Analisi dei contenuti audio di fenomeni Larsen emergenti in una stanza

Luca Spanedda

July, 2022

Abstract

L'obiettivo dell' Audio Content Analysis - **ACA** - è di estrarre informazioni a diversi livelli di astrazione da segnali audio, sia in tempo reale che in tempo differito. L'ACA rientra nella disciplina del Music Information Retrieval - **MIR** -, un'area dell'informatica musicale che studia i metodi e gli strumenti per l'estrazione dell'informazione musicale per mezzo di un computer. Oltre alle analisi in tempo differito che richiedono di prendere in considerazione l'intera lunghezza del segnale su cui si effettua l'analisi, oggi è possibile operare alcune di queste analisi in tempo reale grazie ad alcuni linguaggi di programmazione. In questa ricerca andremo ad utilizzare il linguaggio di programmazione Faust (GRAME) per il tempo reale e la libreria di Python pyACA per il tempo differito, al fine di condurre un'analisi sperimentale su alcuni effetti Larsen generati all'interno di una stanza.

L’effetto Larsen

Citando una definizione che il compositore Agostino Di Scipio ha esposto in un suo articolo pubblicato questo stesso anno presso la rivista Online: *ECHO* dell’ *Orpheus Institute, in Ghent*.[1] (tutte le fonti sono reperibili in Bibliografia e Sitografia)

Si può definire l’effetto Larsen come segue:
<<A condenser microphone (M1) and a dynamic loudspeaker (L1) stand in the performance place (S), few or several meters apart, maybe not too far from walls (or curtains, or other larger surfaces). They are connected (through one or more amplification stages) to realize a very basic electroacoustic chain: M1→L1→S. There’s no sound M1 should capture, though, no sound source save the minimal, barely audible turbulence of the background noise, in a situation of ‘silence’. This ‘sound-of-nothing’ is amplified and heard through L1, whence it comes back in S.
If amplification suffices, the L1 sound feeds back into M1 and the chain design closes onto itself, making a ‘reinjection’ circuit – a feedback loop. The amplitude level, the transductive technical features of M1 and L1, their relative distance, the distance from walls, etc. – all of that (and much more) sets the actual feedback loop gain. With not-too-high gain levels, what is engendered is an audible nuisance, a kind of ‘halo’: the sound reinjection decays more or less rapidly, in a kind of badly sounding, spectrally uneven reverb effect. With higher gain levels, the loop eventually enters a self-oscillatory regime, it may ‘ring’ or ‘howl’, as is often said. Because of the iterated reinjection, the barely audible but spectrally wide background noise accumulates in the loop and finally (quickly) yields an increasingly louder sustained sound of narrower spectrum – this is often heard as a peaking tone of definite pitch, or a tone cluster. That’s the Larsen effect: a self-sustaining feedback resonance occasioned by a positive feedback loop (FB+) (‘positive’ here means greater than unit gain).>>

Riprendendo la spiegazione, traducendo in italiano ed esemplificando il processo illustrato: In una stanza vengono collegati fra loro (tramite uno o più stadi di amplificazione) gli elementi di una catena elettroacustica molto elementare: Microfono ed Altoparlante. Anche in una situazione di "silenzio" ideale, viene catturata dal microfono in ogni caso, inevitabilmente, perfino la turbolenza minima appena udibile presente nel rumore ambientale. Questo suono catturato dal microfono, viene amplificato e riprodotto dall’altoparlante a sua volta. E se l'amplificazione è sufficiente il suono dall’altoparlante ritorna al microfono e il design della catena elettroacustica si chiude su se stessa, creando un circuito di retroazione (loop), anche detto di feedback. Il livello di ampiezza, le caratteristiche tecniche trasduttive di microfono e altoparlante, la loro distanza relativa, la distanza dalle pareti ed altri potenziali fattori influenti, delineano un’oscillazionein fase con il segnale che viene a sommarsi ad esso e viene amplificata e riprodotta a sua volta con ampiezza via via crescente, idealmente illimitata. Con livelli di guadagno non troppo elevati, ciò che si genera è un fastidio udibile, una sorta di 'alone': la reiniezione del suono decade più o meno rapidamente, in una specie di effetto riverbero composto da suono spettralmente irregolare. Con livelli di guadagno più elevati, il feedback loop entra in un regime di auto-oscillazione. A causa della reiniezione ripetuta, il rumore di fondo appena udibile ma spettralmente ampio si accumula nel loop e alla fine (rapidamente) produce un suono sostenuto sempre più forte di uno spettro più ristretto - questo è spesso sentito come un tono di picco di altezza definita o un gruppo di toni. E questo è l'effetto Larsen: (dal nome del fisico Søren Absalon Larsen che per primo ne scoprì il principio), detto anche feedback acustico o più prosaicamente ritorno, è timbricamente riconoscibile come un tipico fischio stridente, che si sviluppa come abbiamo detto quando i suoni emessi da un altoparlante vengono captati con sufficiente "potenza di innesco" da un trasduttore (che può però anche essere oltre al microfono, un pick-up di uno strumento musicale elettrico, come una chitarra o un basso, o un trasduttore di altra natura).

SPL del segnale di fondo nella stanza

Essendo l'effetto Larsen innescato dal segnale di fondo di un ambiente che contribuisce a sua volta a delineare le condizioni iniziali dell'effetto stesso, la prima cosa che possiamo fare per conoscere meglio l'ambiente operativo dove andremo a condurre le analisi è misurare il livello di pressione sonora, o SPL (sound pressure level) del segnale di fondo della stanza, che viene misurato in Decibel (dB SPL). Il normale orecchio umano può rilevare suoni compresi tra 0 dB -soglia dell'udito- e circa 140 dB -soglia del dolore-, con suoni tra 120 dB e 140 dB che possono causare danni permanenti all'udito.
Il valore riportato di seguito è stato ricavato come media su un tempo di 4 minuti circa. ed è stato pesato con la curva di pesatura A. Il livello del segnale di fondo, misurato con la stanza chiusa è di **42 dBA** che rappresenta un buon dato per un ambiente domestico nel pomeriggio.

Il fonometro utilizzato per tale misurazione è della marca
V·RESOURCING [Portable Sound Level Meter],
e misura il valore in **dB** facendo riferimento alla curva di pesatura "**A**" (di 40 phon).

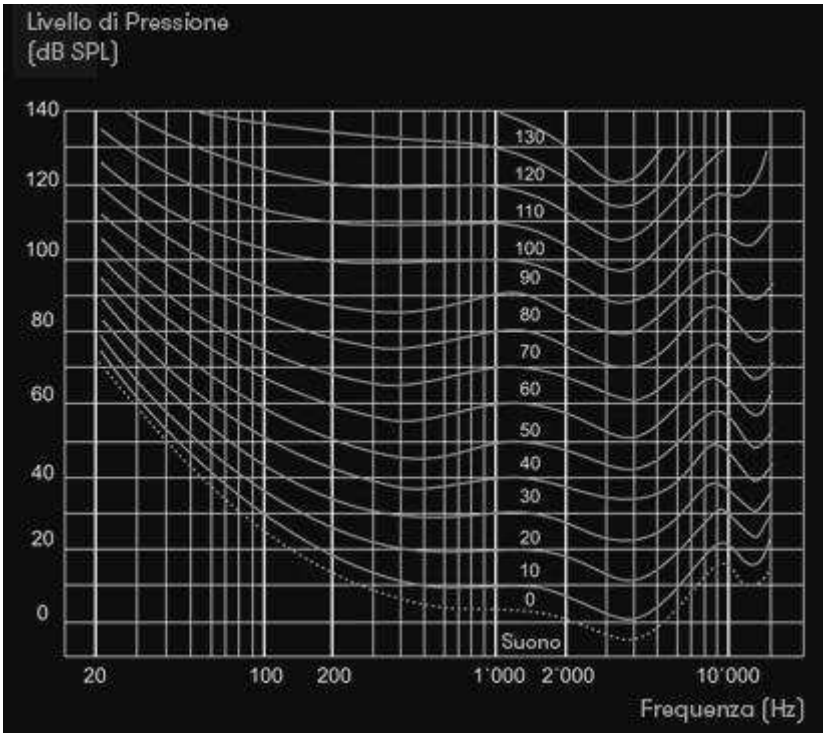
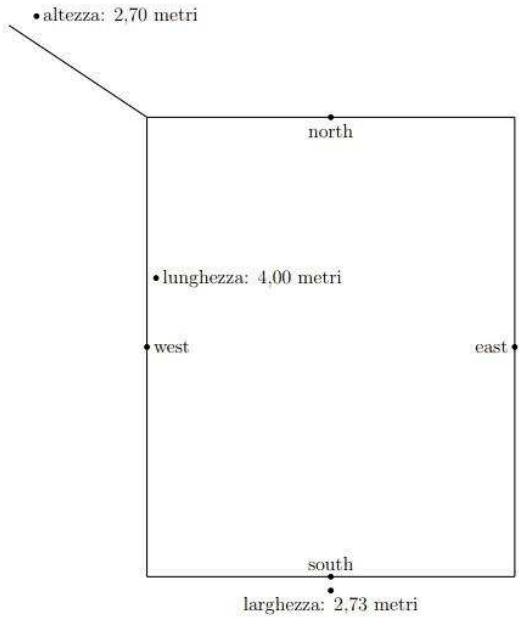


grafico reperibile dalla guida fonometri in sitografia[2]

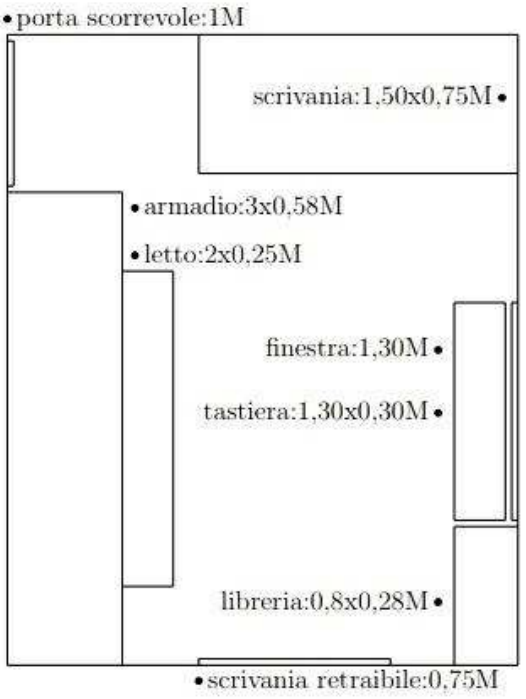
Mappa della stanza

La stanza dove verranno fatte le analisi dell'effetto Larsen si presenta al momento delle misurazioni come nelle mappe riportate qui, escludendo per il momento la presenza e il posizionamento della catena elettroacustica.

Stanza Vuota



Stanza Arredata



La catena elettroacustica verrà dunque posizionata al centro della stanza arredata, procedendo poi a diverse misurazioni che dovrebbero corrispondere a diversi effetti Larsen generati per ogni variazione delle posizioni di microfono e altoparlante.

La catena elettroacustica

La catena elettroacustica che verrà utilizzata per produrre il fenomeno del Larsen nella stanza è composta da 3 elementi, controllati da un algoritmo di DSP.
Fatta eccezione per il computer, gli elementi che costituiscono la catena sono:

- Uno studio monitor di riferimento full-range da 5.25" - 30W
  BEHRINGER AURATONE C50A
- Un Microfono di misura a condensatore a elettrete
  BEHRINGER ECM8000
- Una scheda audio multicanale
  FOCUSRITE SCARLETT 18i20

Ognuno di questi elementi contribuisce insieme alla stanza, sulla base delle sue caratteristiche fisiche, a determinare il carattere dell'effetto Larsen.
Le caratteristiche riportate dai costruttori nelle schede tecniche degli elementi hardware che compongono la catena elettroacustica sono le seguenti:

- **BEHRINGER AURATONE C50A**

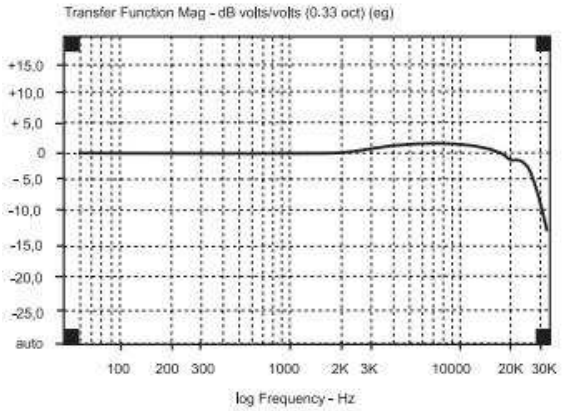
Informazioni sul prodotto - Behringer C50A

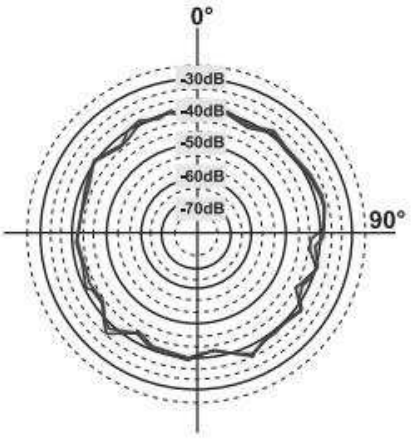
Caratteristiche:

- Build: Banda larga
- Interception distance: Near field
- Bluetooth: No
- Schemato magneticamente: Sì
- Room adjustment: No
- DSP: No
- Pair: No
- Wideband (inch): 5,25
- Total output in watts (RMS): 30
- Classe amplificatore: Classe D
- Frequency range: 90 - 17000Hz
- Risposta in frequenza da (Hz): 90
- Risposta in frequenza a (Hz): 17000
- Inputs analog: 3-Pole XLR, RCA, jack TRS
- Larghezza (mm): 165
- Profondità (mm): 198
- Altezza (mm): 165
- Weight (kg): 3,3

- **BEHRINGER ECM8000**

Type	electret condenser, omni-directional
Impedance	600 Ohms
Sensitivity	-60 dB
Frequency response	15 Hz to 20 kHz
Connector	gold-plated XLR
Phantom power	+15 V to +48 V
Weight	app. 120 g





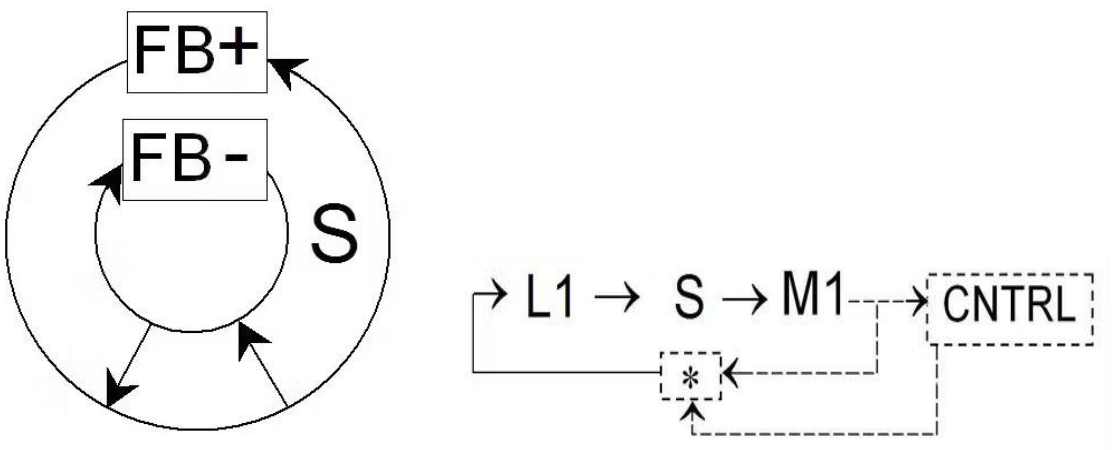
- **FOCUSRITE SCARLETT 18i20**

Microphone Inputs 1 to 8	
Dynamic Range	111 dB (A-weighted)
Frequency Response	20 Hz to 20 kHz ±0.1 dB
THD+N	< 0.0012% (minimum gain, -1 dBFS input with 22 Hz/22 kHz bandpass filter)
Noise EIN	–128 dB (A-weighted)
Maximum Input Level	+9 dBu (no PAD); +16 dBu (PAD selected); measured at minimum gain
Gain Range	56 dB
Input Impedance	3 kohms
Line Outputs 1 to 10	
Dynamic Range	108.5 dB (A-weighted)
Maximum Output Level (0 dBFS)	+15.5 dBu (balanced)
THD+N	< 0.002% (minimum gain, -1 dBFS input with 22 Hz/22 kHz bandpass filter)
Output Impedance	430 ohms

Il controbilanciamento del Feedback (Meccanismo LAR)

Tornando a citare un altro passo dall'articolo di Di Scipio per la rivista *ECHO*:[1]

<<The sound onset curve and the actual spectrum width will depend not only on the gain but on several factors (including the mentioned relative distance and position between M1 and L1, several acoustical properties of S, not to mention the technical features of a transducer systems such as M1 and L1, e.g. its frequency response, sensitivity, directivity, etc.). The minimal non-linear transfer in any sound-related component will in some way affect the feedback process, shaping the timbre and the dynamics of occasioned sounds. In extreme cases, limitations and non-linearities (especially in amplifiers and transducers) may cause signal saturation: the spectrum gets wider and increasingly more varied or complex (harmonic distortion); amplitude, on the other hand, won't really increase indefinitely, it can't grow beyond inherent limits (relative to the maximum power or the amplification as well as to the sensitivity of transducers and the elasticity of included membranes).
In common sound engineering practice, audible feedback phenomena are a nuisance, a problem one should get rid of or substantially minimize. When direct level manipulation is not enough, one resorts to hard-limiting circuits, 'feedback killers' and alike devices (for an overview, see (Waterschoot and Moonen 2011)). In a different attitude, one may instead consider feedback as a resource, a deliberately designed sound-making mechanism one can play with.
To that aim, let's pull in a real-time processing computer and make it calculate the Root Mean Square (RMS) over short, windowed segments of the microphone signal (that's like an averaging filter, i.e. a simple finite-impulse-response filter with cutoff frequency close to, but higher than 0 Hz). This provides a continuing estimation of the input signal amplitude, allowing us to implement a basic 'amplitude following' technique – the simplest among 'feature extraction' techniques (Widmer et al. 2005). We then complement the RMS values and use the complemented signal to drive the FB+'s gain. In the context of my discussion, this low-frequency signal stands for a typical control signal (CNTRL). As illustrated in figure 2, the operation requires a second feedback loop embedded in the first one.>>



Schemi del meccanismo LAR di Agostino Di Scipio

Traducendo il testo di Di Scipio:
Il feedback, entrando in un regime di autoscillazione, tende a crescere all'infinito; tuttavia nella pratica non aumenterà davvero all'infinito, infatti non può crescere oltre i limiti intrinseci (relativi alla potenza massima, all'amplificazione, nonché alla sensibilità dei trasduttori e all'elasticità delle membrane incluse)... per utilizzare il feedback come una risorsa, evitando una crescita che porti alla saturazione dei sistemi, possiamo far calcolare al computer in tempo reale il Root Mean Square (RMS) su segmenti brevi e finestrati del segnale captato dal microfono (è come un filtro di media, ovvero un semplice filtro a risposta all'impulso finita con frequenza di taglio vicina, ma superiore allo 0 Hz). Ciò fornisce una stima continua dell'ampiezza del segnale di ingresso, consentendoci di implementare una

tecnica di base di ‘amplitude following’ - la più semplice tra le tecniche di ‘feature extraction’ del segnale (MIR).

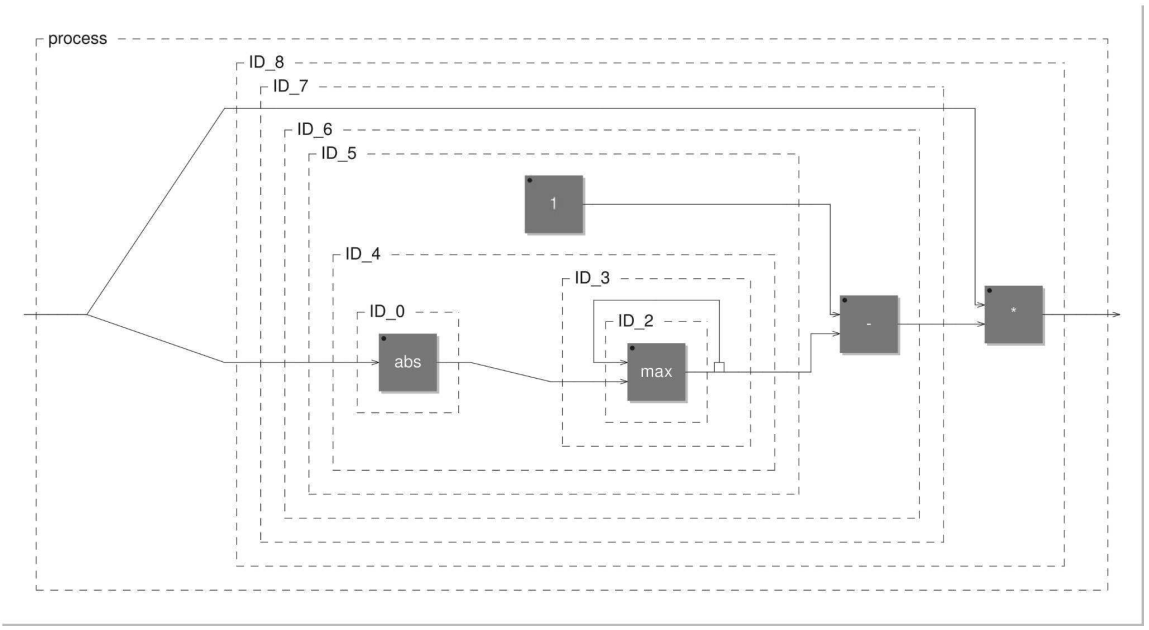
Dunque alla luce delle considerazioni appena fatte, è necessario un elemento per tenere l’effetto Larsen in uno stato stazionario una volta innescato, senza terminare nella saturazione dei sistemi che lo generano; questo elemento è il controbilanciamento del guadagno del larsen, che nella sua logica Di Scipio chiama col nome di LAR, e che può essere implementato con diverse tecniche di ‘amplitude following’ in DSP.

Algoritmi di controbilanciamento in FAUST

Ci sono diversi modi per realizzare un algoritmo di controbilanciamento del feedback in tempo reale nel linguaggio di programmazione FAUST.
Alcuni di questi possono riguardare tecniche di filtraggio automatizzate (adattive) che eliminino dallo spettro la presenza del Larsen *'Larsen Suppression'*,
o come nel nostro caso d'interesse possono riguardare controbilanciamenti in ampiezza automatizzati che non permettano al Feedback di avere un guadagno troppo alto e giungere dunque ad uno stato di saturazione del sistema.
Nel secondo caso possono dunque essere implementati algoritmi di *'amplitude following'* in tempo reale con dei filtri che operano la media *'RMS'*, *filtri 'peakholder'* che mantengono il valore di picco, o di altra natura.

Controbilanciamento via Peakholder

Il modo più semplice per mantenere l'effetto Larsen in uno stato stazionario, è attraverso un valore costante in retroazione che controbilancia l'ampiezza del segnale in ingresso; la costante è determinata dal valore del campione più alto, *'peakholder'* che mantiene il valore di picco per un tempo infinito. L'algoritmo presenta comunque alcuni problemi: non possedendo una funzione di *smoothing* del segnale, si verificano problemi di segnali di differenza a banda molto larga che possono generare *aliasing* e contributi spuri che tendono a permanere nel segnale complessivo.



