

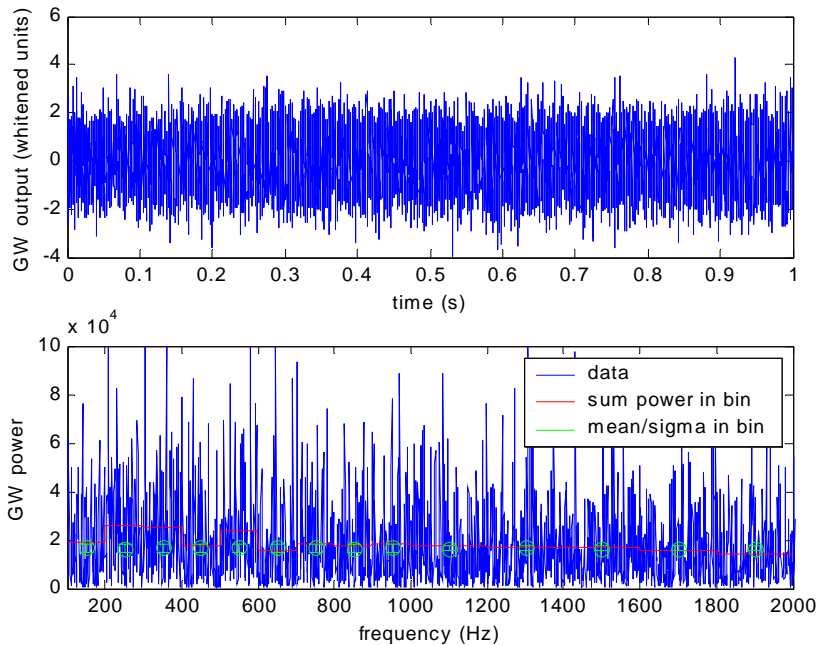
## Toy Monte Carlo taken through the whole analysis procedure

AJW, 4/22/01

In an effort to understand some of the more subtle issues associated with generating publishable results on burst rates, I went through the following "toy" Monte Carlo procedure. This was done entirely in Matlab, since LDAS is not quite ready...

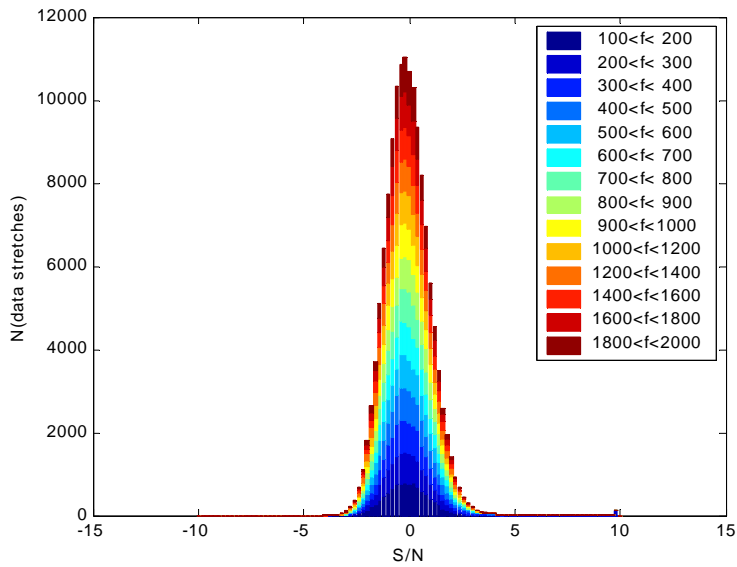
- 1) Generate one-second stretches of Gaussian white noise with mean = 0 and stdev = 1. This would correspond to properly whitened GW data from an ideal, in-lock IFO.
  - a. Question: what is the appropriate stretch of time to analyze the data? Here I'm using 1-second stretches, sampled at 16384 Hz, with no overlap. I gather that the binary inspiral group is planning to ingest 75-minute stretches, down-sampled to 2048 or less. Presumably they chop the data into smaller stretches, and overlap them by the duration of the longest chirp they're looking for.
  - b. Question: to properly whiten the data, we need a good, recent estimate of the noise power spectrum. This, I believe, should be accumulated continuously (eg, keep a running avg of the power spectrum of the last minute or 10 minutes of data). The binary inspiral guys are stuck with the noise spectrum accumulated in the previous 75-minute stretch. I think this is unacceptable (at least, for us), since the noise may never be stationary over 75-minute stretches.
- 2) Define a set of frequency bands over which power will be integrated (our "filter bank"). I used 15 bands, defined by lower edges (in Hz):  
fbin = [100 200 300 400 500 600 700 800 900 1000 1200 1400 1600 1800 2000];
- 3) Now start injecting chirp signals into the 1-second frames. I do this in every 20th frame; so that the injected event rate is 0.05 Hz or 5%. I inject chirps with chirp mass of  $2e-6$  (in geometrical units), yielding frequencies from 317 Hz (one second before merger) to 12084 Hz ( $1/16384$  seconds before merger). Each chirp has a fixed maximum amplitude  $h_{\max}(12084 \text{ Hz})$  of 2 noise power sigmas. This can be converted to an hrms in dimensionless strain units, assuming the noise power spectrum is known in those units (through IFO calibration). For now, I don't bother to do that - but it's a next step.
- 4) I choose a random polarization and altazimuth in the sky, and modulate  $h_{\max}$  by the LIGO antenna pattern.
- 5) The resultant signal is added to the Gaussian white noise.

