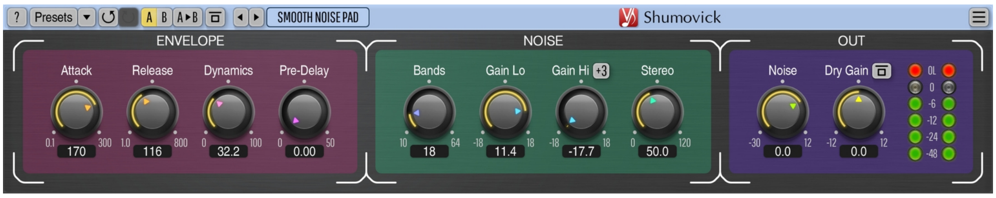
Voxengo
Sound with a soul

Voxengo Shumovick User Guide



Attack 170
Release 116
Dynamics 32.2
Pre-Delay 0.00
Bands 18
Gain Lo 11.4
Gain Hi +3 -17.7
Stereo 50.0
Noise 0.0
Dry Gain 0.0

Version 2.2

https://www.voxengo.com/product/shumovick/

Contents

Introduction

Shumovick plug-in for professional music production applications produces a creative dynamic noise-padding effect. This effect is most effective on beats and synth sounds used in electronic music production: EDM, hip-hop, and many others. The noise-padding effect created by this plug-in is correlated to the spectral content of the sound being processed.

As a result, on beats, this creates mellow noise bursts that make bassdrum sounds a lot softer, with a bit of stereo-field enhancement effect. On synths, on the other hand, such noise-padding effect may boost the power and density of the sound. Shumovick can be also effective when used on sterile-sounding basslines where it can add a bit of “pillowy” noise background. In practice, Shumovick can be also used on acoustic drum buses adding “vintage” feel and aggression to them. When used on a full mix lightly, Shumovick can add a slight “vintage” feel to the sound, make it slightly rougher, and add a stereo-widening effect: this is reminiscent of what a reel-to-reel tape and vinyl recording media does to the sound, naturally.

Shumovick is a relatively CPU-demanding effect plug-in, so it requires a higher-end computer processor for a comfortable use. However, the CPU requirements can be lowered by adjusting plug-in’s processing band count.

Features

- Noise-padding signal envelope controls
- Noise-padding signal equalization
- Selectable processing band count
- 64-bit floating point processing
- Preset manager
- Undo/redo history
- A/B comparisons
- Contextual hint messages
- All sample rates support
- Zero processing latency

Compatibility

This audio plug-in can be loaded into any audio host application that conforms to the AAX, AudioUnit, VST, or VST3 plug-in specification.

This plug-in is compatible with Windows (32- and 64-bit Windows XP, Vista, 7, 8, 10 and later versions, if not announced otherwise) and macOS (10.11 and later versions, if not announced otherwise, 64-bit Intel and Apple Silicon processor-based) computers (2.5 GHz dual-core or faster processor with at least 4 GB of system RAM required). A separate binary distribution file is available for each target computer platform and audio plug-in specification.

User Interface Elements

Note: All Voxengo plug-ins feature a highly consistent user interface. Most interface elements (buttons, labels) located at the top of the user interface are the same in all Voxengo plug-ins. For an in-depth description of these and other standard features, and user interface elements, please refer to the “Voxengo Primary User Guide”.

Envelope

This group of knobs affects envelope characteristics used to shape the internal noise signal sent to output. The algorithm derives envelope from the input spectral signal.

The “Attack” parameter specifies the attack time of the envelope signal, in milliseconds. Lower values produce an envelope with sharp transients – useful to react on high-hats, for example. Higher values reduce transient response, this can be useful when smooth noise “padding” is suitable for the source signal.

The “Release” parameter specifies the release time of the envelope signal, in milliseconds. Lower values produce an envelope with fast fall time, with a bit of “stuttering” flavor. Higher values produce a longer fall time, but this may have an effect of “noise overcrowding”.

The “Dynamics” parameter adjusts the dynamic response of the envelope signal. This parameter works as either a compressor of dynamics (lower values) or an expander of dynamics (higher values).

The “Pre-Delay” parameter allows you to delay the noise signal. Delaying can be used to make the overall sound smoother, by decorrelating the envelope from the signal. It is also useful for achieving a stronger stereo-widening effect.

Noise

This group of knobs affects noise signal’s spectral shape. These controls work like a Baxandall equalizer.

The “Bands” parameter controls the number of bands the noise signal is split into. This parameter affects both the CPU resources required for effect processing and the overall sonic quality of the produced noise signal. Note that this parameter also slightly affects loudness of the noise, in a non-linear way.

The “Gain Lo” parameter controls the gain of lower frequencies of the noise signal.

The “Gain Hi” parameter controls the gain of higher frequencies of the noise signal. Lowering this value usually allows you to get a smoother-sounding result. The “+3/0/-3” selector chooses between different spectral slopes (in decibel per octave), affecting the “brightness” of the noise signal.

The “Stereo” parameter adjusts “stereo width” of the noise signal, with 0 producing a “mono effect”, 100 being “no width change”, and 120 being “slightly anti-phase”. This is a simple mid-side balance parameter.

Out

The “Noise” parameter controls the loudness of the noise signal.

The “Dry Gain” parameter controls the loudness of the dry input signal. The “Mute” switch can be used to disable the dry signal in the output mix. Muting the dry signal permits use of the plug-in as a “send” effect. Note that the “Mute” switch is “persistent”: its state is not affected by preset loading using plug-in's preset features.

Both knobs can be changed synchronously with the right mouse button or holding down the Ctrl (Cmd on macOS) key.

This block also displays the output signal level in decibel. “OL” indicator reacts on signal levels above 0 dBFS.

Credits

DSP algorithms, internal signal routing code, user interface layout by Aleksey Vaneev.

Graphics user interface code by Vladimir Stolypko. Graphics elements by Vladimir Stolypko and Scott Kane.

This plug-in is implemented in multi-platform C++ code form and uses “zlib” compression library (written by Jean-loup Gailly and Mark Adler), “LZ4” compression library by Yann Collet, “base64” code by Jouni Malinen, filter design equations by Magnus Jonsson and Robert Bristow-Johnson, VST plug-in technology by Steinberg, AudioUnit plug-in SDK by Apple, Inc., AAX plug-in SDK by Avid Technology, Inc., Intel IPP and run-time library by Intel Corporation (used under the corresponding licenses granted by these parties).