Analysis for Recovered Compact Binary Coalescence Hardware Injections

Researching Process and The Final Results

Name: Jinghong Liang

Mentor: Professor Eric Myers

Pioneer Spring Study

Investigating Gravitational Waves with Data from LIGO

July 2015

Abstract

This paper is about the study of gravitational waves with the data from LIGO's 5th science run (S5). It provides basic background information about gravitational waves, its source and the LIGO. It described the preliminary goal of research, the following process during the half of year and the final researching target. The methodology of how to recover the compact binary coalescence injections and how to create the plots is mentioned, as well as a brief analysis based on these resources. The reader is assumed to be familiar with basic physical knowledge, having interests in the cutting-edge area of physics, but not necessarily has any background information about LIGO or gravitational waves.

1. Introduction of Gravitational Waves

In 1916, Albert Einstein predicted the existence of gravitational waves in his theory of General Relativity, 11 years after the publication of Special Theory of Relativity. He compared the universe to a stretched fabric made up of space and time. Gravitational force is the curvature caused by the presence of mass in the fabric of space-time [1]. If there are large accelerating masses moving or orbiting around each other, the movement would generate ripples in space-time; when these ripples spread outward, they present wave-like properties, form the gravitational waves [2]. Scientists have already given scientific evidences to support the existence of gravitational waves indirectly, involving large amount of calculation and observations toward astronomical objects. Based on the analogy between the equations for electromagnetic waves (Maxwell's Equation) and gravitational waves (Einstein's Equation), the scientists began to predict unknown qualities of gravitational waves [3]. However, until now, no one has direct evidence to prove the validity of gravitational waves.

2. Sources of Gravitational Waves

Based on the equations of gravitational waves, scientist concluded that the gravitational waves had quadrupole, which means they have two polarization states, "+" and "×" [3] . In order to generate gravitational waves, the masses' quadrupole must present asymmetrical movement, which, in other words, means that all spherical symmetrical events cannot produce any gravitational waves. There are four main kinds of astronomical sources.

Firstly, stochastic background from the early universe. This kind of wave is likely to be born from the events of very beginning of the universe, such as the "Big Bang." It could come from every direction of the sky continuously, so the scientists used the method "all sky blind search" — which means to search for the whole sky without a specific target in mind— to look for it. The scientists are paying efforts in the production of "cosmic gravitational wave background," to

plotting the distribution of gravitational waves sources, just like the graph of "cosmic electromagnetic wave background," and hope it could provide more valuable information about the birth of our universe [3].

The second category is the bursts from supernovae or other cataclysmic events. The exact cause behind these sources is still unclear since people do not know how gravitational waves are produced in bursts. However, one thing is certain: no gravitational wave would be generated if the burst is spherical symmetrical [3].

The third kind includes continuous wave sources. The persistent gravitational waves could be generated by asymmetrically spinning binary systems. Scientists get information about the strength of waves through the measurement of the strain amplitude, which reflects the condition of stretching or shrinking in spacetime. The rotational frequency of this kind of binary systems is nearly constant, and the frequency of gravitational waves generated is twice the rotational frequency. Scientists are also expecting the appearance of some kind of "gravitational pulsars" to produce gravitational waves continuously, just like the pulsars producing electromagnetic waves [3].

The last type is coalescence of binary systems. Difference between the continuous wave sources and compact binary coalescence is that the former one is a constant spinning movement, and the two masses rotating about each other has a certain distance between; while in the CBC condition the pair of circling masses would gradually getting closer to each other, and at the same time, their speed would increase largely and produce gravitational waves having higher and higher frequencies. The binary system could always produce gravitational waves, but initially the frequency is too low for LIGO to detect. At the last moments before the two objects merged into each other, the frequency would reach the range which LIGO is most sensitive, and thus can be detected. The system include "neutron star - neutron star", "neutron star - black hole" and "black hole - black hole" three types [3]. This is the type of injection which will be discussed later in this paper.

3. Introduction of LIGO

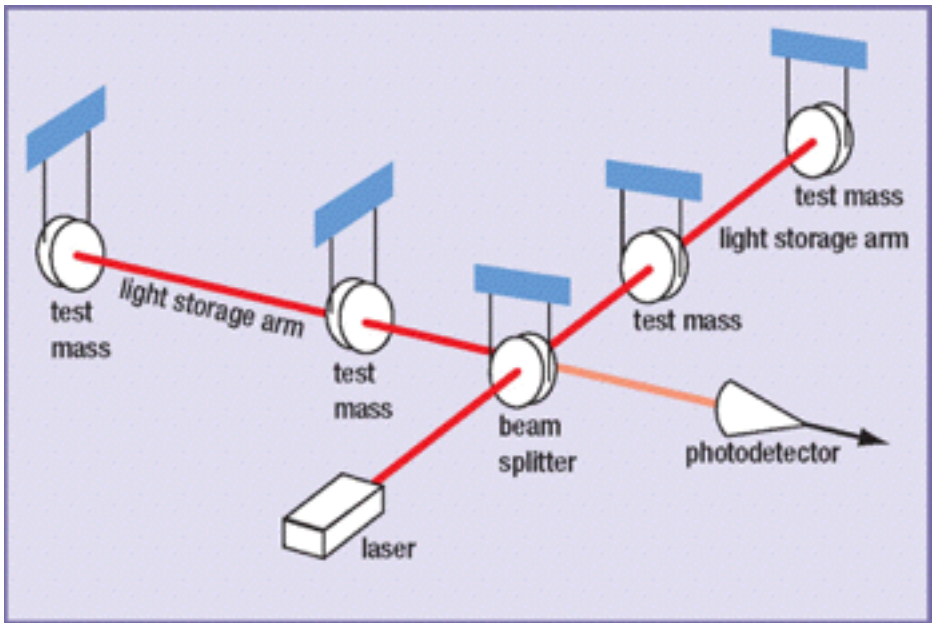


Figure 1: A schematic diagram about inner structure and basic principles of how LIGO works

LIGO, which stands for Laser Interferometer Gravitational Wave Observatory, was established in 1992 as a result of the cooperation between Caltech and MIT. LIGO has two observatories in the United States: one is in Hanford, Washington, while the other is in Livingston, Louisiana [4]. By comparing the data collected from these two LIGO stations at the same time, scientists are able to determine the validity of one possible gravitational wave signal. The LIGO Scientific Collaboration (LSC) searches for any signals of gravitational waves from the data LIGO collected, about the waves generated from binary system coalescences (neutron stars or black holes), or about the orbiting continuous system. In order to examine the precision and accuracy of instruments of the interferometers or the efficiency of measurement, scientists put testing signals in the experimental process [5].

