

Calibration of bursts in LIGO E2 data

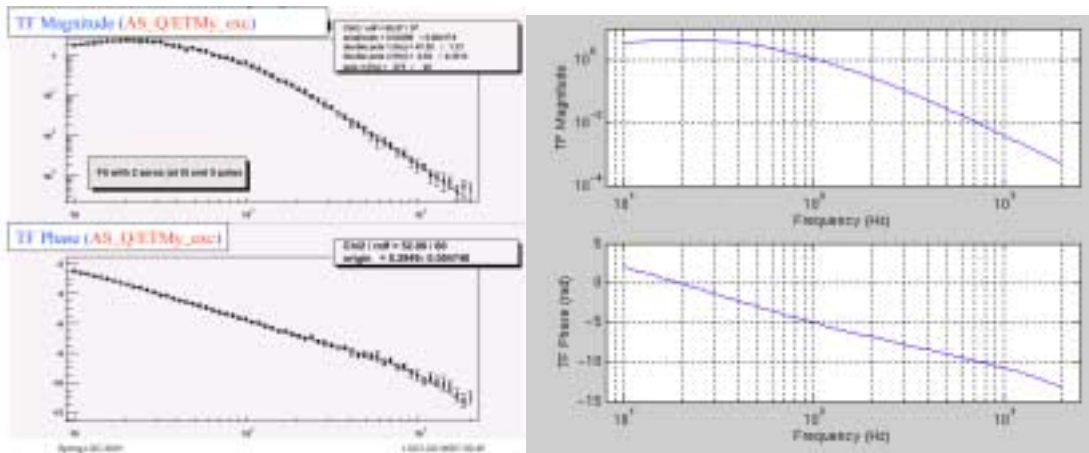
AJW, 4/25/01

The goal is to inject a burst waveform (ie, an $h(t)$) into the LIGO GW channel (ASC_Q) data stream, with the right transfer function and overall scale.

1) The LIGO H2K transfer function was measured during the E2 run by Landry et al; reported at the March 2001 LSC meeting, G010057.

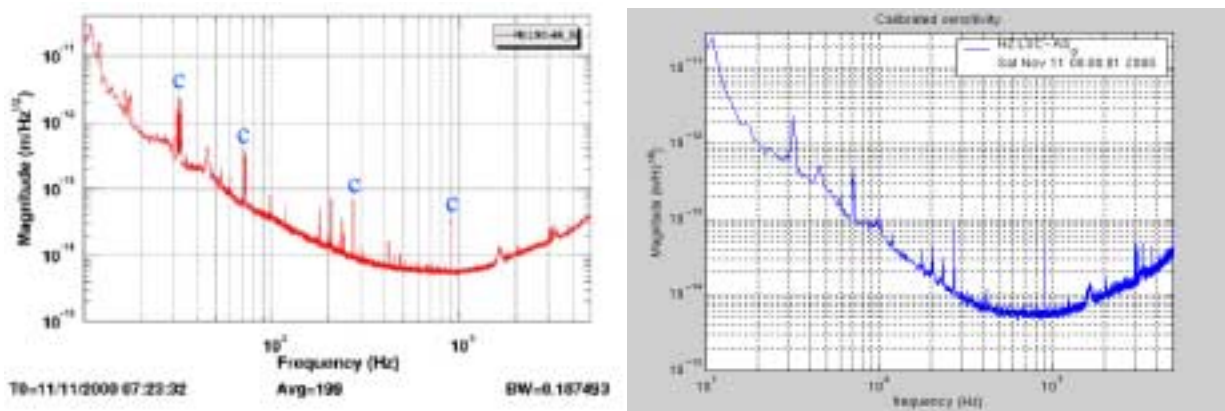
<http://www.ligo.caltech.edu/docs/G/G010057-00/G010057-00.pdf>

The transfer function was fit to a form with 2 poles and 5 zeros, as shown below.



Left: LIGO transfer function as measured by Landry et al; Right, my Matlab realization.

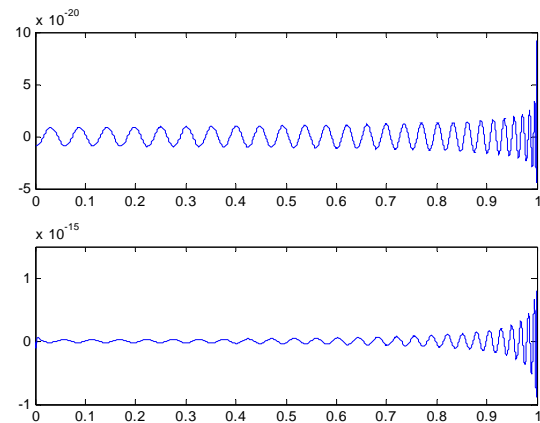
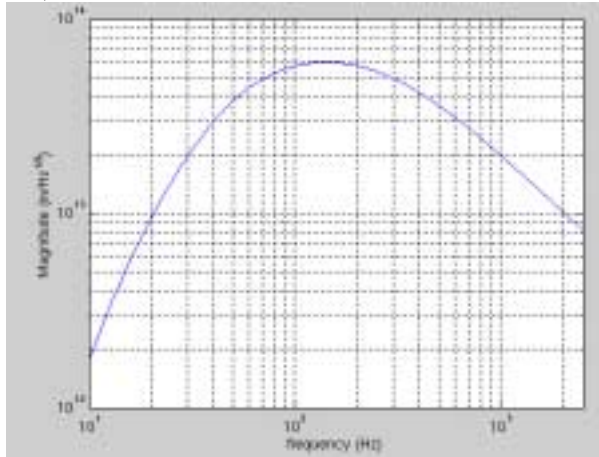
2) This transfer function has arbitrary units (ADCs per DAC). To put it into physical units, calibration lines are applied, giving known displacements to the mirrors. I can adjust my Matlab transfer function overall gain to reproduce the results of that procedure. I use 60 seconds of E2 data from the archives, calculate an average power spectrum, take the square root, divide by the above transfer function, and adjust the gain till I get agreement with the calibrated sensitivity of Landry et al.



