TECH 3342

LOUDNESS RANGE: A MEASURE TO SUPPLEMENT EBU R 128 LOUDNESS NORMALIZATION

SUPPLEMENTARY INFORMATION FOR R 128

VERSION 3.0

Geneva
January 2016
Contents

1. Introduction .................................................................................... 5

2. Loudness Range ................................................................................. 5

3. Algorithm Description ........................................................................ 5
   3.1 Algorithm Definition ............................................................................................. 6

4. Minimum requirements, compliance test................................................. 7

5. MATLAB implementation ..................................................................... 8

6. References ...................................................................................... 9

7. Further reading ................................................................................ 9
Loudness Range:
A measure to supplement Loudness normalisation
in accordance with EBU R 128

<table>
<thead>
<tr>
<th>EBU Committee</th>
<th>First Issued</th>
<th>Revised</th>
<th>Re-issued</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC</td>
<td>2010</td>
<td>2011, 2016</td>
<td></td>
</tr>
</tbody>
</table>

Keywords: Loudness, normalisation, dynamic range, statistics

1. Introduction
The EBU has studied the needs of audio signal levels in production, distribution and transmission of broadcast programmes. It is of the opinion that an audio-levelling paradigm is needed based on loudness measurement. This is described in EBU Technical Recommendation R 128 [1]. In addition to the average loudness of a programme (‘Programme Loudness’) the EBU recommends that the measures ‘Loudness Range’ and ‘Maximum True Peak Level’ be used for the normalisation of audio signals and to comply with the technical limits of the complete signal chain as well as the aesthetic needs of each programme/station depending on the genre(s) and the target audience.

In this document the measure ‘Loudness Range’ and the algorithm for its computation will be introduced and explained in detail.

The algorithm was kindly provided by the company TC Electronic.

2. Loudness Range
Loudness Range (abbreviated ‘LRA’) quantifies the variation in a time-varying loudness measurement. Loudness Range is supplementary to the main audio measure, Programme Loudness, of EBU R 128. Loudness Range measures the variation of loudness on a macroscopic time-scale, in units of LU (Loudness Units). The computation of Loudness Range is based on a measurement of loudness level as specified in ITU-R BS.1770 [2], albeit with a different relative gating threshold (see § 3.1). Loudness Range should not be confused with other measures like dynamic range or crest factor, etc.

3. Algorithm Description
The computation of Loudness Range is based on the statistical distribution of measured loudness. Thus, a short but very loud event would not affect the Loudness Range of a longer segment. Similarly the fade-out at the end of a music track, for example, would not increase Loudness Range noticeably. Specifically, the range of the distribution of loudness levels is determined by estimating the difference between a low and a high percentile of the distribution. This method is analogous to the Interquartile Range (IQR), used in the field of descriptive statistics to obtain a robust estimate of the spread of a data sample.

Loudness Range furthermore employs a cascaded gating method. Certain types of programme may be, overall, very consistent in loudness, but have some sections with very low loudness, for example only containing background sounds (e.g. low-level atmospheres) or background noise (e.g. air conditioning hum). If Loudness Range did not use the gating, such programmes would (incorrectly) get quite a high Loudness Range measurement, due to the relatively large difference in loudness between the regions of background sounds/noise and those of normal (foreground) loudness.
The **Loudness Range** algorithm is independent of the sample rate and format of the input signal.

3.1 **Algorithm Definition**

The input to the algorithm is a vector of loudness levels, computed as specified in ITU-R BS.1770 with different relative-threshold gating, using a *sliding analysis-window* of length 3 seconds for integration. An overlap between consecutive analysis-windows must be used in order to prevent loss of precision in the measurement of shorter programmes. A minimum block overlap of 2.9 s between consecutive analysis windows (i.e. ≥10 Hz sampling of the loudness level) is required; the exact amount of overlap is implementation-dependent.

A cascaded gating scheme is employed which uses an absolute threshold of very low level, in combination with a relative threshold of higher, signal-dependent, level.

The purpose of the relative-threshold gating is to gate out any periods of silence or background sounds, using a method that is independent of any level-normalisation of the input signal. The lower edge of **Loudness Range** should not be defined by the noise floor (which may be inaudible), but should instead correspond to the weakest ‘real’ signal. The relative threshold is set to a level of $-20 \text{ LU}$ relative to the absolute-gated loudness level. The purpose of the absolute-threshold gate is to make the conversion from the relative threshold to an absolute level robust against longer periods of silence or low-level background noise. The absolute threshold is set to $-70 \text{ LUFS}$, because no relevant signals are generally found below this loudness level.

It is noted that measurement of very short programmes, where leading or trailing silence is included, or of programmes consisting, for example, of isolated utterances, could result in misleadingly high values of LRA.

The application of the cascaded gating leaves only the loudness levels of the sliding-window blocks that contain foreground and (medium-level) background sounds, eliminating low-level signals, background noise, and silence. The width of the distribution of these loudness levels is then quantified using a *percentile range*. Percentiles belong to *non-parametric statistics* and are employed in the computation of **Loudness Range** because the loudness levels cannot in general be assumed to belong to a particular statistical distribution.

LRA is defined as the difference between the estimates of the 10th and the 95th percentiles of the distribution. The lower percentile of 10%, can, for example, prevent the fade-out of a music track from dominating **Loudness Range**. The upper percentile of 95% ensures that a single unusually loud sound, such as a gunshot in a movie, cannot by itself be responsible for a large **Loudness Range**.
In Figure 1, the absolute threshold is marked at −70 LUFS. The absolute-gated loudness level from that is −21.6 LUFS (marked as \textit{abs-gated, integrated}). The relative threshold is shown 20 LU below that at −41.6 LUFS. The resulting \textit{Loudness Range} (LRA = 25.0 LU) is shown between the 10\textsuperscript{th} and 95\textsuperscript{th} percentiles of the distribution of loudness levels above the relative threshold.

