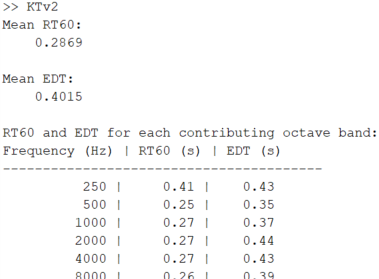
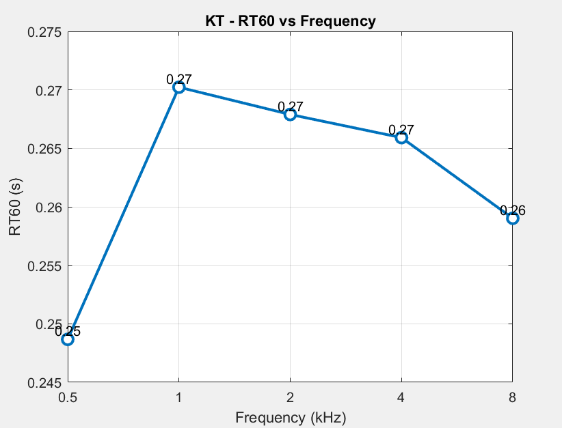
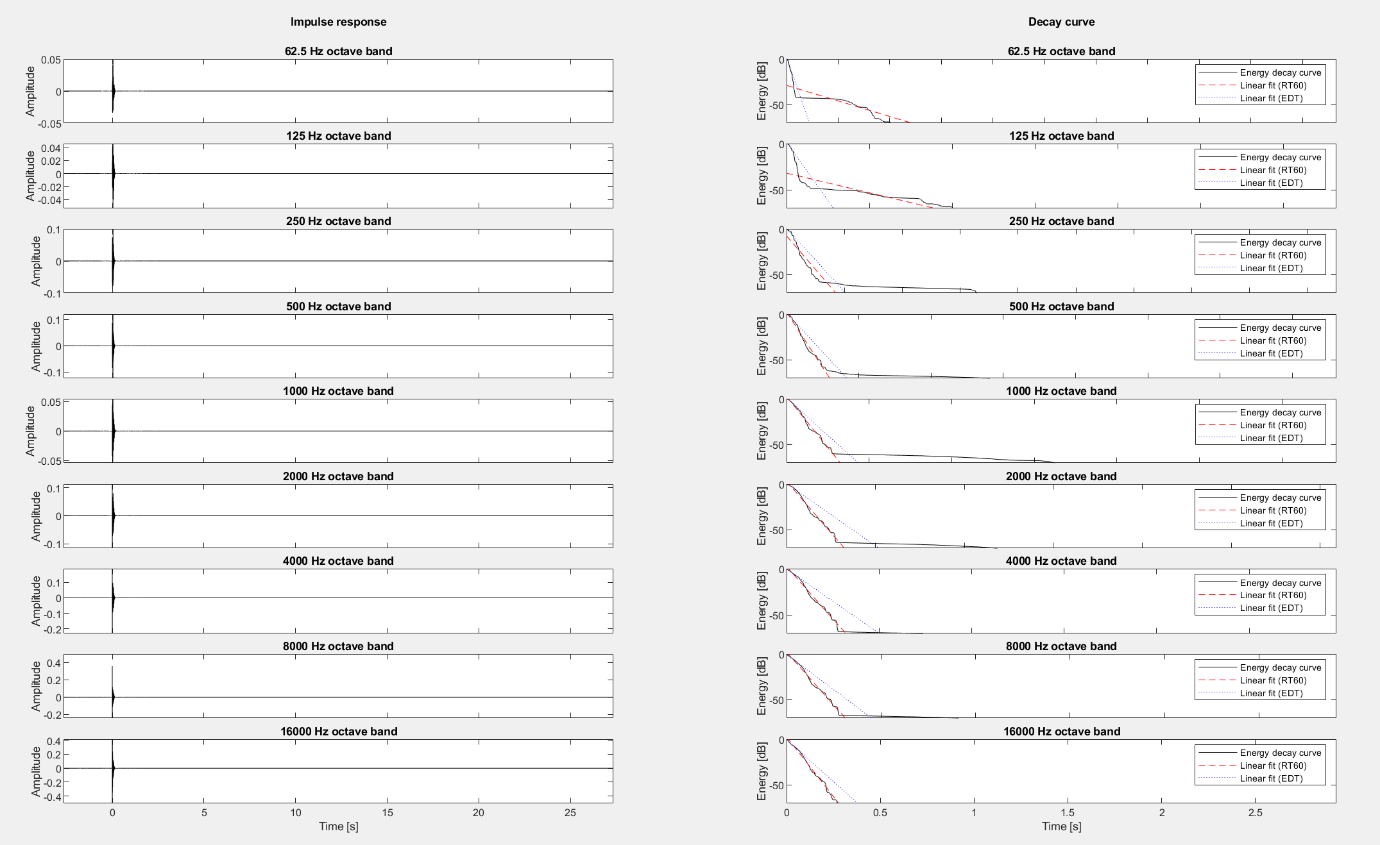
legends: Orange (Medium Priority), Red (Critical/High Priority), Blue (Info), Green (Good work!)

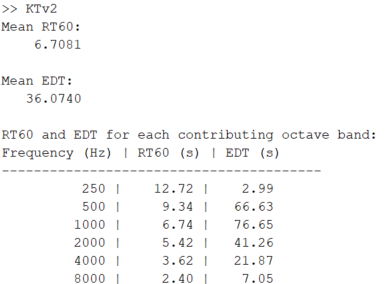
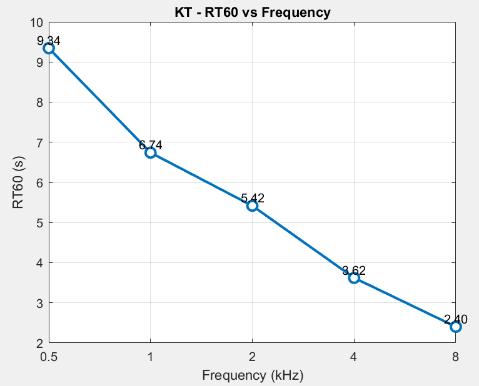
Done, Important

