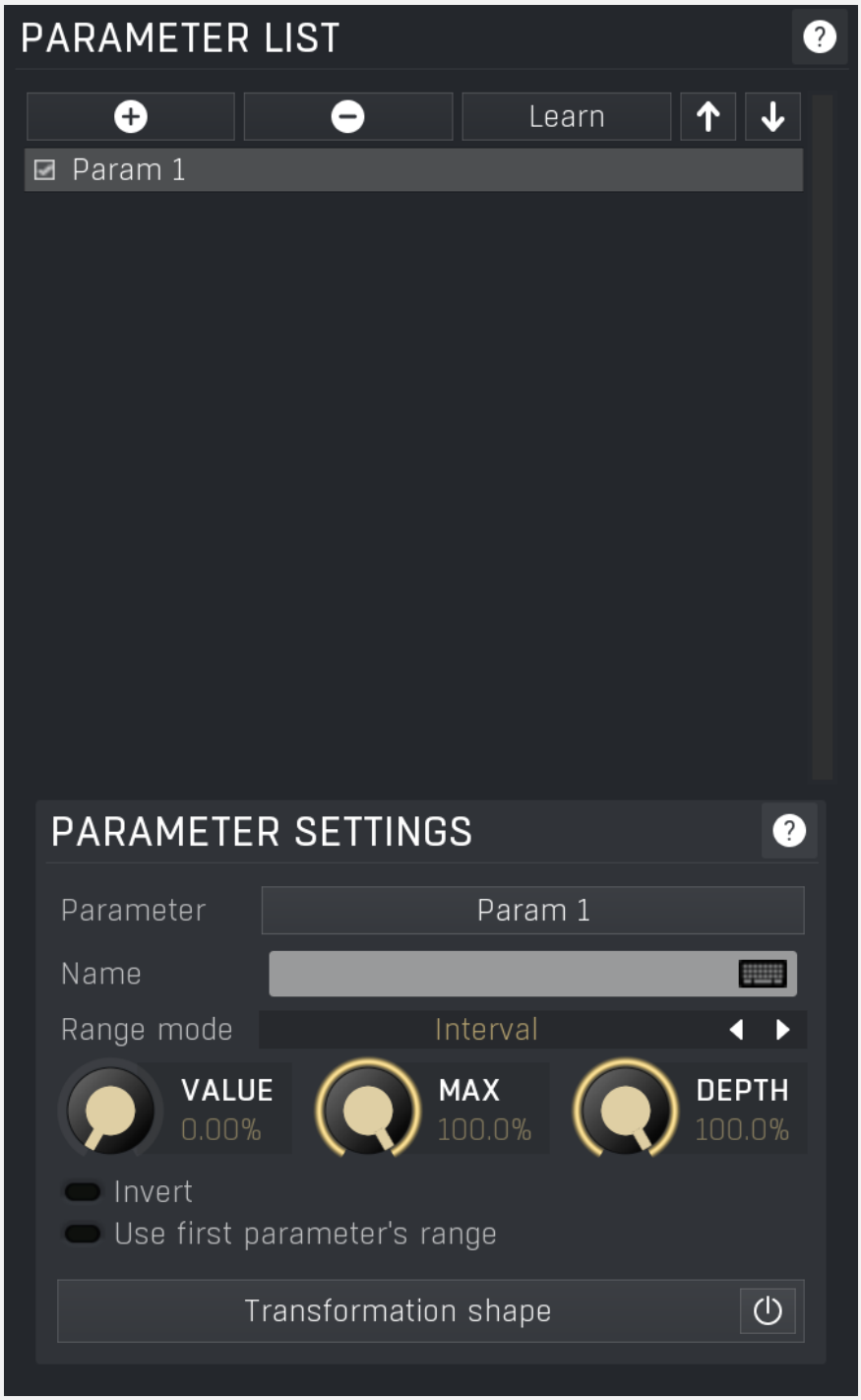
Right arrow

Right arrow button loads the next preset.

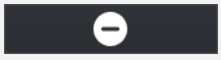
Randomize
Randomize button loads a random preset.

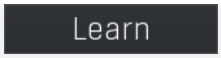
Save
Save button replaces the current preset.

Parameter list



Add
Add button adds a parameter to the list of controlled parameters. Alternatively you can use the learn feature available by right-clicking the modulator button.

 **Delete**
Delete button deletes the selected parameter from the list of controlled parameters.



Learn

Learn button starts or stops the learning. Click it, then move some parameters in the plugin, then click it again. Learning can also be accessed from the global modulator menu.

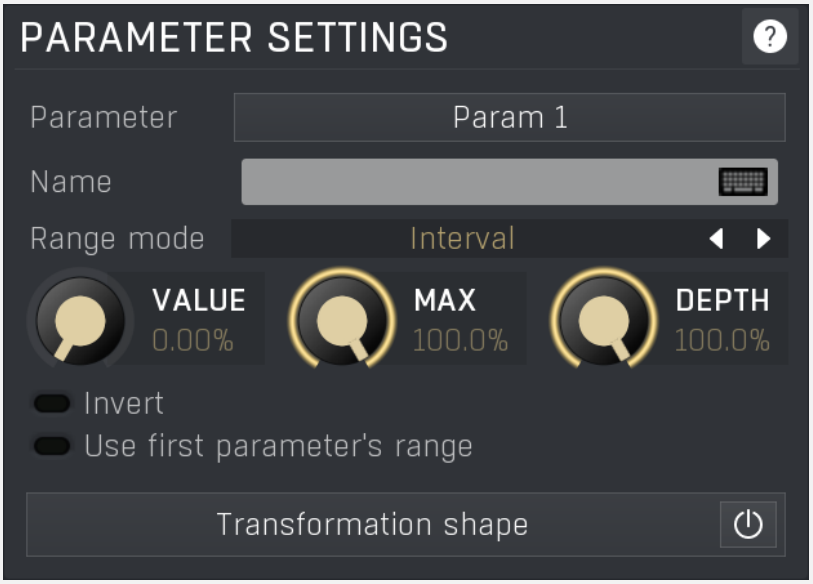
Up

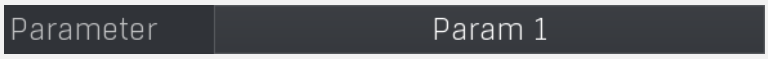
Up button moves the selected parameter up one item, if possible. This may be useful when keeping things organized, but please note that if you have some other multiparameter, modulator or another subsystem access the ranges of individual parameters, this function will reorder them, so these connections will no longer be correct.

Down

Down button moves the selected parameter down one item, if possible. This may be useful when keeping things organized, but please note that if you have some other multiparameter, modulator or another subsystem access the ranges of individual parameters, this function will reorder them, so these connections will no longer be correct.

Parameter Settings





Parameter

Parameter defines the target parameter which is being modulated. The set contains all automatable parameters.

Name

Name lets you name the parameter somehow and may be helpful in situations, where there are many parameters being edited without obvious meanings.

Range mode

Range mode defines how the parameter range is selected. While sometimes it is better to specify minimum and maximum, other times it is better to use a nominal center and depth (% of full scale). This control allows you to define which one it will be.

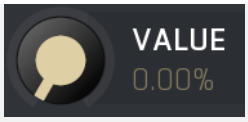
Up and down mode makes the values go above and below the selected **Value**, which is considered the center. The interval is made smaller if necessary.

Full range mode is similar, except the range is symmetrically constrained, so the selected **Value** may not be the center anymore.

Up/down only modes goes from the selected value up/down only.

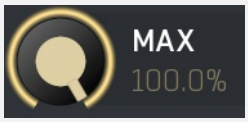
Let's compare these 4 modes. Taking a value of -12dB value, with a depth of 75% and a scale of +/- 24dB. The nominal range is therefore = +/-24 dB * 75% = 36dB. With values of 0%, 50% and 100% the outputs are:
Up and down: -24, -12, 0 (range constrained to 12 dB either side)
Full range: -24, -6, 12 (range limited to minimum, but not constrained)
Up only: -12, 6, 24 (range not constrained = +/-24 dB * 75% = 36dB)
Down only: -12, -18, -24 (range limited to minimum)

Interval mode is the most simple one and goes from **Value** to **Maximal value**.



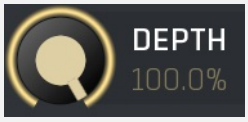
Value

Value defines the center of the target parameter's range or the minimum if the **Range mode** is set to **Interval**.



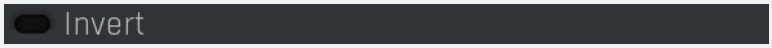
Maximal value

Maximal value defines the upper limit of the target parameter's range. It is available only if the **Range mode** is set to **Interval**. This value can be lower than **Value**. 0% is always mapped to reference>Value and 100% to reference>Maximal value.



Depth

Depth defines size of the target parameter's range. It is used only if the **Range mode** is not set to **Interval**.



Invert

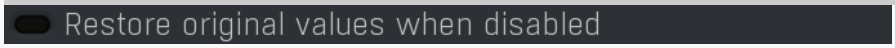
Invert checkbox inverts the target parameter's range, so that minimum becomes maximum and vice versa.

Use first parameter's range

Use first parameter's range makes the parameter display use the same range as the first parameter in the list. This is often useful if want to control the range in some way and apply the range to multiple parameters.

Transformation shape

Transformation shape button displays the graph editor, which lets you tweak the shape of the curve used to control the selected parameter. The X axis shows the original values, the Y axis defines the results. Note that this takes some CPU, therefore you have to enable it using the enable button in the title.



Restore original values when disabled

Restore original values when disabled makes the modulator restore the original parameter values when it is disabled by automation or modulation. Normally when you manually disable the modulator, the original values are restored as that is usually desired. However when you control the modulator enable state by automation or modulation, you may or may not want this to happen.

Assignable parameter ranges

Assignable parameter ranges allows you to assign parameter ranges of several first parameters to other subsystems such as multiparameters or modulators. By default it is disabled, which removes all the relevant parameters to save valuable resources. This feature is available only if automation compatibility mode for V10 is disabled.

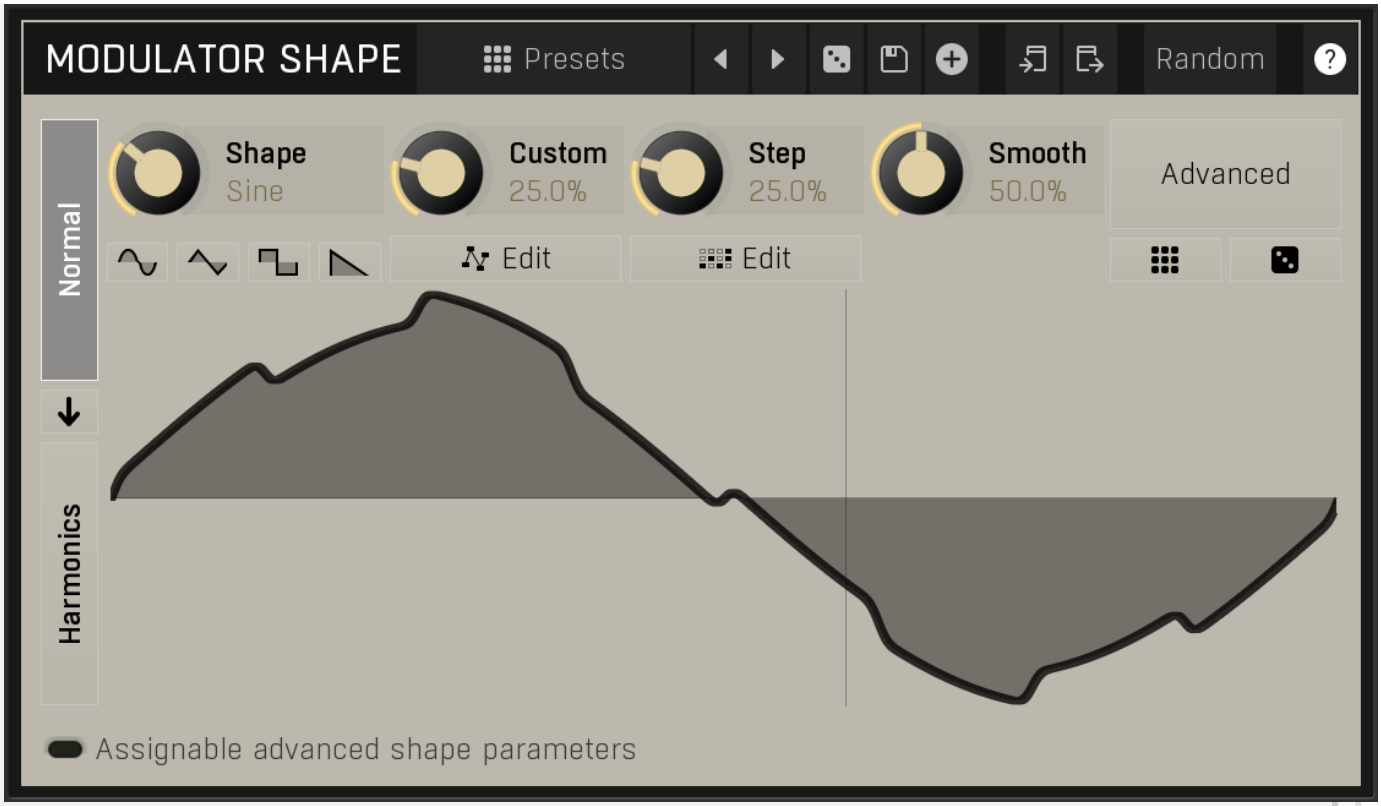
Mode

Mode defines the way in which the modulator works. The modulator is like a black box that generates one number in range 0% to 100% at each moment and then assigns the appropriate value to each of the target parameters. The mode defines what this number will be. Select the particular tab to control the modulator's behaviour.
Normal mode uses a standard low-frequency oscillator (LFO) to drive the parameters.
Follower mode uses the level of the input signal.
Envelope generates an envelope using MIDI notes or by following input signal level.
Random generates randomized output which is however the same every time you render the song.
Pitch detects and follows the pitch of the input signal.

Normal mode

Normal mode makes the modulator work as a traditional low-frequency oscillator (LFO). Note that even if the modulator itself is running in a different mode, you can still blend this LFO using the **LFO modulation** parameter available on each tabbed page. The LFO parameters themselves are available on the first tabbed page only though.

Signal generator



Signal generator defines the modulation LFO shape. It is used by the LFO generator, but also for the **Project** feature.Signal-generator is an incredibly versatile generator of low & high frequency signals. It offers 2 distinct modes - Normal and Harmonics.
Normal mode is appropriate for low-frequency oscillators, where the graphical shape is relevant and is used to drive some form of modulation. For example, a tremolo uses this modulation to change the actual signal level in time. Frequencies for such oscillators usually do not exceed 20Hz as this is a sort of limit above which the frequencies become audible.
Harmonics mode is designed for high-frequency oscillators, where the actual shape is not as important as the harmonic content of the resulting signal, hence it is especially useful for actual audio signals. Please note that since a shape can contain more harmonics than those available from the harmonic generator, the results may not be exactly the same. As an example, a rectangular wave in normal mode may sound fuller than when converted to the harmonic mode.

Use the arrow-down button to switch from normal mode to harmonics mode or click the **Normal** and **Harmonics** buttons

Normal mode

The generator first uses a set of predefined signal shapes (sine, triangle, rectangle...), which you can select directly by right-clicking on the editor and choosing the requested shape from the menu. This menu also provides a link to the modulator shapes preset manager, normalization and randomization. You can also use the **Main shape** parameter, which generates a combination of adjacent signals to provide a nearly inexhaustible number of basic shapes.

The engine then combines the predefined shape with a **Custom shape**, which may be anything you can draw using the advanced envelope engine, depending on the level set by the **Custom shape** control. Use the **Edit** button to edit the custom shape.

You can also combine those results with a fully featured step sequencer, with variable number of steps and several shapes for each of them, depending on the level set by the **Step sequencer** control. Use the lower **Edit** button to edit the step sequence.

Those results may be mixed with a custom sample, which is available from the advanced settings, accessed by clicking the **Advanced** button.

Smoothness softens any abrupt edges, generated by the step sequencer for example.

Finally there are **Advanced** features providing more complex transformations, adding harmonics etc. or you can click the **Randomize** button in the top-left corner to generate a random, but reasonable, modulator shape.

Harmonics mode

Harmonics mode represents the signal as a series of harmonics (that is, multiples of the base frequency). For example,

when your oscillator has a frequency of 2Hz (set in the **Rate** panel), then the harmonics are 2Hz, 4Hz, 6Hz, 8Hz etc. In theory, any signal can be created by mixing a potentially infinite number of these harmonics.

The harmonics mode lets you control the levels and phases of each harmonic. The top graph controls the levels of individual harmonics, while the bottom one controls their phases. Use the left-mouse button to change the values in each graph, the right-mouse button sets the default for the harmonics - 0% level and 0% phase. In both graphs the harmonics of power 2 (that is octaves) are highlighted. Other harmonics may actually sound disharmonic, despite their names.

For example, if you reset all harmonics to the defaults and increase only the first one, you will get a simple sine wave. By adding further harmonics you make the output signal more complex.

Harmonics controls the number of generated harmonics. The higher the number is, the richer the output signal is (unless the levels are 0% of course). This is useful to make the sound cleaner. For example, if you transform a saw-tooth wave to harmonics, it would not sound like a typical saw-tooth wave anymore, but more like a low-passed version of one. The more harmonics you use, the closer you get to the original saw-tooth wave.

Generator is a powerful tool for generating the harmonics, which are otherwise rather clumsy to edit. The generator provides several parameters based upon which it creates the entire series of harmonic levels and phases. These parameters are usually easier to understand than the harmonics themselves. Part of the generator is the randomizer available via the **Random seed** button, which smartly generates random settings for the generator. This makes the process of getting new sounds as simple as possible.

Signal generation fundamentals

The signal generator produces a periodic signal with specified wave shape. This means that the signal is repeating over and over again. As a result it can only contain multiples of the fundamental frequency. For example, if the generator is producing 100Hz signal, then it can contain 100Hz (fundamental or 1st harmonic), 200Hz (2nd harmonic), 300Hz (3rd harmonic), 400Hz (4th harmonic) etc. However, it can never produce 110Hz. You can then control the level of each harmonic and their relative phases. It does not matter whether you use the normal mode using oscillator shapes, or harmonics mode where you can control the harmonics directly. If both modes result in the same wave shape (such as sine wave vs. 1st harmonic only), then the result is exactly the same.

Sine wave is the simplest of all as it contains the fundamental frequency only. The "sharper" the signal shape is, the more harmonics it contains. The biggest source of higher harmonics is a "discontinuity", which you can see in both rectangle and saw waves. In theory, these signals have an infinite number of harmonics. However since our hearing is highly limited to less than 20kHz, the number of harmonics which are relevant is actually pretty small. If you generate a 50Hz signal, which is very low, and assuming that you have extremely good ears and you actually hear 20kHz, then the number of harmonics audible for you is 20000 / 50 = 400.

What happens above 20kHz?

Consider the example above again, what happens with harmonics above 400? These either stay there and simply are not audible, disappear if anti-aliasing is used, or get aliased back under 20kHz in which case you get the typical digital dirt.

When you convert a rectangle wave to harmonics mode, only the first 256 harmonics are used, so it basically works like an infinitely steep low-pass filter. What is the limit then? 50 Hz * 256 = 12.8kHz. The harmonic mode will not produce anything above this limit if you are generating a 50Hz signal. Most people do not hear anything above 15kHz, so this is usually enough, but if not, you may need to use the normal mode where you get the "infinite" number of harmonics.

What you see is not always what you get!

Say you want a rectangle wave and play a 440Hz tone(A4). You would expect the output signal to be a really quick rectangle wave, right? Wrong! If you would do that, and actually most synthesizers on the market do that, you would get the infinite number of harmonics. And, since you are working in say 48kHz sampling rate, the maximum frequency that can actually exist in your signal is 24kHz. So everything above it would get aliased below 24kHz, and there would be a lot of aliased dirt.

The "good" synthesizers perform a so-called anti-aliasing. There are several methods, most of them require quite a lot of CPU or have other limitations. The goal is to remove all frequencies above the 24kHz in our case or in reality, it is more about removing all aliased frequencies above 20kHz - this means, that we do not care about frequencies above 20kHz, because we do not hear them anyway. But we will keep it simple. Let's say we remove everything above 20kHz. You already know that the rectangle wave can be created using an infinite number of harmonics or sine waves. We removed everything above the 45th harmonic (20000 / 440) so our rectangle wave is trying to be formed using just 45 harmonics, so it will not really look like a rectangle wave.

After some additional filtering (like DC removal), the rectangle wave may look completely different than a true rectangle wave, yet it would sound the same! Does it matter? Not really. You simply edit the shape as a rectangle wave and let the synthesizer do the ugly stuff for you. But do not check the output, because it may be very different than what you would expect ;).

How can I generate non-harmonic frequencies?

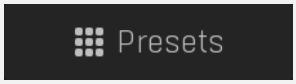
Ok, so now you are playing a 440Hz (A4) saw wave, it contains 440Hz, 880Hz, 1320Hz etc. Anything generated using the signal generator can contain only these frequencies, the only difference is the levels and phases of each of them. What if you want to make the signal dirty by adding say 500Hz? Well, that is not that simple! Here we are getting into audio synthesizer stuff, so let us just give you a few hints.

The traditional way is to use modulation. One particular method is called frequency modulation (FM). Instead of generating a 440Hz saw wave with your generator, you change the pitch, up and down. You are modulating the frequency, that's why FM. It is basically a vibrato, but as you increase the speed of the vibrato, it gets so quick that you stop noticing the pitch changes (that's very simplified but it serves the purpose) and instead it starts producing a very complex spectrum. Will the 500Hz be there? Well, if setup correctly, yes, but there will also be lots of other non-harmonic frequencies.

Another way is possible without any other tools. Let's say you do not want 440Hz, but 660Hz. Then you may generate 220Hz instead of 440Hz (which is one octave below it) and voila, 660Hz is the 3rd harmonic (3 x 220 is 660)! But you need to shift the saw wave one octave above. Fortunately it is not that hard here - go to the normal mode, select saw tooth, click advanced, and use the harmonics panel to remove the fundamental and leave just the 2nd harmonic, then convert it to harmonic mode. Well, it's not that hard, but it's not exactly simple either...

The only way is, of course, additive synthesis. In that case you do not use one oscillator, but many of them. It lets you generate just about anything. But there is a catch, actually many of them. First, you need to say "ok I want this frequency and that frequency...", the setup is actually infinitely hard as there may be an infinite number of frequencies :). And the second is, of course, CPU requirements.

So is there some ultimate solution? Nope, sorry. The good thing is, you will not probably need it, because while what you see is not always what you get, also what you want is often not what you really want to hear :).



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

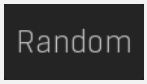
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.



Random

Random button generates random settings using the existing presets.

Normal

Normal button switches the generator into the normal mode, which lets you edit the shape of the oscillator. This is especially advantageous for low-frequency oscillators, where the shape matters even though it doesn't have any physical meaning.

Convert

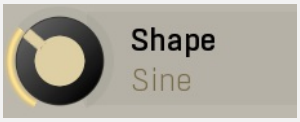
Convert button converts the current shape into harmonic-based representation. Please note that since the number of harmonics is limited, the result will not perfectly resemble the original shape.

Harmonics

Harmonics button switches the generator into the harmonics mode, which lets you edit the levels and phases of individual harmonics. This is especially advantageous for high-frequency oscillators, hence sound generators.

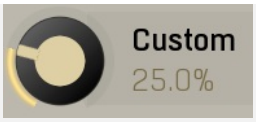
Signal generator in Normal mode

Signal generator in Normal mode works by generating the oscillator shape using a combination of several curves - a predefined set of standard curves, custom shape, step sequencer and custom sample. It also post-processes the shape using several filters including smoothing to custom transformations. This is especially useful when using the oscillator as an LFO (low-frequency-oscillator), where the harmonic contents does not really matter, but the shape does.



Shape

Shape controls the main shape used by the signal generator. There are several predefined shapes: exponential, triangle, sine power 8, sine power 4, sine square, sine, harmonics, more harmonics, disharmonics, sine square root, sine 4 root, rectangle, rect-saw, saw, noise and mess. You can choose any of them or interpolate between any 2 adjacent shapes using this control.

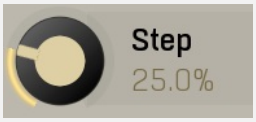


Custom

Custom controls the amount of the custom shape that is blended into the main shape.

Edit

Edit button shows the custom shape editor.

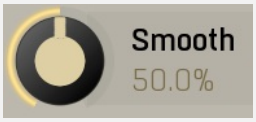


Step

Step controls the amount of the step sequencer shape that is blended into the main shape (which has already been blended with the custom shape).

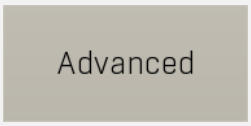
Edit

Edit button shows the step sequencer editor.



Smooth

Smooth controls the amount of smoothing. Many shapes, especially those produced by the step sequencer, have rough jagged edges, which may be advantageous, but when used to modulate certain parameters, the output may be clicking or causing other artifacts. Smoothness helps it by smoothing the whole signal shape out and removing these rough edges.



Advanced

Advanced button displays an additional window with more advanced settings for post-processing the signal shape, such as harmonics or custom transformations.

Assignable advanced shape parameters

Assignable advanced shape parameters allows you to assign advanced parameters such as step sequencer values to other subsystems such as multiparameters or modulators. By default it is disabled, which removes all the relevant parameters to save valuable resources.

Signal generator in Harmonics mode

Signal generator in Harmonics mode works by generating the oscillator shape using individual harmonics. Essentially a harmonic is a sine wave. The first harmonic, known as the fundamental, fits once in the oscillator time period, hence it is the same as selecting sine wave in the **Normal mode**. The second harmonic fits twice, the third three times etc. In theory, any shape you create in normal mode can be converted into harmonics. However, this approach to signal generation needs an

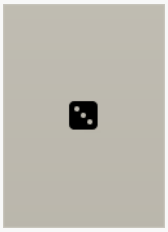
enormous number of harmonics, which is both inefficient to calculate and mostly hard to edit. Therefore, the harmonic mode can process up to 256 harmonics, which is enough for very complex spectrums, however it is still not enough to generate an accurate square wave for example. If your goal is to create basic shapes, it is better to use the normal mode.

It is nearly impossible to say how a particular curve will sound when used as a high-frequency oscillator in a synthesizer, just by looking at its shape. Harmonics mode, on the other hand, is directly related to human hearing and makes this process very simple. In general, the more harmonics you add, the richer the sound will be. The higher the harmonic, the higher the tone. Usually, one leaves the first harmonic enabled too, as this is the fundamental tone, however you may experiment with more dissonant sounds without it.

Editing harmonics can be time consuming unless you hear what you want, so a signal generator is also available. This great tool lets you generate a random spectrum by a single click. You can also open the **Generator** settings and edit its parameters, which basically control the audio properties in a more natural way - using parameters such as complexity, harmonicity etc.

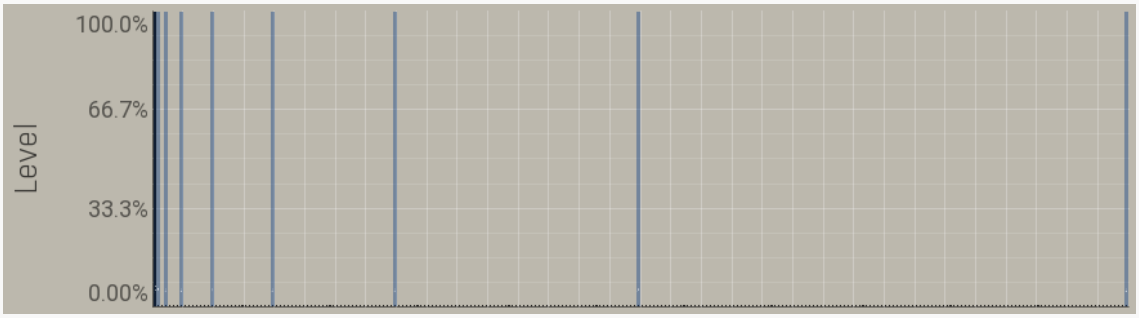
Generator

Generator button shows a powerful harmonics generator, which can create unlimited number of various timbres and even analyze a sample and extract harmonics from it.



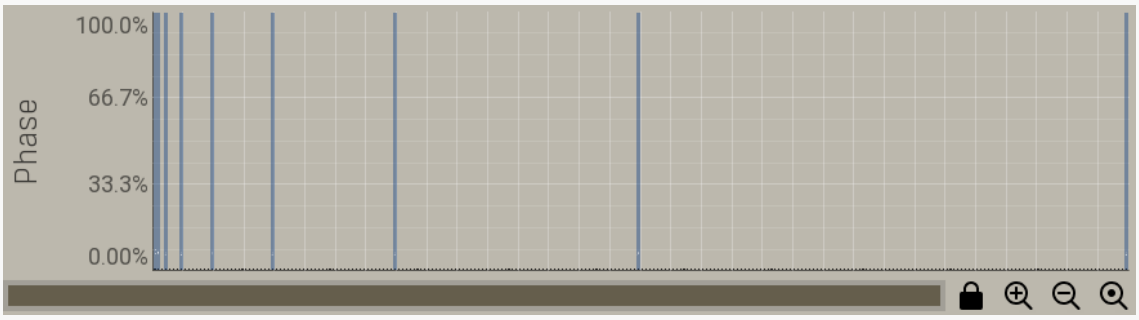
Randomize

Randomize button selects random parameters for the harmonics generator, so you can use it to get a random sound character instantly. Hold **Ctrl** to slightly modify existing generator settings instead of completely changing them.



Magnitudes graph

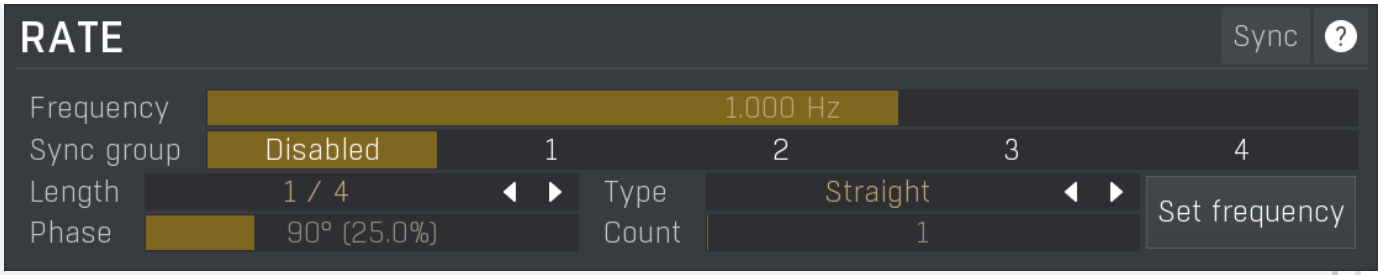
Magnitudes graph contains the levels of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.



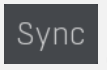
Phases graph

Phases graph contains the phases of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.

Rate panel



Rate panel contains parameters controlling the speed of the LFO, whether the modulator is set to **Normal** mode or any other mode while the **LFO modulation** is used.



Sync

Sync switch turns the modulator into synced mode, where its speed is not defined by frequency, but it uses musical units instead.

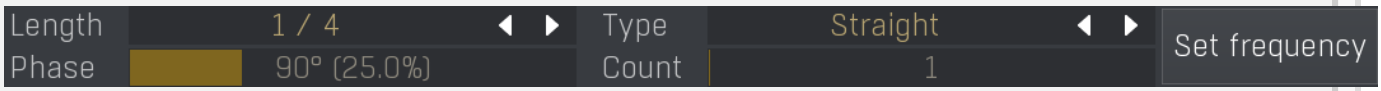
Frequency

Frequency defines the modulation speed.

Sync group

Sync group lets you synchronize the modulators with each other and potentially with other parts of the plugin. It can be controlled only when **to-host synchronization** is disabled, otherwise it is overridden by synchronization from the host. By using the same synchronization group for all modulators you ensure they will always be in-sync even though no other synchronization is used. This can be useful, for example, when you want to modulate different parameters with different shapes or when using some more advanced method, such as using a follower. When the synchronization is enabled, it works on the 'first is the leader' basis, hence the first modulator (the one with lowest index) in Sync group, controls the rest of the modulators in the same group.

