**NOTE:** the test case files should produce the results below, if the filter configurations are also set as described to self verify, If you want to run generate\_test\_case.py standalone and verify, note that the noise, and spectra will vary slightly because of the the injected noise is random, and will produce different files.

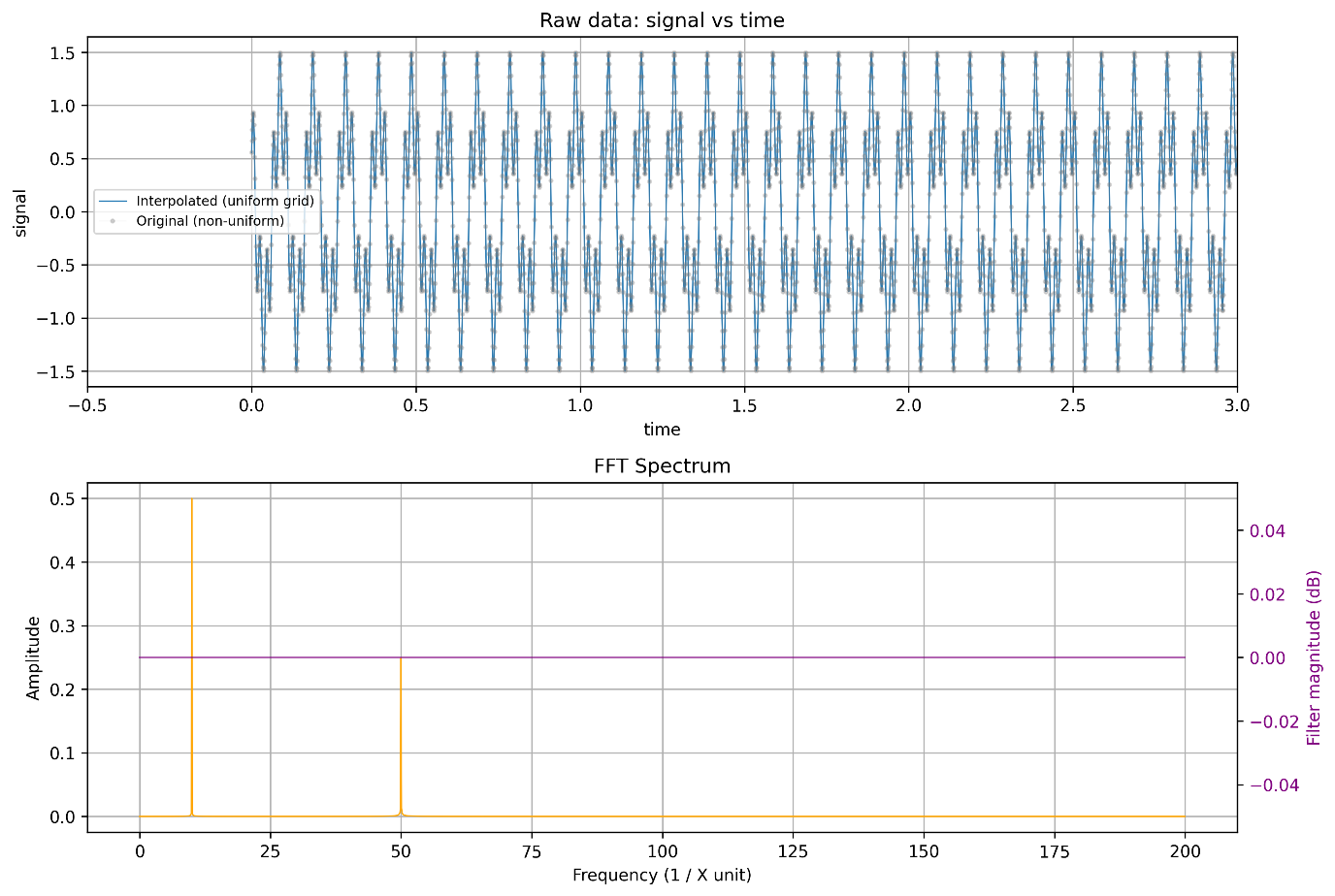


Image 1: Test case 1 wwith 10 Hz and 50 Hz. The x axis is truncated. This is quite clearly visible.