Topologia dell'algoritmo LAR via Peakholder

The *LARPeakholder* program evaluates the signal transformer denoted by `process`, which is mathematically defined as follows:

1. Output signal y such that

$$y(t) = x(t) \cdot (1 - r_1(t))$$

2. Input signal x
3. Intermediate signal r_1 such that

$$r_1(t) = \max\left(r_1(t-1), |x(t)|\right)$$

Definizione Matematica dell'algoritmo LAR via Peakholder

```
// FAUST standard library
import("stdfaust.lib");


// Peakholder IIR - Loop Stage
        Peakholder = _ : abs <: loop ~_
                with{
                        loop = ((_,_) : max);
                };

// Counterbalancing
        LARPeakholder = _ <: *(1-Peakholder);

process = LARPeakholder;
```

Codice in FAUST

Controbilanciamento via RMS

Per questi motivi un modo più efficace per mantenere il Larsen in uno stato stazionario, è attraverso algoritmi che effettuano la media '*RMS*' in tempo reale. Questo tipo di algoritmo viene descritto nel libro: *Digital Audio Signal Processing - Second Edition di Udo Zölzer.* [3] Riporto qui di seguito i passi che ci interessano.

7.3 Dynamic Behavior

Besides the static curve of dynamic range control, the dynamic behavior in terms of attack and release times plays a significant role in sound quality. The rapidity of dynamic range control depends also on the measurement of PEAK and RMS values [McN84, Sti86].

7.3.1 Level Measurement

Level measurements [McN84] can be made with the systems shown in Figs 7.4 and 7.5. For PEAK measurement, the absolute value of the input is compared with the peak value $x_{\mathrm{PEAK}}(n)$. If the absolute value is greater than the peak value, the difference is weighted with the coefficient AT (attack time) and added to $(1-AT)\cdot x_{\mathrm{PEAK}}(n-1)$. For this attack case $|x(n)| > x_{\mathrm{PEAK}}(n-1)$ we get the difference equation (see Fig. 7.4)

$$x_{\mathrm{PEAK}}(n) = (1-AT)\cdot x_{\mathrm{PEAK}}(n-1) + AT\cdot|x(n)| \tag{7.11}$$

with the transfer function

$$H(z) = \frac{AT}{1-(1-AT)z^{-1}}. \tag{7.12}$$

If the absolute value of the input is smaller than the peak value $|x(n)| \le x_{\mathrm{PEAK}}(n-1)$ (the release case), the new peak value is given by

$$x_{\mathrm{PEAK}}(n) = (1-RT)\cdot x_{\mathrm{PEAK}}(n-1) \tag{7.13}$$

with the release time coefficient RT. The difference signal of the input will be muted by the nonlinearity such that the difference equation for the peak value is given according to (7.13). For the release case the transfer function

$$H(z) = \frac{1}{1-(1-RT)z^{-1}} \tag{7.14}$$

is valid. For the attack case the transfer function (7.12) with coefficient AT is used and for the release case the transfer function (7.14) with the coefficient RT. The coefficients (see Section 7.3.3) are given by

$$AT = 1-\exp\left(\frac{-2.2T_S}{t_a/1000}\right), \tag{7.15}$$

$$RT = 1-\exp\left(\frac{-2.2T_S}{t_r/1000}\right), \tag{7.16}$$

where the attack time t_a and the release time t_r are given in milliseconds (T_S sampling interval). With this switching between filter structures one achieves fast attack responses for increasing input signal amplitudes and slow decay responses for decreasing input signal amplitudes.

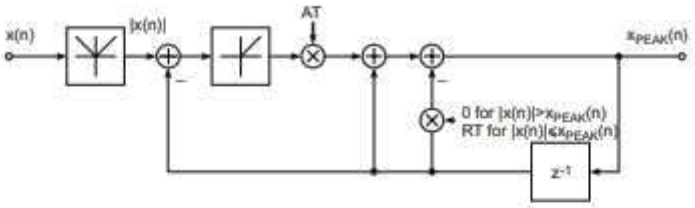


Figure 7.4 PEAK measurement.

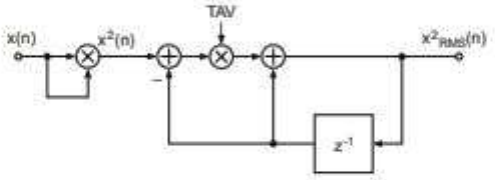


Figure 7.5 RMS measurement (TAV = averaging coefficient).

The computation of the RMS value

$$x_{\mathrm{RMS}}(n) = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1} x^2(n-i)} \tag{7.17}$$

over N input samples can be achieved by a recursive formulation. The RMS measurement shown in Fig. 7.5 uses the square of the input and performs averaging with a first-order low-pass filter. The averaging coefficient

$$\mathrm{TAV} = 1 - \exp\left(\frac{-2.2T_A}{t_M/1000}\right) \tag{7.18}$$

is determined according to the time constant calculation discussed in Section 7.3.3, where t_M is the averaging time in milliseconds. The difference equation is given by

$$x^2_{\mathrm{RMS}}(n) = (1 - \mathrm{TAV}) \cdot x^2_{\mathrm{RMS}}(n-1) + \mathrm{TAV} \cdot x^2(n) \tag{7.19}$$

with the transfer function

$$H(z) = \frac{\mathrm{TAV}}{1 - (1 - \mathrm{TAV})z^{-1}}. \tag{7.20}$$

Riportando questo algoritmo in un codice Faust apparirà come segue:

```
// FAUST standard library
import("stdfaust.lib");

// RMS with independent attack and release time
RMS(att,rel,x) = loop ~ _ : sqrt
    with {
        loop(y) = (1.0 - coeff) * x * x + coeff * y
            with {
                attCoeff = exp(-2.0 * ma.PI * ma.T / att);
                relCoeff = exp(-2.0 * ma.PI * ma.T / rel);
                coeff = ba.if(abs(x) > y, attCoeff, relCoeff);
            };
    };
// Counterbalancing
LARRMS(att,rel,z) = z*(1-(z:RMS(att,rel)));
process = LARRMS(0.01,0.01) <: _,_;
```

Codice in FAUST

Test per diversi valori di attacco nel controbilanciamento

Ho confrontato diversi risultati nella generazione del Larsen per diversi valori di attacco dell'algoritmo RMS di *Udo Zölzer* utilizzato come controbilanciamento; registrando 10 campioni che vanno da un minimo di *1 millisecondo* ad un massimo di *800 millisecondi*.

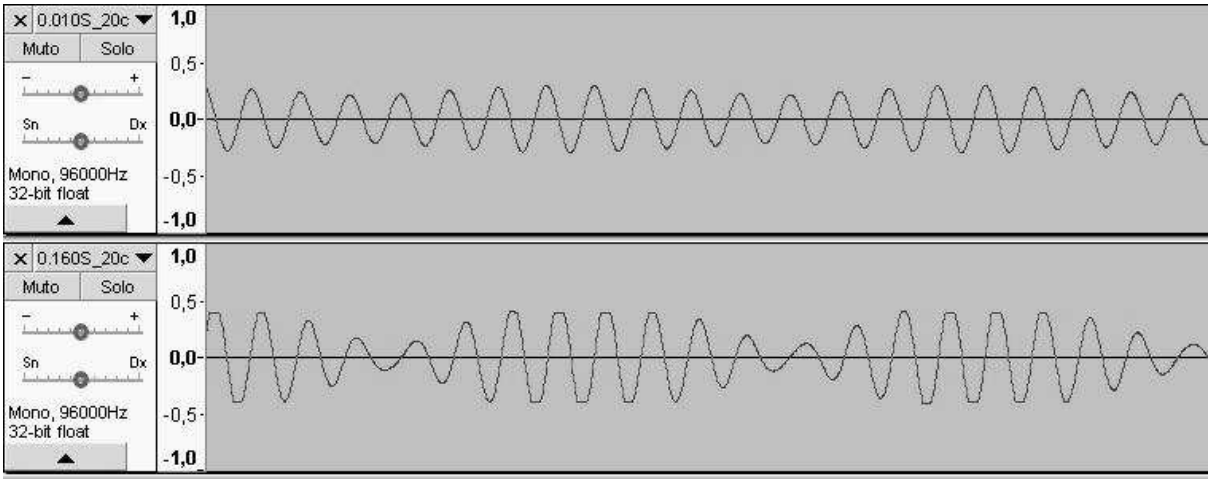
L'elenco delle registrazioni fatte per questa analisi è il seguente:

1 registrazione per un tempo di attacco di - 0.001 Secondi
1 registrazione per un tempo di attacco di - 0.005 Secondi
1 registrazione per un tempo di attacco di - 0.010 Secondi
1 registrazione per un tempo di attacco di - 0.020 Secondi
1 registrazione per un tempo di attacco di - 0.040 Secondi
1 registrazione per un tempo di attacco di - 0.080 Secondi
1 registrazione per un tempo di attacco di - 0.160 Secondi
1 registrazione per un tempo di attacco di - 0.300 Secondi
1 registrazione per un tempo di attacco di - 0.500 Secondi
1 registrazione per un tempo di attacco di - 0.800 Secondi

Quello che varia in queste registrazioni al crescere dei tempi di attacco dell'algoritmo di *Zölzer*, è l'efficienza del controbilanciamento. Per *t_a* più lunghi i sistemi vanno in saturazione (*clipping*), per *t_a* brevi non si arriva alla saturazione dei sistemi e il controbilanciamento è efficace, e per *t_a* troppo brevi come visto in precedenza generiamo *aliasing*. Fenomeni di *clipping* del segnale iniziano a manifestarsi in queste registrazioni a partire da quella con un *t_a* pari a *160 millisecondi* rendendoci consapevoli del fatto che l'ultimo valore efficace nella nostra situazione è quello di *80 millisecondi*.

A partire da queste considerazioni potremmo dunque ipotizzare tre soglie che corrispondono all'efficacia dell'algoritmo per i diversi *t_a*

aliasing (valore non efficace) < valore efficace < clipping (valore non efficace)



Visualizzazione in Audacity della forma d'onda di un segnale nella soglia efficace e uno in clip

Dunque per tutte le registrazioni fatte a seguito per le analisi è stato scelto di utilizzare un *t_a* comune di *10 millisecondi*.

Le Misurazioni

Per procedere alla misurazione degli effetti Larsen ho utilizzato due computer (Linux) connessi via protocollo ssh. Il primo è il computer fisso collegato alla catena elettroacustica, che ha il compito di processare e registrare il segnale prodotto. Il secondo è un computer portatile collegato in ssh al primo, che ha invece il compito di aprire e chiudere la catena elettroacustica tramite alsamixer. Lo scopo di questo meccanismo è quello di non essere fisicamente presenti in stanza nel momento della generazione del Larsen.

```
#! /bin/bash
# script per avviare una connessione ssh col PC fisso nella stanza.
# per controllare l'ip di rete del computer usare - $ hostname -I
# in $ ssh -p = il numero della porta, per default 22
ssh utente@server -p 22
```

Esempio Script sh per connessione ssh

Avviato lo script e connesso da remoto al computer il comando per aprire alsamixer da terminale: (da tastiera, aperto il mixer, M per mutare o attivare i canali).

```
$ alsamixer
```

In ambiente Windows invece esistono dei software dedicati che permettono il controllo remoto, in versione desktop e semplificata, come ad esempio *TeamViewer*.

Fatto questo procedo dunque a generare dei Larsen documentando per ogni registrazione le posizioni del microfono e dell'altoparlante.

per le registrazioni ho scelto un punto fisso, che corrisponde ad una posizione al centro della stanza arredata che dista dalle pareti:

parete nord- 2,00M
parete ovest - 1,80M
parete est - 1,18M
parete sud - 2,00M

le registrazioni sono state fatte confrontando la distanza fra il punto fisso utilizzato per posizionare il cono dell'altoparlante, ed un nuovo punto che corrisponde invece alla posizione della capsula del microfono di misurazione traslata di volta in volta di 20cm in direzione dell'asse della parete sud (dunque rispetto al cono).

L'elenco delle registrazioni fatte per le analisi è il seguente:

4 registrazioni - a 20 cm di distanza con l'altoparlante rivolto verso la parete sud ed il microfono verso la parete nord
2 registrazioni - a 40 cm di distanza con l'altoparlante rivolto verso la parete sud ed il microfono verso la parete nord
2 registrazioni - a 60 cm di distanza con l'altoparlante rivolto verso la parete sud ed il microfono verso la parete nord
2 registrazioni - a 80 cm di distanza con l'altoparlante rivolto verso la parete sud ed il microfono verso la parete nord
2 registrazioni - a 1 metro di distanza con l'altoparlante rivolto verso la parete sud ed il microfono verso la parete nord
2 registrazioni - a 1 metro di distanza con l'altoparlante e il microfono rivolti verso la parete nord
2 registrazioni - a 1 metro di distanza con l'altoparlante rivolto verso la parete nord ed il microfono verso la parete sud

in ognuna di queste registrazioni e per queste posizioni, emergono diversi effetti Larsen con diversi modi di risonanza, che andremo a confrontare ed analizzare.

Le analisi in tempo differito

I segnali audio descrivono le variazioni di pressione che il nostro orecchio percepisce come suoni. Ad un certo punto fra il XVIII ed il XIX secolo, un matematico e fisico francese, Jean Baptiste Joseph Fourier, dimostrò che qualsiasi segnale periodico può essere scomposto in una somma di (infiniti) segnali sinusoidali, dando vita a quella che chiamiamo oggi come Serie di Fourier. Il problema con i segnali musicali è che il suono è sostanzialmente una funzione quasi periodica o aperiodica non stazionaria; l'analisi di un segnale non "stazionario" deve fornire quindi informazioni su quando le diverse componenti in frequenza sono attive e con quale contenuto energetico. E' quindi opportuno adottare un modello di analisi che fornisca informazioni sul contenuto spettrale di un segnale, ma anche su come questo varia nel tempo.
La trasformata di Fourier a Tempo Breve (STFT Short Time Fourier Transform) fornisce l'analisi spettrale di un segnale per ognuno dei segmenti temporali (*frame*) in cui viene suddiviso.
In pratica, la procedura per calcolare la STFT consiste nel dividere un segnale di una certa durata in segmenti più brevi di uguale durata e quindi calcolare la DFT (*Discrete Fourier Transform*)su ciascuno di quei segmenti. Ogni segmento di analisi restituisce un vettore di numeri complessi che, trasformato in coordinate polari, rappresenta lo spettro dei moduli e quello delle fasi iniziali; la rappresentazione dello spettro dei moduli dei segmenti consecutivi mappato su una scala di colori sul piano congiunto tempo-frequenza, fornisce quello che si chiama spettrogramma, ovvero l'evoluzione nel tempo dello spettro del segnale.
Con la libreria PyACA per Python realizzata da Alexander Lerch e che accompagna il suo libro sull'ACA: An Introduction to Audio Content Analysis[4], abbiamo la possibilità di poter fare STFT su delle registrazioni ed estrarne dei grafici di riferimento, e più in generale astrazione delle informazioni musicali in tempo differito su delle registrazioni, potendo dunque ricavare da questi algoritmi presenti nella libreria informazioni utili per la nostra analisi.

Analisi: Forma d'onda, RMS, Spettrogramma

Le prime analisi che ho condotto tramite pyACA esportando i relativi campioni dell'intera registrazione sono state: la rappresentazione della forma d'onda, il calcolo RMS sull'intera curva, e la proiezione in uno spettrogramma tramite l'algoritmo di STFT. Tutti i codici Python con le loro librerie sono stati scritti per essere avviati tramite un terminale bash.

```
$ sudo python3 nome-dello-script.py
```

I codici Python scritti per esportare questi grafici sono i seguenti. A seguire i risultati delle analisi.

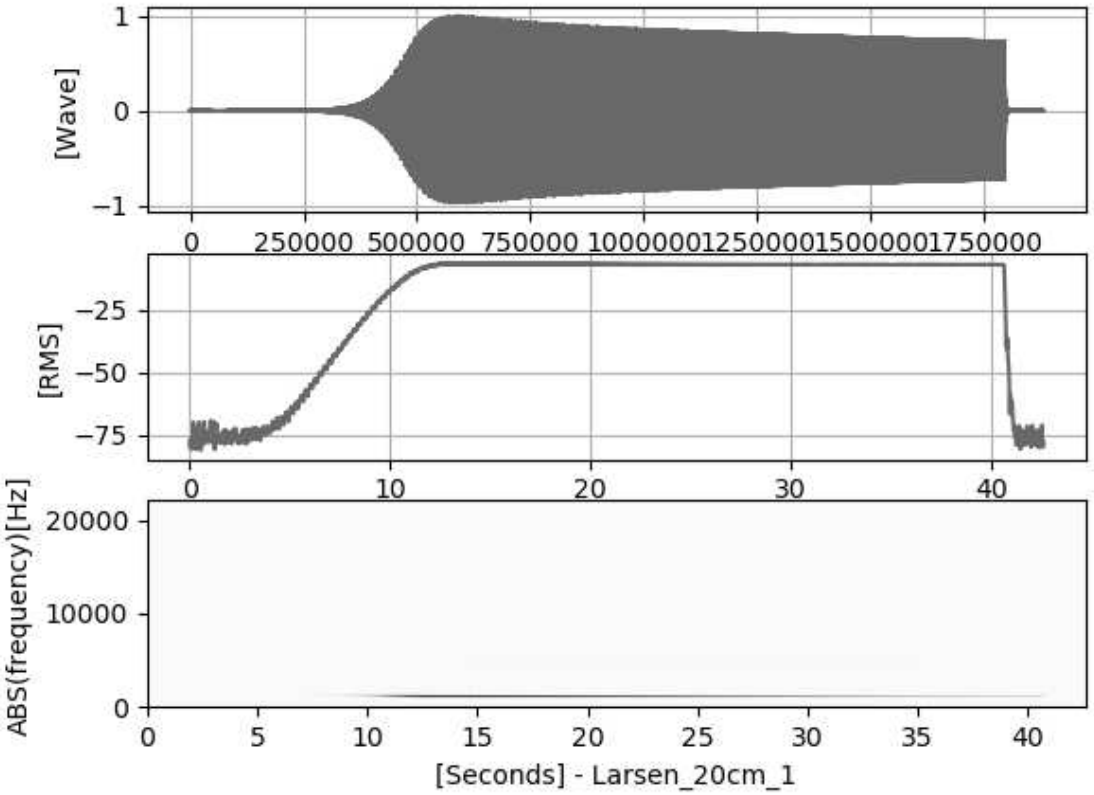
```
# CALCOLO RMS E RAPPRESENTAZIONE DELLA FORMA D'ONDA E ABS STFT


# librerie
import numpy as np
import sys
import pyACA
import matplotlib.pyplot as plt
from scipy.io.wavfile import read
from scipy import signal


# leggiamo un file audio mono importandolo da terminale
audiofile = input("Enter your file name (without extension):")

# print informazioni su file di testo
#with open(audiofile+"-INFO.txt", 'w') as fwr:
    #fwr.writelines('readme')

# leggiamo il file audio e ne stampiamo la durata
[fs, x] = read(audiofile+".wav")
print("SR: ",fs,"\ndimensione del file audio in campioni: ", np.size(x))
#fwr.writelines("SR: ",fs,"\ndimensione del file audio in campioni: ", np.size(x))

# normalizziamo il file audio(a 0 dB)
# dividendo il file per il modulo del suo massimo
x = x/np.max(abs(x))

t = np.arange(0,np.size(x))/fs

#calcoliamo gli RMS
[RMS, t] = pyACA.computeFeature("TimeRms", x, fs, iBlockLength=1024, iHopLength=512)

#calcoliamo la matrice STFT
f, t2, STFT = signal.stft(x, fs, nperseg=4096, noverlap=512, window='blackmanharris')
# window, documentazione: https://docs.scipy.org/doc/scipy/reference/signal.windows.html


# plot
# numero di grafici verticali, numero di finestre orizzontali, numero progressivo
plt.subplot(3,1,1)
plt.grid()
plt.plot(x)
plt.ylabel('[Wave]')
plt.xlabel('')

plt.subplot(3,1,3)
plt.pcolormesh(t2, f, np.abs(STFT), vmin=0, vmax=np.max(abs(STFT)*0.1), shading='auto',
cmap='Blues')
plt.xlabel('[Seconds] - '+audiofile)
plt.ylabel('ABS(frequency)[Hz]')

plt.subplot(3,1,2)
plt.grid()
plt.plot(t, RMS)
plt.ylabel('[RMS]')
plt.xlabel('')
```

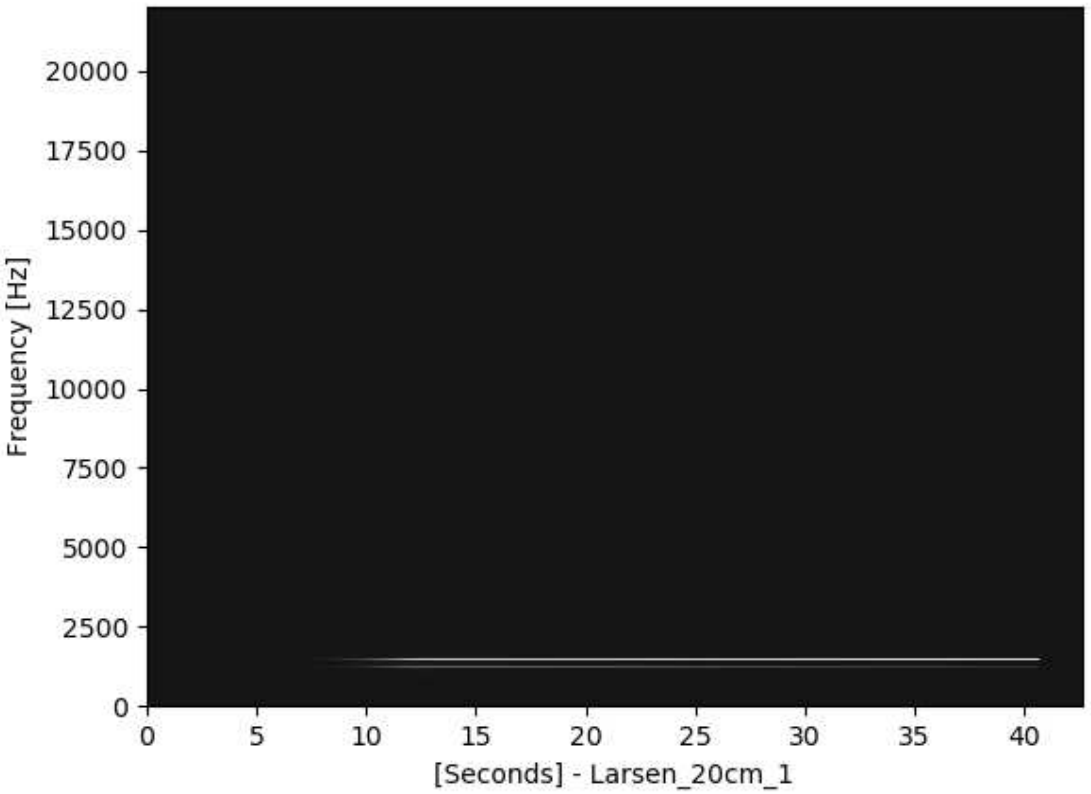
```
# ------------------------------------- FIGURE

# nomina output
nomeplot = audiofile+"-RMS-STFT-Wave.png"

# salva output
plt.savefig(nomeplot)
```

