Audio Denoising Tutorial

This GitHub repo contains sample audio files recorded in adverse conditions to practice denoising. The goal is to make these two audio files recorded by different talkers in different conditions through different microphones sound as much as possible like they were collected in a controlled environment. This process as laid out here uses three separate third-party programs: Praat, iZotope RX, and Adobe Audition. The general workflow is:

1. Low pass
2. High pass
3. Denoise
4. Remove reverb
5. EQ
6. Normalize volume
7. Add uniform noise

1. Low pass

The most informative acoustic cues for speech perception generally live below about 12,000 Hz. Cues higher than that are also informative but generally contain more information about the space where a recording was made than about the speech. That means it's a good idea to low-pass audio at around 12,000 Hz.

To "low-pass" audio means to define some cut-off point below which all frequencies are still audible, but above which, frequencies are removed. This can be implemented in Praat relatively straightforwardly. To low-pass manually, open a sound file in Praat, and then from the objects window, select the audio file, and click "Filter" in the menu on the right. Then select "Filter (pass Hann band)…" This will open a new window where we can specify the frequency range and the amount of smoothing. We don't want to remove anything from the low end, so we set "From frequency" to "0". We want to low pass at 12 kHz, so in "To frequency", we can enter "12000". The parameter "Smoothing" specifies how abrupt the cutoff is. We want the filter to be effective without sounding harsh, so an appropriate value here may be around 300 Hz. Running the script outputs a new, filtered audio file, which you'll need to save.

Since your low-pass procedure is the same for all talkers, it's easier to automate this process than to do it manually for each audio file. In the folder "praat scripts", there's a file called 'lowPass.praat' that does just that. You can open that file in Praat and then click run. There are two text boxes to fill in. The first, "Dir name", takes the directory containing all the audio files (ending with a slash), and the second takes the output directory where you want the filtered audio to be saved.

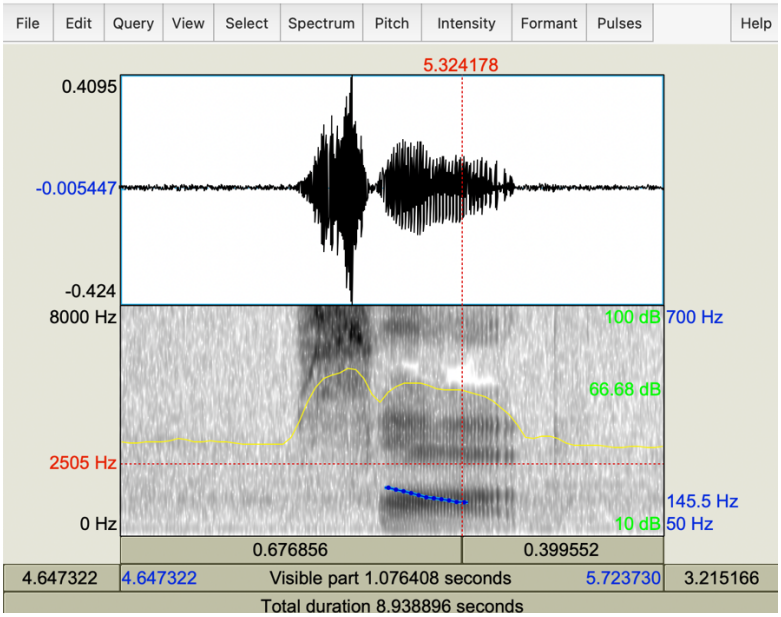
Note that you should never overwrite audio from previous steps! If you save the filtered audio over the original audio, it is gone! Depending on your processing pipeline, that may mean that you have to re-record if you make a processing error! This case, you should create a new directory alongside your original audio files called “low_pass” and then specify that as the output. Run the script, and that directory should be automatically populated with new, low-pass filtered audio files!

2. High pass

The upper regions of the audible spectrum are full of room noise, but so are the lower regions. In fact, anything below the talker’s lowest f0 is exclusively extraneous noise. To take care of that, we can use a process similar to a low-pass called a “high-pass”. High-pass filters, as you can guess, shave off low-frequency information and do not affect high-frequency information.