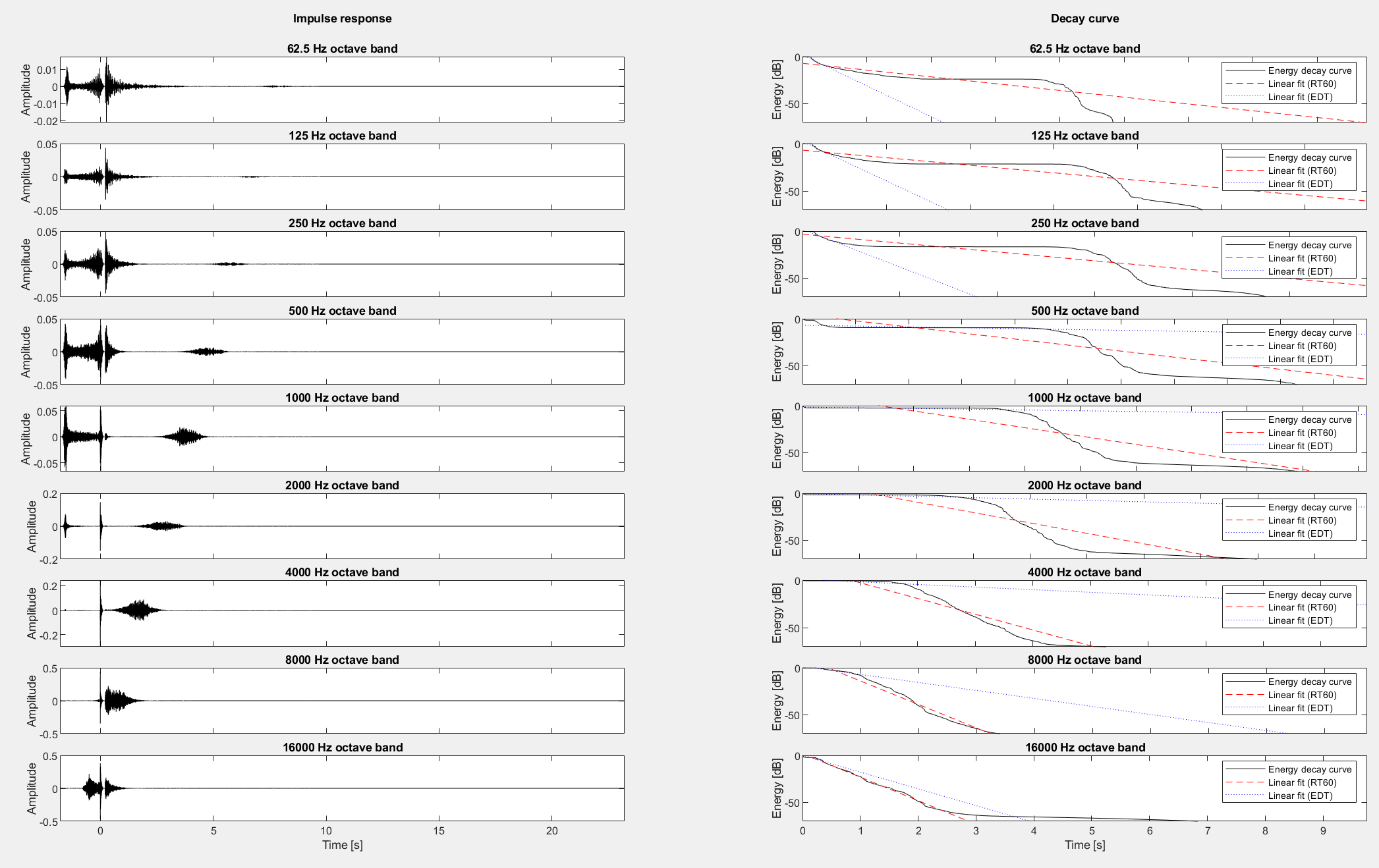
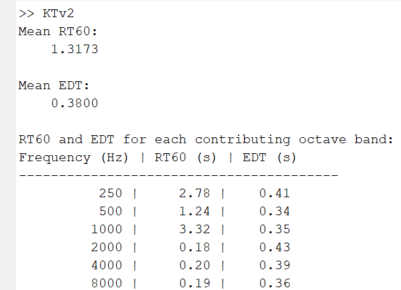
# Monday, 12 August 2024

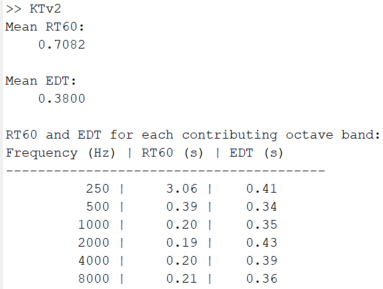
Refreshing what I had got last week and reconfirming the results.

KT\_improvedsweep\_vol0.1\_1to1.wav result w/ default y\_fit (RT60):  
As seen, it was too low for RT60, EDT is correct however. No audio clipping.

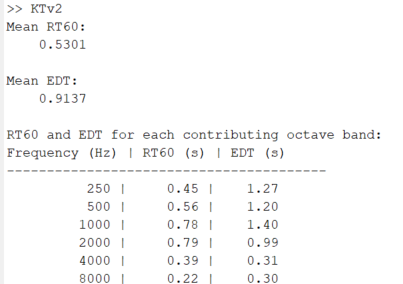
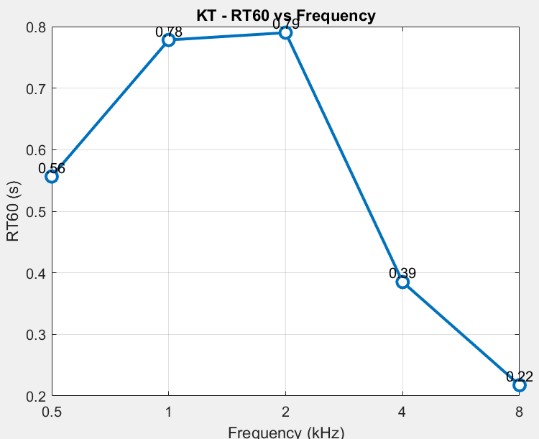
Let’s first increase volume only, from 0.1 to 0.5:

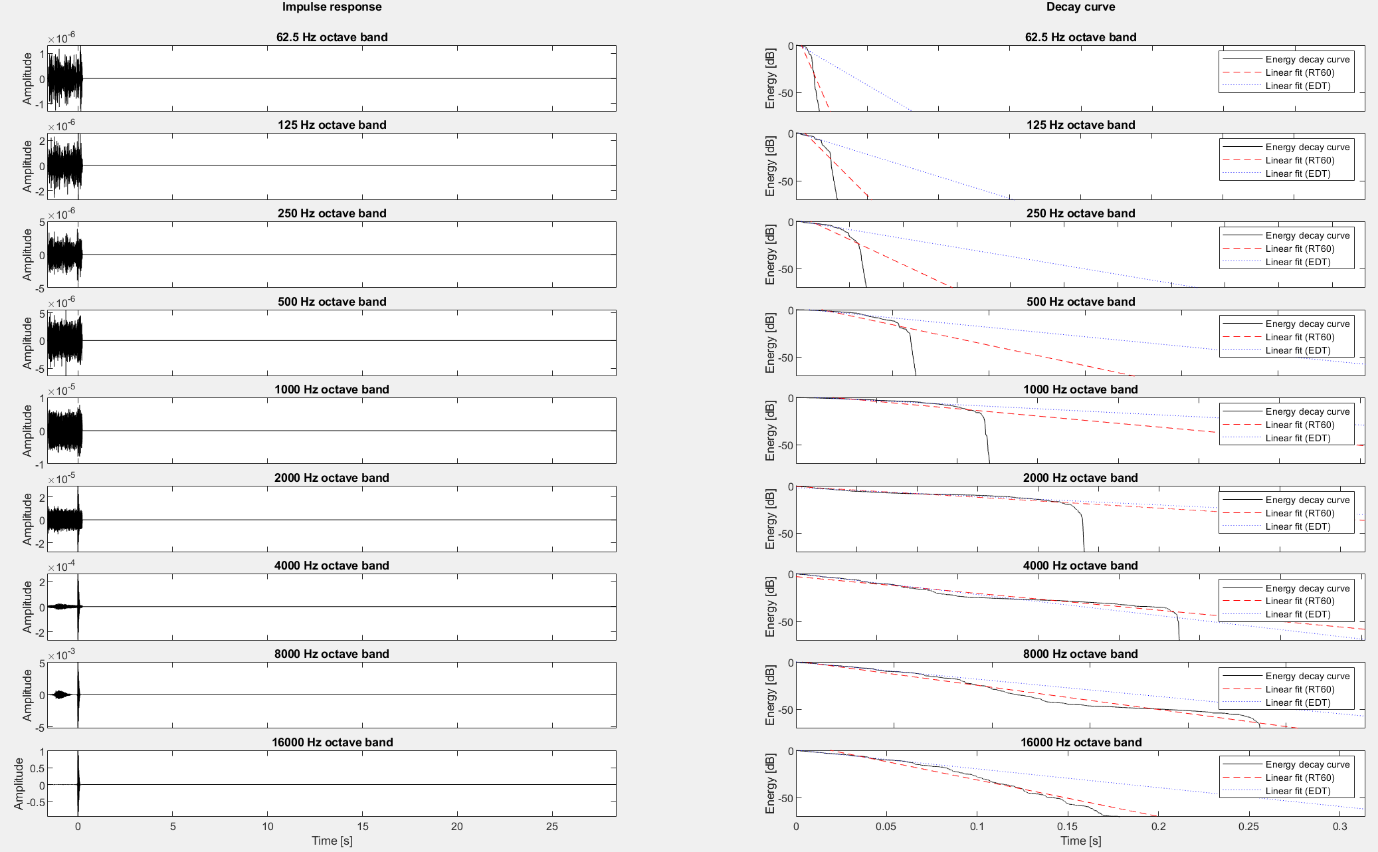
 Now its too high, and (probably due to audio clipping?) we got lots of distortion and weird result…

