LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Report **LIGO-T040069-01-E** 2004/04/19

Software Review for the GRB030329 Analysis

Teviet Creighton and Peter Shawhan

Distribution of this draft:

LIGO Scientific Collaboration

California Institute of Technology
LIGO Project - MS 18-34
Pasadena CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project - MS NW17-161
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

WWW: http://www.ligo.caltech.edu/

Processed with LaTeX on 2004/04/19

1 Introduction

In November 2003, Alan Wiseman asked us to undertake a review of the software used to search for a gravitational-wave signature near the time of the "monster" gamma-ray burst, GRB030329, which occurred during the S2 science run. Our charge from Alan was as follows:

> Dear Teviet and Peter,
>
> Szabi Marka would like to present some preliminary results of his external trigger (grb) burst search at the GWDAW. Would you guys be willing to review the code Szabi uses to generate his results.
>
> Pretty Please!!!
>
> Although the results have been discussed a few times, the code has received little public exposure in the collaboration, and therefore Peter Saulson, Stan and I think the code should have some fresh eyes look at it. We have no reason to believe the code has any problems, and we realize this is more intrusive than our previous software reviews, but before the results go public we feel that we should take an extra look.
>
> I haven't even seen the code, so it is hard to give guidance on what form the review should take, so I will leave that up to you guys to figure out. The standard should be that you are solidly convinced that the code and the results are correct. Your final report can be brief, but it should spell out what you did to arrive at your conclusion (eg "I checked it against some code I wrote", or "I downloaded the code and ran it on some signals I generated", or "I had Szabi run the following N tests while watched over his shoulder.", or ... you get the idea) Basically, the purpose of your report is to reassure the rest of the collaboration that the results are sound. This is best done with a written explanation of what tests were done.
>
> Unfortunately, GWDAW is fast approaching, and therefore this needs to be done in the next three weeks. If there is any issue about setting priorities for your time, I would be glad to discuss it with your supervisor(s). It is a cool topic of study, so please help out.
>
> Alan Wiseman, LSC Software Coordinator

Part of the motivation for this special code review was the fact that the analysis was implemented entirely in MATLAB, which at that time was deemed an "unapproved" analysis environment. MATLAB has subsequently been accepted as an environment for LSC data analysis, as long as the analysis meets certain standards for availability, version control, documentation, and validation/review.

We carried out what we felt was a pretty thorough review before GWDAW, and gave some feedback to Szabi on making the analysis more solid (as described below), which he acted upon. However, we never got around to writing the final report... until now.

2 Algorithm review

In order to check the correctness of the software, we first had to understand how the algorithm was intended to work. Szabi spent a few hours with each of us (independently) describing the analysis algorithm, starting from his presentation at the November LSC Meeting (http://www.ligo.caltech.edu/docs/ScienceDocs/G/G030613-00/) and then going into much more detail. We felt that the analysis as formulated was a reasonable one. (Peter, writing this, admits that he does not feel as much on top of all the gory details as he did a few months ago, but will do his best to summarize the analysis from his memory and notes.)

Basically, the analysis consists of looking for a brief correlation between the data streams of the H1 and H2 detectors in a time period of a few minutes around the time of the GRB trigger.

(The L1 detector was not operating at the time.) The integration length, over which the correlation is calculated, varies from 2 ms to ∼120 ms (with a variable step size). The correlation integral is done for start times which span the few-minute time interval of interest, in steps of 4 ms in the code as it was made available to us. A two-dimensional pixel array, with start time in the X direction and integration length in the Y direction, is calculated, and "clusters" (at least 3×3 pixels) in this array with the highest power are found iteratively. Because there may be instrumental correlations between the H1 and H2 data streams, what is searched for is a cluster in the time interval around the GRB which has an unusually high power, using data from *other* (nearby) time intervals to get the baseline distribution of cluster powers.

The efficiency of the search algorithm is evaluated using software injections of simulated burst signals—sine-Gaussians, Zwerger-Müller waveforms, etc.—with various amplitudes. These are added to the data before any processing, and then the entire analysis is performed as if it were real data. The key issue here are whether the simulated signals are added to the data correctly and in a way which is indistinguishable from a real signal in the data. One specific issue is whether the calibration information is used correctly when doing the injections. Isabel Leonor and Rauha Rahkola did a check on whether the calibration calibration done in the MATLAB code is the same as what would be obtained with the standard LDAS calibration method. Their study is described at http://www.uoregon.edu/∼ileonor/ligo/s2/inject/codereview/comparecal.html . They found exact agreement for H1 and a very small discrepancy for H2, but Peter determined that this was due to an error in specifying which reference calibration to use, from between the two H2 reference calibrations which differ by an overall 3% scale. (However, it does not look like the web page was ever revised to reflect this information.) We also verified that the up-to-date (version 3) calibration files were used for this analysis.