Our goal with the high pass stage is to remove as much low-frequency information as possible without cutting out anything informative. The lowest f0 value in talkers’ vocal ranges varies widely across people and contexts, so we can’t really automate this process. And it’s best to be conservative, because accidentally removing f0 information can make a recording sound pretty bad.

To estimate the cutoff point for the filter, open up a talker’s audio file in Praat’s “View & Edit window. Make sure that the pitch tracker is visible by going to the “Pitch” menu and then selecting “Show pitch”. Now you can scan through and look at different words to find the lowest f0. To do this, place your cursor on a low point and click. The frequency in Hz will display in blue on the right. Below, the frequency at the selected timepoint is 145.5 Hz.

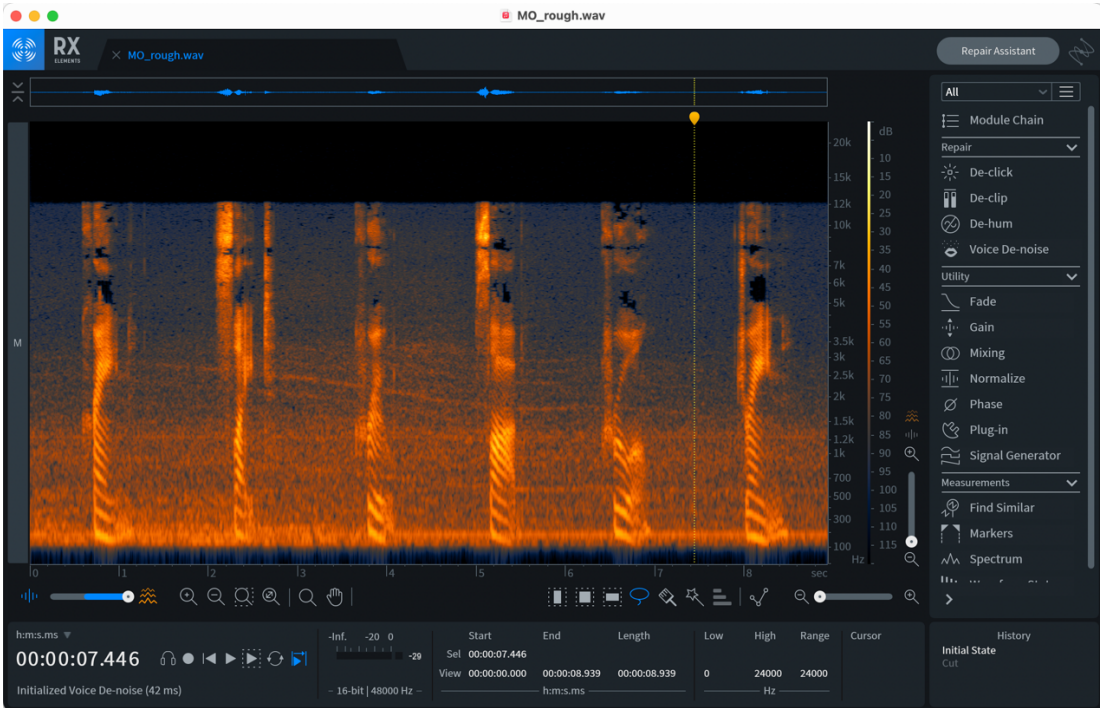


Scanning the file “MO_rough”, it seems that this talker’s lowest reliably measure f0s are somewhere around 140 Hz. Some lower frequencies are marked, but these are not reliable because they occur during periods of creaky voice. We haven’t sampled exhaustively, and we wanted to make this estimate conservatively, so let’s add a 15 Hz buffer zone and filter at 125 Hz. Select the audio file “MO_rough” in the objects window, and then use the same command from the low-pass stage, “Filter (pass Hann band)…” Since we’re high-passing this time, set the “From frequency” argument to the floor we determined—in this case, 125. Then set the “To frequency argument to some value that won’t interfere with our previous low pass. An appropriate value would be something like 20,000. Our smoothing value should be much smaller this time. (Remember that Hz are logarithmic.) Something like 10 may be appropriate. Click “OK” and a new file is generated in the objects window. To avoid overwriting your low-passed files, save it in a new directory, maybe called “high_pass”.