Left: LIGO E2 sensitivity; Right: my Matlab realization.

3) This transfer function is from the suspension to the sensor; ie, it includes the pendulum transfer function. By contrast, a GW signal $h(t)$ will produce a displacement of the mirror Lh , with no pendulum response. Thus, the transfer function from mirror to sensor must be modified by dividing out the pendulum (two poles at $i*2*\pi*f_0$), or equivalently, adding two zeros to the above transfer function (totalling 4 zeros and 2 poles, resulting in a $1/f$ response at frequencies above the arm cavity pole, as expected).

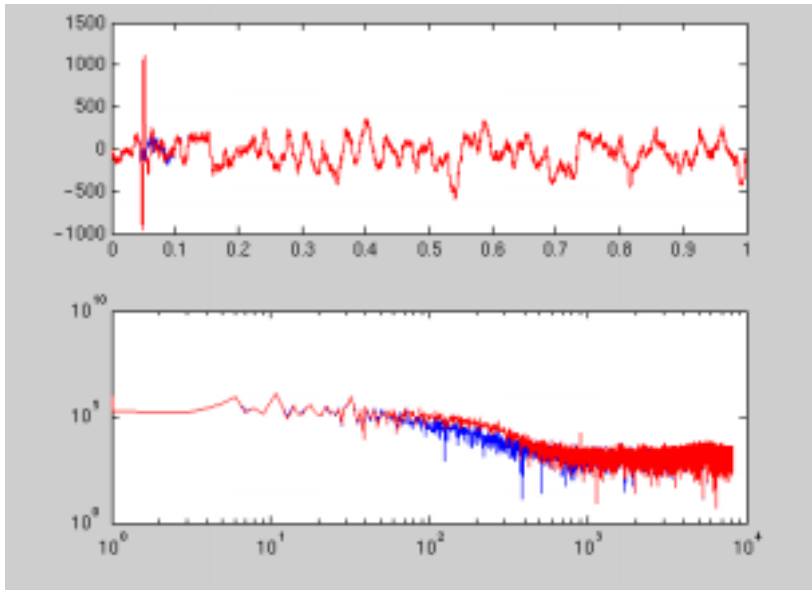
4) I can filter an $Lh(t)$ waveform (sampled at 16384 Hz) through a linear filter using that transfer function. It will suppress all frequencies relative to the peak response (100-200 Hz).



Left: Transfer function from $Lh(t)$ to ASC_Q . Right: Filtered chirp waveform.

5) Now I want to add a ZM-SN waveform to real E2 noise (ASC_Q). First, choose a SN distance that can give a measurable signal.

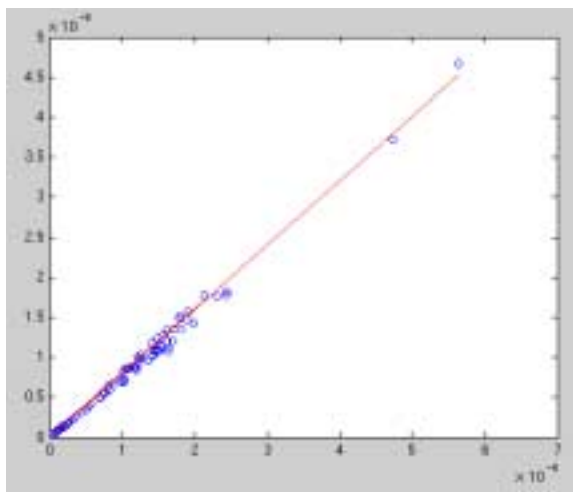
- Generate the $h(t)$, multiply by L , and calculate an x_{rms} for that signal. Check that you get the same x_{rms} in the time and frequency domains (Parseval's thm).
- Pass the signal through the LIGO E2 transfer function linear filter, get a waveform in ADC counts.
- Add this waveform to a 1-second stretch of noise: real E2 ASC_Q data (ok, using a full 1-sec of data is very non-optimal, since these waveforms are usually less than 0.1 sec long).
- FFT the resulting stretch of data, and compare it with the FFT of the noise data alone.
- Subtract the noise spectrum from the signal+noise spectrum. Divide by the LIGO E2 transfer function. What remains should be the spectrum of the signal, referred back to the x of the mirror. Calculate the x_{rms} . Compare with the original waveform x_{rms} .



Top: Red is the signal (one of 78 ZM-SN waveforms, filtered through the LIGO E2 transfer function) plus noise (1 second of E2 ASC_Q data); blue is noise alone. Bottom: same, for FFT spectrum.

In order to get a typical ZM-SN waveform to be visible (by eye) against the E2 noise, I had to put it at a distance such that the peak displacement was typically $3e-11$ meters; this corresponds to a ZM-SN at 0.01 parsec. Yes – in our solar neighborhood! Ouch!

Using 78 waveforms, and exactly the same 1-second stretch of E2 ASC_Q data, I was able to recover an x_{rms} (as described above) in good agreement with the x_{rms} of the original waveform, ... well, not exactly!



There was good correlation between the “generated” x_{rms} and the “reconstructed” x_{rms} , but the line in the above figure has a slope of 0.8, not 1! So I have a bug...