Interim Report 2: Using adaptive filtering to track noise lines or signals with varying frequency and amplitude

Rebecca White
Dornsife College of Letters, Arts and Sciences, University of Southern California, Los Angeles, California 90089, USA

Mentor: Ling Sun
LIGO Laboratory, California Institute of Technology, Pasadena, California 91125, USA and
OzGrav-ANU, Centre for Gravitational Astrophysics, College of Science,
The Australian National University, ACT 2601, Australia
(Dated: November 2, 2020)

Advanced Laser Interferometer Gravitational-Wave Observatory (LIGO) and Virgo have observed gravitational wave (GW) signals from compact binary coalescences. Yet undetected GWs from other types of sources, including quasimonochromatic continuous waves (CW) from individual spinning neutron stars and long-transient signals from newly born neutron stars, are of interest. One of the issues with the LIGO data that plays a role in making it challenging to detect these narrow-band GW signals is the instrumental spectral lines, which could obscure astrophysical signals at the frequencies where they occur. We use an adaptive filter, named "iWave", to track, study, and remove these instrumental artifacts from the interferometric data. Further, we explore the capability of iWave to track and detect weak CW or long-transient GW signals and quantify the sensitivity. This new tracking method, operating on time-series data, provides an efficient alternative to existing frequency-domain matched filter search methods. We discuss its application to low-latency follow-ups of binary neutron star postmerger remnants.

I. INTRODUCTION

On September 14th, 2015, Advanced Laser Interferometer Gravitational-Wave Observatory (LIGO) detected its first gravitational wave (GW) event, GW150914 [3]. This event was detected in the first observing run (O1) of Advanced LIGO (from September 12th, 2015 to January 19th, 2016). After O1 concluded, the second observing run (O2) began on November 30th, 2016 and ended on August 25th, 2017 [5]. During the O2 run the binary neutron star (BNS) merger event, GW170817, was detected [4]. The third observing run (O3) had three confirmed mergers – GW190412 [6], GW190425 [7], and GW190814 [8]. Throughout the entire O1–O3 runs, there have been 14 confirmed GW events, most of them binary black hole (BBH) mergers.

The raw data from the LIGO detectors contain noise and glitches, so one of the most important aspects of LIGO data analysis is detector characterization and noise modeling. Without being able to monitor the detector or track the noise accurately, the significance of an event could be incorrectly estimated. A significant part of LIGO noise is instrumental lines, which can obscure astrophysical signals at the frequencies where they occur. They cannot be factored out easily and some are not very well understood. Some may even have wandering frequencies and/or varying amplitudes, which makes them even more difficult to track [2].

Adaptive filtering is a dynamic approach for characterizing the features in input data, including noise lines or signals with wandering frequencies and amplitudes. This method is very helpful when the input signal changes over a period of time. It could prove help in analyzing the detector interferometric data, tracking the varying instrumental lines over the run. A phase locked loop (PLL) is a control system that uses an oscillator to produce an output signal at a frequency and phase synchronized with the input signal. iWave is a hybrid method of a traditional PLL and an adaptive filter that is used for line tracking. This allows us to combine the wandering frequency/amplitude aspects of adaptive filters with the oscillation frequencies of PLLs to help us better track these spectral lines over a period of time.

There are other types of GWs in addition to those from compact binary coalescences (CBCs). Continuous waves (CWs) are produced by a single spinning neutron star. There are also long-transient GW signals that can be produced by a post-merger remnant from a BNS merger. Burst GWs are not yet very well understood since they are difficult to model. Once detected, however, they will be able to reveal a significant amount of astrophysical information. Stochastic background GWs from the early evolution of the universe are another type of weak signals remaining to be detected [1]. Existing techniques, e.g., the hidden Markov model tracking, can also track wandering signals and has been used for tracking both instrumental lines as well as CW and long-transient GWs [9, 11, 12]). Such methods mainly work in the frequency domain while iWave could prove to be an alternative method for analyzing the data in time domain.

In this project, we aim at using iWave to successfully track weak synthetic long-transient GW signals and investigating if iWave can track and remove spectral noise lines and hence clean the data.