- 6) The 1-second frame (with or without signal) is FFT'ed, and its square is the noise power spectrum. The noise power in each of the frequency bands defined above is calculated.
- 7) Use the first ~500 1-second stretches to establish the mean and stdev power per bandwidth in each of those bands. These 500 seconds of data are (assumed to be) free of anything but Gaussian noise.
- 8) The actual power in each of these bands, for each data stretch, is compared to the mean and stdev, and a S/N is calculated for each of the 15 filters. For ease of handling, I limit the S/N to  $< 10$ ; that is, if the calculated  $S/N > 9.9$ , I set it equal to 9.9. An example is shown below:



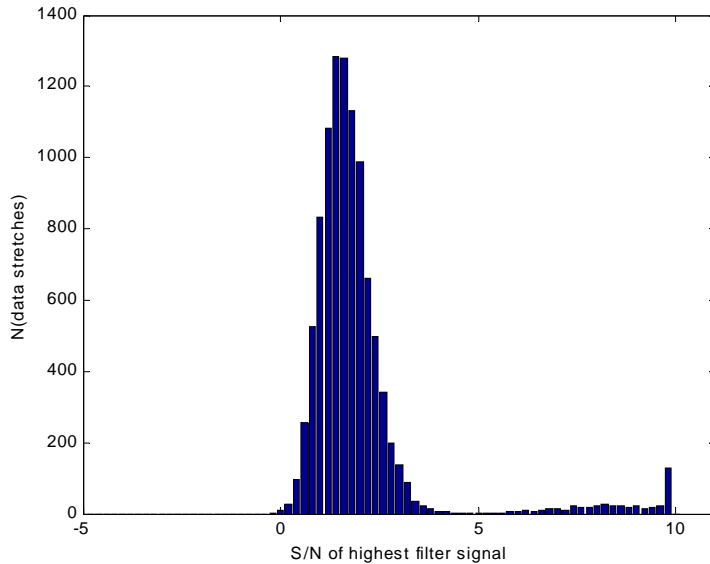
Top: time series of whitened Gaussian noise from the IFO, with (nominal) mean = 0 and stdev = 1. A chirp has been added to this time series, but it's hard to see.  
 Bottom: The FFT of the above time series, squared (noise power). The red histogram shows the summed noise power per bandwidth for each of the frequency bands defined in our bank of excess power filters. The green circles with error bars show the expected mean  $\langle \text{power} \rangle$  and stdev  $\sigma(\text{power})$ . The difference  $(\text{power} - \langle \text{power} \rangle) / \sigma(\text{power})$  is taken to be the S/N for that frequency band.

