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

Presentation	**LIGO-G1601123-v2**	2016/06/30

Extending a Plotting Application for the LIGO Open Science Center Progress Report 1

Nicolas Rothbacher, Mentor: Eric Fries
University of Puget Sound

Distribution of this document:

LIGO Scientific Collaboration

Draft

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, Room NW17-161
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352
Phone (509) 372-8106
Fax (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189
E-mail: info@ligo.caltech.edu

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

1 Introduction to Gravitational Wave Research

Albert Einstein first theorized gravitational waves in 1916 as a consequence of his theory of general relativity. The predicted amplitudes of these waves were incredibly small, making observation of these waves difficult [1]. In the 1960s, Joseph Weber led the first serious efforts to detect gravitational waves [2]. Weber's original design used a large mass of aluminum and a piezoelectric detector to sense changes in length of the aluminum. In the late 1960s and 1970s, prototype interferometer detectors were constructed, and in 1980 the NSF funded the construction of a 40-meter prototype at Caltech and a 1.5-meter prototype at MIT. These prototypes demonstrated the feasibility of a large interferometer and led to the creation of the Laser Interferometer Gravitational-Wave Observatory (LIGO) [3]. On September 14th, 2015 Advanced LIGO (aLIGO) directly detected a gravitational wave [4].

aLIGO is pair of detectors constructed using a modified Michelson interferometer design: one in Hanford, Washington and the other in Livingston, Louisiana. The detectors are separated by 10 ms of light travel time, and are oriented slightly off-axis from one another. The interferometers use 1064 nm lasers and $L = 4$ km arms, along with a number of signal enhancements. A passing gravitational wave induces a change in the length of the arms $\Delta L(t) = \delta L_x - \delta L_y = h(t)L$, where h is the strain amplitude projected onto the detector and δL_x and δL_y are the changes in arm length. h is the primary output channel, and the sensitivity of this output is determined by two main factors: the antenna pattern and the frequency of the wave detected. The antenna pattern is dependent on the sky position of the binary merger that produced the gravitational wave, the inclination of the binary orbit, and the orientation of the arms of the detector [6].

Figure 1 shows the amplitude spectral density of the primary output, demonstrating a variation in sensitivity dependent on the signal frequency. This variation in strain sensitivity is due to different sources of noise. At low frequency, the limiting noise source is seismic vibration. At intermediate frequencies it is Brownian thermal noise originating from the mirror coating [5]. At high frequencies, the quantum shot noise produced by the Heisenberg uncertainty principle become the major noise source.

The gravitational wave event detected on September 14, 2015, GW150914, swept through frequencies from 35 to 250 Hz in 0.2 s with a maximum strain amplitude of 1.0 x 10^{-21} occurring at the highest frequency. GW150914 is the first direct detection of gravitational waves. This event has great historical value, and it is vital that the data and analysis be presented to the public in an effective and understandable format. The second gravitational wave event detected by LIGO, GW151226 was more faint, with a maximum strain of 3.4 x 10^{-22}, sweeping from 35 to 450 Hz in 1 s [7]. Having an understandable representation for this event will be even more important than for GW150914, since it is less apparent on first glance, making signal processing vital to identifying the signal.