Synchronization panel



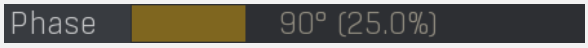
Synchronization panel contains parameters for the to-host synchronization.

Length

Length defines the note length to be used.

Type

Type defines the note type, such as straight notes or triplets, to be used. Together the **Length** and **Type** determine the actual time/delay.
Example: '1/4 Straight' at 120 bpm = a delay of 500 ms, '1/4 Triplet' at 160 bpm = a delay of 281.25 ms.

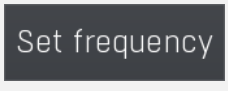


Phase

Phase defines the phase offset of the to-host synchronization.
Range: 0Â° (0%) to 360Â° (100.0%), default 90Â° (25.0%)

Count

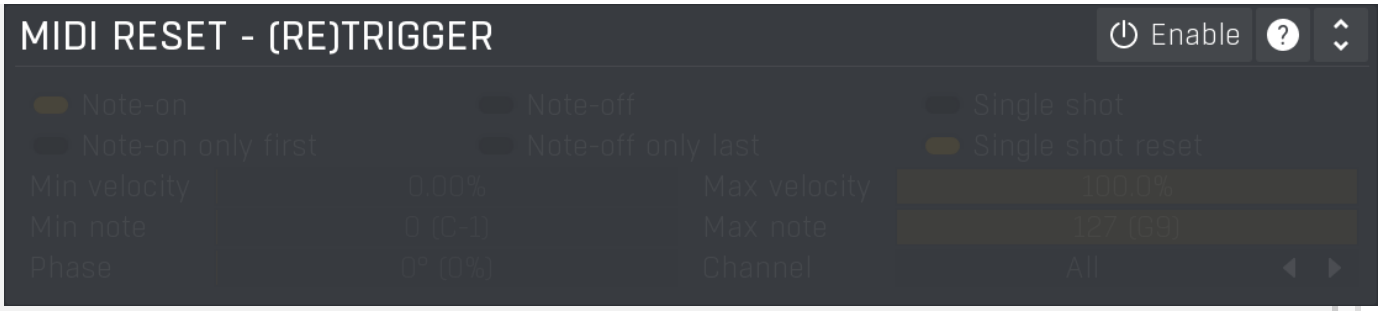
Count defines the number of the units, hence multiplies of the sync length.
Range: 1 to 64, default 1



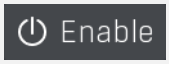
Set frequency

Set frequency button sets the **Frequency** parameter available for the frequency mode so that it matches the current synchronization. That way you can set the modulator's frequency to the current synchronization and then change it a little for example.

MIDI reset panel

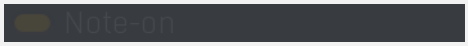


MIDI reset panel configures the MIDI reset feature, which will reset the oscillator when a MIDI note is received or its **MIDI reset parameter** is a target of another modulator or multiparameter. This way you can make the oscillator perform "in-sync" with your playing. Please note that once you enable it, the oscillator will not be in phase-sync with the host.



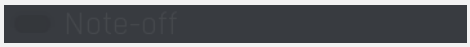
Enable

Enable button enables or disables the feature.



Note-on

Note-on controls if the MIDI reset should occur when a note is pressed.

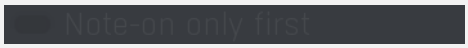


Note-off

Note-off controls if the MIDI reset should occur when a note is released.

Single shot

Single shot button activates the single shot mode in which the oscillator doesn't cycle around but instead only goes once from left to right, then stops until the MIDI reset occurs.

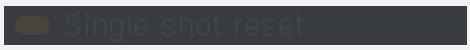


Note-on only first

Note-on only first controls if the MIDI reset should occur when a note is pressed only if it is the first note (thus no other note is being held).

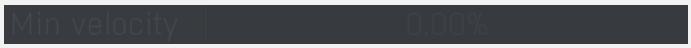
Note-off only last

Note-off only last controls if the MIDI reset should occur when a note is released only if it is the last note (that is, no other note is being held afterwards).



Single shot reset

Single shot reset button defines if the phase should reset to 0 after a single shot period ends. For most waves such as sine it doesn't really matter since the value at 0 (the start of the cycle) is the same as value at 1 (the end of the cycle). But it might matter for saw wave for example.

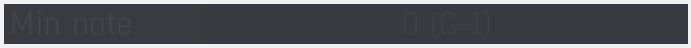


Min velocity

Min velocity defines the minimum velocity that will reset the oscillator.

Max velocity

Max velocity defines the maximum velocity that will reset the oscillator.

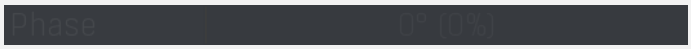


Min note

Min note defines the minimum note that will reset the oscillator.

Max note

Max note defines the maximum note that will reset the oscillator.



Phase

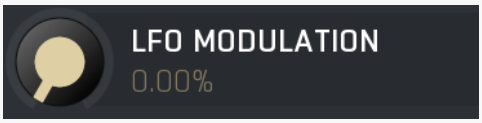
Phase defines the initial oscillator phase after a reset.

Channel

Channel defines note MIDI channel to reset the oscillator.

Follower mode

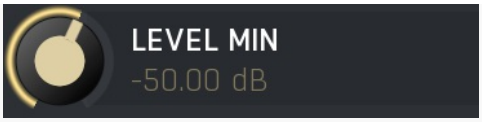
Follower mode makes the modulator follow the input signal level.



LFO modulation

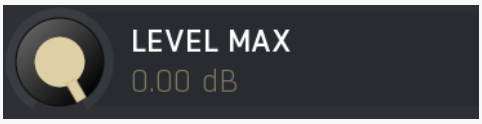
LFO modulation defines the amount of LFO modulation to be applied in addition to the follower. With 0% the modulator uses only the follower; with 100% the modulator does the same job as if the modulator were in **Normal** mode. To set the LFO parameters switch to normal mode temporarily.

Range: 0.00% to 100.0%, default 0.00%



Level min

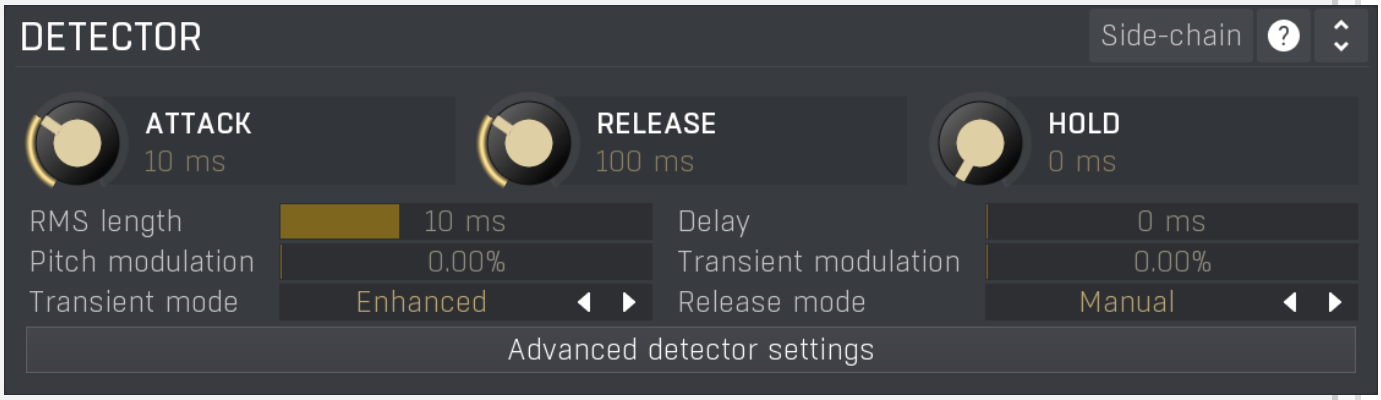
Level min defines the minimum input level that is transformed into a modulator value of 0%. For example if you set the minimum / maximum levels to -50dB ... -20dB, then an input level of -50dB or lower results in a value of 0% and an input level at -20dB or above results in 100%.
Range: -120.00 dB to 0.00 dB, default -50.00 dB



Level max

Level max defines the maximum input level that is transformed into a modulator value of 100%. For example if you set the minimum / maximum levels to -50dB ... -20dB, then an input level of -50dB or lower results in a value of 0% and an input level at -20dB or above results in 100%.
Range: -120.00 dB to 0.00 dB, default 0.00 dB

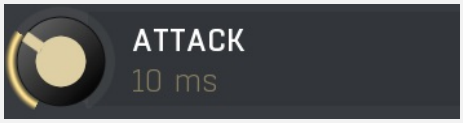
Detector panel



Detector panel contains the dynamic detector parameters, which control how the signal level is measured.

Side-chain input

Side-chain input makes the modulator analyze the side-chain input instead of the regular input.



Attack

Attack defines the attack time, that is how quickly the level detector increases the measured input level. When the input peak level is higher than the current level measured by the detector, the detector moves into the attack mode, in which the measured level is increased depending on the input signal. The higher the input signal, or the shorter the attack time, the faster the measured level rises. Once the measured level exceeds the **Threshold** then the dynamics processing (compression, limiting, gating) will start.

There must be a reasonable balance between attack and **release** times. If the attack is too long compared to the release, the detector will tend to keep the measured level low, because the release would cause that level to fall too quickly. In most cases you may expect the attack time to be shorter than the release time.

To understand the working of a level detector, it is best to cover the typical cases:

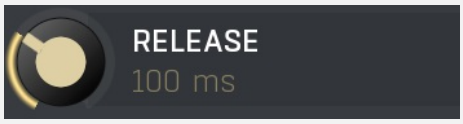
*In a **compressor** the attack time controls how quickly the measured level moves above the threshold and the processor begins compressing. As a result, a very short attack time will compress even the beginning transient of a snare drum for example, hence it would remove the punch. With a very long attack time the measured level may not even reach the threshold, so the compressor may not do anything.*

*In a **limiter** the attack becomes a very sensitive control, defining how much of the signal is limited and how much of it becomes saturated/clipped. If the attack time is very short, limiting starts very quickly and the limiter catches most peaks itself and reduces them, providing lower distortion, but can cause pumping. On the other hand, a higher attack setting (typically above 1ms) will let most peaks through the limiter to the subsequent in-built clipper or saturator, which causes more distortion of the initial transient, but less pumping.*

*In a **gate** the situation is similar to a compressor - the attack time controls how quickly the measured level can rise above the threshold at which point the gate opens. In this case you will usually need very low attack times, so that the gate*

reacts quickly enough. The inevitable distortion can then be avoided using look-ahead and hold parameters.

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level rising, use a shorter attack time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 1000 ms, default 10 ms



Release

Release defines the release time, that is how quickly the level detector decreases the measured input level. The shorter the release time, the faster the response is. Once the attack stage has been completed, when the input peak level is lower than the current level measured by the detector, the detector moves into the release mode, in which the measured level is decreased depending on the input signal. The lower the input signal, or the shorter the release time, the faster the measured level drops. Once the measured level falls under the **Threshold** then the dynamics processing (compression, limiting, gating) will stop.

There must be a reasonable balance between **attack** and release times. If the attack is too long compared to release, the detector would tend to keep the level low, because release would cause the level to fall too quickly. Hence in most cases you may expect the attack time to be shorter than the release time.

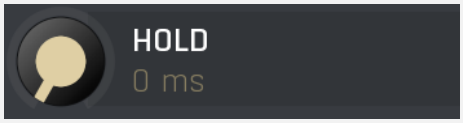
To understand the working of a level detector, it is best to cover the typical cases:

*In a **compressor** the release time controls how quickly the measured level falls below the threshold and the compression stops. As a result a very short release time makes the compressor stop quickly, for example, leaving the sustain of a snare drum intact. On the other hand, a very long release keeps the compression working longer, hence it is useful to stabilize the levels.*

*In a **limiter** the release time keeps the measured level above the limiter threshold causing the gain reduction. Having a very long release time in this case doesn't make sense as the limiter would be working continuously and the effect would be more or less the same as simply decreasing the input gain manually. However too short a release time lets the limiter stop too quickly, which usually causes distortion as the peaks through the limiter to the subsequent in-built clipper or saturator. Hence release time is used to avoid distortion at the expense of decreasing the output level.*

*In a **gate** the situation is similar to a compressor - the release time controls how quickly the measured level can fall below the threshold at which point the gate closes. Having a longer release time in a gate is a perfectly acceptable option. The release time will basically control how much of the sound's sustain will pass.*

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level falling, use a shorter release time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 10000 ms, default 100 ms



Peak hold

Peak hold defines the time that signal level detector holds its maximum before the release stage is allowed to start. As an example, you can imagine that when an attack stage ends there can be an additional peak hold stage and the level is not yet falling, before the release stage starts. This is true only when **true peak** mode is enabled (check the advanced detector settings if available).

It is often used in **gates** to avoid the gated level falling below the threshold too quickly, while having short release times. If you want the gate to close quickly, you need a short release time. But in that case the ending may be too abrupt and even cause some distortion. So you use the peak hold to delay the release stage.

It is also used along with **look-ahead** to avoid distortion in **limiters and compressors**. If you need a very short attack, the attack stage may be too quick and cause distortions. In limiters this attack time is often 0ms, in which case it becomes a clipper. Setting look-ahead and peak hold to the same value will make the detector move ahead in time, so that it can react to attack stages before they actually occur and yet hold the levels for the actual signal to come.
Range: 0 ms to 10000 ms, default 0 ms

RMS length

RMS length smoothes out the values of the input levels (not the input itself), such that the level detector receives the pre-processed signal without so many fluctuations. When set to its minimum value the detector becomes a so-called "peak detector", otherwise it is an "RMS detector".

When you look at a typical waveform in any editor, you can see that the signal is constantly changing and contains various transient bursts and separate peaks. This is especially noticeable with rhythmical signals, such as drums. Trying to imagine how a typical attack/release detector works with such a wild signal may be complex, at least. RMS essentially takes the

surrounding samples and averages them. The result is a much smoother signal with fewer individual peaks and short noise bursts.

RMS length controls how many samples are taken to calculate the average. It stabilizes the levels, but it also causes a slower response time. As such it is great for mastering, when you want to lower the dynamic range in a very subtle way without any instabilities. However, it is not really desirable for processing drums, for example, where the transient bursts may actually be individual drum hits, hence it is usually recommended to use peak detectors for percussive instruments.

Note that the RMS detector has 2 modes - a simplified approximation is used by default, and a true RMS is processor can be enabled from the advanced settings (if provided). Both respond differently, neither of them is better than the other, they are simply different.
Range: 0 ms to 1000 ms, default 10 ms

Delay

Delay defines how much the follower output should be delayed. It is a powerful way to keep attacks intact for example.
Range: 0 ms to 10000 ms, default 0 ms

Pitch modulation

Pitch modulation lets you employ the pitch detector (configurable from the **Pitch tab**) in the detector. This may sound odd at first, but thinking of the input signal, you may measure its level, but you can also measure other properties, such as its pitch, and use them in exactly the same way. While an input level is usually understood as a value in decibels, pitch is a frequency in Hz, so the plugin smartly transforms the frequency to mimic the level axis. When you look at the detector graph afterwards, you can hardly tell the exact pitch in Hz, but that's not really relevant or necessary.
What is this for? Let's show it with an example. Let's say you have an instrument, say a bass, which is playing legato, and you want some kind of effect at the beginning of the note (in case of **Follower** mode) or you just want to restart some kind of filter at the beginning of each note (in case of **Envelope** mode). But since the performance is legato, meaning there are no gaps in between the notes, the level graph is just a steady horizontal line, which is pretty much useless for us. The pitch modulation lets you replace this horizontal line with something much more useful - the pitch. While the level isn't changing much at all, the pitch is changing. The plugin then takes the actual pitch as the input signal, so you can let the plugin follow it in some way, start envelopes when a certain pitch is exceeded, or using **Transformation** you can even let the plugin restart an envelope every time the pitch changes. By setting the pitch modulation half-way you can let the plugin react to both properties, the level and the pitch.
Range: 0.00% to 100.0%, default 0.00%

Transient modulation

Transient modulation lets you detect transients and blend that detection with the control signal. This way you may let the modulator be controlled not by level (alone or in combination with pitch for example), but also by transients detected in either of these properties - level or pitch.
Range: 0.00% to 100.0%, default 0.00%

Transient mode

Transient mode controls the way in which the transients are detected. These simply provide different results, so you should just try the alternative modes if the default one doesn't suit your audio material. **Attack only modes** ignore sustain transients - those moments when the level decreases.

Release mode

Release mode defines how the plugin performs when decreasing level. In **manual mode** this is based only on the **release time**, which is suitable for most cases when the signal has constant characteristics. Automatic release modes can adapt to signals with unstable characteristics.

Automatic and **Automatic fast** modes: the longer the level stays above the threshold, the longer the release time will be and thus, the longer it will take to move below the threshold and end the release stage. The idea is that if the input is loud for some time, it will most likely stay that way for some more time, hence it should be stabilized to avoid unnecessary temporary fluctuations, which could result in pumping.
Both automatic modes increase the release time when the input signal is above the threshold and vice versa. The speed of the increase depends on the **Auto speed** parameter. Automatic fast mode uses full speed immediately after crossing the threshold, automatic mode varies the speed according to the current signal level.

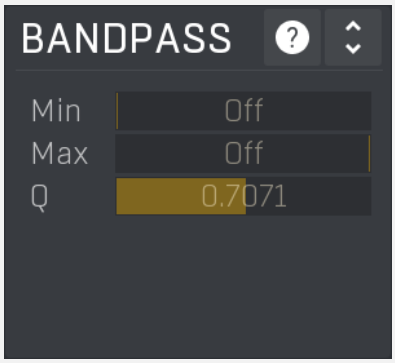
For example, when a guitarist plays softly, the level is low and fluctuates around the threshold and the release time gets slower. So the processor quickly responds to sudden changes. However, when the guitarist starts playing a solo, the level rises and, the longer the solo is, the longer the release time becomes, hence the response becomes slower avoiding unnecessary fluctuations (pumping) when the solo contains small silent sections.

Linear 1 and **Linear 2** modes: the higher the level is, the longer the release. The idea is that if the input is very loud, it will probably stay that way for some time, so it is wise to keep the levels up too. This is similar to the automatic modes, however the main factor is not how long the level is high, but how high it is.
Below the threshold the release time is the same as the attack time, above the threshold the release time rises from the attack time up to the specified release time parameter. Linear 1 mode usually provides higher release times than does Linear 2.

Opto mode: the higher the level is, the shorter the release. So this is kind of the opposite of linear modes. The idea is, that you are expecting short transients, which you wish to deal with. Normally the higher the level would get in such a transient, the longer it would take to get the level below the threshold, so, when used in a compressor for example, these transients would cause unnecessary compression in the sustain stage. The opto detector lowers the level quickly, minimizing the amount of compression in the sustain stage.

*For example, let's say you are compressing a full drumset, but there is a very dominant sharp and short hi-hat sound, so it is appropriate to have short release times. You would use **Opto** mode. But the rest of the drumset deserves a softer treatment, so you want to keep longer release times. Use one of the other modes.*

Band-pass panel



Band-pass panel contains parameters of the follower band-pass filter. Using this feature you can make the follower detect the level of just part of the spectrum instead of all frequencies. For example, when using band-pass from 20Hz to 100Hz the modulator will react mainly to a bass or bass-drum signal.

Minimum

Minimum defines the high-pass filter cut-off frequency. The band-pass is disabled if both the minimum and maximum frequencies are set to their limits, thus from 20Hz to 20kHz.
Range: Off to 20.0 kHz, default Off

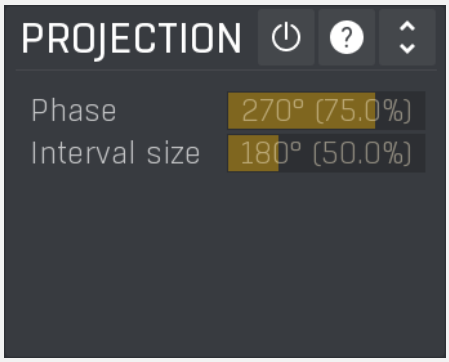
Maximum

Maximum defines the low-pass filter cut-off frequency. The band-pass is disabled if both the minimum and maximum frequencies are set to their limits, thus from 20Hz to 20kHz.
Range: 20.00 Hz to Off, default Off

Q

Q defines the bandwidth for the high-pass and low-pass filters.
Range: 0.0500 to 10.0000, default 0.7071

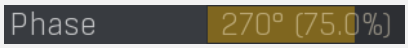
Projection panel



Projection panel contains parameters of projection onto the LFO oscillator shape, which takes the value generated by the modulator and puts it onto the LFO oscillator shape. This features is useful for several creative effects.

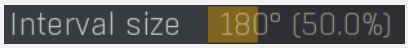
Enable

Enable button enables or disables the projection onto the LFO oscillator shape.



Phase

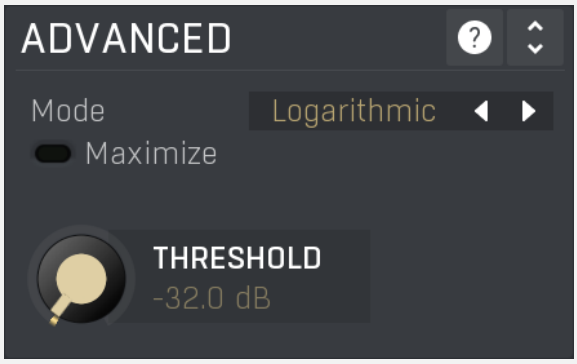
Phase defines the offset from zero of the signal curve. By default it is 75%, because when you look at common oscillator shapes, such as a sine or triangle, at position 75% its value is minimal. Then when you look at the right side, the value is growing up to the 25%, where it becomes the maximum.
Range: 0Â° (0%) to 360Â° (100.0%), default 270Â° (75.0%)



Interval

Interval defines the size of the interval from the oscillator shape in addition to **Phase**. As a result, phase defines where you start on the shape and interval specifies size of the window on the shape. Default value is 50% as for example sine grows from minimum to maximum in 50% of the period.
Range: 0Â° (0%) to 720Â° (200.0%), default 180Â° (50.0%)

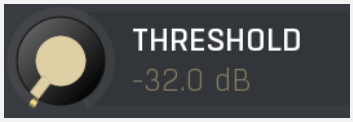
Advanced panel



Advanced panel contains some more advanced features of the level follower.

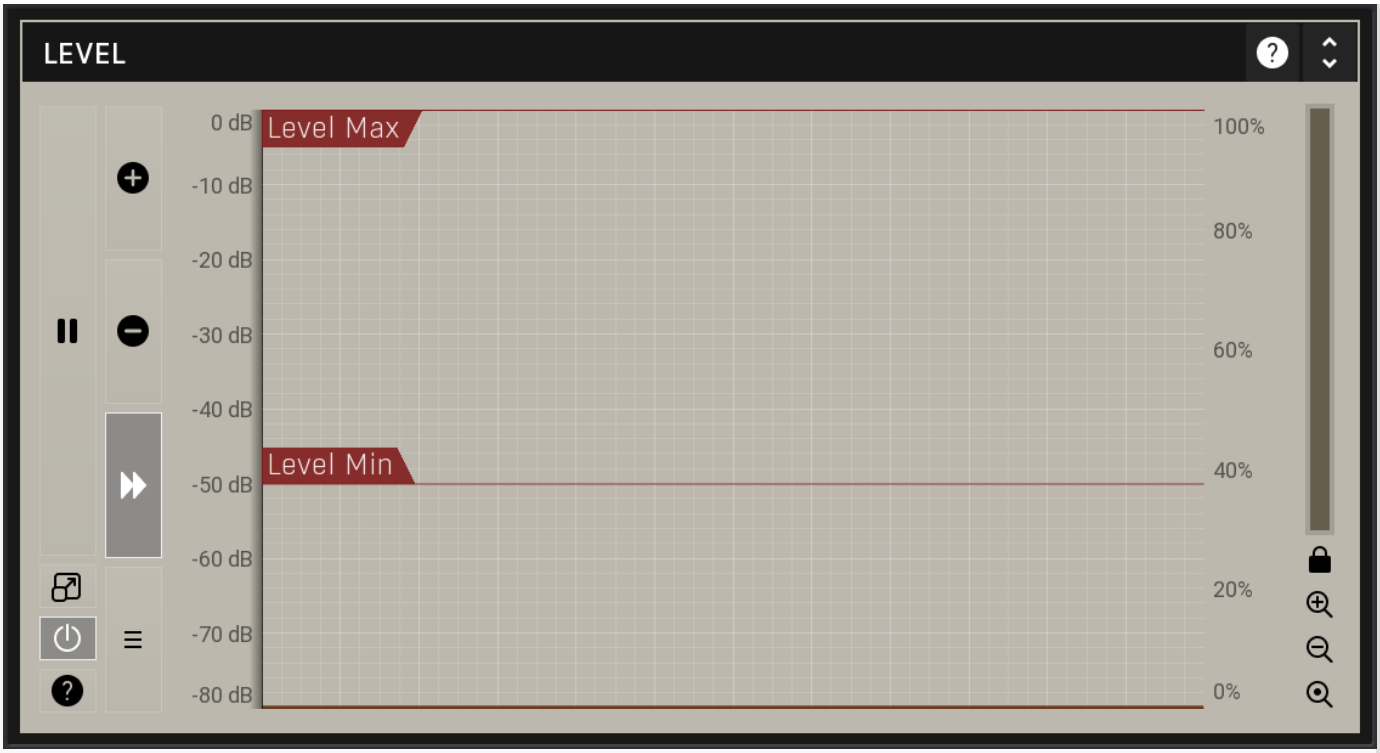
Mode

Mode controls the way in which the audio level is treated. **Linear** mode takes the audio level and uses it directly. This often tends to result in very low modulator values. **Squared** mode treats the squared levels. This is a compromise between linear and logarithmic modes. **Logarithmic** mode is the most aggressive one and usually also the most natural as it emulates the logarithmic behaviour of our ears.
Direct mode is quite different as it doesn't really follow the level but instead takes the audio directly without any attack/release processing. It always takes the mid-range value, (minimum + maximum)/2, from each audio block. This is mostly useful for control signals.
For example, let's say your audio level is now -40dB. Then in linear mode it is treated as 1%, because the value of -40dB equals 0.01. In squared mode it becomes 10%. And finally in logarithmic mode it is 33%, because -40dB is 33% of the way from -60dB to 0dB.



Maximize

Maximize enables threshold-based maximization. Normally the input signal is used to drive the level follower. In this mode however each input sample is treated as 0 or 1, depending on whether it is below or above the threshold. As a result you can get very fast and sharp transitions.
Range: silence to 0.00 dB, default -32.0 dB



Level panel

Level panel contains the metering system showing the follower level. It is indispensable when setting up the follower.
The **orange graph** displays the measured level, which depends on the detector parameters, such as **Attack** or **Release**. In most cases you will want to set the follower so that it responds well to the full range of the audio material. After the detector parameters the level range is the next most important and is available via **Level min** and **Level max** parameters(these can be also adjusted directly from the analyser). In most cases you will want the minimum level to lie just below the lowest signal peaks and maximum level just above the highest peaks.
The **white graph** displays the output modulator values. It includes all the processing that affects the modulator including **LFO modulation** and **Project** features.

Pause

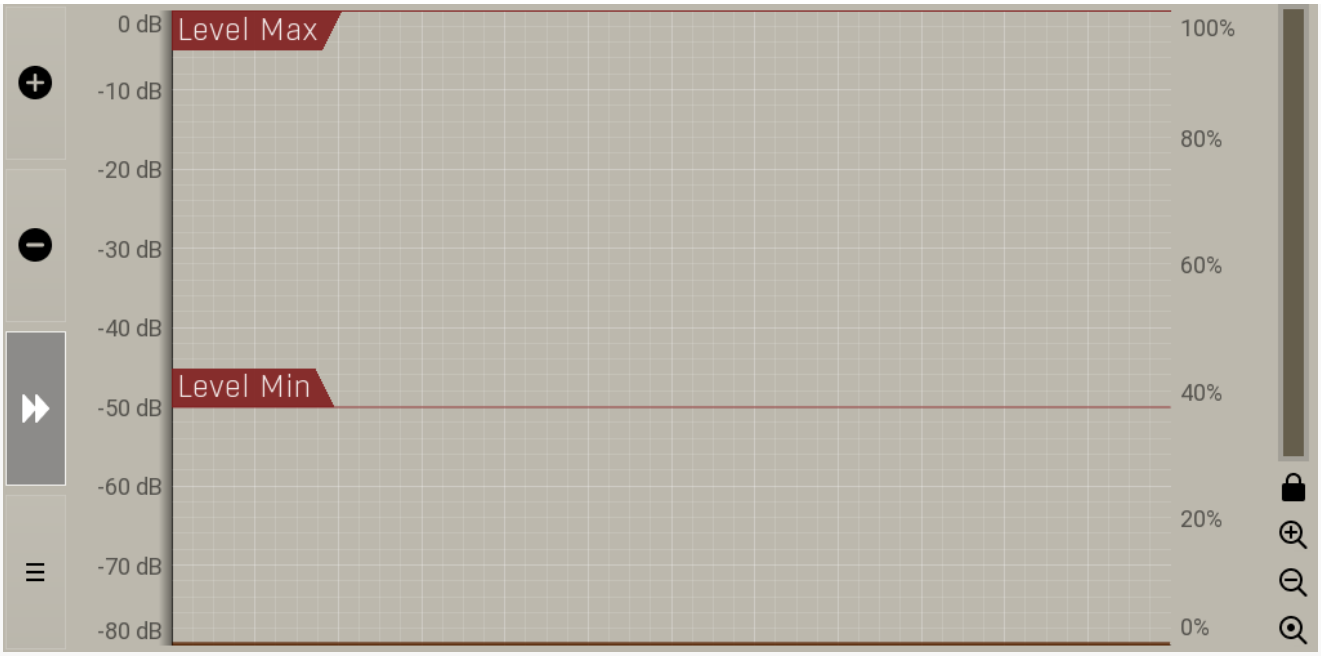
Pause button pauses the processing.

Popup

Popup button shows a pop-up window and moves the whole metering / time-graph system into it. This is especially useful in cases where you cannot enlarge the meters within the main window or such a task is too complicated. The pop-up window can be arbitrarily resized. In metering mode it is useful for easier reading from a distance for example. In time-graph mode it is useful for getting higher accuracy and a longer time perspective.

Enable

Enable button enables or disables the metering system. You can disable it to save system resources.



Time-graph view

Time-graph view shows the measurements over a period of time.

Plus

Plus button increases the time-graph speed (reduces the period that is displayed).

Minus

Minus button decreases the time-graph speed (increases the period that is displayed).

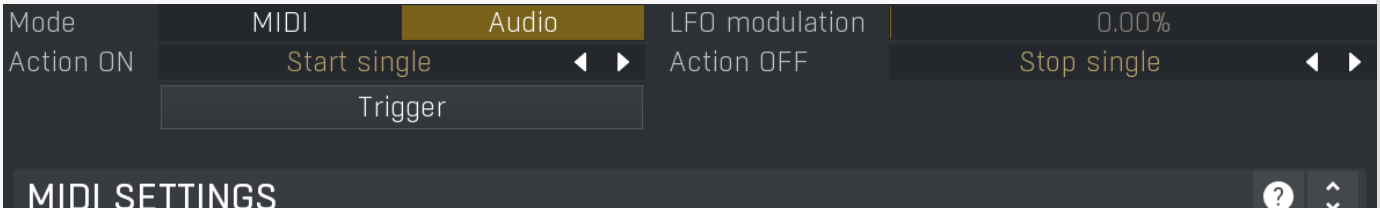
Rewind

Rewind button enables or disables the time-graph static mode. In static mode the graphs are fixed and the current position cycles from left to right; otherwise the graphs move from right to left and the current position is fixed (at the right-hand side).