LIGO has period when it works and collect data reached scientific standards. After the construction of the first LIGO, there are periods that the researching process was interrupted to upgrade the LIGO instruments. Only during the Science Run, data from LIGO instruments are

reliable and investigable. The S5 (the 5th Science Run) period starts from the fall of 2005 to the fall of 2007 [3]. All datafile used in this paper come from the S5 Period.

The interferometer LIGO uses was invented by the American physicist A. A. Michelson (1852-1931). As Fig.1 shows, the light beam from laser transmitter is splitter into two when it passes through the beam splitter; after the reflection, the two beams of light combine into one again and travel to the photodetector. During this process, if there is any changes in the distance (which is precisely related to the exact number of wavelength of light) the light beams travel, their phases would change (not perfectly matched), and they would perform interference effect, which would be detected by the photodetector. The scientists are expecting the weak changes in spacetime could be detected through this precise and accurate device [4].

4. Injections

Injection,"is the process of adding a waveform to interferometer data to simulate the presence of a signal in the noise. We use injections to measure the performance of analysis pipeline" [5]. Scientists mainly use two different types of injections, software injections and hardware injections.

Software injection means the added imitated signal to the data recorded during the experimental period. The software injections are the ideal condition of possible detected samples, and they would not affect the real operating situation of the machines of LIGO. The results could show the current capability of the software which is used to calculate all the data and to find specially potential signals.

Hardware injection is to add a real waveform to the machine of interferometer. It represents the physical stimulated signal applied to the system. The main focus of hardware injection is the examination of the sensitivity of the pipelines, to show that the current system is capable of catching the theoretical signals from the outer space. To get the real measure of accuracy and

sensitivity of the interferometer, the scientists would inject certain number of signals to the pipelines, and after the recovery of data, try to calculate the percentage of received signals [5]. In fact, the hardware injection is one of the most complete and thorough way to carry on the test of the instruments. The examination could approve advantages of current version of LIGO, reveal short comes in many different aspects and give more suggestions for the future generation of LIGO.

5. Researching Process and Working Direction

Before discussing about my specific working direction for this project, I want to talk about my process of research, since my research proposal changed again and again, and the current direction is the result of modification over several times.

The original research question is: “Are there any differences between hardware injection and software injection in the data stream?” I came up with this first question because I noticed that although there are two different kinds of way to examine the instrument or software of LIGO, people always tend to pay more attention at hardware injection instead of the one of software. The hardware injection is added during the science run, and the software injection is added directly into the database of collected signals. I wanted to know the dissimilarities of these two due to the different ways they were injected to the system. This plan failed since there were too little information about software injection, and the list for software injection was nowhere to find.

To shift the focus, I narrowed the question down to discover more on the qualities of hardware injections. I raised the topic: “Are there any differences in the data stream between compact binary coalescence injections and burst injections?” In order to understand more of the two kinds of hardware injections, I need to analyze some of the samples on my own. I learnt the method listed on the LSC official website and, with the help of tutorials, compiled correspondent python scripts to create graphs for both injections and locate the injection signal. Unfortunately, the script for burst injections never showed the graph, and despite all the effort I couldn’t figure out the problem; for

the compact binary coalescence injections, the spectrogram didn't demonstrate the expected results and clarity as the sample graph given on the LSC official website.

Finally, I settled on the question: "Is it possible for me to restore the graph showing on the LSC official website for Compact Binary Coalescence Injections with the help of official tutorial?" The spectrogram placed on the website had great clarity, and it is easy for us to find the uniqueness of this injection. I decided to discover more on this topic.

For this research about gravitational waves using the data from LIGO, I am very interested in the Compact Binary Coalescence injection. The more specific topic would be "Is it possible for me to restore the graph showing on the LSC official website for Compact Binary Coalescence Injections with the help of official tutorial?"

I have several reasons for researching on such topic. First of all, compact binary coalescence injection is one type of the hardware injections, and there are still something remain to be discovered. Secondly, it should be possible for the starters to follow the tutorials posted on the official website and recover the injections, thus the example of the website, especially the clear and precise graphs, should be able to be reproduced. If we cannot even reproduce the sample, or we get a different conclusion, the result we come up with might be dissimilar with the scientific proven ones, and the reliability would be reduced. To recover the official example should be our first step to get to know a certain topic.

6. Methodology

In order to give a close look at the graphic quality of compact binary coalescence injection, I need to find the injections from database and analyze them for the first step. LOSC (LIGO Open Science Center) has provided lists for injections. The list of CBC injections include the GPS time, specific interferometer (L1, H1 or H2, stands for the interferometers in Livingston and the two arms

of different length in Hanford), the mass 1 and mass 2 in Solar Mass ($M_\odot$, which stands for the mass of the object over the mass of our sun) for rotating objects, the distance between the source and the Earth in Mpc, the state for injection (successful or not) as well as the expected and recovered SNR (Signal-Noise Ratio, pointed out the clarity of this gravitational wave).

To discover something about the compact binary coalescence injection first, I used the method called Control Variable to determine the factor influencing the SNR of compact binary coalescence injections, by filtering out the effect of mass and distance under the same situation of LIGO in H1. After choosing one certain line of injection within the list, I went to the online database of LOSC to search for the HDF5 file of the time period which include the time of injection. Based on the tutorial given by LOSC, I had compiled some python scripts to read HDF5 files.

