AUTOCUE

Liquidshop 4 Presentation

Matthias C. Hormann (“Moonbase59”)

2024-05-27

NAVIGATION

- space – next slide
- arrow keys ←→↓↑ – navigate
- Shift+↑↓ – jump to top/bottom of a topic
- Esc – overview
- Ctrl+Mouseclick – zoom in/out
- Home, End – jump to start/end

AUTOCUE

On-the-fly JSON song cue-in, cue-out, overlay, replaygain calculation for Liquidsoap, AzuraCast and other AutoDJ software.

Phew! Now what does *that* mean?

BETTER SONG TRANSITIONS FOR RADIO AUTOMATION (“AUTODJ”).

- Remove silence at start & end of tracks.
- Find ideal point for starting the next track.
- Auto loudness correction (“ReplayGain”).
- Based on loudness perception of the *human ear*, not simple dB, amplitude or RMS values.
- Can work *on-the-fly*, without pre-processed files.

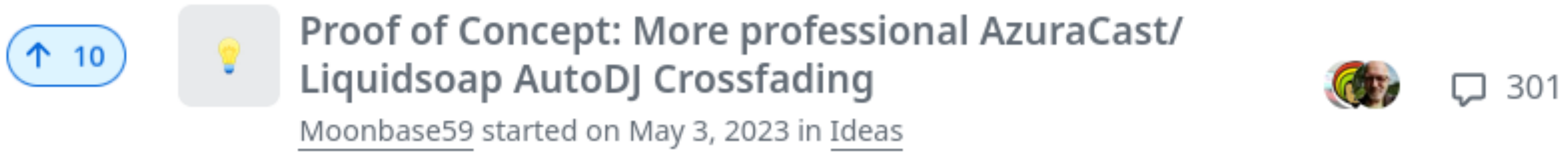
AND MORE...

- Keep songs with long endings intact.
- Skip silence within songs (“hidden tracks”).
- Clipping prevention.
- Use file *tags* for less CPU & higher speed.
- Follows EBU (European Broadcasting Union) recommendations.

THE STORY

USER DEMAND

On the AzuraCast GitHub, the "Professional Crossfade" thread is the single most active discussion.


↑ 10

Proof of Concept: More professional AzuraCast/ Liquidsoap AutoDJ Crossfading

Moonbase59 started on May 3, 2023 in Ideas

301

There is a huge user demand for radio-like, more professional song transitions!

PREVIOUS WORK

John Warburton ("Warblefly"), an industry professional and Tonmeister, already talked about "Easing automation and improving your sound with Liquidsoap and FFmpeg" in 2021 (Liquidshop 1).

He also made available his pre-processing and playlist annotation scripts. Thanks for sharing, John!

CUE_FILE

Inspired by John’s work, I started writing `cue_file` in early February 2024, as a proof of concept, to see if “on-the-fly” processing could be done.

`cue_file` is a Python3 script, that in turn uses *ffmpeg* and *ffprobe* to analyse an audio file for cueing and transition data, based on the loudness perception of the human ear. It uses the EBU R.128 algorithms and returns JSON data.

LIQUIDSOAP INTEGRATION

Many talks and tests with **RM-FM**, **toots** and **Stefan** (gAlleb) brought up two solutions:

- RM-FM and toots worked on an “all-Liquidsoap” approach.
- I favoured and worked on the “external” solution, for more flexibility and pre-processing purposes.
- toots came up with a Liquidsoap integration API for both variants.

TWO INTEGRATIONS

AUTOCUE.INTERNAL

- No external dependencies (apart from ffmpeg).
- Easy to use.
- Made by **RM-FM** and **toots**.

AUTOCUE.CUE_FILE

- Requires *ffmpeg*, *ffprobe*, *Python3* and `cue_file`.
- `%include` or copy-paste (AzuraCast).
- Relatively easy to use, great defaults.
- Many additional features.
- Perfect for pre-processing.
- Can use file *tags* for dramatic speed increase.
- Made by **Moonbase59** and **toots**.

OVERVIEW

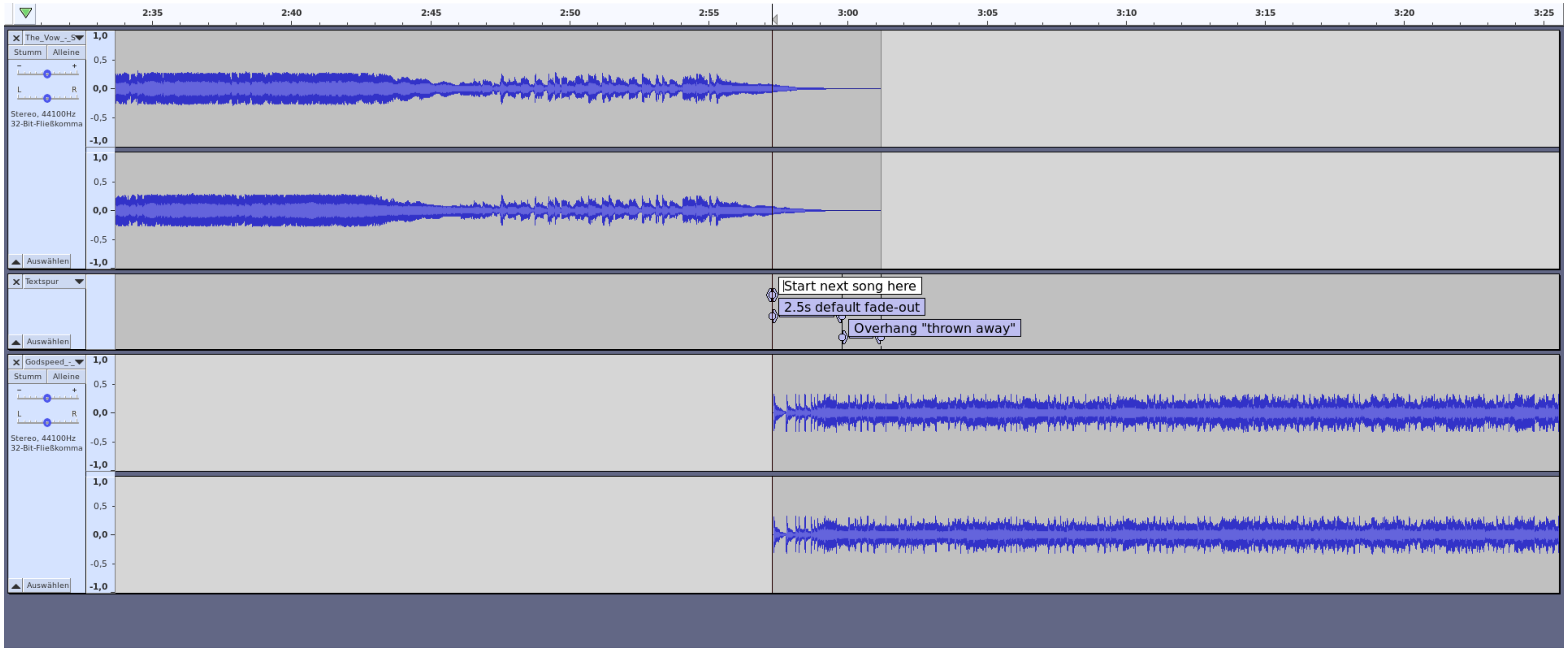
- Many concepts are similar in both implementations.
- Some differ.
- Both use the same integration API.
- It’s *your* choice.

In this presentation, we will concentrate on **autocue.cue_file**.

A REAL EXAMPLE

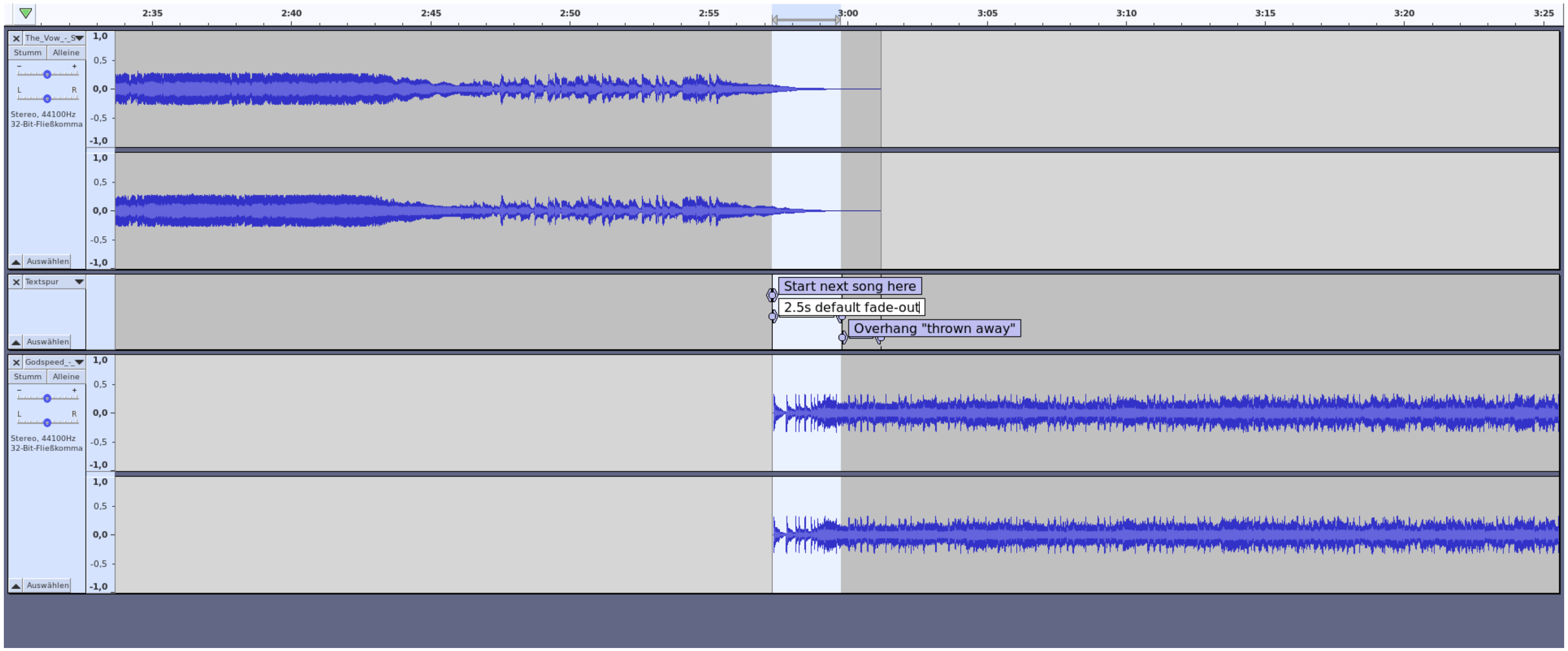
Let’s visualize what autocue does.

1. FIND START POINT FOR NEXT SONG



Long smooth endings will be kept intact.