- 9) This process is repeated for 10000 1-second data stretches. The S/N for each stretch, for each of the 15 frequency bands, is shown in the following histogram:



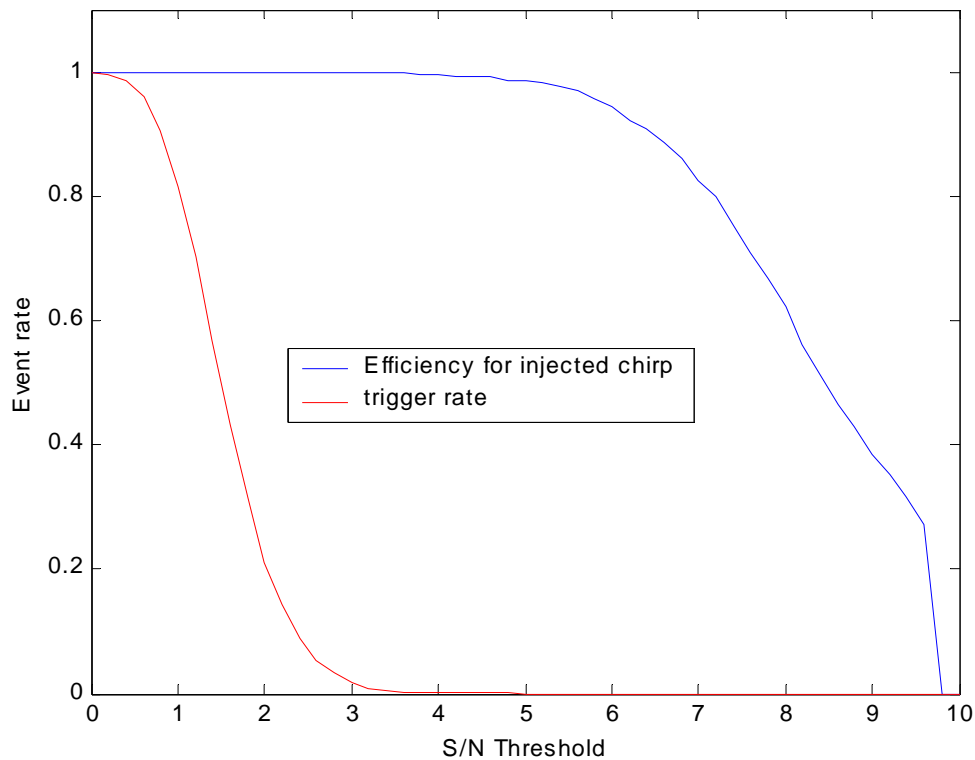
Histogram of S/N for each of 10000 1-second data stretches, for each of 15 frequency bands (excess power filters). Note the small peak at 10, which is actually a dribble of events extending all the way out; the peak at 10 is an artificial "overflow bin".

10) Now we select the highest S/N from each of the 15 frequency bands:



10) Now we "mock up" the database, by making a list of generated events. For each event, we record its time (for now, just the integer index of the 1-second stretch in which the chirp was injected; but ultimately, it would be the GPS time);  $h_{\max}$ ,  $f_{\min}$ , and  $f_{\max}$ . We also make a list of triggered events, ie, 1-second stretches where the S/N of the highest filter signal exceeds 3. For each triggered event, we

- record the time, S/N, and  $f_{\min}$  and  $f_{\max}$  of the filter frequency bands whose S/N exceeds 3.
- 11) Then we loop over generated events, and for each, search for a triggered event coincident in time. If one exists, record its S/N.
  - 12) From these lists, we can form (a) a generated event efficiency versus S/N threshold, and (b) a "fake rate" vs S/N threshold. To determine the fake rate, we removed time stretches that contained generated events.