For the Compact Binary Coalescence Injection, I used the script in Appendix A. After reading the file, it would provide a graph for data quality, which 0 stands for no data, and 1 stands for data in good quality. I would check that if the injection is included inside the range of good quality data file. Then, python would calculate the length of noise segment and injection segment. This step tells us the lasting time for injection segment (usually about 100 seconds), not the time for injection itself, which would only last for seconds. Import the given script for template [Appendix B], I created the specific template for the signal, for the template of each signal is somehow different. Based on the feature that the noise is high at low frequency, it eliminated background noises at low frequency. By the matched filter, I located the signal, and then printed the spectrograms which stands for the intensity of power. Finally, the script gives the repost about expected and recovered SNR.

For the recovery of the sample on official website, I used the same python script for CBC injections [Appendix A] but paid more attention to the spectrogram to find the similarity between my graph and the official graph. Before it, I have to try to locate the signal to see if it is at the location described by LSC.

7. Research Result Presentation

7.1 List of Recovered CBC Injections [6]

GPS Time	Info	M1/M ⊙	M2/M ⊙	Distance/ Mpc	Log	Exp_SNR	Rec_SNR
817372998	H1	1.4	1.4	2	Successful	99.00	82.14
826891834	H1	1.4	1.4	0.1	Successful	2283.05	1804.37
824555709	H1	1.4	1.4	10	Successful	22.83	20.41
826833378	H1	3	3	10	Successful	43.77	41.77
833608964	H1	10	10	10	Successful	92.43	83.39
817645695	H1 Sample	1.4	1.4	5	Successful	38.47	31.56

7.2 Clarification of Concepts

In order to explain my results in a more clearly way, I would like to introduce several concepts that will be mentioned in the report.

The first idea is about data quality. Data quality is very important for my analysis to the resources because I need to identify the proper time range in which I can get data whose quality are not compromised. To present this character in a more direct way, I printed the plot for data quality before any other data analysis began.

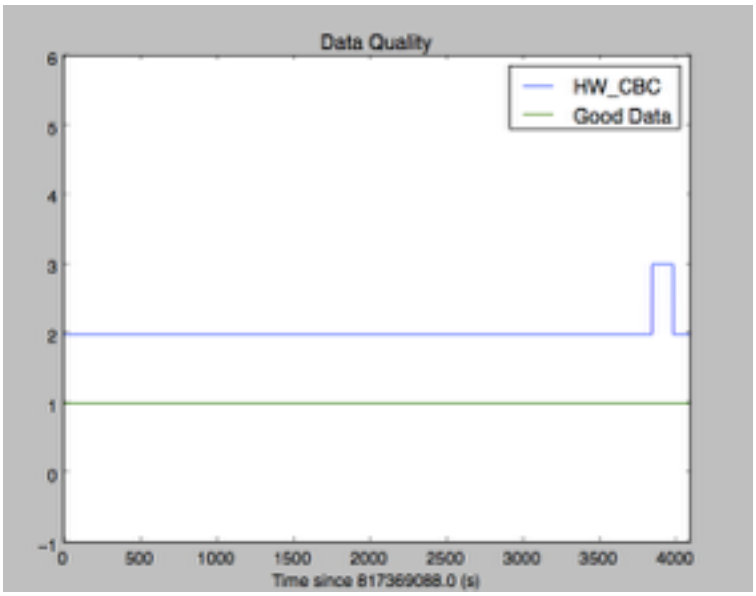


Figure 2: An example for the data quality plot

Figure 2 is an good example for the data quality graph I plotted with the help of python scripts. There are two lines in the graph: the line for “Good Data” and the line for “HW_CBC.” The “Good Data” line has two states: 0, which stands for no good data, and 1, which means that the data quality is good enough for scientific research. The “HW_CBC” line also has two states, but in order to put both lines inside one plot, I added 2 to each of the point to move it upward, and give a comparable view to the information. “1” state for “HW_CBC” stands for injection period, and “0” means no injection at all. In the example of Figure 2, the data quality is always good for research purposes, and the injection happened between 3500s to 4000s after the GPS time 8173690880, which is the starting point of this HDF5 frame file.

In later descriptions of the results, I would not mention the graph of data quality for every single sample; instead, I will say “the data quality is good during the injection period” to show the same effect.

The second concept is about the template. The template is the standard scientists use to find the injection. For the research toward totally unknown objects (with unknown mass, distance and so on), it would be hard to make a template and try to match every signal collected with it. However, my mission here is much easier. I have the information for specific injection time, solar mass for the coalescing objects, and the distance between the source and the earth. According to these information, I am able to create template for every injection with the python script [Appendix B]. Each template is different in waveform or other factors, so I must create correspondent template for the injections to get the right result.

Figure 3 is an example for the template. To say it more clearly, a frequency domain template. The reason why I created frequency domain template is that I could zero out the noises. Noises which have frequency lower than 25 Hz are very strong, and they can be misleading when we read the graph without clearing them out. In Figure 3, the template has frequency starts from 25 Hz to about 350 Hz. One thing I need to do is to Fourier Transform all my data,

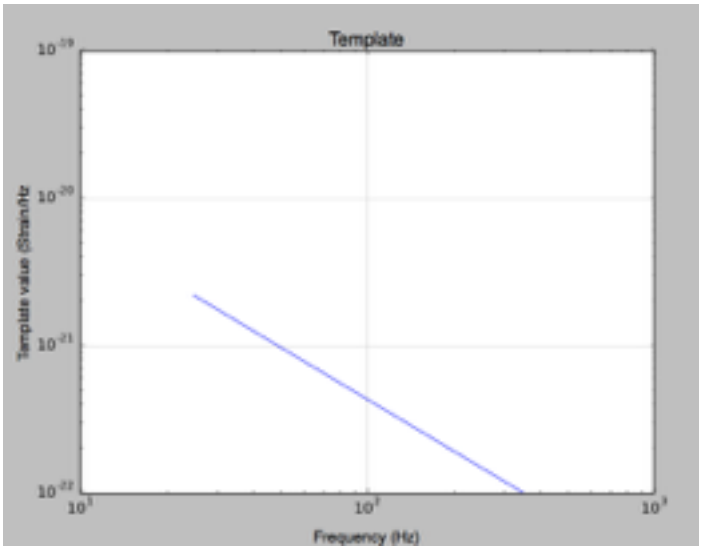


Figure 3: An example for the template of one injection

since in the frame file, they are time domain data. Through Fourier Transform, the data could become comparable with the template, thus I can find where the injection is.