Do the same for the other file, “SJ_rough”. This talker has a noticeably lower voice. His floor seems to be around 100 Hz. I’d maybe filter from 85 Hz to 20,000 Hz with 10 Hz of smoothing. Process that file, and then save in the “high_pass” directory.

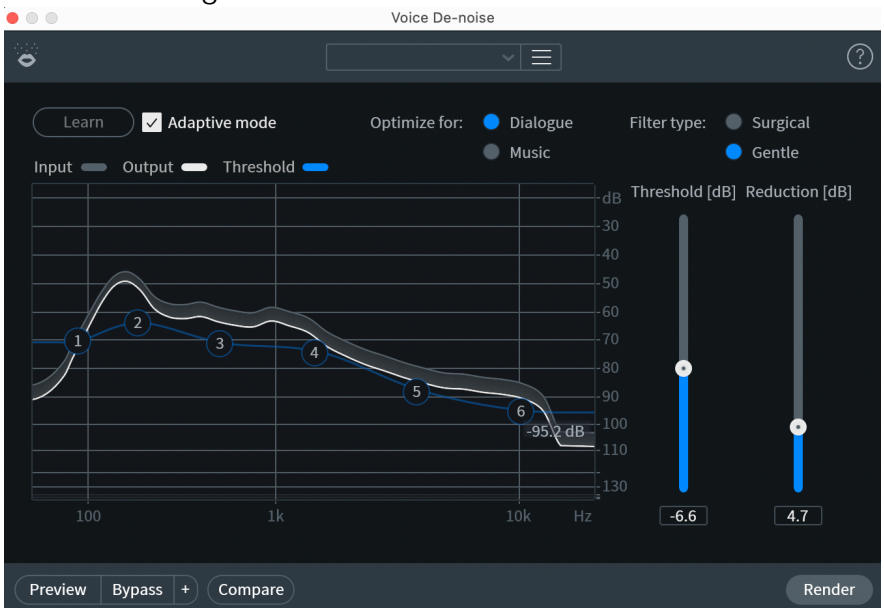
4. Denoise

All recordings have some kind of noise floor, but the noise floor is usually lower for recordings made in controlled environments with reasonably nice equipment. Lots of things can increase a noise floor, including hisses and buzzes introduced by low quality microphones, cables, or audio interfaces or ambient noise from busy streets, refrigerators, or air conditioners. Cutting into a noise floor without damaging the speech signal is an absolutely wild signal processing challenge, so we’ll need something a little heavier duty than Praat. I recommend a desktop app called iZotope RX, which I think we’ll have on one of the lab computers soon.



When you open up the sound file “MO_rough” in RX, you should see something like the above, with a spectrogram, player controls, and a menu on the right. You may also see a waveform in blue, but the spectrogram view will be more useful for this process.

To denoise, select all the audio with command-A. Then, from the menu on the right, select “Voice De-noise”. This will open up a new window with de-noise-specific controls. This looks something like:



A good place to start is with “Adaptive mode” turned on. This means that the levels of reduction in different frequency bands are determined by the content of the audio in a

