

([<--- back to main-page](#))



JELSTUDIO's "Otto"

Hi, I'm the author of Otto. I will talk a bit about where this plugin may come in handy and a bit about how it works.

Usage (single sound file):

Let's say you've been to a concert or live performance and recorded with a hand-held device (perhaps a wave-recorder, a video-camera, or a smart-phone). Maybe you moved around in the crowd, or moved the device around to film different things. The end-result may be a piece of audio that isn't rock-solid, but has shifting stereo-image and volume.

Running such a track through Otto may stabilize it and make it sound more coherent and solid. I actually saved a live music-performance track (a rock-band performing live at a small open-air Christmas-festival at a town-square) myself during testing and fine-tuning of the plugin. The actual recording wasn't useable, with the sound-volume being all over the place, but running it through Otto glued the whole thing together. So for such a specific situation, Otto has in fact already been trialed by fire and passed.

It could also be an interview you've made with a stereo-mic, where one person is either shifting the volume of their speech during the interview, or perhaps there's a stereo-imbalance (a volume-difference between left and right audio-channel) between interviewer and interviewee. In such a case Otto can bring things closer to a uniform level.

A more exotic use of Otto might be to use it to make the viewing of certain modern movies more enjoyable. If you've ever watched a movie on your home-system and been annoyed at how loud the sound-effects are compared to the dialogue, you could use Otto to Equalize speech and sound-fx more so you don't have to ride the remote-control's volume-knob through the whole film. I call this use 'exotic' because VST support is not quite common in movie-players yet.

Usage (multiple sound files):

Say you're loading up a series of different multi-track sessions in your DAW to preview. If you have an instance of Otto on each track that you load sound-files on to, then each multi-track session will be quickly peak-leveled to -6 dBFS, giving a much more consistent sound between different sessions.

If you like the sound of Otto enough on the material you load into your DAW tracks, then you could also use it as a quick way to create solidified demos. Then you could save the work of setting up manual volume- and balance-envelopes and compressor- and limiter-work to when you do the final high-grade

master.

You could also use Otto during tracking, as it is zero-latency, if your talent needs a headphone-mix with less dynamics than is recorded.

Or if you need to send a quick-mix to a sound-booth or back-stage for simple monitoring, then Otto can be used to keep the sound under tighter constraints and catch unwanted drastic shifts in sound-levels that a sound-engineer may miss in the heat of a live-moment.

Usage (things to be aware of):

Otto has no user-controls. It is fully automatic and always ON when activated.

Otto syncs to your audio-engine's output sample-rate. Audio-files should match this sample-rate for optimal results (if sound-quality is less important you can ignore this). Mixing different sample-rates in the same session is not recommended. In other words; if your audio-files are sampled at 44.1kHz it is recommended to also run your audio-engine at 44.1kHz. If you find yourself with a session involving files with multiple different sample-rates (perhaps some are sampled at 44, others at 96, etc), it is recommended to re-sample these files to the same sample-rate you run your audio-engine at. So if you run your audio-engine at 96, re-sample all files to 96, etc.

All bit-depths (up to 64) are handled by Otto and need not be matched. It's OK to mix files with different bit-depths in the same session (say, 16 with 24, for example)

There is no DRM built into the plugin itself, so every CPU-cycle it uses goes to treatment of your audio. This also means you can safely move a session between home-systems, studio-systems and live situations (the same way a keyboard, guitar, microphone, etc, would work whether at home, in the studio or in a live situation)

Otto is best used as the first plugin in your processing chain.

Heavily pre-compressed audio can be problematic as subtle distortions may become very prominent sounding if run through Otto. Examples of heavily pre-compressed audio can be some types of radio-speak or news-television speak.

You can use Otto on mono and stereo audio, but if your audio has deliberate stereo-imbalance (an imbalance made on purpose or a natural imbalance you wish to keep), Otto may break this or minimize it. You can of course introduce stereo-imbalance later in your processing chain.