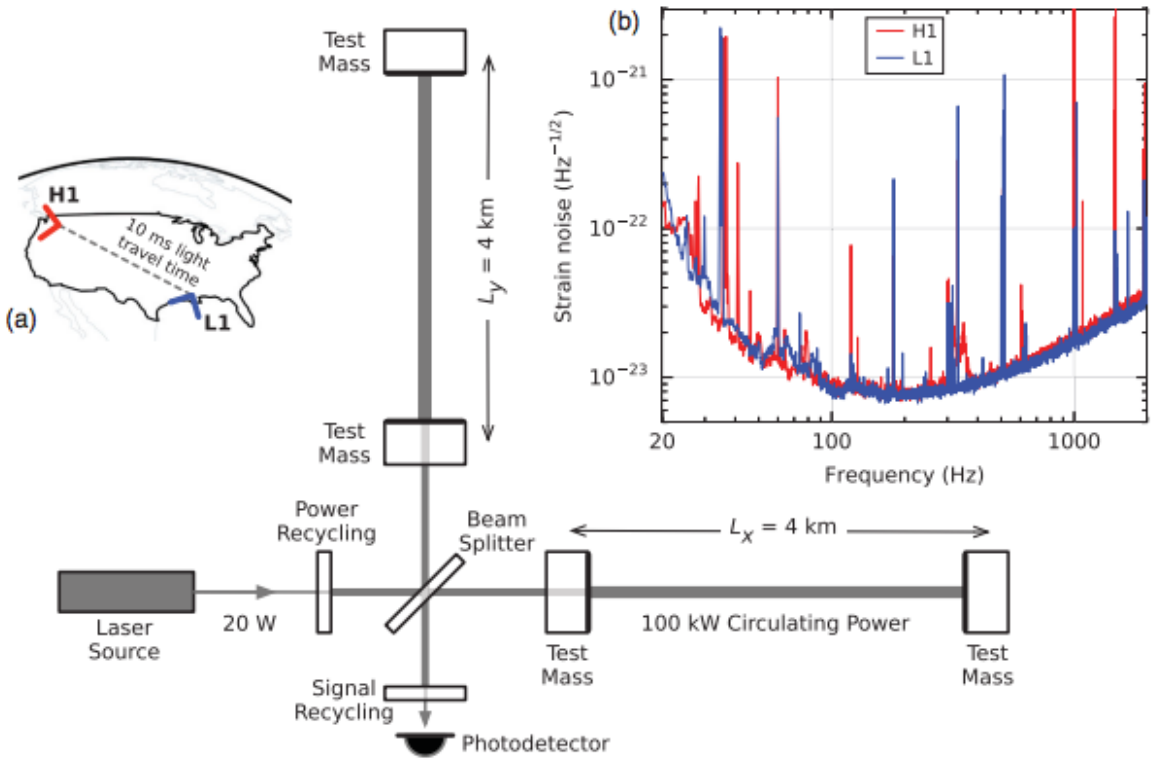

Figure 1: a) Arrangement of detectors and configuration of interferometer. b) Strain noise vs. frequency, showing the highest strain sensitivity in the middle part of the frequency spectrum.

2 LIGO Data Processing and splot's Place in the Picture

To process these signals and parse out the noise, LIGO performs a range of calculations and creates a number of plots. The first of these, an amplitude spectral density (ASD), helps to identify noise by calculating the square root of the power spectral density; the resulting plot is shown in Figure 1 b. This set of amplitude data in the frequency domain is then used to "whiten" the strain data of the signal by fast Fourier transforming (FFT) the strain to the frequency domain and dividing by the ASD and using inverse FFT to return to the time domain. The resulting time series is as seen in the top row of Figure 2. This data can be plotted to a spectrogram of the data, an optimised version of this graph is shown on the bottom. Bandpass filtering and matching to theoretical models the data are also employed to further confirm and clarify the gravitational wave signal. The final step in the data presentation is to convert the signal to an audio file since these waves propagate in frequencies within human hearing [8]. At the conclusion of this project all of these representations are or will be incorporated into splot, the web-based plotting interface.

This project will expand the existing plotting software used by LIGO to represent the data in an interactive and publicly useful manner. splot, generates static plots of a time series of the strain, as well as a spectrogram and ASD of the strain. splot can also apply various types of filters to the data, adjust the sampling rate, and perform similar data manipulations. These options and filters must all be applied prior to submitting a request, making changing the parameters a slow and awkward process. The goal of this project is to expand this functionality to include interactivity and other