In fact, the template would not affect my explanation of the final result, so I just give an example here to make the idea of template clear. Later I would not show templates again.

7.3 Research Result Part 1

In this part I would discuss my current results for the research on compact binary coalescence injections. I would utilize the data from the former 5samples in my list, and this part would not include the report for observation of the last sample on the official website.

As I mentioned before in this paper, I used the method of control variable to seek the cause which influenced the magnitude of SNR the most. To discover the influence of distance between the source of the compact binary coalescence and the Earth, I gave the graphs and analysis for the first three samples, in which both of the objects have the solar mass of 1.4 $M_\odot$ but with different distances. Two points are noteworthy: firstly, 1.4 $M_\odot$ can be the mass of neutron star, so for these 3 samples are the neutron stars revolving and colliding into each other; secondly, the solar mass for every set of circling objects are the same. For the second point, I have to admit that I did not find the answer yet.

Table 1: Effect of Distance

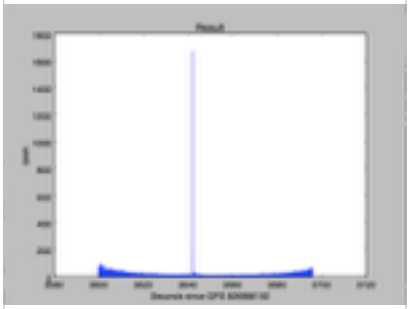
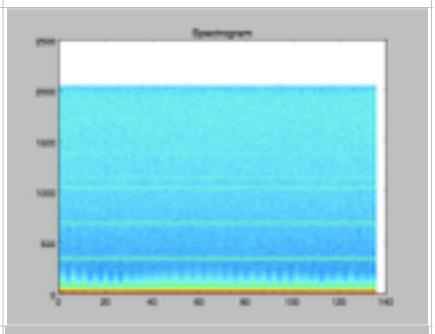
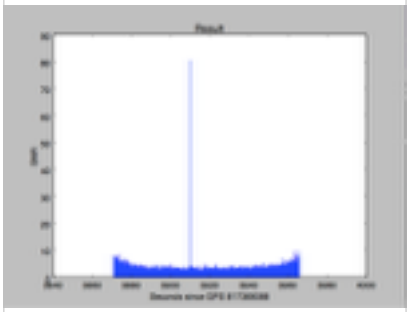
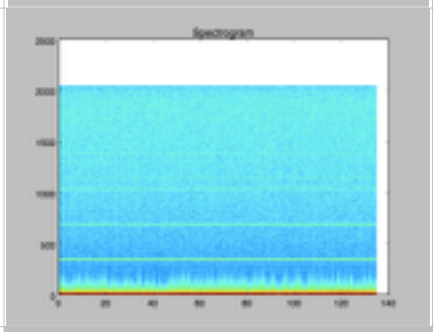
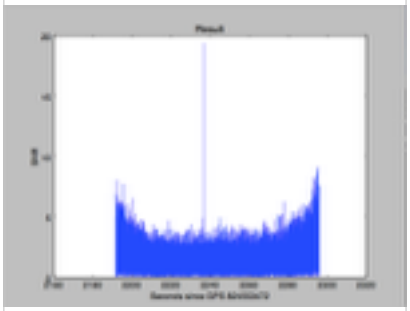
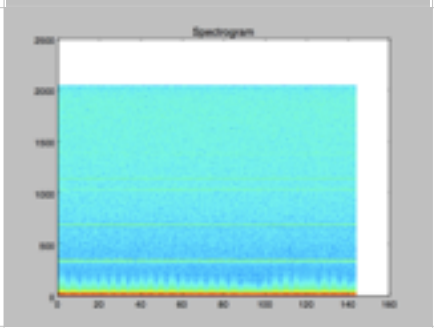
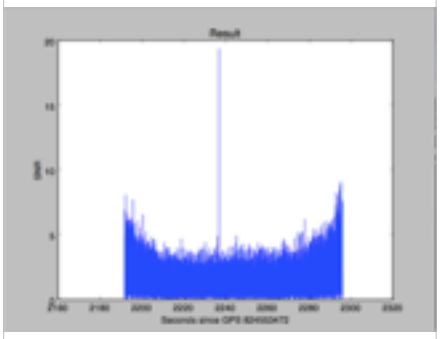
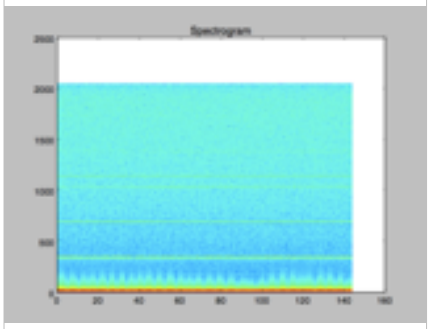
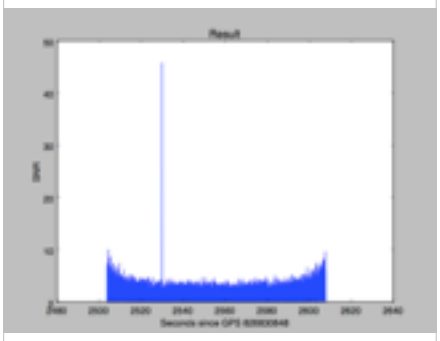
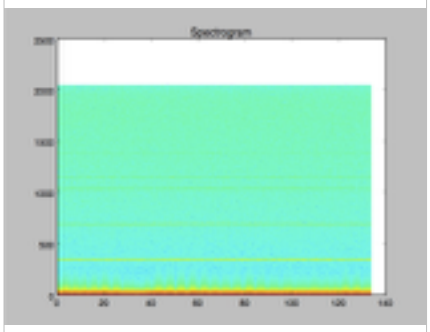
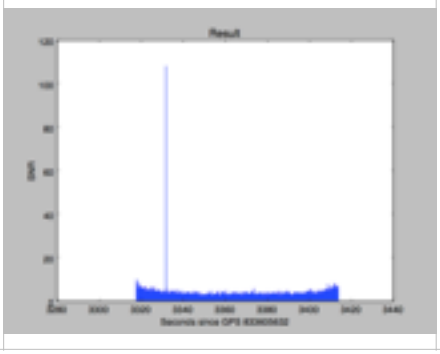
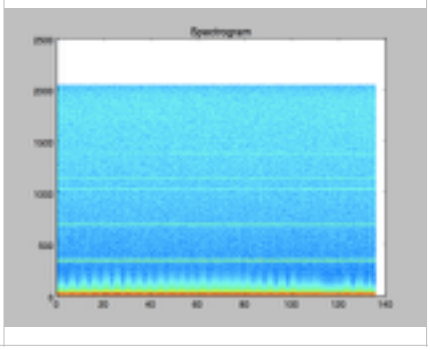
GPS Time	Plots (Figure 4 to Figure 6)	Result	Distance	Spectrogram (Figure 7 to Figure 9)
826891834		Expected to find SNR 2297.0 Recovered SNR 1720.5 Recovered time GPS 826891834.1	0.1	
817372998		Expected to find SNR 106.0 Recovered SNR 81.4 Recovered time GPS 817372998.1	2	
824555709		Expected to find SNR 24.8 Recovered SNR 20.0 Recovered time GPS 824555709.1	10	