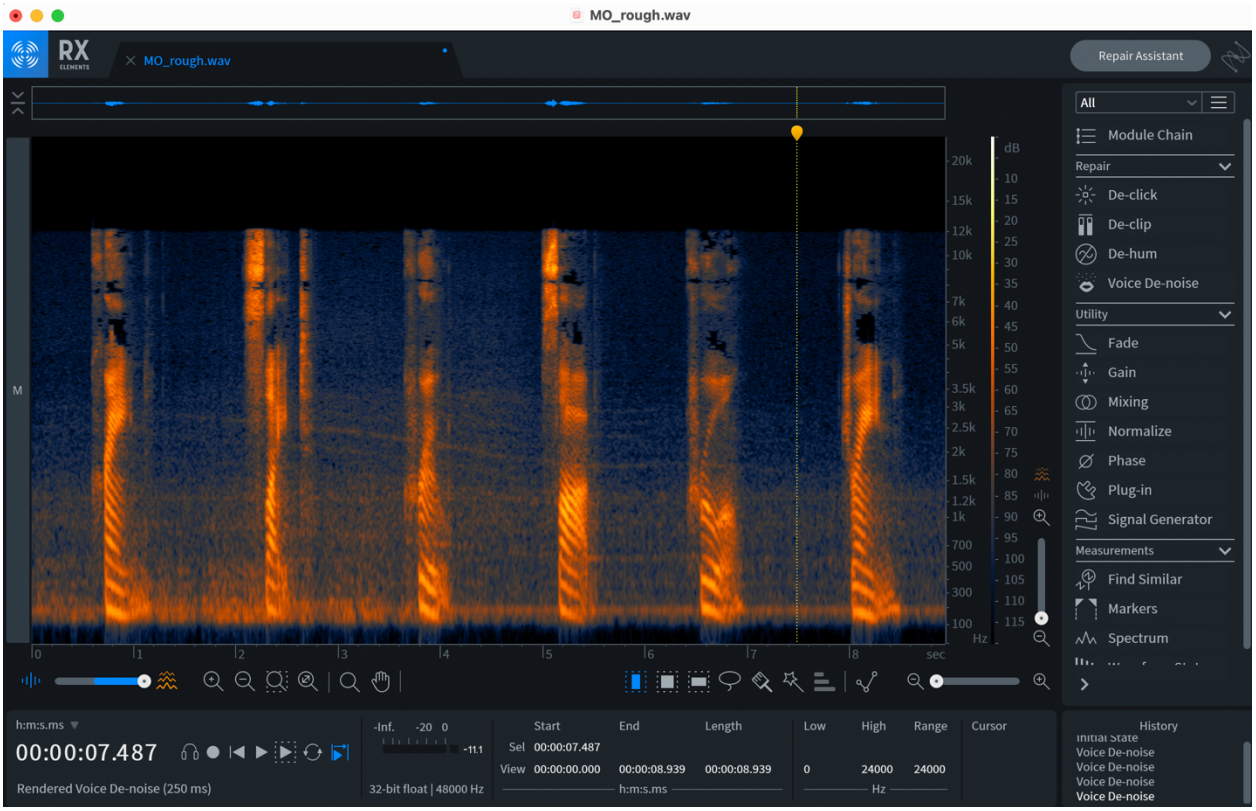
given time window. We can hear the results of the denoising by clicking “Preview”, which will play the selected audio with de-noising applied. In the visual representation, we can see the input signal represented by the gray line, the output by the white line, and the reduction by the space between. Each of the circled numbers represents the level of reduction at a given band. When “Adaptive mode” is turned on, we can see these change as the audio plays.

Most of the influence you have here is with the faders on the right for “Threshold” and “Reduction”. The fader under “Threshold” controls the level at which reduction kicks in, i.e. how loud does a sound have to be before it gets dampened? The “Reduction” fader controls the extent to which the signal is dampened. Both of these have trigger more intense reduction when turned up than down. It’s important to have a light touch with these controls. Several light passes will sound better than one heavy pass. When the reduction is to heavy, it will create artifacts in the signal. Characteristic artifacts include a “watery” sound or some hollowness in the speech.

A good first pass for this audio file “MO_rough” is maybe around -10 db Threshold and 4-5 db reduction. Play around with the settings though to hear their influence on the output signal. Click “Bypass” to hear the original signal. Flip it on and off to hear the comparison clearly. When you’re happy with the settings, click “Render” to process the audio.

Another strategy which is often good for a second pass is turn “Adaptive mode” off. Then with your cursor, select a section of the audio consisting only of unwanted noise, and click “Learn” in the top left. The automatically takes the footprint of the noise and sets the frequency-based threshold controls accordingly. Now you can mess with the faders again, find settings that sound good and process.