2. DEFAULT FADE-OUT



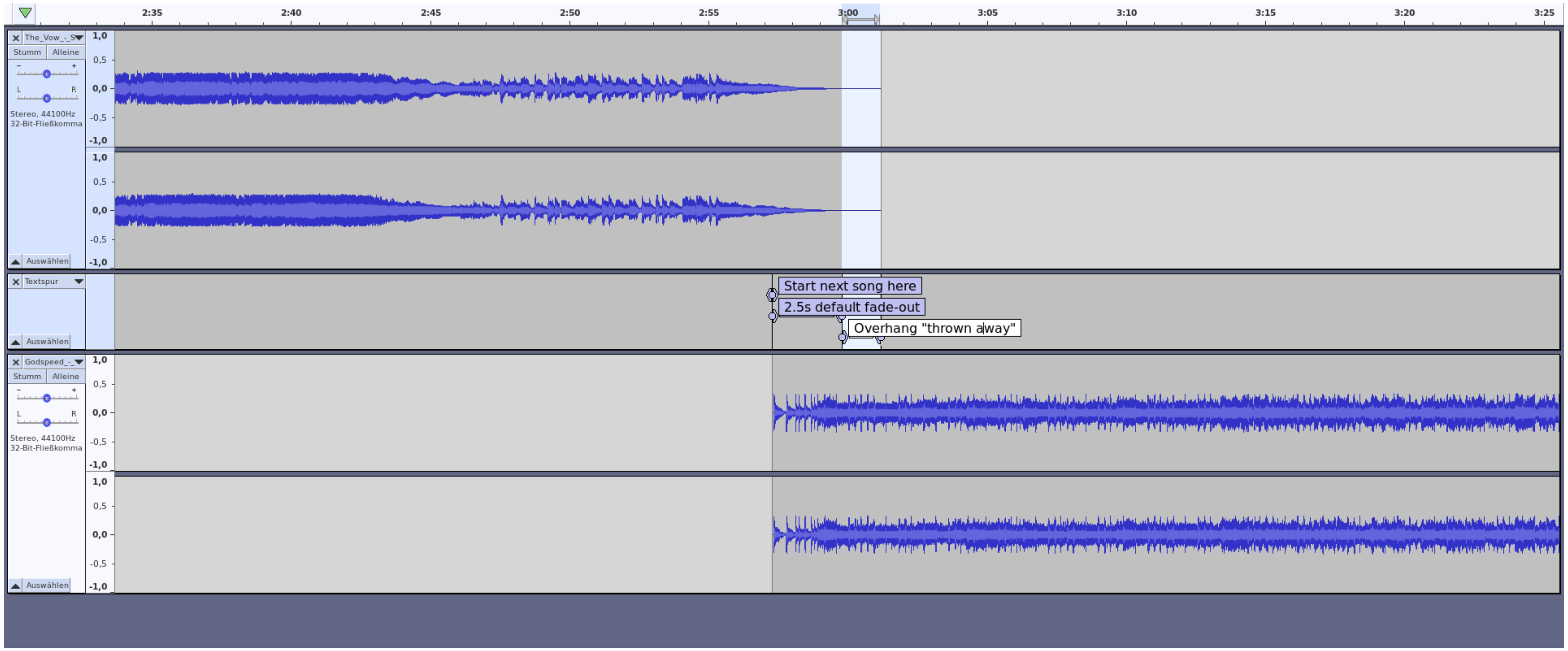
Start next song here

2.5s default fade-out

Overhang "thrown away"

Limits overlay length (too long sounds bad).

3. CUT OFF “OVERHANG”

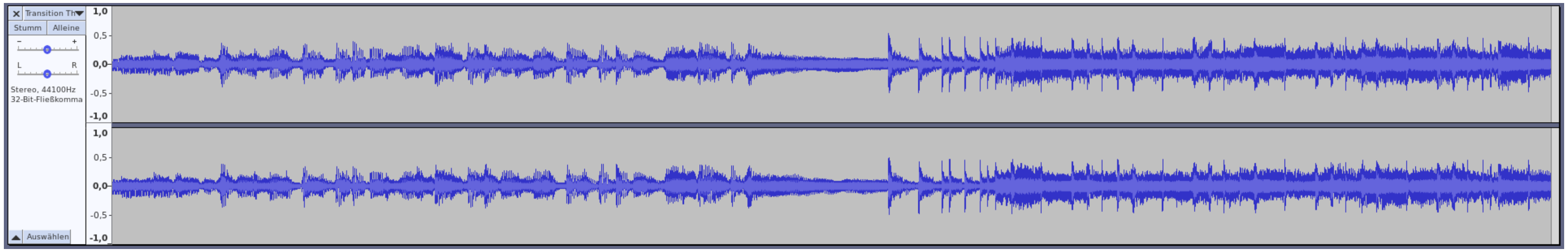


Start next song here

2.5s default fade-out

Overhang "thrown away"

THE RESULT



0:00 / 0:21

Smooth, continuous playout, radio-style.
And perfect transitions—everytime.

HOW DOES IT WORK?

Concepts, units, and inner workings explained.
With visual examples.

VOLUME VS LOUDNESS

- Often misunderstood, and complicated to explain correctly (volume, level, gain, amplitude, dB, SPL, RMS, VU, LUFS, LKFS, …)

Let’s make it easy and just say:

- **Volume** = *quantity* or *power* of a sound
- **Loudness** = human *perception* of sound
- Autocue works **loudness-based** (what you *hear*).

UNITS WE USE

- **Amplitude** (0.0 .. 1.0, silence to loudest, linear)
- **dB** (ratio between measurement and reference)
- **dBFS** (dB relative to full scale)
- **LU** (loudness units; 1 LU ≙ 1 dB)
- **LUFS** (loudness units relative to full scale)
- dBFS/LUFS scale (logarithmic):
  - 0.0 = loudest signal without distortion
  - -96.0 = digital silence for 16-bit audio data

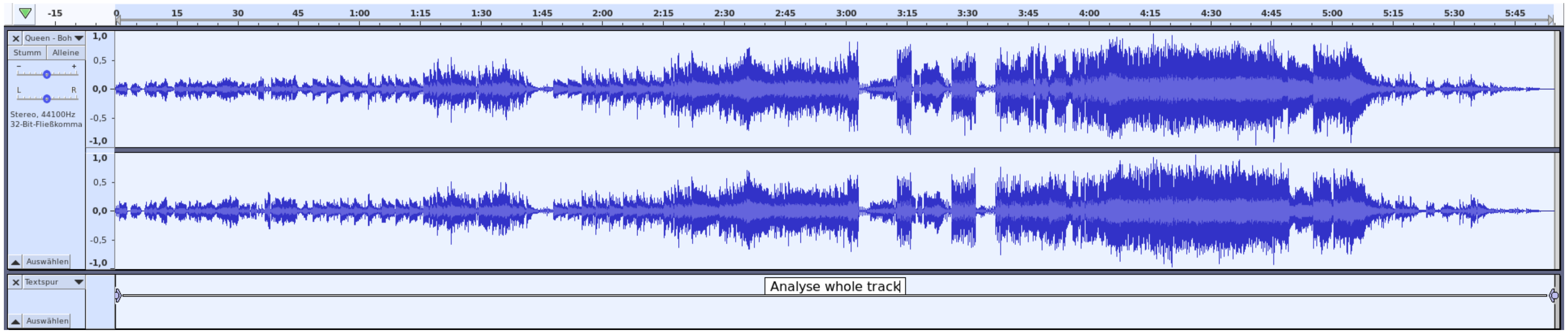
HOW MUCH IS “TWICE AS LOUD”?

+10 dB	2x *perceived* loudness (psycho-acoustics)	mostly *sensed*
+6 dB	2x sound pressure (RMS voltage, amplitude)	mostly *measured*
+3 dB	2x intensity (power, energy)	mostly *calculated*

We have to be *specific* in acoustics!

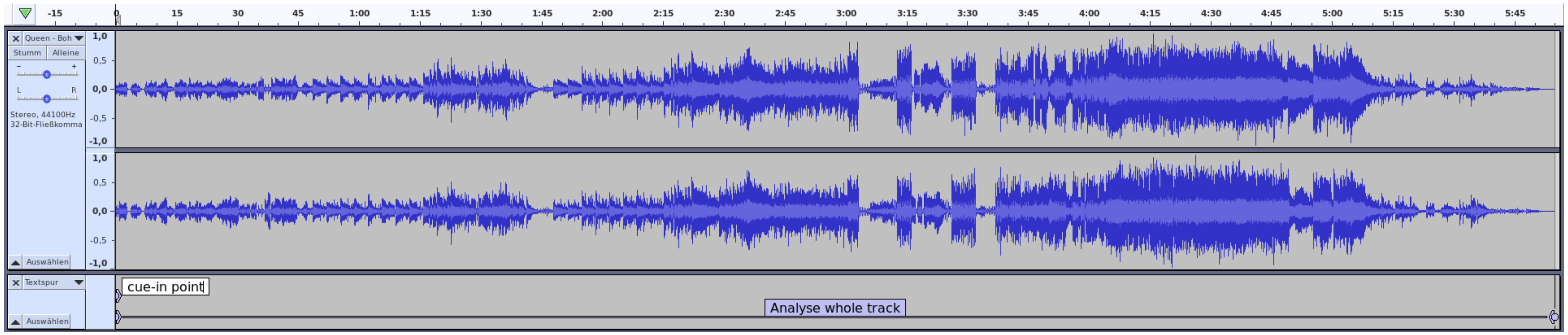
ANALYSING A TRACK

Queen: Bohemian Rhapsody



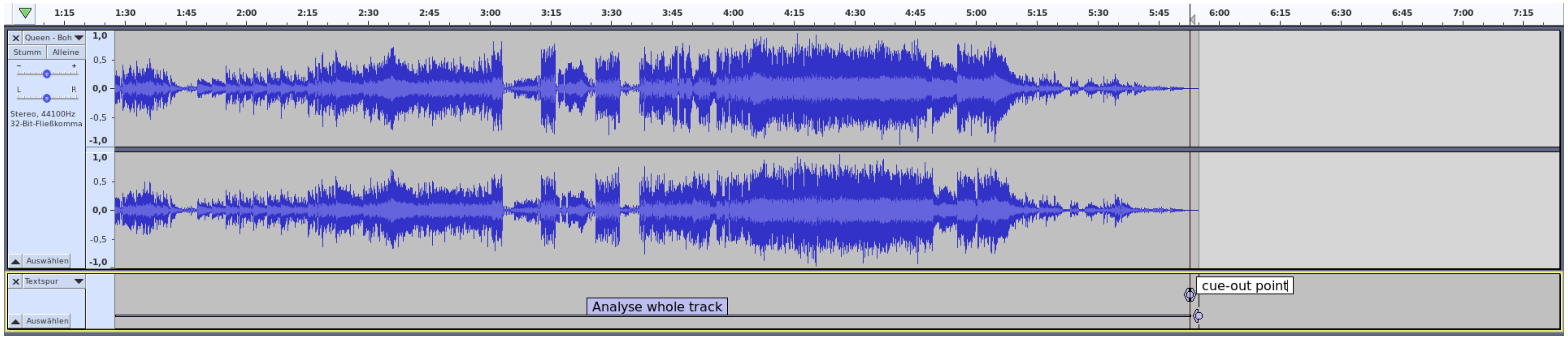
- Analyse whole track, measuring
  - momentary loudness of a 400 ms sliding window, every 100 ms
  - integrated loudness over total duration, using a noise gate
  - loudness range (dynamic range)
  - true peak, all channels, using oversampling
- Results in `liq_loudness`, `liq_loudness_range`, `liq_true_peak`

CUE-IN POINT



- Silence level: **-42 LU** referencing integrated track loudness. For a song with -18 LUFS loudness, the noise floor would thus be at -60 LUFS.
- `settings.autocue.cue_file.silence`
- Look *forward from the start*, find where momentary loudness goes above silence level.
- This is our cue-in point (`liq_cue_in`).