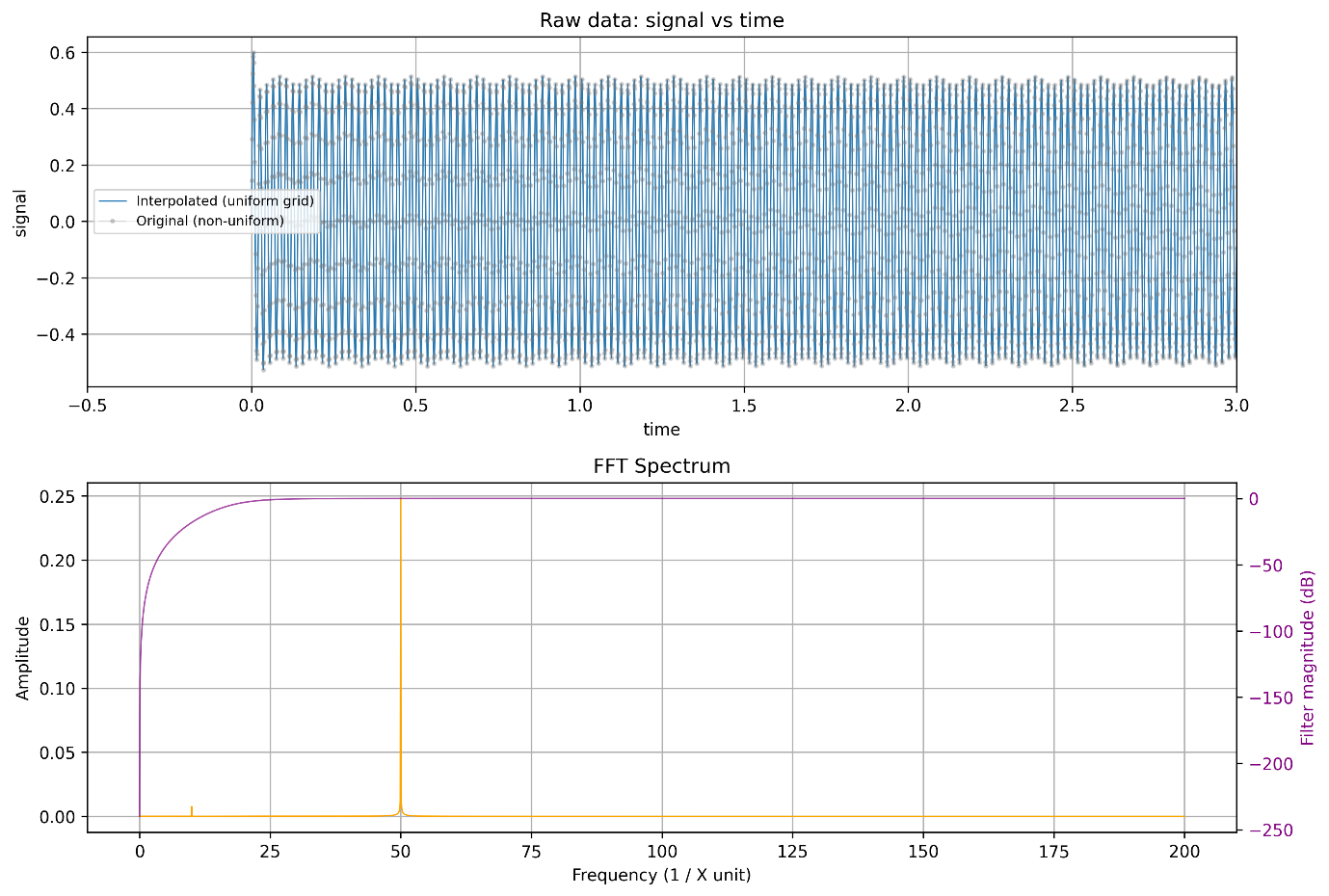


Image 2: simple test of the High pass cut off, this does indeed attenuate the 10 Hz!

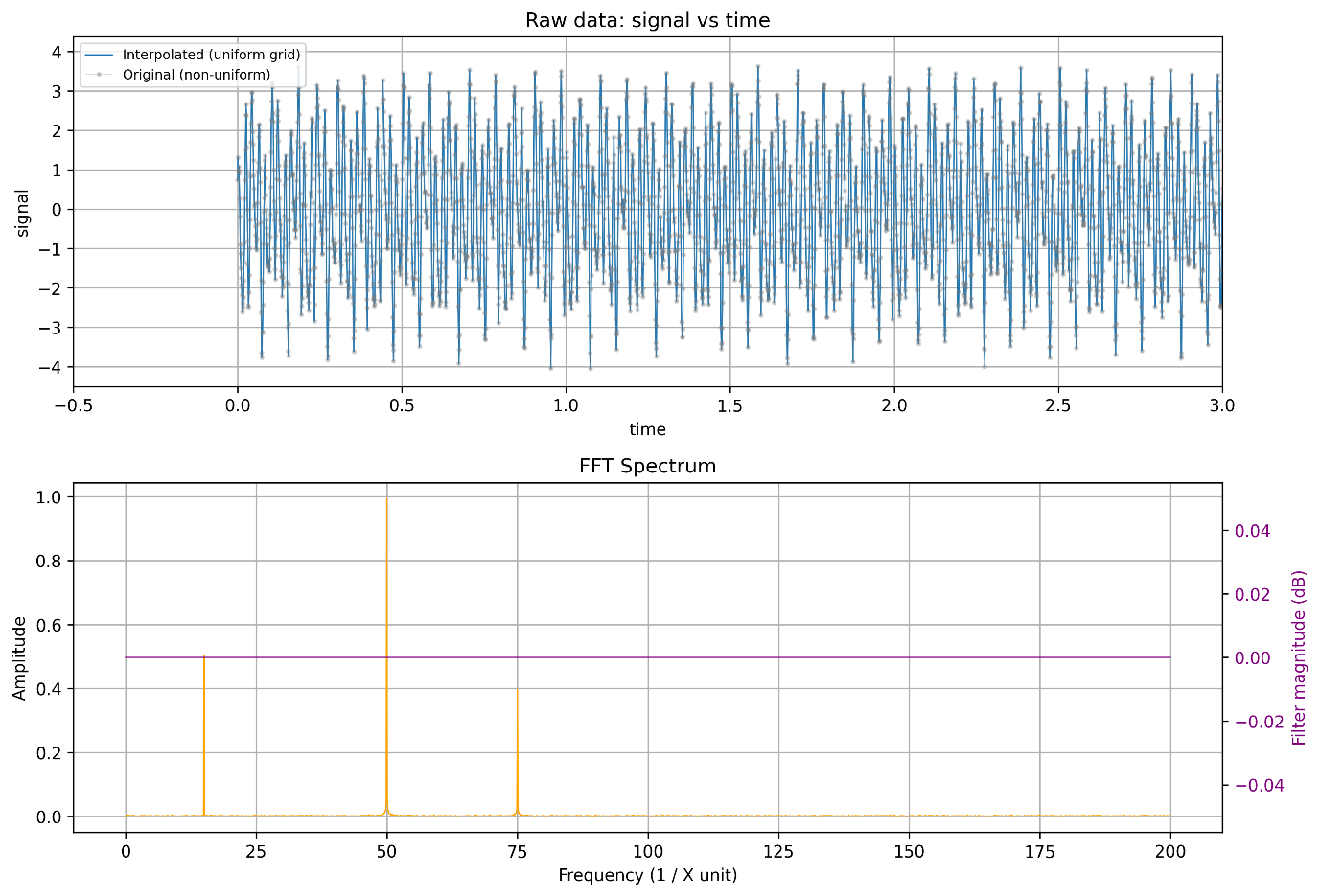


Image 3: signal of frequency content at 15 Hz, 75 Hz – with strong mains interference, which is also present in the signal.