II. METHODS

A. iWave and PLLs

Traditional PLLs are a loop containing a voltage-controlled oscillator (VCO), loop filter (LF), and phase detector (PD). The VCO is an oscillator set to a frequency ω which controls the signal. The VCO input is set by the output of the PD since the PD determines the phase difference between the oscillating input and the VCO output. The LF controls the bandwidth of the PLL. However, there are two relevant limitations when using PLLs – the input amplitude and the PD introducing an oscillating component. Traditional PLLs do not determine the input amplitude and the output of the PLL is not related to the amplitude of the input. This amplitude information is crucial for noise subtraction. The second flaw with using only a traditional PLL is that the PD introduces an oscillating component at 2ω. This needs to be filtered out before being fed into the VCO because if not, there will be sidebands on the output. Through filtering it out, a phase shift could occur which could cause more issues with the output.

iWave is a traditional PLL that replaces the reference oscillator with an adaptive filter, making it a hybrid of a traditional PLL and an adaptive filter. It utilizes a resonant orthogonal system generator to produce two oscillating outputs – d_{out} being in-phase with the input and q_{out} being 90 degrees out of phase (the quadrature phase). The bandwidth of iWave is set using single response time parameters so the phases will automatically line up. iWave is a technique for dynamically characterizing oscillations whose amplitude and frequency may vary. And unlike traditional PLLs, it can calculate the amplitude of the input wave using its two outputs through the equation $A_{\text{out}} = \sqrt{d_{\text{out}}^2 + q_{\text{out}}^2}$.

iWave is meant to characterize the oscillating input x_n through the output y_n. The iteration equation for the output of iWave is:

$$y_n = e^{-w}e^{i\Delta}y_{n-1} + (1 - e^{-w})x_n \tag{1}$$

Where $1/w$ is the number of samples corresponding to one e-folding relative to previous samples, Δ is the frequency in radians per sample, y_{n-1} is the previous output, and x_n is the input.

B. iWave Tests

To better understand iWave's outputs, a static test was done as seen in Figure 1. The red sine wave is the input signal. The blue wave is the q_{out}, which is the output of iWave phase shifted by $\pi/2$ to be the quadrature phase. The green line represents $A_{\text{out}} = \sqrt{d_{\text{out}}^2 + q_{\text{out}}^2}$ where d_{out} is the direct output of iWave and A_{out} is the amplitude of the output which exponentially increases until it reaches the input signal's amplitude. The black curve is the input signal subtracted by d_{out}, i.e., the error signal.

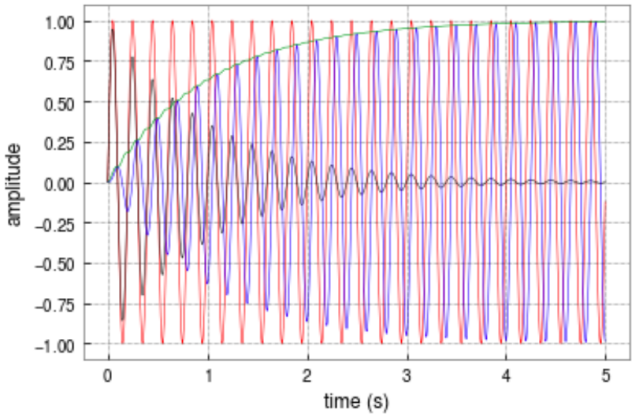

FIG. 1. iWave static test. The red sine wave is the input signal with a frequency of 5 Hz and amplitude of 1. The blue wave is the q_{out} signal. The green line is the amplitude of the output. The black wave is the error signal, which becomes less than 0.01 at about 4.62 seconds.

The two input parameters of iWave that have an effect on its ability to track lines are the initial guess of the signal frequency f_{guess} and the characteristic timescale of the signal τ. In order to better understand how iWave works in relation to these parameters, tests were done with different iWave f_{guess} and τ values in multiple different circumstances.

In order to test iWave's responses to different f_{guess} values, a few multi-line tracking tests were done. This test was done without noise and with lines at 30 Hz, 32 Hz, ad 35 Hz. As a starting point, Figure 2 sees a clear lock on because the f_{guess} values which iWave starts its search at are 35 Hz, 32 Hz, and 30 Hz which are the same as the frequencies injected that iWave should be tracking. However, if the f_{guess} values were changed to the point where two f_{guess} values were closer to a single line than either of the other lines, iWave would track the same line twice. This iWave duplication tracking can be clearly seen in Figure 3 and Figure 4. This implies that, when tracking noise lines close together, having the correct f_{guess} values is important to avoid duplicate tracking.