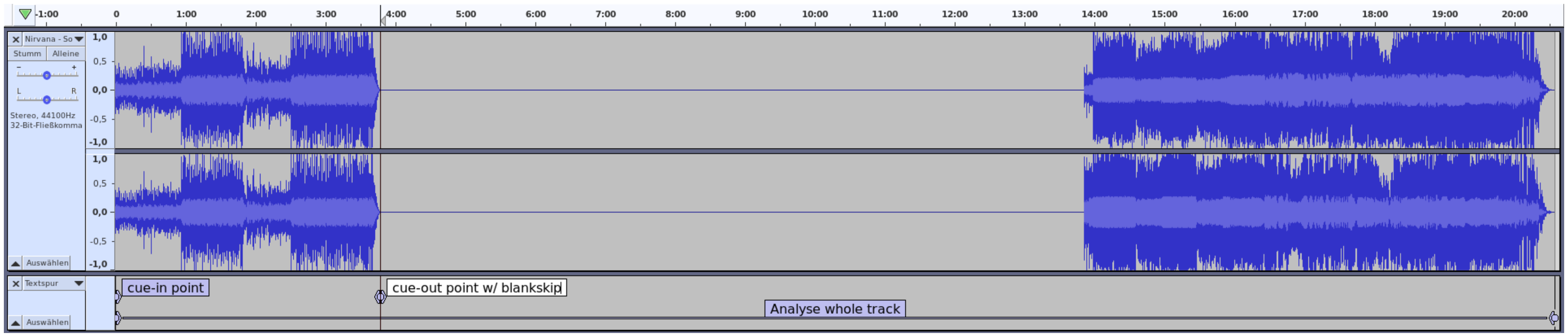
CUE-OUT POINT



- Look *backwards from the end*, find where momentary loudness goes above silence level.
- This is our cue-out point (`liq_cue_out`).

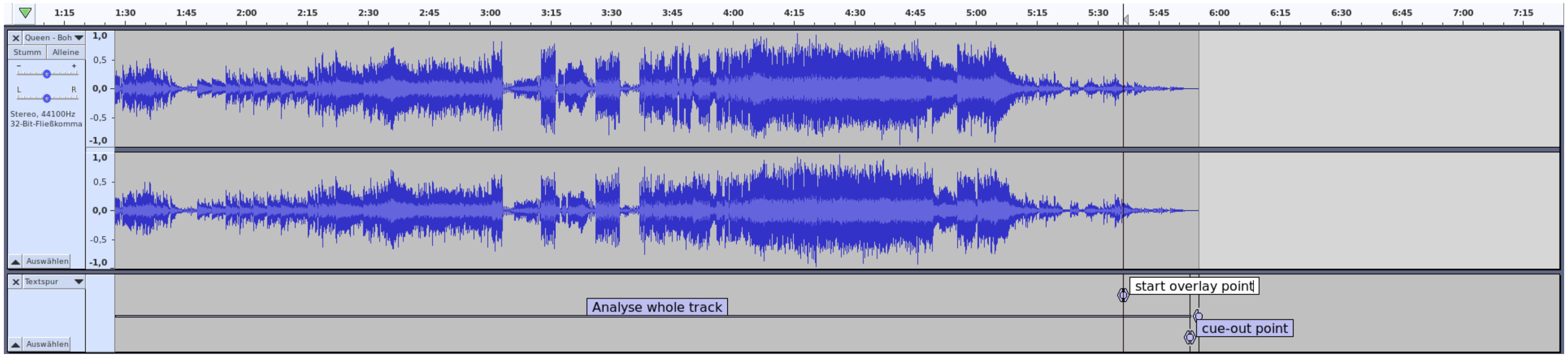
CUE-OUT POINT W/ BLANKSKIP

Nirvana: Something in the Way / Endless, Nameless



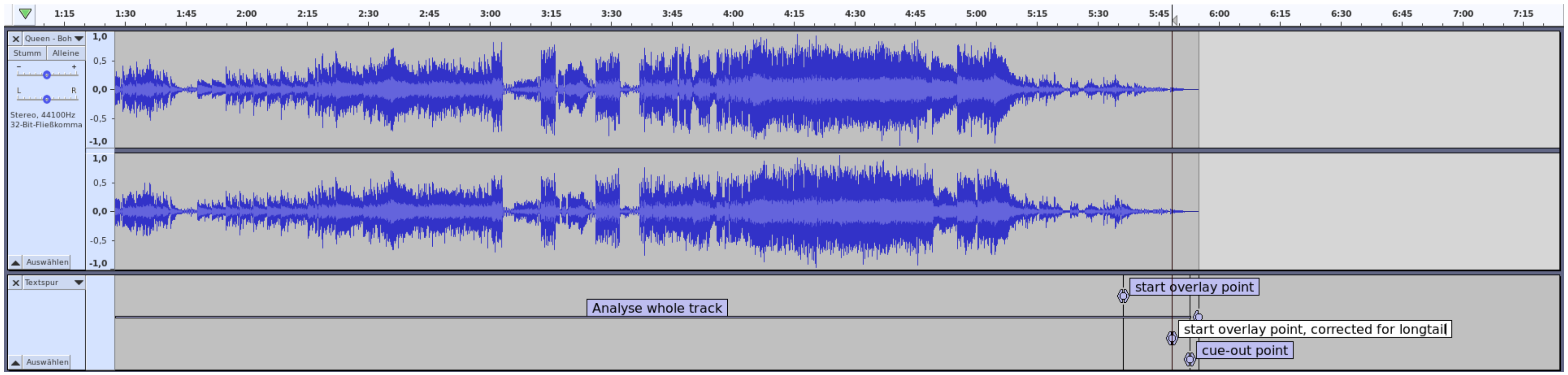
- Look *forward from cue-in*, find where momentary loudness goes below silence level.
- We're now cueing out *early* (at the start of the silent part in the song), avoiding "dead air" for songs with "hidden tracks".
- Results in `liq_cue_out`, `liq_blank_skipped`.

START OVERLAY POINT (NEXT SONG)



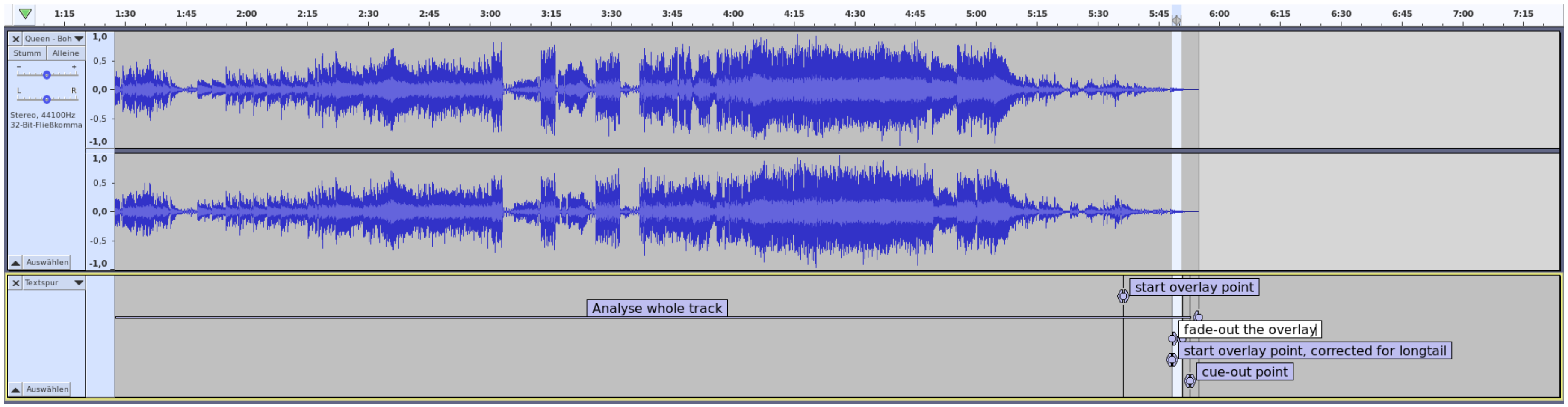
- Overlay level: **-8 LU** referencing integrated track loudness.
- `settings.autocue.cue_file.overlay`
- Look *backwards from cue-out*, find where momentary loudness goes above overlay level.
- This would be an ideal point to start the next song,
  but it *might* cut short important long song endings (as shown above).
- Result in `liq_cross_start_next`.

LONG TAILS



- Check if calculated overlay duration > **15 s** (a "long tail").
- `settings.autocue.cue_file.longtail`
- If so, reduce overlay level by an extra **-15 LU** and repeat the calculation.
- `settings.autocue.cue_file.overlay_longtail`
- We now start the next song much later, keeping the song's "long tail" intact!
- `liq_longtail` shows if a long tail was detected.