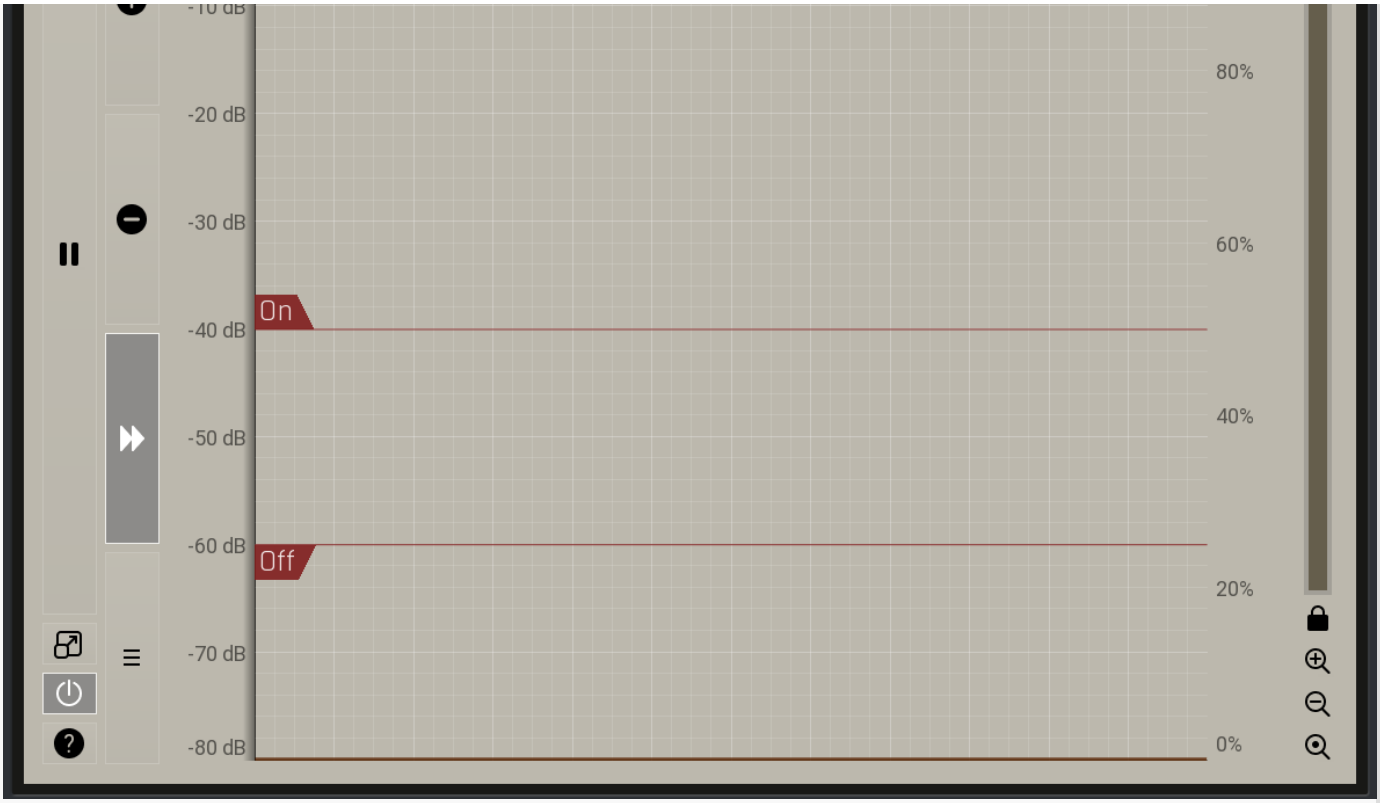
Menu

Menu button displays the time-graph settings. In this window you can control which graphs are displayed, the speed and other relevant parameters.

Envelope mode



CHANNEL
All
NOTE MIN
0 (C-1)
NOTE MAX
127 (G9)
DETECTOR
Side-chain
THRESHOLD ON
-40.00 dB
THRESHOLD OFF
-60.00 dB
DETECTOR HOLD
20 ms
ATTACK
10 ms
RELEASE
10 ms
RMS LENGTH
20 ms
Pitch modulation
0.00%
Transient modulation
0.00%
Transient mode
Enhanced
Release mode
Manual
Advanced detector settings
BANDPASS
Min
Off
Max
Off
Q
0.7071
PROJECTION
Phase
270° (75.0%)
Interval size
180° (50.0%)
ADSR
Custom shape
Percussive
Sync
Delay
0 ms
Smoothness
0.00%
Attack
20 ms
0.00%
Hold
0 ms
0.00 dB
Decay
10 ms
0.00%
Sustain
0.00 dB
Tremolo
0.00%
6.000 Hz
0 ms
Release
200 ms
0.00%
0 s
20ms
40ms
60ms
80ms
100ms
120ms
140ms
160ms
180ms
200ms
220ms
240ms
280ms
ANALYZER
0 dB
100%



Envelope mode makes the modulator generate arbitrary envelopes from input MIDI or by analyzing the audio input level. When using MIDI the modulator responds to input note-on and note-off messages. When using an audio input (if available), the modulator detects the input level and when it exceeds the **Threshold On**, it behaves like a note-on MIDI event. Afterwards when the level drops down below **Threshold Off**, it behaves like a note-off MIDI event. Each event can result in just about any action. By default note-on starts the envelope and note-off initiates the release stage, but a different behaviour is also possible depending on **Action ON** and **Action OFF** parameters.

Mode

Mode controls if the envelope is triggered by audio or by MIDI input.

LFO modulation

LFO modulation defines the amount of LFO modulation applied in addition to the envelope. With 0% the modulator uses only the envelope; with 100% the modulator does the same job as if the modulator were in **Normal** mode. To set the LFO parameters switch to normal mode temporarily.
Range: 0.00% to 100.0%, default 0.00%

Action ON

Action ON controls what happens when a note-on event occurs (either via audio or MIDI).
Start single action, which is the default, means that the envelope will start on the note-on event, but only once, it won't start again until you release all of the keys that are relevant for MIDI triggering only.With audio triggering it works the same as **Start** mode.
Start makes the envelope start every time you press a key, whether another key is already pressed or not. The envelope will seamlessly jump to the attack stage avoiding any abrupt changes.
Start forced is similar, but lets the envelope start from the very beginning every time. So for example if the envelope is currently in a long release stage and the new modulator value is 0.5, then the Start action jumps to a location in the attack stage where there is a value of 0.5 as well, hence avoiding abrupt changes. Start forced action on the other hand starts the whole envelope over from the beginning of the attack stage, where the value is most likely 0.
Start legato is in a way an opposite of the **Start single** in that it starts only if there is at least one other note already playing. As such it can only be used with MIDI triggering. With audio triggering it works the same as **Start** mode.
Ignore action simply ignores this event.
The remaining actions are rather creative and let you do the opposite - initiate release stage and stop the envelope when you press a key.

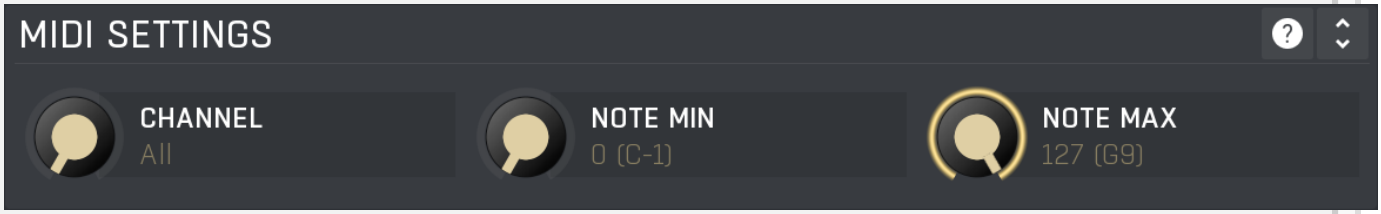
Action OFF

Action OFF controls what happens when a note-off event occurs (either via audio or MIDI).
Stop single action, which is default, means that the envelope will enter the release stage on the note-off event, but only once, at the moment you release the last key (if you were holding more than one) it is relevant for MIDI triggering only.
Stop makes the envelope enter the release stage every time you release a key, whether another key is already pressed or not.
Ignore action simply ignores this event.
The remaining actions are rather creative and let you do the opposite - start the envelope when you release a key.

Trigger

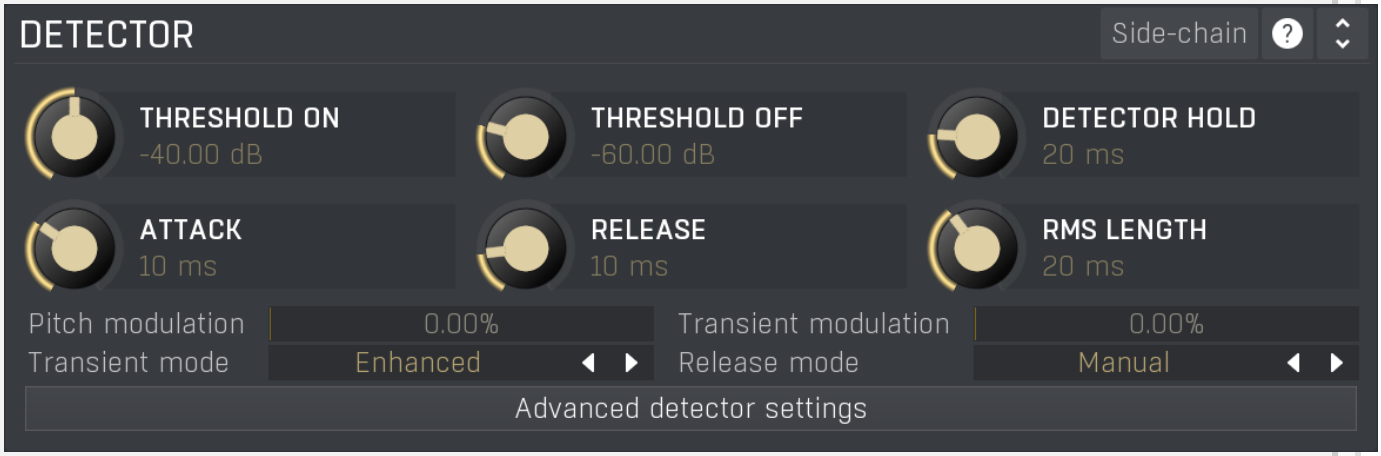
Trigger button servers for manual triggering. It can be associated to other modulators as well for example, so it enables you to trigger the envelope pretty much any way.

MIDI panel



MIDI panel contains parameters of the MIDI event detector.

Detector panel



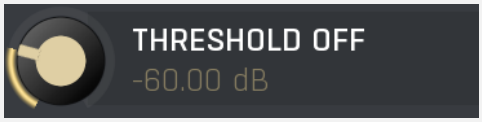
Detector panel contains parameters of the audio event detector.

Side-chain input

Side-chain input makes the modulator analyze the side-chain input instead of the regular input.

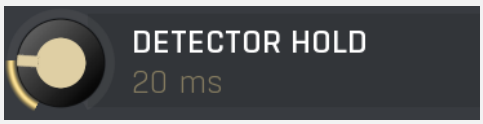
Threshold On

Threshold On defines the note-on level. When the input level rises above it the envelope is started. Then it stays into the sustain stage until the level falls below **Threshold Off** and the release stage is initiated.
Range: -80.00 dB to 0.00 dB, default -40.00 dB



Threshold Off

Threshold Off defines the note off level. When the input level rises above **Threshold On**, the envelope is started. Then it stays into the sustain stage until the level falls below this threshold off and the release stage is initiated.
Range: -80.00 dB to 0.00 dB, default -60.00 dB

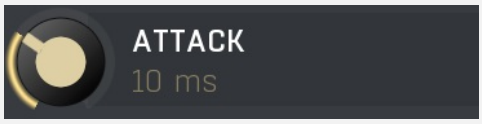


Peak hold

Peak hold defines the time that signal level detector holds its maximum before the release stage is allowed to start. As an example, you can imagine that when an attack stage ends there can be an additional peak hold stage and the level is not yet falling, before the release stage starts. This is true only when **true peak** mode is enabled (check the advanced detector settings if available).

It is often used in **gates** to avoid the gated level falling below the threshold too quickly, while having short release times. If you want the gate to close quickly, you need a short release time. But in that case the ending may be too abrupt and even cause some distortion. So you use the peak hold to delay the release stage.

It is also used along with **look-ahead** to avoid distortion in **limiters and compressors**. If you need a very short attack, the attack stage may be too quick and cause distortions. In limiters this attack time is often 0ms, in which case it becomes a clipper. Setting look-ahead and peak hold to the same value will make the detector move ahead in time, so that it can react to attack stages before they actually occur and yet hold the levels for the actual signal to come.
Range: 0 ms to 10000 ms, default 20 ms



Attack

Attack defines the attack time, that is how quickly the level detector increases the measured input level. When the input peak level is higher than the current level measured by the detector, the detector moves into the attack mode, in which the measured level is increased depending on the input signal. The higher the input signal, or the shorter the attack time, the faster the measured level rises. Once the measured level exceeds the **Threshold** then the dynamics processing (compression, limiting, gating) will start.

There must be a reasonable balance between attack and **release** times. If the attack is too long compared to the release, the detector will tend to keep the measured level low, because the release would cause that level to fall too quickly. In most cases you may expect the attack time to be shorter than the release time.

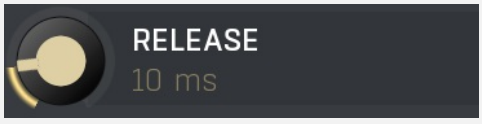
To understand the working of a level detector, it is best to cover the typical cases:

*In a **compressor** the attack time controls how quickly the measured level moves above the threshold and the processor begins compressing. As a result, a very short attack time will compress even the beginning transient of a snare drum for example, hence it would remove the punch. With a very long attack time the measured level may not even reach the threshold, so the compressor may not do anything.*

*In a **limiter** the attack becomes a very sensitive control, defining how much of the signal is limited and how much of it becomes saturated/clipped. If the attack time is very short, limiting starts very quickly and the limiter catches most peaks itself and reduces them, providing lower distortion, but can cause pumping. On the other hand, a higher attack setting (typically above 1ms) will let most peaks through the limiter to the subsequent in-built clipper or saturator, which causes more distortion of the initial transient, but less pumping.*

*In a **gate** the situation is similar to a compressor - the attack time controls how quickly the measured level can rise above the threshold at which point the gate opens. In this case you will usually need very low attack times, so that the gate reacts quickly enough. The inevitable distortion can then be avoided using look-ahead and hold parameters.*

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level rising, use a shorter attack time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 1000 ms, default 10 ms



Release

Release defines the release time, that is how quickly the level detector decreases the measured input level. The shorter the release time, the faster the response is. Once the attack stage has been completed, when the input peak level is lower than the current level measured by the detector, the detector moves into the release mode, in which the measured level is decreased depending on the input signal. The lower the input signal, or the shorter the release time, the faster the measured level drops. Once the measured level falls under the **Threshold** then the dynamics processing (compression, limiting, gating) will stop.

There must be a reasonable balance between **attack** and release times. If the attack is too long compared to release, the detector would tend to keep the level low, because release would cause the level to fall too quickly. Hence in most cases you may expect the attack time to be shorter than the release time.

To understand the working of a level detector, it is best to cover the typical cases:

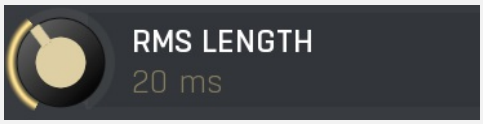
*In a **compressor** the release time controls how quickly the measured level falls below the threshold and the compression stops. As a result a very short release time makes the compressor stop quickly, for example, leaving the sustain of a snare drum intact. On the other hand, a very long release keeps the compression working longer, hence it is useful to stabilize the levels.*

*In a **limiter** the release time keeps the measured level above the limiter threshold causing the gain reduction. Having a very long release time in this case doesn't make sense as the limiter would be working continuously and the effect would*

be more or less the same as simply decreasing the input gain manually. However too short a release time lets the limiter stop too quickly, which usually causes distortion as the peaks through the limiter to the subsequent in-built clipper or saturator. Hence release time is used to avoid distortion at the expense of decreasing the output level.

*In a **gate** the situation is similar to a compressor - the release time controls how quickly the measured level can fall below the threshold at which point the gate closes. Having a longer release time in a gate is a perfectly acceptable option. The release time will basically control how much of the sound's sustain will pass.*

In a modulator, the detector is driving other parameters, a filter cut-off frequency for example, and the situation really depends on the target. If you want the detector to react quickly on the input level falling, use a shorter release time; if you want it to follow the flow of the input signal slowly, use longer attack and release times.
Range: 0 ms to 10000 ms, default 10 ms



RMS length

RMS length defines the window length used for smoothing the input. In most cases the input waveform contains lots of separate peaks and short transients. All of them would generate note-on events and the spaces between them would similarly cause note-offs. RMS is used to smooth the input. The longer the window, the longer interval it takes and the longer delay it exhibits. If it is too short, unpredictable behaviour can be expected.
Range: 0 ms to 1000 ms, default 20 ms

Pitch modulation

Pitch modulation lets you employ the pitch detector (configurable from the **Pitch tab**) in the detector. This may sound odd at first, but thinking of the input signal, you may measure its level, but you can also measure other properties, such as its pitch, and use them in exactly the same way. While an input level is usually understood as a value in decibels, pitch is a frequency in Hz, so the plugin smartly transforms the frequency to mimic the level axis. When you look at the detector graph afterwards, you can hardly tell the exact pitch in Hz, but that's not really relevant or necessary.
What is this for? Let's show it with an example. Let's say you have an instrument, say a bass, which is playing legato, and you want some kind of effect at the beginning of the note (in case of **Follower** mode) or you just want to restart some kind of filter at the beginning of each note (in case of **Envelope** mode). But since the performance is legato, meaning there are no gaps in between the notes, the level graph is just a steady horizontal line, which is pretty much useless for us. The pitch modulation lets you replace this horizontal line with something much more useful - the pitch. While the level isn't changing much at all, the pitch is changing. The plugin then takes the actual pitch as the input signal, so you can let the plugin follow it in some way, start envelopes when a certain pitch is exceeded, or using **Transformation** you can even let the plugin restart an envelope every time the pitch changes. By setting the pitch modulation half-way you can let the plugin react to both properties, the level and the pitch.
Range: 0.00% to 100.0%, default 0.00%

Transient modulation

Transient modulation lets you detect transients and blend that detection with the control signal. This way you may let the modulator be controlled not by level (alone or in combination with pitch for example), but also by transients detected in either of these properties - level or pitch.
Range: 0.00% to 100.0%, default 0.00%

Transient mode

Transient mode controls the way in which the transients are detected. These simply provide different results, so you should just try the alternative modes if the default one doesn't suit your audio material. **Attack only modes** ignore sustain transients - those moments when the level decreases.

Release mode

Release mode defines how the plugin performs when decreasing level. In **manual mode** this is based only on the **release time**, which is suitable for most cases when the signal has constant characteristics. Automatic release modes can adapt to signals with unstable characteristics.

Automatic and **Automatic fast** modes: the longer the level stays above the threshold, the longer the release time will be and thus, the longer it will take to move below the threshold and end the release stage. The idea is that if the input is loud for some time, it will most likely stay that way for some more time, hence it should be stabilized to avoid unnecessary temporary fluctuations, which could result in pumping.
Both automatic modes increase the release time when the input signal is above the threshold and vice versa. The speed of the increase depends on the **Auto speed** parameter. Automatic fast mode uses full speed immediately after crossing the threshold, automatic mode varies the speed according to the current signal level.

For example, when a guitarist plays softly, the level is low and fluctuates around the threshold and the release time gets slower. So the processor quickly responds to sudden changes. However, when the guitarist starts playing a solo, the level rises and, the longer the solo is, the longer the release time becomes, hence the response becomes slower avoiding unnecessary fluctuations (pumping) when the solo contains small silent sections.

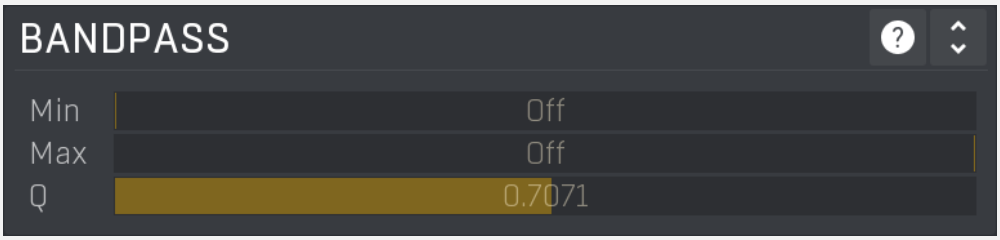
Linear 1 and **Linear 2** modes: the higher the level is, the longer the release. The idea is that if the input is very loud, it

will probably stay that way for some time, so it is wise to keep the levels up too. This is similar to the automatic modes, however the main factor is not how long the level is high, but how high it is.
Below the threshold the release time is the same as the attack time, above the threshold the release time rises from the attack time up to the specified release time parameter. Linear 1 mode usually provides higher release times than does Linear 2.

Opto mode: the higher the level is, the shorter the release. So this is kind of the opposite of linear modes. The idea is, that you are expecting short transients, which you wish to deal with. Normally the higher the level would get in such a transient, the longer it would take to get the level below the threshold, so, when used in a compressor for example, these transients would cause unnecessary compression in the sustain stage. The opto detector lowers the level quickly, minimizing the amount of compression in the sustain stage.

*For example, let's say you are compressing a full drumset, but there is a very dominant sharp and short hi-hat sound, so it is appropriate to have short release times. You would use **Opto** mode. But the rest of the drumset deserves a softer treatment, so you want to keep longer release times. Use one of the other modes.*

Band-pass panel



Band-pass panel contains parameters of the envelope detector band-pass. Using this feature you can make the enveloper detect level of just part of the spectrum instead of all frequencies. For example, when using a band-pass from 20Hz to 100Hz the modulator will react mainly to a bass or bass-drum.

Minimum

Minimum defines the high-pass filter cut-off frequency. The bandpass is disabled if both frequencies are set to their limits, thus from 20Hz to 20kHz.
Range: Off to 20.0 kHz, default Off

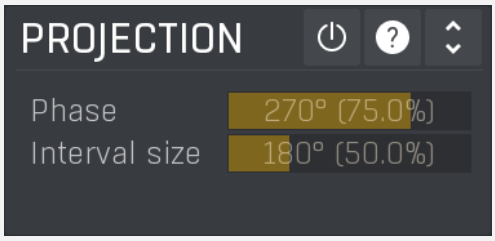
Maximum

Maximum defines the low-pass filter cut-off frequency.The bandpass is disabled if both frequencies are set to their limits, thus from 20Hz to 20kHz.
Range: 20.00 Hz to Off, default Off

Q

Q defines the bandwidth for the high-pass and low-pass filters.
Range: 0.0500 to 10.0000, default 0.7071

Projection panel



Projection panel contains parameters of projection onto the LFO oscillator shape, which takes the value generated by the modulator and puts it onto the LFO oscillator shape. This features is useful for several creative effects.

Enable

Enable button enables or disables the projection onto the LFO oscillator shape.

Phase

Phase defines the offset from zero of the signal curve. By default it is 75%, because when you look at common oscillator shapes, such as a sine or triangle, at position 75% its value is minimal. Then when you look at the right side, the value is growing up to the 25%, where it becomes the maximum.
Range: 0Â° (0%) to 360Â° (100.0%), default 270Â° (75.0%)

Interval

Interval defines the size of the interval from the oscillator shape in addition to **Phase**. As a result, phase defines where you start on the shape and interval specifies size of the window on the shape. Default value is 50% as for example sine grows from minimum to maximum in 50% of the period.
Range: 0Â° (0%) to 720Â° (200.0%), default 180Â° (50.0%)

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Copy

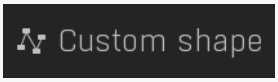
Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.

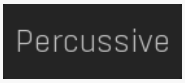
Random

Random button generates random settings using the existing presets.



Custom shape

Custom shape button enables custom shape mode, which lets you draw your own attack and release stages using the envelope system. Both stages are then automatically connected to form the resulting envelope.



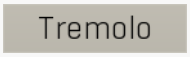
Percussive

Percussive button activates the immediate release mode in which case the note-off causes an immediate switch to the release stage. If this is disabled, the release stage does not occur until the whole attack/decay stage finishes.

Sync

Sync button controls the ADSR tempo sync feature. By default this is disabled and means that all times are followed exactly, meaning that if **Attack** is say 100ms, then it will be 100ms indeed. Tempo sync lets the plugin adjust the times to ensure it will be always in sync with the host tempo. In this case 100ms may become say 125ms if the tempo is 120bpm, because 125ms is the length of a 16th note. This makes it extremely simple to convert any envelope to a tempo-synced one. The plugin always chooses the nearest longer note, in other words it always round up.
Straight and **Triplets** modes automatically find 'nice' values.

For example, if a 16th note takes 100ms, the attack time is 550ms, and the sync mode is straight, then the plugin checks for 100ms, find out that it is too low, so it checks 8th note, being 200ms, still too low, then continues with quarter note, which takes 400ms, and still not enough, finally 800ms corresponding to a half note is the one, so the resulting time will be 800ms.
Triplet cases are more complex, but the principle is the same.
1/16, **1/8** and **1/4** modes choose the nearest higher multiply of the base note length. For example, if a 16th note takes 100ms, the attack time is 550ms, and the sync mode is 1/16, the resulting time will be 600ms.

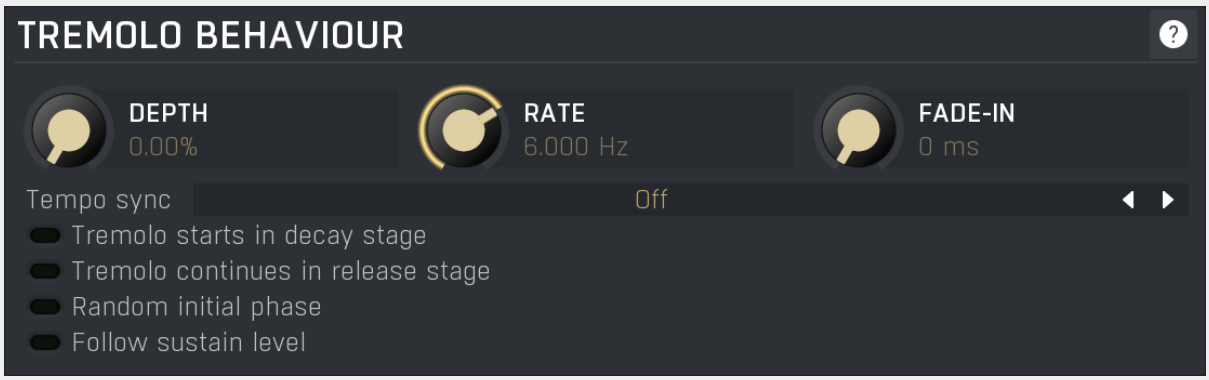


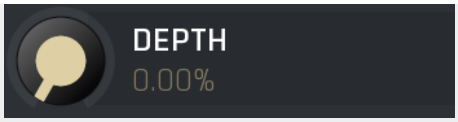
Tremolo

Tremolo button displays additional tremolo settings, containing tremolo behaviour and shape.

Tremolo settings

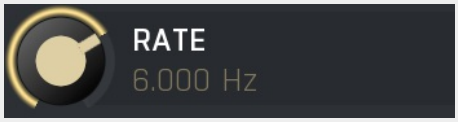
Tremolo behaviour





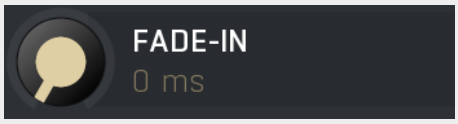
Depth

Depth controls the amount of tremolo mixed in the sustain stage (or potentially before).



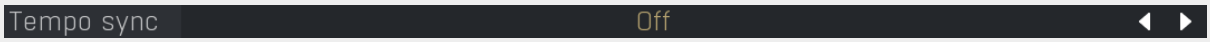
Rate

Rate controls the tremolo rate and is relevant only if tempo sync is not used.



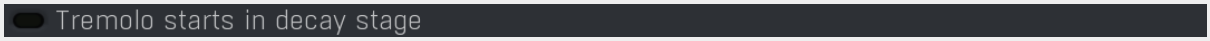
Fade-in

Fade-in controls the length of the tremolo fade-in. It is especially useful when you want to use the random initial phase feature to avoid the initial discontinuity when the tremolo kicks in.



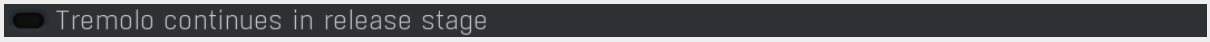
Tempo sync

Tempo sync lets you synchronize the tremolo to the host's tempo.



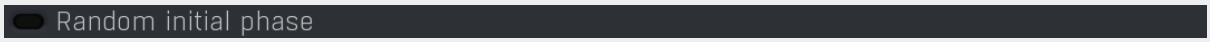
Tremolo starts in decay stage

Tremolo starts in decay stage makes the tremolo start during the decay stage. By default this is disabled and the tremolo starts in the sustain stage. When it is enabled you will most likely have a longer decay and also a longer tremolo fade-in, so that the tremolo slowly comes in as the envelope is decaying.



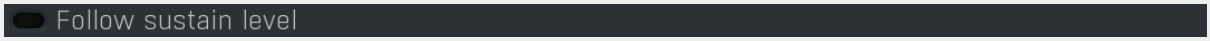
Tremolo continues in release stage

Tremolo continues in release stage makes the tremolo continue with the tremolo during the release stage. By default this is disabled and the tremolo stops as soon as the release stage starts.



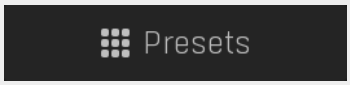
Random initial phase

Random initial phase makes the tremolo start with a random phase. By default this is disabled and the tremolo starts always starts in the 0 phase, which ensures the tremolo always starts in the same way. However if you play multiple notes at once, the tremolo will be exactly the same, while you may want it to be different for each note and make it sound more 'human'. Enabling this option also activates a short **tremolo fade-in** to avoid initial discontinuity.



Follow sustain level

Follow sustain level makes the tremolo level based on sustain level. When this is disabled, the tremolo rarely reaches up to 100% level. However if the sustain level is say -20dB, then the tremolo actually cannot exceed 1% (which is -20dB), so it is clipped. It can however go upwards to 100%. This naturally changes the actual tremolo shape. If you want to avoid that and make sine really be a sine for example, enable this option, and in the case above the tremolo will really go up/down -20dB if set to 100%.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

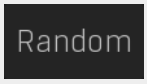
Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipbоard.

Paste

Paste button loads the settings from the system clipboard.



Random

Random button generates random settings using the existing presets.

Normal

Normal button switches the generator into the normal mode, which lets you edit the shape of the oscillator. This is especially advantageous for low-frequency oscillators, where the shape matters even though it doesn't have any physical meaning.

Convert

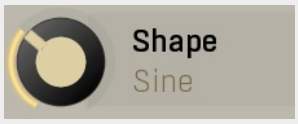
Convert button converts the current shape into harmonic-based representation. Please note that since the number of harmonics is limited, the result will not perfectly resemble the original shape.

Harmonics

Harmonics button switches the generator into the harmonics mode, which lets you edit the levels and phases of individual harmonics. This is especially advantageous for high-frequency oscillators, hence sound generators.

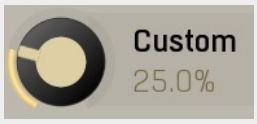
Signal generator in Normal mode

Signal generator in Normal mode works by generating the oscillator shape using a combination of several curves - a predefined set of standard curves, custom shape, step sequencer and custom sample. It also post-processes the shape using several filters including smoothing to custom transformations. This is especially useful when using the oscillator as an LFO (low-frequency-oscillator), where the harmonic contents does not really matter, but the shape does.



Shape

Shape controls the main shape used by the signal generator. There are several predefined shapes: exponential, triangle, sine power 8, sine power 4, sine square, sine, harmonics, more harmonics, disharmonics, sine square root, sine 4 root, rectangle, rect-saw, saw, noise and mess. You can choose any of them or interpolate between any 2 adjacent shapes using this control.



Custom

Custom controls the amount of the custom shape that is blended into the main shape.

Edit

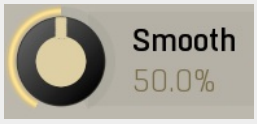
Edit button shows the custom shape editor.

Step

Step controls the amount of the step sequencer shape that is blended into the main shape (which has already been blended with the custom shape).

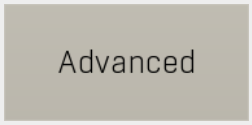
Edit

Edit button shows the step sequencer editor.



Smooth

Smooth controls the amount of smoothing. Many shapes, especially those produced by the step sequencer, have rough jagged edges, which may be advantageous, but when used to modulate certain parameters, the output may be clicking or causing other artifacts. Smoothness helps it by smoothing the whole signal shape out and removing these rough edges.