Technical:

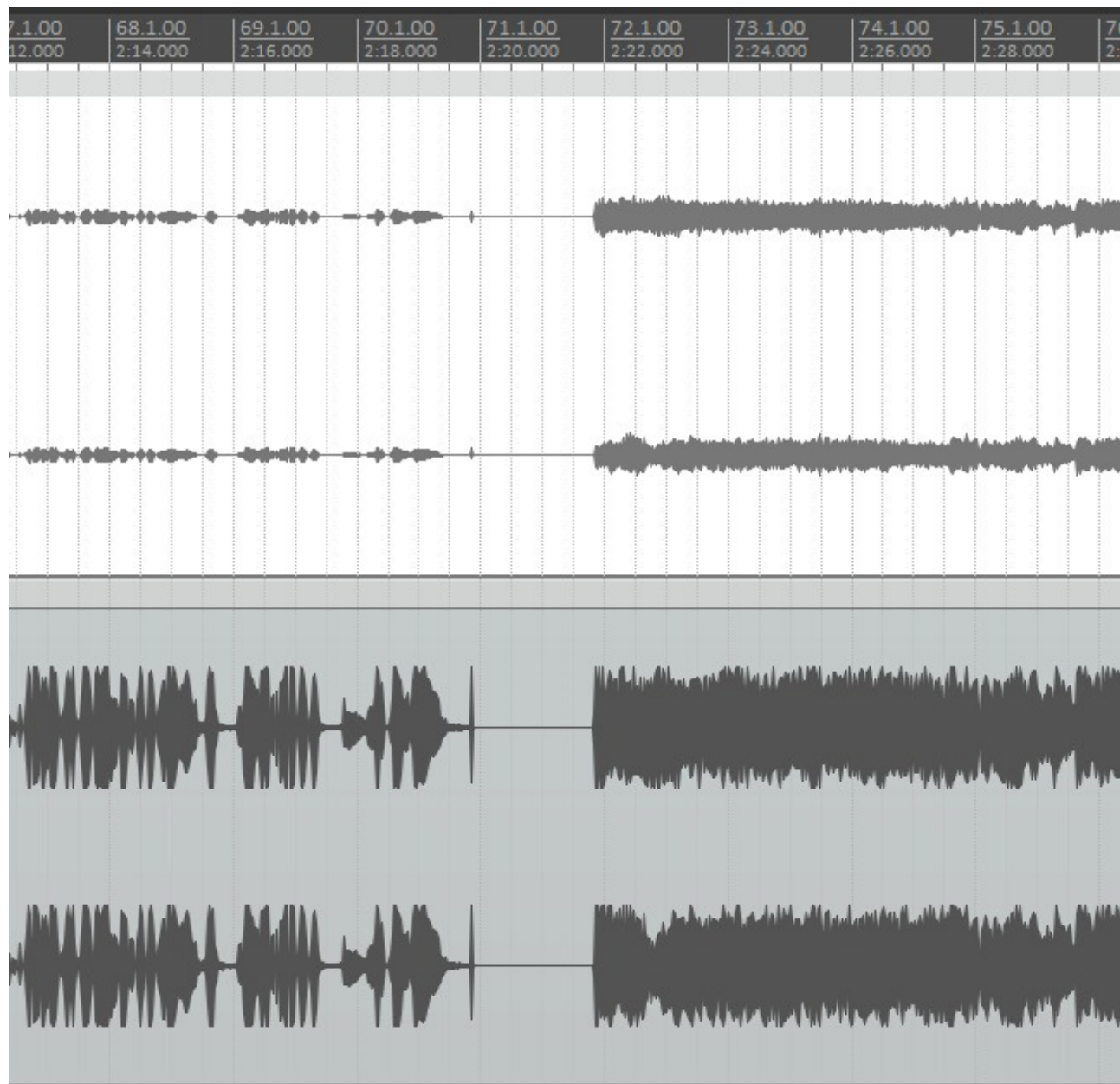
Otto is a zero-latency dynamic wave-shaper. It shapes the amplitude, not the frequency. It works in both positive and negative wave-space (think push and pull of a speaker). This sets it apart from more traditional dynamic gain-systems and auto-levelers. A traditional auto-gain does not change the relativity between the DC-offset and the peaks, it just changes the distance between them by 'inflating' from the center (the zero-cross point). Otto in fact breaks the relativity between the DC-offset and the peaks. The first effect this has is that DC-offset is canceled. The second effect is that the dynamic range is maximized. Let's visualize this by going back to the push/pull of a speaker; since Otto cancels DC-offset, the waveform's center will match the speaker's natural resting-position. A sound with incorrect DC-offset will keep electrical current running through the speaker in such a way that the waveform's center is not

electrically neutral in the speaker. When that happens the distance between max push and max pull is not equal, leading to less possible dynamic range. A traditional DC-offset plugin will move the waveform, but not re-shape it. Effectively this keeps the lower dynamic range in the audio, by keeping the push/pull imbalance (the sound may push more than it pulls, or pull more than it pushes. Either way it will still be out of balance). Otto cancels DC-offset by re-shaping the waveform, rather than moving it. This, however, can lead to some distortion, depending on the sound-material being processed.