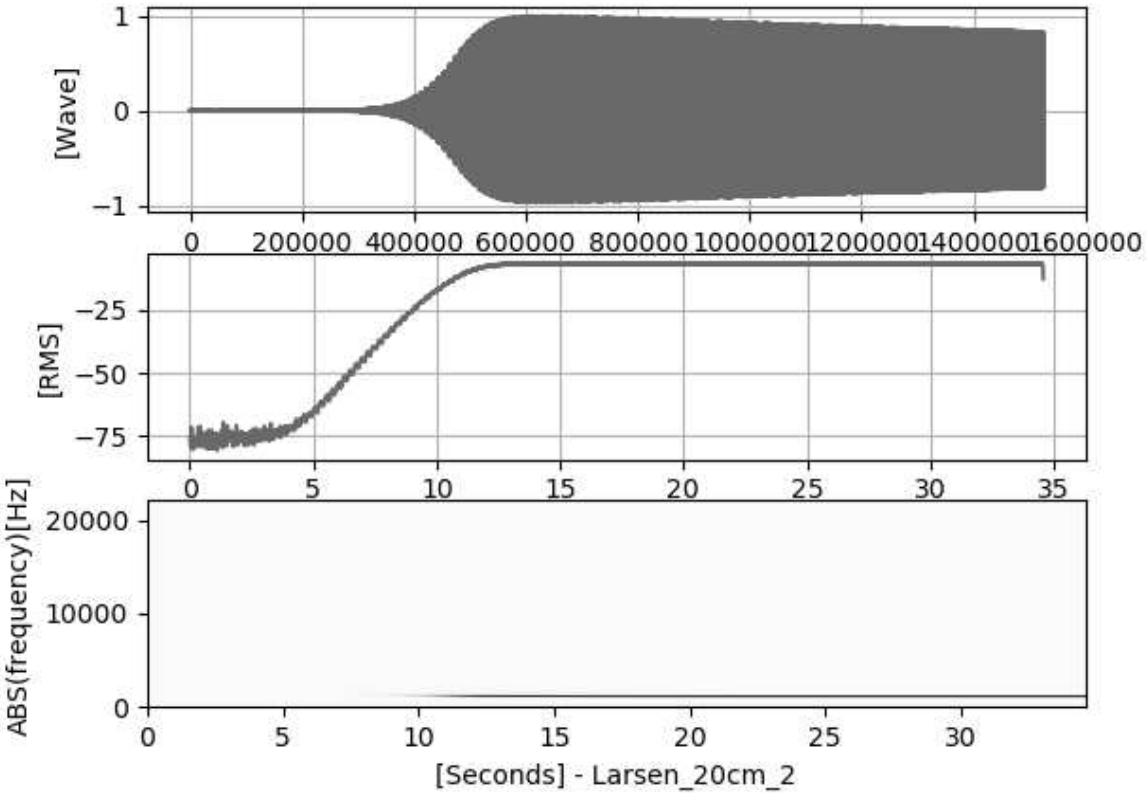
Codice Python per estrazione forma d'onda e calcolo RMS + ABS(STFT)

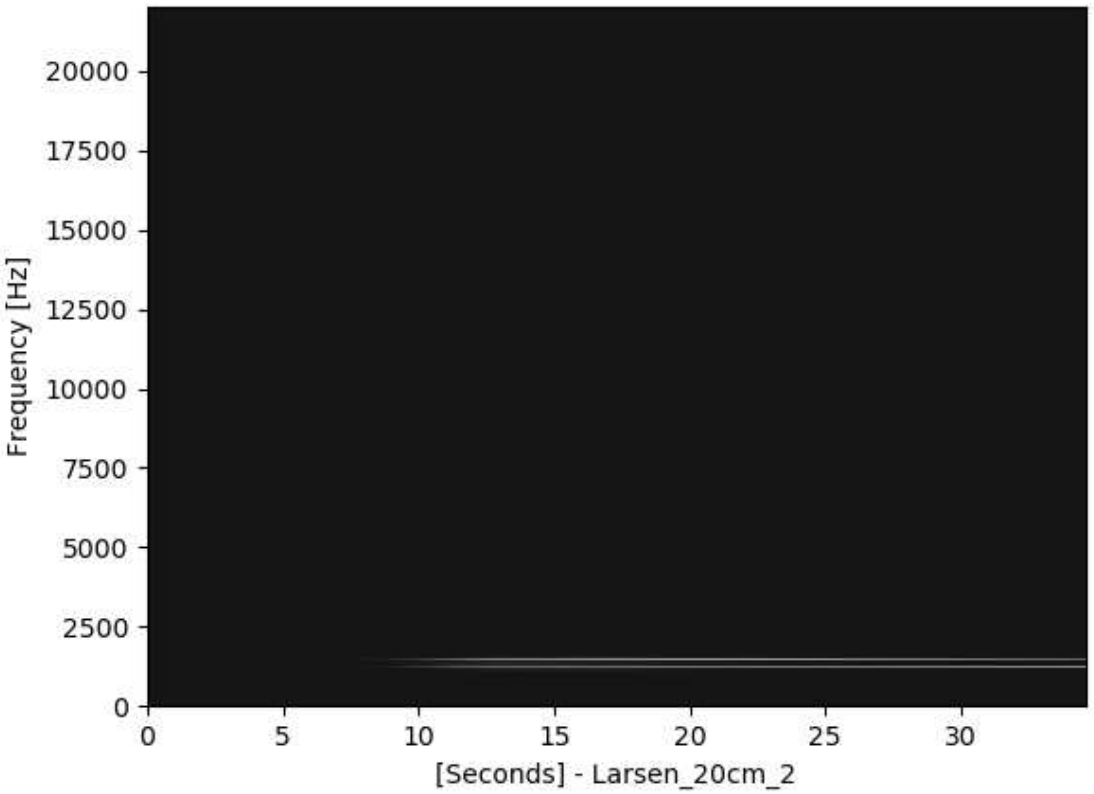
```
# CALCOLO STFT, nel Plot max = ABS


# librerie
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt
from scipy.io.wavfile import read

# leggiamo un file audio mono importandolo da terminale
audiofile = input("Enter your file name (without extension):")

#leggiamo un file audio(mono)
[fs, x] = read(audiofile+".wav")
# print su terminale: samplerate, dimensione del file audio in campioni
print("SR: ",fs,"\ndimensione del file audio in campioni: ", np.size(x))

#normalizziamo il file audio(a 0 dB)
x = x/np.max(abs(x))

#calcoliamo la matrice STFT
f, t, STFT = signal.stft(x, fs, nperseg=4096, noverlap=512, window='blackmanharris')
# window, documentazione: https://docs.scipy.org/doc/scipy/reference/signal.windows.html

# nomina output
nomeplot = audiofile+"-STFT.png"


# plot
plt.figure(1)
plt.pcolormesh(t, f, np.abs(STFT), vmin=0, vmax=np.max(abs(STFT)), shading='auto')
plt.xlabel('[Seconds] - '+audiofile)
plt.ylabel('Frequency [Hz]')

# salva output
plt.savefig(nomeplot)
```

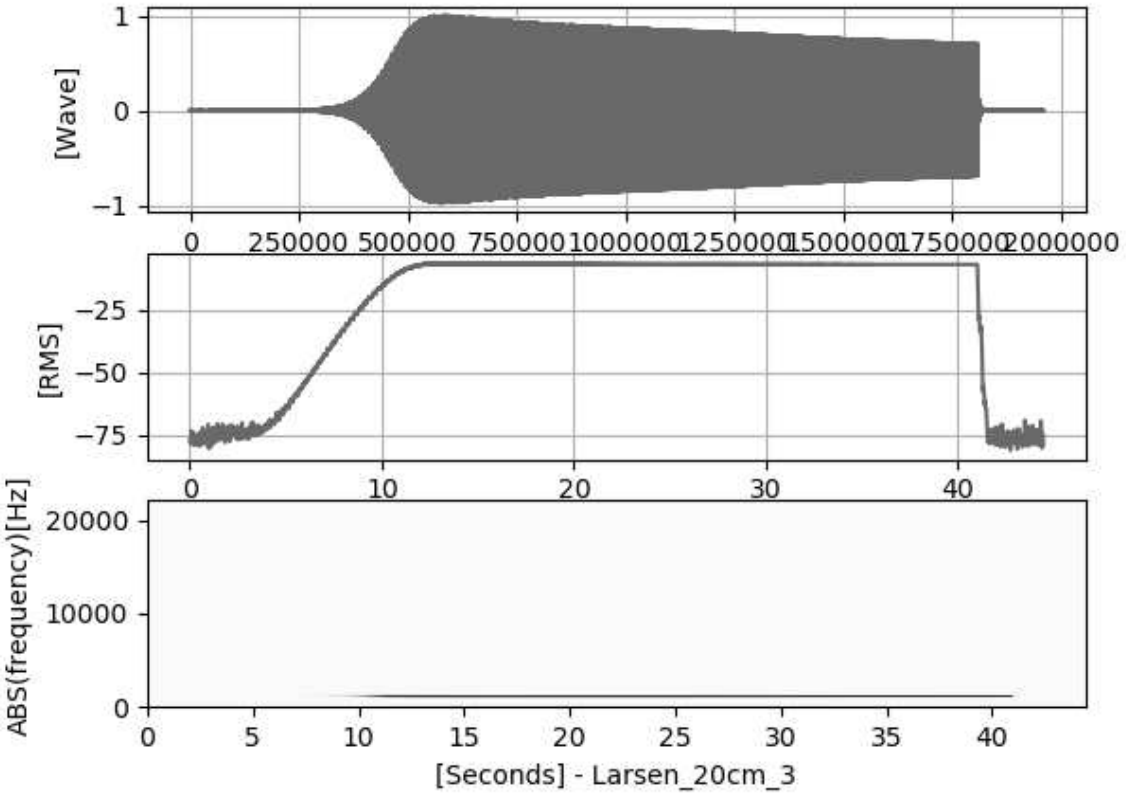
Codice Python per la rappresentazione dello Spettrogramma tramite STFT (ABS)

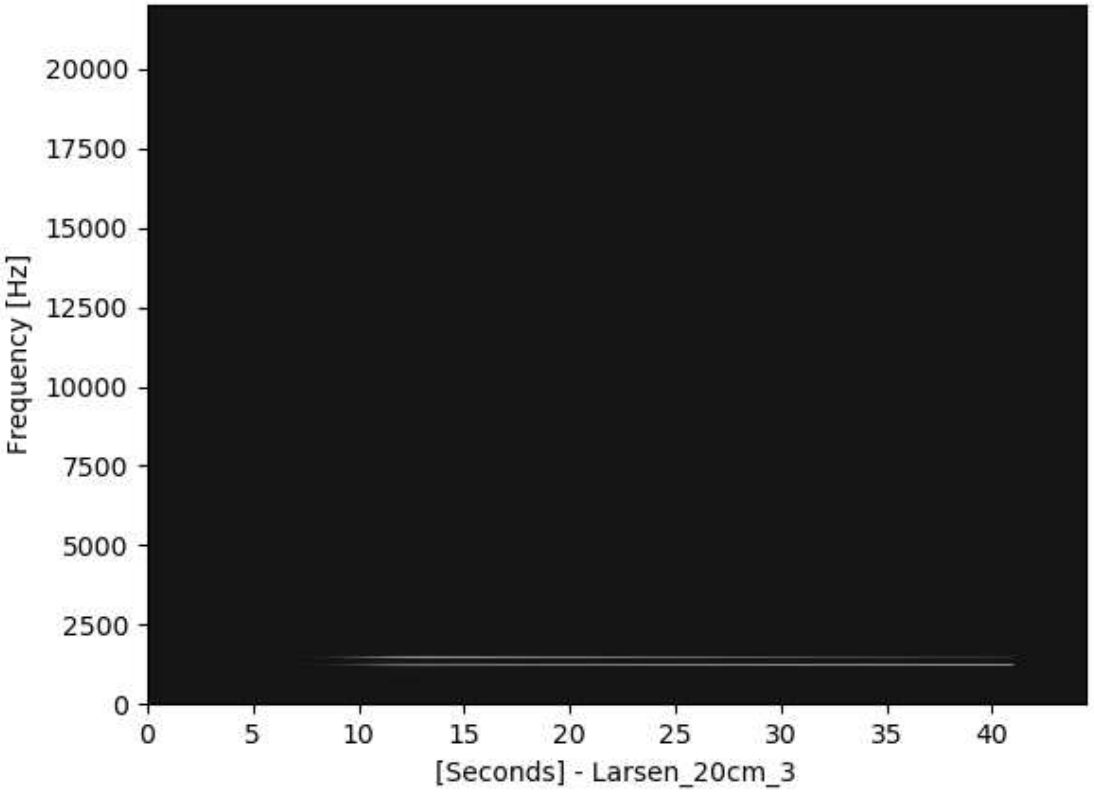

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_1

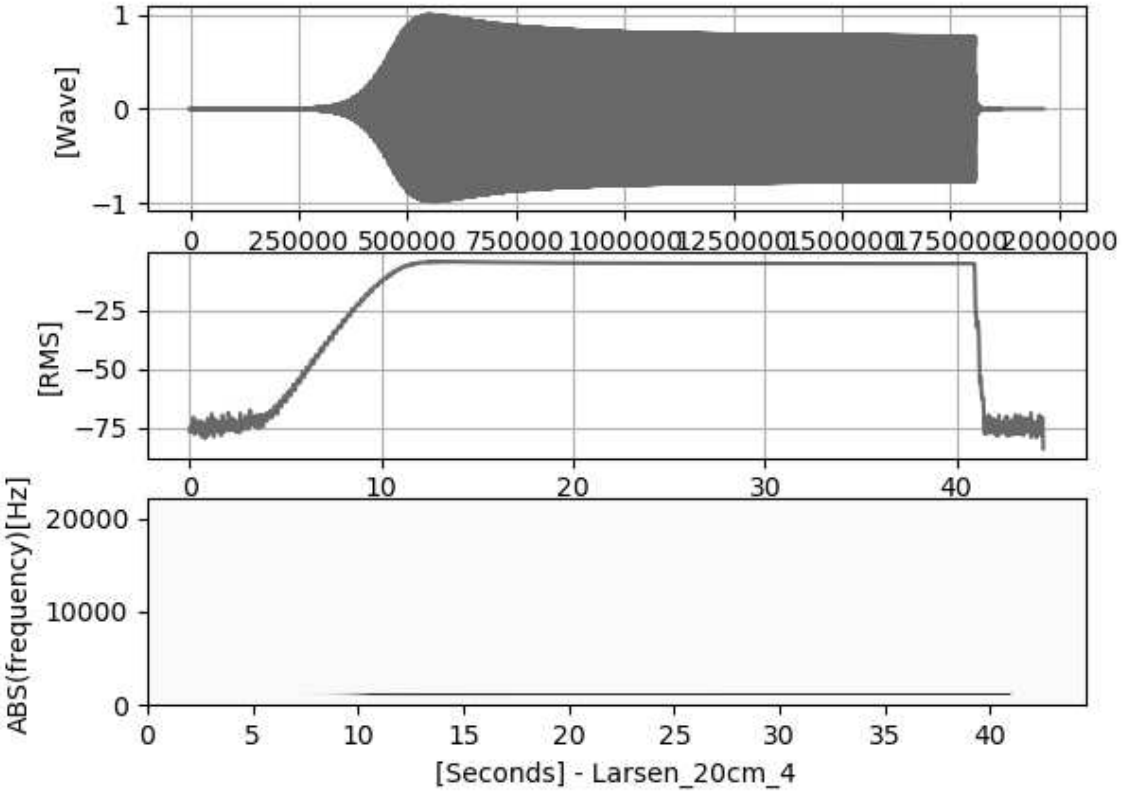

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_1

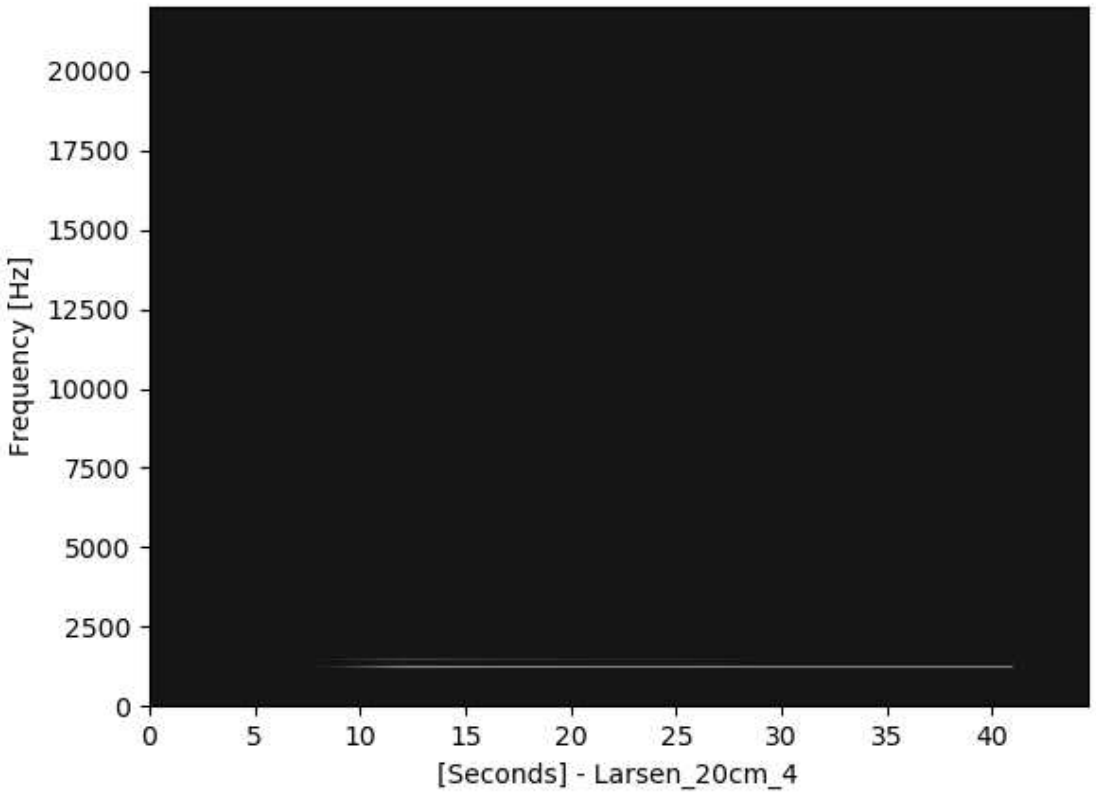

[Wave]
1
0
−1
0
200000
400000
600000
800000
1000000
1200000
1400000
1600000
[RMS]
−25
−50
−75
0
5
10
15
20
25
30
35
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
[Seconds] - Larsen_20cm_2