Advanced

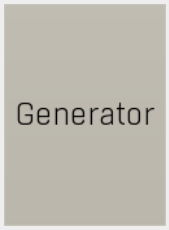
Advanced button displays an additional window with more advanced settings for post-processing the signal shape, such as harmonics or custom transformations.

Signal generator in Harmonics mode

Signal generator in Harmonics mode works by generating the oscillator shape using individual harmonics. Essentially a harmonic is a sine wave. The first harmonic, known as the fundamental, fits once in the oscillator time period, hence it is the same as selecting sine wave in the **Normal mode**. The second harmonic fits twice, the third three times etc. In theory, any shape you create in normal mode can be converted into harmonics. However, this approach to signal generation needs an enormous number of harmonics, which is both inefficient to calculate and mostly hard to edit. Therefore, the harmonic mode can process up to 256 harmonics, which is enough for very complex spectrums, however it is still not enough to generate an accurate square wave for example. If your goal is to create basic shapes, it is better to use the normal mode.

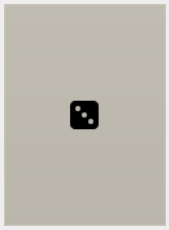
It is nearly impossible to say how a particular curve will sound when used as a high-frequency oscillator in a synthesizer, just by looking at its shape. Harmonics mode, on the other hand, is directly related to human hearing and makes this process very simple. In general, the more harmonics you add, the richer the sound will be. The higher the harmonic, the higher the tone. Usually, one leaves the first harmonic enabled too, as this is the fundamental tone, however you may experiment with more dissonant sounds without it.

Editing harmonics can be time consuming unless you hear what you want, so a signal generator is also available. This great tool lets you generate a random spectrum by a single click. You can also open the **Generator** settings and edit its parameters, which basically control the audio properties in a more natural way - using parameters such as complexity, harmonicity etc.



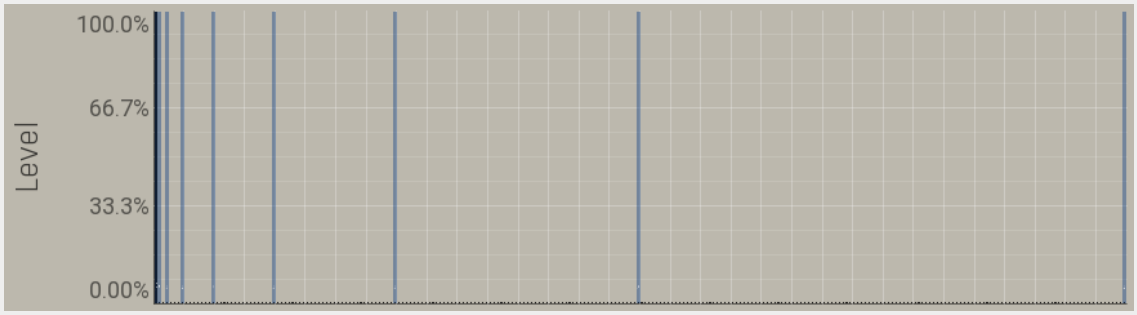
Generator

Generator button shows a powerful harmonics generator, which can create unlimited number of various timbres and even analyze a sample and extract harmonics from it.



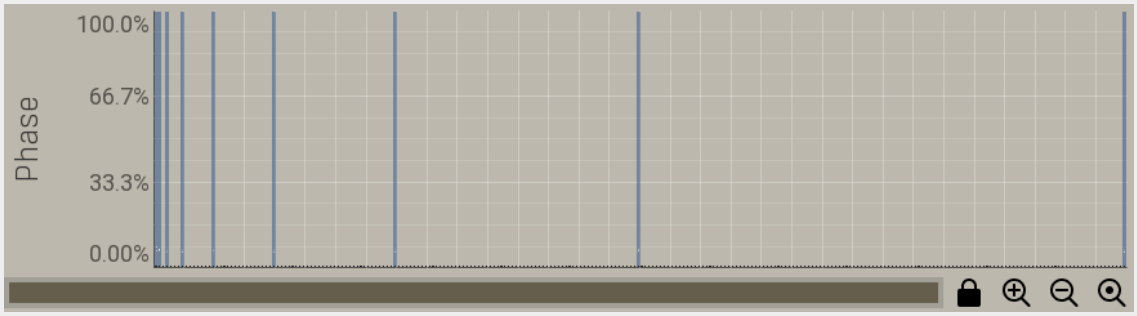
Randomize

Randomize button selects random parameters for the harmonics generator, so you can use it to get a random sound character instantly. Hold **Ctrl** to slightly modify existing generator settings instead of completely changing them.



Magnitudes graph

Magnitudes graph contains the levels of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.



Phases graph

Phases graph contains the phases of the individual harmonics. The highlighted bars are octaves, thus the 1st, 2nd, 4th, 8th harmonic etc.

0 ms **Delay**

Delay lets you shift the entire envelope forwards in time. While this doesn't make much sense for a global instrument envelope for instance, it may be well useful to control characteristics of evolving sounds.

20 ms **Attack**

Attack controls the length of the initial stage of the envelope. It is one of the most important parameters controlling how quick the initial transient is. For most instruments the length is quick short, but for pads and other slowly evolving sounds it is quite common to set this to several seconds.

0 ms **Hold**

Hold specifies the time the level stays at maximum after the attack stage.

10 ms **Decay**

Decay controls the time it takes for the level to drop from the maximum to the **Sustain**. If the sustain is 0dB, then this parameter has no effect, because in a way the sustain stage starts immediately after the attack.

0.00 dB **Sustain**

Sustain controls the sustain level. For most sounds the initial attack transient is the highest point of the entire sound. Imagine playing a string instrument, such as a guitar, the initial hit to the strings is represented by the attack+hold+decay sections and is the most prominent. After that the level drops to the sustain stage, where it holds for most of the time.

0.00% **Tremolo**

Tremolo defines the amount of the tremolo effect that is engaged in the sustain, or even in the decay section and continues until the envelope ends. While this is a rather unusual feature for an envelope to have, it is very handy for simulating various effects human players do when performing on real instruments, such as the tremolo or vibrato.

200 ms **Release**

Release controls the length of the release section, which usually starts when a note is released.

0.00% **Attack shape**

Attack shape controls the shape of the attack section and defines its sound character.

0.00 dB **Hold level**

Hold level controls the level of the hold section. By default it equals maximum meaning that the hold section actually holds the maximum level. However by making it lower you can sort of simulate 2 separate decay sections, first going from maximum to hold level, second going from hold level to sustain.

0.00% **Decay shape**

Decay shape controls the shape of the decay section and defines its sound character.

6.000 Hz **Tremolo rate**

Tremolo rate controls the speed of the tremolo. In the tremolo settings it is possible to control additional characteristics including tempo sync.

0.00% **Release shape**

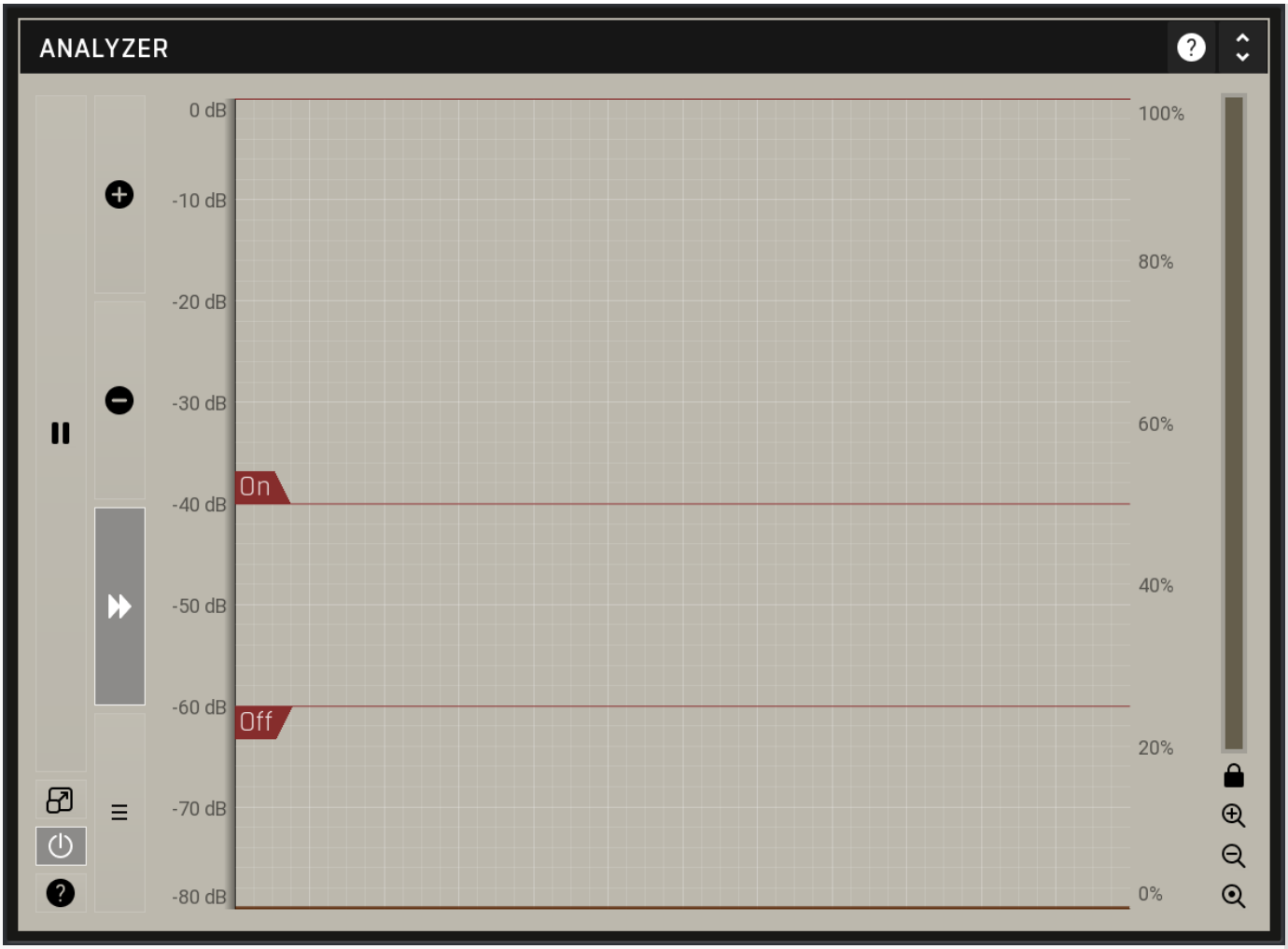
Release shape controls the shape of the release section and defines its sound character.

0.00% **Smoothing**

Smoothing lets you smoothen the entire envelope avoiding abrupt jumps. Note that in some cases involving short jumps the results may be a bit obscure.

0 ms **Tremolo fade-in**

Tremolo fade-in defines the time for the tremolo to reach its full level. It is a natural behaviour of human players (on say a saxophone) that they don't start a full tremolo immediately and rather let the modulation rise to maximum over a period of time.



Analyzer panel

Analyzer panel contains the metering system showing the envelope level. It is indispensable when setting up the envelope. The **orange graph** (assuming the default color) displays the measured level, which depends on the detector parameters, such as **RMS length**. Its purpose is to smooth the input and to avoid extremely fast fluctuations. The main goal will be to set the **Threshold On** and **Threshold Off** properly, so that the events are well detected and there are no false events. Both parameters can be adjusted directly from the graph. In most cases the threshold on will be placed above the threshold off. The **white graph** (assuming the default color) displays the modulator values. It includes all the processing that affects the modulator including **LFO modulation** and **Project** features.

Pause

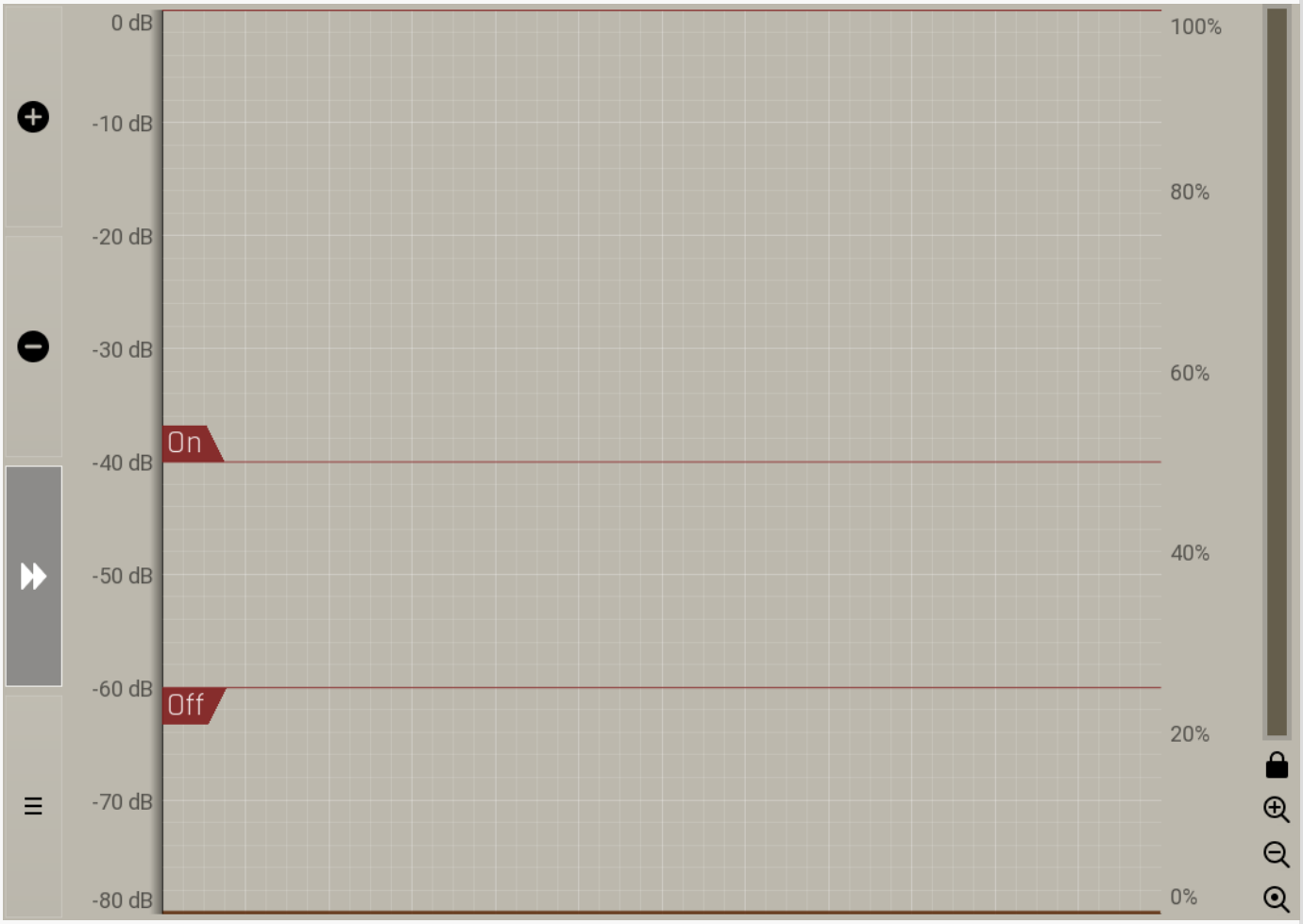
Pause button pauses the processing.

Popup

Popup button shows a pop-up window and moves the whole metering / time-graph system into it. This is especially useful in cases where you cannot enlarge the meters within the main window or such a task is too complicated. The pop-up window can be arbitrarily resized. In metering mode it is useful for easier reading from a distance for example. In time-graph mode it is useful for getting higher accuracy and a longer time perspective.

Enable

Enable button enables or disables the metering system. You can disable it to save system resources.



Time-graph view

Time-graph view shows the measurements over a period of time.

Plus

Plus button increases the time-graph speed (reduces the period that is displayed).

Minus

Minus button decreases the time-graph speed (increases the period that is displayed).

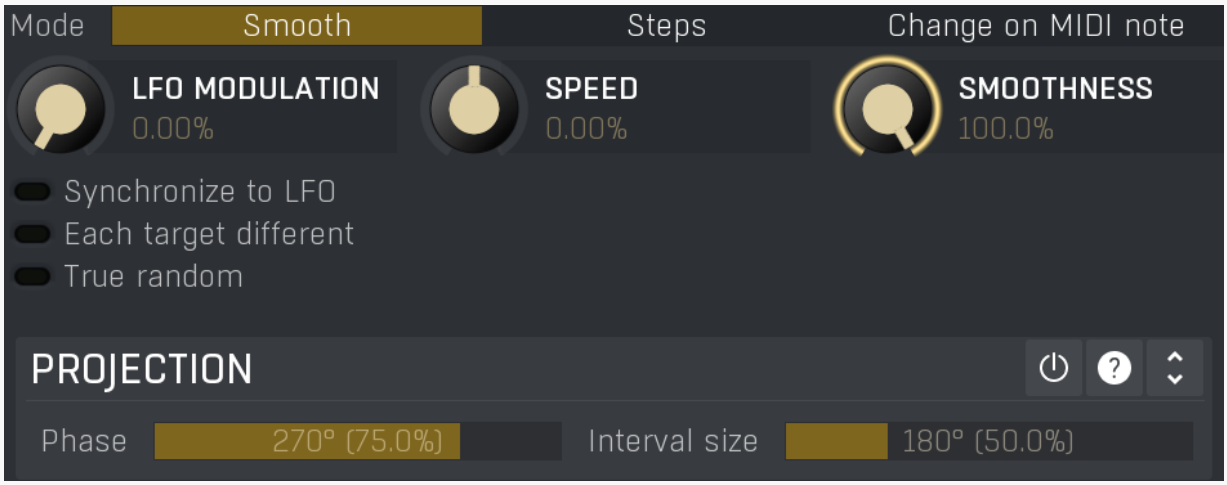
Rewind

Rewind button enables or disables the time-graph static mode. In static mode the graphs are fixed and the current position cycles from left to right; otherwise the graphs move from right to left and the current position is fixed (at the right-hand side).

Menu

Menu button displays the time-graph settings. In this window you can control which graphs are displayed, the speed and other relevant parameters.

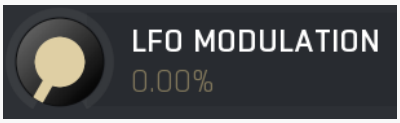
Random mode



Random mode makes the modulator generate a pseudorandom sequence. Please note that despite its name, it is created so that it generates the same sequence every time. However the generator is linked to the **Speed** parameter, so if you change it, the whole sequence changes.

Mode

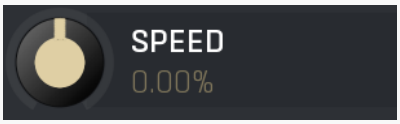
Mode defines the behaviour of the randomizer.
Smooth produces a continuous random modulation. **Smoothness** then controls how smooth it will be, where 0% means it will connect distinct values by straight lines, 100% means the modulation will be a completely smooth curve walking through these random points.
Steps produces a step change every particular time interval. It can also granularize it to s specified number of possible values according to **Smoothness** value. 100% disables the granularization. Otherwise the number of steps is the number of percentage values, so 3% means there will be 3 possible values, equally distributed over the range, let's call them 0%, 50% and 100%. Since it doesn't make sense to have 0 or 1 steps, the minimum is always 2. 2 steps essentially means the modulator is randomly switching between the minimum and maximum values for all associated parameters.
Change on MIDI note generates a random value every time a MIDI note is received by the plugin.



LFO modulation

LFO modulation defines the amount of LFO modulation applied in addition to the random generator. With 0% the modulator uses only the randomizer; with 100% the modulator does the same job as if the modulator were in **Normal** mode. To set the LFO parameters switch to normal mode temporarily.

Range: 0.00% to 100.0%, default 0.00%



Speed

Speed defines the speed of the random changes proportional to the current tempo. 0% means that the speed is the same as your song's tempo.
Range: -1000.0% to 1000.0%, default 0.00%

Smoothness

Smoothness defines the amount of smoothing of the randomizer curve in order to minimize abrupt edges.
Range: 0.00% to 100.0%, default 100.0%

Synchronize to LFO

Synchronize to LFO lets you synchronize the speed of the random sequence to LFO (Normal mode), hence also to your host. **Speed** is still applicable and, for example, +100% means 2x speed, +200% means 4x the speed etc.

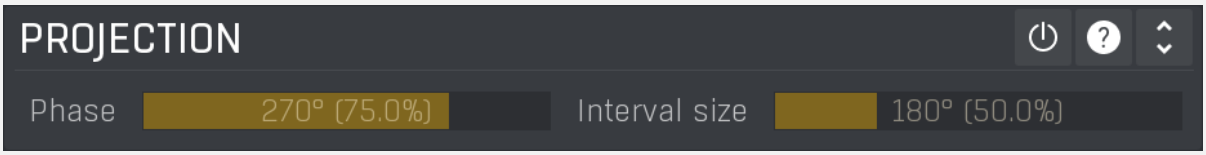
Each target different

Each target different transforms value for each target, so that you can easily produce lots of different random values.

True random

True random makes the modulator produce a true pseudo-random sequence independent of the current position within the project. By default this is disabled, so that every time you play your project, it sounds the same. But you might want to enable this option, for live performances for example.

Projection panel



Projection panel contains parameters of projection onto the LFO oscillator shape, which takes the value generated by the modulator and puts it onto the LFO oscillator shape. This features is useful for several creative effects.

Enable

Enable button enables or disables the projection onto the LFO oscillator shape.

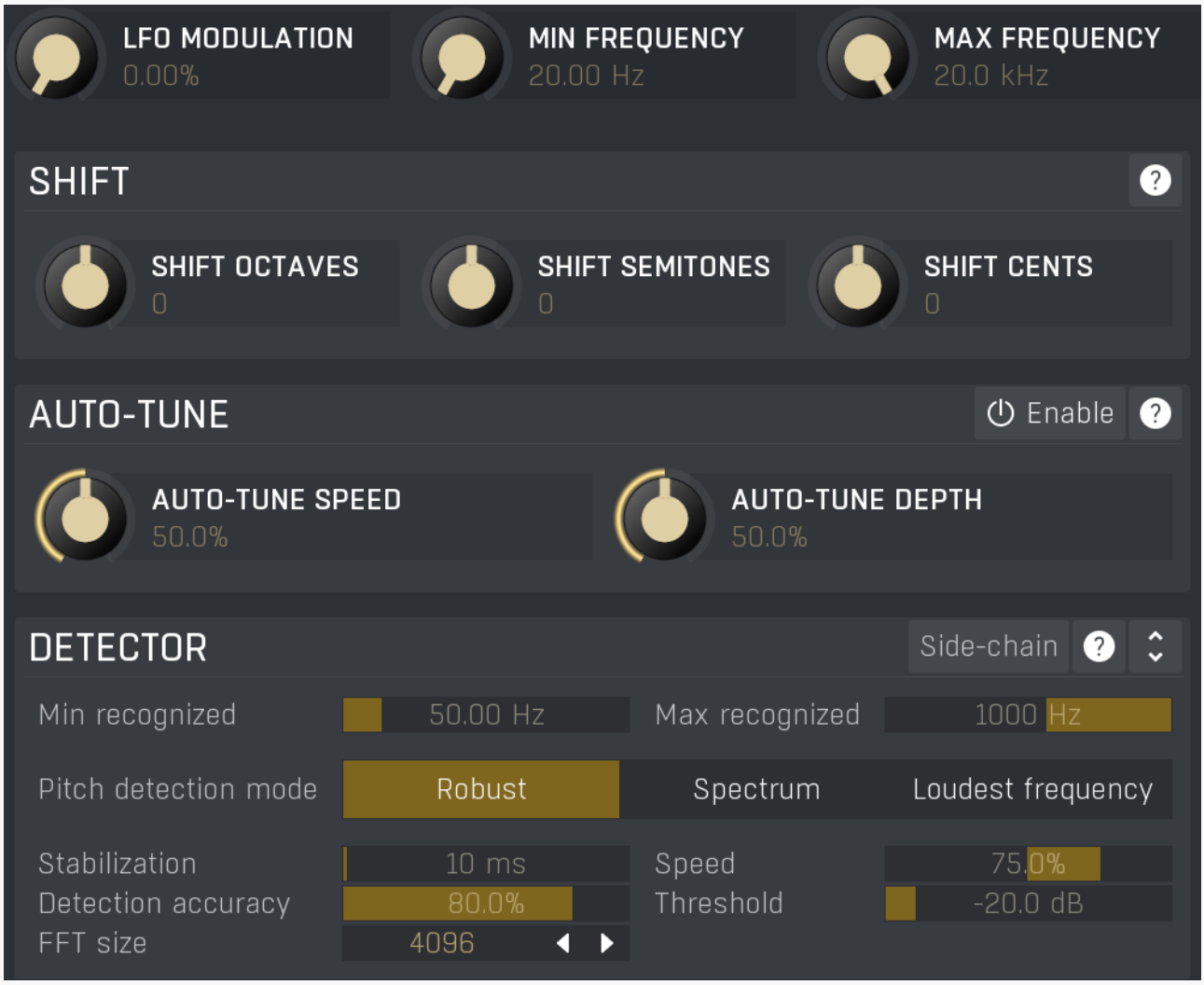
Phase

Phase defines the offset from zero of the signal curve. By default it is 75%, because when you look at common oscillator shapes, such as a sine or triangle, at position 75% its value is minimal. Then when you look at the right side, the value is growing up to the 25%, where it becomes the maximum.
Range: 0Â° (0%) to 360Â° (100.0%), default 270Â° (75.0%)

Interval

Interval defines the size of the interval from the oscillator shape in addition to **Phase**. As a result, phase defines where you start on the shape and interval specifies size of the window on the shape. Default value is 50% as for example sine grows from minimum to maximum in 50% of the period.
Range: 0Â° (0%) to 720Â° (200.0%), default 180Â° (50.0%)

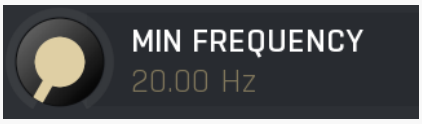
Pitch mode



Pitch mode makes the modulator detect the input pitch.

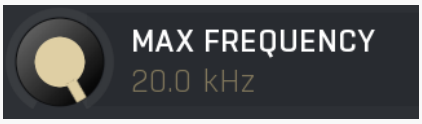
LFO modulation

LFO modulation defines the amount of LFO modulation applied in addition to the pitch detector. With 0% the modulator uses only the pitch detector; with 100% the modulator does the same job as if the modulator were in **Normal** mode. To set the LFO parameters switch to normal mode temporarily.
Range: 0.00% to 100.0%, default 0.00%



Min frequency

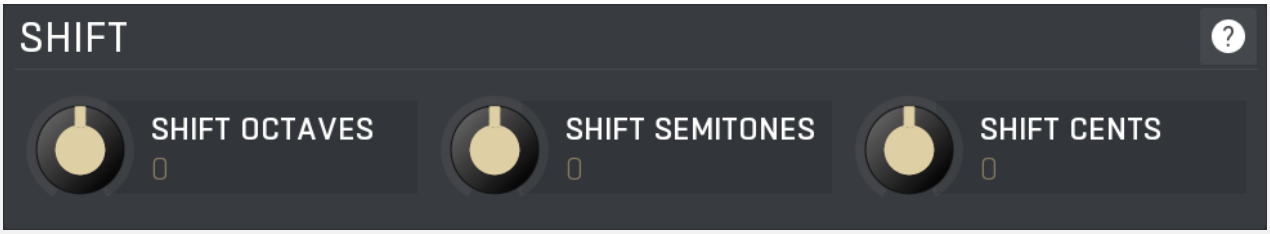
Min frequency defines the frequency, which will cause the modulated parameters to have their minimum values. This basically manipulates the range of the parameters, but it is based on the frequency rather than on parameter values.
Range: 20.00 Hz to 20.0 kHz, default 20.00 Hz



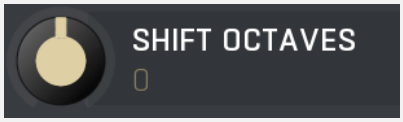
Max frequency

Max frequency defines the frequency, which will cause the modulated parameters to have their maximum values. This basically manipulates the range of the parameters, but it is based on the frequency rather than on parameter values.
Range: 20.00 Hz to 20.0 kHz, default 20.0 kHz

Shift panel

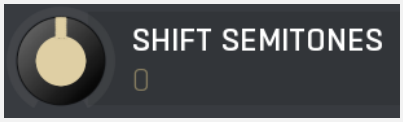


Shift panel lets you shift the detected frequency by specified different amounts. All of its parameters do basically the same thing, but in different units.



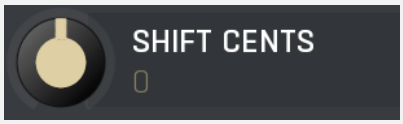
Octaves

Octaves shifts the detected pitch by the specified number of octaves.
Range: -10 to +10, default 0



Semitones

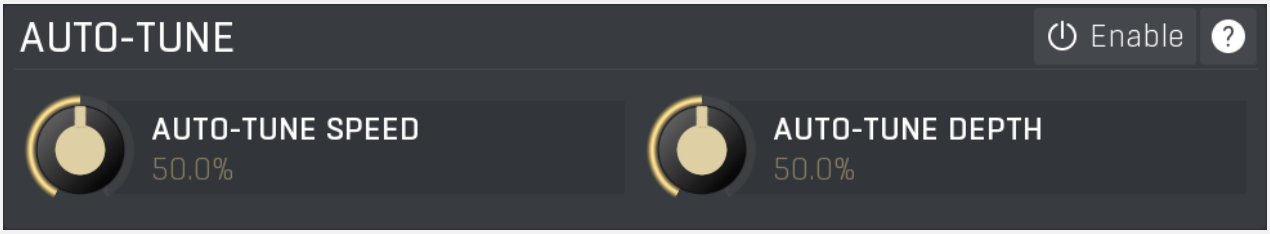
Semitones shifts the detected pitch by the specified number of semitones.
Range: -24 to +24, default 0



Cents

Cents shifts the detected pitch by the specified number of cents of a semitone. The actual pitch change is the sum of these 3 control values.
Range: -100.0 to +100.0, default 0

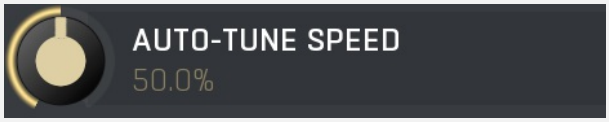
Auto-tune panel



Auto-tune panel contains the automatic tuner parameters. When the pitch detector computes the pitch of the input signal, it can further adjust this value in the same way as an automatic tuner plugin, such as MAutoPitch, works. Note that there are no modifications to the input signal, only the pitch is detected differently.

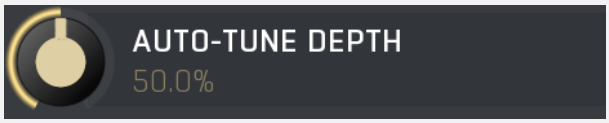
Enable

Enable button enables or disables the auto-tuner.



Speed

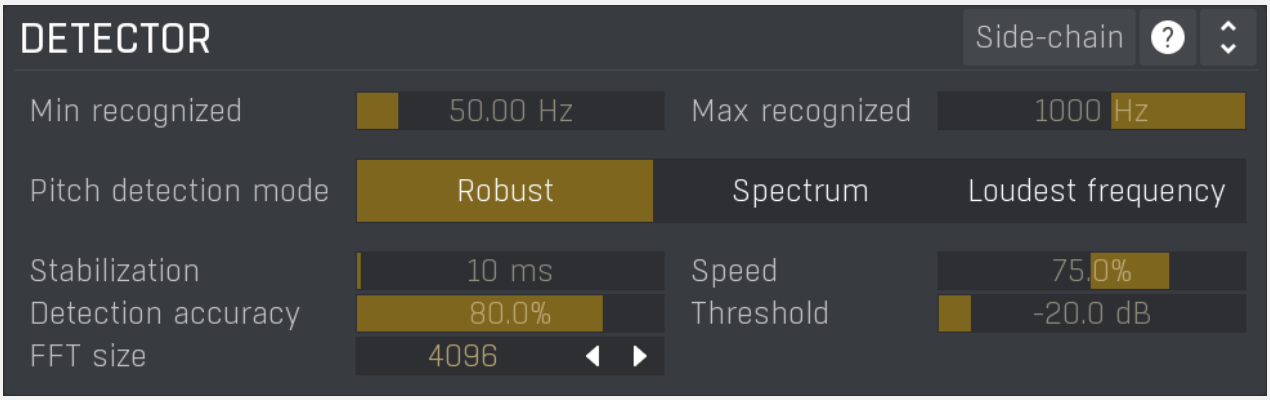
Speed defines how quickly the plugin adjusts, when a note has been changed. Higher speed makes the results immediately in tune, but can cause less natural results.
Range: 0.00% to 100.0%, default 50.0%



Depth

Depth defines how accurate the output should be. With 100% depth the output of the detector shall be exactly in tune. With a lower depth the plugin tolerates more deviation.
Range: 0.00% to 100.0%, default 50.0%

Detector panel



Detector panel contains parameters affecting the pitch detection. You can use them to make the detector work well with your audio material.