Next lets try volume at 0.13, ratio of direct to reflection mix still at 1 to 1, default y fit and RT30:

Still not good, even when y\_fit to only RT30, it still shows bad response for some octave band esp higher freq (too low) so its not reall good either…, and 250Hz still too high due to high noise floor level..

Thus, lets try improve the sine sweep again instead. The main imrovement is using longer sine sweep duration (15 sec instead of 10) to give better resolution and power(?) for lower frequency

OK, very interesting result (positive imo), using vol0.1 and 1to1 ratio on default RT60 y\_fit:

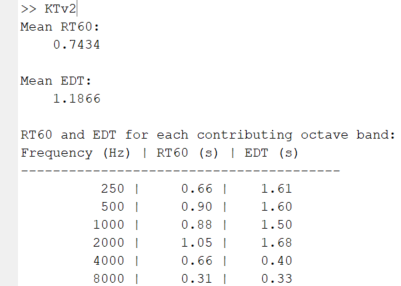
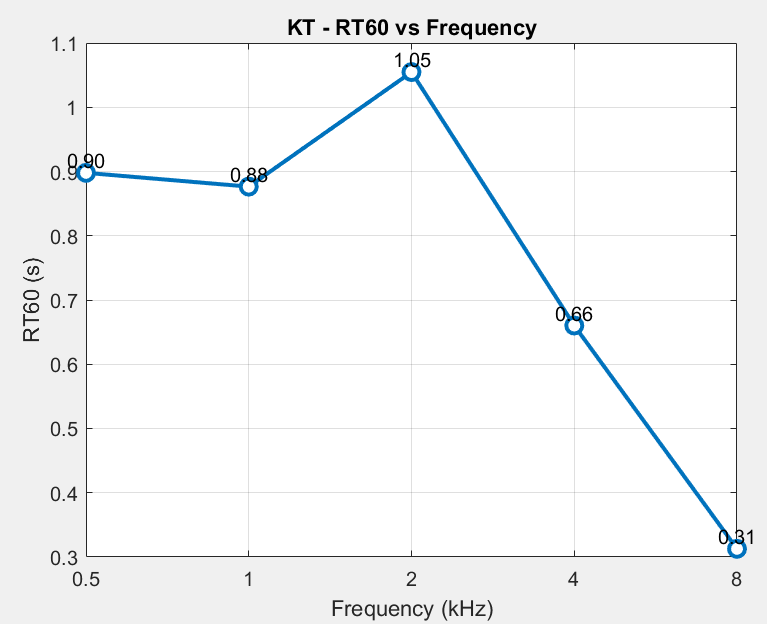
As seen, this is an improvement! Next I think it easier and better to keep improving the sine sweep signal instead of trying trial and error on Unity audio source (also because we still have issue as seen below, 1. Distortion on RIR before 0s on all freq band, also possibly causing high EDT, and tad bit too low RT60 on 8kHz, and too high on 1 and 2kHz bands.)

Yea, that’s good progress for today, lets call it a day, and tomorrow lets continue optimising this sine sweep.

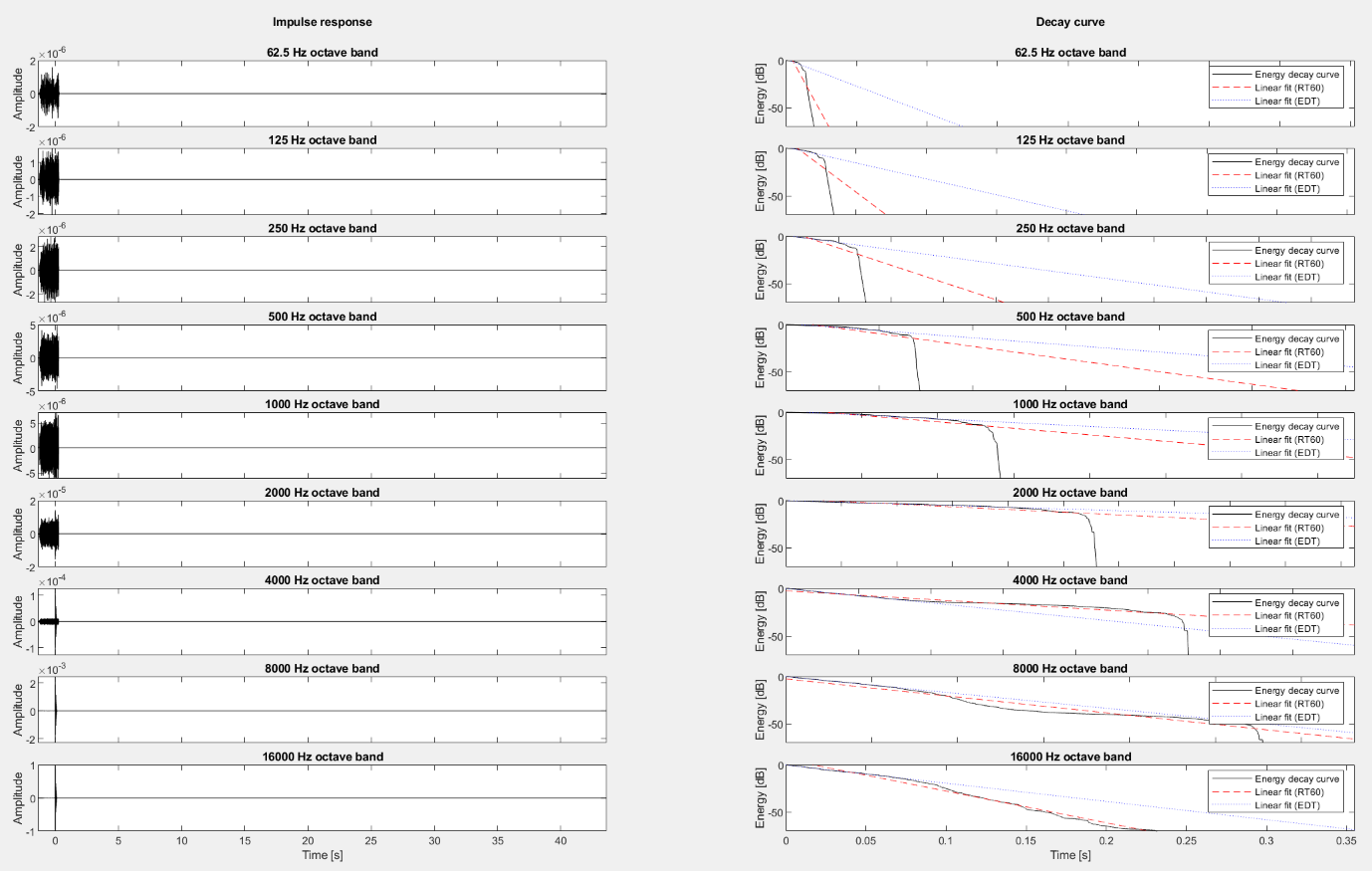
# Tuesday, 13 August 2024

improved\_sine\_sweep\_v3.wav improvements:

* Increased duration to 20 seconds for better low-frequency resolution.
* Extended frequency range (10 Hz to 22 kHz) to cover a wider spectrum.
* Increased silence at start and end to 1 second each, helping to reduce pre-ringing artifacts.
* Adjusted tapering for smoother start and end of the sweep.