Generally, you’ll want to do three or four of these light passes. This is what our spectrogram looks like after four passes with slightly different settings. Notice that the noisy sections are far lighter than before we processed the audio. Meanwhile, the spoken words look relatively intact.

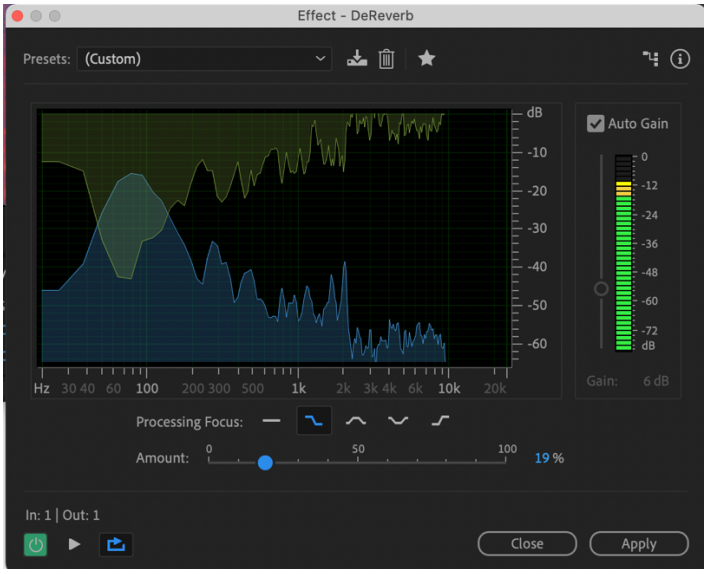


When you're happy with the results, "Save as" in a new directory called "denoised" to avoid overwriting your high-passed audio. Then repeat for the other audio file.

4. Remove reverb

One of the most distinctive characteristics of a room is its natural reverberations. Just like an oral cavity emphasizes certain harmonics to create formants, spaces amplify and dampen acoustic reflections at particular frequencies, creating a natural reverb. Hopefully, iZotope helped with this, but if audio files still have noticeable reverb, as is the case for "SJ_rough", we can open them up in Adobe Audition, which includes a de-reverb plugin.

From the toolbar, select "Effects" -> "Noise Reduction / Restoration" -> "DeReverb". This will show you a window like that shown below. You'll be working with the Processing Focus selector and the Amount slider. Processing focus controls where in the spectrum most of the DeReverb is applied. In this case, it sounds like a low, booming reverb, so it probably makes sense to select the low-end focus. Select different options and find the one that cuts the bad while leaving the good maximally intact. Too much DeReverb will make the audio sound unnatural, so be careful, and generally keep the "Amount" below 25-30%.

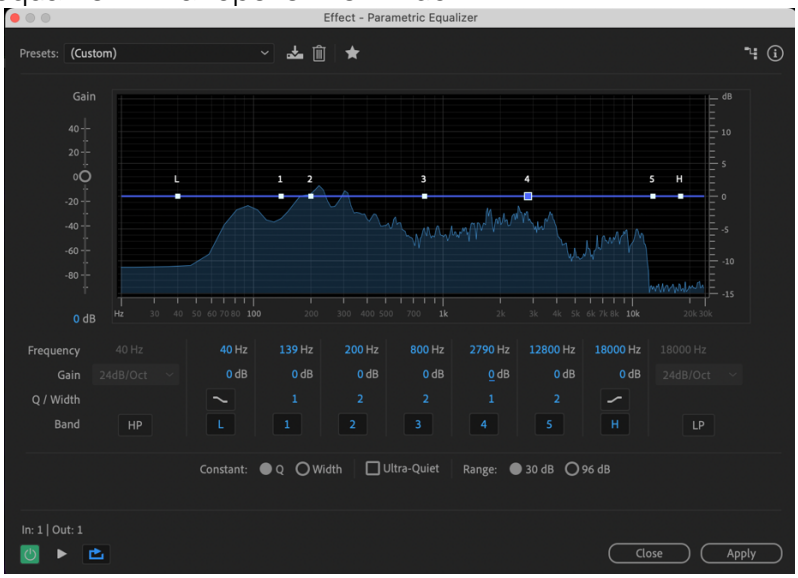


When you're happy, click "Apply" to process the original audio. From the main window, click "Save as", create a new folder called "dereverb" and save the file in there. Do the same for the other file, "MO_rough" if you feel it needs it.