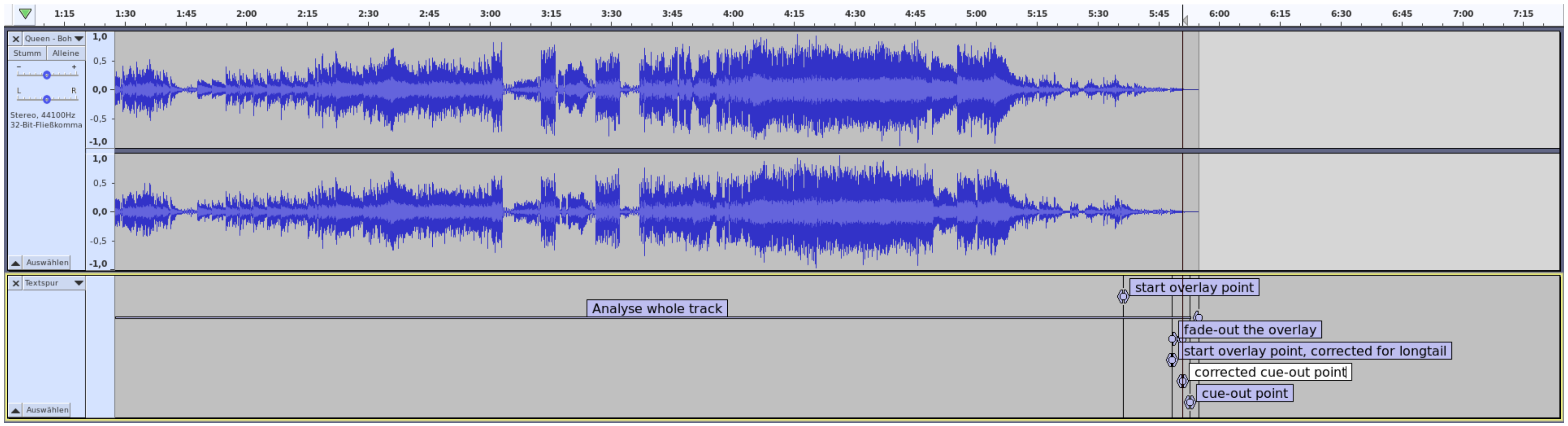
FADE-OUT



Analyse whole track
start overlay point
fade-out the overlay
start overlay point, corrected for longtail
cue-out point

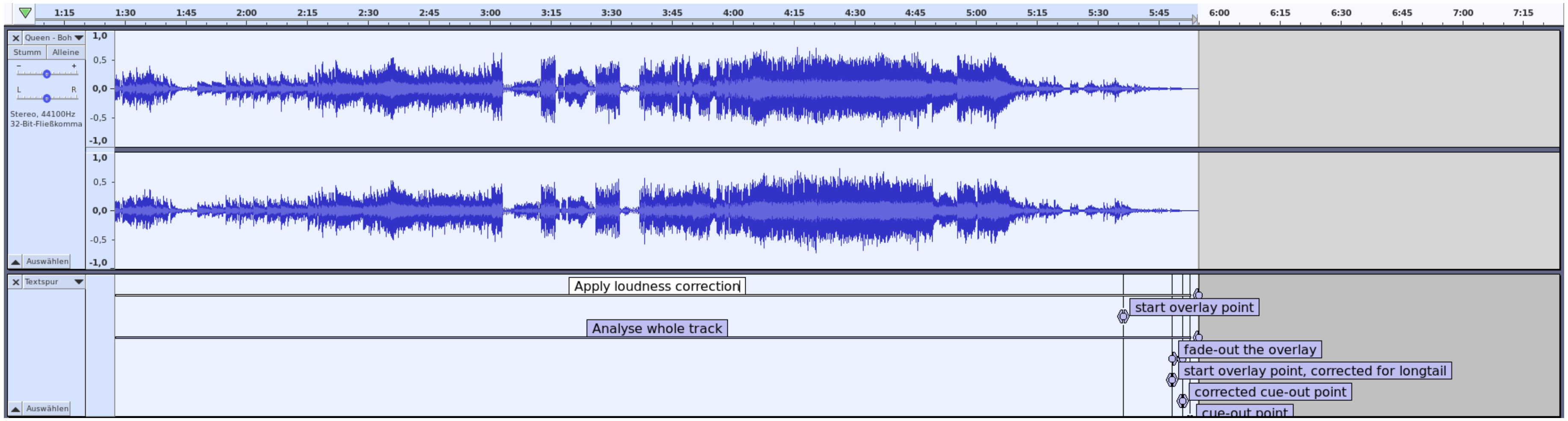
- Apply fade-out, so overlay isn't too long.
- `settings.autocue.cue_file.fade_out`
- Too long overlays sound bad, especially when a jingle follows.
- `liq_fade_out`

CORRECT CUE-OUT



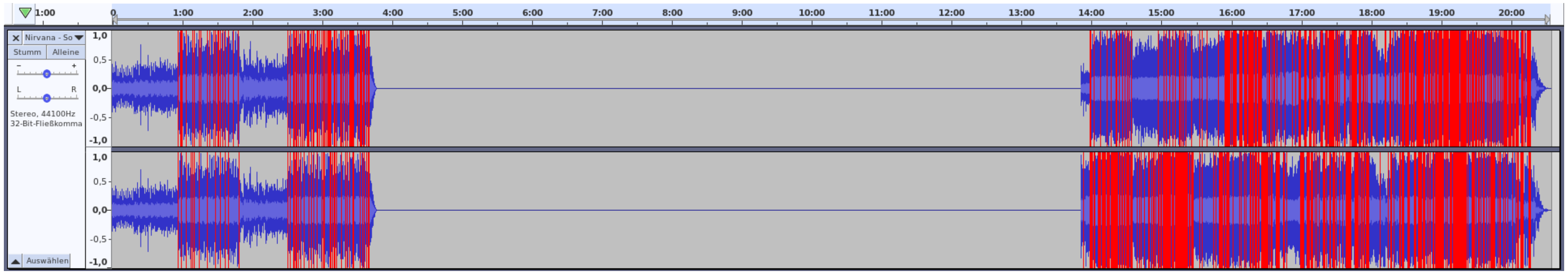
- Correct cue-out (overhang “thrown away”)

AMPLIFY & REPLAYGAIN



- From the integrated loudness of the track, and the desired loudness target, we can now calculate the *amplification* and *ReplayGain* values.
- `settings.autocue.cue_file.target`
- `settings.autocue.cue_file.unify_loudness_correction`
- Recommended loudness targets:
  - Europe: **-23 LUFS** (EBU) or **-18 LUFS** (EBU, "temporarily allowed")
  - U.S.: **-24 LUFS** (ATSC), not (yet?) supported in *ffmpeg*
- Results in `liq_amplify`, `liq_reference_loudness`, `replaygain_track_gain`, `replaygain_reference_loudness`.

CLIPPING PREVENTION



- Modern highly-compressed ("loudness war") music and file formats using lossy compression can easily *clip* (distort).
- To prevent this, both *loudgain* and *cue_file* can reduce the amplification/ReplayGain values, using the measured true peak values, so that the EBU-recommended -1 dBFS is not exceeded.
- `settings.autocue.cue_file.noclip`
- **Note:** This is just a loudness reduction, not a brickwall limiter or the like!
- Applied correction amount shown in `liq_amplify_adjustment`.

LET’S NOW MOVE TO REAL-LIFE USAGE

That’s much easier. Promised.

Because autocue does all the work for you.

A MINIMAL EXAMPLE

Using standalone Liquidsoap

PREPARATION

- Copy `cue_file` to appropriate location in the path. On Linux, this is usually one of these:
  - `~/bin`
  - `~/.local/bin`
  - `/usr/local/bin` (needs `sudo`)
- Ensure you have *Python3*, *ffmpeg* and *ffprobe* available. On almost all distros, these are pre-installed.

LIQUIDSOAP CODE

```
# Minimal example for the `autocue.cue_file` protocol.
# Uses one playlist and outputs to sound card.

%include "autocue.cue_file.liq"
enable_autocue_metadata()

radio = playlist("Classic Rock.m3u")

radio = amplify(1., override="liq_amplify", radio)
radio = crossfade(radio)

radio = mksafe(radio)
output(radio)
```

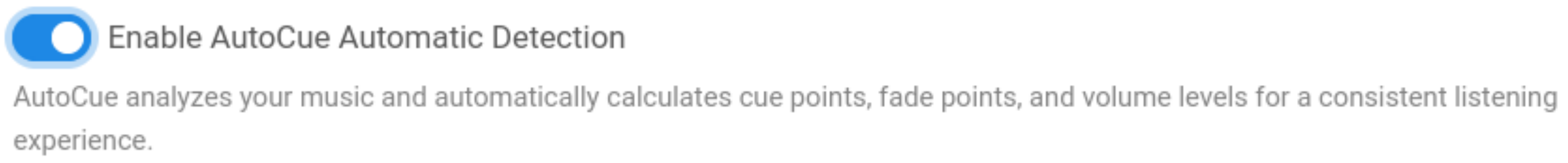
NOW THAT *WAS* EASY, RIGHT?

USAGE WITH AZURACAST

IT'S INCLUDED!

Since 2024-05-21, AzuraCast Rolling Releases have *autocue.cue_file* included, ready to use!

Switch it on in *Edit Station Profile → AutoDJ*:



Enable AutoCue Automatic Detection

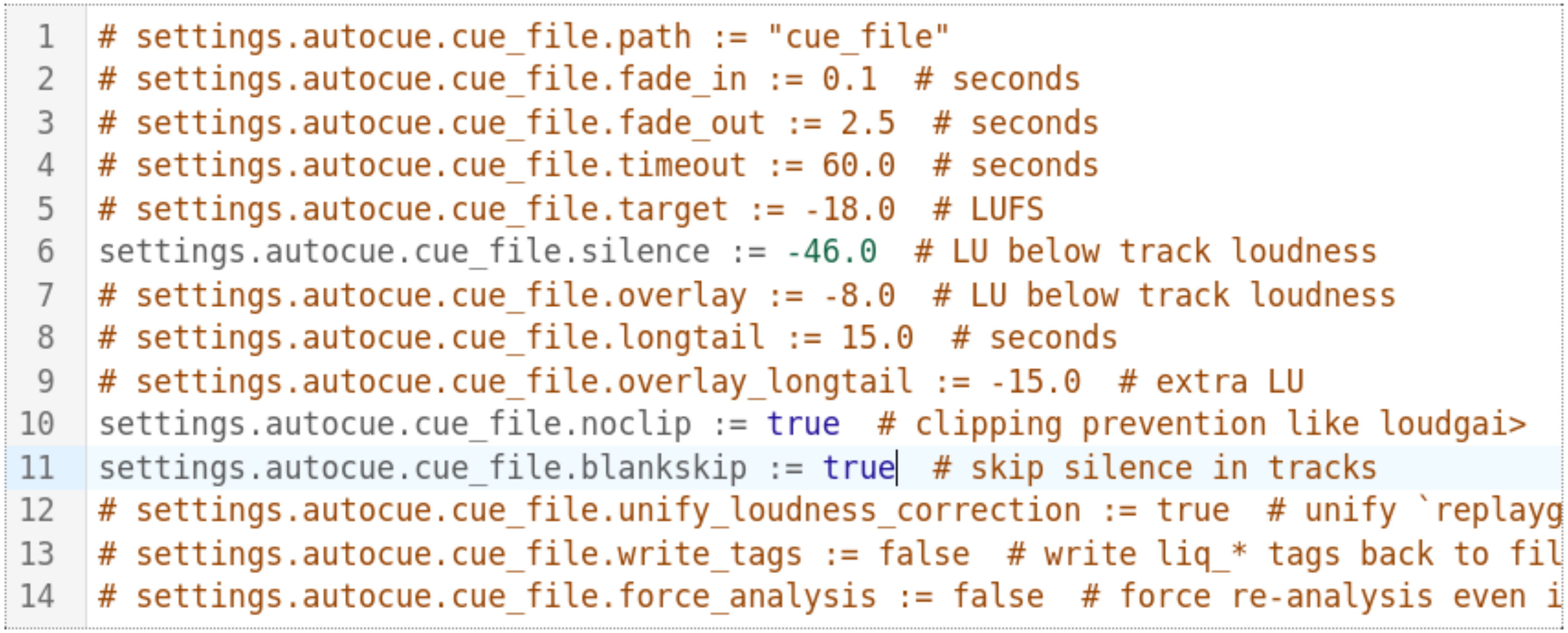
AutoCue analyzes your music and automatically calculates cue points, fade points, and volume levels for a consistent listening experience.