Side-chain input

Side-chain input makes the modulator analyze the side-chain input instead of the regular input.

Min frequency

Min frequency defines the minimum recognizable frequency. Any frequency below this value will be considered an error and ignored. For example the fundamental frequency of female vocals rarely goes below 100 Hz, so it may be useful to set this value to this limit to ensure that the detector won't pick up hum or vocal tract noises.

Range: 20.00 Hz to 20.0 kHz, default 50.00 Hz

Max frequency

Max frequency defines the maximum recognizable frequency. Any frequency above this value will be considered an error and ignored. For example the fundamental frequency of any vocal rarely goes above 1000 Hz, so it may be useful to set this value to this limit to ensure the detector won't pick harmonics as fundamental.

The pitch detector uses a smart search for fundamentals and avoids harmonics. However if you set this value higher than 2200Hz, it's technically impossible to avoid picking harmonics, so this smart search is disabled. This may be useful for scientific audio analysis, but is not desired for common audio processing.

Range: 20.00 Hz to 20.0 kHz, default 1000 Hz

Pitch detection mode

Pitch detection mode controls the way the pitch is detected. By default the **Robust** algorithm is used, which takes into account both spectrum and time properties of the audio signal. In most signals the fundamental frequency is related to the loudest harmonics as well, so normally the engine analyses these relations to find the most probable fundamental. However in some instruments such as bells, there may be lots of inharmonic content available and not so many harmonics, which may confuse the engine. In that case try some of the other modes to see which works the best.

Stabilization

Stabilization specifies how quickly can the pitch make bigger changes. This can be useful for more complicated material, such as voice, which often contains short pieces of inharmonic material, which would normally make the detector jump too quickly.

Range: 0 ms to 1000 ms, default 10 ms

Speed

Speed specifies how quickly the pitch can change. By lowering this value the pitch won't be able to change so quickly, which can improve audio quality when modulating parameters, which are not handling abrupt changes well. It can also be used creatively.

Range: 0.00% to 100.0%, default 75.0%

Accuracy

Accuracy defines how quickly and accurately the detector will work at the expense of higher CPU usage.

Range: 0.00% to 100.0%, default 80.0%

Threshold

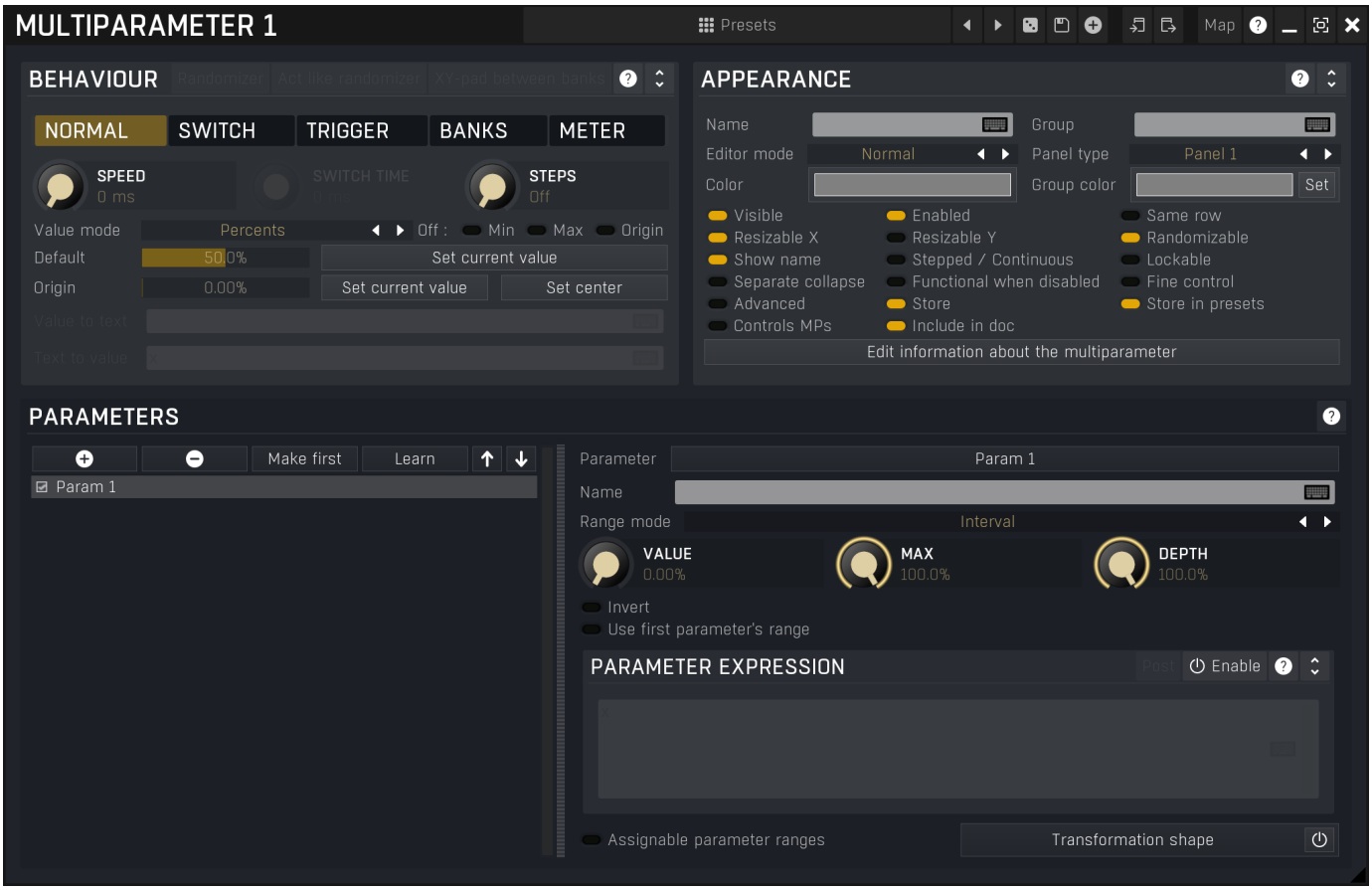
Threshold controls the minimum input level for the pitch to change. This is provided to minimize artifacts caused by the

beginning and ending sections of vocals for example, where the pitch usually fluctuates a lot. It also makes the detector ignore noise in-between actual performances.
Range: silence to 0.00 dB, default -20.0 dB

FFT size

FFT size defines the resolution of the pitch detector (for 44/48kHz sampling rates, automatically converted when needed). The higher it is, the more accurate the pitch will be, but its response will also be slower.
Range: 256 to 16384, default 4096

MultiParameter editor



Multiparameter is a powerful structure, which can speed up your workflow significantly and even perform automatic tasks, often useful when performing in real-time for example. Essentially a multiparameter is a controller which controls other parameters, in fact, an unlimited number of them. Each parameter has limits and potentially a transformation curve for more advanced processing. By manually moving the multiparameter (or automating/modulating it) you can control all of the associated parameters at once.

*This is just the beginning, but it is worth demonstrating how it could be used. We will show it on a vibrato effect. MVibratoMB (and partly MVibrato) is very good at simulating rotary speakers. A rotary speaker traditionally contains a speed switch, or in our case we will think of it as a speed knob - a control that alters the spin speed of the rotary. This would normally be the **Rate** parameter of the vibrato. However, when the rate is increased, the vibrato starts changing the pitch too much, sounding a little too "honky-tonk". We can compensate for this by lowering the **Depth** parameter. As it is not very convenient to control 2 parameters at once, we use a multiparameter to control both parameters with appropriate ranges (ascending for the **Rate** and descending for the **Depth**).*

Besides this basic usage, multiparameters can also work as triggers and switches. Set a multiparameter's mode to **Trigger** or **Switch** and it stops being a slider and becomes a button. When you click the button, the multiparameter starts moving on its own - over the dialled-in switch time it will increase its value (and also the values of any associated parameters) to a maximum and, in the case of trigger mode, then decrease it back to a minimum. In switch mode clicking the button again, the multiparameter decreases back to the minimum value. To make the multiparameter into a simple switch, we can set the switch time to minimum, but in this case we want to extend the functionality in our rotary example.

As mentioned, rotary speakers often have a speed switch. Once switched on, the speed starts increasing until it reaches the "fast" setting, and when switched off, the speed starts decreasing to the original "slow" rate. All we need to do to replicate this functionality is to set the multiparameter's mode to 'switch'.

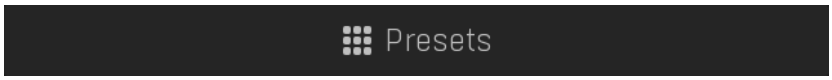
A real rotary actually has 2 speakers, one for low frequencies and the other for the higher ones. As you might expect, these do not have the same spin rate nor do they speed up or slow down equally either. Here is where we can start showing the true potential of multiparameters.

To simulate this, we have to use two bands of MVibratoMB, the first one will simulate the lower reproductor, and the second will be the higher. We use the first multiparameter to control the first band's rate in the same way as described in the example above. Similarly, we use the second multiparameter to control the second band's rate. Now we have 2 switches and can make each band speed-up or slow-down separately, but we want just one switch for both bands. To do this, we use a third multiparameter to control the first and second multiparameters, in switch mode again but with a 0ms switch time. Pressing the button of the 3rd

multiparameter instantly activates the other 2 multiparameters, they both start speeding-up, over a different time period as we requested. Pressing the button again, releases it which also instantly releases the first 2 multiparameters and they start slowing down. Just like the real thing.

Now that we have shown you what is possible with multiparameters, it is worth mentioning that they are used extensively for building devices on the easy screens of most Melda plugins. Every multiparameter given a name in the **Information** panel will be shown on the Easy screen (if the plugin has one). Check our online video tutorials to get more information about **multiparameters and building devices**.

It is also worth mentioning that you can access the multiparameter settings directly from easy screen by holding Ctrl+Alt and clicking on the target control. It may simplify building devices. Note that this may not work for some editor modes such as meters or bar graphs.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

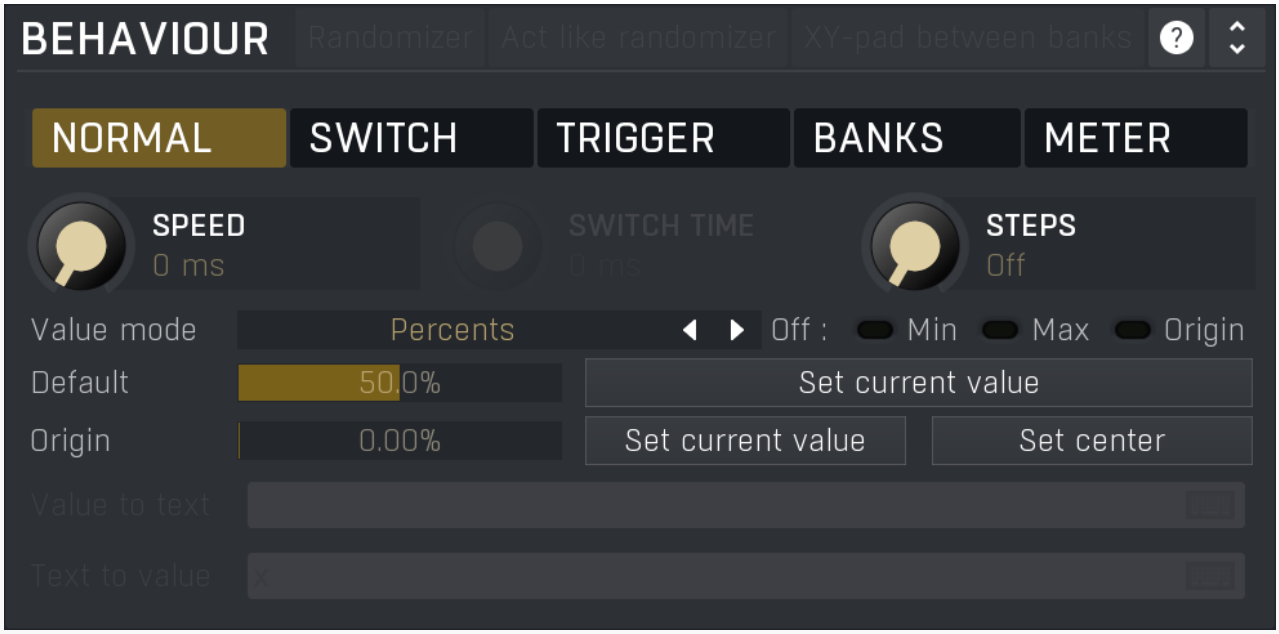
Paste

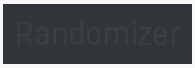
Paste button loads the settings from the system clipboard.

Map

Map button displays all current mappings of modulators, multiparameters and MIDI (whichever subsystems the plugin provides).

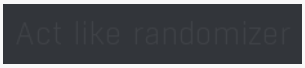
Behaviour





Randomizer

Randomizer switch is available only for **Trigger** mode and it makes the multiparameter produce random values for each associated parameters. This is useful to implement some sort of randomization feature, which covers a set of parameters. You usually want to set the **Switch time** to 0, so that the randomization is instant, but longer values may be useful for some creative effects.



Act like randomizer

Act like randomizer switch is similar to **Randomizer** switch in the way it works, but it does not generate random values itself. It only makes sure it is triggered instantly and stores proper undo point.

XY-pad between banks

XY-pad between banks switch is available only for Banks mode and it lets you create XY pads, that would interpolate between 3 or more banks that you specify. With 4 banks the engine creates a classic XY pad, where the 1st bank belongs to the left top corner, 2nd to the right top, 3rd to left bottom and 4th to the right bottom. With more banks the engine creates a circular pad with vertices associated to individual banks.
Please note that in order for this to work, the multiparameter actual needs 2 multiparameters (X and Y values), hence must NOT be the last one and it occupies the next multiparameter as well. It is recommended to name the next multiparameter and associate it to some parameters, ideally the same ones, just to make sure the engine won't remove it. But in fact only the first multiparameter will actually be working.

Mode

Mode controls the behaviour of the multiparameter.

Normal mode makes the multiparameter work like any other control.

Switch mode hides the slider and shows a button instead. The button has 2 states. By pushing the button, the multiparameter value starts rising from 0% to 100% over a specified time interval. By pushing it again the value starts falling back to 0%. You could do the same thing having the multiparameter in normal mode and moving the slider from left to right and then back, but mode this performs that automatically and maintains a constant time period.

Trigger mode is similar to switch mode, but the button has only a single state and when you push it, the value automatically goes from 0% to 100% and then back without any need to push the button again.

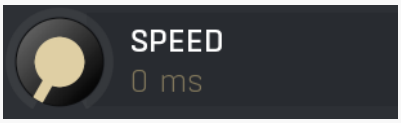
Banks mode is very different. A multiparameter in banks mode keeps several states (called banks) for all of the parameters, much like A-H presets, but only with a limited set of parameters. The multiparameter then morphs between the banks or can be set to switch directly between them (no interpolated values). This is a marvellous way to control many parameters with complex settings by using a single multiparameter.

Let's explain the banks mode in more detail. Say you switch a multiparameter to banks mode, learn a few parameters and set the number of banks to 4. Then bank 1 contains a value for all of the parameters. Similarly bank 2 contains a different value for each of them. And so on. If you set the multiparameter slider to 0%, the associated parameters will be set to values in bank 1. If you set the slider to 100%, bank 4 will be used. If you set the slider to 33.3%, bank 2 will be used. And what if you select 50%? Then it will be halfway between bank 2 and bank 3.

You can have many banks, you can edit each of them, generate random settings etc. So let's say you want to create some

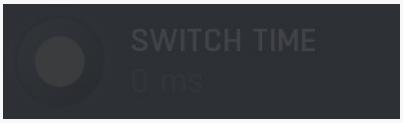
complex movement. You use a multiparameter in banks mode, select a reasonable number of banks. You can edit each of them, but it is easier to use the randomization button to generate random settings for each of them. Then every time you move the multiparameter, all of the associated parameters will move, somewhere between the banks. You can then use a modulator or automation to slowly adjust the multiparameter.

Meter mode makes the multiparameter work as a meter. Instead of controlling other parameters it starts following the value of them. You can then use that to implement a simple meter on the easy screen (if the plugin has one).



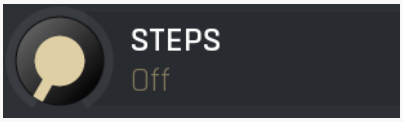
Speed

Speed controls the interpolation time. When it is zero and you change the multiparameter value, the associated parameters are adjusted immediately. If this is non-zero however, the actual parameters won't change immediately but will interpolate over time. The speed value is actually the time needed to go from minimum to maximum or vice versa. So if this is 1 second and the current value is say 0% and you click 100%, it will take 1 second for the multiparameter to get there.
This feature is provided mainly because changing some parameter via MIDI or mouse may cause unnecessary zipper noise or inaccuracies due to low MIDI precision. Using the interpolation you can somewhat slow everything down, so that the artifacts become negligible. It can also be used creatively. The default value has been experimentally tested to avoid all artifacts for most parameters.



Switch time

Switch time defines the time needed to switch from the minimum value to the maximum one, or conversely. It is used only in **switch** and **trigger** modes.



Steps

Steps lets you create an arbitrary number of equi-distant steps for the multiparameter values. While this technically limits the possibilities of the multiparameter by limiting the number of accessible values, it is sometimes easier to choose from a predefined number of options than from the full range. If you want to use different ranges between the steps, use the Banks mode with Interpolate values disabled.

Value mode

Value mode defines the units displayed on the multiparameter.
Percents mode lets the multiparameter display percentages between 0% to 100%.
Percents (-100% to 100%) displays percentages between -100% to 100%.
By first parameter mode uses the current value of the first parameter that is controlled by the multiparameter. For example, if you want to control a plugin gain, but also in addition to the changed gain control other parameters, you may still want to call the multiparameter \"gain\" and the units should be decibels as usual, not percentages which do not make much sense for such a multiparameter.
By bank name displays the name of the nearest bank. In some controls, such as switchers, it is possible to display the set of the values as a menu. The menu is created automatically and it even creates groups for better clarity, based on the prefix of the bank names. You can use ' # ' (hashtag surrounded by spaces) to define the groups manually like this: "group # name". For example, you can name one bank "Main group # First bank" and another "Main group # Second bank", and these will be displayed in a single group in the menu. If you are going to use this method, make sure the ' # ' sequence is present in each bank's name.
By bank name interpolated considers name of all banks numbers. It then interpolates between them and displays the result as a number.
By bank name interpolated log is similar, but interpolates the values in logarithmic domain.considers name of all banks numbers. It's useful for units, which are naturally logarithmics, such as frequency.
By bank number shows the index of the nearest bank.
Expression lets you formulate the value -> text and back using mathematical expressions and can be used for some more complex MPs or if you need some custom units.

Default

Default controls the default value of the multiparameter. You can edit it directly or just set the MP into its reasonable default and click the **Set current value**.Most GUI components created for the multiparameter respond to right-click by setting the default value in the same way that other parameters do. It is essential for user experience when building your own devices.

Set current value

Set current value stores the current value as the default one for the multiparameter.

Origin

Origin informs the GUI engine of the origin of the value. For instance, a default value for panorama is in the center and it is logical that visual elements controlling panorama should somehow highlight the center position. If, for example, you are using

a value button to edit the panorama, by default it displays the current value using a bar starting from the left side (being the origin defined as minimum) towards the actual value, but here it is better to display the bar from the center towards the current value, whether it is on the left or right of the center. Therefore the center should be the origin.

Set current value

Set current value stores the current value as the origin for the multiparameter.

Set center

Set center sets the center (50%) as the origin for the multiparameter. This is often the case for parameters such as gain and panorama, so it deserves a dedicated button. It is supported only for knobs.

Value to text

Value to text is available only with Expression **Value mode** and lets you enter a text featuring mathematical expressions to produce the units. This can be used for more complex multiparameters. The text itself is just another text, but it can contain substrings of this structure:

{expression}
or
{expression;decimals}
or
{expression;text convertor}

where expression is the actual mathematical expression, decimals is the number of decimal places in the resulting value. Text convertor provides several predefined "value to text" modes:

volume : converts linear volume (multiplier) to volume in dB, e.g. 0.00 -> **silence**, 0.50 -> **-6.02 dB**...
panorama : converts -1..+1 to panorama, e.g. -0.74 -> **74% left**, 1.00 -> **100% right**...
percents : e.g. 0.54 -> **54.0%**, -1.30 -> **-130.0%**...
time : e.g. 0.00 -> **0 ms**, 0.0002 -> **0.20 ms**, 1.20 -> **1200 ms**...
phase : phase in degrees, e.g. 0.30 -> **108ÃƒÂ,Ã,Â° (30.0%)**, 1.00 -> **360ÃƒÂ,Ã,Â° (100.0%)**, -2.00 -> **- (-200.0%)**...
frequency : e.g. 66.00 -> **66.00 Hz**, 2345.00 -> **2345 Hz**, 12100.00 -> **12.1 kHz**...
note : converts frequency into nearest MIDI note name, e.g. 66.00 -> **C2**, 440.00 -> **A4**...
cents : converts frequency deviation from nearest MIDI note, e.g. 66.00 -> **+16 cents**, 440.00 -> **0 cents**...
noteandcents : converts frequency into nearest MIDI note name and the deviation, e.g. 66.00 -> **C2 (+16 cents)**, 440.00 -> **A4 (0 cents)**...
ranges(x_0,x_1,...x_n; "{e1;d1}|1", "{e2;d2}|2",...,"{en;dn}|n") : finds the interval [x_(i-1),x_i) containing the value of the expression, substitutes this value as "x" into "{ei;di}|i}" and renders the result...

Here is the list of variables available for each expression:

x = the multiparameter value in 0..1
sr = current sampling rate
mp{i} = value of a multiparameter, e.g. mp1 for the first multiparameter.
p{i} = value of a parameter attached to this multiparameter, e.g. p1 for the first parameter. Note that this includes disabled parameters, which provides a handy way to use any value from the plugins, even if it is not supposed to be controlled by this MP - map it and disable it, then use the p{i} variable to use in the expression.

Functions and features of the expressions are available below. Now let's see a few examples:
Test: {x} - 0.1234 -> **Test: 0.12**
*{x} ({x*100;0}%)* - 0.1234 -> **0.12 (12%)**
{x} ({x;percents}) - 0.1234 -> **0.12 (12%)**
{ffrom01(x);frequency} - 0.1234 -> **46.91 Hz**, 0.00 -> **20.0 Hz**, 1.00 -> **20 kHz**
{ffrom01(x);frequency} - {ffrom01(x);noteandcents} - 0.1234 -> **46.91 Hz - F#1 (+24 cents)**
{todb(sqr(x));4} dB - 0.00 -> **-200.0000 dB**, 0.50 -> **-12.0412 dB**. "sqr" is used to mimic the transformation used in pretty much every Volume parameter in MeldaProduction plugins.
{sqr(x);volume} - 0.00 -> **silence**, 0.50 -> **-12.0412 dB**
*{2*x - 1;panorama}* - 0.00 -> **100% left**, 0.50 -> **center**, 1.00 -> **100% right**
{100(2*x - 1);ranges(-100,-0.1,0.1,100;"{abs(x);0} L", "C", "{x;0} R"}* - 0.00 -> **100% L**, 0.50 -> **C**, 1.00 -> **100% R**

Expression evaluator uses traditional C/C++ style formating, which is natural for most people. It provides arithmetics, logical and conditional operators. Following terms are supported:
Constants: **pi**, **e**, **sqrt2**, **ln2**

Arithmetic operators:
-a inverts the sign, e.g. "-x" produces +2 for x=-2
a+b = addition
a-b = subtraction
a*b = multiplication

a/b = division
a%b = modulo, remainder after division
a^b = power, e.g. "2^3" produces 2*2*2 = 8

Arithmetic functions:
min(a,b) = minimum of both values
max(a,b) = maximum of both values
limit(a,min,max) = a limited into the interval min..max
to01(a,min,max) = converts "a" as min..max to 0..1
from01(a,min,max) = converts "a" as 0..1 to min..max
tom11(a,min,max) = converts "a" as min..max to -1..1
fromm11(a,min,max) = converts "a" as -1..1 to min..max

Basic mathematic functions: **abs(x)** = absolute value, e.g. abs(-3) = 3 **sqr(x)** = x*x **sqrt(x)** = square root **exp(x)** = natural exponential e^x **ln(x)** = natural logarithm **log10(x)** = logarithm with base 10 **log(x, base)** = logarithm with specified base **inv(x)** = 1/x **sgn(x)** = sign of x, -1 or 0 or +1 depending on x*x **round(x)** = rounding to the nearest value **floor(x)** = rounding to the nearest lower value, e.g. floor(-2.3) = -3 **ceil(x)** = rounding to the nearest higher value, e.g. ceil(-2.3) = -2 **rand(x)** = random value from 0 to x

Functions for specific units:
f01(a) = converts "a" as frequency from 20..20000 into log scale 0..1
ffrom01(a) = converts "a" as 0..1 (log scale) to frequency from 20..20000
ffrom01limit(a) = same, but makes sure the output is in 20..20000 range
f01range(a,min,max) = converts "a" as frequency from min to max into log scale 0..1
ffrom01range(a,min,max) = converts "a" as 0..1 (log scale) to frequency from min to max
ffrom01rangelimit(a,min,max) = same, but makes sure the output is in min..max range
todb(a) = converts "a" as multiplier to dB value by calculating "20*log10(a)"
fromdb(a) = converts "a" as dB value to multiplier by calculating "10^(a/20)"

Trigonometric functions: **sin(x)**, **asin(x)**, **cos(x)**, **acos(x)**, **tan(x)**, **atan(x)**, **sinh(x)**, **cosh(x)**, **tanh(x)**

Logical operators:
a==b = comparison producing 1 if "a" and "b" are equal, 0 otherwise
a!=b = comparison producing 1 if "a" and "b" are NOT equal, 0 otherwise
a<b = comparison producing 1 if "a" is lower than "b", 0 otherwise
a<=b = comparison producing 1 if "a" is lower or equal to "b", 0 otherwise
a>b = comparison producing 1 if "a" is greater than "b", 0 otherwise
a>=b = comparison producing 1 if "a" is greater or equal to "b", 0 otherwise
!a = logical negation, 0 produces 1, 0 otherwise
a&&b = logical AND, produces 1 if both "a" and "b" are nonzero
a||b = logical OR, produces 1 if any of "a" and "b" are nonzero
a^^b = logical XOR, produces 1 if "a" and "b" are logically different
a ? b : c = if a is nonzero, then the result is b, otherwise it is c}}}

Text to value

Text to value is available only with Expression **Value mode** and lets you convert a number user enters as a text input into the multiparameter value in 0..1. This can be used for more complex multiparameters. Unlike **Value to text** here you need to enter a single expression, no need for any other text. If the resulting value exceeds the 0..1 interval, it is automatically limited. Here is the list of variables available for the expression:

x = the value the user entered
sr = current sampling rate
mp{i} = value of a multiparameter, e.g. mp1 for the first multiparameter.
p{i} = value of a parameter attached to this multiparameter, e.g. p1 for the first parameter. Note that this includes disabled parameters, which provides a handy way to use any value from the plugins, even if it is not supposed to be controlled by this MP - map it and disable it, then use the p{i} variable to use in the expression.

Functions and features of the expressions are available below. Now let's see a few examples:
x - sets MP to 0.1234 if the user entered "0.1234"
x/100 - sets MP to 0.1234 if the user entered "12.34" (or "12.34%" for example)
f01(x) - sets MP to the proper frequency in log scale, e.g. 20 is translated to 0, 20000 to 1
sqrt(fromdb(x)) - sets MP to 0.5 if the user entered "-12.0412" (or "-12.0412 dB" for example). "sqrt" is used to mimic the transformation used in pretty much every Volume parameter in MeldaProduction plugins.

Expression evaluator uses traditional C/C++ style formating, which is natural for most people. It provides arithmetics, logical and conditional operators. Following terms are supported:
Constants: **pi**, **e**, **sqrt2**, **ln2**

Arithmetic operators:
-a inverts the sign, e.g. "-x" produces +2 for x=-2
a+b = addition
a-b = subtraction
a*b = multiplication
a/b = division
a%b = modulo, remainder after division
a^b = power, e.g. "2^3" produces 2*2*2 = 8

Arithmetic functions:
min(a,b) = minimum of both values
max(a,b) = maximum of both values
limit(a,min,max) = a limited into the interval min..max
to01(a,min,max) = converts "a" as min..max to 0..1
from01(a,min,max) = converts "a" as 0..1 to min..max
tom11(a,min,max) = converts "a" as min..max to -1..1
fromm11(a,min,max) = converts "a" as -1..1 to min..max

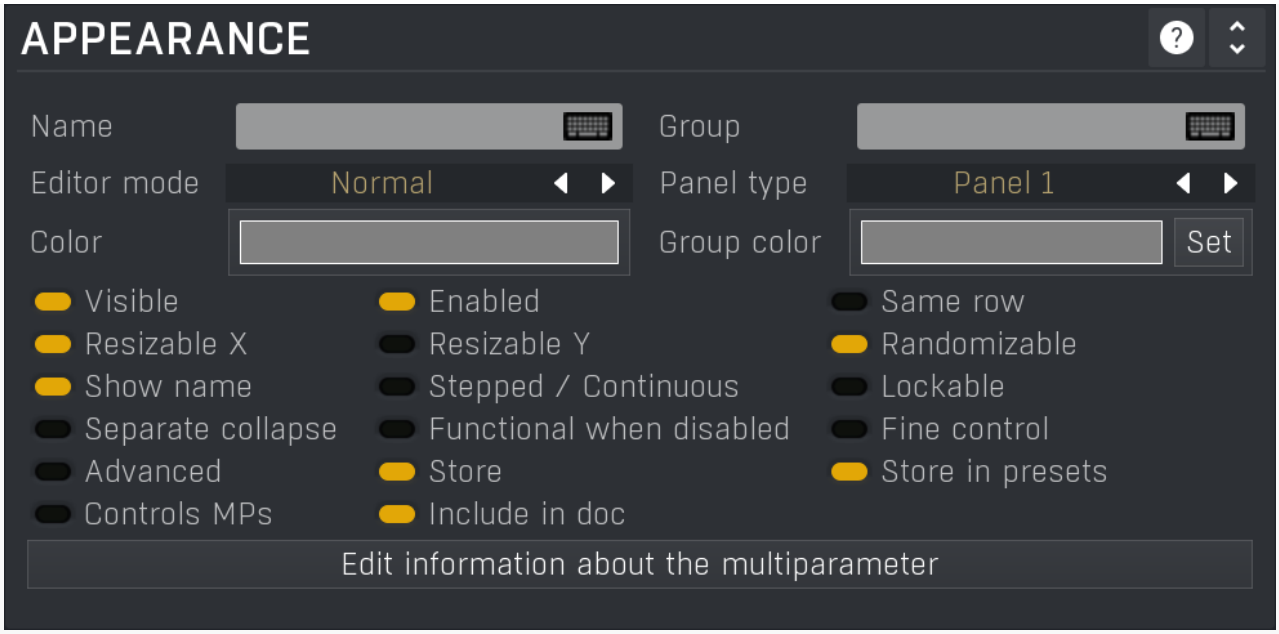
Basic mathematic functions: **abs(x)** = absolute value, e.g. abs(-3) = 3 **sqr(x)** = x*x **sqrt(x)** = square root **exp(x)** = natural exponential e^x **ln(x)** = natural logarithm **log10(x)** = logarithm with base 10 **log(x, base)** = logarithm with specified base **inv(x)** = 1/x **sgn(x)** = sign of x, -1 or 0 or +1 depending on x*x **round(x)** = rounding to the nearest value **floor(x)** = rounding to the nearest lower value, e.g. floor(-2.3) = -3 **ceil(x)** = rounding to the nearest higher value, e.g. ceil(-2.3) = -2 **rand(x)** = random value from 0 to x

Functions for specific units:
f01(a) = converts "a" as frequency from 20..20000 into log scale 0..1
ffrom01(a) = converts "a" as 0..1 (log scale) to frequency from 20..20000
ffrom01limit(a) = same, but makes sure the output is in 20..20000 range
f01range(a,min,max) = converts "a" as frequency from min to max into log scale 0..1
ffrom01range(a,min,max) = converts "a" as 0..1 (log scale) to frequency from min to max
ffrom01rangelimit(a,min,max) = same, but makes sure the output is in min..max range
todb(a) = converts "a" as multiplier to dB value by calculating "20*log10(a)"
fromdb(a) = converts "a" as dB value to multiplier by calculating "10^(a/20)"

Trigonometric functions: **sin(x)**, **asin(x)**, **cos(x)**, **acos(x)**, **tan(x)**, **atan(x)**, **sinh(x)**, **cosh(x)**, **tanh(x)**

Logical operators:
a==b = comparison producing 1 if "a" and "b" are equal, 0 otherwise
a!=b = comparison producing 1 if "a" and "b" are NOT equal, 0 otherwise
a<b = comparison producing 1 if "a" is lower than "b", 0 otherwise
a<=b = comparison producing 1 if "a" is lower or equal to "b", 0 otherwise
a>b = comparison producing 1 if "a" is greater than "b", 0 otherwise
a>=b = comparison producing 1 if "a" is greater or equal to "b", 0 otherwise
!a = logical negation, 0 produces 1, 0 otherwise
a&&b = logical AND, produces 1 if both "a" and "b" are nonzero
a||b = logical OR, produces 1 if any of "a" and "b" are nonzero
a^^b = logical XOR, produces 1 if "a" and "b" are logically different
a ? b : c = if a is nonzero, then the result is b, otherwise it is c}}}

Appearance



Name

Name specifies the name of the multiparameter, which is shown on the multiparameter button. The name is also used for devices - the multiparameter serves as a parameter for the device (on the Easy screen). If no name is specified or if the first character is an *, then the parameter is hidden. This is useful if you need some internal multiparameters which you don't want to show on the Easy screen for some reason.