Another test that was done with iWave involved tracking a signal within noise. The main purpose of this test was to clearly see the effect of τ on iWave's ability to track this signal with varying frequency. The signal starts at 1000 Hz, so the f_{guess} value was 1000 Hz. τ values of 0.1 s, 0.5 s, 1.0 s, 1.5 s, and 2.0 s were all tried. In this case, a τ value of 1.5s yielded the smallest frequency-based root-mean-square error (rmse) once iWave locked onto the signal. Figure 5 contains the iWave output plots for this test with $\tau = 1.5$s. This test proved that, when tracking a signal, different iWave tau values should be tested to see which iWave tau value tracks the signal the best.

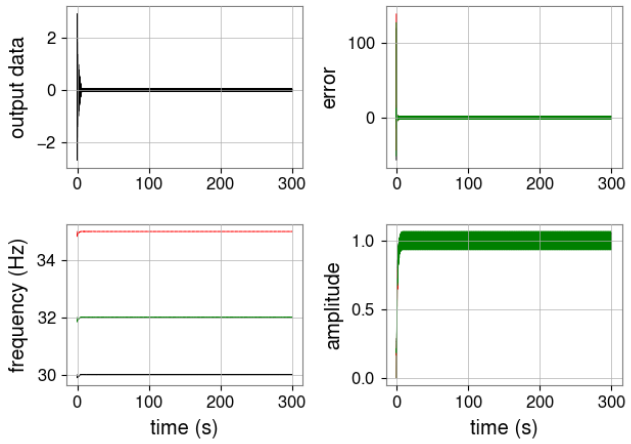

FIG. 2. iWave Multi-Line Test with Default f_{guess} and τ Values. In this, iWave f_{guess} values are set to the exact frequencies of each line (30 Hz, 32 Hz, and 35 Hz) and $\tau = 1$. The top left graph is iWave's output. The top right graph is iWave's error signal when it comes to tracking the 3 lines. The bottom left graph is the frequency throughout the output time series. The bottom right graph is the amplitude of the output data. As seen by the frequency and error graphs, iWave locks onto all three signals.

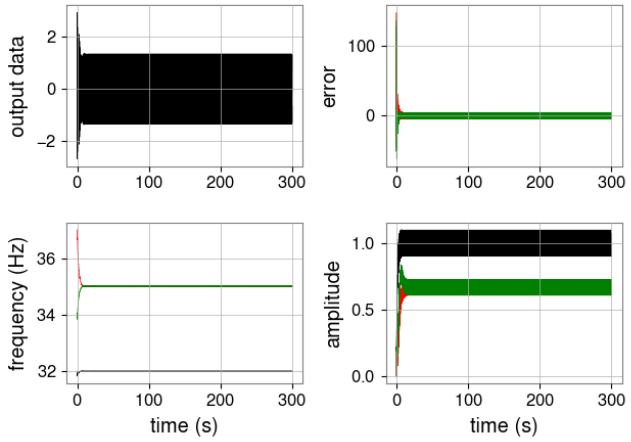

FIG. 3. iWave Multi-Line Test with f_{guess} Values Increased by 2 Hz and τ=1. The f_{guess} values are 32 Hz, 34 Hz, and 37 Hz, which is 2 Hz more than the actual frequencies of 30 Hz, 32 Hz, and 35 Hz. As seen in the bottom left frequency estimate plot, iWave only locks onto the 32 Hz and 35 Hz signals since those are the closest lines to the f_{guess} values despite this meaning that the 35 Hz line is tracked twice.