Interestingly enough, this seems to make it worse in all aspect and didn’t solve anything:

As seen below, the distortion on RIR for all band still exist before 0s, and RT60 increased for all bands (not ideal) with 8kHz still being a tad bit lower than it should but also increases. Maybe I should also modify the deconvolve function?

In that case, lets revert the changes (v4) and modify deconvolve as well.

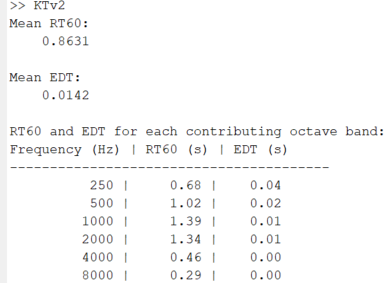
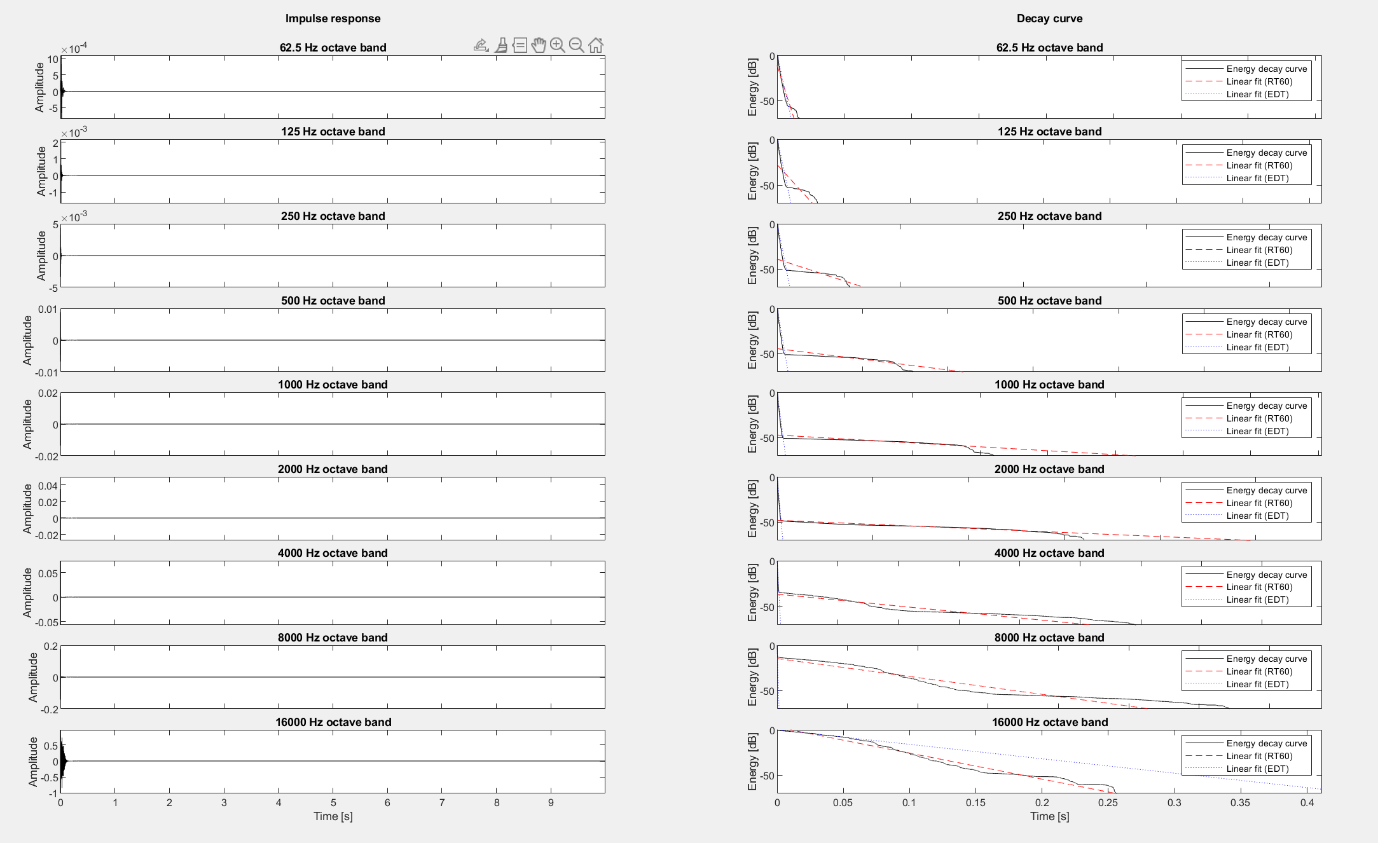
Ok, it went back to Mona’s like response… Idk, for now lets disable all HRTF/binaural/stereo and make it pure mono for RIR measurement. NVM, for some reason, changing stereo to mono in project settings cause audio to not work at all. So for now, I’ll just disable HRTF/binaural for RIR (Eval scenes) and see if this improve anything. 

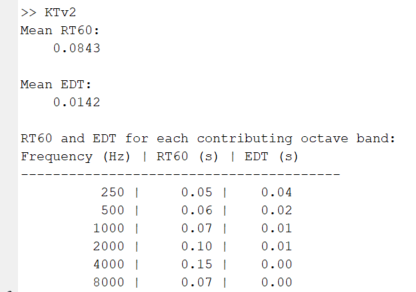
OK, that should be fine because as seen above, both channels are now the same so technically mono, compared to the one on top (with HRTF).

I think this does help a bit in terms of EDT but sweepv4 is really bad so lets go back to v2 and see with noHRTF setting.

Not OK, now the EDT is too low around 0.0142 s… and RT60 is still wrong… tbh I have no clue now. Let’s just commit this for safe measure and declutter some stuff.

Rewrote the new main\_KT ipynb to be more modular. Still having same problem (unbalanced RT60 bands) and too low EDT even using newly generated sine\_sweep.wav.

As seen on the left and below especially, it doesn’t feel like the RT60 line is correct as its measuring the noise floor instead(?).