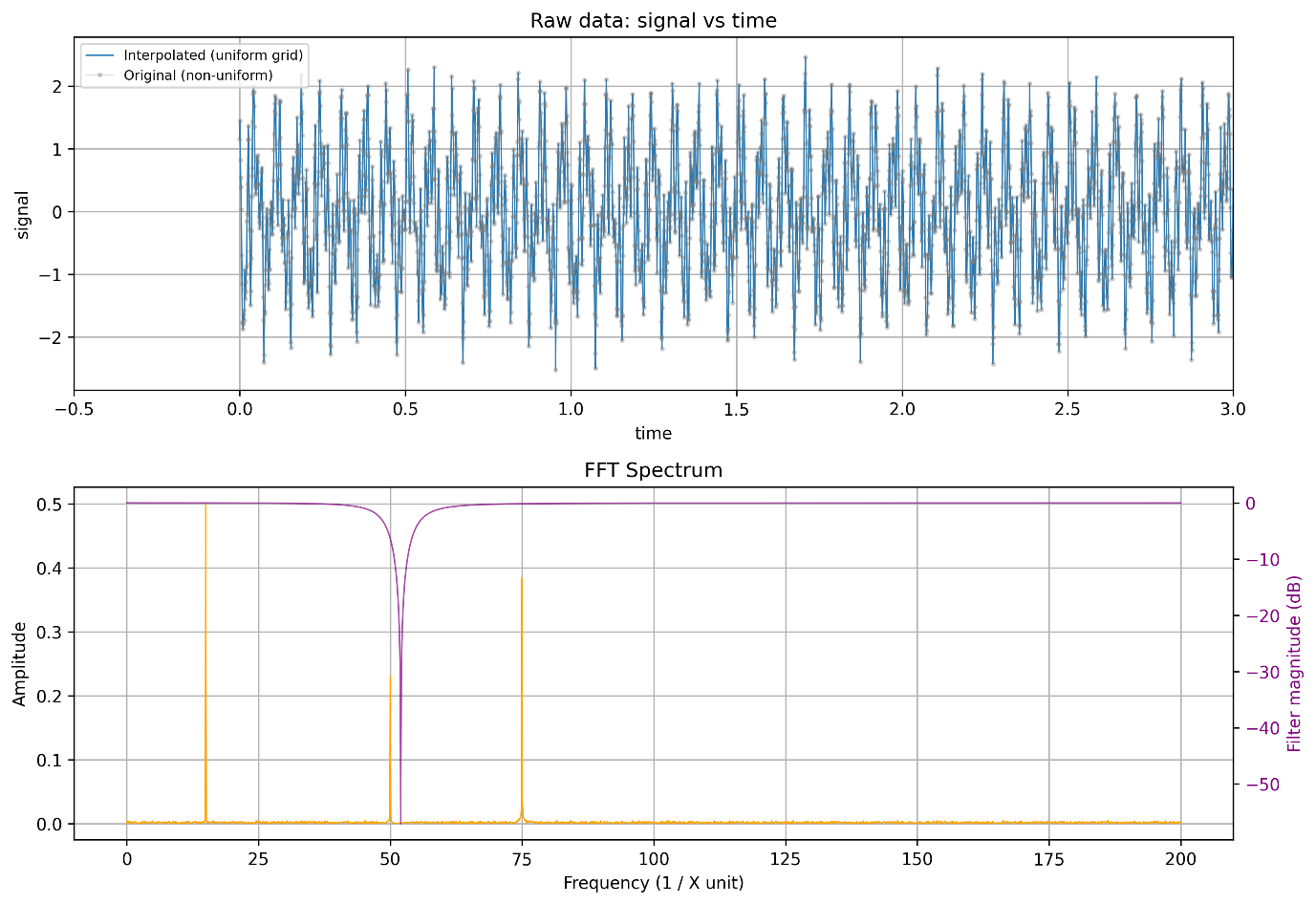


Image 4: The notch filter is applied at 52 Hz, with a Q-factor of 7. Because of the smearing of the band-stop, it attenuates some of the main noise but not all of it, and is still quite strong relative to the signal.

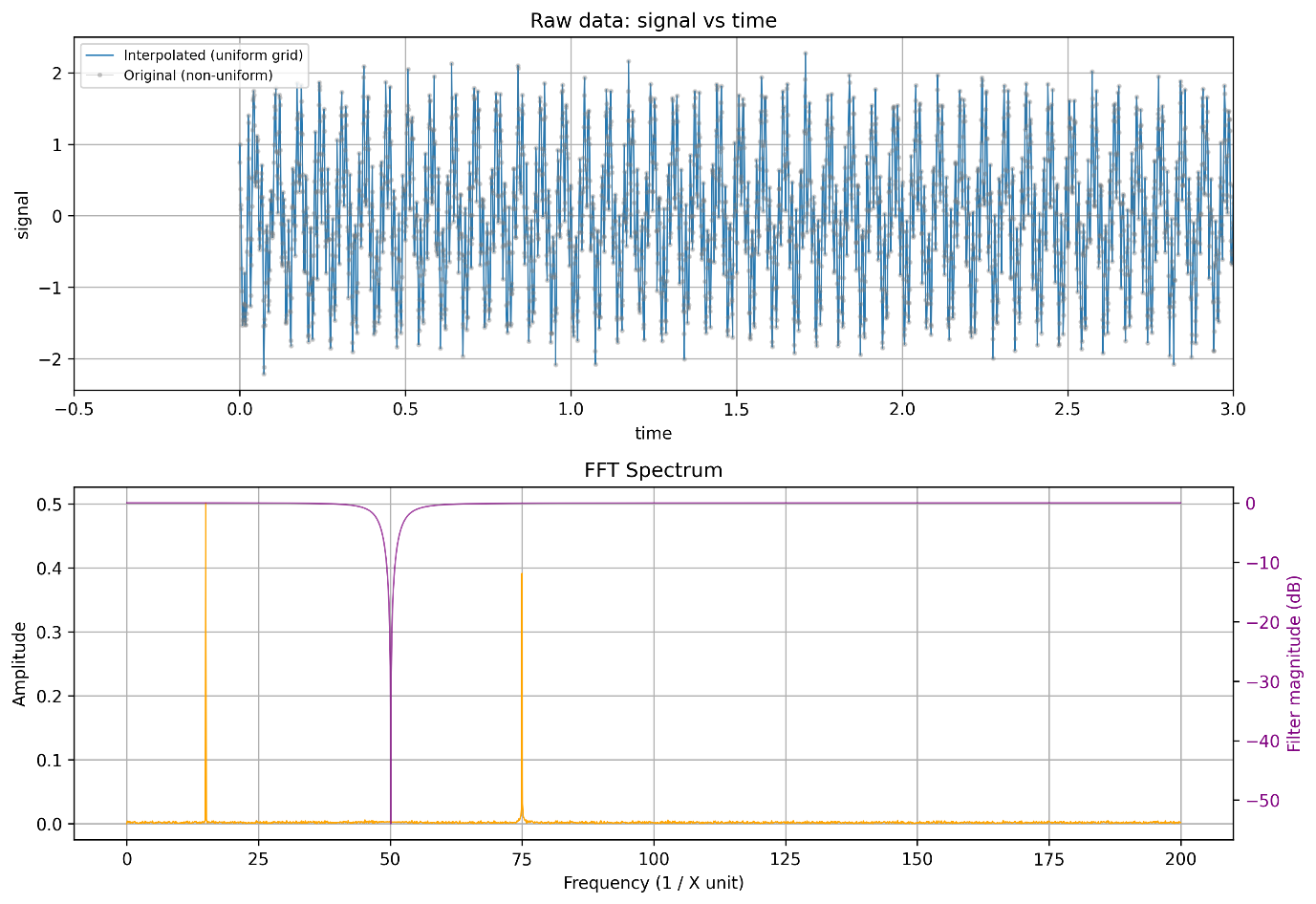


Image 5: Now we test the notch filter, but much closer to the rejection frequency at 50.1. The quality factor is still not perfect, a Q = 10. NOTE: this is simply to simulate a slightly imperfect notch filter, both in not having a perfect quality factor, and because of very real fluctuations in mains noise, losses in the filter itself etc…   
  
So counterintuitively, a lower quality factor, which reduces the sharpness of the band stop is actually better if the filter is not centred at the EXACT frequency we want to reject. However – it almost completely gets rid of the mains component, So pretty good eh?