5. EQ

Now we get to be a little subjective and use a graphical equalizer to try to make the audio files sound as similar as possible. Different microphones and rooms have different acoustic footprints, and the processing we've done so far has hollowed out the audio in unique ways. We can use an EQ to try to supplement missing frequency bands or tame overrepresented bands.

To open the EQ, go to the toolbar and click "Effects" -> "Filter & EQ" -> "Parametric equalizer". That opens this window:



You can use each node on the EQ to increase or decrease the presence of each band. The control for Q affects the width of the band, where lower Q results in a wider band. Use your ears to figure out which parts of the signal could benefit from modifications. Drag a band around to amplify different regions to see what sounds good and what sounds bad. As always, be subtle. Keep changes with about 6 db of the original (usually). Starting with SJ, it sounds like we could use a boost around f0, so add something like a 3.5 db boost at 100 Hz with a Q around 1. Sometimes doing a very small cut around the first formant can decrease muddiness, so I added a -1.7 db cut at 200 Hz with a Q around 2. There's considerable harshness in the high mids, so I did a larger cut of -4.5 db around 2800 Hz with a Q of 2. When it sounds good, click "Save as", create a new folder called EQ, and save.

Repeat the process for the file MO. Play around with settings to see what works. Listen to both files together and compare them. Try to think not only about what makes each file sound good, but also how you can make them sound more alike.

6. Normalize volume

As our penultimate step, we will normalize the overall intensity of the recordings. To do this, we can move back to Praat. First, create a new directory inside of your "EQ" folder called 'normalized', then open up the script "normalizeIntensity.praat" and run it. Like the low-pass script, it'll ask you for an input directory and an output directory. Give it the paths to your "EQ" folder and your new "Normalized" folder and then click OK.

By default, the script scales the audio to 65 dB SPL, which is generally a safe level. If you want a different scaled volume, you can just change the number 65 to something else in the script. However, beware that if you go too much louder than this, you will likely run into problems with clipping, which causes unpleasantness.

7. Add uniform noise

If we just spent so long sscrubbing noise out of our audio files, why would we add noise back in? All rooms, even sound attenuated recording booths, have a noise floor. Taking an otherwise-silent recording of a room and mixing it into each audio file can be a nice way to make recordings sound like they were recorded in the same space, even if they weren't. This is a very common trick used in radio and podcasting. Additionally, when you hear a recording with a digitally silent noise floor, it will likely sound kind of freaky. Some of our recordings at this point in the process may be subject to this kind of freakiness. Adding uniform noise can make them sound more natural.

To help with this, I've included a script called "renoise.praat" and a WAV file called "noise.wav" in the Praat scripts folder. Inside your "normalized" folder, make a new folder called "renoised", then run the script. You'll just need to provide paths to your input and output directories, along with the file path to the provided noise file. The script assumes

that the noise file is longer than your target audio file, and the noise file is about 3 minutes long. So, if your longest audio file is longer than 3 minutes, you'll need to extend the noise file. It's fine to do this by looping it. One easy way to achieve something like that is to open the file multiple times in Praat and "Combine" -> "Concatenate with overlap" with an overlap of maybe 50-200 ms. The overlap prevents the cracks and pops that arise when audio starts or stops abruptly. When you're sure your noise file is longer than the longest sound file, run the script. New files will be produced in the "renoised" directory.

These are your cleaned audio files! And you can now process them however you normally would! If anything doesn't sound quite right, try to work backward and see where it went astray. This is always an iterative process and even if you're careful, you'll wind up redoing things, which is okay! Remember that audio processed this way will never sound as good as audio recorded in a sound booth, but it will sound a whole lot better than raw recordings your participants made on their iphones in their kitchens.