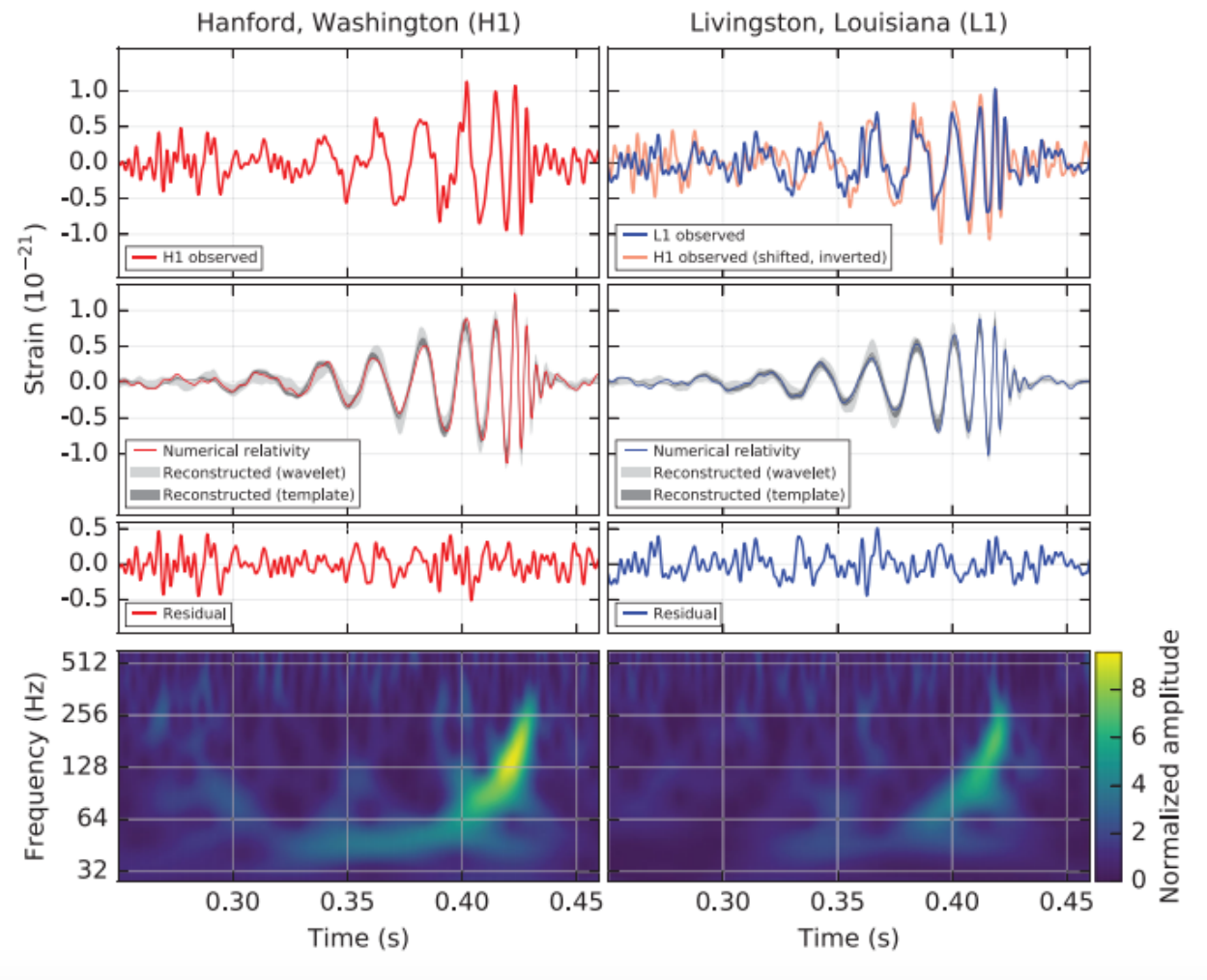

Figure 2: Three sets of figures from GW150914, the left column from the Hanford detector, the right from the Livingston detector. The top row shows whitened gravitational wave strain over time. The second row shows a combination of different predictive techniques for theoretical wave models. The third row shows the residuals of the signal when the theoretical model is subtracted. The bottom row is a spectrogram of the data.

clarity by moving plotting to the client and allowing the user to dynamically apply each step of the data processing.

3 Next Steps for splot

This project has three goals: to introduce interactive features to splot, to integrate additional existing data analysis algorithms, and to create a simplified version that does not use JavaScript. The primary objective is to expand splot to allow the user to interactively zoom in on different parts of the plots, and apply a filter or modify an existing filter. Both of these features would allow real time manipulation and would not require the user to reload the page. An extension of this objective will

allow the user to work through the data analysis process step by step, applying filters and making plots along the way. This might only materialize surrounding the documented events, but could be a valuable resource to understanding the process to confirming thse signals. The secondary objective is to allow the user to produce an Omega-scan, a correlogram, and an audio file of the strain. An Omega-scan, shown at the bottom of Figure 2, is a version of a spectrogram that tunes input parameters to maximize the power recovered by the spectrogram. A correlogram displays the correlation between the strains in the two detectors, after correcting for differences in phase and shifting in time. This will allow the user to see the importance of having two detectors and to visualize how the two detectors confirm the observation. An audio file is of interest as the signals are usually in the frequency band that is audible to the human ear. The tertiary objective is to implement a version of splot that works without the use of JavaScript. This version of splot would have to be greatly simplified.

The primary objective of implementing interactive elements will be accomplished by using the preexisting Python code in conjunction with JavaScript to ensure that the user can access these features without needing to reload the page. This will be accomplished with the open source plotly.js library [9]. This objective is nearly complete, as plotly comes fully built with all of the plot types used in this tool, data processing will need to stay on the server for now, as JavaScript is not as useful as Python for this purpose.