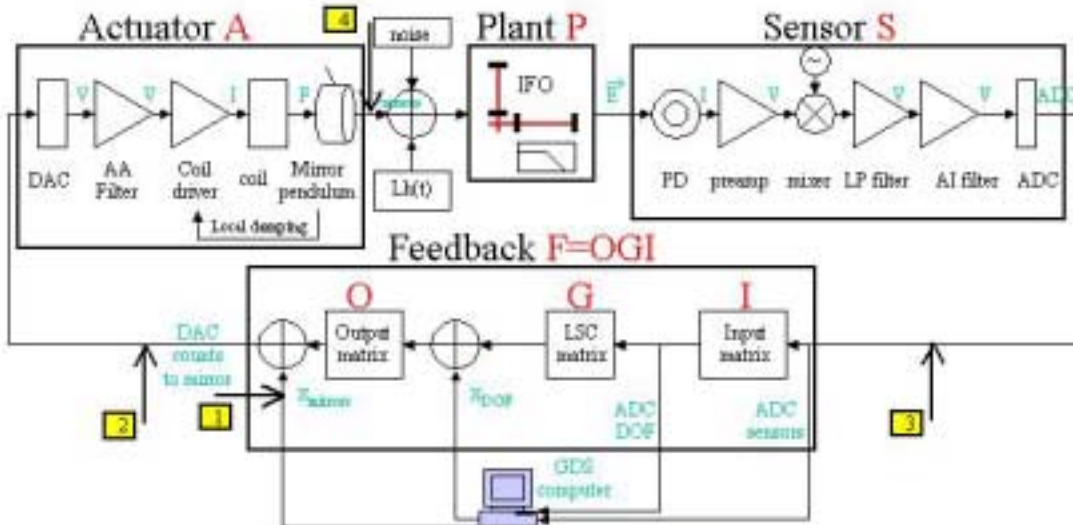


Next step is to mock up a "calibration", so that the x-axis will read " $h_{\text{rms}}$  in frequency band of LIGO detector" instead of S/N threshold.

## Calibration and MC simulation

AJW, 4/22/01

Here's an overview of the LIGO calibration system:



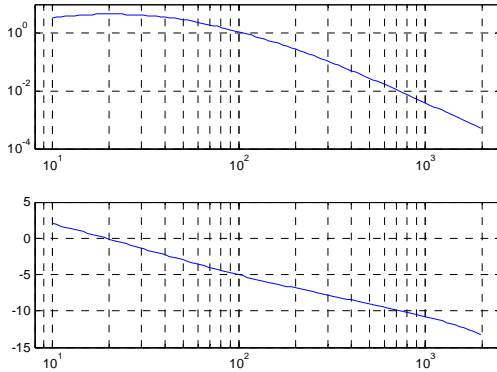
We measure: 
$$T(GDS_{sensor} \Rightarrow ADC_{DOF}) = \frac{ISPA}{1 - ISPAOG}$$

We want: 
$$T(Lh \Rightarrow ADC_{DOF}) = \frac{ISP}{1 - ISPAOG}$$

We need (?) 
$$A = A_0 \left( \frac{f_0}{f} \right)^2$$

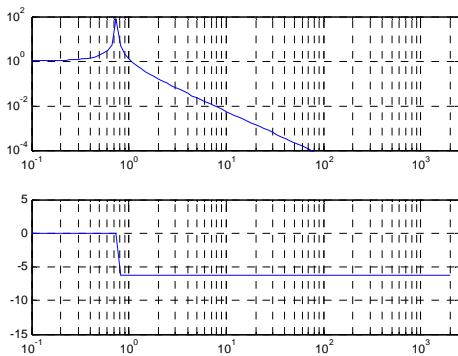
- The actuator is the OSEM. It's pretty high bandwidth, except for the suspension pendulum. Thus, it is assumed that  $A(f)$  is dominated by the pendulum double-pole,  $A(f) = -f_0^2 / (f^2 - f_0^2 - i f_0^2 \phi_0^2)$ , with  $f_0 = 0.744$  Hz, and  $\phi_0 \sim 1e-5$ .
- The sensor is presumably a constant, independent of  $f$ .
- The plant is presumably dominated by the cavity pole  $P(f) = -f_p^2 / (f^2 - f_p^2 - i f_p^2 \phi_p^2)$ , with  $f_p = 91$  Hz for 4K, 182 Hz for 2K.
- The Input matrix is a constant matrix converting from 4 sensors (eg, ASY\_Q) to degrees of freedom  $L_+, L_-, l_+, l_-$ . (Matt's code is more complicated during acquisition). It is set using measurements made as part of calibration.
- The Output matrix converts from DOFs to mirrors:  $[ 1 \ 1 ; -1 \ 1 ]/2$ .
- The LSC matrix is a bunch of complicated filters, whose net effect is obtained from transfer function measurements.
- The calibration procedure measures the transfer function from 1 to 3 (closed loop) or 2 to 3 (open-loop) from  $\sim 10$  Hz to  $\sim 3000$  Hz. The result from E2 (H2K) is given in G010057 (E2 Calibration, Landry et al) page 7.

- The fit to the transfer function detail is given in G010057 p 10 and is shown here:

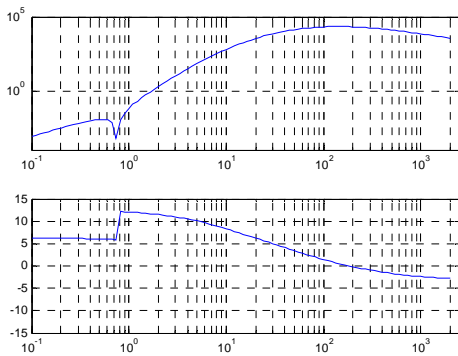


Fit to data from  $ETM_y$  transfer function from E2. Top: magnitude; Bot: phase.