Table 2: Effect of Solar Mass

GPS Time	Plots (Figure 10 to Figure 12)	Result	Solar Mass	Spectrogram (Figure 13 to Figure 15)
824555709		Expected to find SNR 24.8 Recovered SNR 20.0 Recovered time GPS 824555709.1	1.4	
826833378		Expected to find SNR 52.2 Recovered SNR 47.9 Recovered time GPS 826833378.2	3	
833608964		Expected to find SNR 124.5 Recovered SNR 110.7 Recovered time GPS 833608964.1	10	

In order to learn more about the effect of the objects' solar mass on the SNR, I controlled the variable of distance to make it a constant of 10 Mpc. I also gave 3 examples in this part, for there are only three category of solar mass given on the official website. The first is 1.4 $M_\odot$, the mass of a neutron star. Then, 3 $M_\odot$, the upper most limit for the neutron star, since everything has a mass above 3 $M_\odot$ would be considered as possible blackhole. Finally, 10 $M_\odot$, the mass of large blackholes merging into each other. With a constant distance between the Earth and the source of compact binary coalescence, I was able to determine the relationship between solar mass and SNR better.

For the part of the influence of the distance, from the list of results we can see that with the increase of the distance, the SNR is becoming smaller and smaller.

In the first example at GPS time 826891834 has a very short distance of 0.1 Mpc, in figure 4, a little bit past 3640 seconds after the beginning of this frame file, there is a nice big peak which is quite outstanding among all other shorter blue lines. This peak stands for the injection. From the y-axis in this graph we can read that the SNR of this injection is already over 1700, which is a very large number for this term. The injection is very distinguishable since other noises are having a SNR lower than half of 200. However, when I looked at the spectrogram, the very obvious injection in the SNR graph is nowhere to see.

The second injection has a distance of 2 Mpc and happens at GPS time of 817372998. The data quality is good over the period of injection. From Figure 5 I read that the actual recovered SNR is a little bit higher than 80. On average, the SNR of background signal level is less than 10. From the recovered SNR we can notice a significant decrease compared with the former signal which has a distance of 0.1 Mpc.

Next injection with solar mass of 1.4 $M_\odot$ has the distance of 10 Mpc, further away from the Earth. The correspondent result shown in the graph Figure 6 tells us that this time the recovered

SNR does not even reach 20. The SNR of surrounding background signals are still lower than 10, but it seems that compare to the background noise in the two injection periods above, there is a tendency of decrease in the background noise. The exact reason causes this change could have a variety.

From these three graphs (Figure 4 to Figure 6) I observed some patterns. Firstly, under the condition of the same solar mass but different distances, the recovered SNR of compact binary coalescence injection is inversely proportional with the distance. Secondly, there are several lines shown in the background of the spectrogram, and they could stand for the signals of constant frequency on Earth, possibly the electrical signals or others. Thirdly, because all of these graphs only show part of the full frame file (the part with the injection segment, lasting for about 100 seconds), they seems to have a "decrease—increase" pattern within the background noises. A rough curve over the SNR graph could has a concave up quality, and the specific reason behind this phenomenon is remained to be observed.

The second table gives a list of graphs which is aimed for the discovery of the effect of solar mass for both objects under the same distance. We can see that with the increment of the solar mass, the SNR is growing larger and larger.

The first example in this part is the last example in table 1, so I would not repeat the result of observation again.

The two objects in the injection in Figure 11 have solar mass of 3 $M_\odot$, about twice the solar mass of the former injection. 3 $M_\odot$ means that this is the signal generated by two large revolving neutron stars. The SNR did come up, to about 45 in the graph. However, the SNR is not about twice the last SNR. So I realized that the relationship between solar mass and SNR is not simply a linear relationship. There is another worth mentioned thing: the recovered GPS time is quite accurate compared to the information given by the official website. I do not know the GPS time list was

directly given by the scientists who performed the injection or was generated after the blind search for injections.

Third injection has solar mass of 10 $M_\odot$. 10 $M_\odot$ pointed out that the wave could be produced by two coalescing black holes. The recovered SNR goes up to more than 100 in the graph, and this result proved further about my assumption for the relationship between SNR and solar mass. The wave generated by objects with larger mass would be stronger than those having smaller mass.