Group

Group can be used to put some multiparameters into the same group, which results in them being placed in the same panel on the Easy screen (the device editor). Additionally you can actually place the groups into tabs by setting group to "tabname#groupname". The name of the tab needs to be there only for the first parameter of the new group. This makes it possible to build a complex devices with dozens of parameters.

Editor mode

Editor mode controls the way the multiparameter are to be displayed on the Easy screen.
Normal is the default mode and is represented by a small knob or button.
Big mode is similar, but uses a big knob or big button.
Button mode displays a value button, which is usually more compact than knobs.
Check-boxes makes the multiparameter displayed as a set of checkboxes (also called radio buttons). It is relevant only in **Banks** mode.
Check-boxes horiz & below is similar but displays the checkboxes in a single row, hence horizontally. Below mark makes the label underneath the actual checkbox.
Switcher and Selectors are useful for selecting a number of discrete values and similarly to check-boxes these are working only in **Banks** mode.
Title button places the control into the title bar of the panel to which it belongs.
Title enable button places the control into the title bar of the panel to which it belongs and makes it a standard enable button (which also makes all controls within the panel unavailable if it is itself disabled).
XY pad creates a 2 dimensional XY pad control, that edits this multiparameter in the X axis and the next multiparameter in the Y axis. There are multiple versions of this control, all of them differ only by size.
Spacer is a helper mode for device design, which doesn't display anything and only keeps empty space.
Meter creates a simple meter instead. You will probably want to set the multiparameter to Meter mode as well or attach it to a modulator. Meters don't really control anything and their purpose is purely to get a visual feedback. The meters can be horizontal or vertical and they can be up or down. Up is the usual choice useful for peak meters for example. Down is useful for gain reduction meters.
Bars start/end mode creates an editor, similar to step sequencer editor, where each parameter has its own bar. The **Bars start** starts the editor and all multiparameters are then added to it until a multiparameter with **Bars end** mode is found or until there are no remaining multiparameters. Note that this kind of editor doesn't show units and may have several other limitations.
Order is a very specific editor for Order modules available in modular systems such as MXXX. It lets you provide an processing order editor on the easy screen. To use it, attach the MP to Order parameter of the Order module and edit the MP information field, so that it contains all the items to be ordered, separated by ';'. The number of items must match the number of items in the Order module, otherwise the order won't work properly. You can also include colors for each item separated by # (hexa or one of the predefined set: Dynamics, Distortion, Modulation, Stereo, Spectral, Synthesis, Instrument, MDrummer, Reverb, Delay, EQ, Filter, Saturation, Limit, Time, Pitch, FX) and enable MP indices.
Example of the info: Compressor#Dynamics;EQ#EQ;Limiter#007F7F;Something

Panel type

Panel type defines the type of panel in which multiple controls of the same group are placed. These differ only in their graphics

display.

Color

Color defines colorization for the element on the Easy screen (if the plugin has one). The feature is disabled if the Alpha value of the color is 0. Using this feature often increases memory consumption of the plugin, so make sure you use it only if necessary and try to use as low a number of different colors as possible. It is recommended to use only the snapshot colors to make sure the same colors are used in most cases, reducing the memory consumption. It is also highly recommended to use colors with a value (lightness) of 128 (the middle value), which makes sure that the lightness of the elements won't be changed. This works best for most styles. Please note that the style may be configured to simply ignore this color, so there may be no change at all. If you use this feature, make sure that you test it with all styles.

For the sake of workflow the colors have predefined meanings. It's highly recommended to follow this standard:
Orange - dynamics
Green - equalization, filtering
Brown/yellow - reverb, delay
Blue - modulation
Red - limiting, saturation, distortion
Cyan/yellow - stereo
Purple/pink - time, pitch, unison...
Grey - utilities, tools

Group color

Group color defines colorization for the group panel on the Easy screen (if the plugin has one) and is ignored for all multiparameters except for the first one in a group. The feature is disabled if the Alpha value of the color is 0. Using this feature often increases memory consumption of the plugin, so make sure you use it only if necessary and try to use as low number of different colors as possible. It is recommended to use only the snapshot colors to make sure the same colors are used in most cases, reducing the memory consumption. It is also highly recommended to use colors with a value (lightness) of 128 (the middle value), which makes sure that the lightness of the elements won't be changed. This works best for most styles. Please note that the style may be configured to simply ignore this color, so there may be no change at all. If you use this feature, make sure you test it with all styles.

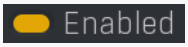
For the sake of workflow the colors have predefined meanings. It's highly recommended to follow this standard:
Orange - dynamics
Green - equalization, filtering
Brown/yellow - reverb, delay
Blue - modulation
Red - limiting, saturation, distortion
Cyan/yellow - stereo
Purple/pink - time, pitch, unison...
Grey - utilities, tools

Set

Set button sets the color and group color for all multiparameters in the same group. It is pretty sensible to do that as all controls should look similar within each group. This can also be done by editing each parameter, but this way is easier.

Visible

Visible checkbox controls if the parameter is visible on the Easy screen (if the plugin has one). Its effect is similar to the '*' prefix in the parameter name, but the multiparameter's name is also available to the plugin host. This is useful when you wish to automate that multiparameter from the host but not show it on the Easy screen. This parameter can also be attached to another multiparameter for example in order to change the GUI somehow.

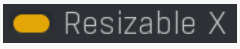


Enabled

Enabled switch enables/disables the multiparameter. If disabled, it is grayed on the easy screen.

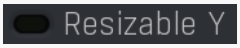
Same row

Same row checkbox defines if the parameter should be displayed next to the previous one on the Easy screen. Otherwise it will be placed on the next row. This setting serves as a hint and the plugin may ignore it, if it is impossible to do.



Resizable X

Resizable X switch lets you specify if the panel could be resized. It is on by default to make sure everything gets resized, however when using multiple panels next to each other, it may be advantageous to disable resizing of some of them to save space. Otherwise each panel's size is proportional to number of controls it contains, which could make some of the panels larger than actually necessary.



Resizable Y

Resizable Y switch lets you specify if the panel could be resized vertically. It is off by default to make sure everything has the

minimum size it requires, but for aesthetic reasons you may want to make all groups on the same row the same size even if the controls inside them are not.

Randomizable Randomizable

Randomizable option defines if the multiparameter can be randomized on the easy screen. You may want to disable this for input/output gain for example.

Show name Show name

Show name option lets you show or hide the name of the multiparameter for some editor modes. The option has no effect for several editor modes.

Stepped / Continuous Stepped / Continuous

Stepped / Continuous option tells the engine that the multiparameter can be in 2 modes, stepped or continuous. If so, it is assumed that you either used **Banks mode** or **Steps** to produce some sort of predefined set of values for the stepped mode. By enabling this option you allow the engine to convert the multiparameter to continous mode by either ignoring the steps or interpolating the bank values. It can be used when designing devices.

Lockable Lockable

Lockable option creates a lock button next to the parameter on the Easy screen, allowing the user to browse through presets without this parameter changing. Please note that this feature is available only for some editor modes.
When the parameter is first locked on the Easy screen it is added to the set of lockable parameters (which are listed in the Global Lock window).

Separate collapse Separate collapse

Separate collapse checkbox makes the panel collapsable separately on the Easy screen. By default it is disabled and that makes the engine find all panels on the same row and collapse all of then or none of them.

Functional when disabled Functional when disabled

Functional when disabled switch makes the multiparameter work even when disabled. This may be useful in some complex scenarios, where you need to make the MP control the target parameters and only use the **Enabled** flag to grey out the controls on the easy screen.

Fine control Fine control

Fine control switch makes the multiparameter editor steps extra small, which is useful, when you need very high precision. This is often handy when using banks mode with many banks interpolating.

Advanced Advanced

Advanced switch makes the multiparameter 'advanced', disabled by default. If such an multiparameter exists, the engine also creates a button to show/hide all advanced multiparameters. In effect this lets you create a simple GUI with optional advanced controls.

Store Store

Store makes the multiparameter stored. Relevant when creating devices. In some rare cases you may want to disable this, e.g. if the value is a meter or somehow it just doesn't need to be save at all.

Store in presets Store in presets

Store in presets makes the multiparameter stored in presets. Relevant when creating devices. In some cases you may want to store the value, but ignore it when browsing presets. For instance, input gain may be ignored if the user is supposed to set the gain to some predefined value and then browse the presets without changing that gain.

Controls MPs Controls MPs

Controls MPs informs the engine that this MP is controlling other multiparameters. As such it shouldn't be 'invoked' unless it is explicitly touched by the user. This is relevant for example when it is used in a device with a device presets - loading a device preset invokes all relevant MPs, so that the state of the device is remembered properly, but invoking this one would actually make things incorrect.

Include in doc Include in doc

Include in doc makes the multiparameter's help included in the documentation / panel help. You may want to disable this if there are multiple 'identical' controls with the exact same help info, for all of them except for one.

Edit information about the multiparameter Edit information about the multiparameter

Edit information about the multiparameter lets you edit the information displayed as help for the multiparameter control on the easy screen. It can use a simplified HTML format, but in practice only following tags are likely to be useful:
<name> = name of the control, the info should begin with that.
<reference> = reference to another control.

<value> = specific value the control can have.
<b> = bold text
<i> = italic text
<cred> = red
<cgreen> = green
<cblue> = blue

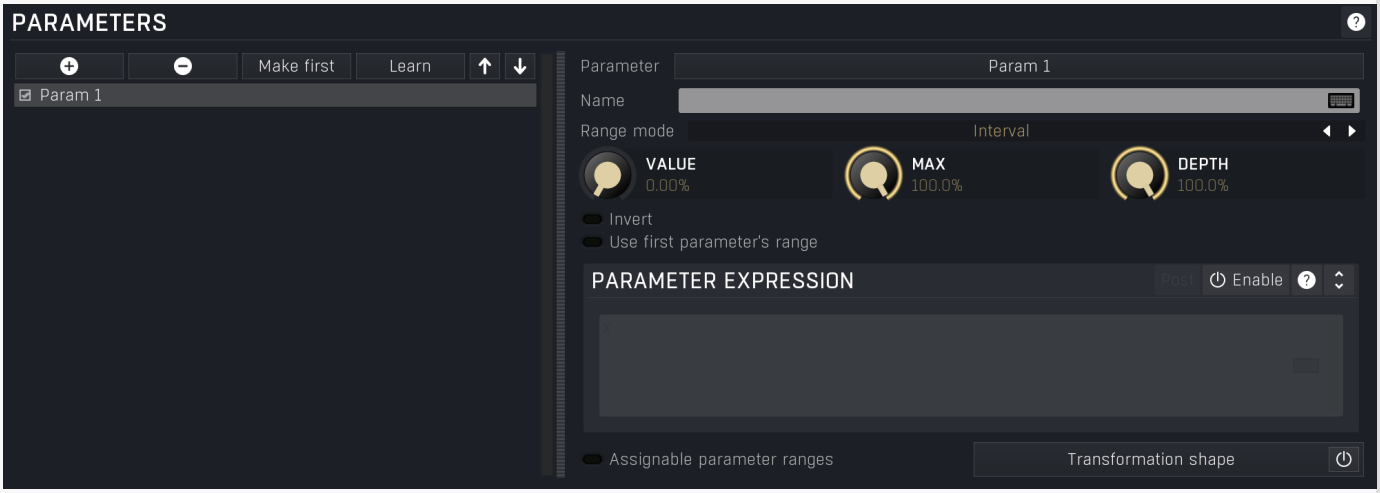
For example: *<name>Gain</name> controls the output level. It can be used in conjunction with <reference>Volume</reference>. When set to <value>silence</value>, it won't do anything and consume no CPU power.*

When **Order** control is used, the first line of the information can contain additional info of ";" separated individual items in the following format:
*name{#hexcolor}{*index of the enable MP}* where {} denotes an optional item.

For example: *Reverb;EQ#7F0000;Flanger#007F00*3*

In this example, the order is probably used to control order of effects. When the order MP value is set to 0, Reverb is the first, with no specific color and no MP to enable/disable it. Flanger is next, the color is red, but no enable either. Finally Flanger is the third, it is green and MP3 is used to enable/disable it.

Parameters panel



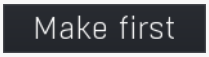
Parameters panel configures how the multiparameter assigns values to the target parameters.

Add

Add button adds a parameter to the list of controlled parameters. Alternatively you can use the learn feature available by right-clicking the multiparameter button.

Delete

Delete button deletes the selected parameter from the list of controlled parameters.



Make first

Make first button moves the selected parameter to the first item in the list. This is useful for sake of the **By first parameter** value mode, which makes the multiparameter show the units of the first parameter in the list. Please note that if you have some other multiparameter, modulator or another subsystem access the ranges of individual parameters, this function will reorder them, so these connections will no longer be correct.

Learn

Learn button starts or stops the learning. Click it, then move some parameters in the plugin, then click it again. Learning can also be accessed from the global multiparameter menu.

Up

Up button moves the selected parameter up one item, if possible. This may be useful when keeping things organized, but please note that if you have some other multiparameter, modulator or another subsystem access the ranges of individual

parameters, this function will reorder them, so these connections will no longer be correct.

Down

Down button moves the selected parameter down one item, if possible. This may be useful when keeping things organized, but please note that if you have some other multiparameter, modulator or another subsystem access the ranges of individual parameters, this function will reorder them, so these connections will no longer be correct.

Parameter

Parameter defines the target parameter which is being modulated. The set contains all automatable parameters.

Name

Name lets you name the parameter somehow and may be helpful in situations, where there are many parameters being edited without obvious meanings.

Assignable parameter ranges

Assignable parameter ranges allows you to assign parameter ranges of several first parameters to other subsystems such as multiparameters or modulators. By default it is disabled, which removes all the relevant parameters to save valuable resources. This feature is available only if automation compatibility mode for V10 is disabled.

Transformation shape

Transformation shape button displays the graph editor, which lets you tweak the shape of the curve used to control the selected parameter. The X axis shows the original values, the Y axis defines the results. Please note that this takes some CPU, therefore you have to enable it using the enable button in the title bar.

Range mode

Range mode defines how the parameter range is selected. While sometimes it is better to specify minimum and maximum, other times it is better to use a nominal center and depth (% of full scale). This control allows you to define which one it will be.

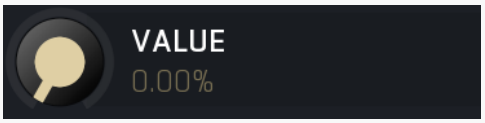
Up and down mode makes the values go above and below the selected **Value**, which is considered the center. The interval is made smaller if necessary.

Full range mode is similar, except the range is symmetrically constrained, so the selected **Value** may not be the center anymore.

Up/down only modes goes from the selected value up/down only.

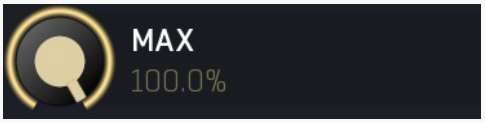
Let's compare these 4 modes. Taking a value of -12dB value, with a depth of 75% and a scale of +/- 24dB. The nominal range is therefore = +/-24 dB * 75% = 36dB. With values of 0%, 50% and 100% the outputs are:
Up and down: -24, -12, 0 (range constrained to 12 dB either side)
Full range: -24, -6, 12 (range limited to minimum, but not constrained)
Up only: -12, 6, 24 (range not constrained = +/-24 dB * 75% = 36dB)
Down only: -12, -18, -24 (range limited to minimum)

Interval mode is the most simple one and goes from **Value** to **Maximal value**.



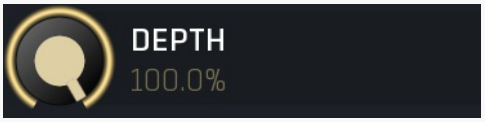
Value

Value defines the center of the target parameter's range or the minimum if the **Range mode** is set to **Interval**.



Maximal value

Maximal value defines the upper limit of the target parameter's range. It is available only if the **Range mode** is set to **Interval**. This value can be lower than **Value**. 0% is always mapped to reference>Value and 100% to reference>Maximal value.



Depth

Depth defines size of the target parameter's range. It is used only if the **Range mode** is not set to **Interval**.

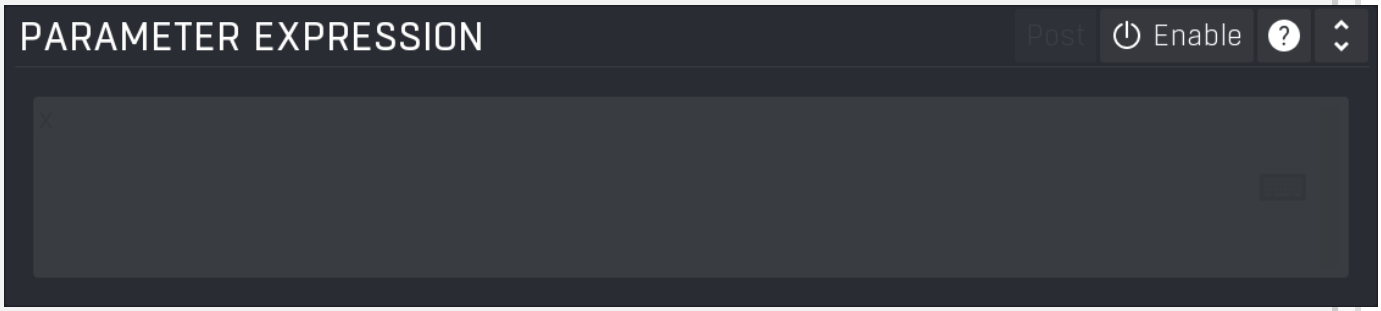
Invert

Invert checkbox inverts the target parameter's range, so that minimum becomes maximum and vice versa.

Use first parameter's range

Use first parameter's range makes the parameter display use the same range as the first parameter in the list. This is often useful if want to control the range in some way and apply the range to multiple parameters.

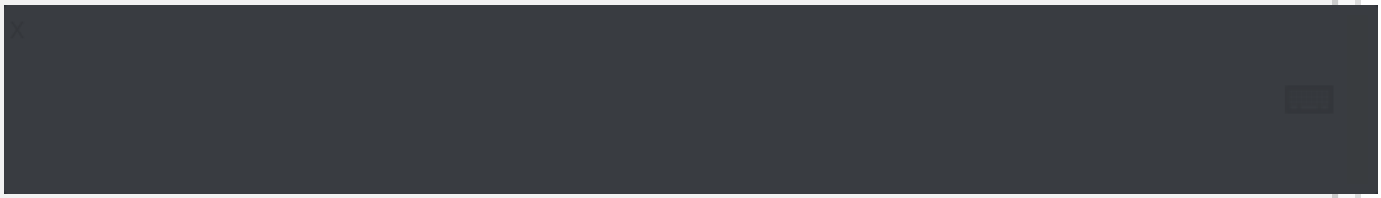
Parameter expression



Parameter expression panel lets you provide an algebraic expression for computing the final parameter value.

Post

Post button toggles whether the transform and min/max range are applied to the value x before evaluating the expression (Postprocessing mode, button on), or to the result of evaluating the expression (Preprocessing mode, button off, default).



Parameter expression

Parameter expression contains the expression used to process the parameter's value.

Here is the list of variables available for each expression:

x = the multiparameter value in 0..1
sr = current sampling rate
mp{i} = value of a multiparameter, e.g. mp1 for the first multiparameter.
p{i} = value of a parameter attached to this multiparameter, e.g. p1 for the first parameter. Note that this includes disabled parameters, which provides a handy way to use any value from the plugins, even if it is not supposed to be controlled by this MP - map it and disable it, then use the p{i} variable to use in the expression.

Some examples:
*x*x* or *x^2* - quadratic scaling making the parameter value "lower" for longer time. Similar to using a transformation and dragging the curvature down.
to01(from01(x,0.1,0.2), 0.001,1.0) - custom range for a parameter that goes linearly from 0.001 to 1.0 (for example) time, but your requested range is 0.1 to 0.2. from01(x,0.1,0.2) takes current value and produces range 0.1 to 0.2. to01 then converts it to the target parameter's range, in our example it is 0.001 to 1.0.
to01(from01(x,0.1,0.2), 0.0,10.0)^0.25 - custom range for a time parameter with range 0 to 10 seconds into linear scale 100ms to 200ms. from01(x,0.1,0.2) is the same as in the previous example. to01 then converts it into the target's parameter range being 0 to 10 seconds. Finally "^0.25" (fourth root) is an inversion of "^4", which is the scaling we use for time parameters most of the time.
f01(ffrom01range(x,200,400)) - custom range for a frequency parameter that goes from 20Hz to 20kHz (common range here), so that the MP goes from 200Hz to 400Hz instead. There are special functions involved here, since conversion of logarithmic scale is especially complex. Expression evaluator uses traditional C/C++ style formating, which is natural for most people. It provides arithmetics, logical and conditional operators. Following terms are supported:
Constants: **pi**, **e**, **sqrt2**, **ln2**

Arithmetic operators:
-a inverts the sign, e.g. "-x" produces +2 for x=-2
a+b = addition
a-b = subtraction

a*b = multiplication
a/b = division
a%b = modulo, remainder after division
a^b = power, e.g. "2^3" produces 2*2*2 = 8

Arithmetic functions:
min(a,b) = minimum of both values
max(a,b) = maximum of both values
limit(a,min,max) = a limited into the interval min..max
to01(a,min,max) = converts "a" as min..max to 0..1
from01(a,min,max) = converts "a" as 0..1 to min..max
tom11(a,min,max) = converts "a" as min..max to -1..1
fromm11(a,min,max) = converts "a" as -1..1 to min..max

Basic mathematic functions: **abs(x)** = absolute value, e.g. abs(-3) = 3 **sqr(x)** = x*x **sqrt(x)** = square root **exp(x)** = natural exponential e^x **ln(x)** = natural logarithm **log10(x)** = logarithm with base 10 **log(x, base)** = logarithm with specified base **inv(x)** = 1/x **sgn(x)** = sign of x, -1 or 0 or +1 depending on x*x **round(x)** = rounding to the nearest value **floor(x)** = rounding to the nearest lower value, e.g. floor(-2.3) = -3 **ceil(x)** = rounding to the nearest higher value, e.g. ceil(-2.3) = -2 **rand(x)** = random value from 0 to x

Functions for specific units:
f01(a) = converts "a" as frequency from 20..20000 into log scale 0..1
ffrom01(a) = converts "a" as 0..1 (log scale) to frequency from 20..20000
ffrom01limit(a) = same, but makes sure the output is in 20..20000 range
f01range(a,min,max) = converts "a" as frequency from min to max into log scale 0..1
ffrom01range(a,min,max) = converts "a" as 0..1 (log scale) to frequency from min to max
ffrom01rangelimit(a,min,max) = same, but makes sure the output is in min..max range
todb(a) = converts "a" as multiplier to dB value by calculating "20*log10(a)"
fromdb(a) = converts "a" as dB value to multiplier by calculating "10^(a/20)"

Trigonometric functions: **sin(x)**, **asin(x)**, **cos(x)**, **acos(x)**, **tan(x)**, **atan(x)**, **sinh(x)**, **cosh(x)**, **tanh(x)**

Logical operators:
a==b = comparison producing 1 if "a" and "b" are equal, 0 otherwise
a!=b = comparison producing 1 if "a" and "b" are NOT equal, 0 otherwise
a<b = comparison producing 1 if "a" is lower than "b", 0 otherwise
a<=b = comparison producing 1 if "a" is lower or equal to "b", 0 otherwise
a>b = comparison producing 1 if "a" is greater than "b", 0 otherwise
a>=b = comparison producing 1 if "a" is greater or equal to "b", 0 otherwise
!a = logical negation, 0 produces 1, 0 otherwise
a&&b = logical AND, produces 1 if both "a" and "b" are nonzero
a||b = logical OR, produces 1 if any of "a" and "b" are nonzero
a^^b = logical XOR, produces 1 if "a" and "b" are logically different
a ? b : c = if a is nonzero, then the result is b, otherwise it is c}}}

Cyclic mode

Cyclic mode switches the multiparameter into so-called cyclic mode. If you have say 4 banks, called A, B, C and D, and gradually increase the multiparameter value, it starts with A, then interpolates to B, then to C and finally to D. But after that you cannot interpolate back to A, because D is the last one, the maximum value. In cyclic mode the multiparameter behaves as if there were a clone of A at the end, hence after D is reached, the multiparameter interpolates back to A and creates a full circle A->B->C->D->A. This is handy for example if you use a saw wave modulator to drive the multiparameter and want to repeat the sequence of the banks.

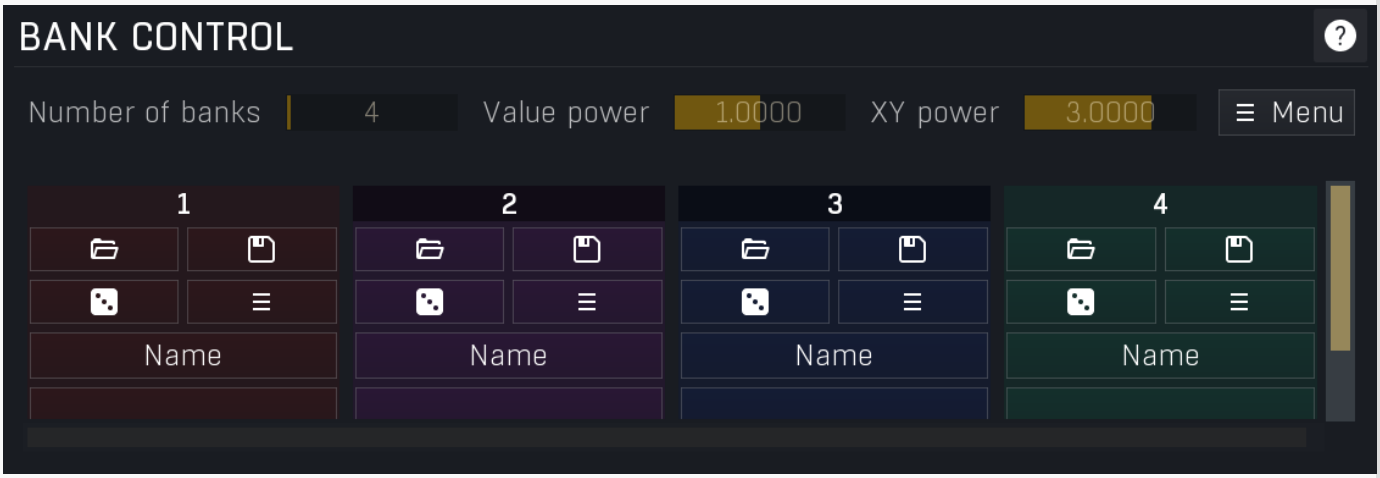
Interpolate values

Interpolate values controls if the parameter value is to be interpolated between the bank values or if it will take the value from the nearest bank. For example, when bank A contains the value 0% for the parameter and bank B contains 100% and you set the multiparameter to 30%, then when interpolation is enabled, 30% is selected for that parameter, when the interpolation is disabled, the nearest value, 0%, is selected. If you want the parameter to step from one bank value to another then disable interpolate values.

Set interpolate to all parameters buttons

Set interpolate to all parameters buttons sets the interpolate values setting for all parameters controlled by that multiparameter.

Bank control panel



Bank control panel is available only in **Banks mode** and contains tools to define the banks between which the multiparameter is interpolating. The multiparameter stores parameter values for each bank. Here you can load and save these values.
Each bank has 5 buttons and a value for each controlled parameter. Click the **load button** to load the bank values into the plugin. If you want to change say bank 3, you first click its **load button**, change whatever you need and resave the settings.
By clicking the **save button** you overwrite the bank's settings from those currently set in the plugin. A typical approach to define the multiparameter's behaviour is to set the number of banks, then go to the plugin editor, set all associated parameters to the values you would like to have in bank 1 and click the save button for bank 1, then modify the parameters to whatever you want in bank 2 and click the save button for bank 2, etc.
You can also use the **Random button** to generate random values using the smart-randomization engine for each of the banks.
And the **menu button** enables you to re-order the banks

For each bank, the values for each parameter are shown and can be changed as desired.

Number of banks

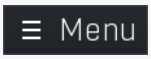
Number of banks controls the number of settings that the multiparameter stores for all parameters. By changing the multiparameter value all associated parameters are then modified according to these settings. Please note that when you change the number of banks, the multiparameter will behave differently, because the multiparameter's range from 0% to 100% will now be distributed between a different number of presets. If you had automated the multiparameter value in your host for example you will almost certainly need to edit / rewrite the automation envelope.

Value power

Value power lets you post-process values for each bank by specific power function. This either leads to higher values for low powers, or lower values for high powers. The same thing can be implemented using transforms, but this is much easier considering the potential number of parameters you may have there. It is especially usedful for circular XY pads. Use trial-and-error approach to set this up to your liking.

XY power

XY power is used only if **XY-pad between banks** is enabled and controls how the engine produces values for individual parameters. The higher the number is, the more will the engine separate them. Use trial-and-error approach to set this up to your liking.



Menu

Menu button provides some additional features for processing the entire set of banks.
Sort banks (up) reorders the banks so that the values of the selected parameter are in increasing order.
Sort banks (down) reorders the banks so that the values of the selected parameter are in decreasing order.
Reverse reverses the order of banks, so that the first bank contains values of the previously last one and so on.
Interpolate lets you change the number of banks, but keeps the values as they are now by calculating values of parameter for all banks. It is usually useful when you want to provide 'banks in between current banks', without manually calculating the new values.
Auto-gain (if available) temporarily enables AGC and automatically sets up the main plugin gain to each bank so that all banks provide similar output loudness. To use it, ensure that the main gain parameter is attached to the multiparameter, start playback of your sound material and press this button. It will take several seconds to complete depending on the number of the banks.
Set names by values sets the names for each bank to the values of the selected parameter. It may be handy when replicating existing parameters for example.

Load

Load button loads the bank settings by setting all associated parameters to the values in the particular bank.

Save

Save button saves the current values of all associated parameters into the particular bank. So you can edit all those parameters in the plugin then click the save button to store them in the bank.

Randomize

Randomize button loads random settings to the bank using the smart randomization engine. Only parameters associated with the multiparameter are randomized.

Generally, randomization in plugins works by selecting random values for all parameters, but rarely achieves satisfactory results, as the more parameters that change the more likely one will cause an unwanted effect. Our plugins employ a smart randomization engine that learns which settings are suitable for randomization (using the existing presets) and so is much more likely to create successful changes.