There are many further details of the analysis which we will not go into here, but which we became familiar with at the time we were doing the review.

3 Scrutiny of the software

With an understanding of the *intended* operation of the search algorithm, our task was to verify that the implementation was correct, *i.e.* that the code did what it was supposed to do. Szabi set up a directory structure for each of us on mirfak.ligo.caltech.edu with all of the MATLAB code used by the analysis and with sample input data files. The code consisted of 22 MATLAB function files (`*.m`) with a total of 2659 lines, of which 1226 were executable MATLAB statements. Of these 22 files, five were unmodified MATLAB toolbox functions, included in this code base only for technical reasons, and four others were Shourov Chatterji's linear predictive error filter code, which was used to whiten the data in a preprocessing step.

One of the requests made during the review process, by various people, was that all of the code should be put into a standard CVS archive. This will be done as soon as the new centralized CVS archive is set up at Caltech.

We both spent many hours (independently) working through every line of the code, and executing various parts of it to check their behavior. The analysis was organized six stages, with intermediate MATLAB data files and a driver function which allowed the stages to be executed all together or one at a time. This organization, plus the fact that the high level of the MATLAB language allowed the analysis to be implemented using a fairly small number of lines of code, made

it fairly straightforward to do an exhaustive check of all parts of the pipeline. In many cases, we printed the values of intermediate result variables or plotted array contents as we traced through the code.

4 Findings and feedback

We found no bugs which had any significant impact on the results. There did seem to be a single-time-sample (0.24 ms) offset in the calculated times of clusters, but that was of no consequence. There were a number of "latent bugs" or "gotchas" which behaved properly for this particular analysis but would cause problems if the analysis were run with different parameters or differently-constructed simulated waveforms. These were conveyed to Szabi.

Peter did manage to uncover a substantive problem with the efficiency calculation for short-duration waveforms that was due to a configuration oversight rather than a bug. He noticed that the time step (for integration start times) being used in the analysis, 4 ms, was rather long compared to the duration of high-frequency, low-Q sine-Gaussians. An early draft of the GWDAW talk included a table of efficiencies which went up to 1.8 kHz for Q's as low as 1. Such short signals could fall between time steps, suffering an efficiency loss. This would be more-or-less harmless if the simulation code added signals at truly random times—so that they would sample the same efficiency loss—but it happened to add them at *fixed* intervals 0.5 seconds apart, always with the same relationship to the 4-ms time binning. This slight difference between the treatment of simulated signals *vs.* real signals was enough to bias the calculated efficiency for sufficiently short signals. This was communicated to Szabi, who confirmed that there was an effect; the sensitivity could be off by as much as 30% for the very shortest waveforms. (Note that this did not have any effect on the astrophysical "bottom line" of the GWDAW presentation, a limit on the energy emitted in gravitational waves from GRB030329, since that was calculated for a sine-Gaussian at 250 Hz with $Q = 8.9$. All sine-Gaussians in the presentation had $Q = 4.5$ or higher, although efficiencies were presented for some rather short Dimmelmeier-Font-Müller core collapse waveforms that might have been affected at some level.) To fix the analysis for short waveforms, Szabi reduced the size of the time step to 1 ms, to minimize the effect of its granularity, and modified the simulation code to introduce some randomness into the injected times, so that the simulation more faithfully represents real signals which might be seen in the data.