- The overall magnitude of this transfer function is in more-or-less arbitrary units, and it can be normalized correctly using the information in G010057 p 13; to be done.
- We need the transfer function between 4 and 3 (the GW path) so we need to divide by the actuator (pendulum) transfer function  $A(f)$ .

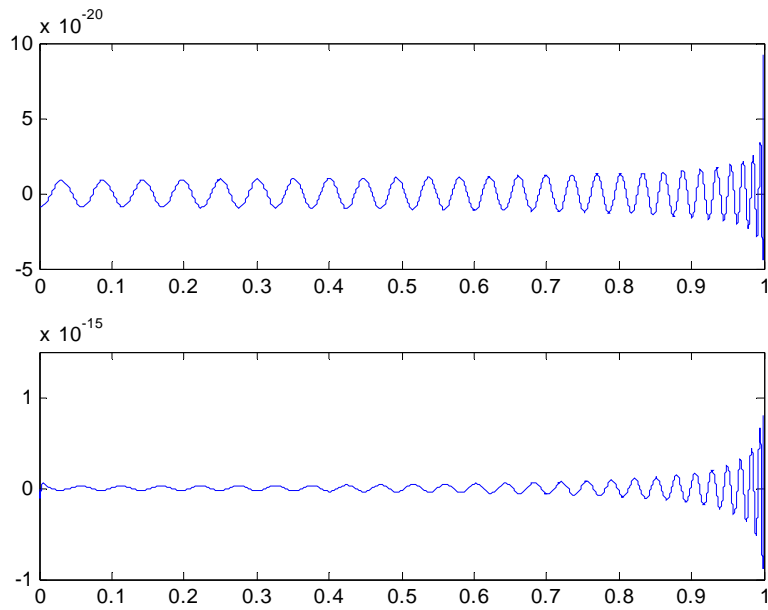


Pendulum transfer function



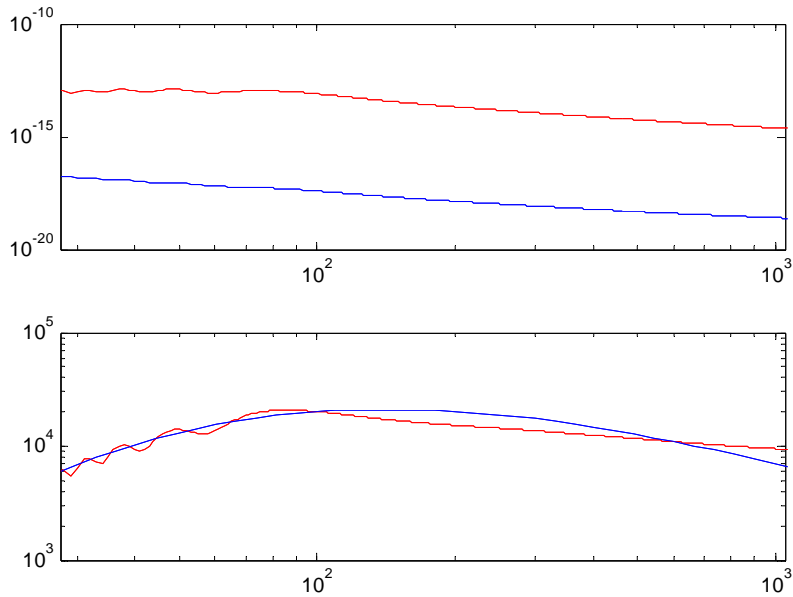
Transfer function from mirror position to ADC counts. Only the part from  $f > 10$  is meaningful, since it is not measured below that. And, the overall scale needs calibrating.

- So if we generate a chirp  $h(t)$ , we must filter it through the above transfer function before we add it to noise (in ADC counts) from real engineering data. This is illustrated below:



Top: A chirp. Bottom: a chirp, filtered through the LIGO E2 transfer function. Again, overall scale must be calibrated.

- We can check this by FFTing both signals, taking their ratio, and comparing it with the parameterized transfer function of LIGO, as shown below.



Top: FFT of original chirp (blue) and filtered chirp (red). Bottom: filtered/original (red) and LIGO transfer function (blue).