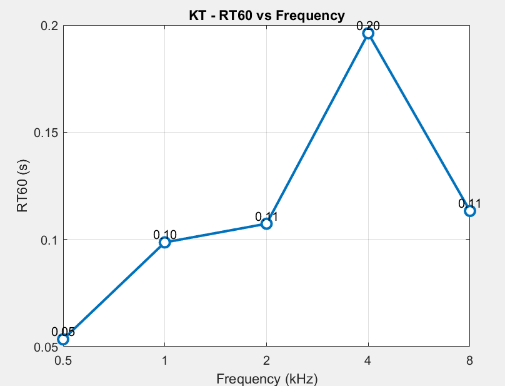
Last one for today, lets use the sweep from original AVVR GDP sound folder (sweep.wav). 🡪

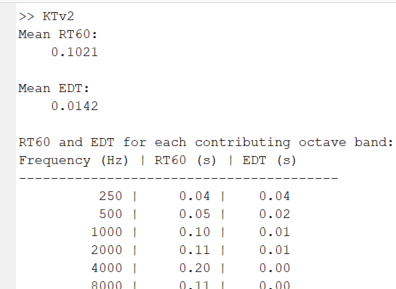
Ok pretty interesting.. its very low but at least consistent and maybe tuning audio source parameter can fix it (reflections mix level and maybe even volume?)

That’s for today, will explore theory above tomorrow, and ready questions to ask Dr Atiyeh.

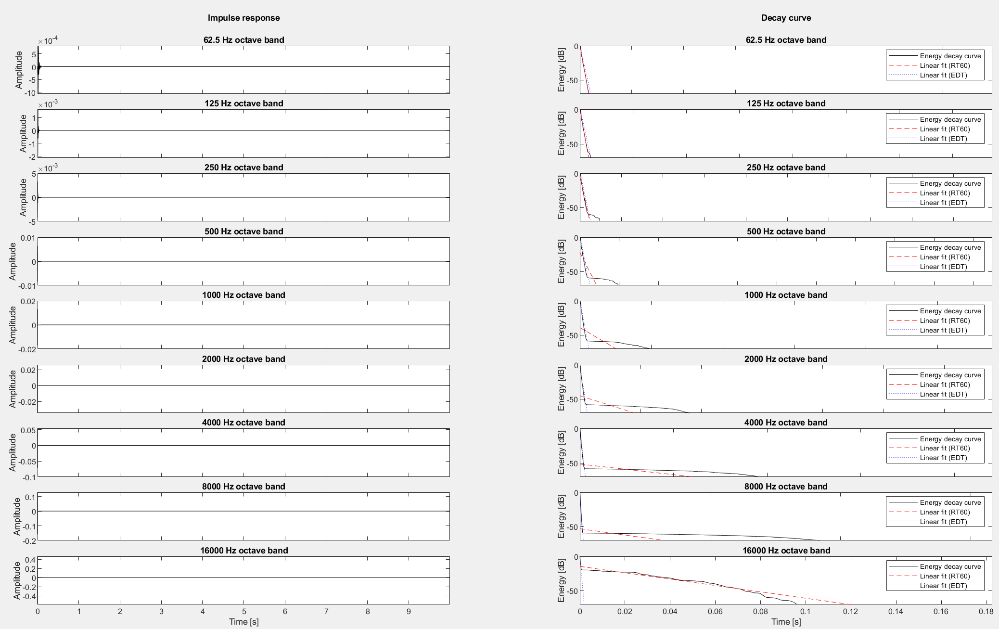
# Wednesday, 14 August 2024

Ok, lets change one variable at a time, first reflections mix level from 1 to 2:

Well it increases slightly but obviously not the solution we’re looking for imo:



As seen below and on table on the left, it still is too

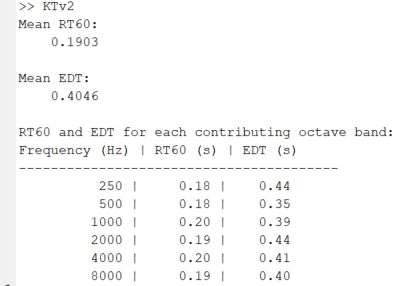
low, and the best line fit doesn’t look right either imo.

## Questions to ask/confirm with Dr Atiyeh:

1. How should a correct decay curve and impulse response look like?   
   - Non sinusoidal envelope for RIR, and decay curve should be with very low noise level.
2. The RT60 across all octave bands should be pretty similar (like Kim21 result) right?   
   - Yes
3. Is a best line fit for RT60 going over noise floor acceptable?   
   - No, either lower noise level or use y\_fit
4. Why is there a noise floor when the simulation should not have any other sound source, is this normal? If normal, can y\_fit be used?   
   - Unsure why theres noise, maybe because volume/ambient from sine sweep itself?  
   - Yes y\_fit can be used. Mona’s ST work pretty well on y\_fit [-5 -55] tbf
5. Sine sweep parameter, is there golden version or it differs case-by-case basis and how does deconvolution works?   
   - No, as long as it includes all frequency with same amplitude.
6. Is the inverse filter and RIR generated looks correct?   
   - No, this is the reason why mine is wrong (Mona’s was correct) but solved, inverse filter is just inverse of sine sweep for deconvolve/cancellation, transfer function to get H(f)/RIR.
7. Is disabling HRTF to get mono-channel like response a right call for RIR measurement?   
   - Yes
8. Is audio clipping acceptable?   
   - No, make sure its within range
9. What causes distortion before 0s usually on Impulse Response?   
   - Unity/simulation related problem probably. Still unsure.
10. Do I need to manually correct/adjust/optimise sine sweep for each scenes/my simulation, even though there should not be any noise as it’s a simulation.   
    - No need, just use same sine sweep for every scene.

Fix deconvolve method! Should not have sine envelope. It was because my inverse filter is wrong!

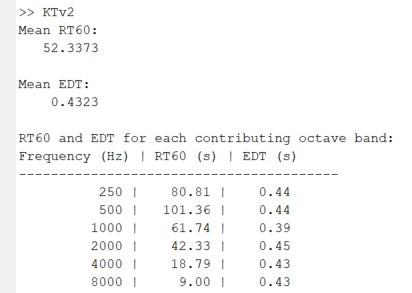
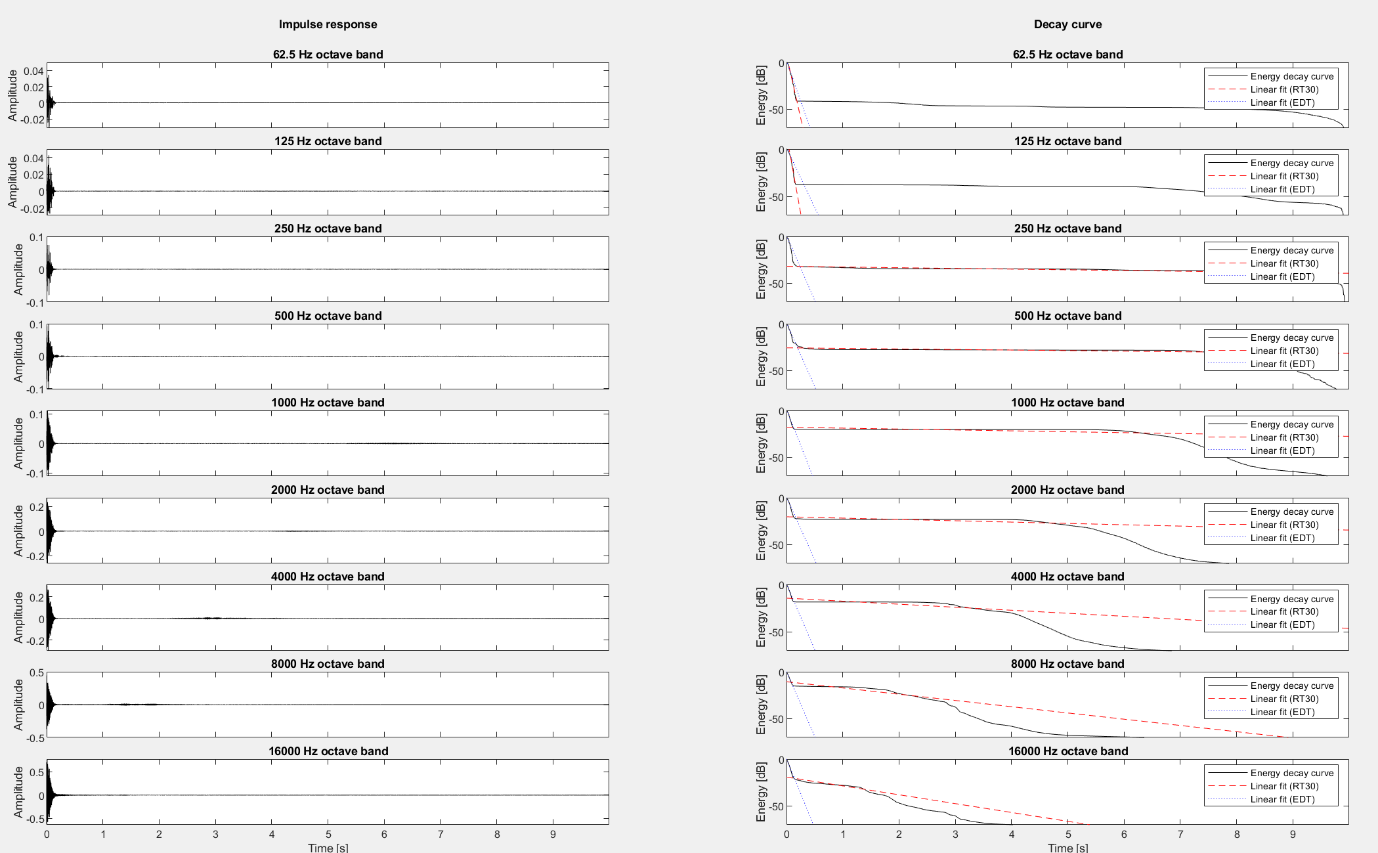
Rewriting the new KT ipynb to be more compact and correct.