NOTES

- No complicated copy-pasting and setup anymore.
- Replaces crossfading code for optimum result.
- Fine-tune your personal settings in *Broadcasting* → *Edit Liquidsoap Configuration*, second input box.
- If you used the manual integration before, you must remove all its traces (copy-pasted code, cue_file) before using this.
- Save changes and *Restart Broadcasting*.

SETTINGS EXAMPLE

```
settings.autocue.cue_file.nice := true
settings.request.prefetch := 2
```



```
# settings.autocue.cue_file.path := "cue_file"
# settings.autocue.cue_file.fade_in := 0.1  # seconds
# settings.autocue.cue_file.fade_out := 2.5  # seconds
# settings.autocue.cue_file.timeout := 60.0  # seconds
# settings.autocue.cue_file.target := -18.0  # LUFS
settings.autocue.cue_file.silence := -46.0  # LU below track loudness
# settings.autocue.cue_file.overlay := -8.0  # LU below track loudness
# settings.autocue.cue_file.longtail := 15.0  # seconds
# settings.autocue.cue_file.overlay_longtail := -15.0  # extra LU
settings.autocue.cue_file.noclip := true  # clipping prevention like loudgai>
settings.autocue.cue_file.blankskip := true  # skip silence in tracks
# settings.autocue.cue_file.unify_loudness_correction := true  # unify `replayg
# settings.autocue.cue_file.write_tags := false  # write liq_* tags back to fil
# settings.autocue.cue_file.force_analysis := false  # force re-analysis even i
```

```
# AutoCue
settings.autocue.target_cross_duration := settings.autocue.cue_file.fade_out()
```

I put *all* settings in, so I don't have to look them up.

INITIAL STARTUP

- Initial startup takes a moment longer.
- Initial startup will use more CPU.
- Liquidsoap has to reach out and prepare (i.e., autocue) the next tracks for all your playlists, to be ready for immediate playout in case of fallbacks.
- **Don’t be alarmed!** CPU load will decrease to normal levels after a few minutes.

ENJOY!

– BREAK –

Thanks for following so far!

More Tech and a Question & Answer section follow in

Part II

HERE BE DRAGONS

It’s now time for the **technical stuff!**

So breathe deeply and get a beverage of your choice.

;-)

SETTINGS

Here’s a list of all possible settings with their defaults.
You *can* fine-tune everything, but the defaults are great for nearly all use cases!

```
# settings.autocue.cue_file.path := "cue_file"
# settings.autocue.cue_file.fade_in := 0.1  # seconds
# settings.autocue.cue_file.fade_out := 2.5  # seconds
# settings.autocue.cue_file.timeout := 60.0  # seconds
# settings.autocue.cue_file.target := -18.0  # LUFS
# settings.autocue.cue_file.silence := -42.0  # LU below track loudness
# settings.autocue.cue_file.overlay := -8.0  # LU below track loudness
# settings.autocue.cue_file.longtail := 15.0  # seconds
# settings.autocue.cue_file.overlay_longtail := -15.0  # extra LU
# settings.autocue.cue_file.noclip := false  # clipping prevention like loudgain's `-k`
# settings.autocue.cue_file.blankskip := false  # skip silence in tracks
# settings.autocue.cue_file.unify_loudness_correction := true  # unify `replaygain_track_gain` &
`liq_amplify`
# settings.autocue.cue_file.write_tags := false  # write liq_* tags back to file
# settings.autocue.cue_file.force_analysis := false  # force re-analysis even if tags found
# settings.autocue.cue_file.nice := false  # Linux/MacOS only: Use NI=18 for analysis

# `enable_autocue_metadata()` will autocue ALL files Liquidsoap processes.
# You can disable it for selected sources using 'annotate:liq_cue_file=false'.
# Remember you won't get `liq_amplify` data then -- expect loudness jumps!
enable_autocue_metadata()
```

THE REQUEST QUEUE

- Autocue, if using `enable_autocue_metadata()`, automatically sets

```
settings.request.prefetch := 2
```

- This means we will at all times have *the next two requests* available for immediate playout. It also gives autocue enough time to process requests in advance.
- In AzuraCast, this blocks the first two entries in the “up next” queue from being deleteable.

COMMANDLINE USAGE

You can use `cue_file` on the commandline.
It returns standard JSON data:

```
$ cue_file "The_Vow_-_Spread_Some_Love.mp3"
{"duration": 181.237551, "liq_cue_duration": 181.1, "liq_cue_in": 0.0, "liq_cue_out": 181.1,
"liq_cross_start_next": 177.3, "liq_longtail": false, "liq_loudness": "-6.72 LUFS", "liq_loudness_range":
"5.86 LU", "liq_amplify": "-11.28 dB", "liq_amplify_adjustment": "0.00 dB", "liq_reference_loudness":
"-18.00 LUFS", "liq_blankskip": false, "liq_blank_skipped": false, "liq_true_peak": "1.42 dBFS"}
```

This is ideal for debugging or pre-processing scripts.

For sorted, more human-readable output, use `jq -S`:

```
$ cue_file "The_Vow_-_Spread_Some_Love.mp3" | jq -S
{
  "duration": 181.237551,
  "liq_amplify": "-11.28 dB",
  "liq_amplify_adjustment": "0.00 dB",
  "liq_blank_skipped": false,
  "liq_blankskip": false,
  "liq_cross_start_next": 177.3,
  "liq_cue_duration": 181.1,
  "liq_cue_in": 0,
  "liq_cue_out": 181.1,
  "liq_longtail": false,
  "liq_loudness": "-6.72 LUFS",
  "liq_loudness_range": "5.86 LU",
  "liq_reference_loudness": "-18.00 LUFS",
  "liq_true_peak": "1.42 dBFS"
}
```

Use `cue_file --help` for more information.

```
$ cue_file --help
usage: cue_file [-h] [-t TARGET] [-s SILENCE] [-o OVERLAY]
                [-l LONGTAIL] [-x EXTRA] [-k] [-b] [-w] [-f]
                [-n]
                file

Return cue-in, cue-out, overlay and replaygain data for an
audio file as JSON. To be used with my Liquidsoap "autocue:"
protocol.

positional arguments:
  file                  File to be processed

options:
  -h, --help            show this help message and exit
  -t TARGET, --target TARGET
                        LUFS reference target (default: -18.0)
  -s SILENCE, --silence SILENCE
                        LU below integrated track loudness for
                        cue-in & cue-out points (silence
                        removal at beginning & end of a track)
                        (default: -42.0)
  -o OVERLAY, --overlay OVERLAY
                        LU below integrated track loudness to
                        trigger next track (default: -8.0)
  -l LONGTAIL, --longtail LONGTAIL
```

METADATA

Metadata is used in a *prioritized* manner, so parameters can easily be *stored* and *overridden* if needed.

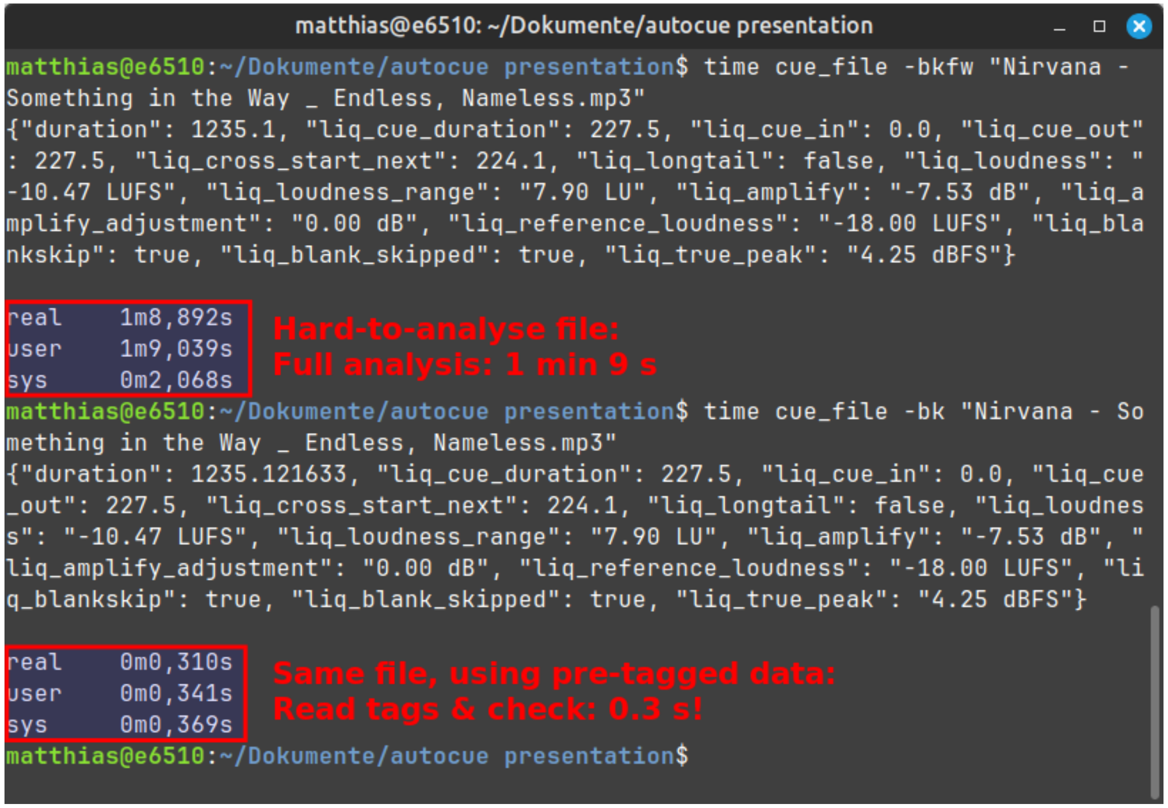
The priorities are, from low to high:

- Metadata calculated by `cue_file`
- Metadata stored in file *tags*
- Metadata given in *annotations*

THIS MEANS:

- **Tags** in files can override `cue_file` behaviour, and allow it to just *use* these values instead of doing a costly re-analysis.
- The user can still override these by using **annotations**. This mechanism is also used by AzuraCast’s *Visual Cue Editor*, so the user settings always “win” over defaults or stored tags.