With the same distance between the source of gravitational wave and the Earth (although it is assumed), the recovered SNR of the injection shows a pattern of direct proportional toward the solar mass of both objects.

Nevertheless, I still cannot read the information hidden inside the spectrograms. I hope I can accomplish further research on this specific topic.

7.4 Research Result Part 2

In this part, I would discuss my central topic: “Is it possible for us to recover the spectrogram on the LSC official website for the compact binary coalescence injection?”

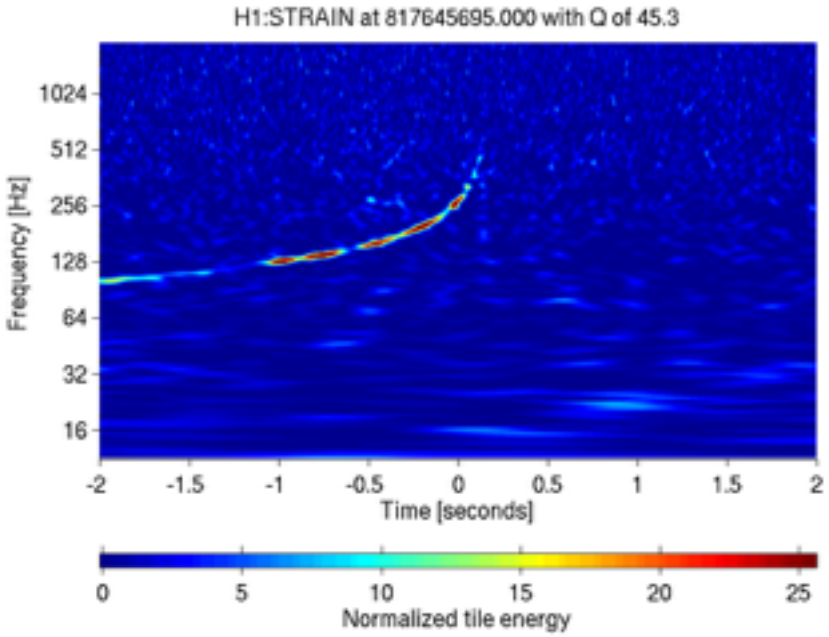


Figure 16: Spectrogram for CBC injection on the LSC official website [7]

Figure 16 is the nice picture of one sample compact binary coalescence injection. The color shows the power distribution in this spectrogram. Y-axis tells us that the frequency of this injections starts from about 100 Hz and goes up to about 500 Hz. This part is correspondent to the actual situation of a compact binary coalescence system: when the two objects revolving toward each other, the frequency of the generated gravitational waves would grow higher and higher, and then suddenly, the frequency stopped— as soon as the process of colliding is finished, there will be no longer any gravitational waves generated. This graph is normalized, and from the dark blue background we can see that the creator of this graph has already zeroed out most of the background noises and only left the most obvious injection signals in this graph. The bright red color field in the graph means these parts have higher energy level than the background, and they can be considered as “peak” in an assumed z-axis.

This graph is very clear and intuitionistic, for both professionals and the students. If I can reproduce this graph in my project, it would be easier for me to have deeper research on the spectrograms of other injections, and these kinds of pictures are better form of presentation for the final results. Different trace of the upward curve could stand for dissimilar meanings.

To recover this graph, the first thing to do is to find and download the frame file which contains this injection. However, here came the first problem I met.

817645695	H1 Sample	1.4	1.4	5	Successful	38.47	31.56

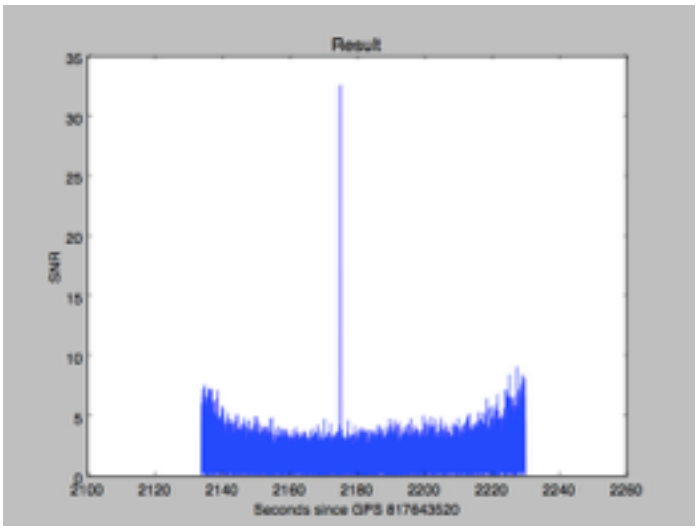
This is the information provided on the official list for compact binary coalescence injections. The first column pointed out the GPS time for this injection, which is the same as the graph’s, so it means that it does have an injection at that special time. The injection is performed in the H1 channel. The solar mass for objects is 1.4 $M_{\odot}$. However, the recovered SNR is 31.56, not 45.3, as the graph shows.

The differences between the SNR on the list and the SNR on the graph revealed another problem in my project. As the two table shows my results in plots and in words, the expected/ recovered SNR always has some deviation compared to the statistics given by the official list. I followed every step of the tutorial, so now I am not sure about where the problems is about.

I downloaded the frame file and run the python script to see if the time and spectrogram are the same as this precise and accurate graph. Here is my result.