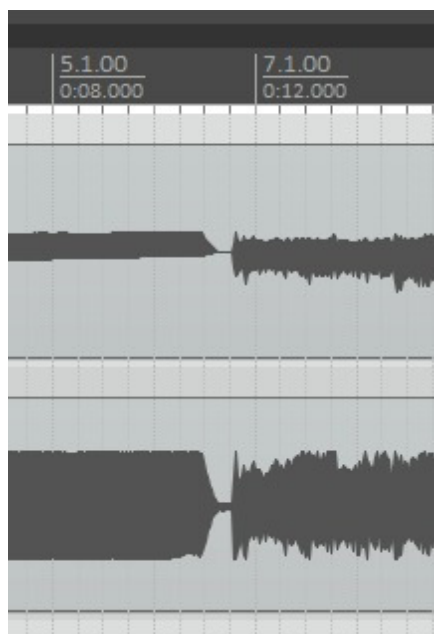
A few images showing waveforms before and after Otto:

The top waveform is the original audio and the bottom is processed with Otto. The first half is speak at lower volume, the middle is 2 seconds of silence, and the second half is music at higher volume. The numbers on the top are bars and time (minutes and seconds)

As you can see; the waveform processed with Otto peaks at -6dBFS for both the speak at lower volume and the music at higher volume, without destroying the 2 seconds of silence in the middle. It's relevant to understand that Otto does not just raise the volume and make things louder, as a traditional auto-leveler might do, Otto re-shapes the audio which helps avoid silent parts in audio from becoming pumped and excessively noisy.



In this image we see a mono waveform from an old multi-track tape-recording where 2 instruments were punched in and out on the same single track. The first half is one synth and the second half is a different synth. It's obvious that these 2 synths have very different DC-offsets (both are old analog synths). The first is almost entirely in the upper-half of the waveform, while the second synth is a bit more well-behaved. This leads to some severe and unwanted speaker-movements and thus bad sound. The bottom waveform is after Otto, where the change is easy to spot. Here both synths are now well-behaved and use the waveform-space (and thus speaker-movement) optimally.