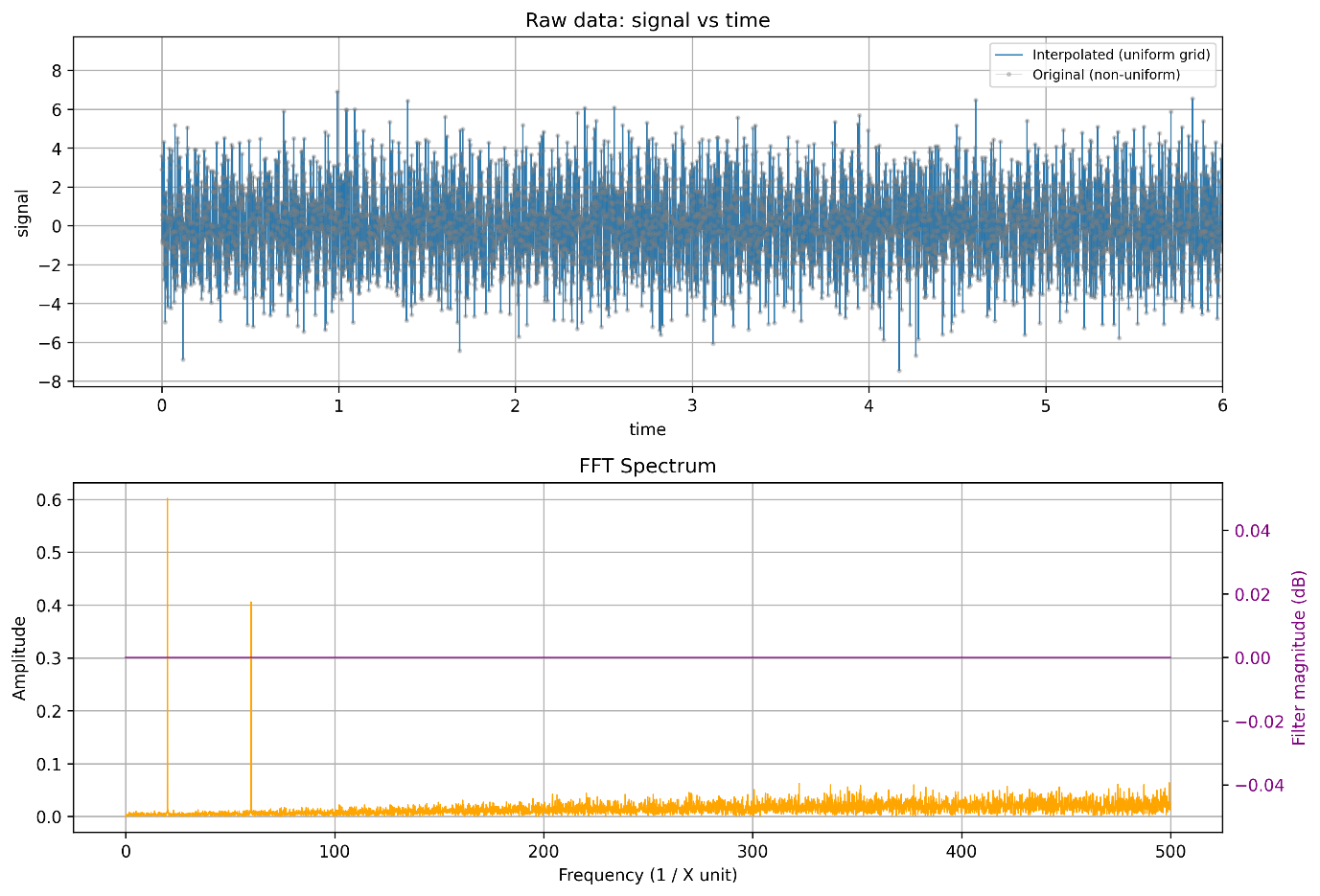


Image6: Test case 3: High frequency noise, which pollutes the signal. Even though the peaks are obvious, if there were lower amplitude signals of interest at higher freq. (towards the 0.5 kHz range) it might be clouded.