USING PRE-TAGGED FILES IS *FAST!*


matthias@e6510: ~/Dokumente/autocue presentation

```
matthias@e6510:~/Dokumente/autocue presentation$ time cue_file -bkfw "Nirvana -
Something in the Way _ Endless, Nameless.mp3"
{"duration": 1235.1, "liq_cue_duration": 227.5, "liq_cue_in": 0.0, "liq_cue_out"
: 227.5, "liq_cross_start_next": 224.1, "liq_longtail": false, "liq_loudness": "
-10.47 LUFS", "liq_loudness_range": "7.90 LU", "liq_amplify": "-7.53 dB", "liq_a
mplify_adjustment": "0.00 dB", "liq_reference_loudness": "-18.00 LUFS", "liq_bla
nkskip": true, "liq_blank_skipped": true, "liq_true_peak": "4.25 dBFS"}

real    1m8,892s
user    1m9,039s
sys     0m2,068s
matthias@e6510:~/Dokumente/autocue presentation$ time cue_file -bk "Nirvana - So
mething in the Way _ Endless, Nameless.mp3"
{"duration": 1235.121633, "liq_cue_duration": 227.5, "liq_cue_in": 0.0, "liq_cue
_out": 227.5, "liq_cross_start_next": 224.1, "liq_longtail": false, "liq_loudnes
s": "-10.47 LUFS", "liq_loudness_range": "7.90 LU", "liq_amplify": "-7.53 dB", "
liq_amplify_adjustment": "0.00 dB", "liq_reference_loudness": "-18.00 LUFS", "li
q_blankskip": true, "liq_blank_skipped": true, "liq_true_peak": "4.25 dBFS"}

real    0m0,310s
user    0m0,341s
sys     0m0,369s
matthias@e6510:~/Dokumente/autocue presentation$
```

Hard-to-analyse file:
Full analysis: 1 min 9 s

Same file, using pre-tagged data:
Read tags & check: 0.3 s!

Nirvana song: 222 times faster!

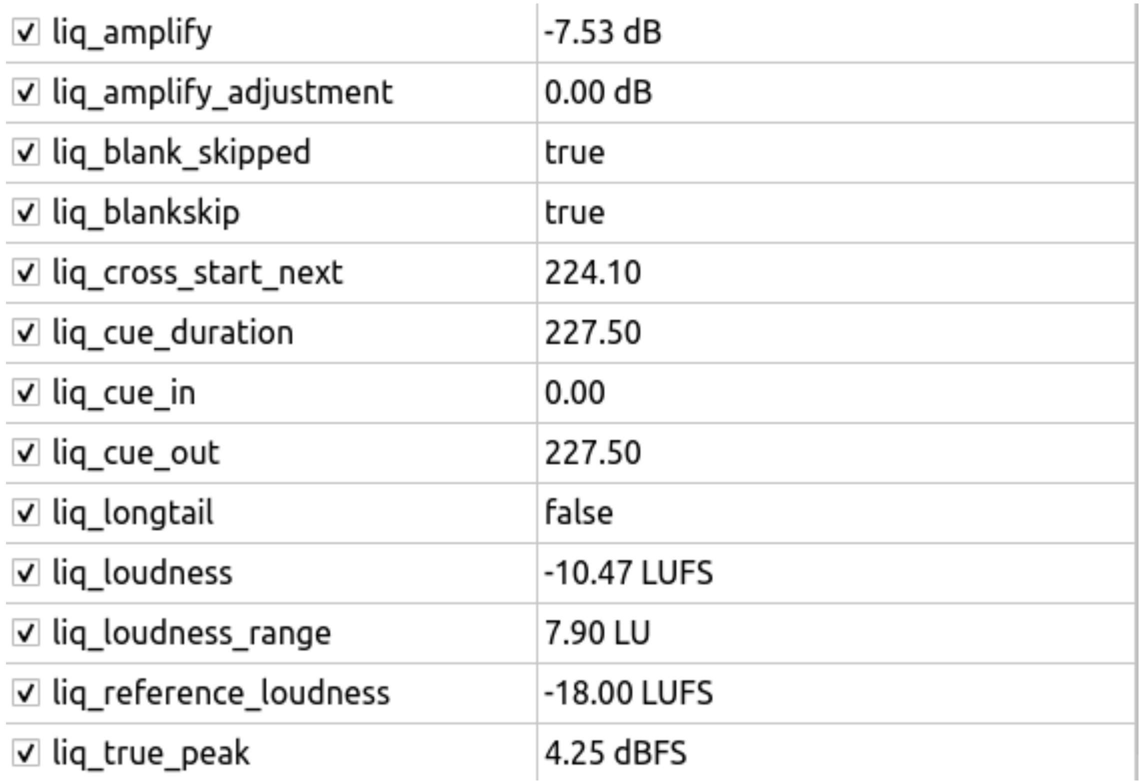
CUE_FILE IS “INTELLIGENT”

- Depending on requested parameters and stored file tags, it tries to avoid a costly re-analysis.

Examples:

- Needed file tags missing → new analysis (slow)
- Tags fit request → use tags, no new analysis (fast)
- Playout at -14 LUFS requested, tags are -18 LUFS → can recalculate, no new analysis (fast)
- blankskip disabled, tags include blankskip (and vice versa) → re-analysis forced (slow)

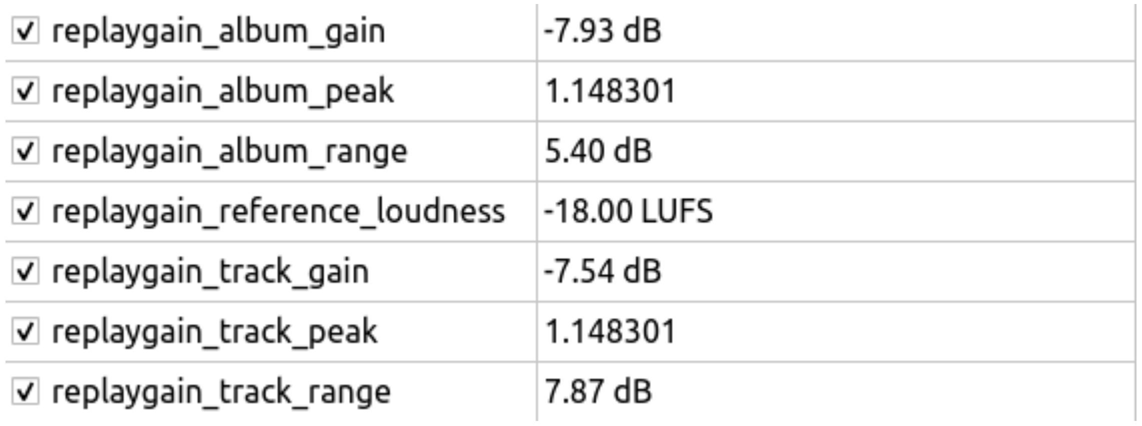
TAG EXAMPLE



✓	liq_amplify	-7.53 dB
✓	liq_amplify_adjustment	0.00 dB
✓	liq_blank_skipped	true
✓	liq_blankskip	true
✓	liq_cross_start_next	224.10
✓	liq_cue_duration	227.50
✓	liq_cue_in	0.00
✓	liq_cue_out	227.50
✓	liq_longtail	false
✓	liq_loudness	-10.47 LUFS
✓	liq_loudness_range	7.90 LU
✓	liq_reference_loudness	-18.00 LUFS
✓	liq_true_peak	4.25 dBFS

Tags written by `cue_file -w`.

REPLAYGAIN TAGS



✓	replaygain_album_gain	-7.93 dB
✓	replaygain_album_peak	1.148301
✓	replaygain_album_range	5.40 dB
✓	replaygain_reference_loudness	-18.00 LUFS
✓	replaygain_track_gain	-7.54 dB
✓	replaygain_track_peak	1.148301
✓	replaygain_track_range	7.87 dB

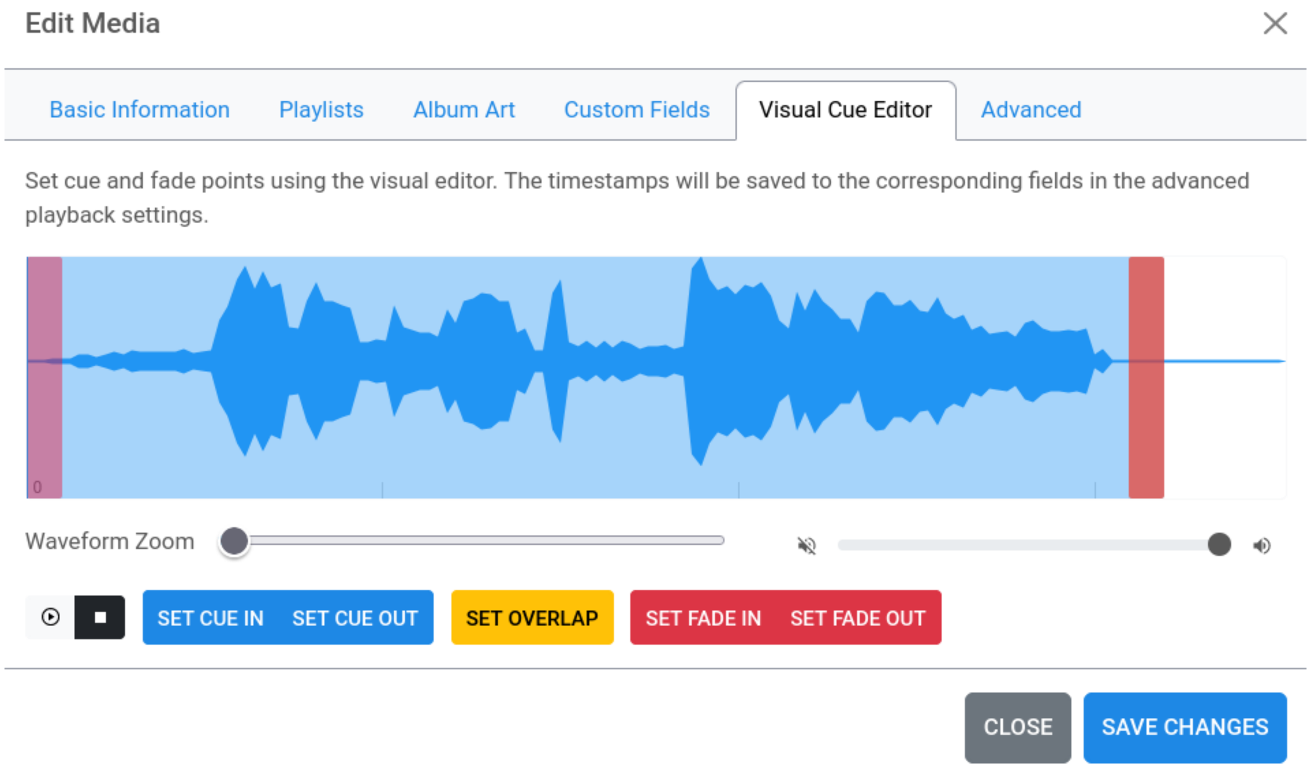
ReplayGain tags are *used* by `cue_file`
but *never written back to audio files.*
This preserves your data from unintended changes.