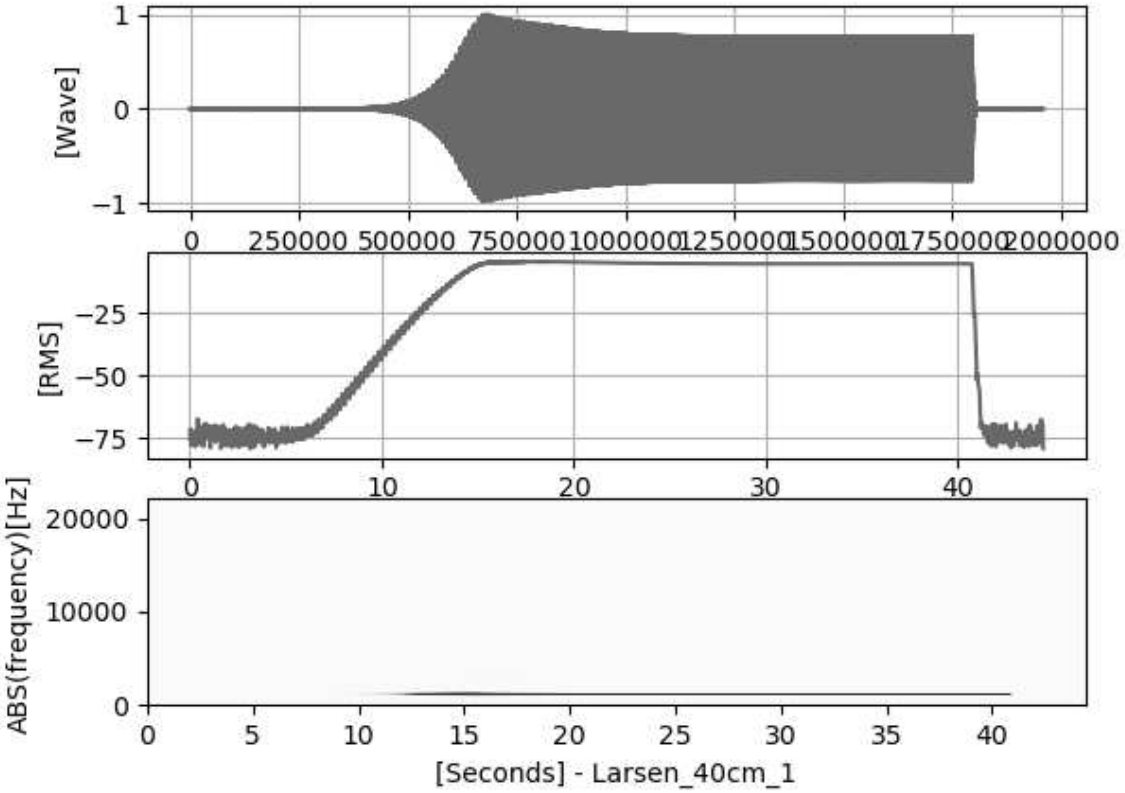

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
[Seconds] - Larsen_20cm_2

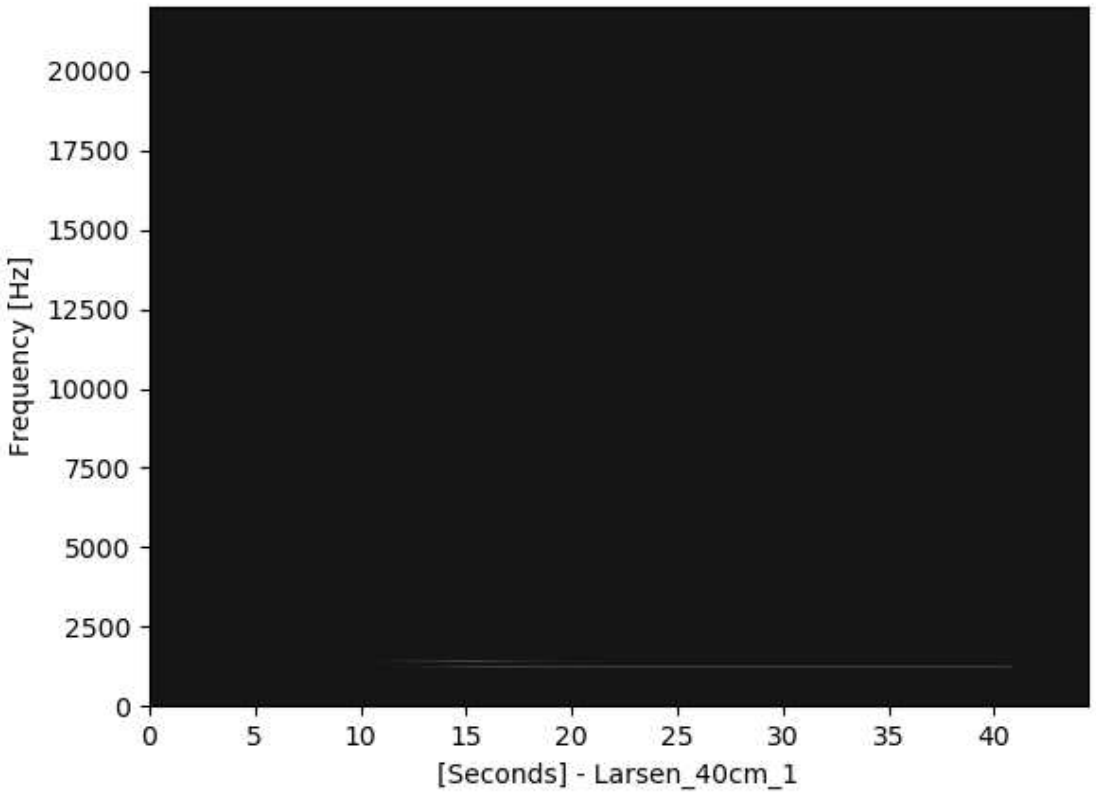

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_3

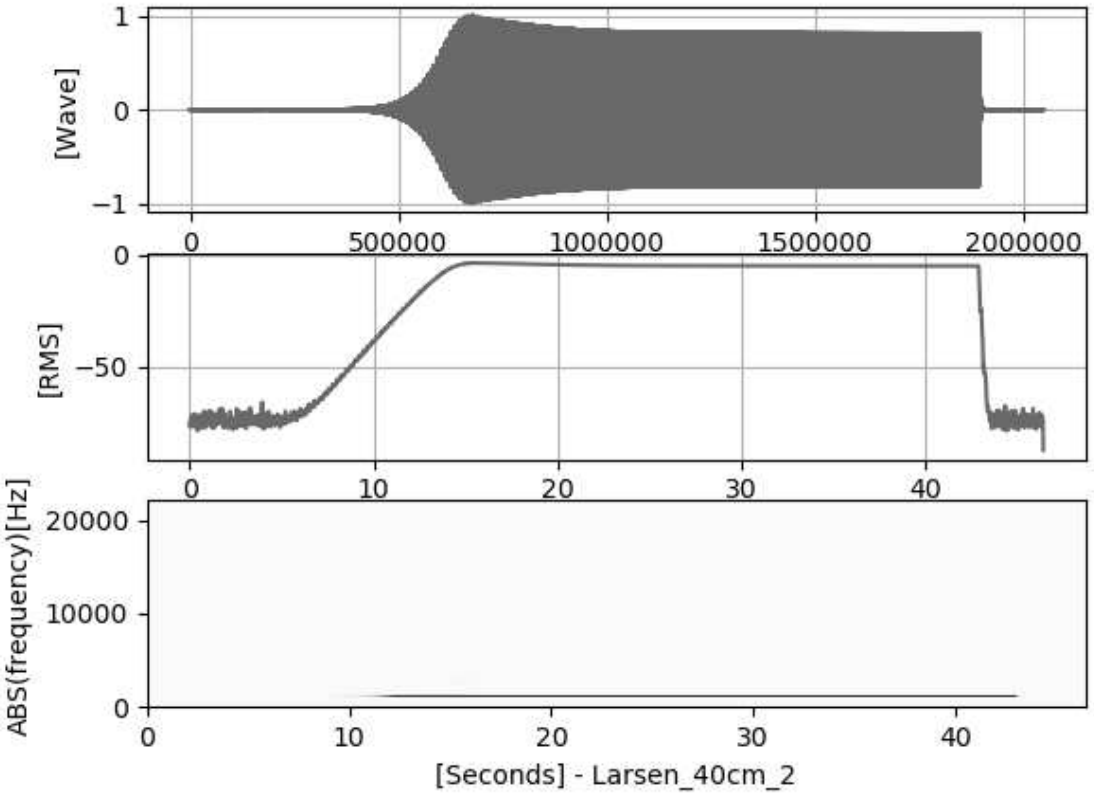

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_3

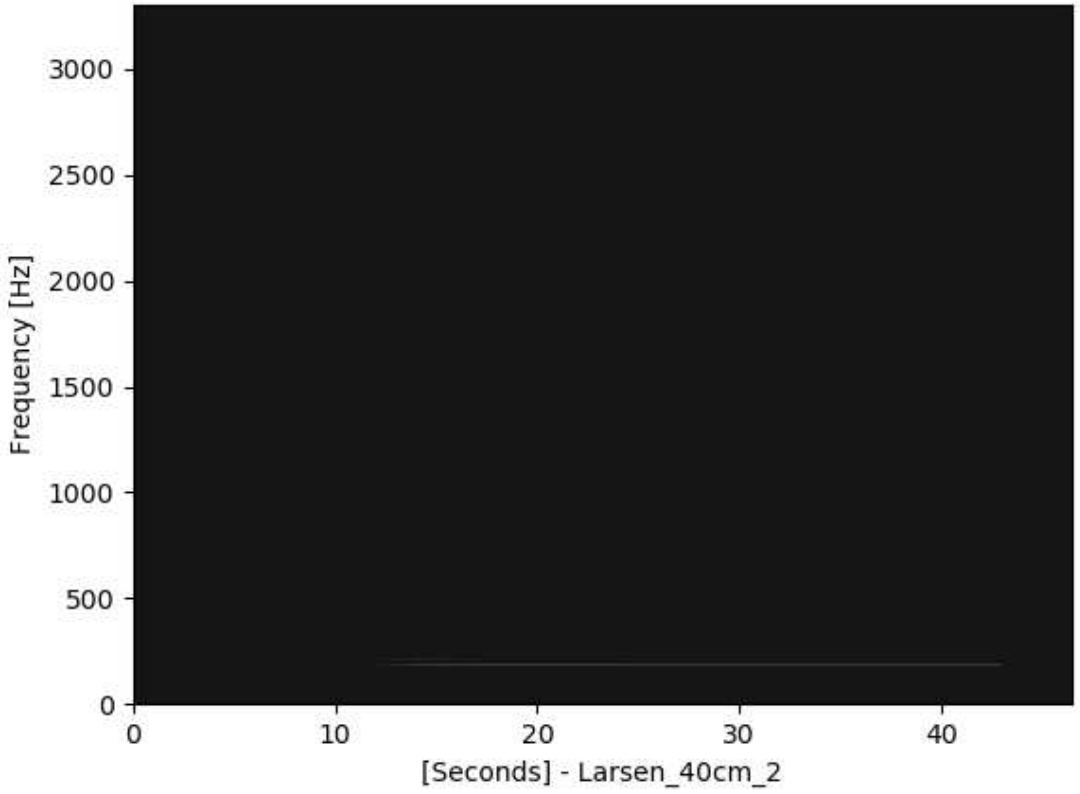

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_4

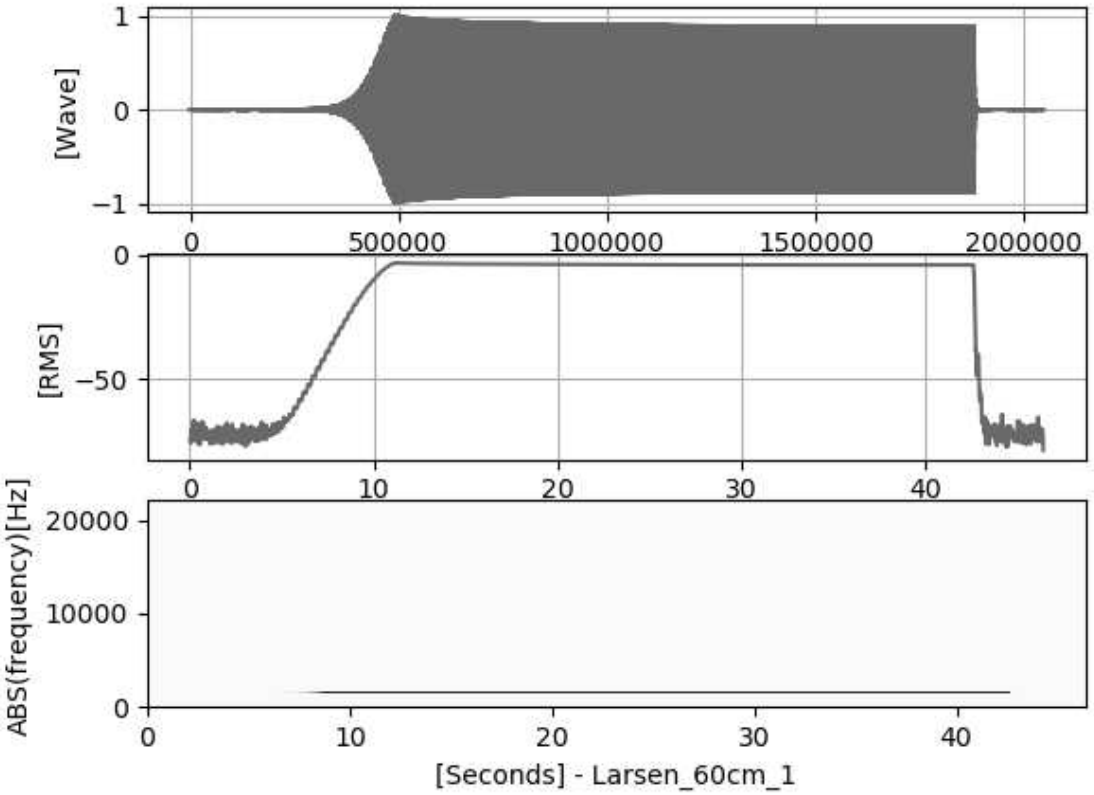

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_20cm_4

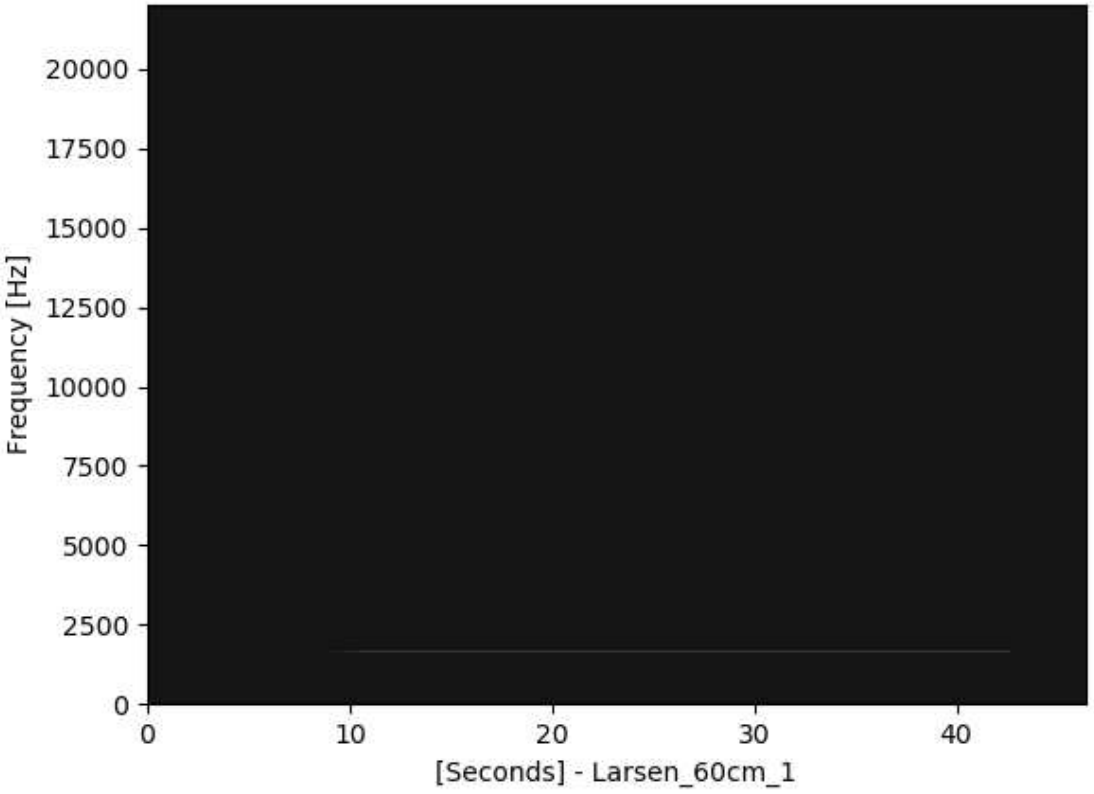

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_40cm_1

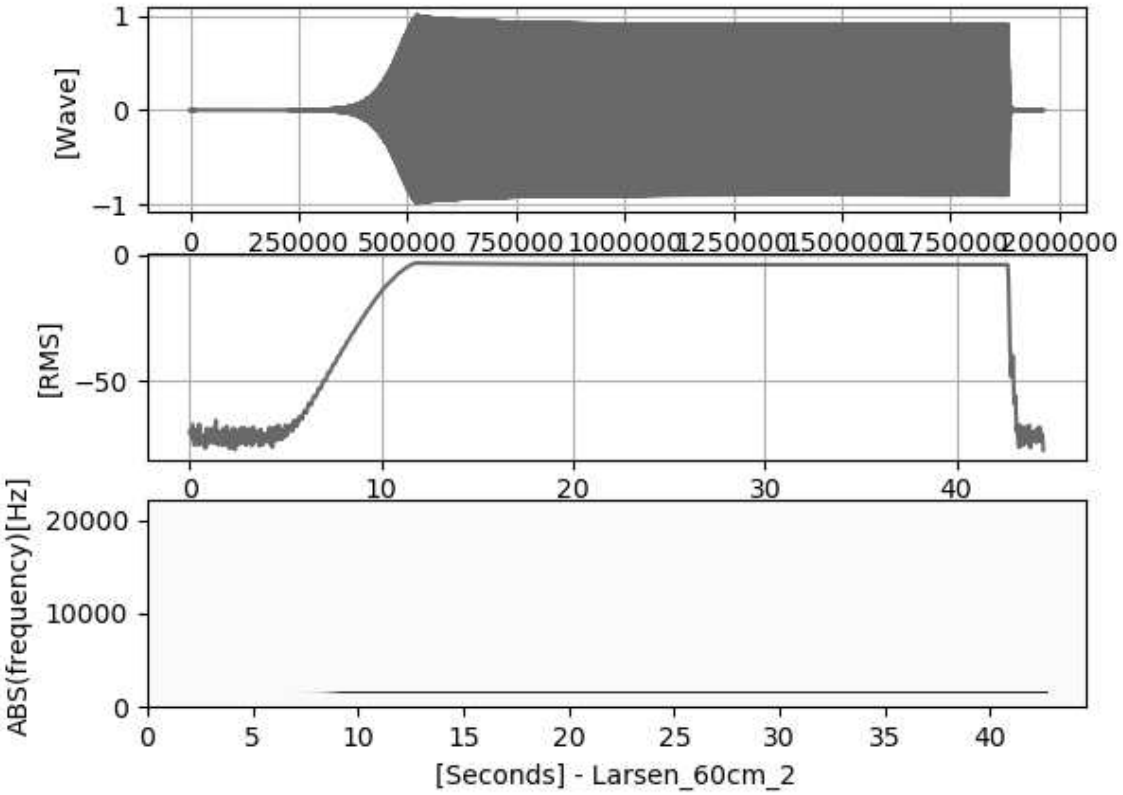

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_40cm_1

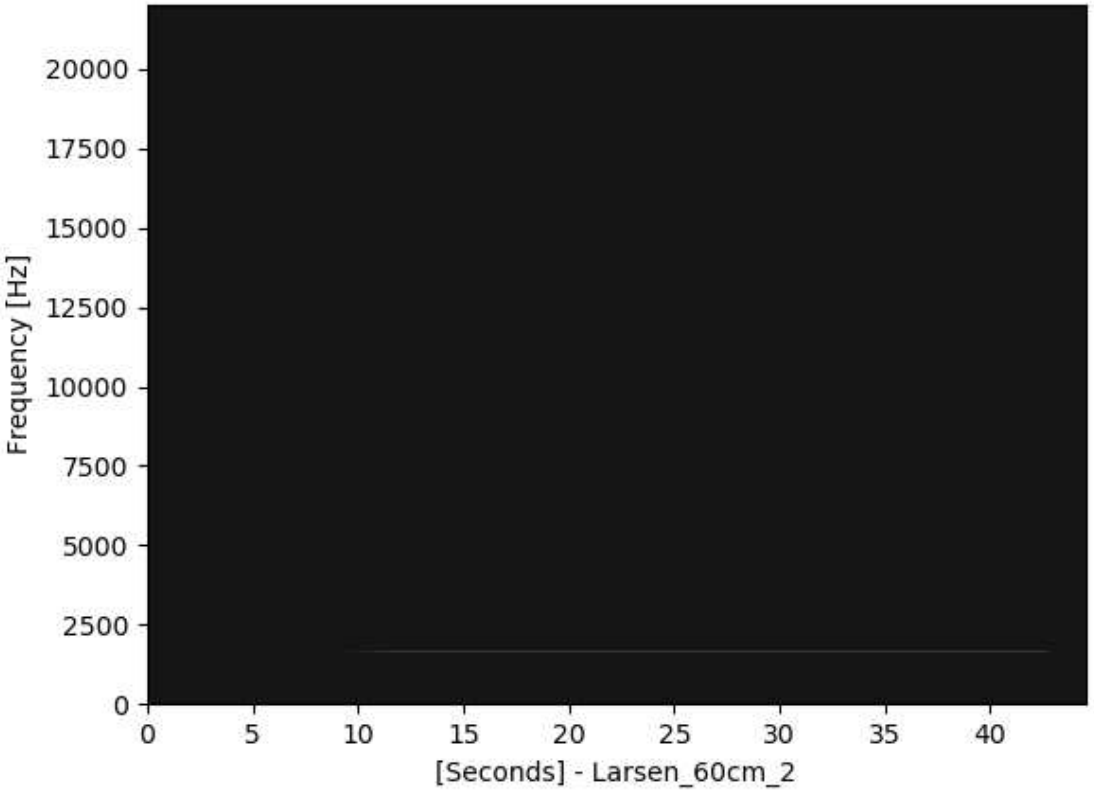

[Wave]
1
0
−1
0
500000
1000000
1500000
2000000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
10
20
30
40
[Seconds] - Larsen_40cm_2

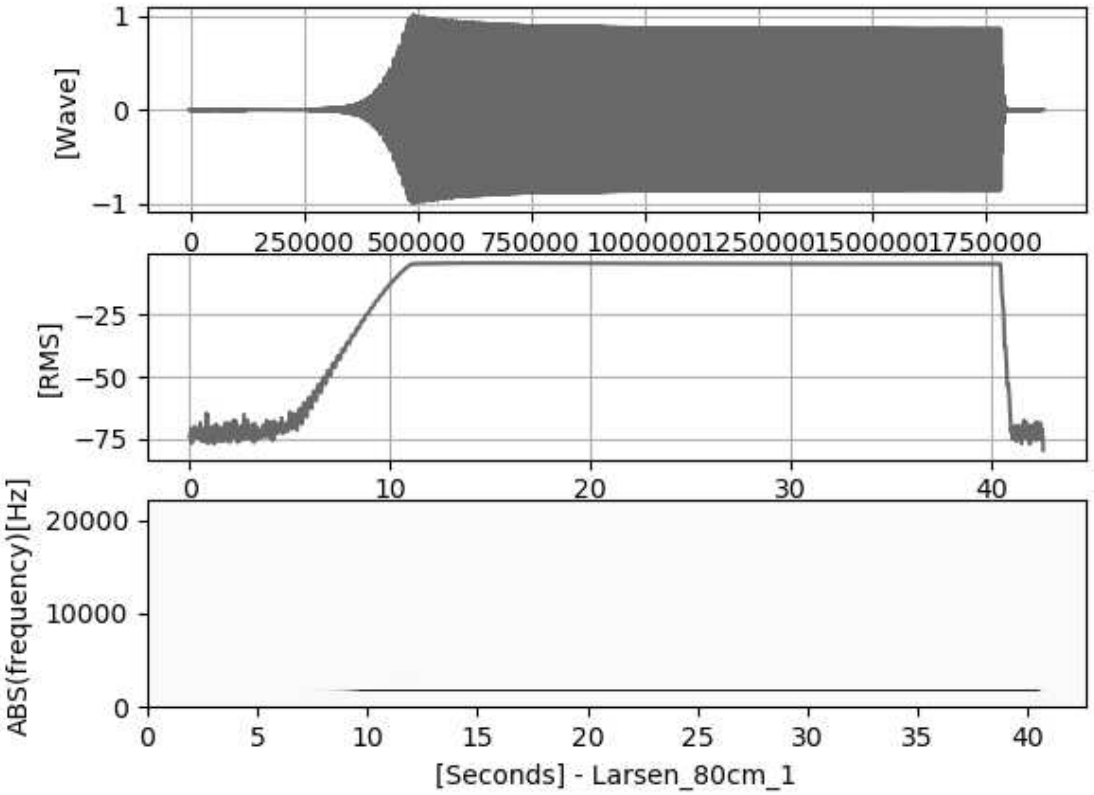

Frequency [Hz]
3000
2500
2000
1500
1000
500
0
0
10
20
30
40
[Seconds] - Larsen_40cm_2

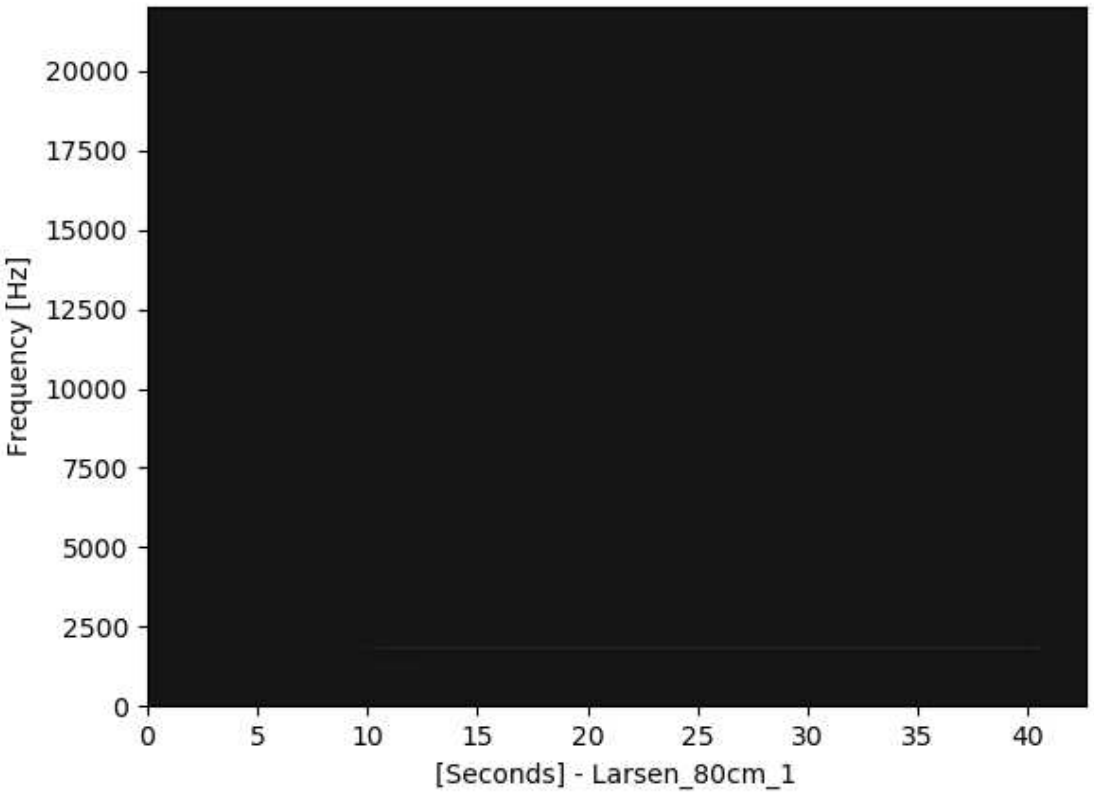

[Wave]
1
0
−1
0
500000
1000000
1500000
2000000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
10
20
30
40
[Seconds] - Larsen_60cm_1