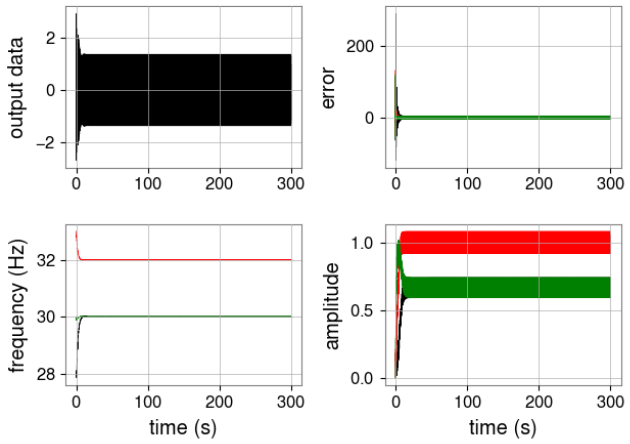

FIG. 4. iWave Multi-Line Test with f_{guess} Values Decreased by 2 Hz and τ=1. The f_{guess} values are 28 Hz, 30 Hz, and 33 Hz which is 2 Hz less than the actual frequencies of 30 Hz, 32 Hz, and 35 Hz. As seen in the bottom left frequency estimate plot, iWave only locks onto the 30 Hz and 32 Hz signals since those are the closest lines to the f_{guess} values despite this meaning that the 30 Hz line is tracked twice.

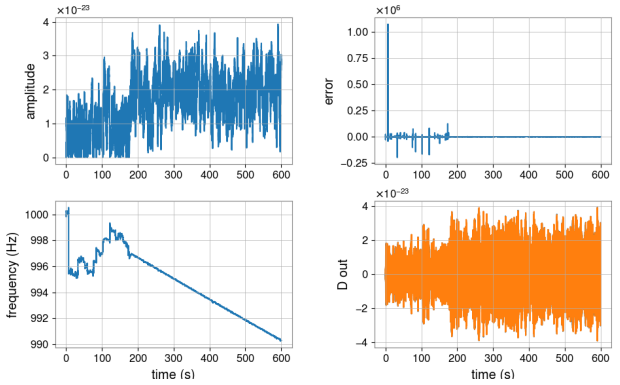

FIG. 5. iWave Outputs for Injected signal in Gaussian Noise when $\tau = 1.5$s and $f_{\text{guess}} = 1000$ Hz. As seen by the error and frequency graphs, iWave tracks the wandering line in the test frame GW data and can lock on to the signal between the 100 and 200 second marks. The calculated rmse is less than the rmse when $\tau = 1$s.

III. GOALS

We wish to determine how well iWave can track long-transient GW signals, specifically from post-merger remnant. Determining the sensitivity of iWave to weak synthetic CW or long-transient GW signals is a way to see how useful iWave would be in this application. In preparation of this goal, we see if iWave is able to track and remove spectral lines from the LIGO data as an alternative application.

IV. SIGNAL

The simulated signal used in our simulations models the gravitational-wave emission from a rapidly spinning-down millisecond magnetar born during a binary neutron star merger [10]. This magnetar model for long-transient

remnant has two main equations that determine how the signal looks. The frequency equation for this signal is:

$$f_{\rm gw}(t) = f_0(1 + \frac{t}{\tau_{\rm gw}})^{\frac{1}{1-n}} \quad (2)$$

where $\tau_{\rm gw}$ is the spindown timescale, f_0 is the initial GW frequency, and n is the breaking index. A braking index of n = 5 is preferred in our study because it simulates a signal whose energy loss is dominated by GW emission. The signal simulates a signal whose spindown is only being effected by the GW emission. The amplitude equation for this simulated GW signal is:

$$h_{\rm t} = \frac{4\pi^2 G}{c^4}\frac{I\epsilon}{d} f_{\rm gw}^2(t) \quad (3)$$

where G is Newton's gravitational constant, c is the speed of light, I is the principal moment of inertia in $\rm g\times cm^2$, ϵ is the ellipticity of the star, and d is the distance away the signal is in Mpc seconds.

V. SIMULATIONS

Given the signal discussed in Section IV, all of the following signals have the following parameters: f_0 = 2000Hz, $\tau_{\rm gw}$ = 10000s, n = 3, ϵ = 0.01, and $I = 1 \times 10^{45}$ g cm^2. They are also all set in Gaussian noise. The d parameter changes so that we can test iWave's signal sensitivity. Figure 6 and Figure 7 show the iWave output plotted on top of the simulated signal. These two figures have a distance of 1.0 Mpc seconds. As seen in Figure 7, the iWave amplitude estimation is far below what the signal's amplitude is. Despite attempts to fix this, the amplitude remains incorrect.