This time, the SNR calculated by my computer has not diverged from the official one too far away.

```
Expected to find SNR 43.8
Recovered SNR 33.1
Recovered time GPS 817645695.1
```



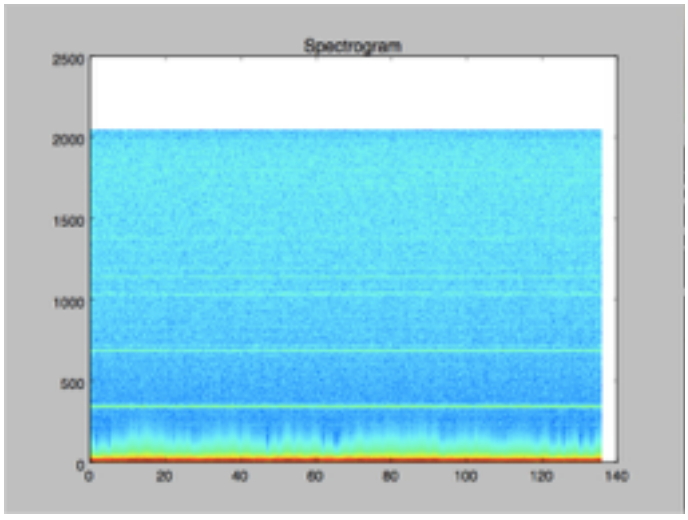


Figure 17 (Left): The SNR/GPS Time graph for the sample CBC injection

Figure 18 (Right): The Spectrogram for the sample CBC injection

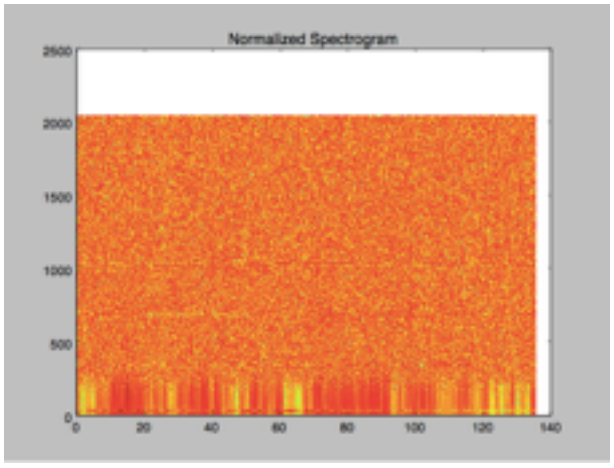


Figure 19: The normalized spectrogram of the sample injection

In Figure 17, the huge peak in the middle of the graph shows that at the time around 2170 seconds after the starting point of this frame file, which is 817643520 GPS time, there should be one injection. According to the official graph, we should see specific signs of injections in the spectrogram or the normalized version of it (Figure 18 and Figure 19). It seems that there are no special lines for the injection's frequency raised from around 100Hz up to 500Hz. Also, the background of the spectrograms are too chaotic to distinguish the signals.

Unfortunately, I could not recover the spectrogram on the website with the help of scripts from "find a hardware injection" and "lots of plots" category.

8. Conclusion

From the data and graphs I have, I can get some brief conclusions. The recovered SNR of compact binary coalescence injections is inversely proportional to the distance between the source and the Earth, and it is directly proportional to the solar mass of the circling objects. I am not able to recover the spectrogram for the injection at GPS time 817645695 on the LIGo Science Center website using the python scripts also provided by LSC.

During the process of research, I find the following problems. First, the SNR computed by my computer is different from the data on LSC website. Second, only with the python script of "lots of plots" I combined into my script is not able to zero out all the noises, thus I cannot see the hidden lines for the injection.

This project fits into my final goal of examining the feasibility of following up the official tutorial and get the same results as the scientists did. I hope my preliminary results can help students in the future who are interested in the area of compact binary coalescence injections of LIGO, and I hope my problems could be solved soon.

Work Cited

[1] Hendry Martin. An Introduction to General Relativity, Gravitational Waves and Detection Principles [Online]. Second VESF School on Gravitational Waves (2007). http://star-www.st-and.ac.uk/~hz4/gr/hendry_GRwaves.pdf [04 Apr. 2015].

[2] Aron Jacob. Einstein's Ripples: Your Guide to Gravitational Waves [Online]. NewScientists (2014). N.p. http://www.newscientist.com/article/dn25243-einsteins-ripples- your-guide-to-gravitational-waves.html [03 Apr. 2015].

[3] Myers Eric. Investigating Gravitational Waves using LIGO S5 data. Meeting 1: Introduction to LIGO [PowerPoint]. https://www.schoology.com/course/231344542/materials/gp/246927172 [Spring 2015].

[4] Schantz Matthew, Travis Herfferman, Younes Ataiiyan. LIGO: Laser Interferometer Gravitational Wave Observatory [PowerPoint]. [04 Apr. 2015].

[5] Brown Duncan A. Testing the LIGO Inspiral Analysis with Hardware Injections [Online]. ArXiv.org (2003). http://arxiv.org/abs/gr-qc/031203110 [June 2015].

[6] LIGO Open Science Center. LIGO Open Science Center [Online]. https:// losc.ligo.org/inj/ step1/ [15 July 2015].

[7] LIGO Open Science Center. S5 Compact Binary Coalescence Hardware Injections [Online]. https://losc.ligo.org/s5hwcbc/ [23 July 2015]

Appendix A

Compact Binary Coalescence Injection Recovery Python Script [6]:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import readligo as rl
# -- Read the data
strain, time, dq = rl.loaddata('filename')
strain, time, channel_dict = rl.loaddata('filename')
dt = time[1] - time[0]
fs = 1.0 / dt
print dq.keys()
# -- Plot data quality
plt.plot(dq['HW_CBC'] + 2, label='HW_CBC')
plt.plot(dq['DEFAULT'], label='Good Data')
plt.xlabel('Time since ' + str(time[0]) + ' (s)')
plt.axis([0, 4096, -1, 6])
plt.legend()
plt.title("Data Quality")
plt.show()
# -- Injection segment
inj_slice = rl.dq_channel_to_seglist(dq['HW_CBC'])[0]
inj_data = strain[inj_slice]
inj_time = time[inj_slice]
# -- Noise segment
noise_slice = slice(inj_slice.start-8*len(inj_data), inj_slice.start)
noise_data = strain[noise_slice]
# -- Length of the segment
seg_time = len(inj_data) / fs
print "The injection segment is {0} s long".format(seg_time)
segList = rl.dq_channel_to_seglist(channel_dict['DEFAULT'], fs)
length = time # seconds
strain_seg = strain[segList[0]][0:(length*fs)]