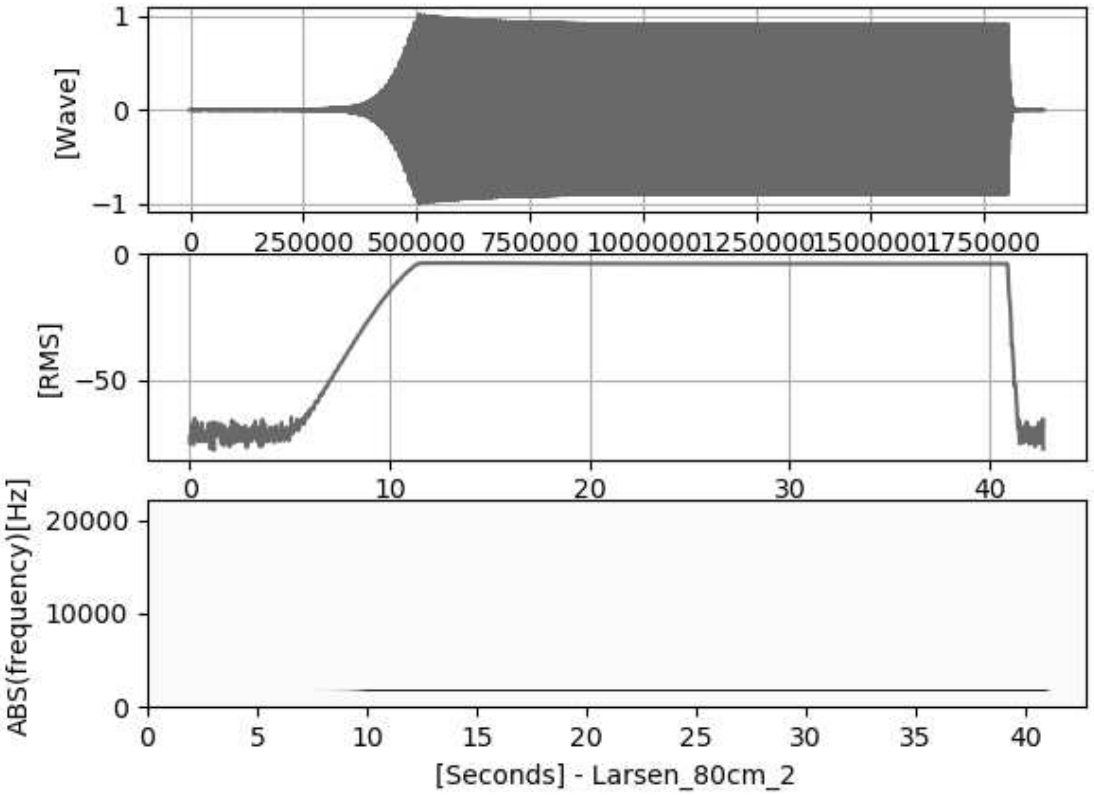

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
10
20
30
40
[Seconds] - Larsen_60cm_1

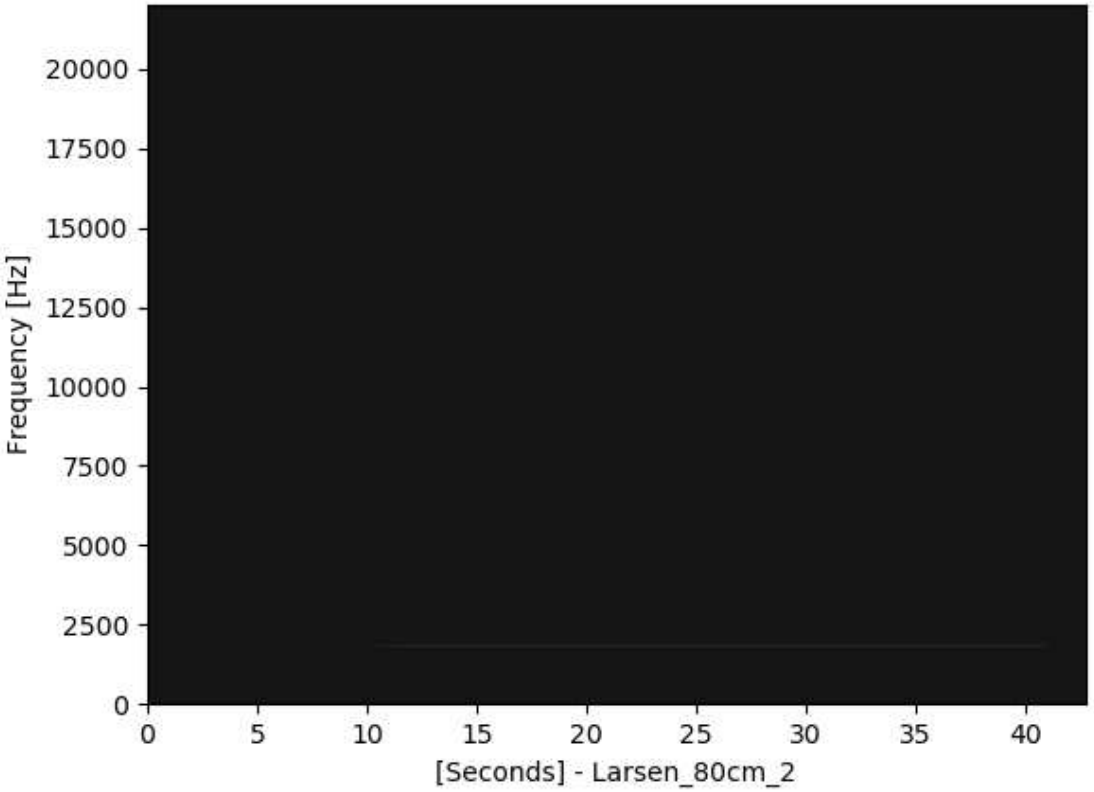

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_60cm_2

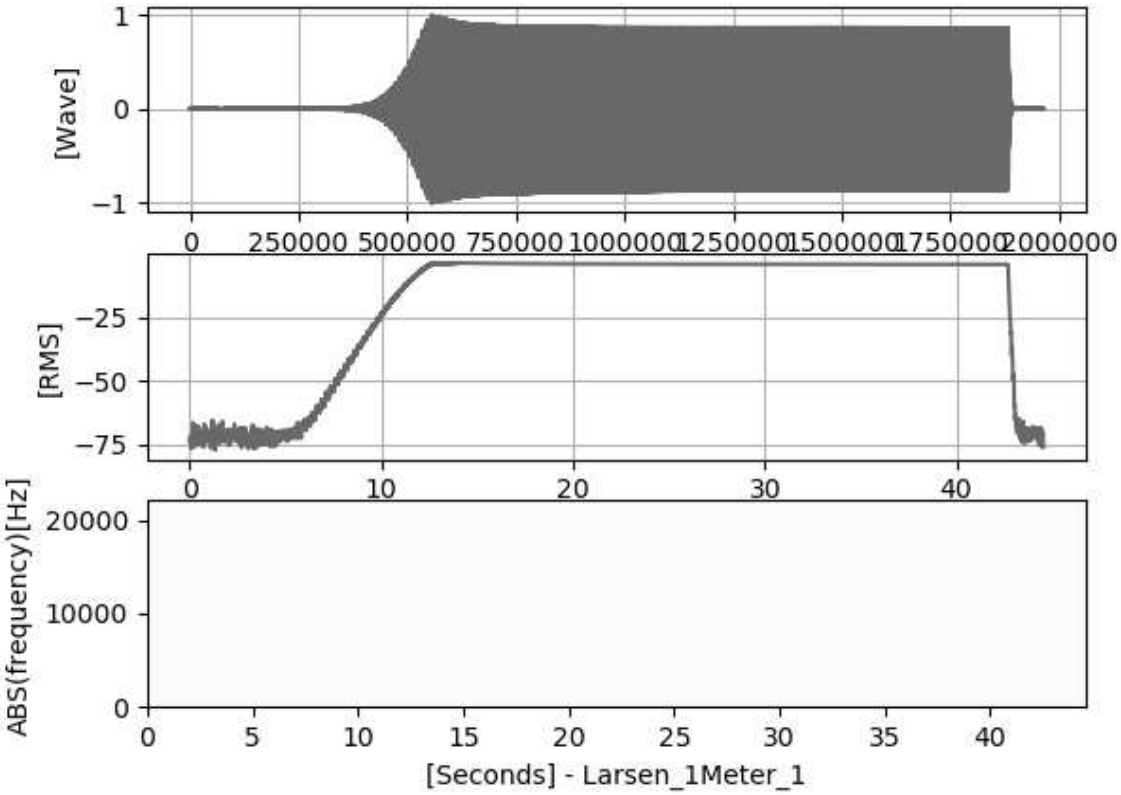

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_60cm_2

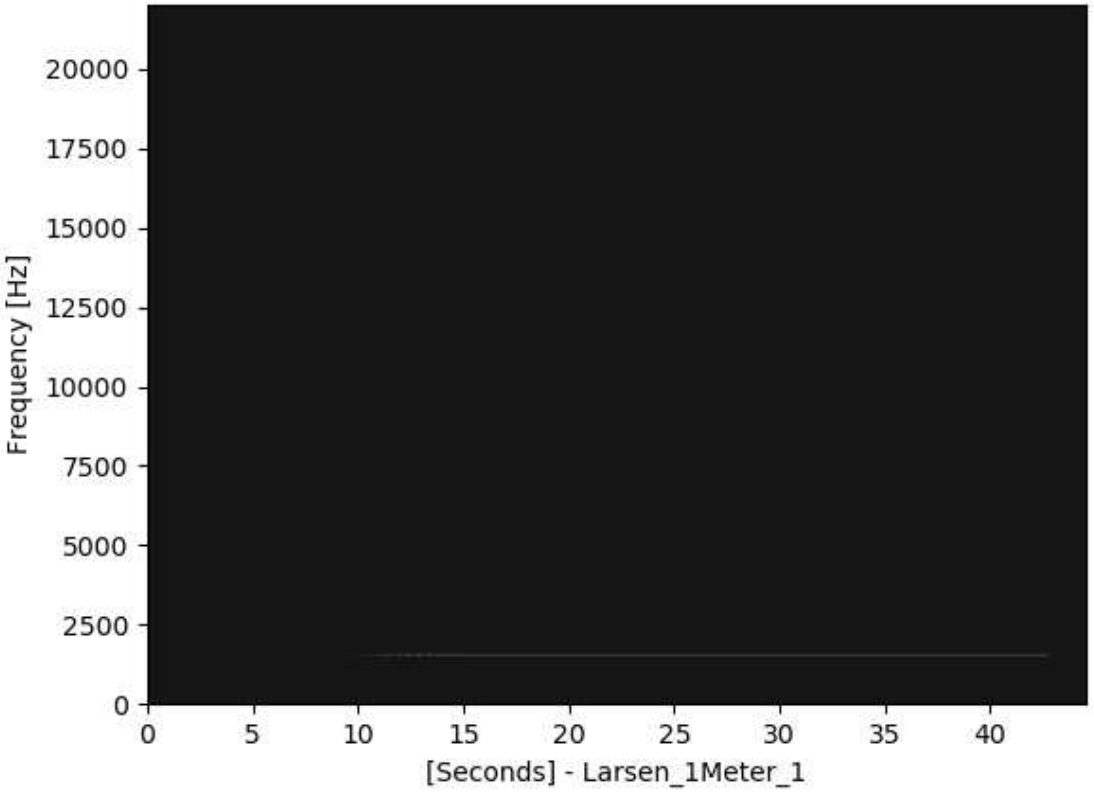

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_80cm_1

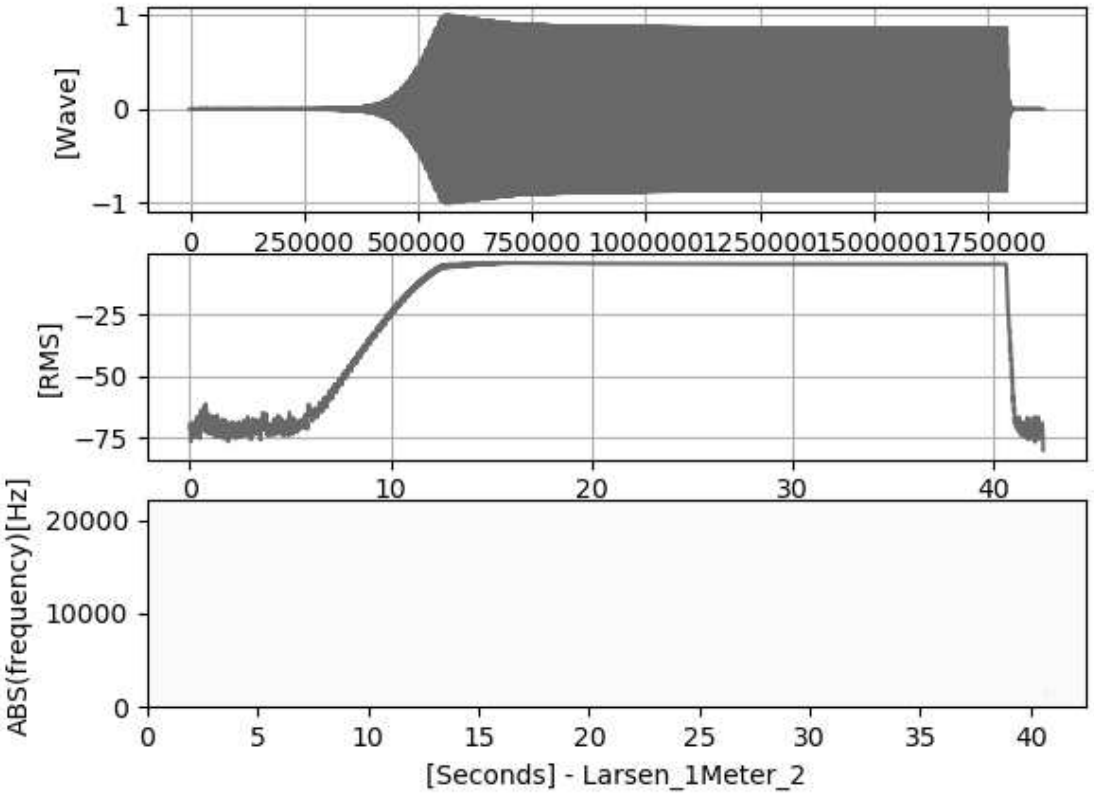

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_80cm_1

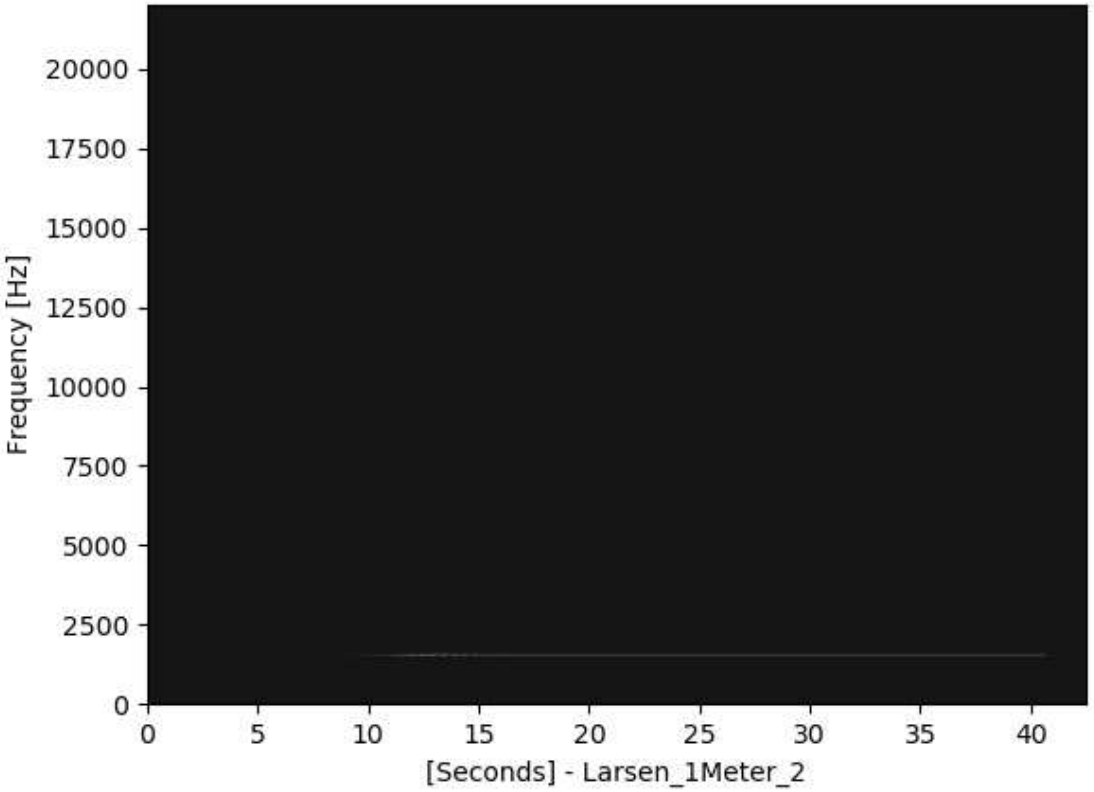

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_80cm_2

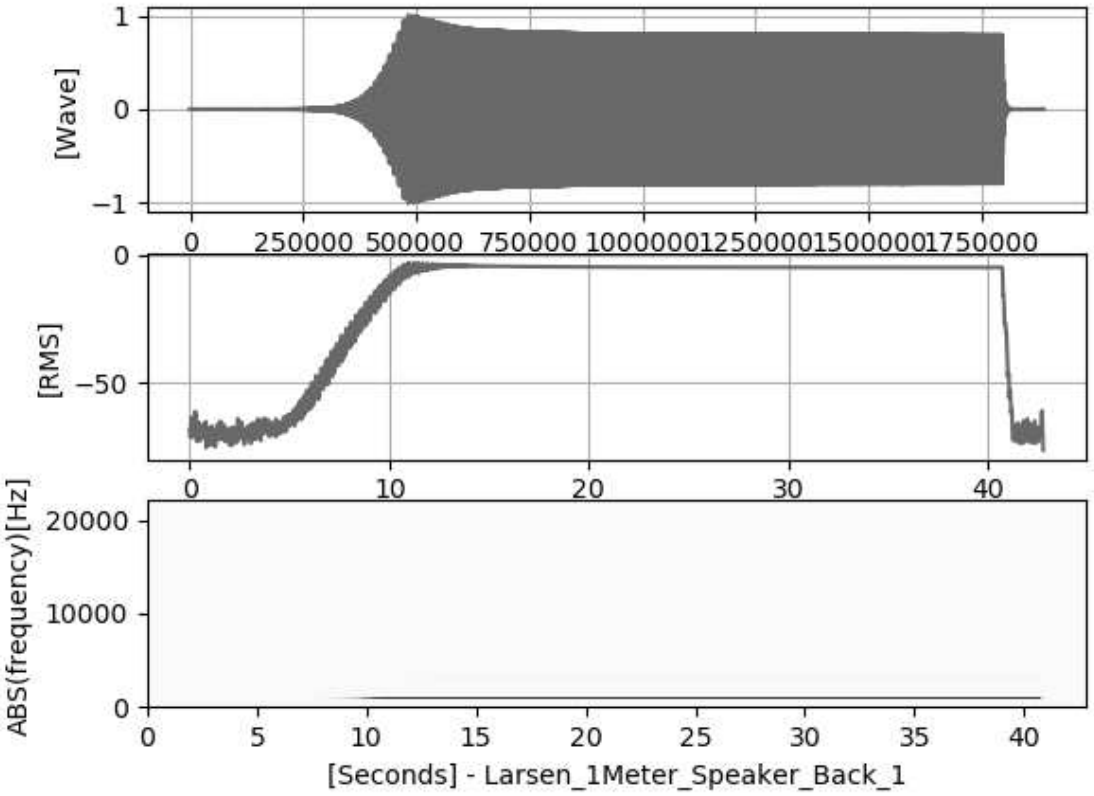

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_80cm_2

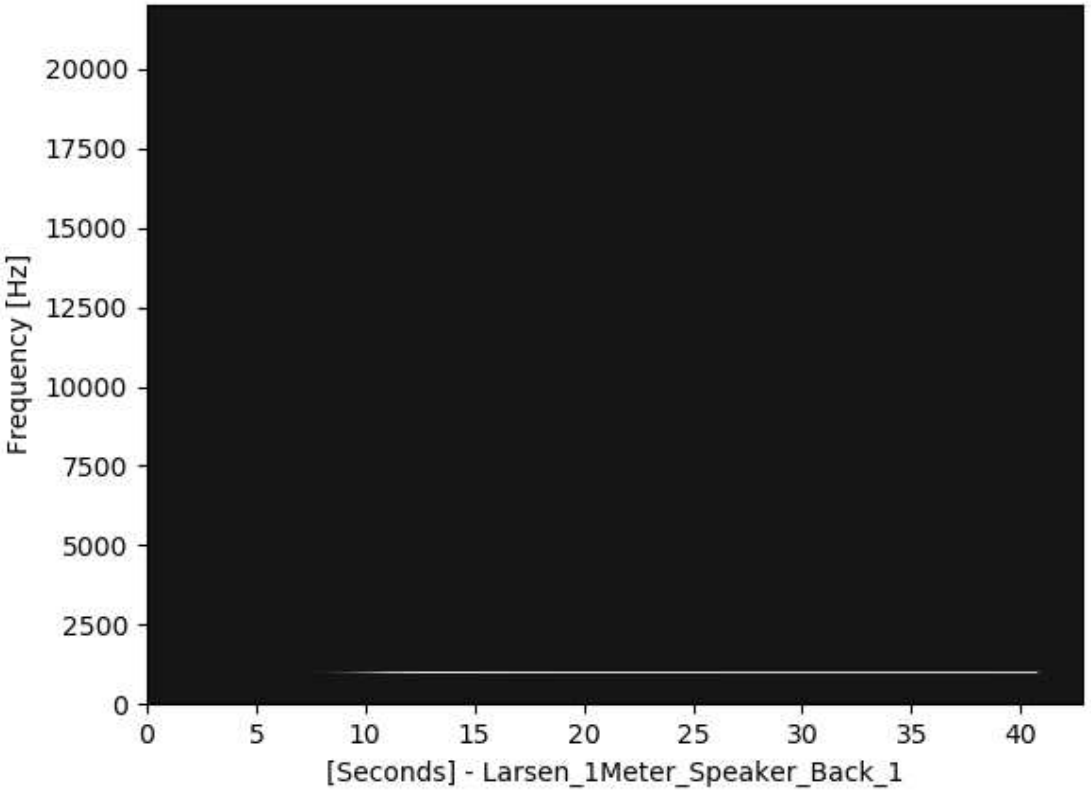

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
2000000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_1