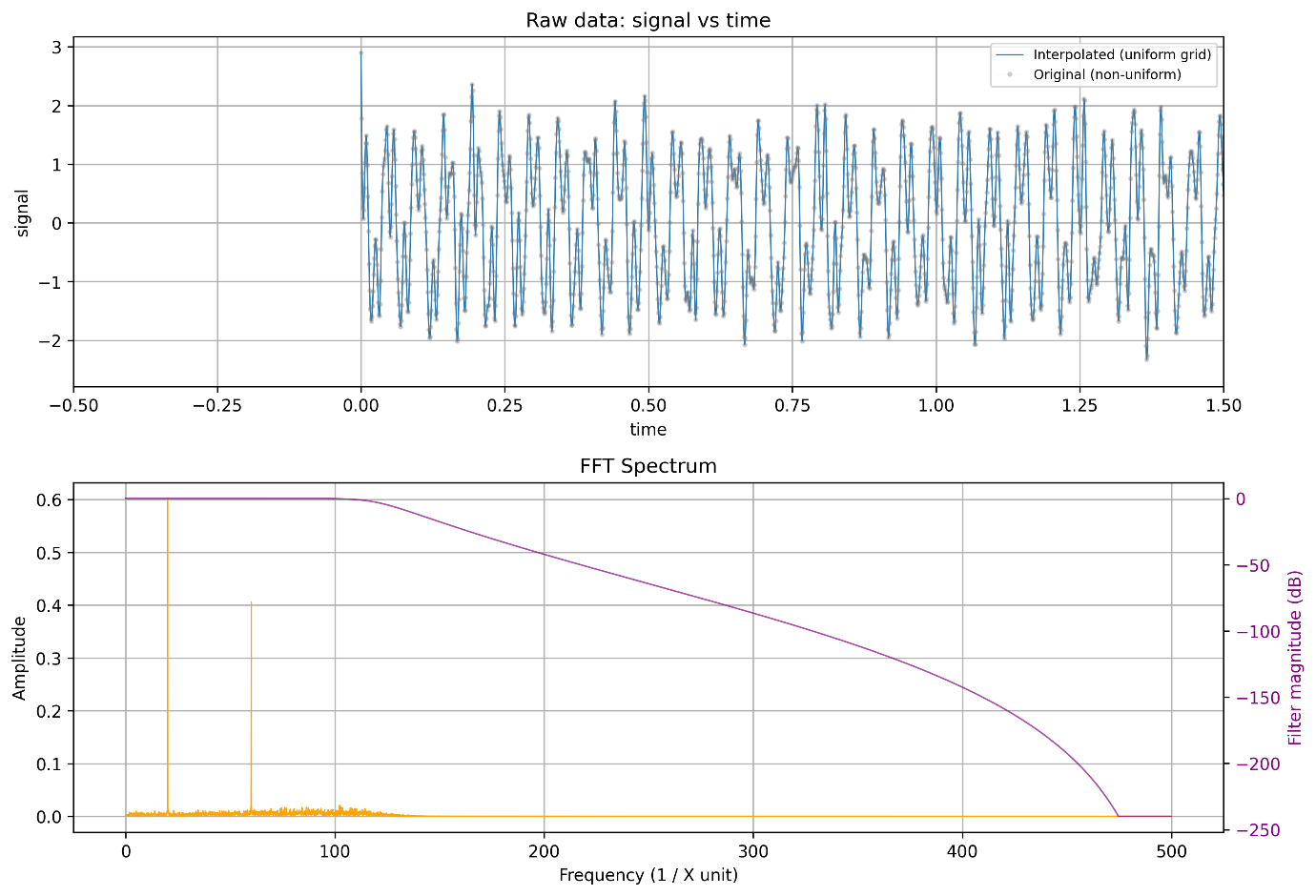


Image 7: Allows us to test the low pass filter – we have also zoomed into the data on the top graph on the X-axis. This indeed cleans up the signal significantly. It is an 8th order filter with the corner frequency at 120 Hz.  
  
A graph of a sound wave

AI-generated content may be incorrect.

Image 8: for test case 8, this is a purposefully polluted signal with peaks at 35 Hz and 90 Hz – with a fairly low SNR.

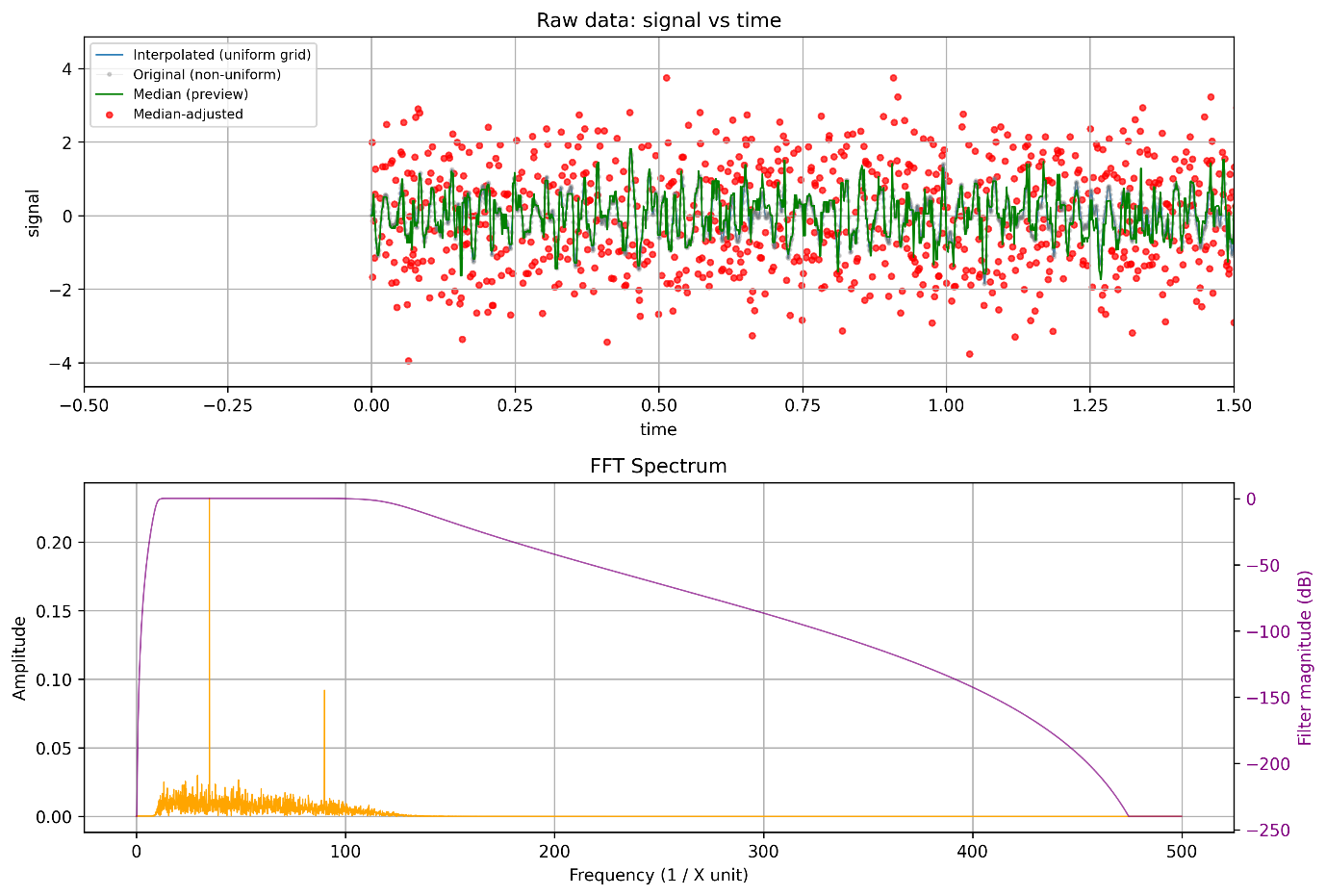


Image 9: cleaning it up is best with a somewhat bandpass. The low pass is set to 120 Hz, an the High pass to 10 Hz – both with an order 8 roll-off. This removes most of the high frequency, and very low frequency drift content. A median filter is also used in general to reject spikes around data points which ‘are’ actually meant to peak at the corresponding frequencies.

A graph with blue lines

AI-generated content may be incorrect.

Image 10, for test case 09: close frequency pairs. For and 95 and 96 pair, we can also text the effectiveness of a sharp notch filter. This data is to test the robustness of the FFT at distinguishing close frequencies.