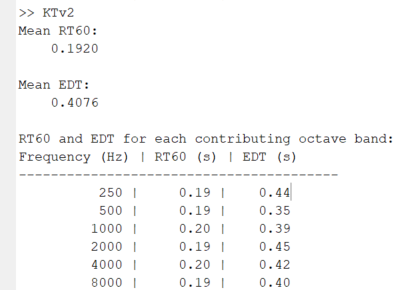
Commented some functions which deemed not useful? Need more testing… Gonna continue tomorrow. Best I got today using y\_fit 30 which obviously still wrong as EDT is higher.

# Thursday, 15 August 2024

Let’s figure out how to fix EDT, and maybe increase reflection mix level to get higher RT60.

Using reflections mix level 2 (ratio 1 to 2), vol 0.1, no HRTF:

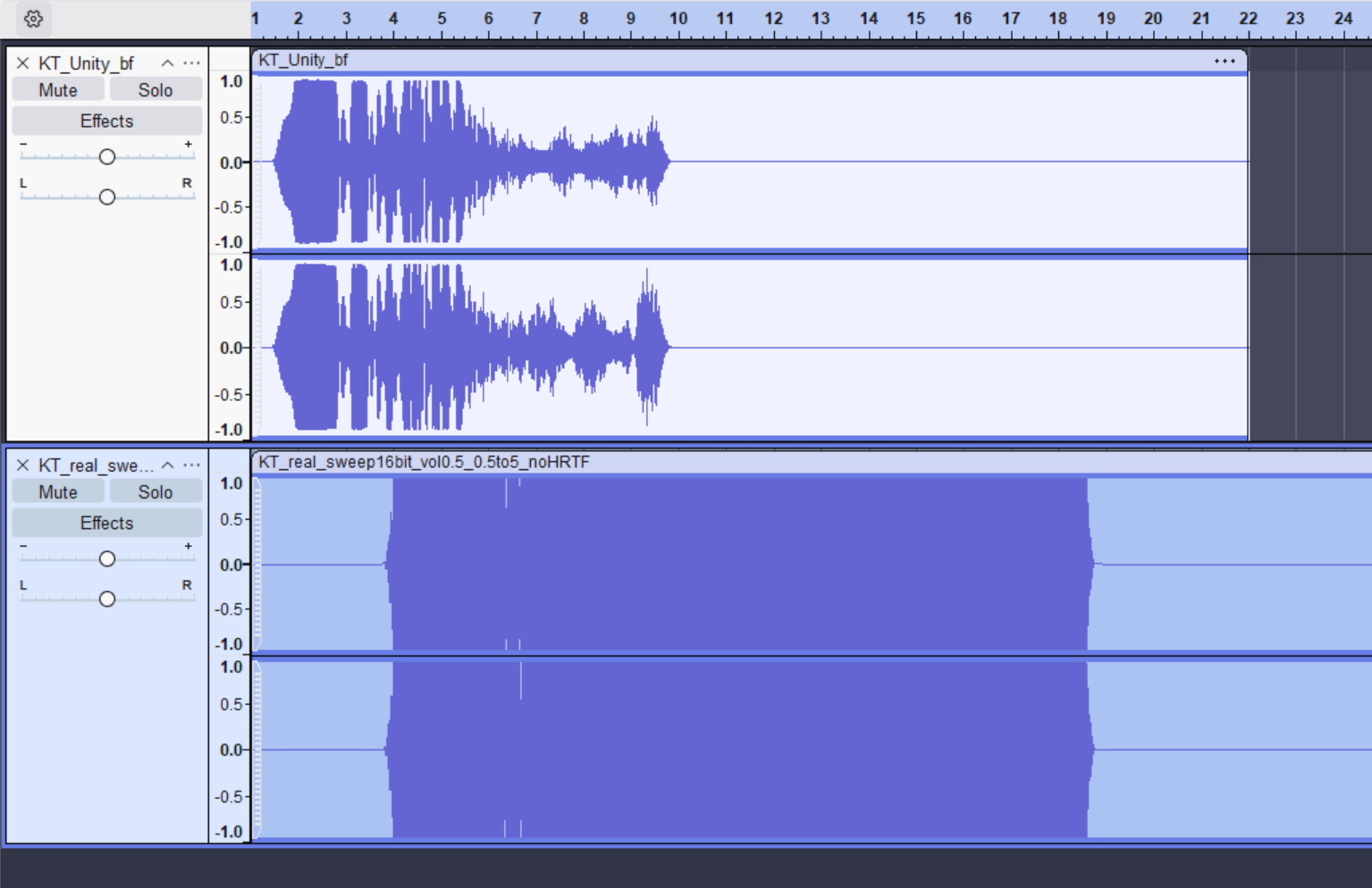
This is a dead end, even using y\_fit [-5 -35], I got such high RT60, because it seems increasing reflection mix level directly increase the noise floor.  
Even using lower y\_fit like [-5 -15] which is not only wrong for RT60 (too low, might as well be EDT v2), it still is wrong because EDT is higher for some reason.

I don’t think simply increasing volume would do anything.