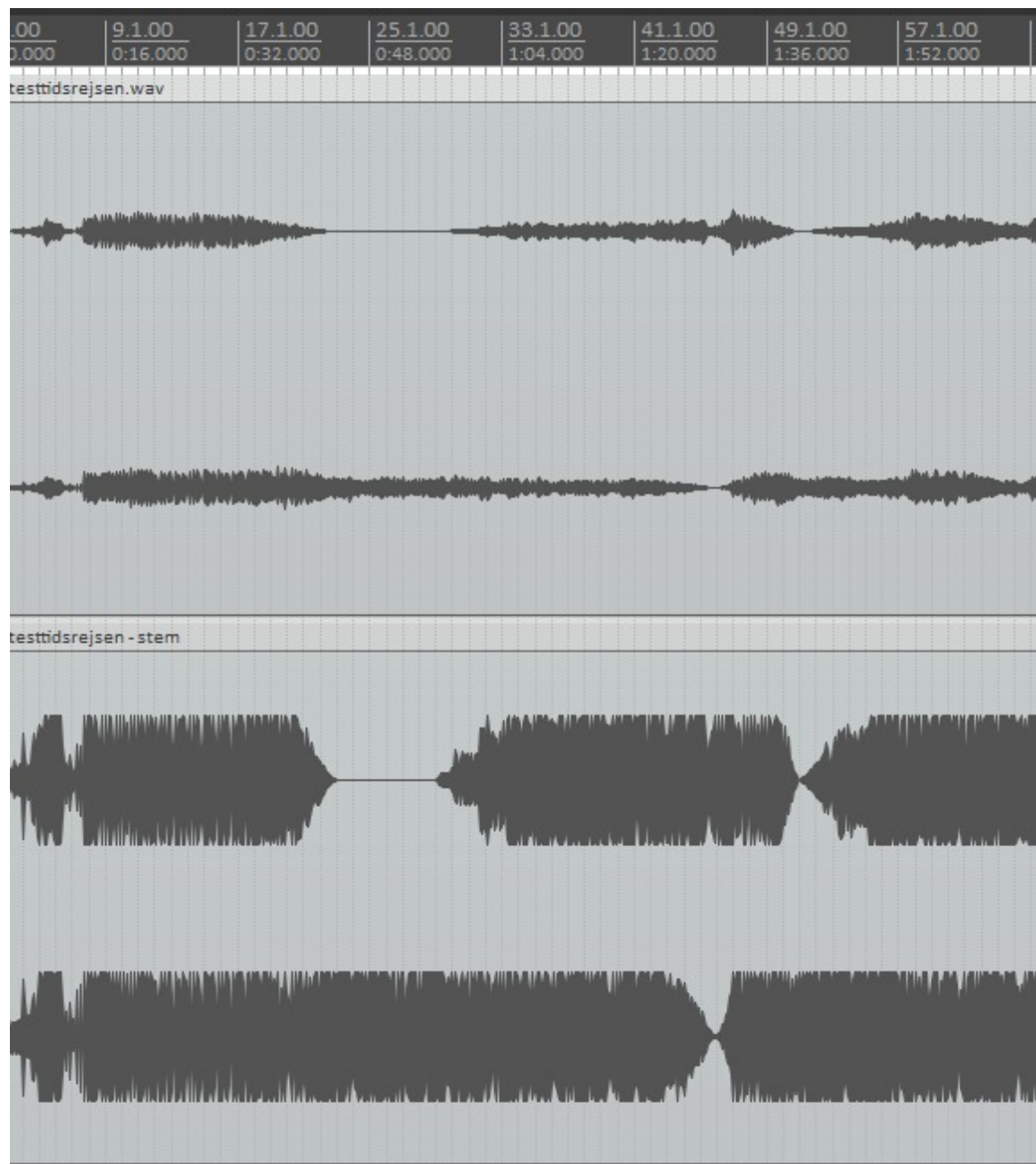
This image is also from an old multi-track tape-recording. Again it's a mono-waveform, this time of a bass. The upper waveform is the original sound from the tape. It has peak-balance issues, even though its DC-offset is actually nicely centered. Playing this sound through a speaker would either push or pull more (depending on phase-settings) and lead to less than optimal sound. The bottom waveform is after it has been through Otto, which has balanced the waveform so it now extends equally from the center-line in both positive and negative waveform-space. When this sound is played through a speaker, it will push and pull naturally.



This image is of a stereo-track of a piece of music. On the original piece (again the upper half of the

image) a hard-pan was introduced. The sound starts balanced, then is panned hard-right (at ~48 seconds in), then re-balanced (at ~1 minute and 4 seconds in), then panned hard-left and straight to hard-right (at ~1 minute and 36 seconds in), then re-balanced (at ~1 minute and 52 seconds in).

The bottom half has been through Otto, and peaks at -6dBFS while hard-pans are retained. The panning sounds steeper/faster, because the total volume change is larger, but still smooth/even in panning-tempo (it's not an abrupt cut-off of sound, just quicker panning movement)



History:

I'm a hobby-musician and this is the first serious audio-plugin I've coded. My coding-experience mainly involve flight-simulation systems (autopilots and guidance-systems, cockpit-computers and displays, vehicle sub-systems, etc).

The name Otto comes from the most famous auto-pilot in the world of movie-making, the inflatable; "Otto the autopilot" from the movie "Airplane".

Otto actually started as a de-orbit guidance auto-pilot I coded for the Orbiter (the space-shuttle simulator).

It was the first true-modular auto-pilot (true-modular in the sense that it was not hard-coded into the vehicle-simulation, but was completely independent).

Otto later evolved into an advanced true-dual system civil jet-liner aviation aircraft-autopilot (with 'first-of-its-kind' automatic emergency safety-procedures that didn't require pilot-crew interaction) that can run on Lockheed Martin's simulator, and has now, in its current iteration here, also reached the level of music-autopilot (or rather a waveform-autopilot). The only thing these different Ottos have in common is really only the name, as the code-base is obviously different for each system. But it's a fun little anecdote, for me as the author at least.

end-of-file