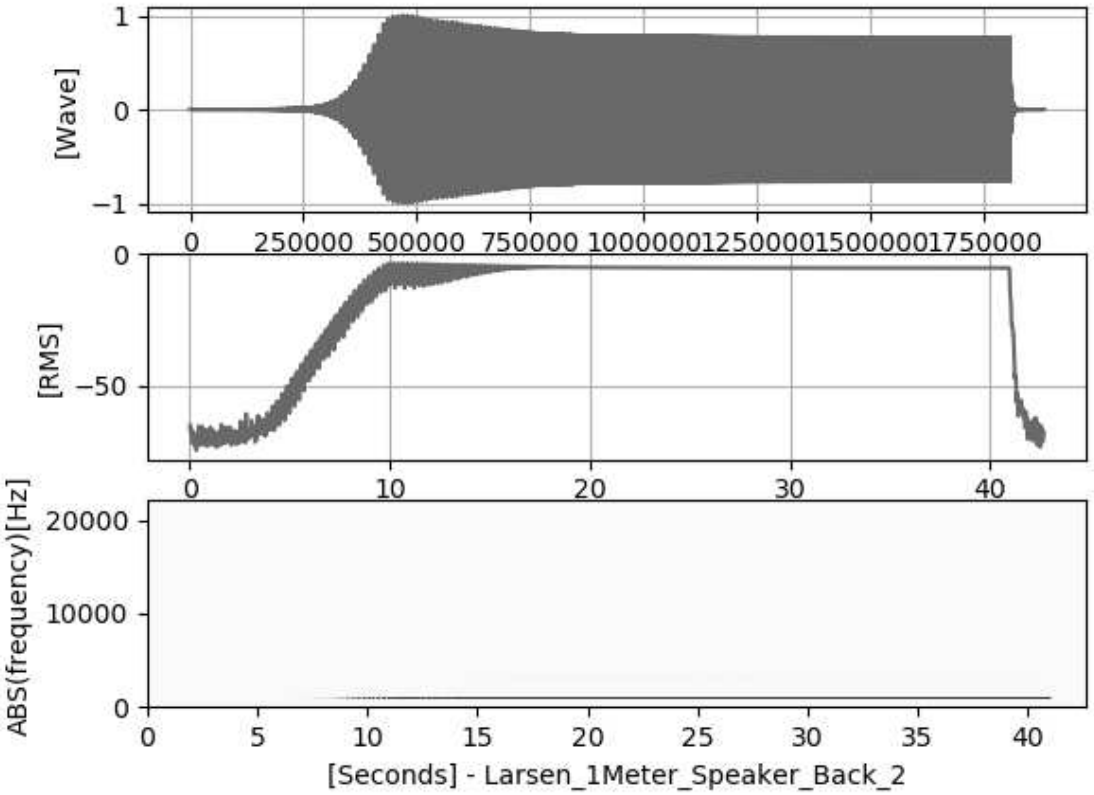

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_1

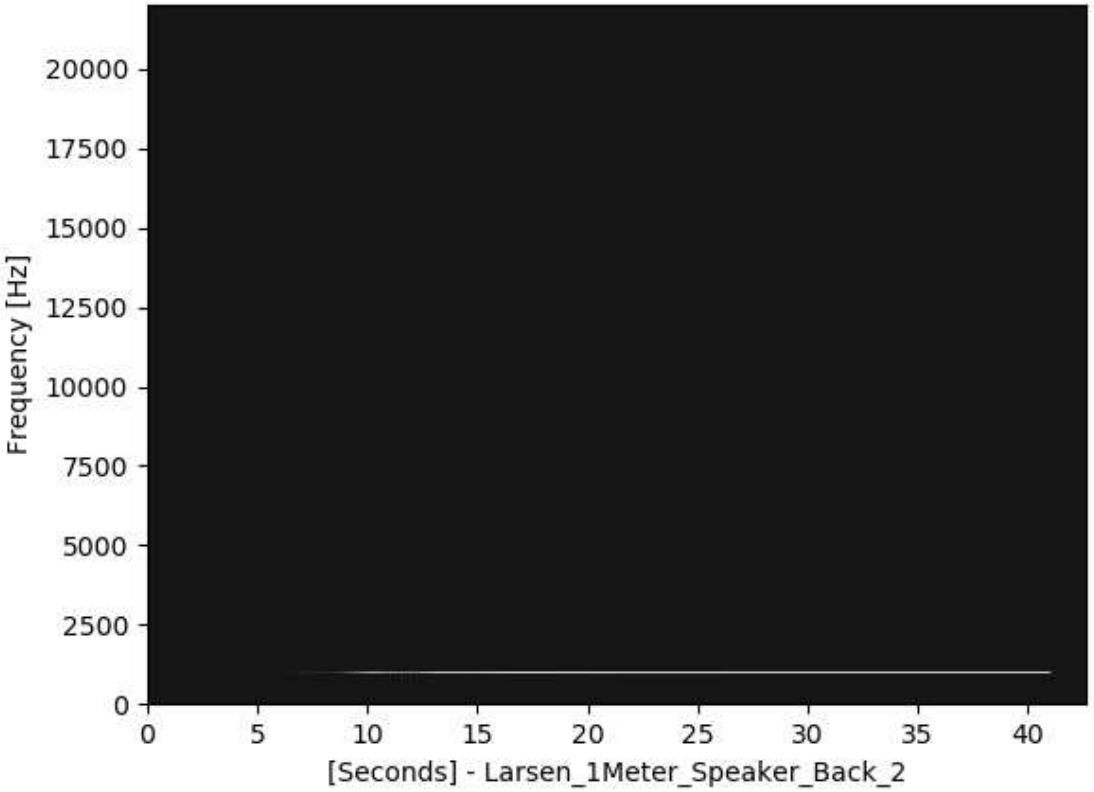

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
−25
−50
−75
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_2

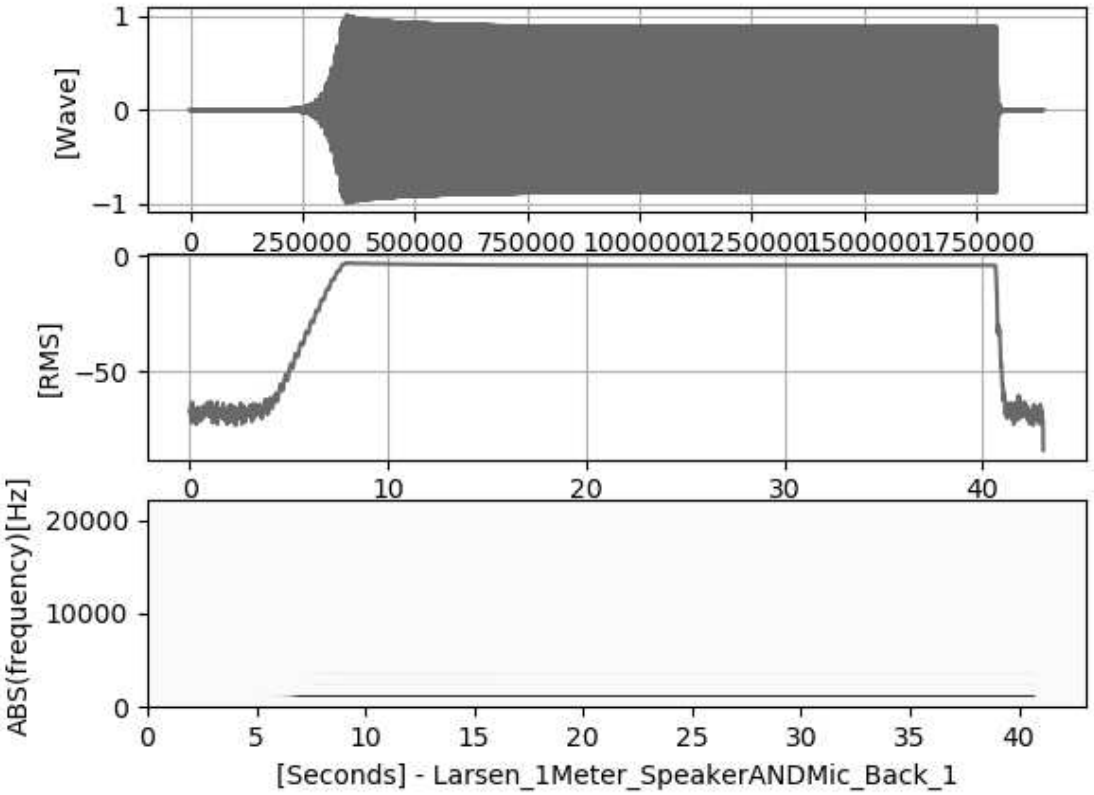

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_2

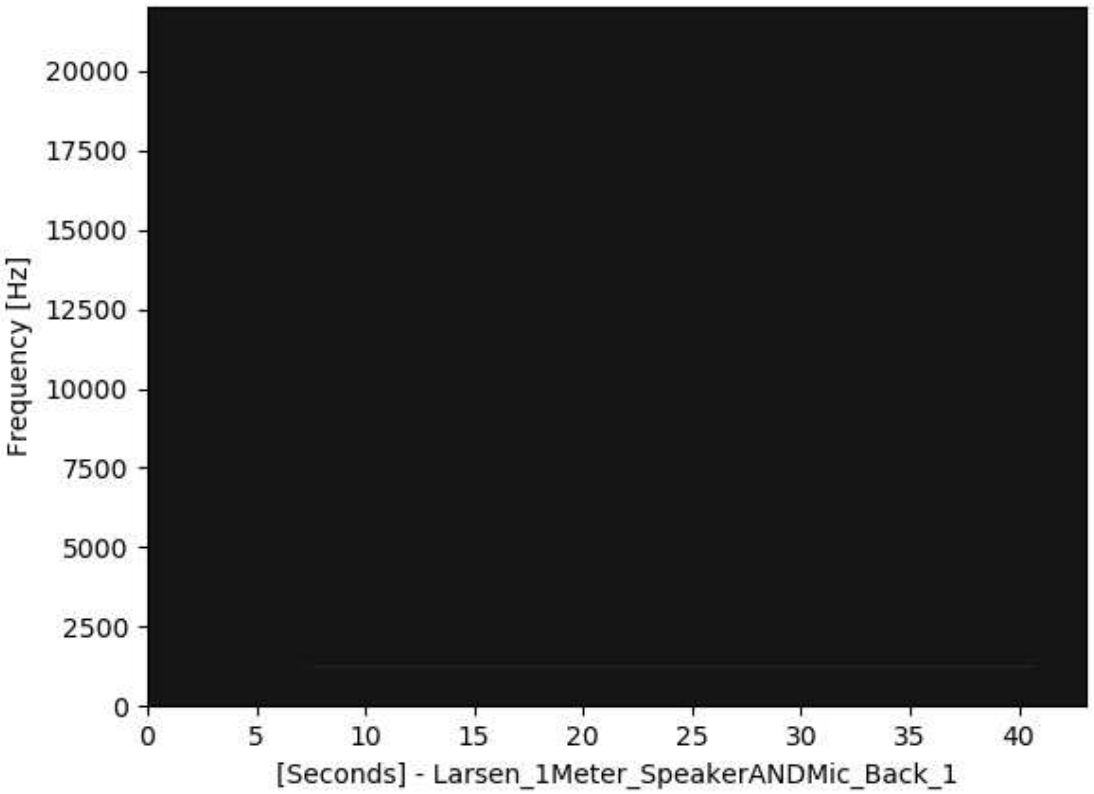

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_Speaker_Back_1

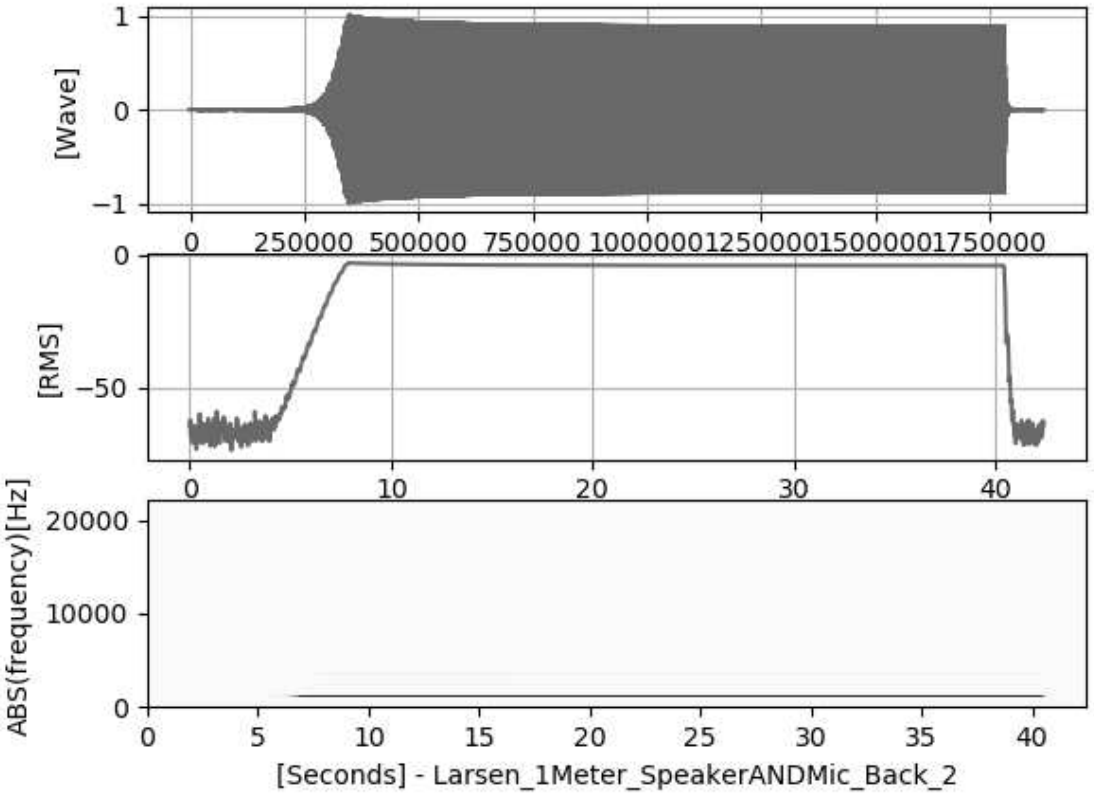

Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_Speaker_Back_1

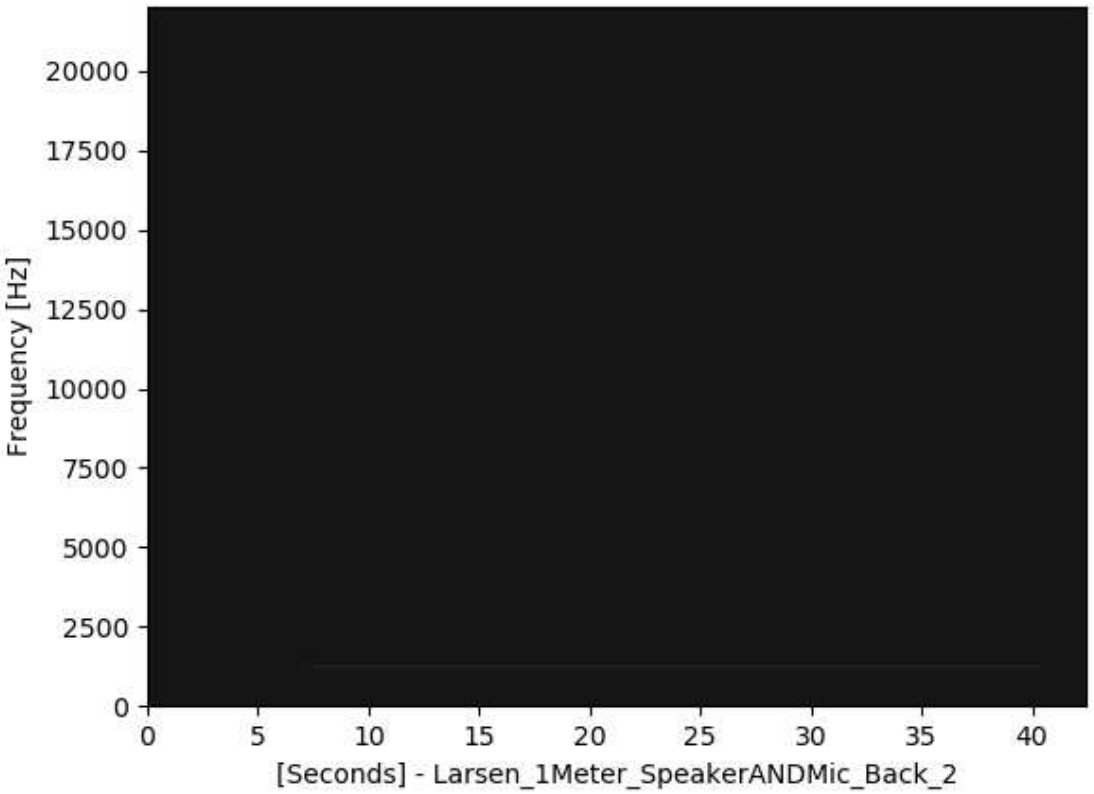

[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_Speaker_Back_2


Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_Speaker_Back_2


[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_1


Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_1


[Wave]
1
0
−1
0
250000
500000
750000
1000000
1250000
1500000
1750000
[RMS]
0
−50
0
10
20
30
40
ABS(frequency)[Hz]
20000
10000
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_2


Frequency [Hz]
20000
17500
15000
12500
10000
7500
5000
2500
0
0
5
10
15
20
25
30
35
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_2

Considerazioni sulle analisi: Forma d’onda, RMS, Spettrogramma

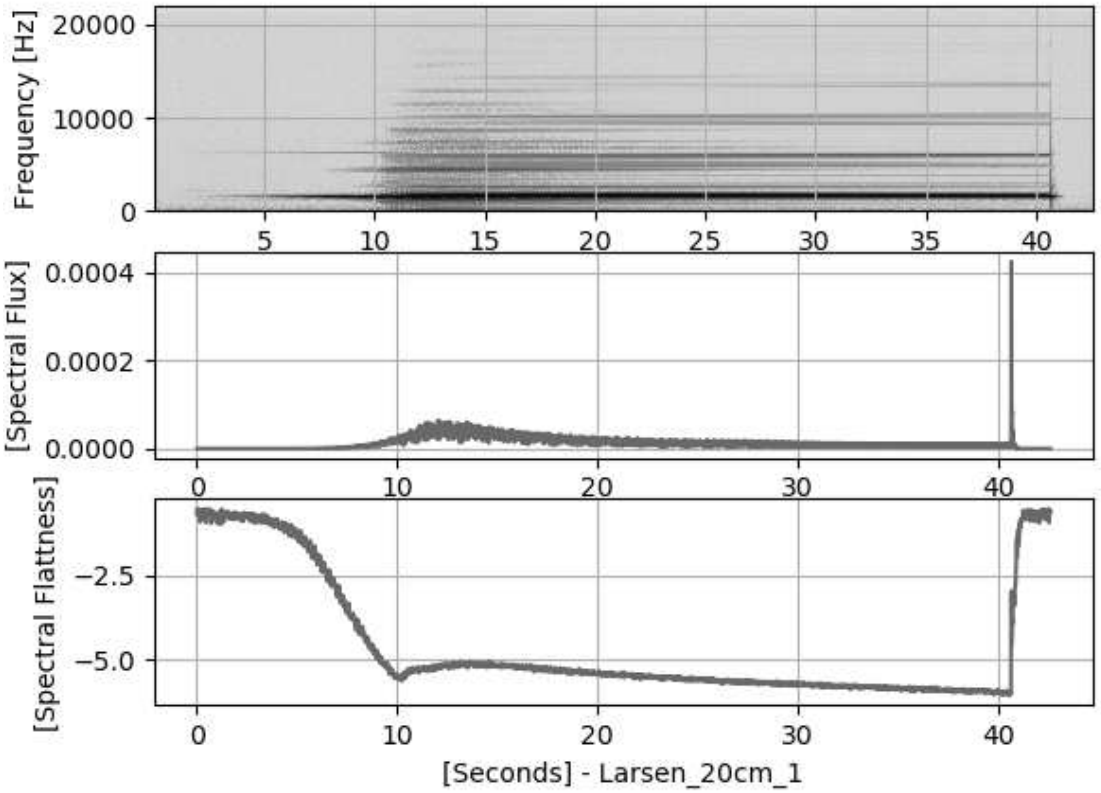
Le considerazioni che possiamo fare a proposito delle analisi appena svolte sono le seguenti. Dalla forma d’onda e dall’RMS, possiamo notare che il Larsen ha un comportamento in ampiezza che si ripete ugualmente al variare delle condizioni iniziali, una soglia temporale di alcuni secondi in cui emerge il *feedback positivo* che al raggiungimento di un picco, grazie al nostro controbilanciamento con un *feedback negativo* decade per poi assestarsi lentamente in uno stato stazionario. Questo prova il funzionamento del nostro algoritmo di controbilanciamento in Faust. Molto interessanti sono anche le analisi che emergono dallo spettrogramma, dalle prime quattro (i Larsen generati a una distanza di 20cm) notiamo come per le stesse condizioni iniziali ci siano comunque alcune differenze che emergono nell’andamento spettrale; un'analisi del *flusso spettrale* potrà mettere in evidenza questo comportamento di instabilità. Una seconda considerazione che emerge dalle analisi è che in generale l’effetto Larsen produce uno spettro molto simile ad una sinusoide; il comportamento viene rafforzato energeticamente nelle analisi in cui è stato rivoltato l’altoparlante verso la parete e questo mette in evidenza che un contributo molto importante nella generazione del fenomeno è dato dalla riflessione delle pareti; l’effetto Larsen trova dunque un importante sostegno nei fenomeni di riflessione e possiamo dunque ipotizzare che in un ambiente povero di fenomeni di riflessione ambientale il Larsen emergerà con più difficoltà. Nelle analisi spettrali condotte in questa prima istanza viene messo in evidenza come nonostante l’emissione di suono da parte dell’altoparlante avvenga direttamente sulla capsula microfonica che capta più suono diretto rispetto a quello riflesso, l’effetto dalle analisi sembra essere energeticamente più povero rispetto a quelle in cui l’altoparlante è rivolto verso le pareti. La frequenza dell’effetto Larsen varia all’incirca in ogni registrazione, ma possiamo affermare che la variazione è fortemente sensibile alla distanza tra altoparlante e microfono. In ogni caso per un approfondimento del contenuto spettrale passiamo ora a delle ulteriori analisi sullo *spectral flux* e *sulla spectral flatness*.