The secondary objective of producing additional outputs could be somewhat more difficult for two technical reasons. First, the servers may not be able to return an Omega-scan or a correlogram in a timely manner. In this case, the possibility the user submitting a request for the plot and alerting the user when the plot is ready should be investigated. Second, the integration may be difficult because it is plausible the existing analysis code cannot be easily merged with splot. Either of these technical issues may make the secondary objective intractable.

The tertiary objective is a much lower priority than the other two. It would involve writing a configuration case with a much reduced set of features that could be used on a browser without JavaScript enabled. This could require rewriting a considerable amount of code and would only serve a very small portion of the public, so this would be attempted only after exhaustive work on the other two objectives.

4 Progress So Far

The first task undertaken was the research of different JavaScript libraries to be applied to the goals. Several libraries were investigated, including mpld3, bokeh and plotly.js. mpld3 and bokeh were both not suited to the task as they take an already generated python plot and generate a JavaScript version. This does not provide total control in real time, so plotly.js was determined to be more applicable as it simply receives data and creates a plot on the users' computer, better for interactivity. plotly.js was then tested in a prototype HTML file and was shown to have functionality for all the representations that LIGO outputs, heatmaps for the spectrograms and Omega-scan, log-log plots for the ASD and standard linked scatter plots for time series and correlograms. A feature

of plotly.js to allow data and formatting linking accross multiple plots was also discovered in this process, opening up a new feature, to allow more intuitive zooming across multiple domains. This concluded the initial research phase and work moved to implementing plotly.js to splot.

Starting in the second week, work began on the existing splot program. splot is a Python script hosted on a Django web server that outputs resulting plots to images in a generated HTML file. This generation is triggered by the submission of a Python dictionary from an HTML form that acts as the parameters of a plot creation function. To implement plotly.js into this process, splot was not changed except in the output to image form: instead of an image, the values to be plotted were instead fed to a JSON object and sent to the resulting HTML document to be handled in the front end.

4.1 Challenges

The greatest obstacles with the progress so far have come from administrative or security requirements to beginning work on the project. Having the permissions and resources to develop using the web server required the most work, with a number of different people involved in approving my permissions and making sure that the development server was operational. Other obstacles came in the actual implementation of plotly.js to the splot code since it works in two different languages problems arise with variable type conversions and function incompatibility coming up fairly often. In order to counter some of these errors, an open source Python linter was used to automatically check the code for syntax errors [10]. Despite these issues, the most basic features of plotly.js, plotting the data on the client side, are very close to being successfully implemented.

5 Goals for the Near Future

The biggest goal for the next month of the project will be to fully implement all of the features plotly.js has to offer and to begin the process of adding the new representation types into splot. These features will allow the project to demonstrate the data processing that occurs at LIGO to the user and accomplish the biggest goals of the project. In conjunction with these larger goals, the process of improving on the current code and improving the user interface would also be good objectives. Both of these aspects are fairly basic, serving in a utilitarian fashion and it would be beneficial to make them more developed. The resource required to accomplish these goals will be development time, as many are simply code manipulation and the others involve implementing code from other LIGO projects

Going forward with these goals, many of the same challenges will be prevalent. splot will always be working between multiple languages and will only become more complicated as its functionality is spread between multiple files. Some of these code upgrades will also require changes to the web server code, which will involve Django and introduce more chances for bugs and errors. Additional problems may come from difficulties with processing time on the server side since most of the computational work is done on the server. Processing times are already fairly long and adding

more plotting ability would only increase the requirements. There are client side ways to deal with this but could introduce more complexity and bugs so this challenge will not be a simple fix.

References

[1] A. Einstein, Sitzungsber. K. Preuss. Akad. Wiss. 1, 688 (1916).

[2] LIGO, *A Brief History of LIGO*,
`https://www.ligo.caltech.edu/system/media_files/binaries/313/original/LIGOHistory.pdf`

[3] J. Weber, Phys. Rev. 117, 306 (1960).

[4] B. P. Abbott et al., PRL **116**, 061102 (2016).

[5] S. J. Waldman, *The Advanced LIGO Gravitational Wave Detector*,
`https://arxiv.org/pdf/1103.2728.pdf`

[6] D. G. Keppel, Doctoral Thesis, 33 (2009).

[7] B.P. Abbott et al., Phys. Rev. Lett. **116**, 241103 (2016).

[8] A. Weinstein, *Signal Processing with GW150914 Open Data*
`https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.html`

[9] Plotly.js, `https://plot.ly/javascript`

[10] Pylint, `https://www.pylint.org/`