ANNOTATION EXAMPLE

```
uri = "/home/matthias/Musik/Other/Jingles/Short"
jingles = playlist(prefix='annotate:liq_blankskip=false,'
  ^ 'liq_fade_in=0.10,liq_fade_out=0.10'
  ^ ':', uri)
```

A jingles playlist: We want to disable blank skipping and set fade-in and fade-out times to 0.1 s, respectively.

AZURACAST VISUAL CUE EDITOR


Edit Media

Basic Information | Playlists | Album Art | Custom Fields | Visual Cue Editor | Advanced

Set cue and fade points using the visual editor. The timestamps will be saved to the corresponding fields in the advanced playback settings.

Values set here are used as *annotations*, which have the highest priority. Just what we want.

METADATA CATEGORIES

Basically, we use three *types* of metadata:

- *“Switches”* that control autocue functionality, on a *per-file* or *per-playlist* basis.
- *Results* that are used in further playout processing.
- *Informational* metadata that might come in handy.

“SWITCHES”

- `liq_blankskip` (bool)
  Activates/deactivates blank skipping mode
- `liq_cue_file` (bool)
  Enables/disables autocue (i.e., for large video files)
- AzuraCast `jingle_mode` (bool)
  Disables blank skipping for AzuraCast “Jingle Mode” playlists
- SAM Broadcaster `songtype` (char)
  Disables blank skipping for song types other than “S” (Song)

RESULTS

- `duration` (s)
- `liq_amplify` (dB)
- `liq_amplify_adjustment` (dB)
- `liq_cross_start_next` (s)
- `liq_cue_in` (s)
- `liq_cue_out` (s)
- `liq_reference_loudness` (LUFS)
- `replaygain_track_gain` (dB)
- `replaygain_reference_loudness` (LUFS)

INFORMATIONAL

- `liq_blank_skipped` (bool)
- `liq_cue_duration` (s)
- `liq_longtail` (bool)
- `liq_loudness` (LUFS)
- `liq_loudness_range` (LU)
- `liq_true_peak` (dBFS)

OTHERS

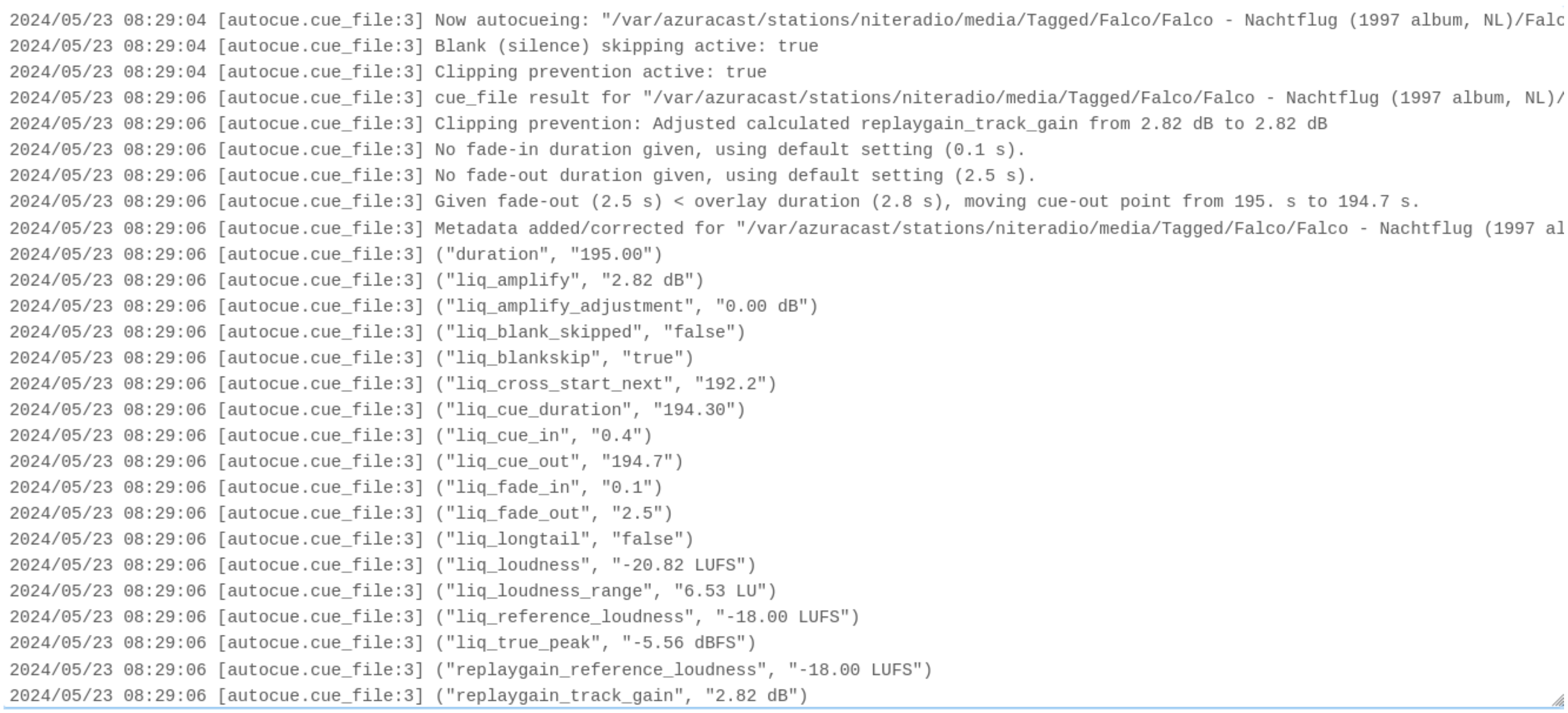
There *are* a plethora of other metadata that are either used internally, or reserved for future expansion.

Fading data (duration, type, curve), cross duration, Opus Gain, ramp and hook points belong to this category.

THE LOGFILE

- The Liquidsoap *log file* clearly shows autocue's workings.
- It's an invaluable tool for diagnosing problems.
- Logging Levels:
  - 2: Severe (errors/problems detected)
  - 3: Important (autocue information and results)
  - 4: Info (for debugging, *lots* of output)

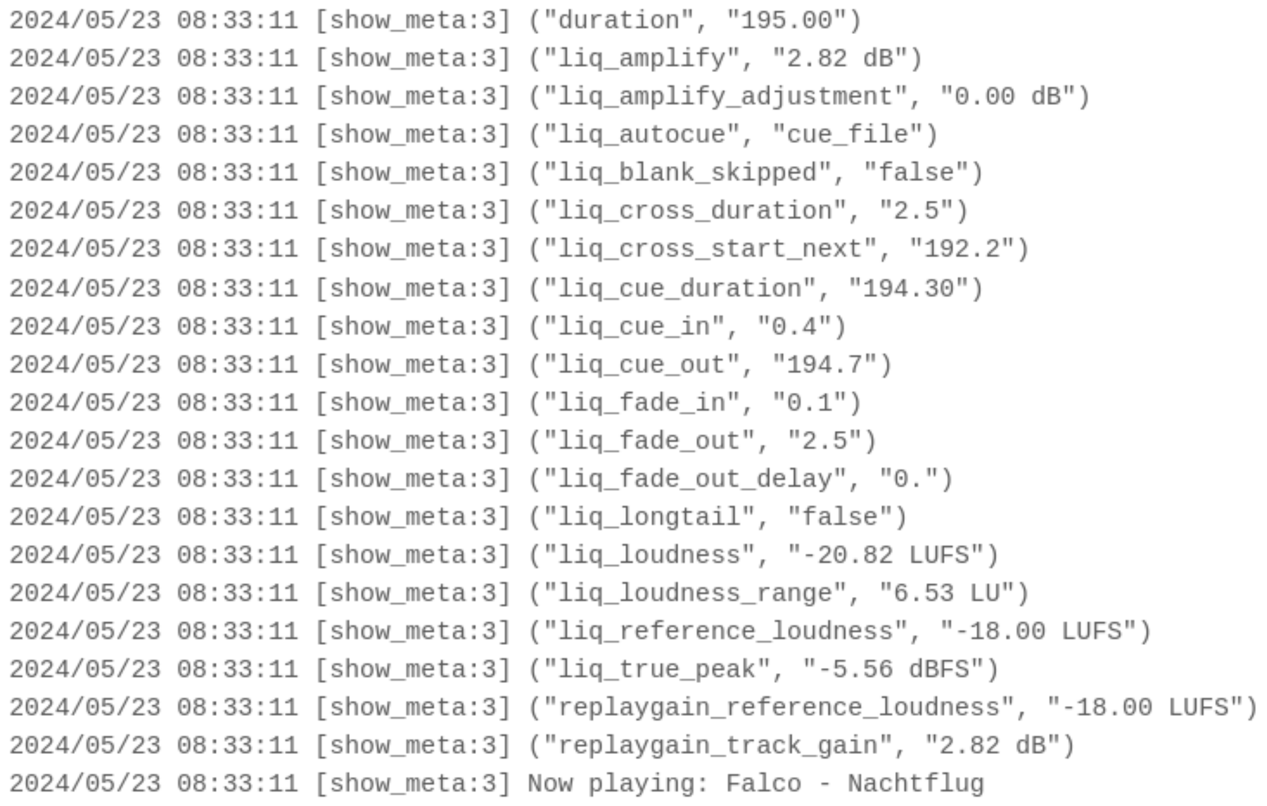
LOG: AUTOCUE.CUE_FILE



```
2024/05/23 08:29:04 [autocue.cue_file:3] Now autocueing: "/var/azuracast/stations/niteradio/media/Tagged/Falco/Falco - Nachtflug (1997 album, NL)/Falc
2024/05/23 08:29:04 [autocue.cue_file:3] Blank (silence) skipping active: true
2024/05/23 08:29:04 [autocue.cue_file:3] Clipping prevention active: true
2024/05/23 08:29:06 [autocue.cue_file:3] cue_file result for "/var/azuracast/stations/niteradio/media/Tagged/Falco/Falco - Nachtflug (1997 album, NL)/
2024/05/23 08:29:06 [autocue.cue_file:3] Clipping prevention: Adjusted calculated replaygain_track_gain from 2.82 dB to 2.82 dB
2024/05/23 08:29:06 [autocue.cue_file:3] No fade-in duration given, using default setting (0.1 s).
2024/05/23 08:29:06 [autocue.cue_file:3] No fade-out duration given, using default setting (2.5 s).
2024/05/23 08:29:06 [autocue.cue_file:3] Given fade-out (2.5 s) < overlay duration (2.8 s), moving cue-out point from 195. s to 194.7 s.
2024/05/23 08:29:06 [autocue.cue_file:3] Metadata added/corrected for "/var/azuracast/stations/niteradio/media/Tagged/Falco/Falco - Nachtflug (1997 al
2024/05/23 08:29:06 [autocue.cue_file:3] ("duration", "195.00")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_amplify", "2.82 dB")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_amplify_adjustment", "0.00 dB")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_blank_skipped", "false")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_blankskip", "true")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_cross_start_next", "192.2")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_cue_duration", "194.30")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_cue_in", "0.4")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_cue_out", "194.7")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_fade_in", "0.1")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_fade_out", "2.5")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_longtail", "false")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_loudness", "-20.82 LUFS")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_loudness_range", "6.53 LU")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_reference_loudness", "-18.00 LUFS")
2024/05/23 08:29:06 [autocue.cue_file:3] ("liq_true_peak", "-5.56 dBFS")
2024/05/23 08:29:06 [autocue.cue_file:3] ("replaygain_reference_loudness", "-18.00 LUFS")
2024/05/23 08:29:06 [autocue.cue_file:3] ("replaygain_track_gain", "2.82 dB")
```