In addition, there are some mouse modifiers that assist this process. The smart randomization engine is used by default if no modifier keys are held.

Holding **Ctrl** while clicking the button constrains the randomization engine so that parameters are only modified slightly rather than completely randomized. This is suitable to create small variations of existing interesting settings.

Holding **Alt** while clicking the button will force the engine to use full randomization, which sets random values for all reasonable automatable parameters. This can often result in "extreme" settings. Please note that some parameters cannot be randomized this way.

Hold **Shift** while clicking the button to undo the previous randomization.

Menu

Menu button provides some additional options related to the bank.

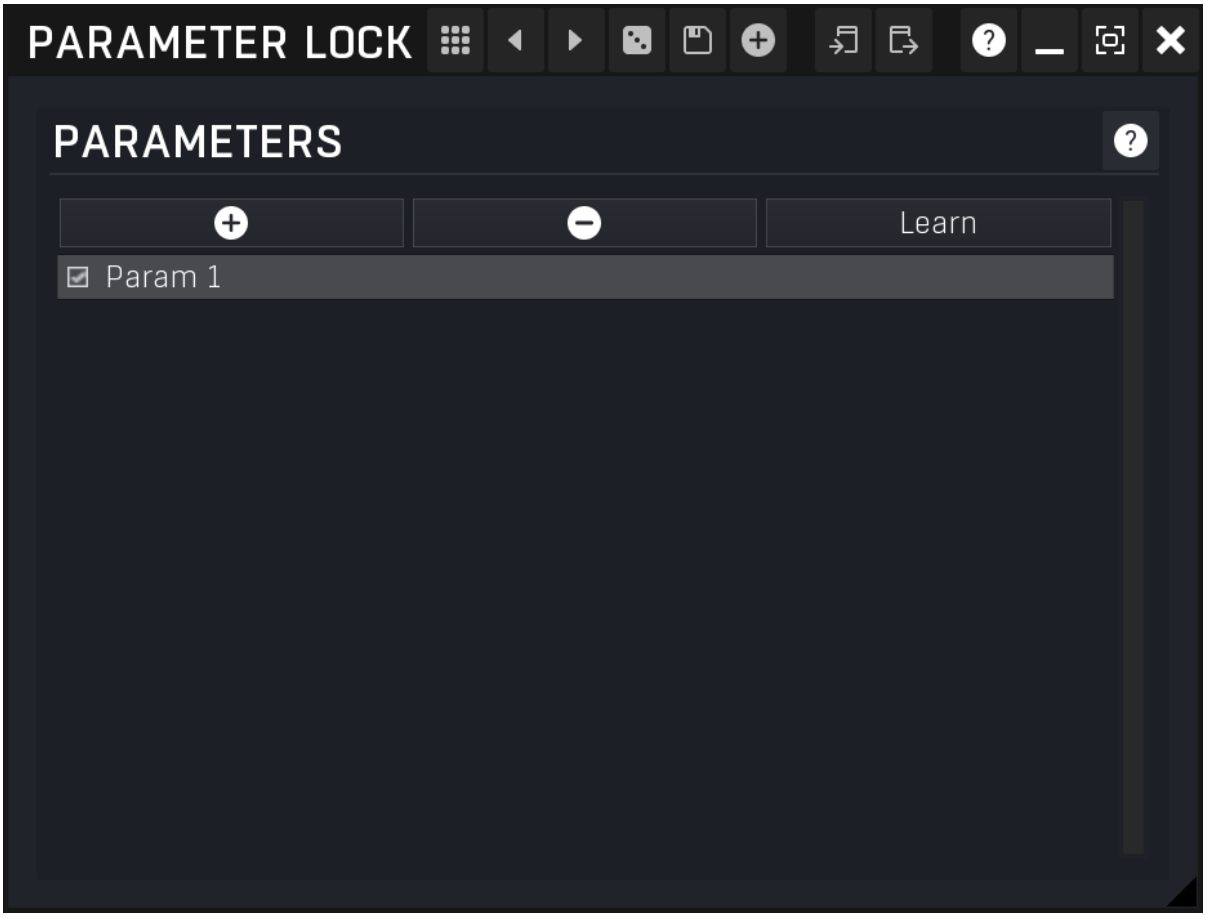
Name

Name button lets you rename the bank.

Name check

Name check button lets you rename the bank. This is a secondary name used for controls such as checkboxes and selectors if defined.

Parameter lock editor



Lock provides a simple way to keep some parameters unchanged when using randomization or browsing presets. You can still change these locked parameters by adjusting the control directly. You simply use the learn feature (right click) in the same way you would with modulators or multiparameters, and touch every parameter you want to keep locked. You can also select them directly in the Parameter Lock window where you can also save them as presets, copy & paste etc. Learning mode is ended by clicking the button again. Please note that this list is not saved with global plugin presets for obvious reasons. The parameters can be locked or unlocked directly in the list or by clicking the lock button associated with the parameter on the Easy screen.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

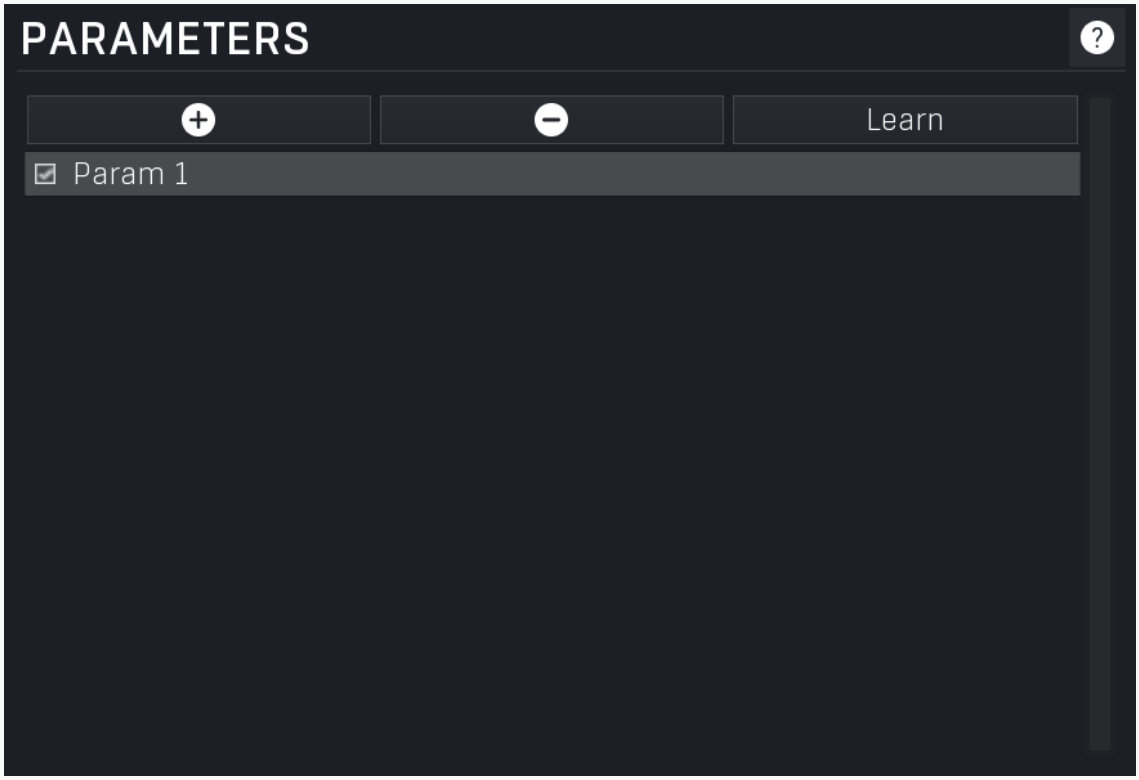
Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.

Parameters panel



Parameters panel configures the list of the parameters which are locked.

Add

Add button adds a parameter to the list of locked parameters. Alternatively you can use the learn feature available by right-clicking the paramlock button for example.

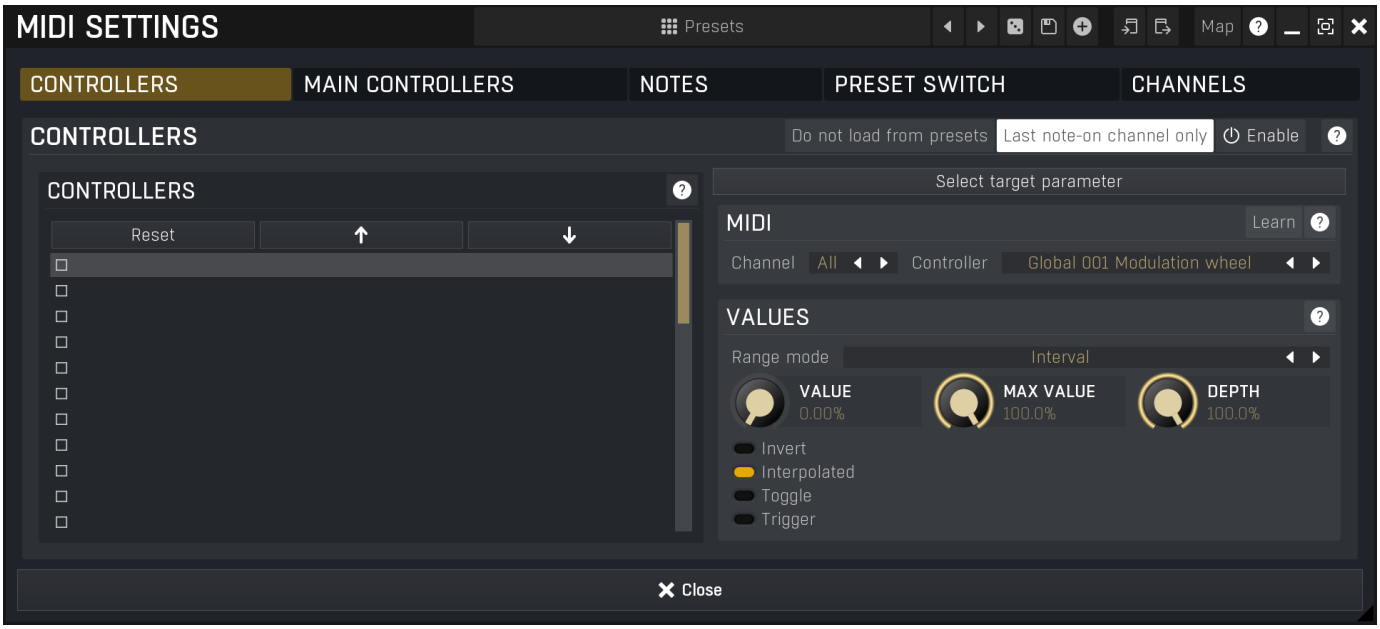
Delete

Delete button deletes the selected parameter from the list of controlled parameters.

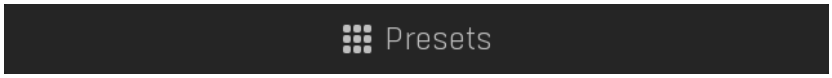
Learn

Learn button starts or stops the learning. Click it, then move some parameters in the plugin, then click it again. Learning can also be accessed from the global parameter lock menu.

MIDI editor



MIDI settings window lets you configure, how the plugin reacts to various MIDI messages. You can use MIDI controllers or MIDI notes and you can also configure a controller to switch between presets, which is especially useful for realtime performances.



Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

Copy

Copy button copies the settings onto the system clipboard.

Paste

Paste button loads the settings from the system clipboard.

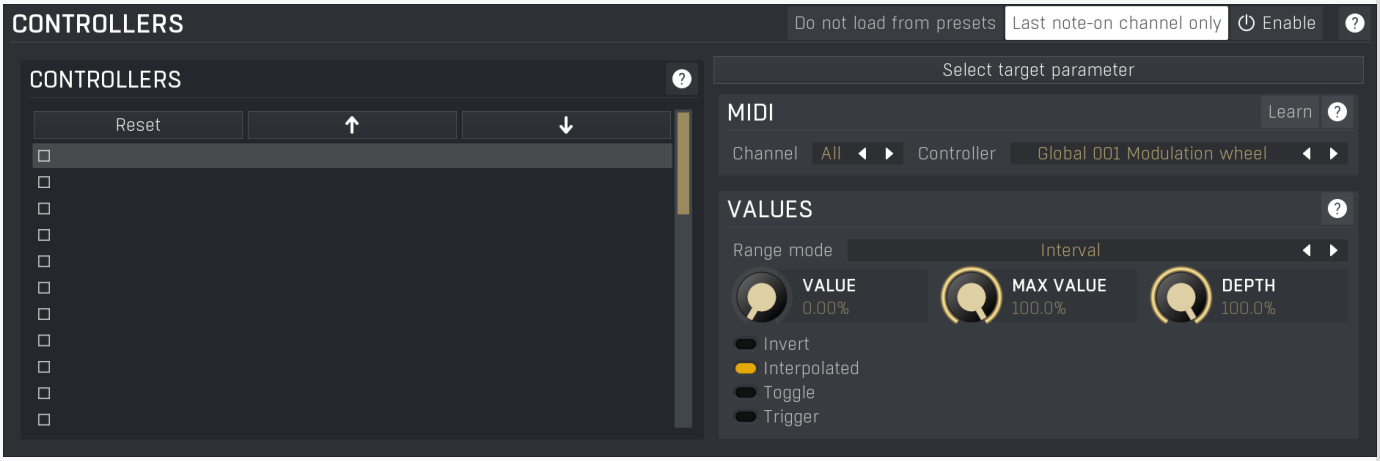
Map

Map button displays all current mappings of modulators, multiparameters and MIDI (whichever subsystems the plugin provides).

Tab selector

Tab selector switches between subsections.

Controllers panel



Controllers panel contains settings of MIDI controllers.

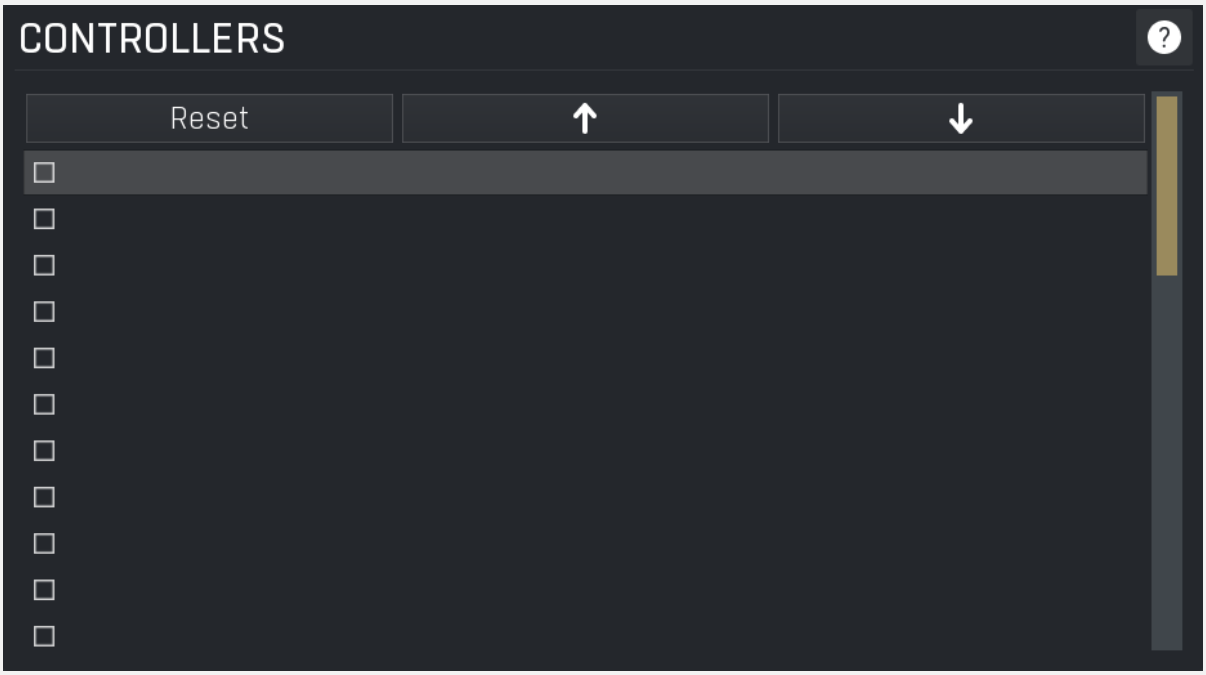
Do not load from presets

Do not load from presets button disables loading the controllers from presets. This may be handy if you have configured specific MIDI controllers with target parameters and you want to browse the presets without the need to configure them every time. Please note that some presets may rely on specific controllers though. For example, if a preset requires a velocity controller to provide velocity-dependent response, this option will avoid loading it, so the preset won't be complete, until you reconfigure it.

Last note-on channel only

Last note-on channel only button makes the engine more suitable for voice-per-channel devices. These devices are able to send different controllers for each note you press, which however means that these could collide. This option makes the engine pass only the controllers that are related to the last note you pressed. For classic keyboards it is not relevant as you will usually use a single MIDI channel to transmit both the controllers and notes. Some more modern keyboard controllers will allow you to select one MIDI channel for the notes and a different one (or the same one) for the controllers.

Controllers



Reset

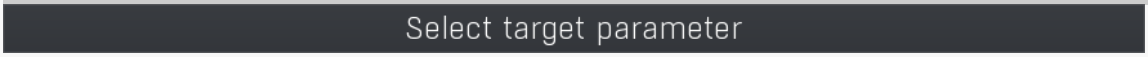
Reset button resets the selected controller to undefined state.

Up

Up button moves the selected controller up one item, if possible. This may be useful when keeping things organized, but please note that if you have some multiparameter, modulator or another subsystem access the ranges of individual controllers, this function will reorder them, so these connections will no longer be correct.

Down

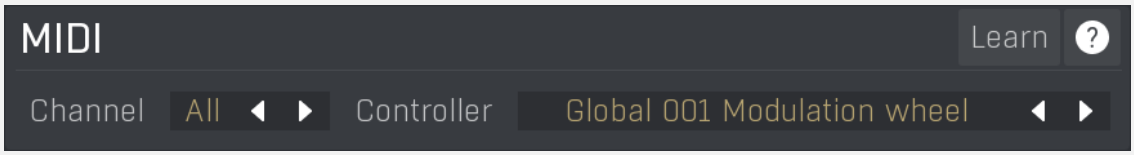
Down button moves the selected controller down one item, if possible. This may be useful when keeping things organized, but please note that if you have some multiparameter, modulator or another subsystem access the ranges of individual controllers, this function will reorder them, so these connections will no longer be correct.

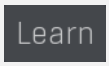


ParameterIndex

ParameterIndex button lets you choose the parameter being controlled. The set contains all automatable parameters.

MIDI





Learn

Learn enables or disables MIDI learn. When enabled, the plugin listens to both the MIDI CC messages from the controllers that you touch and the target parameters that you touch and associates the last-touched of each with the selected slot. You can perform several mappings by selecting another slot, adjusting a hardware controller. Then adjusting a target parameter, and repeating those steps for each mapping desired.

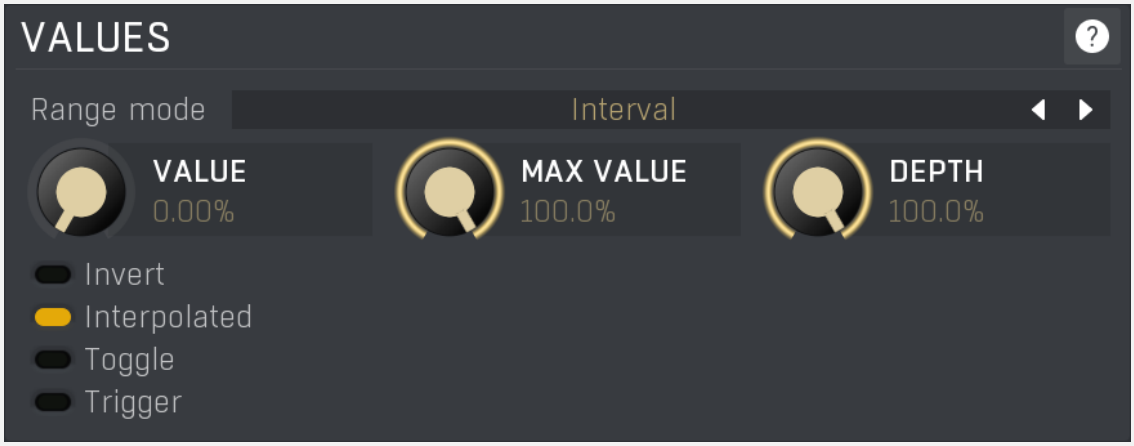
Channel

Channel defines the controller MIDI channel.

Controller

Controller defines the source controller.

Values



Range mode

Range mode defines how the parameter range is selected. While sometimes it is better to specify minimum and maximum, other times it is better to use a nominal center and depth (% of full scale). This control allows you to define which one it will be.

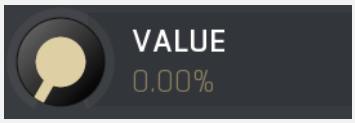
Up and down mode makes the values go above and below the selected **Value**, which is considered the center. The interval is made smaller if necessary.

Full range mode is similar, except the range is symmetrically constrained, so the selected **Value** may not be the center anymore.

Up/down only modes goes from the selected value up/down only.

Let's compare these 4 modes. Taking a value of -12dB value, with a depth of 75% and a scale of +/- 24dB. The nominal range is therefore = +/-24 dB * 75% = 36dB. With values of 0%, 50% and 100% the outputs are:
Up and down: -24, -12, 0 (range constrained to 12 dB either side)
Full range: -24, -6, 12 (range limited to minimum, but not constrained)
Up only: -12, 6, 24 (range not constrained = +/-24 dB * 75% = 36dB)
Down only: -12, -18, -24 (range limited to minimum)

Interval mode is the most simple one and goes from **Value** to **Maximal value**.

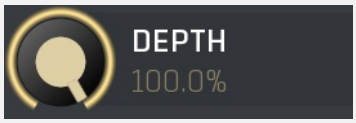


Value

Value defines the center of the target parameter's range or the minimum if the **Range mode** is set to **Interval**.

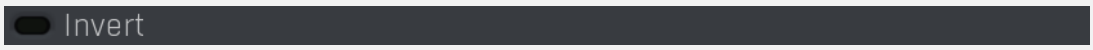
Maximal value

Maximal value defines the upper limit of the target parameter's range. It is available only if the **Range mode** is set to **Interval**. This value can be lower than **Value**. 0% is always mapped to reference>Value and 100% to reference>Maximal value.



Depth

Depth defines size of the target parameter's range. It is used only if the **Range mode** is not set to **Interval**.



Invert

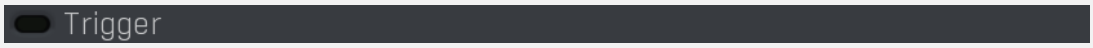
Invert checkbox inverts the controller shape, so the minimum becomes the maximum etc.

Interpolated

Interpolated makes the controller value interpolated over the time using the smart interpolation. This approach ensures there won't be abrupt changes, which could lead to clicks and pops. However sometimes you may want to apply these changes immediately - for example when changing ADSR based on the note velocity, in which case this parameter should be disabled.

Toggle

Toggle mode makes the controller switch between the maximum and minimum of the target parameter whenever triggered. By default triggering it means going from values below 50% to above 50%. By enabling **Trigger** you can make it perform the trigger everytime the value is changed.



Trigger

Trigger mode makes the controller automatically produce maximum and the minimum right after it. It can be handy with some buggy MIDI controllers providing buttons, which however do not send value 0, and only repeat value 127. Trigger makes it behave like the minimum was actually sent by the MIDI controller a little bit after the original message.

Main controllers panel

Main controllers panel lets you define the set of main MIDI controllers on your MIDI device. These are not stored with the presets, so using them lets you easily switch between MIDI controllers, create presets that will work for users of other MIDI controllers etc. Using the Main controllers is no different than using the standard MIDI controllers, but the extra 'layer' can make things simple when using multiple controllers and also in general situations where your MIDI device has several controllers with quite 'random' numbers.

Presets

Presets button displays a window where you can load and manage available presets. Hold **Ctrl** when clicking to load a random preset instead.

Left arrow

Left arrow button loads the previous preset.

Right arrow

Right arrow button loads the next preset.

Randomize

Randomize button loads a random preset.

Save

Save button replaces the current preset.

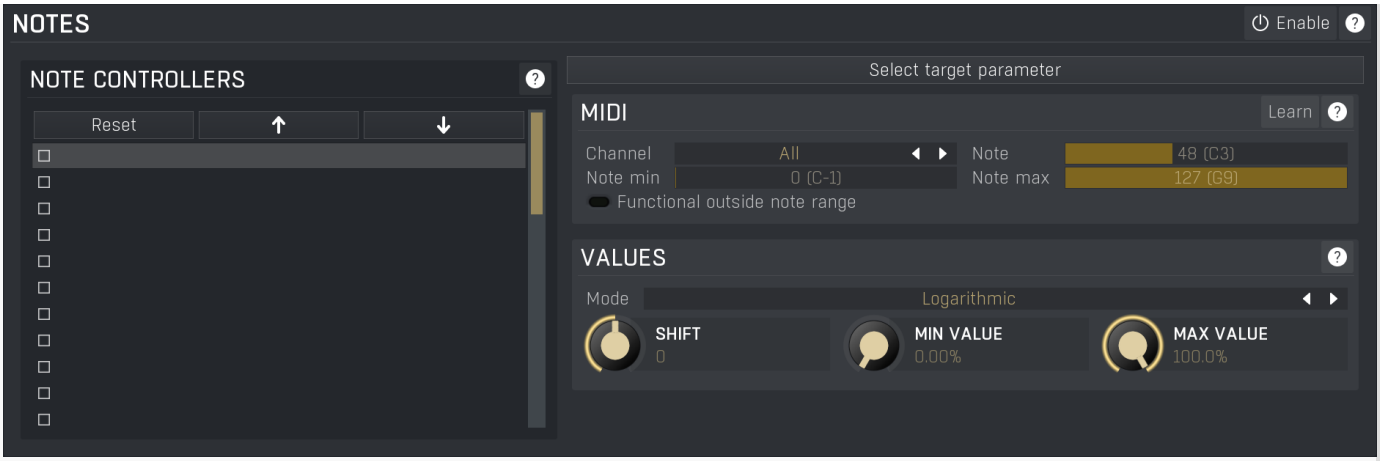
Controller

Controller defines the MIDI controller associated to this Main controller.

Learn

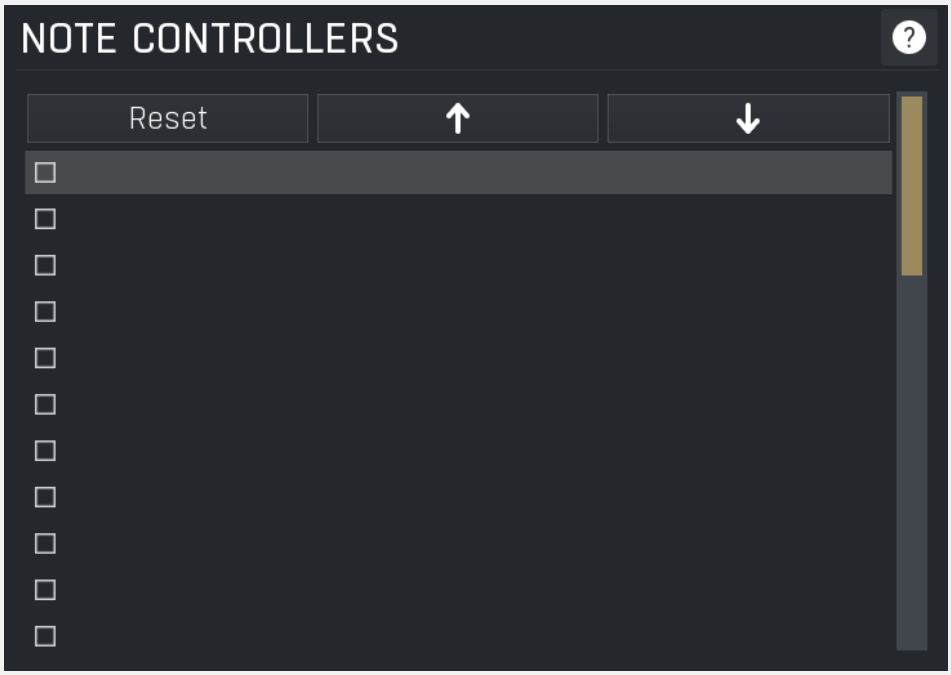
Learn enables or disables MIDI learn. When enabled, the plugin listens to the controllers you touch and associates them to the main controller.

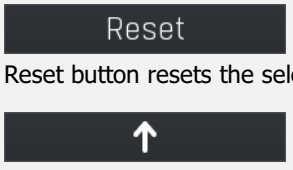
Notes panel



Notes panel contains settings of MIDI note controllers, if you want to control parameters using MIDI keys.

Note controllers





Reset

Reset button resets the selected controller to undefined state.

Up

Up button moves the selected controller up one item, if possible. This may be useful when keeping things organized, but please note that if you have some multiparameter, modulator or another subsystem access the ranges of individual controllers, this function will reorder them, so these connections will no longer be correct.

Down

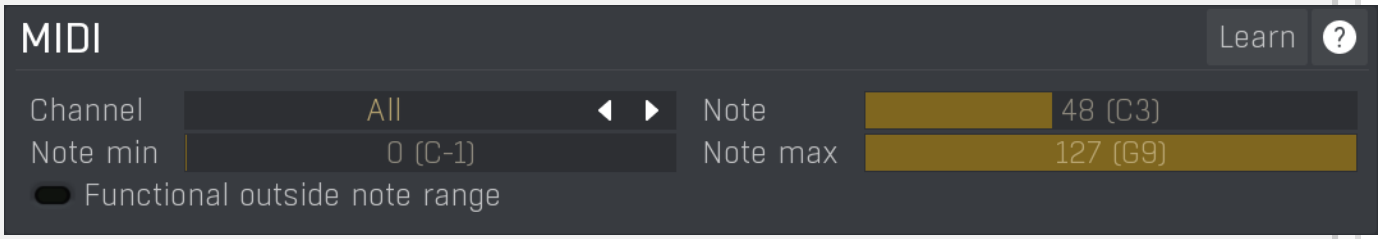
Down button moves the selected controller down one item, if possible. This may be useful when keeping things organized, but please note that if you have some multiparameter, modulator or another subsystem access the ranges of individual controllers, this function will reorder them, so these connections will no longer be correct.

Select target parameter

Learn

Learn enables or disables MIDI learn. When enabled, the plugin listens to both the notes you touch and the parameters you touch and associates them with the selected slot.

MIDI



Channel

Channel defines the controller MIDI channel.

Note

Note defines the controller's target MIDI note. It is used only in On/off and Switch modes, which you can set using **Mode** parameter (in the **Values** panel).

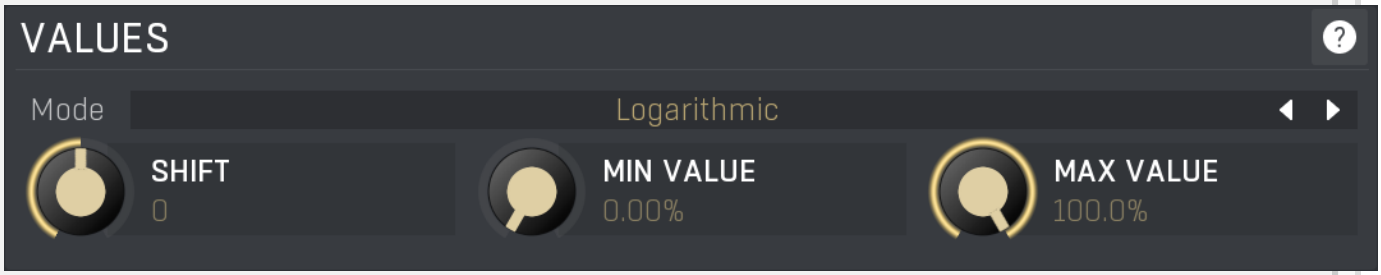
Note min

Note min controls the lowest note to be used by a controller in Linear or Logarithmic mode. The minimum value of the target parameter will then be associated to this note.
If both Note min and Note max parameters are default, the plugin takes the actual frequency of each note, and transforms it into the range 20Hz to 20kHz, which is the range used by all equalizers and filters, so that you can literally play a parameter on a MIDI keyboard. If you change either of these 2 parameters however, the plugin takes the range of notes as the requested interval. This is useful for example if you have a small MIDI keyboard used for soloing and you want increase some parameter the higher you play. In the default mode it would be difficult, since the range of frequencies is much bigger than the range of your MIDI keyboard.*Set the **Note min** and **Note max** to C0 and B0 respectively, the **Mode** to Logarithmic and select a suitable target parameter (Dry/Wet is fine). Send MIDI notes in the specified range to the plugin and you will see the target parameter increase (by 9.09% (= 100 / (12-1)) for a 100% range).*

Functional outside note range

Functional outside note range makes the note controller work even if the note isn't in the specified range, clamping the value to the minimum or maximum.