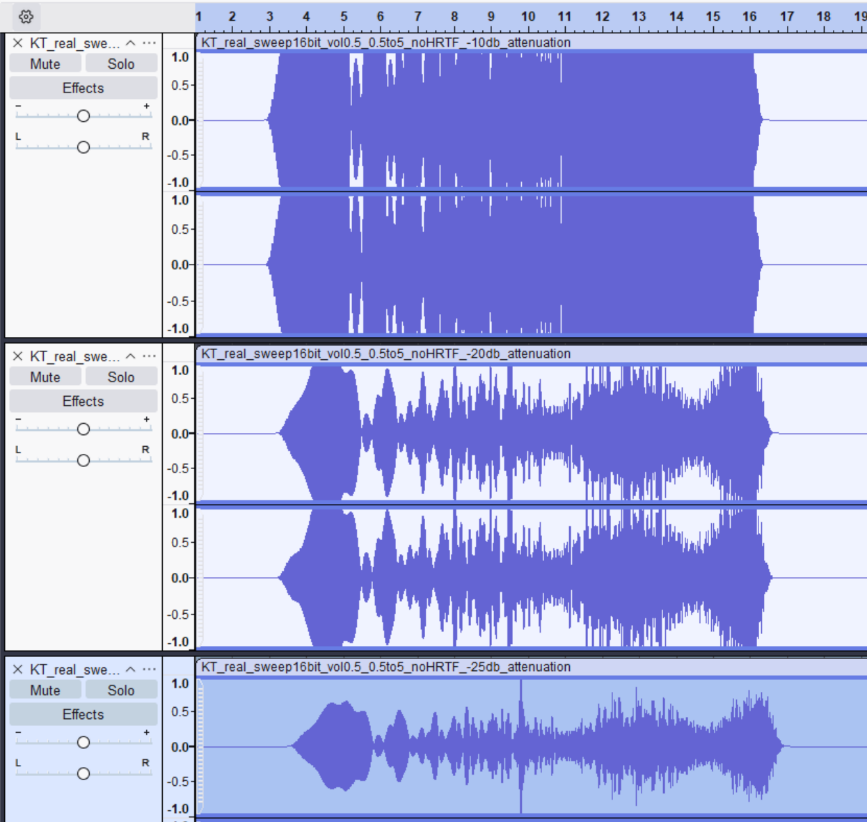
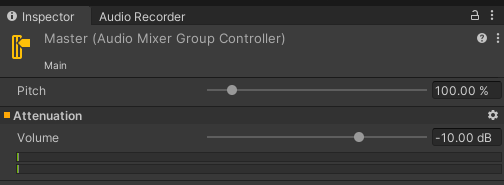
So, let’s modify the sine sweep, normalised it to have lower amplitude. Unfortunately, barely any difference (on same setting, vol 0.1, 1to1, no HRTF) and y\_fit [-5 -35]

Let’s increase volume to 0.2

This also increases the noise level, idk man I think something is horribly wrong with my code, lets just copy mona code and make it more compact. And start again using Mona’s setting.

Interestingly enough, Mona’s recorded response on her setting don’t really clip that much while mine is clipping like crazy. 

And considering Mona don’t really know what setting (volume mixer) she’s on, its now just guessing game for me to figure out why mine is maxing while she’s not (could even be due to how mesh work or maybe I even mess up steam audio somewhere…)

Anyway, instead of dreading over this, one workaround is to use volume mixer in unity itself and give attenuation troughout, imma try -10dB first using Mona setting.

As seen, -10dB attenuation is not enough, -25 is abit overkill, so let’s try analyse -20 first (though it still have some clipping.)

Nope even this doesn’t work, still got way too high RT60 due to noise floor level problem…

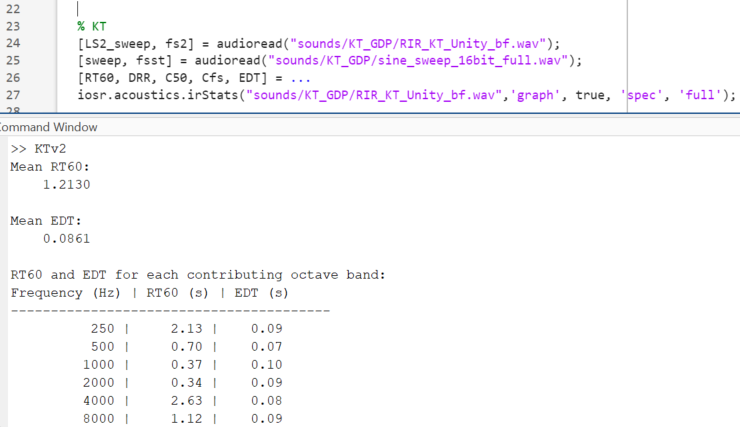
This basically means anything we do is not gonna work because it will increase noise floor level regardless so yea im stumped.. Still don’t know why EDT is higher either..

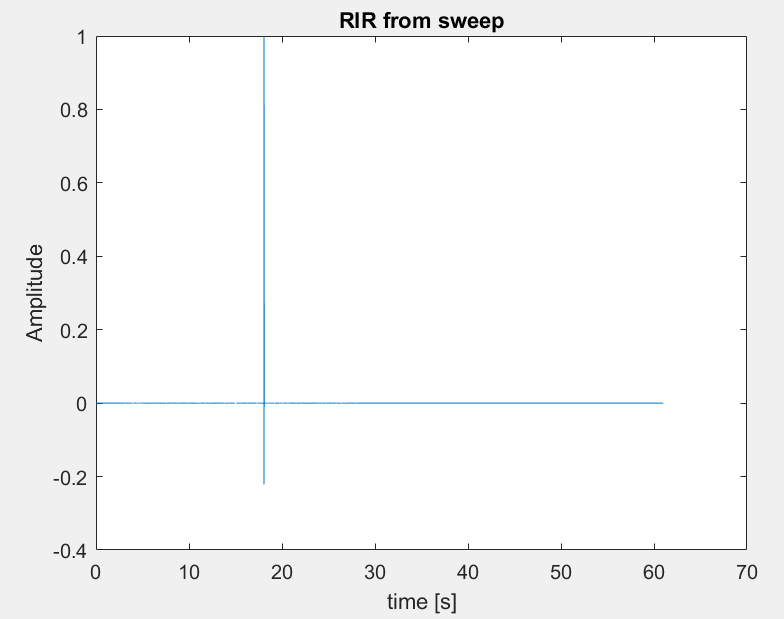
This noise floor can be seen in RIR in time domain graph (Figure 2 of matlab script) that the reverb tail is longer and higher…

# Friday, 16 August 2024

Let’s go back to drawing board and create test audio project/scene to see if maybe its steam audio broken and also test on Kim21 and groundtruth meshes as well..

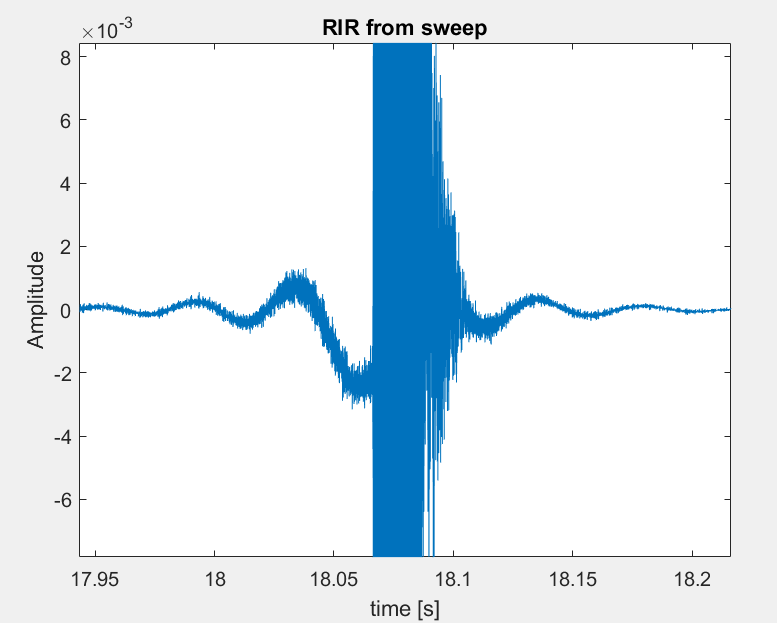
Let’s first create test empty scene open air (low reverb) scene with floor..