Shows autocue information and results

LOG: SHOW_META



```
2024/05/23 08:33:11 [show_meta:3] ("duration", "195.00")
2024/05/23 08:33:11 [show_meta:3] ("liq_amplify", "2.82 dB")
2024/05/23 08:33:11 [show_meta:3] ("liq_amplify_adjustment", "0.00 dB")
2024/05/23 08:33:11 [show_meta:3] ("liq_autocue", "cue_file")
2024/05/23 08:33:11 [show_meta:3] ("liq_blank_skipped", "false")
2024/05/23 08:33:11 [show_meta:3] ("liq_cross_duration", "2.5")
2024/05/23 08:33:11 [show_meta:3] ("liq_cross_start_next", "192.2")
2024/05/23 08:33:11 [show_meta:3] ("liq_cue_duration", "194.30")
2024/05/23 08:33:11 [show_meta:3] ("liq_cue_in", "0.4")
2024/05/23 08:33:11 [show_meta:3] ("liq_cue_out", "194.7")
2024/05/23 08:33:11 [show_meta:3] ("liq_fade_in", "0.1")
2024/05/23 08:33:11 [show_meta:3] ("liq_fade_out", "2.5")
2024/05/23 08:33:11 [show_meta:3] ("liq_fade_out_delay", "0.")
2024/05/23 08:33:11 [show_meta:3] ("liq_longtail", "false")
2024/05/23 08:33:11 [show_meta:3] ("liq_loudness", "-20.82 LUFS")
2024/05/23 08:33:11 [show_meta:3] ("liq_loudness_range", "6.53 LU")
2024/05/23 08:33:11 [show_meta:3] ("liq_reference_loudness", "-18.00 LUFS")
2024/05/23 08:33:11 [show_meta:3] ("liq_true_peak", "-5.56 dBFS")
2024/05/23 08:33:11 [show_meta:3] ("replaygain_reference_loudness", "-18.00 LUFS")
2024/05/23 08:33:11 [show_meta:3] ("replaygain_track_gain", "2.82 dB")
2024/05/23 08:33:11 [show_meta:3] Now playing: Falco - Nachtflug
```

Shows final values used in playout
(`liq_*` & `replaygain` metadata)

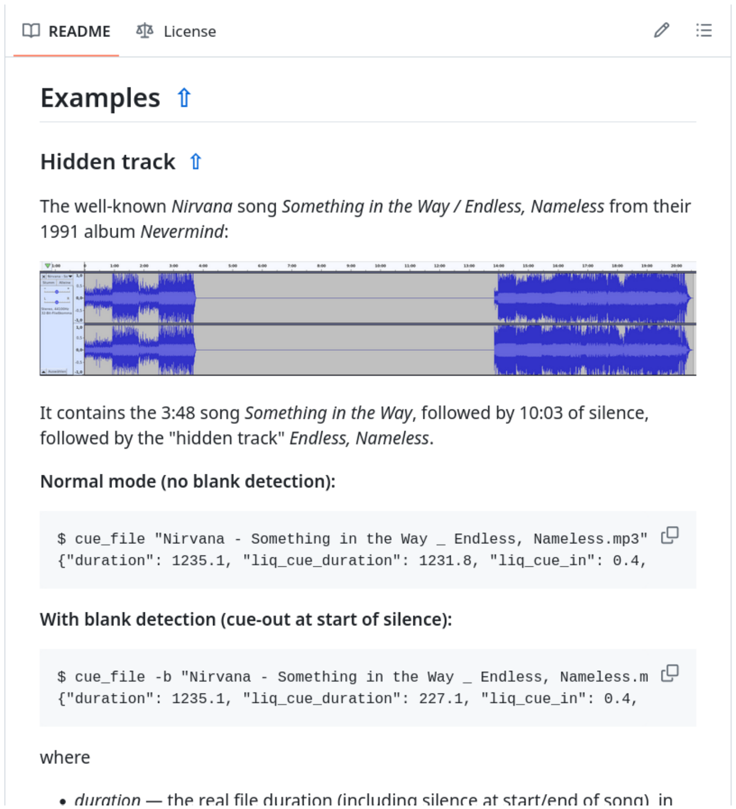
DOWNLOAD, DOCUMENTATION

GitHub repo:

https://github.com/Moonbase59/autocue/

Currently (May 2024) please use the files from the `integrate-with-liquidsoap` branch!

DOCS & EXAMPLES ON GITHUB

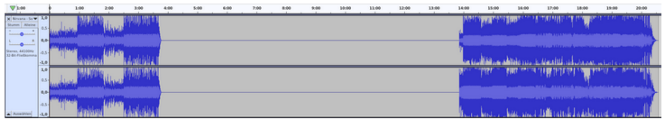


README License

Examples ⇧

Hidden track ⇧

The well-known *Nirvana* song *Something in the Way / Endless, Nameless* from their 1991 album *Nevermind*:



It contains the 3:48 song *Something in the Way*, followed by 10:03 of silence, followed by the "hidden track" *Endless, Nameless*.

Normal mode (no blank detection):

```
$ cue_file "Nirvana - Something in the Way _ Endless, Nameless.mp3"
{"duration": 1235.1, "liq_cue_duration": 1231.8, "liq_cue_in": 0.4,
```

With blank detection (cue-out at start of silence):

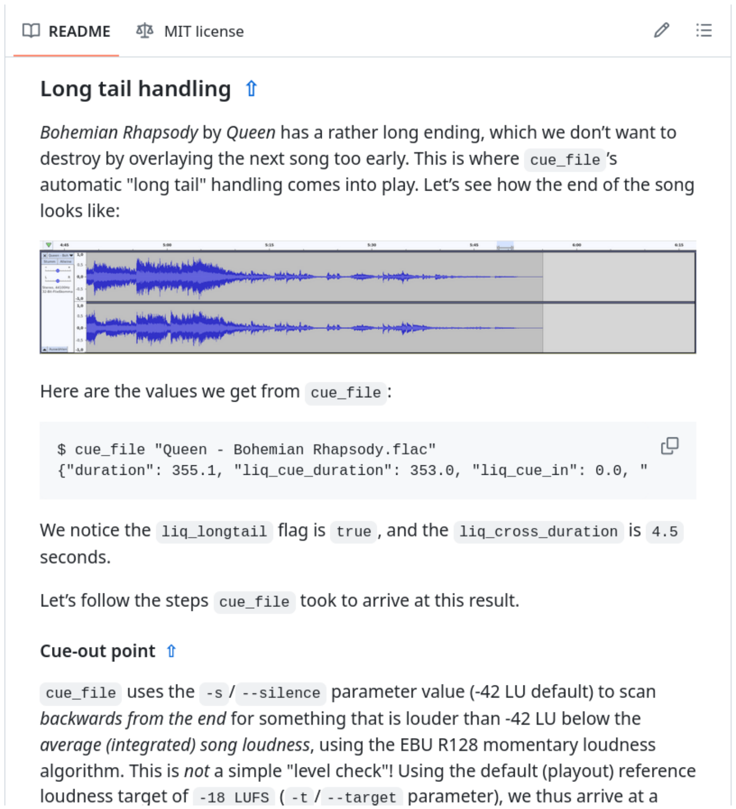
```
$ cue_file -b "Nirvana - Something in the Way _ Endless, Nameless.m
{"duration": 1235.1, "liq_cue_duration": 227.1, "liq_cue_in": 0.4,
```

where

- *duration* — the real file duration (including silence at start/end of song), in

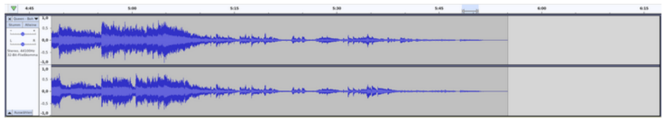
Hidden track

DOCS & EXAMPLES ON GITHUB


README MIT license

Long tail handling ⇧

Bohemian Rhapsody by *Queen* has a rather long ending, which we don't want to destroy by overlaying the next song too early. This is where `cue_file`'s automatic "long tail" handling comes into play. Let's see how the end of the song looks like:



Here are the values we get from `cue_file`:

```
$ cue_file "Queen - Bohemian Rhapsody.flac"
{"duration": 355.1, "liq_cue_duration": 353.0, "liq_cue_in": 0.0, "
```

We notice the `liq_longtail` flag is `true`, and the `liq_cross_duration` is `4.5` seconds.

Let's follow the steps `cue_file` took to arrive at this result.

Cue-out point ⇧

`cue_file` uses the `-s` / `--silence` parameter value (-42 LU default) to scan *backwards from the end* for something that is louder than -42 LU below the *average (integrated) song loudness*, using the EBU R128 momentary loudness algorithm. This is *not* a simple "level check"! Using the default (playout) reference loudness target of `-18 LUFS` (`-t` / `--target` parameter), we thus arrive at a

Long tail handling

ROADMAP

- Scrap `autocue2`. `autocue.cue_file` is the supported integrated solution for LS 2.2.5 & newer.
- Update documentation.
- ~~Fix “double autocue” issue.~~
- Fix “new fade-in > old fade-out” issue with **toots**.
- Testing with LS 2.3.x.
- ~~AzuraCast integration with **BusterNeece**.~~ (WIP)

QUESTIONS & ANSWERS

LINKS

- Liquidsoap: https://github.com/savonet/liquidsoap
- AzuraCast: https://github.com/AzuraCast/AzuraCast
- Autocue: https://github.com/Moonbase59/autocue

- I’m also on the *Liquidsoap* and *AzuraCast* servers on Discord, as “Moonbase59”.

- This presentation will be made available as:
  - recording on YouTube (check the Liquidshop 4 page)
  - web page (reveal.js)
  - downloadable PDF file

THANKS!

Matthias C. Hormann
("Moonbase59")