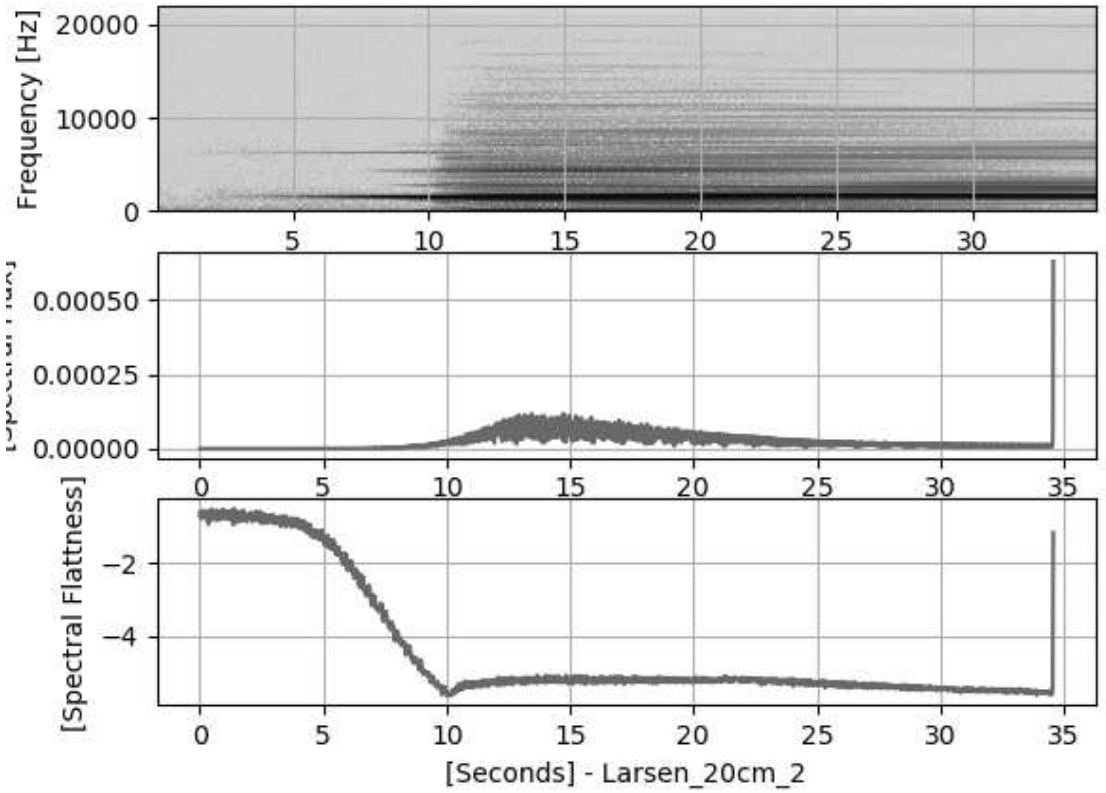
Analisi: Spectral Flux, Spectral Flatness

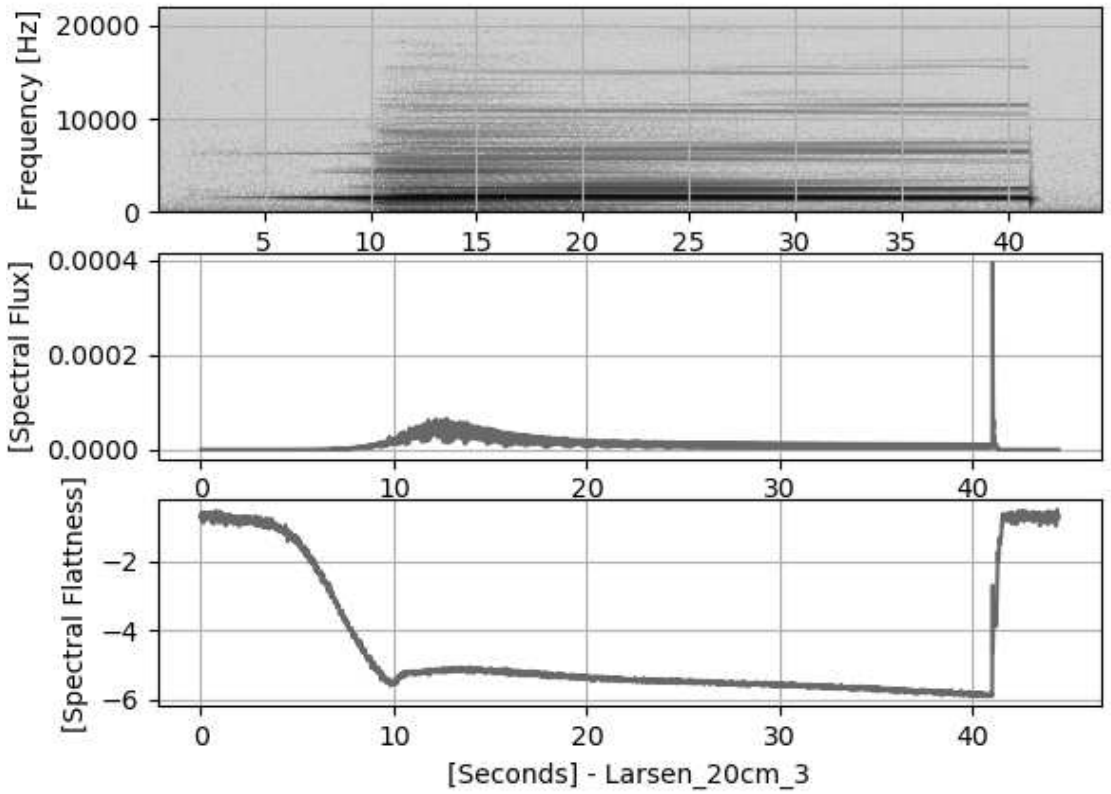
Il flusso spettrale *(spectral flux)* è una misura della velocità con cui cambia lo spettro di un segnale, calcolato confrontando una *frame* STFT e la precedente, mentre invece la piattezza spettrale *(spectral flatness)* nota anche come entropia di Wiener, è una misura in decibel del tasso di somiglianza di un suono ad un suono con spettro "rigato" o ad un rumore perfettamente stocastico. Il codice Python per il calcolo di questi descrittori è il seguente. Di seguito i risultati delle analisi.

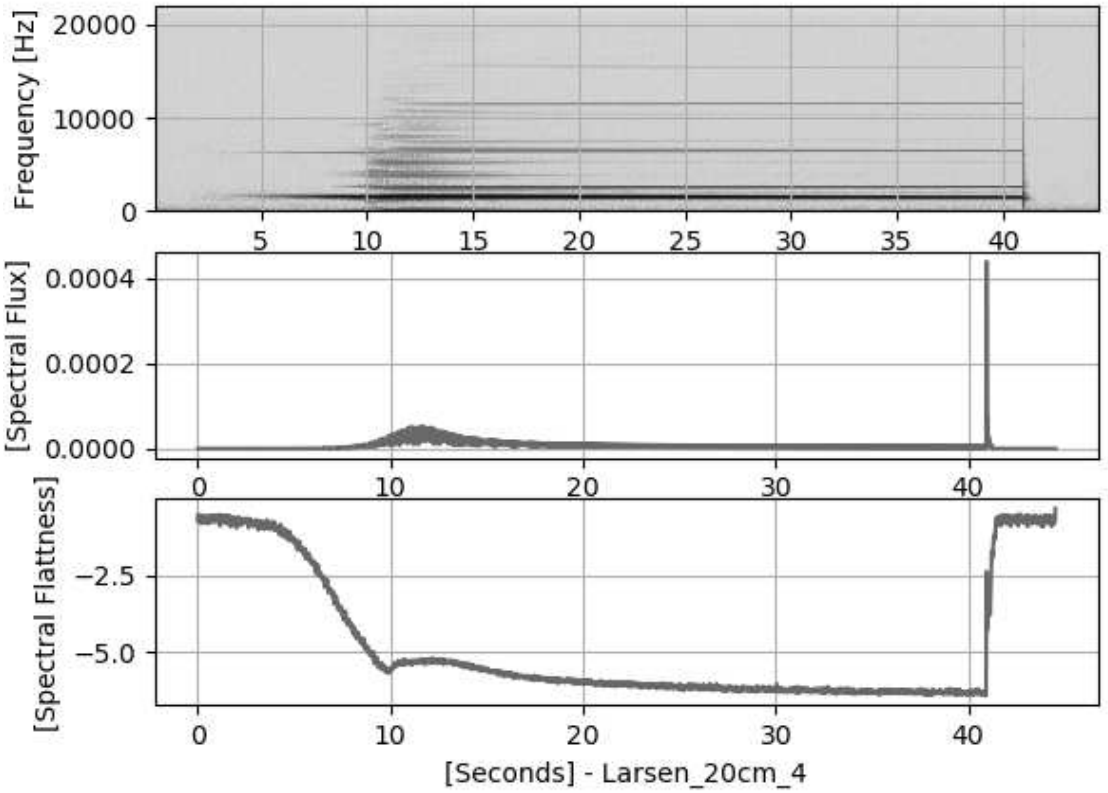
```
# CALCOLO SPECTRAL FLUX / FLATNESS

# librerie
import pyACA
import numpy as np
from scipy import signal
import matplotlib.pyplot as plt
from scipy.io.wavfile import read

# leggiamo un file audio mono importandolo da terminale
audiofile = input("Enter your file name (without extension):")

# leggiamo un file audio(file audio mono)
[fs, x] = pyACA.ToolReadAudio(audiofile+".wav")

# normalizziamo il file audio(a 0 dB)
x = x/np.max(abs(x))

# genero una base tempi coerente con il segnale analizzato, usando l'istruzione arange
time = np.arange(0,np.size(x))/fs

# spectral flux
[dsx, t] = pyACA.computeFeature("SpectralFlux",x,fs,iBlockLength=1024, iHopLength=512)

# spectral flatness
[dsf, t] = pyACA.computeFeature("SpectralFlatness",x,fs,iBlockLength=1024, iHopLength=512)

# plot
# nomina output
nomeplot = audiofile+"-Spectral-flux-flatness.png"

# numero di grafici verticali, numero di finestre orizzontali, numero progressivo
plt.subplot(3,1,1)
plt.grid()
plt.specgram(x, NFFT=2048, Fs=fs, noverlap=1024, cmap='jet_r')
plt.xlabel("[Seconds]")
plt.ylabel("Frequency [Hz]")

plt.subplot(3,1,2)
plt.grid()
plt.plot(t,dsx)
plt.xlabel("[Seconds]")
plt.ylabel("[Spectral Flux]")

plt.subplot(3,1,3)
plt.grid()
plt.plot(t, np.log(dsf))
plt.xlabel("[Seconds] - "+audiofile)
plt.ylabel("[Spectral Flatness]")

# salva output
plt.savefig(nomeplot)
```

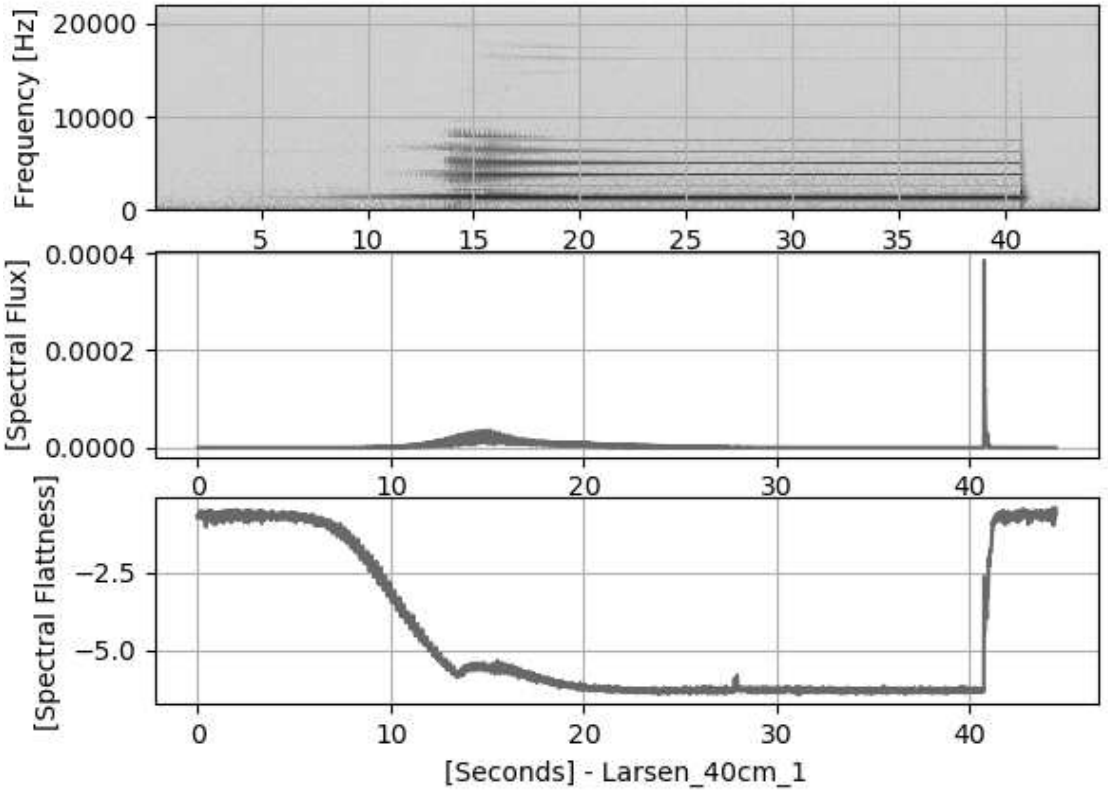
Codice Python per estrazione dello Spectral Flux e Spectral Flatness

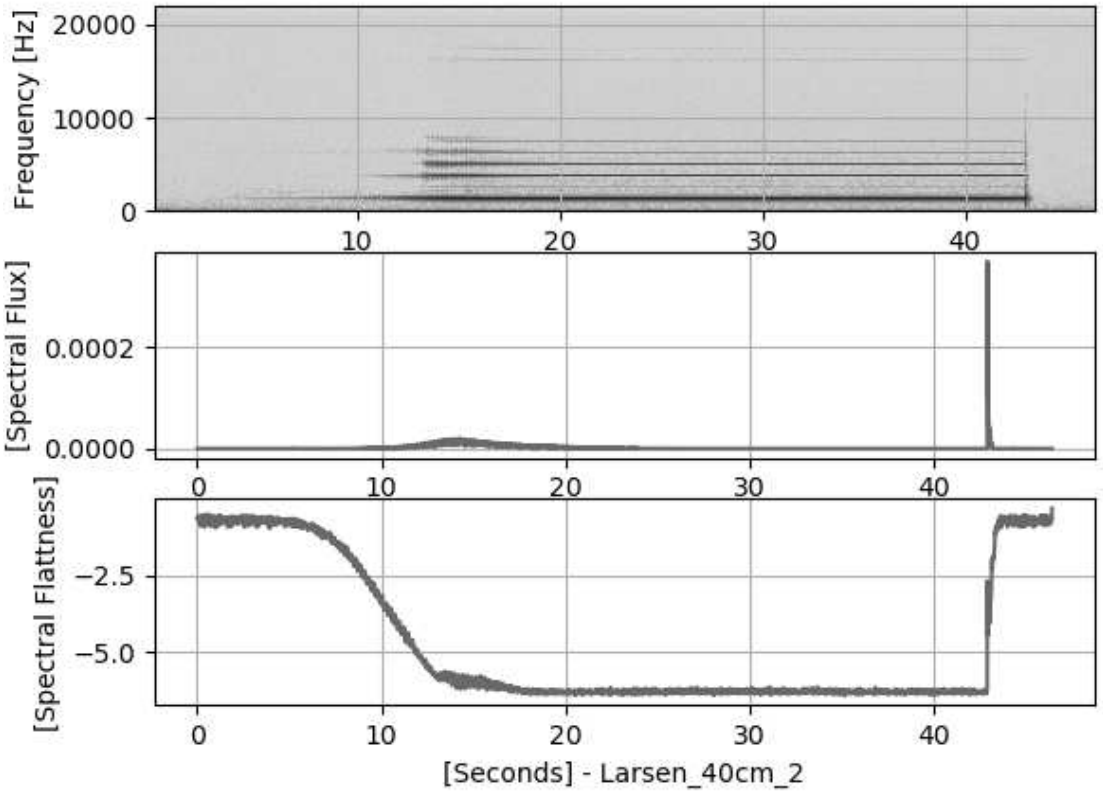

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_20cm_1

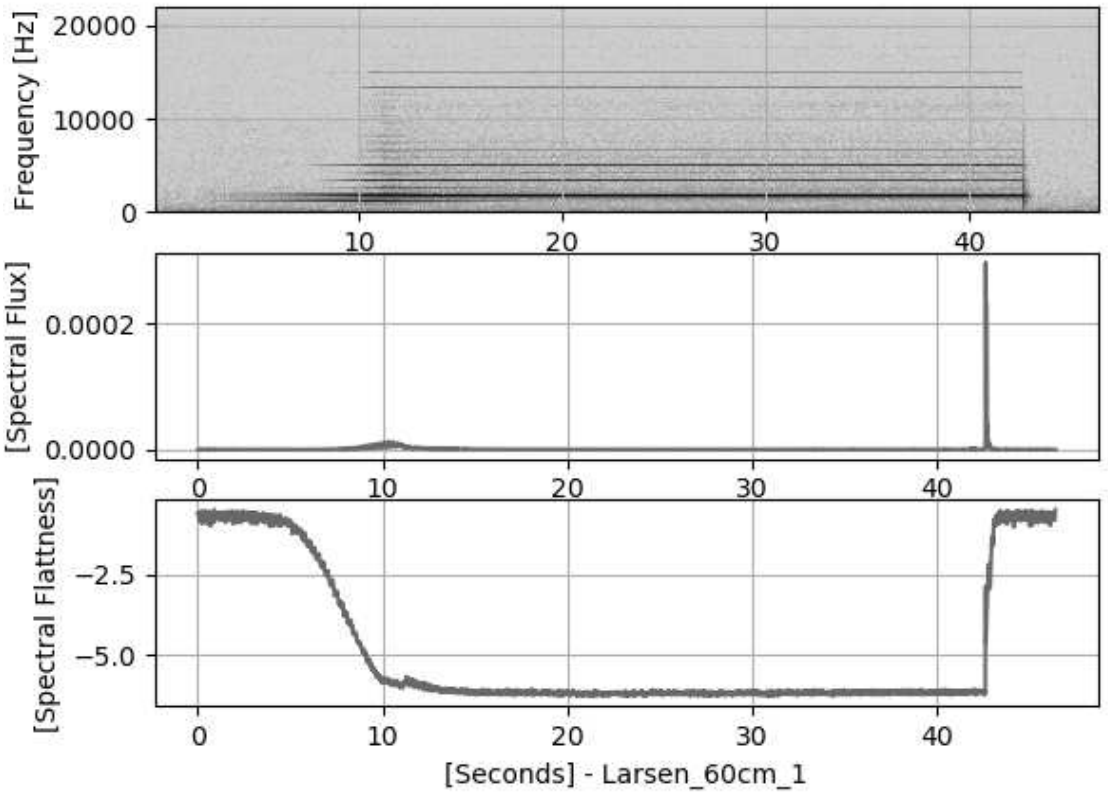

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
0.00050
0.00025
0.00000
0
5
10
15
20
25
30
35
[Spectral Flattness]
−2
−4
0
5
10
15
20
25
30
35
[Seconds] - Larsen_20cm_2

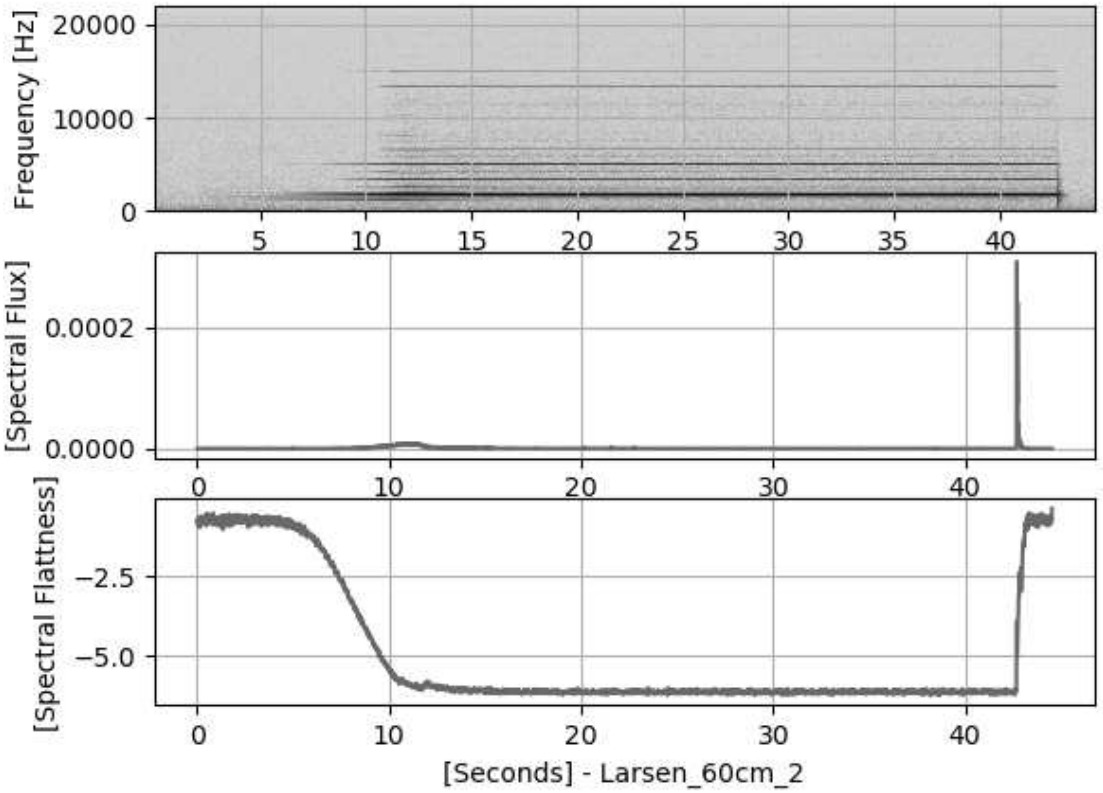

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2
−4
−6
0
10
20
30
40
[Seconds] - Larsen_20cm_3

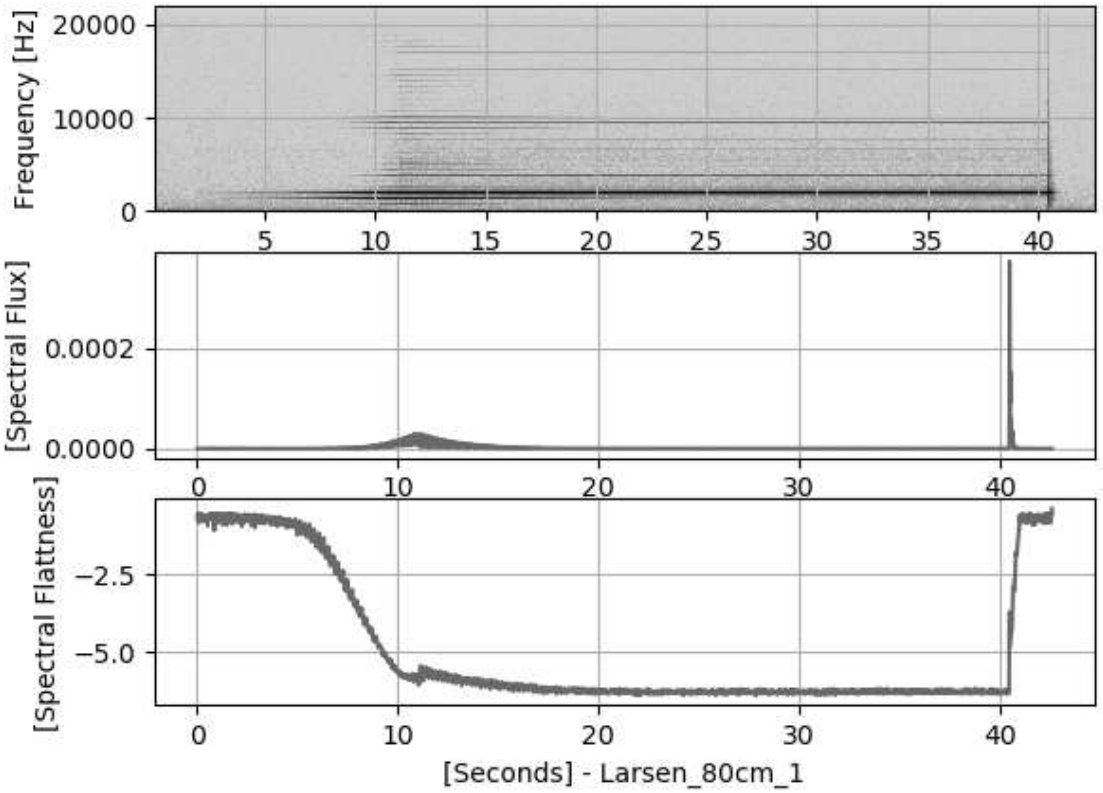

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flatness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_20cm_4