**NOTE:**

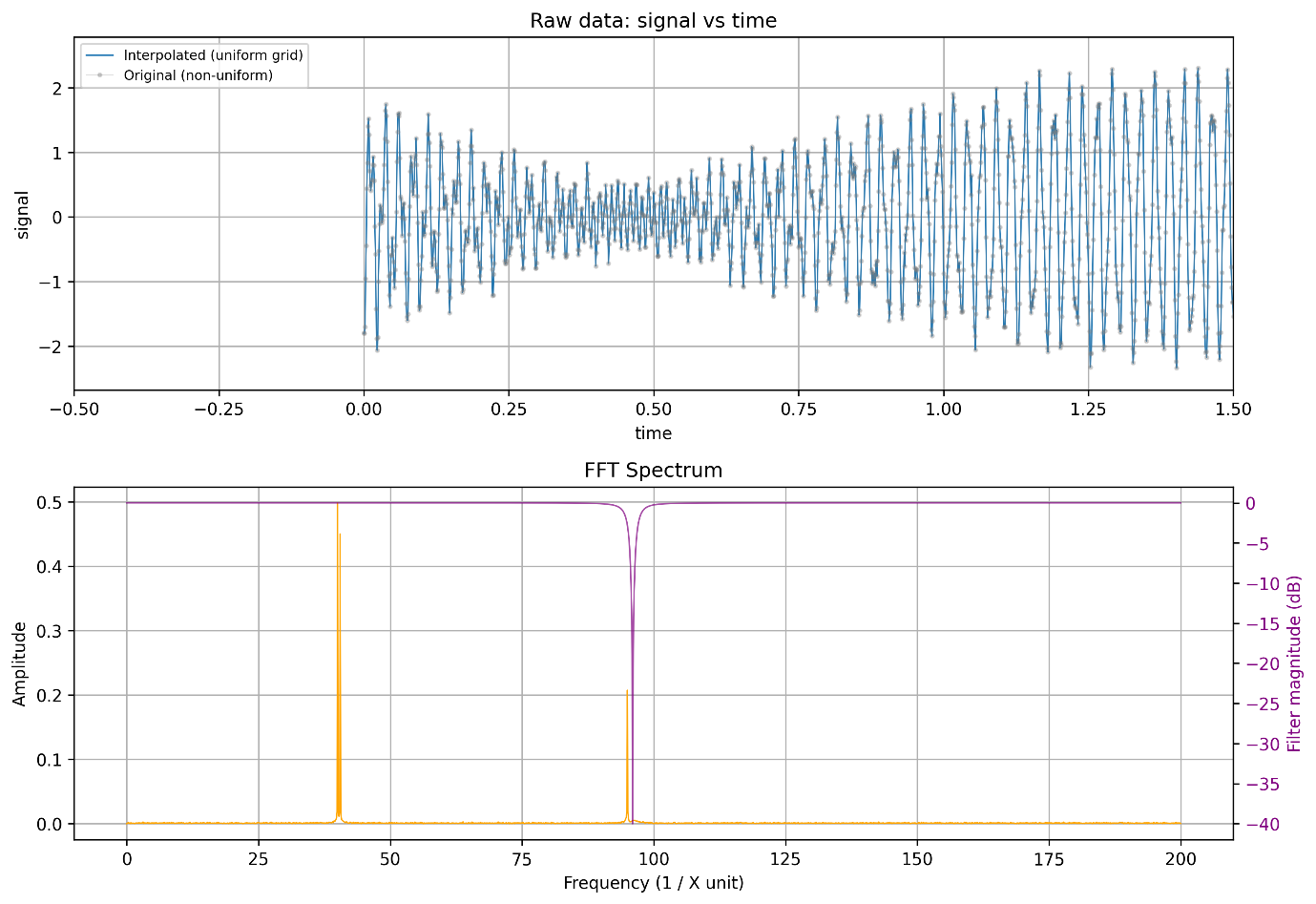


Image 11: the notch filter is applied, with a quality factor of 50 (huge!) centred at 96 Hz. Unfortunately it does indeed attenuate the nearby pair by 50 %.