4. \textbf{Minimum requirements, compliance test}

The \textit{Loudness Range} measure is a part of an EBU Mode loudness meter, as defined in EBU Tech Doc 3341 [4]. In the following, a set of Minimum Requirements for the \textit{Loudness Range} computation is provided, in the form of ‘minimum requirements test signals’ with corresponding expected response and accepted tolerances.

If a loudness meter, offering EBU Mode, does \textit{not} pass these ‘minimum requirements’ tests, there is a considerable risk that the meter is \textit{not} compliant with EBU Mode. If, on the other hand, a meter does pass the ‘minimum requirements’ tests this does \textit{not} imply that the meter is sufficiently accurate in all respects of its implementation.
Table 1: Minimum requirements test signals

<table>
<thead>
<tr>
<th>Test case</th>
<th>Test signal</th>
<th>Expected response and accepted tolerances</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Stereo sine wave, 1000 Hz, −20.0 dBFS (per-channel peak level); signal applied in phase to both channels simultaneous, 20 s duration; followed immediately by the same signal at −30.0 dBFS (i.e. the tones are 10 dB apart)</td>
<td>LRA = 10 ±1 LU</td>
</tr>
<tr>
<td>2</td>
<td>As #1, with the 2 tones at −20.0 dBFS and −15.0 dBFS, respectively</td>
<td>LRA = 5 ±1 LU</td>
</tr>
<tr>
<td>3</td>
<td>As #1, with the 2 tones at −40.0 dBFS and −20.0 dBFS, respectively</td>
<td>LRA = 20 ±1 LU</td>
</tr>
<tr>
<td>4</td>
<td>As #1, but with 5 tone-segments at −50.0 dBFS, −35.0 dBFS, −20.0 dBFS, −35.0 dBFS, and −50.0 dBFS, respectively; 20 s duration each tone</td>
<td>LRA = 15 ±1 LU</td>
</tr>
<tr>
<td>5</td>
<td>Authentic programme 1, stereo, narrow Loudness Range (NLR) programme segment; similar in genre to a commercial/promo</td>
<td>LRA = 5 ±1 LU</td>
</tr>
<tr>
<td>6</td>
<td>Authentic programme 2, stereo, wide Loudness Range (WLR) programme segment; similar in genre to a movie/drama</td>
<td>LRA = 15 ±1 LU</td>
</tr>
</tbody>
</table>

In all the above test cases, the expected response is unchanged if the test signal is repeated one or more times in its full length. The loudness meter shall be reset before each measurement.

The ‘minimum requirements test signals’ [5] are available for download from the EBU Technical website.

5. MATLAB implementation

An algorithm for computing Loudness Range is provided below using the MATLAB® language (no MATLAB toolbox functions are used). This MATLAB implementation is intended to complement the textual definition of the LRA algorithm. However, other implementations would be equally valid provided that the measurements stay within the permitted tolerance, and even though they might yield slightly different LRA measurements for some input signals.

```matlab
% A MATLAB FUNCTION TO COMPUTE LOUDNESS RANGE
% --------------------------------------------------------------------
function LRA = LoudnessRange( ShortTermLoudness )

% Input: ShortTermLoudness is a column vector of loudness levels, computed
% as specified in ITU R BS.1770 without gating, using a sliding
% analysis-window of length 3 s, computed at a rate of at least 10 Hz.
% For file-based measurements, the signal should be followed by at least 1.5 s of silence
% (corresponding to the latency of the loudness analysis-window) before the final LRA
% value is determined.

% Constants
ABS_THRES = −70; % LUFS (= absolute measure)
REL_THRES = −20; % LU (= relative measure)
PRC_LOW = 10; % lower percentile
PRC_HIGH = 95; % upper percentile

% Apply the absolute-threshold gating
abs_gate_vec = (ShortTermLoudness >= ABS_THRES);
% abs_gate_vec is indices of loudness levels above absolute threshold
stl_absgated_vec = ShortTermLoudness(abs_gate_vec);
% only include loudness levels that are above gate threshold
```
% Apply the relative-threshold gating
n = length(stl_absgated_vec);
stl_power = sum(10.^(stl_absgated_vec./10))/n; % undo 10log10, and calculate mean
stl_integrated = 10*log10(stl_power); % LUFS
rel_gate_vec = (stl_absgated_vec >= stl_integrated + REL_THRES);
% rel_gate_vec is indices of loudness levels above relative threshold
stl_relgated_vec = stl_absgated_vec( rel_gate_vec );
% only include loudness levels that are above gate threshold

% Compute the high and low percentiles of the distribution of
% values in stl_relgated_vec
n = length(stl_relgated_vec);
stl_sorted_vec = sort(stl_relgated_vec);
% sort elements in ascending order
stl_perc_low  = stl_sorted_vec(round((n-1)*PRC_LOW/100 + 1));
stl_perc_high = stl_sorted_vec(round((n-1)*PRC_HIGH/100 + 1));

% Compute the Loudness Range measure
LRA = stl_perc_high - stl_perc_low; % in LU

6. References
[1] EBU Technical Recommendation R 128 ‘Loudness normalisation and permitted maximum level of audio signals’

7. Further reading

EBU Tech Doc 3343 ‘Guidelines for Production of Programmes in accordance with EBU R 128’
EBU Tech Doc 3344 ‘Guidelines for Distribution and Reproduction of Programmes in accordance with EBU R 128’