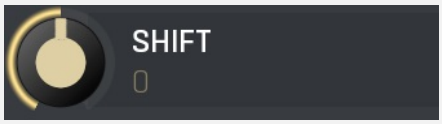
Values



Mode

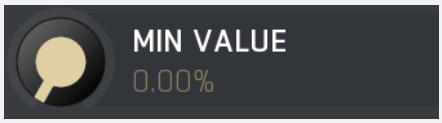
Mode controls how the controller works.
Key takes the note index and transforms it into 0..1, which is the output of any controller. This mode is useful for scale switches for example - if you want to use MIDI keys to change values linearily.
Key (in octave) is similar but it has only 12 values - one per each key and it doesn't matter which octave you press it in.
Linear converts the notes into frequencies and then transform them into the linear scale from 20Hz to 20kHz.
Logarithmic converts the notes into the frequencies and then into the logarithmic scale from log(20) to log(20000). A typical use case is when you want to control an equalizer band using a MIDI keyboard. Since EQ frequencies work in logarithmic scale, this mode makes both things compatible and the EQ frequency will be set to the note frequency.
On/off modes react only to single notes and can be used for triggers. When the Note On is received the parameter is changed to its **Max value** and when the Note Off is received the parameter is changed to its **Min value**. So this mode can also be used to change between any 2 parameter values.
Switch modes are similar, but only recognize when a note is pressed. The Note Offs are ignored. Note Ons select the **Max value** and **Min value** alternately. In all octaves mode it doesn't matter which octave is used. For example, this is useful when you want to use any note C to switch something on and off.
Velocity modes do not actually follow the note number being pressed, but it's velocity instead. While you can do the

same thing with normal MIDI controllers using the special Velocity controllers, this one allows you to select only some notes to follow.



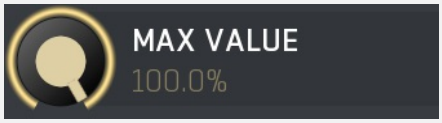
Shift

Shift lets you shift the original note up or down by the specified number of semitones.



Min value

Min value defines the minimum value for the target parameter.



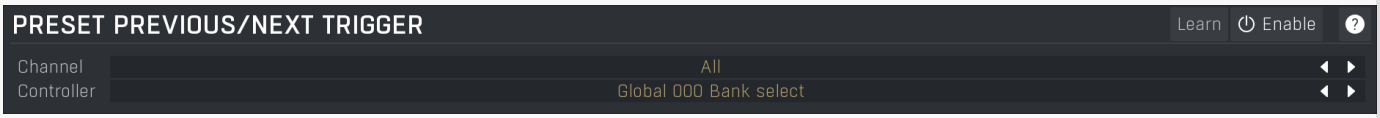
Max value

Max value defines the maximum value for the target parameter.

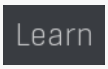
Enable MIDI program change

Enable MIDI program change enables processing program change MIDI message.

Preset previous/next trigger panel

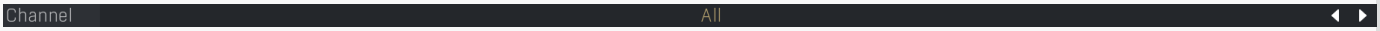


Preset previous/next trigger panel lets you select a MIDI controller, which will switch presets. It provides the same action as clicking the arrows next to the main preset button. When the controller value gets below 33%, the previous preset is loaded. When the controller value gets above 66%, the next preset is loaded.



Learn

Learn enables or disables MIDI learn.



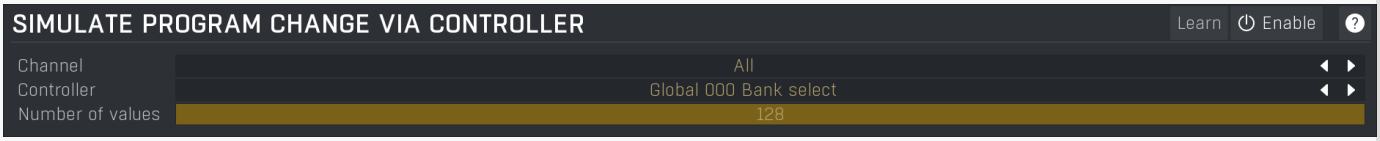
Channel

Channel defines the controller MIDI channel.

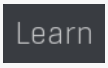
Controller

Controller defines the source controller.

Simulate program change via controller panel

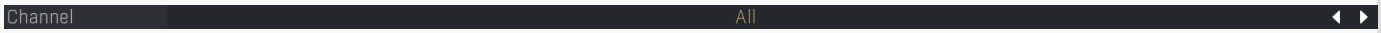


Simulate program change via controller panel lets you select a MIDI controller, that will work as program change, for convenience. You can use it then to switch between A-H presets or presets via panel below.



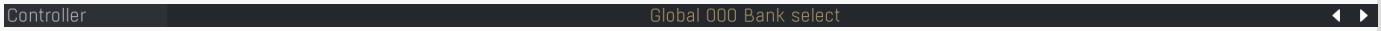
Learn

Learn enables or disables MIDI learn.



Channel

Channel defines the controller MIDI channel.



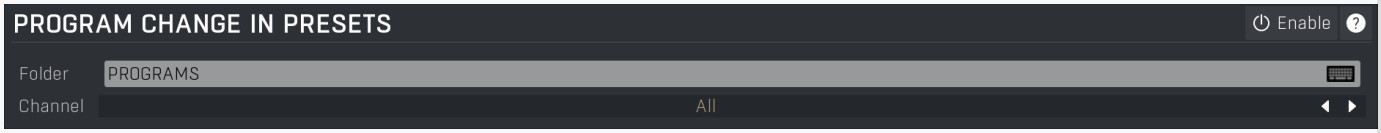
Controller

Controller defines the source controller.

Number of values

Number of values defines the number of programs to switch between. By default Program change MIDI standard offers 128 programs. However it may by too many and could be hard to actually control with the specific controller. Hence you can lower the number of actual programs.

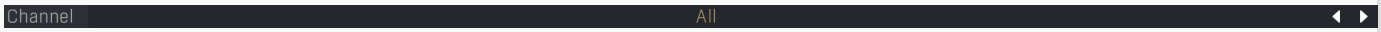
Program change in presets panel



Program change in presets panel enables the MIDI program change processing. If disabled, the plugin follows Program Change messages by changing the A-H presets. The obvious disadvantage is that this way there are just 8 presets. By enabling this feature the plugin stops selecting A-H presets and rather loads different presets from the specified preset folder, including all sub-folders. The default folder is called "Programs". To use it, you simply need to create a preset folder called Programs and put the presets into it. Note that the order matters of course. And you can change the folder name at any time, so you can have several sets of selectable presets.

Folder

Folder defines the preset folder from which the presets for program-change MIDI messages are taken.



Channel

Channel defines the program change MIDI channel.

Ignore pitch-bend range message

Ignore pitch-bend range message allows one to ignore the MIDI Registered Parameter message communicating the pitch-bend range and to keepthe value set by the **Pitch-bend range** control.

Ignore MPE Configure message

Ignore MPE Configure message toggles whether the plugin should respond to the MPEConfigure message, or should just rely on the configuration provided in the **MPE** panel.

Used controls

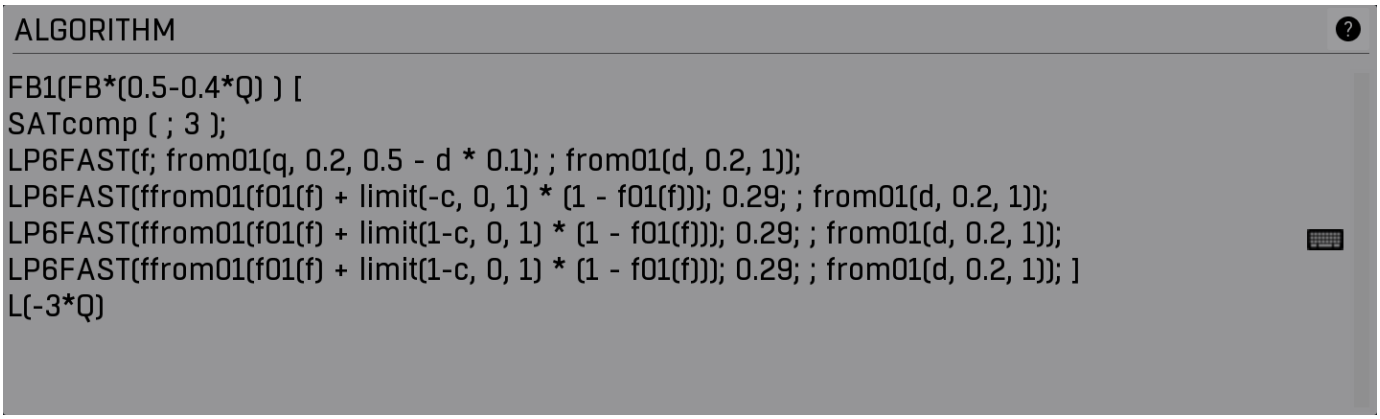
Here we discuss the general properties of all application controls. As a most important rule you should note, that you can always use any question mark button or F1 (or Ctrl+F1 or Ctrl+H) key with the mouse cursor over a specified control to get detailed information about what it does and how to use it.

Value button

Value button is an alternative to the knobs and its main advantage is that it is very small. In some cases the button simply serves as a clickable item and a menu is shown when clicked. However the mouse wheel and other controls still do work.

- **Click and drag using the left mouse button** to change the value.
- **Right mouse button** shows a menu with additional options. Hold **ctrl/cmd** to select the default value.
- **Mouse wheel**, **arrow keys** and vertical drag using **middle mouse button** or using **left mouse button while holding Ctrl** modifies the value more precisely.
- **Home key** configures the minimal possible value, conversely **end key** setups the maximal one.
- **Esc or Backspace keys** restore the original value when either one is pressed during dragging.
- **Shift + left mouse button** or **double-click using left mouse button** lets you edit the value as text. You can use the virtual keyboard or type on your computer keyboard. In some cases this shows a menu with all possible values instead.
- **Alt + press then release** measures the time between the press and the release and applies it as time/frequency tap. Usable only for certain values of course.

Edit



Edit control provides a standard way to specify any kind of text or numeric string. It can have a limited length or a limited set of allowed characters.

- **Shift+arrows/Home/End** modifies selection.
- **Ctrl+C** copies the selected text to the system clipboard.
- **Ctrl+V** pastes text from the system clipboard replacing the selected text or inserting it at the current cursor position.
- **Ctrl+A** selects the whole text.
- **Ctrl+Tab** inserts a tabulation character. This combination is provided because the Tab key alone moves the keyboard focus to the next control.

Switcher

Switcher is an alternative to a tracker or knob control, but it has a limited set of values.

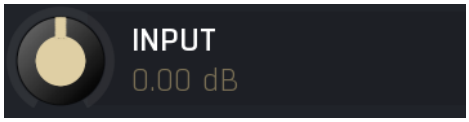
- **Left mouse button** shows a menu with list of all possible values. This function might be unavailable in certain cases when the number of possible values is too high.
- **Right mouse button** shows a menu with additional options. Hold **ctrl/cmd** to select the default value.
- **Up** and **Down** arrow keys, **buttons** in the control and **mouse-wheel** increase or decrease the value.

Zoomer

Zoomer provides a simple way to zoom and move in an enlargeable view.

- **Plus button** zooms-in.
- **Minus button** zooms-out.
- **Zoom default button** zooms to the default ratio, which typically means full zoom-out.
- **Lock button** locks the zoom ratio.

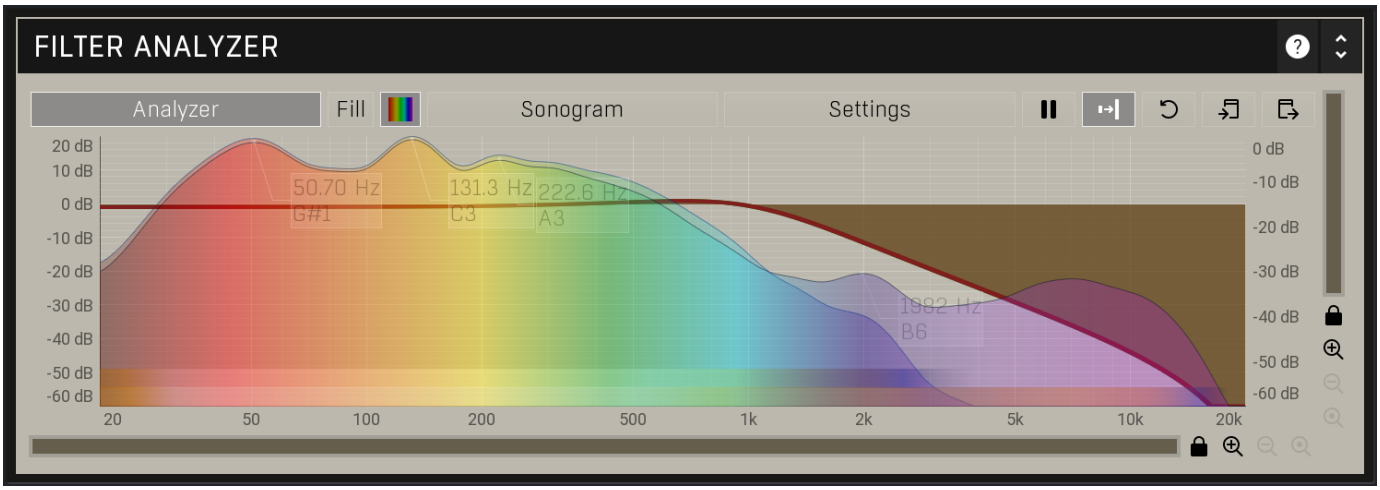
Knob



Knob simulates physical knobs used to edit various values.

- **Click and drag using the left mouse button** to change the value.
- **Right mouse button** shows a menu with additional options. Hold **ctrl/cmd** to select the default value.
- **Mouse wheel**, **arrow keys** and vertical drag using **middle mouse button** or using **left mouse button while holding Ctrl** modifies the value more precisely.
- **Home key** configures the minimal possible value, conversely **end key** setups the maximal one.
- **Esc or Backspace keys** restore the original value when either one is pressed during dragging.
- **Shift + left mouse button** or **double-click using left mouse button** lets you edit the value as text. You can use the virtual keyboard or type on your computer keyboard. In some cases this shows a menu with all possible values instead.
- **Alt + press then release** measures the time between the press and the release and applies it as time/frequency tap. Usable only for certain values of course.

Graph editor



Graph editor will show and edit one or more graphs.

- **Zoomers** below and on the right control the zoom amount and position of the view.
- **Mouse wheel** zooms in or out. Hold **Shift** to move the view horizontally instead, without zooming. Hold **Ctrl** to move the view vertically instead, without zooming. Alternatively you can zoom in using **Alt + right button double click** and out using **Alt + left button double click**. You can also use keyboard **numbers 0 to 9** to quickly set the zoom level.
- **Drag a rectangle using the left mouse button while holding Alt** zooms into the selected rectangle if possible.
- **Drag using the left mouse button while holding Alt and Ctrl** to scroll the view. This is not possible when zoomed all the way out as there is nothing to scroll.

Tab-set

Tab-set is typically used wherever there is too much to edit, but not enough space to display it all. It can be also used to switch between possible alternatives.

- **Left mouse button** selects a tab.

- **Ctrl + Left mouse button or Right mouse button** displays the whole tab in a pop-up window (this is not used for all sets of tabs). This comes handy when you want to have multiple tabs visible at the same time.
- **Left and Right arrows** select the neighbouring tab.
- Click on one of the buttons on the border to scroll the control and show tabs that are currently invisible.

Installation, activation, introduction to audio plugins

Installation

All MeldaProduction plugins are currently available for Windows and Mac OS X operating systems, both 32-bit and 64-bit versions. You can download all software directly from our website. Since the installation procedures for the two operating systems are quite different, we will cover each one separately.

The download files for the effects include all the effects plugins and MPowerSynth. During the installation process you can select which plugins or bundles to install. If you have not licensed all of the plugins in a bundle then you just need to activate each plugin separately.

If you have multiple user accounts on your computer, always install the software under your own account! If you install it under one account and run it under a different one, it may not have access to all the resources (presets for example) or may not even be able to start.

Installation on Windows

All plugins are available for VST, VST3 and AAX interfaces. The installer automatically installs both the 32-bit and 64-bit versions of the plugins.

Note: Always use 32-bit plugins in 32-bit hosts, or 64-bit plugins in 64-bit hosts. 64-bit plugins cannot work in 32-bit hosts even if the operating system is 64-bit. Conversely, never use 32-bit plugins in 64-bit hosts. Otherwise they would have to be 'bridged' and, in some hosts, can become highly unstable.

You can select the destination VST plugins paths on your system. The installer will try to detect your path, however you should check that the correct path has been selected and change it if necessary. In all cases it is highly recommended to use the current standard paths to avoid any installation issues:
32-bit Windows:
C:\Program files\VstPlugins

64-bit Windows:
C:\Program files (x86)\VstPlugins *(for 32-bit plugins)*
C:\Program files\VstPlugins *(for 64-bit plugins)*

If your host provides both VST and VST3 interfaces, VST3 is usually preferable. If a plugin cannot be opened in your host, ensure the plugin file exists in your VST plugin path and that if your host is 32-bit, the plugin is also 32-bit, and vice versa. If you experience any issues, contact our support via info@meldaproduction.com

Installation on Mac OS X

All plugins are available for VST, VST3, AU and AAX interfaces. Installers create both 32-bit and 64-bit versions of the plugins.

If your host provides multiple plugin interface options, VST3 is usually preferable. If you experience any issues, contact our support via info@meldaproduction.com

Most major hosts such as Cubase or Logic should work without problems. In some other hosts the keyboard input may be partly non-functional. In that case you need to use the virtual keyboard available for every text input field. You may also experience various minor graphical glitches, especially during resizing plugin windows. This unfortunately cannot be avoided since it is caused by disorder in Mac OS X.

Uninstallation on Windows

The Uninstaller is available from the Start menu and Control panel, in the same way as for other applications. If you don't have any of these for any reason, go to Program files / MeldaProduction / MAudioPlugins and run setup.exe.

Uninstallation on OSX

The Uninstaller is available from Applications / MeldaProduction / MAudioPlugins / setup.app.

Deleting all data, presets etc.

Even if you uninstall the plugins, some data will be left behind - because of potential crossdependencies or because these are your presets, settings, configurations etc. If you want to wipe out everything, please manually delete following folders:

Windows:
C:\ProgramData\MeldaProduction
C:\Users\{username}\AppData\Roaming\MeldaProduction

OSX:
Macintosh HD/Library/Application support/MeldaProduction/
HOME/Library/Application support/MeldaProduction

Performance precautions

In order to maximize performance of your computer and minimize CPU usage it is necessary to follow a few precautions. The most important thing is to keep your buffer sizes (latency) as high as possible. There is generally no reason to use latency under 256 samples for 44kHz sampling rates (hence 512 for 96kHz etc.). Increasing buffer sizes (hence also latency) highly decreases required CPU power. In rare cases increasing buffer sizes may actually increase CPU power, in which case you can assume your audio interface driver is malfunctioning.

You should also consider using only necessary features. Usually the most CPU demanding features are oversampling and modulation of certain parameters. You can reduce modulation CPU usage at the cost of lower audio quality in Settings/Settings/Modulator protection.

Troubleshooting

The plugins are generally very stable, there are known problems however.

GPU compatibility

The software uses hardware acceleration to move some of the processing (mainly GUI related) from your CPU (processor) to your GPU (graphics processing unit). It is highly recommended to use a new GPU, as it will provide higher performance improvements, and update your GPU drivers. Older GPUs are slower and may not even provide required features, so the software will have to perform all calculations in the main CPU. We also have had extremely bad experiences with GPUs from ATI and despite the fact that software is now probably bulletproof, it is recommended to use NVidia GPUs as there has not been a single case of a problem with them.

If you experience problems with your GPU (crashing, blank/dysfunctional GUI), and that you cannot disable the GPU acceleration from the plugin's Settings window itself, download this file:

http://www.meldaproduction.com/download/GPU.zip

And place the GPU.xml included in the zip into

Windows: C:\Users\{username}\AppData\Roaming\MeldaProduction
Mac OS X: ~/Library/Application support/MeldaProduction

Memory limits of 32-bit platform

Most hosts are now 64-bit ready, however some of them are not or users willingly choose 32-bit edition, because the required plugins are not 64-bit ready yet. All our software is 64-bit ready. Please note that you must NOT use the 64-bit plugins in 32-bit hosts, even if you have a bridge. If you are stuck with a 32-bit host for any reason, note that there is a memory limit (about 1.5 GB), which you may not exceed. This can happen if you load too many samples or different plugins for example. In that case the host may crash. There is no other solution than to use a 64-bit host.

Updating

You can use "Home/Check for updates" feature in any of the plugins. This will check online if there is a newer version available and open the download page if necessary.

To install a newer (or even older) version you simply need to download the newest installer and use it. There is no need to

uninstall the previous version, the installer will do that if necessary. You also do not need to worry about your presets when using the installer. Of course, frequent backup of your work is recommended as usual.

Using touch-screen displays

Touch screen displays are supported on Windows 8 and newer and the GUI has been tweaked to provide a good workflow. Up to 16 connections/fingers/inputs are supported. Any input device such as touch-screens, mouse, tablets are supported. These are the main gestures used by the plugins:
- Tap = left click
- Double tap = double click
- Tap & hold and quickly tap next to it with another finger = right click. Tap & hold is a classic right-click gesture, however that doesn't provide a good workflow, so came up with this method, which is much faster and does not collide with functionality of some elements.

Purchasing and activation

You can purchase the plugin from our website or any reseller, however purchasing directly from our website is always the quickest and simplest option. The software is available online only, purchasing is automatic, easy and instant. After the purchase you will immediately receive a keyfile via email. If you do not receive an e-mail within a few minutes after your purchase, firstly check your spam folder and if the email is not present there, contact our support team using **info@meldaproduction.com** so we can send you the licence again.

To activate the software simply **drag & drop the licence file onto the plugin**. Unfortunately some hosts (especially on Mac OS X) either do not allow drag & drop, or make it just too clumsy, so you can use Home/Activate in any of the plugins and follow the instructions. For more information about activation please check the **online video tutorial**.

You are allowed to use the software on all your machines, but only you are allowed to operate the software. The licences are "to-person" as defined in the licence terms, therefore you can use the software on all your computers, but you are the only person allowed to operate them. MeldaProduction can provide a specialized licence for facilities such as schools with different licence terms.

Quick start with your host

In most cases your host will be able to recognize the plugin and be able to open it the same way as it opens other plugins. If it doesn't, ensure you did installation properly as described above and let your host rescan the plugins.

Cubase

Click on an empty slot (in mixer or in track inserts for example) and a menu with available plugins will be displayed. VST2 version is located in MeldaProduction subfolder. However VST3 version is recommended and is located in the correct folder along with Cubase's factory plugins. For example, dynamic processors are available from the "Dynamics" subfolder.

To route an audio to the plugin's **side-chain** (if it has one), you need to use the VST3 version. Enable the side-chain using the arrow button in the Cubase's plugin window title. Then you can route any set of tracks into the plugin's side-chain either by selecting the plugin as the track output or using sends.

To route **MIDI** to the plugin, simply create a new MIDI track and select the plugin as its output.

Logic

Choose an empty insert slot on one of your audio tracks (or instrument tracks for example) and select the plugin from the popup menu. You will find it in the Audio Units / MeldaProduction folder.

To route an audio to the plugin's **side-chain** (if it has one), a side-chain source should be available in the top of the plugin's window, so simply select the source track you want to send to the plugin's side-chain.

To route **MIDI** to the plugin, you need to create a new Instrument track, click on the instrument slot and select the plugin from AU MIDI-controlled Effects / MeldaProduction. The plugin will receive MIDI from that track. Then route the audio you want to process with the plugin to this track.

Studio One

Find the plugin in the Effects list and drag & drop it onto the track you would like to insert the plugin to.

To route an audio track to the plugin's **side-chain** (if it has one), first enable the side-chain using the "Side-chain" button in the Studio One's plugin window title. Then you can route any set of tracks into the plugin's side-chain from the mixer.

To route **MIDI** to the plugin, simply create a new MIDI track and select the plugin as its output.

Digital performer

In the Mixing Board, find an empty slot in the track you would like to insert the plugin to. Click on the field and select the plugin from the effects list.

To route an audio track to the plugin's **side-chain** (if it has one), choose the track you want to send using Side-chain menu, which appears at the top of the DP's plugin window.

To route **MIDI** to the plugin, simply create a new MIDI track in the Track view and select the plugin as its output.

Reaper

Click on an empty slot in the mixer and a window with available plugins will be displayed. Select the plugin you want to open by double clicking on it or using Ok button.

It is highly recommended to select all MeldaProduction plugins in the plugin window the first time you open it, click using your right mouse button and enable "Save minimal undo states". This will disable the problematic Undo feature, which could cause glitches whenever you change certain parameters.

To route an audio track to the plugin's **side-chain** (if it has one), click on I/O button of the side-chain source track in the mixer. Routing window will appear, there you click "Add new send" and select the track the plugin is on. In the created send slot select the channels (after the "=>" mark) for the send, in stereo configuration 3/4 for example. Note that this way the whole track receives the side-chain signal and all plugins with it. It is possible to send it to a single plugin only, but it is more complicated, please check the Reaper's documentation about that.

To route **MIDI** to the plugin, create a new MIDI track and do the same thing as with side-chain, except you don't need to change output channels.

Live

In Session view, select the track you would like to insert the plugin to. At the left top of Ableton Live's interface, click on the Plug-in Device Browser icon (third icon from the top). From the plugins list choose the plugin (from MeldaProduction folder), double click on it or drag & drop it into the track.

The X/Y grid usually doesn't provide any parameters of the plugin. This is because the plugins have too many of them, so you have to select them manually. Check Live's documentation for more information.

To route an audio to the plugin's **side-chain** (if it has one), select the track you want to send to the side-chain and in the 'Audio To' menu, choose the audio track that has the plugin on it. Then in the box just below that select the plugin from the menu.

NOTE: Live does NOT support any interface correctly, it doesn't use the reported buses properly, hence it doesn't work with surround capable plugins. Therefore you need to use VST version, which reports only stereo capabilities by default.

To route **MIDI** to the plugin, create a new MIDI track and in the 'MIDI to' menu, choose the audio track that has the plugin on it. Note that in Live only the first plugin on any track can receive MIDI.

ProTools

In the mixer click an empty slot to insert the plugin to and select the plugin from the tree. The plugin may be present multiple times, once for each channel configuration (mono->stereo etc.). As of now ProTools do not arrange them in the subfolders, which is a workflow dealbreaker, but we cannot do anything about it. The huge empty space on top of each plugin window, which occupies so much of the precious display area, is part of ProTools and every plugin window and again we cannot do anything about it. In some cases you may experience CPU overload messages, in which case please contact Avid for support. Note that ProTools 10 and newer is supported. RTAS compatibility for PT9 and older will never be added.

To route an audio to the plugin's **side-chain** (if it has one), open the plugin, click on the *No key input* button in the plugin title and select the bus you want the audio taken from. You might need to remember the bus number, unless your ProTools version supports bus renaming. ProTools doesn't support stereo (or surround) side-chains at all.

To route **MIDI** to the plugin, create a new MIDI track and in the mixer click the output field for that track and select the plugin, which should already be in the menu.

FL Studio

First make sure plugins are scanned, either a full scan through the Plugin Manager or an automatic fast scan when you open the Plugin Database section of the browser in FL. The scanned plugins will show up in the Plugin Database > Installed section of the FL browser. The Effects and Generators sections in the Plugin Database will show all "favorite" plugins. These can be checked and

unchecked in the Plugin Manager or added in some other ways. These favorites also show up in the Add menu, the menu for the "+" button in the channel rack, when you right click an existing channel button to replace or insert, in the plugin slot menu in the mixer and in the plugin picker (F8). The menus with favorite plugins also have a "More" choice that will show all scanned plugins. The full explanation is in our help file, on the page **Installing Plugins**.

To route an audio to the plugin's **side-chain**, first set up the mixer: make sure the track you want to receive audio from is sent to the track the plugin as a sidechain (**help**). Then set up the plugin wrapper: choose the desired input on the **Processing tab** of the wrapper options.

To route **MIDI notes** to the plugin, first configure the sender: choose a MIDI port for the input device in the MIDI settings (for a hardware device), or an output port in the **wrapper options** (for a VST plugin that produces MIDI). For the receiving plugin, set the input port in the wrapper options to the same value you chose in step 1.

To route **MIDI controllers**, the procedure is different. The usual method in FL is to link CC messages to plugin parameters (**help file**). VST plugins will also have 128 CC parameters published (through the wrapper) that can be linkes this way. Those will send the specified CC MIDI message to the plugin, instead of changing a published parameter.

GUI styles, editor modes and colors

MeldaProduction plugins provide a state of the art styling engine, which lets you change the appearance to your liking. The first time you run the plugins a style wizard will appear and let you choose the style and other settings. It may not be available in ProTools and other problematic hosts.

By default each plugin has a certain color scheme, which differs based on what kind of plugin is that. Also, sections of some plugins are colorized differently, again, based on what kind of section is that (this can be disabled in global settings). Despite you can change the colors anyhow you want, it is advantageous to keep the defaults as these are standardized and have predefined meaning, so just by looking at a plugin's color you can immediately say what kind of plugin and section is that. Same rules apply when designing devices for easy screens. This is the current set of colors:

Dynamics = orange
Equalization, filtering = green
Reverb, delay = brown/yellow
Modulation = blue
Distortion, limiting = red
Stereo = cyan/yellow
Time, pitch, unison... = purple/pink
Tools = grey

Special colors:
Synchronization = grey
Detection = blue/green
Side-chain = green
Effects = red
Advanced stuff = grey

MELDA production
The only limit is your imagination

About MeldaProduction

The best sound on the market, incredible workflow and versatility beyond your imagination. We create the deepest and the most powerful audio plugins with unbelievable sound and tons of unique features you cannot find anywhere else.

Innovative Thinking

At MeldaProduction, we make the most advanced tools for music production and audio processing. We get inspired by the whole range of tools from the ancient analog gear to the newest digital creations, but we always push forward.
We've always felt the audio industry is extremely conservative, still relying on the prehistoric equipment making the job unnecessarily slow and complicated. That's why we invent new technologies, which make audio processing easier, faster, better sounding and more creative.

Sound Matters

In the world full of audiophiles you just need superb audio quality. And that's why we spend so much time perfecting audio algorithms until they sound unbeatable. Everything from dynamic filters to spectral dynamic processing. Our technologies just sound perfect.

Inspiring User Interface

Modern user interfaces must not only be easy and quick to use, but also versatile and the whole visual appearance should inspire you. MeldaProduction plugins provide the most advanced GUI engine on the market. It is still the first and only GUI engine, which is freely resizable and stylable. Our plugins can look as an ancient vintage gear, if you are working on old-school rock music. Or as super-modern futuristic devices if you are working on modern electronic music.

Easy to Use, Yet Versatile

The only limit is your imagination. Our plugins are with absolutely no doubt the most powerful and versatile tools on the market. Yet we managed to make the plugins easy to use via the devices and smart randomization system. But when you are ready, you are one click away from the endless potential the plugins provide.

Never-Ending Improvements

Most companies create a plugin, sell it and abandon it. We improve our plugins, add features, optimize... until there is nothing left to improve and there are no more ideas. Unfortunately that hasn't happened yet :). And the best thing is that the updates are free-for-life!

MeldaProduction was founded in 2009 by Vojtech Meluzin and is based in Prague, Czech Republic.

www.meldaproduction.com
info@meldaproduction.com
MeldaProduction (c) 2017