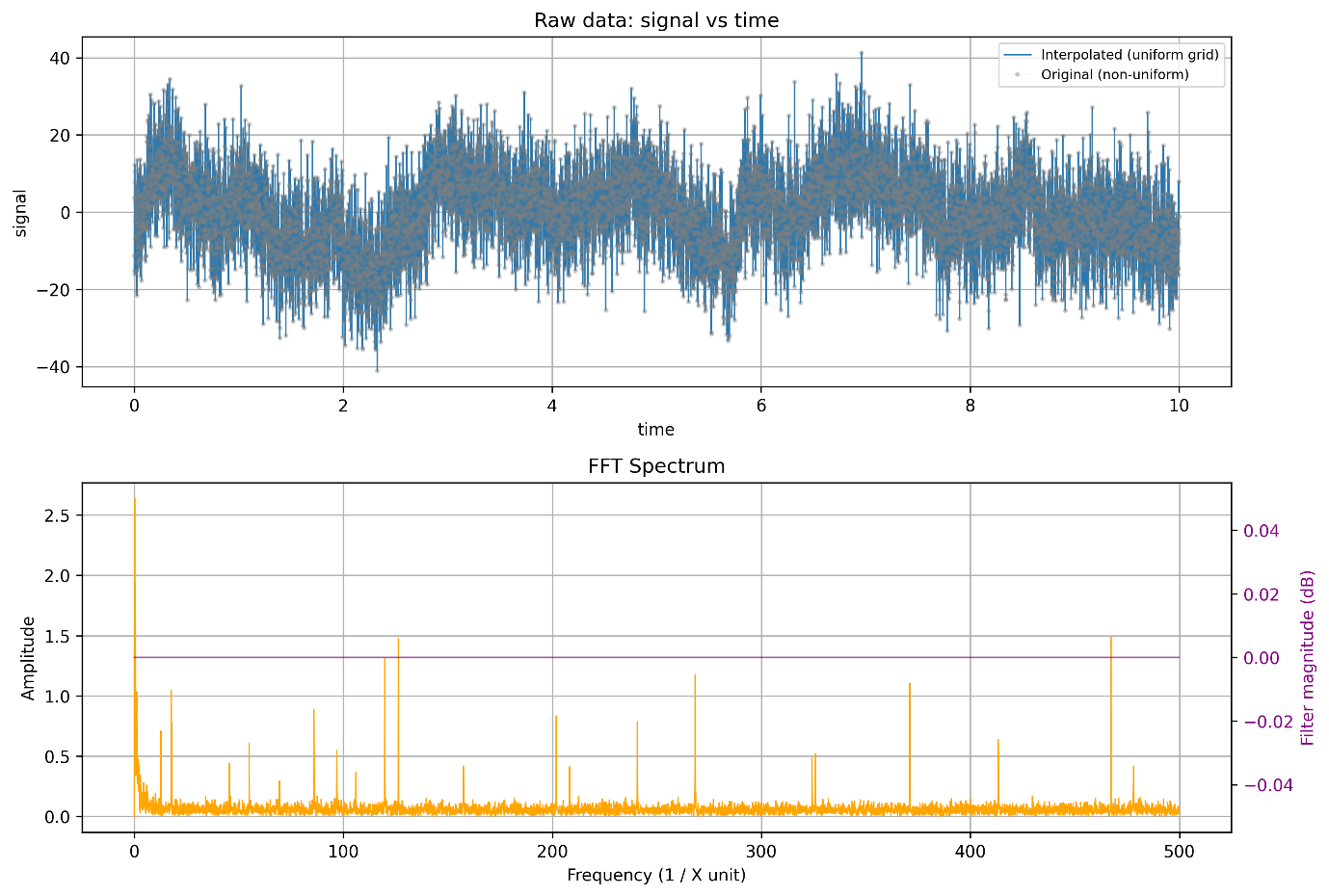


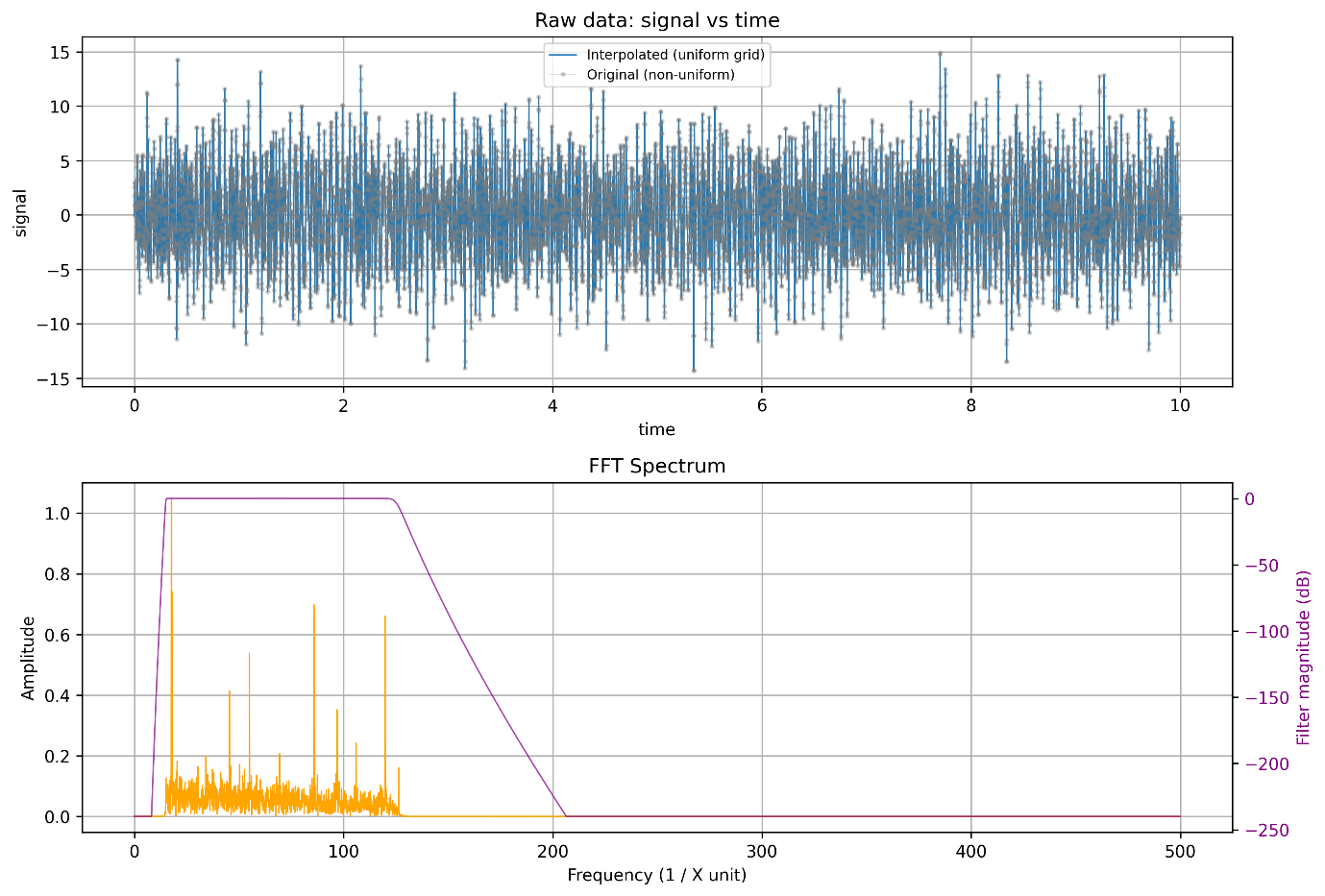
Image 12 – for Test case 12, with very high SNR. The frequencies of interest remain: 18.0, 55.0, 120.0. We can apply a bandpass filter, alongside a median.   
  


Image 13: a combination of different settings. Median filter with a kernel size of 5. Band pass of order 40, set to 15 and 125 Hz cutoffs respectively.