5 An end-to-end check

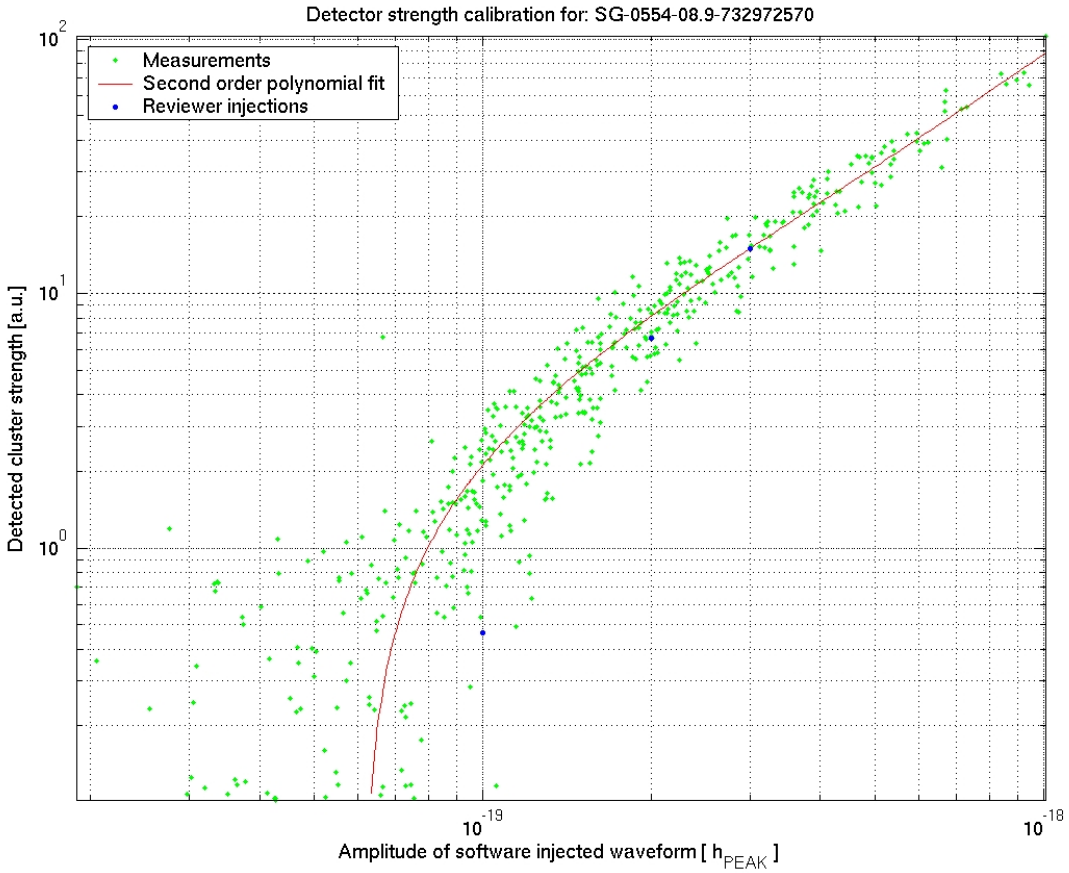
As a final, independent validation of the entire pipeline, Teviet used the MATLAB code to analyze data into which simulated signals had been injected "by hand", completely independently of the analysis code.

Three calibrated Sine-Gaussians with frequency 554 Hz and Q of 8.9 were added to the raw AS_Q data, with different amplitudes and injection times. It was verified that three new clusters appeared in the output list, with detection times and signal strengths consistent with the injection values and the Szabi's Monte-Carlo pipeline calibration. The specifics are listed below:

h_{peak}	Time from start of data (s)		Cluster "strength"	
	Injected	Detected	Expected	Detected
1.0×10^{-19}	235.06787109375000	235.076	0.4–3.3	0.465
2.0×10^{-19}	123.95678710937500	123.960	5.5–12	6.716
3.0×10^{-19}	101.71234130859375	101.716	10–20	15.136

Injection times refer to the centre (peak) of the injected sine-gaussian. Note that the detection times are systematically later than the injection times, but only by one or two 4-ms bins: while puzzling, it is unlikely to affect the scientific conclusions drawn from this search.

Note that the 1×10^{-19} waveform is at the ragged edge of detectability above noise, which is why the expected cluster strength spans such a large range. Expected cluster strengths were measured by hand from the calibration plot produced by the code. This plot is given below, with the code's own Monte-Carlo injections shown in green and the three independent injections shown in blue:



From the graph it appears that the independent injections also appear systematically lower in amplitude, though not by any extreme amount: they still lie within the range of the Monte-Carlo calibration. These few "by-hand" injections were not intended to produce an independent calibration, but only to verify that the MATLAB software was not obviously doing anything wrong in its calibration. The code passed this test, so the reviewers feel confident that the results of the code are trustworthy, to within the purported precision.

6 Conclusion

We felt we were able to do a comprehensive review of all of the code used to do the analysis. The most significant problem we found was due to using too coarse a time step when calculating the efficiency for short-duration signals; this problem was corrected. We also pointed out some minor bugs/gotchas which had no measurable effect on the results. Overall, we judged the analysis to be appropriate and well implemented.