Because of the iWave amplitude estimation inaccuracy, we focused solely on iWave's ability to track the frequency of the line from different distances away. We tested d values from 1 Mpc to 10 Mpc. As seen in Figure 6, iWave's frequency estimation output nearly matches the frequency throughout the time series. This same matching is found until d is 5 Mpc where iWave only partially correctly tracks the signal frequency. Figure 8 is iWave's frequency estimation output over the raw signal when the distance parameter is 5 Mpc. Near the end of the time series, iWave loses the signal and then locks back on. However, after d = 5Mpc, iWave begins to completely lose the signal after a period of time. This is most evident when d is 10 Mpc as seen in iWave's frequency estimation output in Figure 9.

In order to quantify how accurate iWave is in its frequency tracking, Figure 10 is plotting the rms errors by the distances of the signal. The points used are from iWave tracking signals with distances of 1.0, 3.0, 5.0, 7.0, and 10.0 Mpc. The rms error values exponentially increase the larger the distance away of the signal is. For this signal with the parameters f_0 = 2000Hz, $\tau_{\rm gw}$ = 10000s, n = 3, ϵ = 0.01, and $I = 1 \times 10^{45}$g $\times$ cm^2, the rough distance limit for iWave seems to be 4 Mpc.

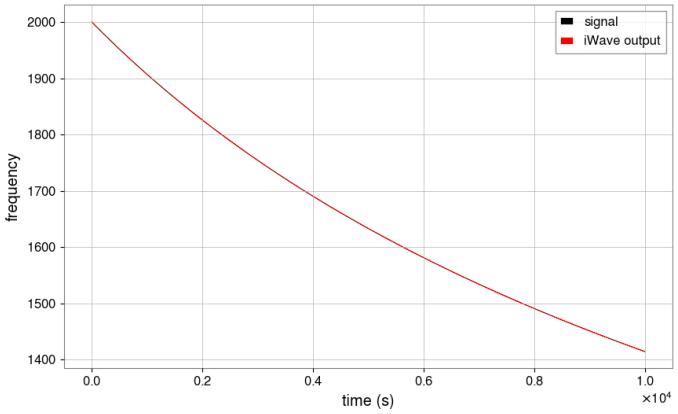

FIG. 6. Simulated Signal and iWave Output Frequency, Distance = 1 Mpc. The black line is the signal frequency over the time series while the red line is the iWave frequency estimation output for this signal. iWave seems to match the frequency throughout the entire time series. The distance value in this graph is 1.0 Mpc.

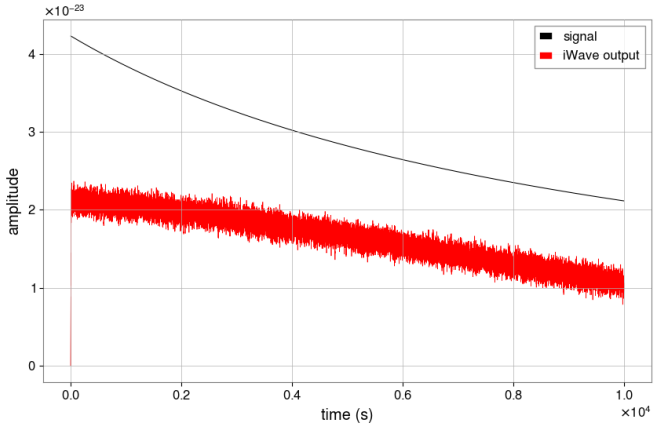

FIG. 7. Simulated Signal and iWave Output Amplitude. The black line is the signal amplitude over the time series while the red line is the iWave amplitude estimation output for this signal. The iWave amplitude estimation error can be clearly seen here and currently has no fix. The distance value in this graph is 1.0 Mpc.

VI. IWAVE LINE TRACKING/REMOVING

1. *Single-Line Noise Tracking*

We had iWave track the 120 Hz power main line. Figure 11 shows the Livingston raw data before filtering on the left as well as the raw data from 119 Hz to 121 Hz