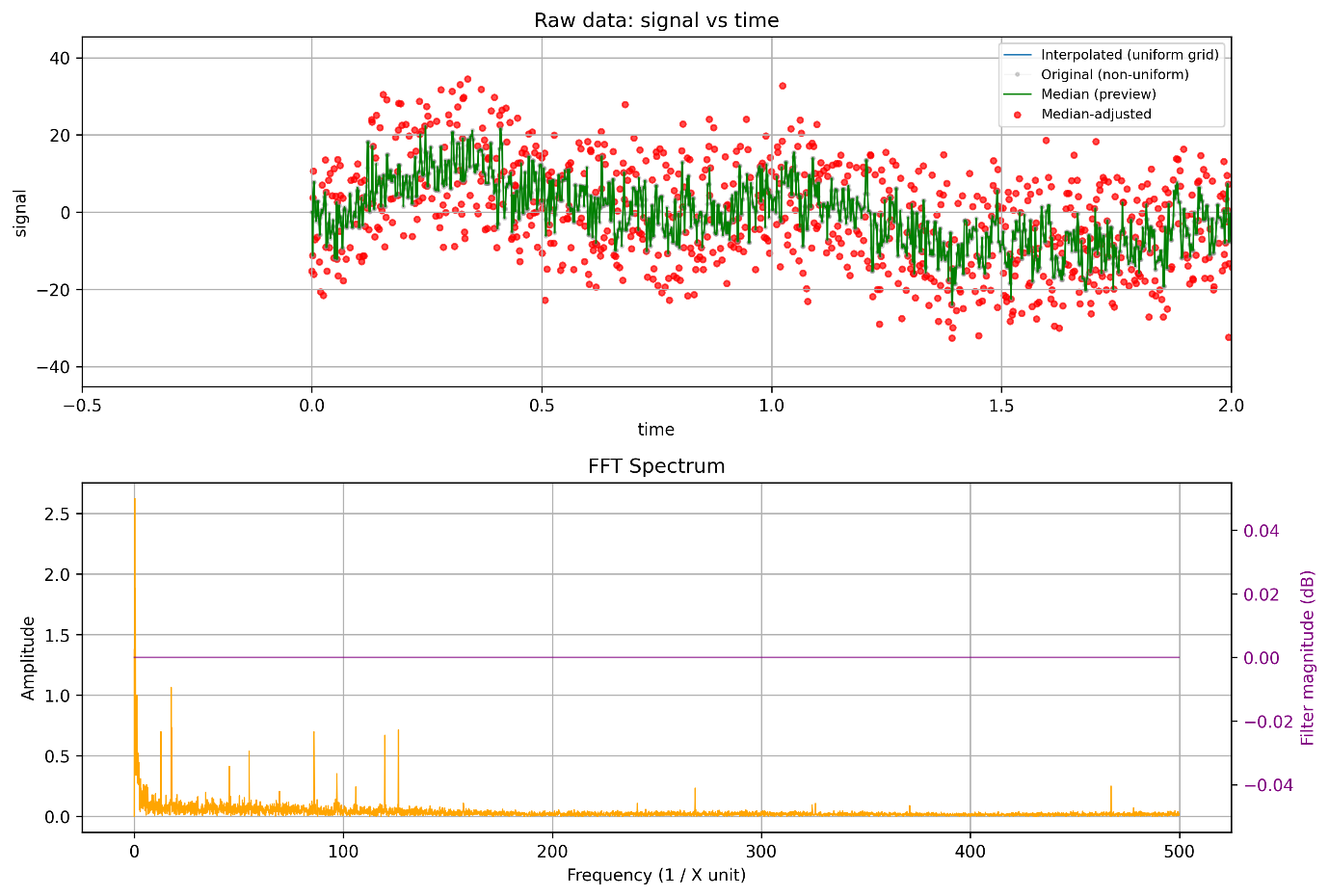
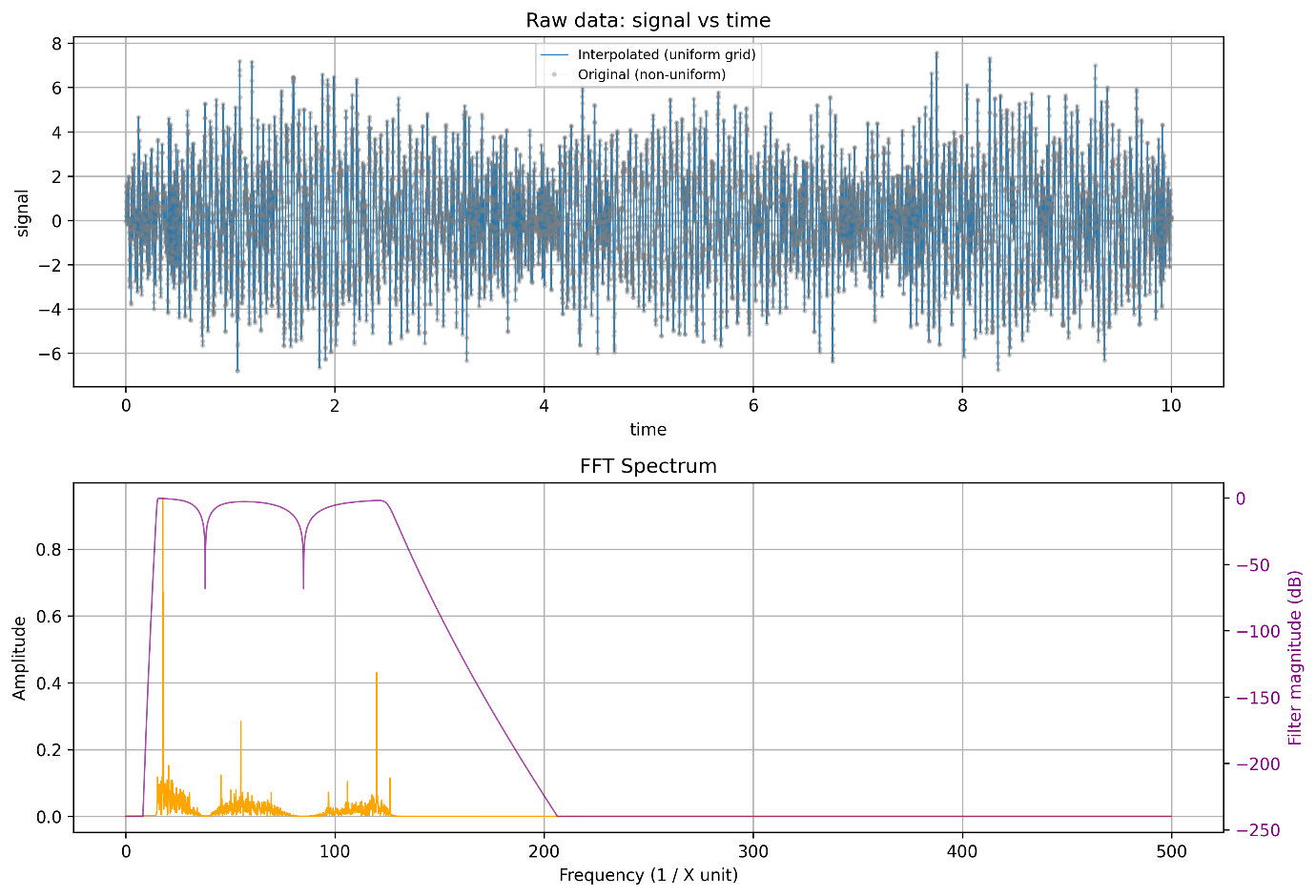


Image 15: Note, the median filter actually does a pretty good by itself at attenuating a lot of the high frequency peaks, by rejecting random noisy spikes

  
  
Image 14: finally, given at least, for this one we expect and know the frequencies we want, we can apply some notch filters with low quality. One at 38 Hz and another at 85 Hz, mid-way between pairs of our 3 frequencies. Because of the low Q factor, their attenuation around is relatively higher, allowing the 3 salient peaks to be seen more clearly!