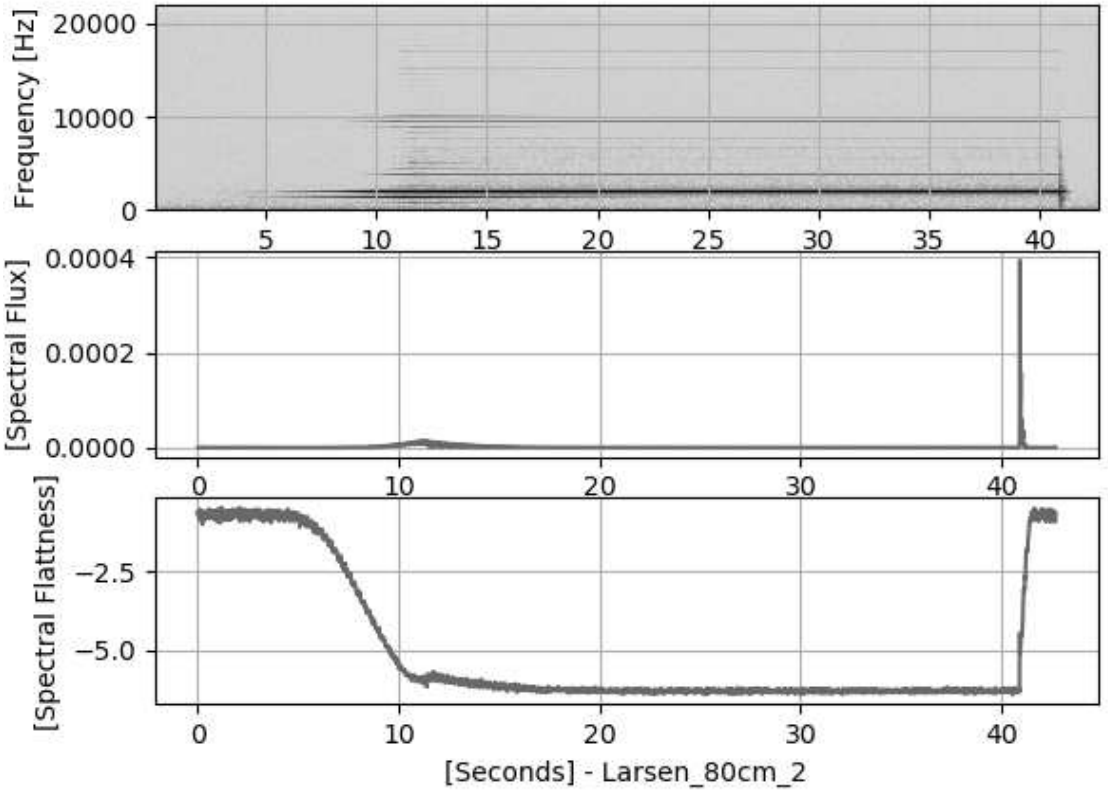

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flatness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_40cm_1

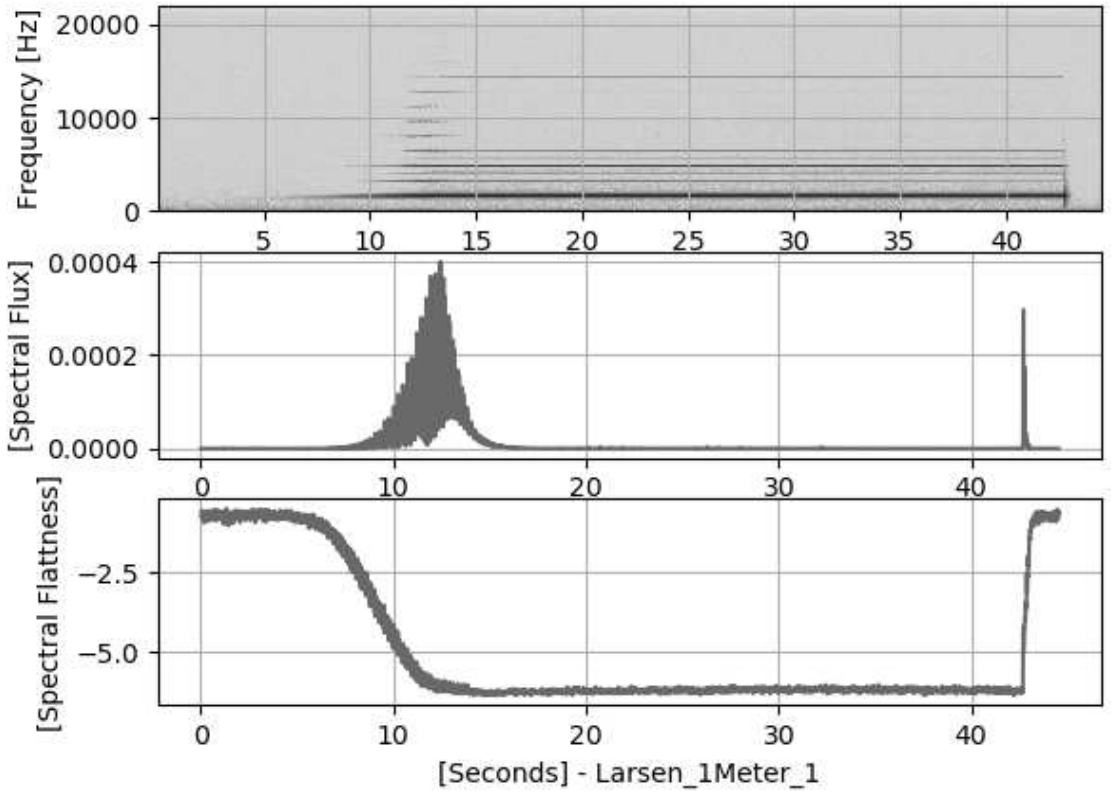

Frequency [Hz]
20000
10000
0
10
20
30
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_40cm_2

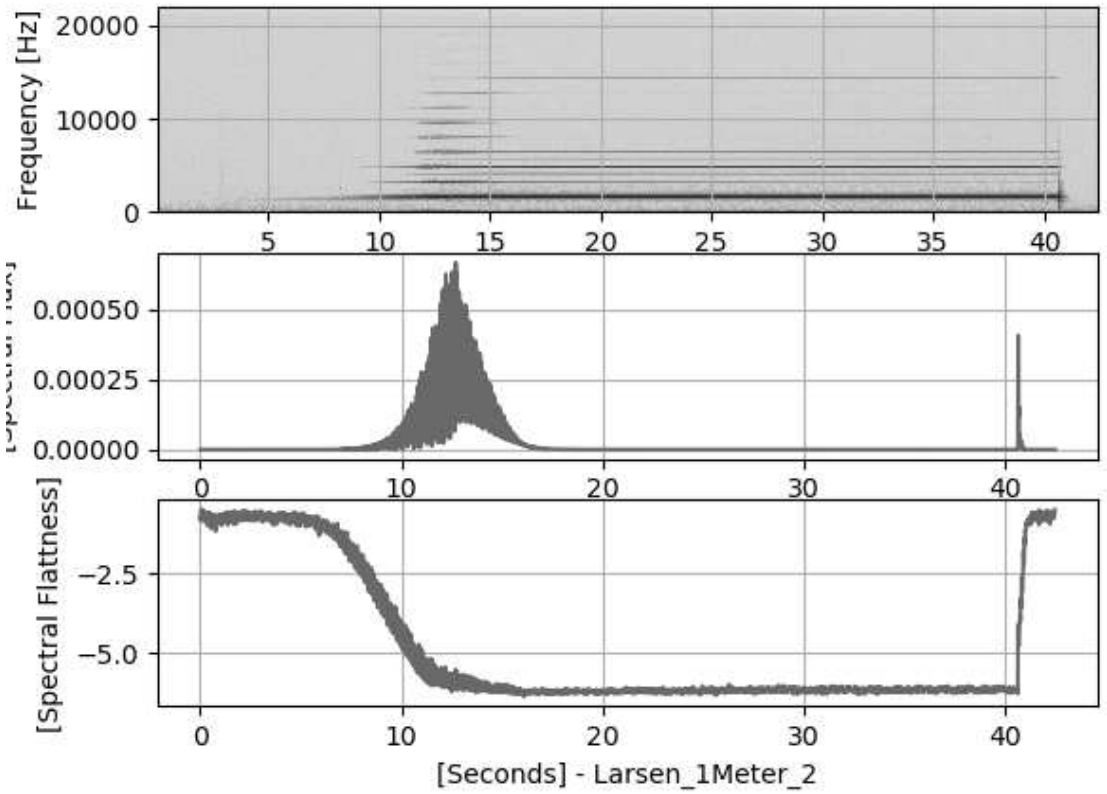

Frequency [Hz]
20000
10000
0
10
20
30
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_60cm_1

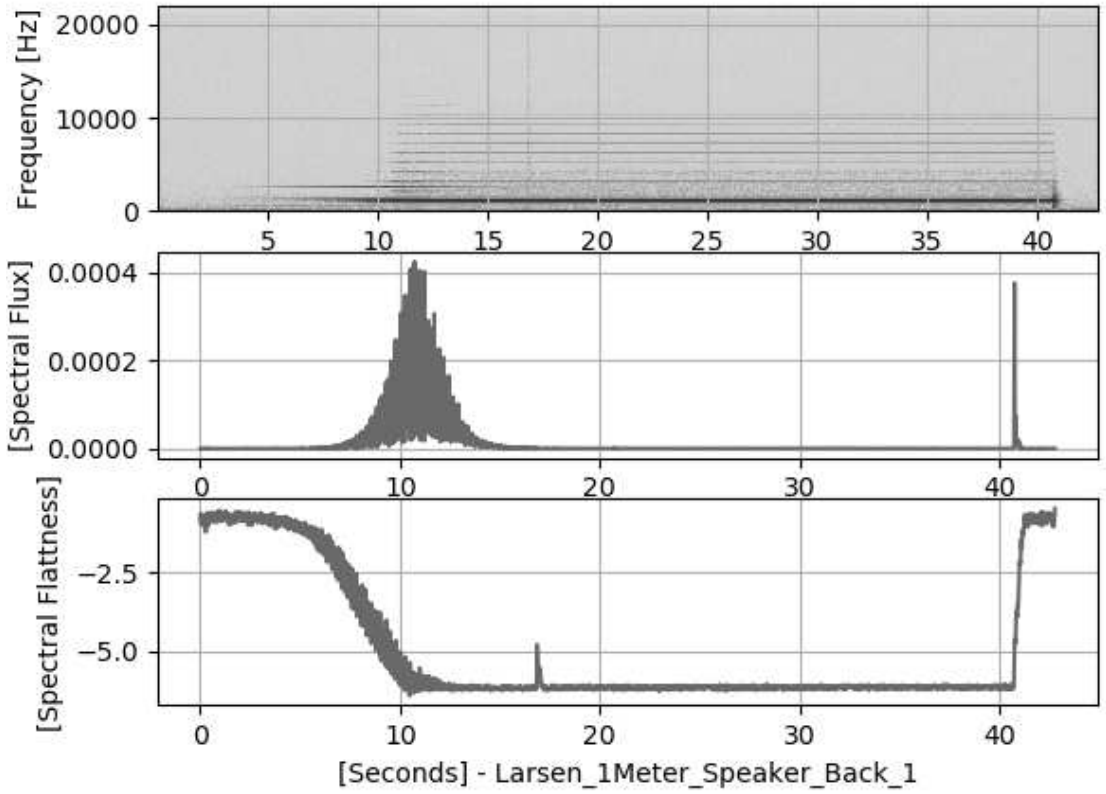

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_60cm_2

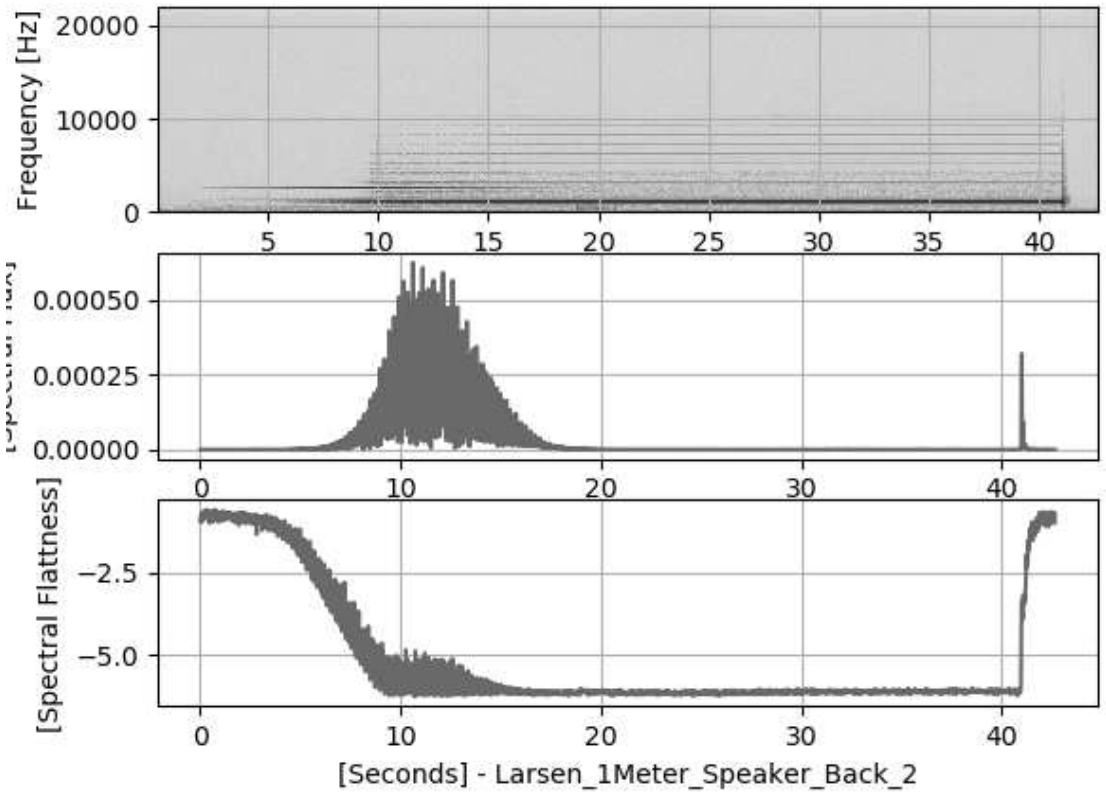

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_80cm_1

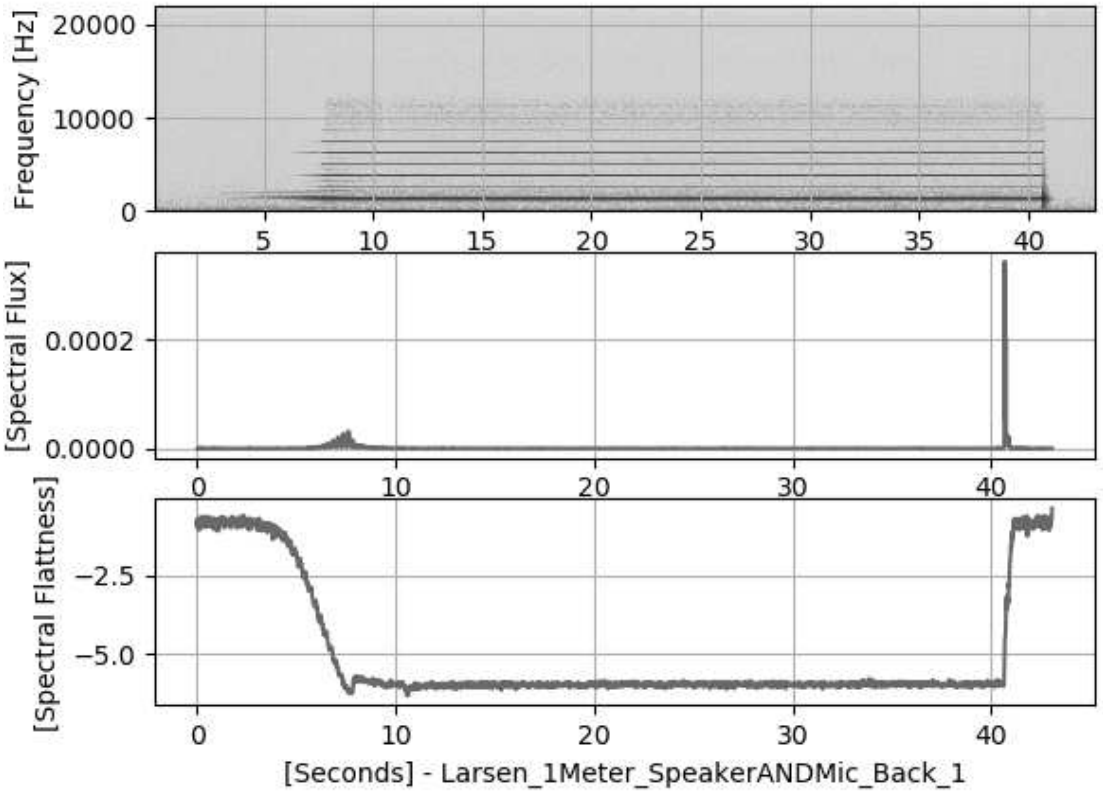

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_80cm_2

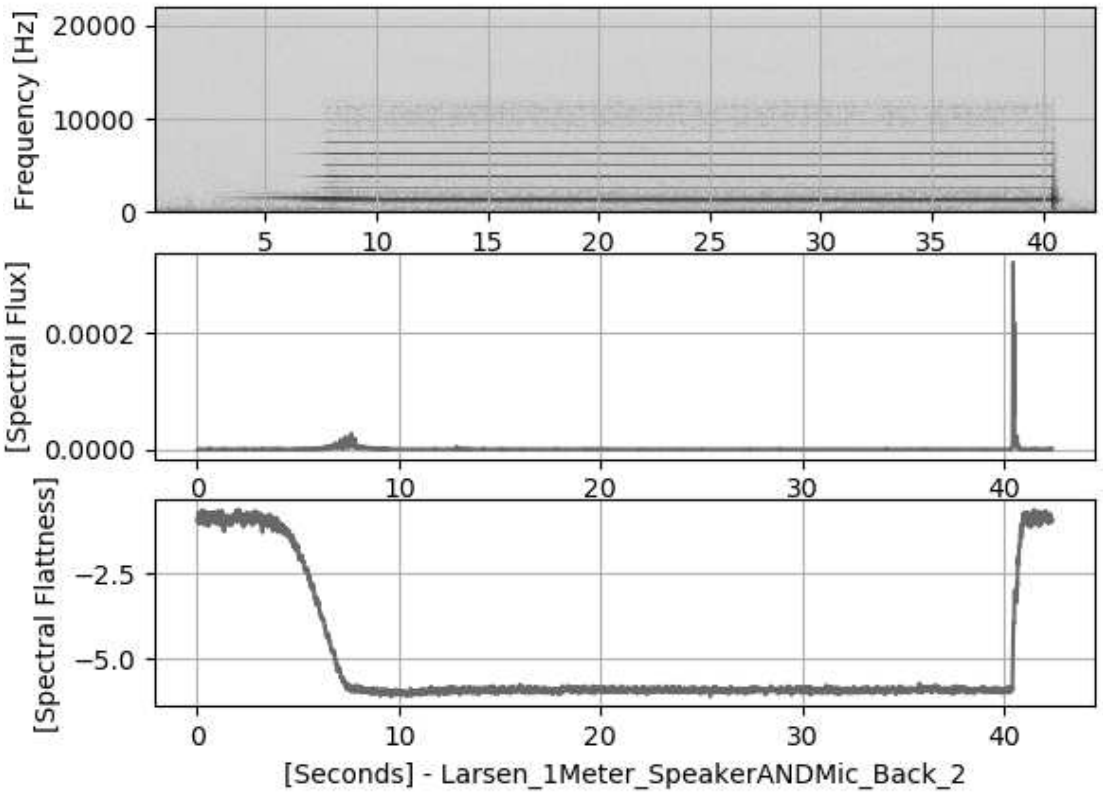

Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_1


Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
0.00050
0.00025
0.00000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_2


Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0004
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_Speaker_Back_1


Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
0.00050
0.00025
0.00000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_Speaker_Back_2


Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_1


Frequency [Hz]
20000
10000
0
5
10
15
20
25
30
35
40
[Spectral Flux]
0.0002
0.0000
0
10
20
30
40
[Spectral Flattness]
−2.5
−5.0
0
10
20
30
40
[Seconds] - Larsen_1Meter_SpeakerANDMic_Back_2

Considerazioni sulle analisi: Spectral Flux, Spectral Flatness

Le considerazioni che possiamo fare a proposito delle analisi appena svolte sono le seguenti. Tutte le analisi condotte sullo *spectral flux* e *sulla spectral flatness* mettono in evidenza dei comportamenti uguali che si ripetono in tutti i Larsen generati. La *spectral flatness* è un valore costante che si stabilizza ad un livello molto basso, negativo in tutte le registrazioni, questo come si può leggere sul testo di Lerch vuole dire che si ha davanti uno spettro - *molto coerente -,* difatti nella piattezza spettrale un valore che corrisponde allo 0 sta ad indicare la massima - *rumorosità* - che può essere oltre a quella del rumore bianco idealmente inteso, quella del silenzio o del rumore di fondo, che in tutte le registrazioni si manifesta prima della comparsa del Larsen; via via che il Larsen emerge si raggiunge un valore costante, molto *rigato*, che rimane tale dopo l'intervento del controbilanciamento quando il Larsen entra in un regime stazionario. Conferma definitiva di uno stato stazionario costante del suono ce lo porta infine l'analisi condotta sul flusso spettrale, dove un valore prossimo allo zero vuol dire che si ha a che fare con una bassissima variazione dello spettro e rugosità del suono, e conferma lo stabilirsi del suono in un regime stazionario.

Conclusioni

Per concludere possiamo affermare che l'effetto Larsen, quando si trova in un regime controllato di autoscillazione, è un segnale stazionario, un vero e proprio oscillatore fortemente dipendente dalle condizioni iniziali nel regime transitorio, ma meno dipendente dalle perturbazioni esterne una volta raggiunto questo regime stazionario di autoscillazione. A causa della sua forte sensibilità alle condizioni iniziali non è prevedibile la frequenza fondamentale dell'oscillazione prima della sua generazione, il che lo rende sostanzialmente differente dal controllo che si può applicare su di uno strumento musicale dove le altezze desiderabili sono determinabili a priori. Ma la ricorrenza di parziali sempre simili, che ne definiscono un timbro, del comportamento del sistema sempre uguale che si presta facilmente ad un controllo del tipo che ne abbiamo fatto in questa ricerca, ed in generale di tutte le sue caratteristiche ricorrenti, lo rendono una risorsa interessante per l'utilizzo in ambiti musicali, come ad esempio nell'ambito dell'elaborazione numerica del segnale in modo artistico, dove il progetto di determinate catene elettroacustiche ed elaborazioni che si possono fare su degli strumenti musicali possono produrre risultati timbrici nuovi rispetto al segnale originale ma sempre ricorrenti e prevedibili. In modo esattamente speculare il Larsen si può elaborare artisticamente, rendendo prevedibile il fenomeno da un punto di vista timbrico, ma imprevedibile nelle sue variazioni di frequenza fondamentale. Eventuali non linearità possono essere introdotte artisticamente aumentando l'entropia di questo comportamento. Un ottimo esempio artistico del Larsen si può vedere realizzato all'interno delle opere di Agostino Di Scipio, soprattutto nel ciclo di lavori *"Audible EcoSystemic"* dove il Larsen diviene la risorsa alla base del disegno di sistemi complessi di elaborazione numerica del segnale (in DSP).[5][1]

Sitografia

- [1]
  DI SCIPIO AGOSTINO -
  A relational ontology of feedback. Situated sound-making in the Audible EcoSystemic project.
  ECHO: Issue 3 -Feedback, Author(s): Adam Pultz Melbye (ed.) Publication year: 31 January2022
  Orpheus Instituut - ECHO: Issue 3 - Feedback, Author(s): Adam Pultz Mel-bye (ed.) Publication year: 31 January 2022
  link:
  https://echo.orpheusinstituut.be/article/a-relational-ontology-of-feedback

- [2]
  Fonometri: guida completa:
  https://it.rs-online.com/web/generalDisplay.html?id=idee-suggerimenti/guida-fonometro

Bibliografia

- [3]
  Udo Zölzer
  Digital Audio Signal Processing
  second edition.
  Ed. Wiley
  20 giugno 2008

- [4]
  Alexander Lerch
  An Introduction to Audio Content Analysis:
  Applications in Signal Processing and Music Informatics
  2012-08-14

- [5]
  DI SCIPIO AGOSTINO ‘Sound is the interface’:
  from interactive to ecosystemic signal processing,
  Published online by Cambridge UniversityPress: 21 April 2004,
  link:
  https://www.ak.tu-berlin.de/fileadmin/a0135/Unterrichtsmaterial/Di Scipio/Sound is the interface.PDF

Documentazione

tutti i materiali della ricerca sono reperibili al seguente link:
https://github.com/LucaSpanedda/Analisi_Effetto_Larsen.git