# -- Make a frequency domain template
import template
temp, temp_freq = template.createTemplate(fs, seg_time, mass 1, mass 2)
# -- Noise is high at low frequency. Get rid of noise
temp[temp_freq < 25] = 0
# -- Magnitude of the template as a function of frequency
plt.figure()
plt.loglog(temp_freq, abs(temp))
plt.axis([10, 1000, 1e-22, 1e-19])
plt.xlabel("Frequency (Hz)")
plt.ylabel("Template value (Strain/Hz")
plt.grid()
plt.title("Template")
plt.show()
# -- Fourier Transform
window = np.blackman(inj_data.size)
data_fft = np.fft.rfft(inj_data*window)
# -- PSD, NFFT
Pxx, psd_freq = mlab.psd(noise_data, Fs=fs, NFFT=len(inj_data))
# -- Matched Filter Output
integrand = data_fft*np.ma.conjugate(temp)/Pxx
num_zeros = len(inj_data) - len(data_fft)
```

```
padded_int = np.append( integrand, np.zeros(num_zeros) )
z = 4*np.fft.ifft(padded_int)
# -- Normalization
kernal =  (np.abs(temp))**2 / Pxx
df = psd_freq[1] - psd_freq[0]
sig_sqr = 4*kernal.sum()*df
sigma = np.sqrt(sig_sqr)
expected_SNR = sigma / distance
# -- SNR
inv_win = (1.0 / window)
inv_win[:20*4096] = 0
inv_win[-20*4096:] = 0
rho = abs(z) / sigma * inv_win
# -- Output Results
# -- Plot rho as a function of time
plt.figure()
plt.plot(inj_time[::8]-time[0], rho[::8])
plt.xlabel("Seconds since GPS {0:.0f}".format(time[0]) )
plt.ylabel("SNR")
plt.title("Result")
plt.show()
#-- Find which time off-set gives maximum value of SNR
snr = rho.max()
found_time = inj_time[ np.where(rho == snr) ]

# -- Spectrograms
print "Plotting spectrogram..."
plt.figure()
NFFT = 1024
window = np.blackman(NFFT)
spec_power, freqs, bins, im = plt.specgram(strain_seg, NFFT=NFFT, Fs=fs,
                                    window=window)
plt.title("Spectrogram")
plt.show()
print "Normalized spectrogram..."
med_power = np.zeros(freqs.shape)
norm_spec_power = np.zeros(spec_power.shape)
index = 0
for row in spec_power:
    med_power[index] = np.median(row)
    norm_spec_power[index] = row / med_power[index]
    index += 1
plt.pcolormesh(bins, freqs, np.log10(norm_spec_power))
plt.title("Normalized Spectrogram")
plt.show()
# -- Report the results
print "\n  --- Printing Results ---"
print "Expected to find SNR {0:.1f}".format(expected_SNR)
print "Recovered SNR {0:.1f}".format(rho.max())
print "Recovered time GPS {0:.1f}".format( found_time[0] )
```

Appendix B

Python Script for the template [6]:

```
# Generate a template for detector strain
# as a function of frequency using eq. 3.4
```

```
# in Allen et. al. 2011 (title: FINDCHIRP)
# arXiv:gr-qc/0509116
#
# Adopted from a module
# by Ashley Disbrow 2013
import cmath
import numpy as np
import math
import matplotlib
import matplotlib.pyplot as plt
########################################
#Define all functions which generate template
########################################
#Calculate strain using frequency domain
def h(f, Amp, Psi, D_eff,imaginary):


    h_f = (1.0/D_eff)*Amp*f**(-7./6)*np.exp(-imaginary*Psi)
    return h_f
#Amplitude equation 3.4b
def Amp(ang_mom):
    G=6.67e-11 #Units: m^3/(kg*s^2)
    c=3e8 #m/s
    Mpc = 3e22 #m
    m_sun = 2e30 #kg
    c1=-math.sqrt(5./(24*math.pi)) #first term eq. 3.4b
    c2=(G*m_sun/(c**2*Mpc)) #unitless
    c3=(math.pi*G*m_sun/c**3)**(-1./6) #(T)**-(1/6)
    c4=ang_mom**(5./6) #unitless
    return c1*c2*c3*c4 #with units of (time)**-(1/6)
#eq. 3.4c
def Psi(f,eta,Mtot):
    G=6.67e-11*2e30 #Units: m^3/(M_sun*s^2)
    c=3e8
    v = (G*Mtot*math.pi*f/c**3)**(1./3) #unitless
    t1=(3715./756+55.*eta/9)
    t2=15293365./508032+27145.*eta/504+3085.*eta**2/72
    t_0 = 0 #assume time is negative until coalescence
    phi_0 = 0 #equal to Phi_c, assuming i=1 and F_cross=0
    Psi = 2*math.pi*f*t_0-2*phi_0-math.pi/4+(3./
(128*eta))*(v**(-5)+t1*v**(-3)-16*math.pi*v**(-2)+t2/v)
    return Psi
########################################
#Main - Give parameters and generate template
########################################
def createTemplate(fs,dataChunk, m1, m2):
    #Definitions necessary for math
    Deff = 1. #Mpc
    j = cmath.sqrt(-1) #define the imaginary variable
    dt=dataChunk/2
    Mtot=m1+m2
    eta = float(m1*m2)/(Mtot**2)
    ang_mom=eta**(3./5)*Mtot
    # Create array of frequencies
    nyquist = fs/2
    f_i = 1./(2*dt)
    # f_i = 25
    frequency = np.arange(0,nyquist+1./(2*dt),1./(2*dt))
    frequency[0] = 1./(2*dt)
    #use frequencies to find strain:
```

```
    amplitude = Amp(ang_mom)
    psi_vector = Psi(frequency,eta,Mtot)
    strain = h(frequency,amplitude,psi_vector,Deff,j)
    # The template should stop at f_isco, the innermost
    # stable circular orbit.
    c = 3e8
G = 6.67e-11
m_sun = 2e30 #kg
f_isco = c**3/(6*math.sqrt(6)*math.pi*G*m_sun*Mtot)
strain[f_isco<frequency]=0
return strain,frequency
```