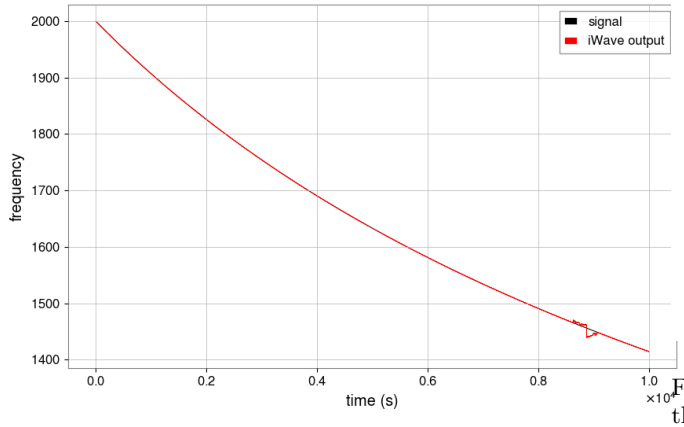

FIG. 8. Simulated Signal and iWave Output Frequency, Distance = 5 Mpc. The black line is the signal frequency over the time series while the red line is the iWave frequency estimation output for this signal. iWave seems to match the frequency well throughout most of the time series, only significantly losing the signal near the end. The distance value in this graph is 5.0 Mpc. This also implies that 4.0 Mpc is the rough limit for iWave's accuracy where it tracks the whole frequency without small deviations.

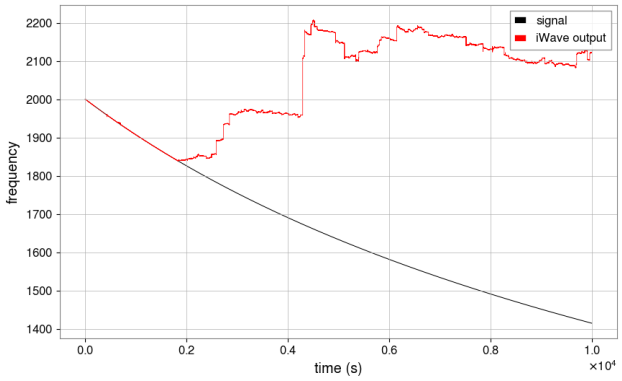

FIG. 9. Simulated Signal and iWave Output Frequency, Distance = 10 Mpc. The black line is the signal frequency over the time series while the red line is the iWave frequency estimation output for this signal. iWave is locked onto the signal at the beginning, however loses the signal before 200 seconds (out of the 1000 second signal) and never locks back onto the signal.

since that is the frequency band we are focusing on. In Figure 12, the filtered data is plotted in red on top of the raw data to show how narrow a frequency band we are taking in order to track this line. Filtering the data limits iWave to focusing on the 119 Hz to 121 Hz frequency band so it does not lock onto a different, louder line in the data. However, this filter tamps down the power of the whole data set. Figure 13 is the iWave output. The 120 Hz power line can change slightly in frequency over

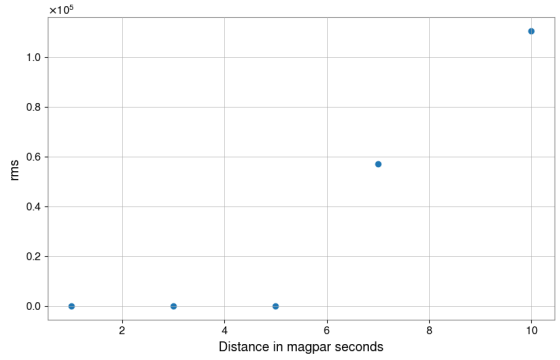

FIG. 10. RMS Error Over Distances of Signal. This plot is of the rms error values over the distance away the signal is. The points used are from iWave tracking signals with distances of 1.0, 3.0, 5.0, 7.0, and 10.0 Mpc.

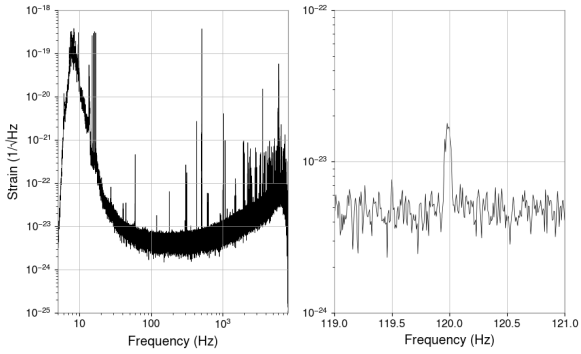

FIG. 11. Livingston Raw Data 119-121Hz. The left graph is of the raw data spanning the whole frequency series. However, the right graph is that data only focused on the 119 Hz to 121 Hz frequency band since that is where the 120 Hz line lies. This 120 Hz line is clearly visible as the peak in the data around 2×10^{-23}.

the time period, so the wandering frequency result seems normal. However, the amplitude estimate is inaccurate – being smaller than the raw/filtered data by an order of magnitude. This problem is currently under investigation.