Ok, this confirm that the deconvolve of real part in Main\_KT ipynb does work properly, though we still have a noise floor (presumably from the floor reflections) which still mess up graph line best fit for RT60 on default y\_fit.

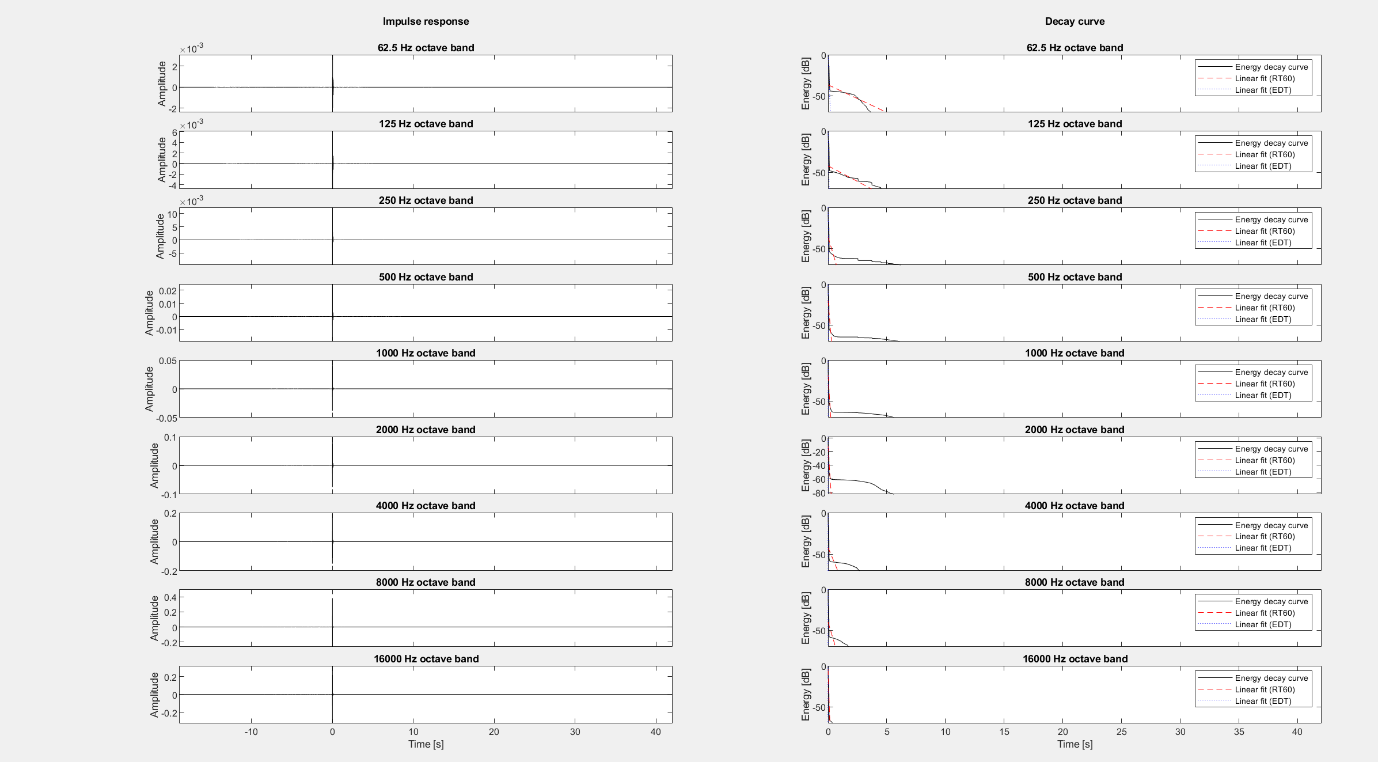
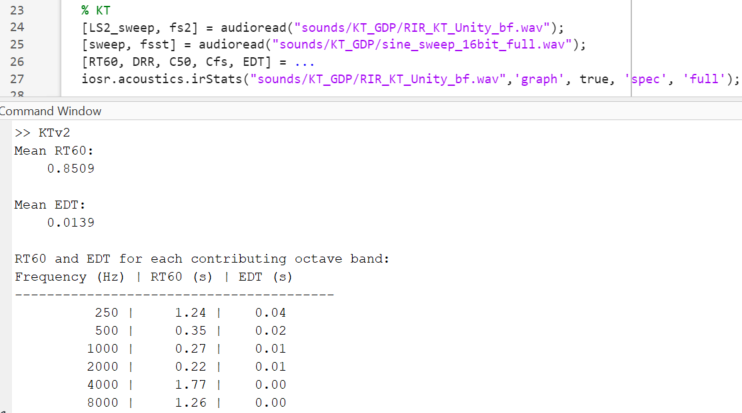
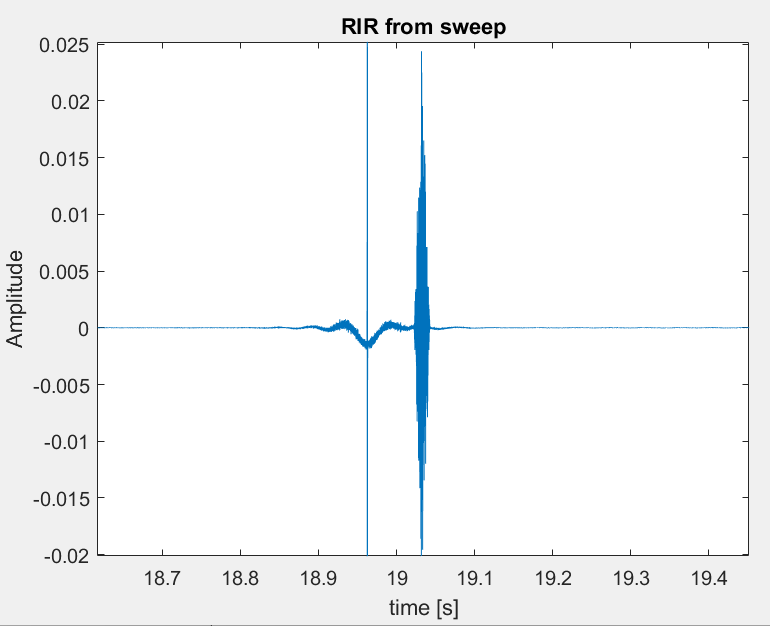


RIR looks very clean as expected (unzoomed)

When zoomed, we can see the short reverb tail.

Next, lets try without Floor at all. To make sure probe is still generated, instead of removing the floor which is needed for generation of probe, I just moved it down 10 meter, and increased height of probe 10 m.

As influence radius is 7m, this should make it seems as if its high on air for no reverb at all.

BIG finding, even one with no floor have small reverb and noise floor…

Even more interesting is that second peak… But it doesn’t really shows on the octave band graph for some reason.. But this means the noise floor is not due to reverb at all or that there’s something fundamentally wrong that needs to be dealt with, unless this is normal behaviour (I don’t think so but better email Dr. Atiyeh for confirmation)

Next, lets reconfirm this behaviour by replicating it on original Mona’s code.  
Then, measure RIR on ground truth and meshes from S3A project files just to be sure.  
Look into RIR measurement in Unity to see if anyone have ever done it!