2. *Multi-Line Tracking*

We had iWave track two violin mode lines in interferometric data. Figure 14 shows the Livingston raw data pre-filtering on the left as well as the raw data from 1012 Hz to 1013 Hz since that is the area we are focusing on. The filtered data compared to the raw data is in Figure 15. Lastly, the iWave output is Figure 16. The frequency estimates for the middle two lines (the two violin mode lines we are tracking) are locked on and do not change much which is to be expected. However, the

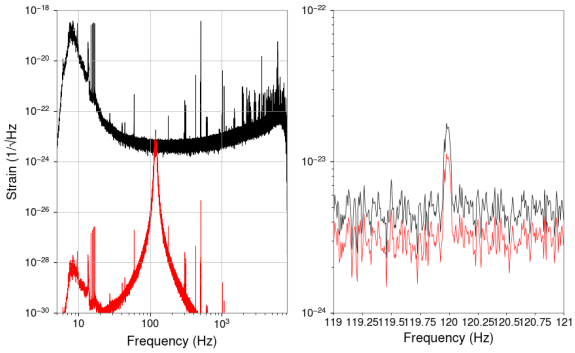

FIG. 12. Comparing the Filtered Data to the Raw Data. The left graph shows the filtered data in comparison to the raw data over the whole data set. The bandpass filter data focuses on the range from 115 Hz to 125 Hz. This way, most of the frequencies we do not need are filtered out. Using this filtered data, we can just focus on the 120 Hz power main line that we want. However, as the right graph shows, this filter also results in a loss in power compared to the raw data where the peak of the 120 Hz power line is only around 1×10^{-23} instead of the raw data around 2×10^{-23}.

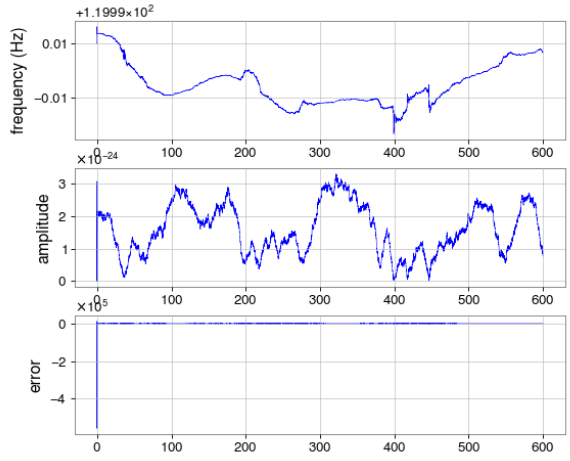

FIG. 13. iWave Output 120Hz Power Main Line. The top graph is of iWave's frequency estimate - since the 120Hz line is known to wander, this graph may be accurate. The amplitude estimate (middle plot) for this line is on the 10^{-24} order of magnitude, which is one order of magnitude less than the peak seen in the raw or filtered data. The last graph is the error signal iWave internally generates. All of these graphs are over the time series domain.

amplitude estimates are inaccurate by an order of magnitude.

While the iWave frequency output seems to lock onto the noise lines as seen in Figure 17, the amplitude estimate is off. At the highest point, the amplitude estimate for a line is around 1.5×10^{-22}, however looking back at the raw (and even filtered) data, the louder of the two lines peaks around 9×10^{-22}.

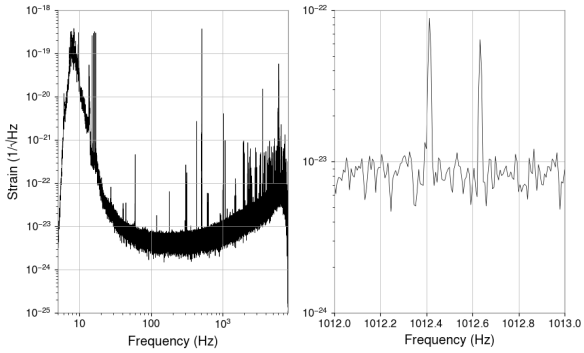

FIG. 14. Livingston Raw Data. This left graph is of the raw data spanning the whole frequency series. However, the right graph is only of the 1012 Hz to 1013 Hz frequency band since the two violin mode lines are in between 1012 Hz and 1013 Hz.

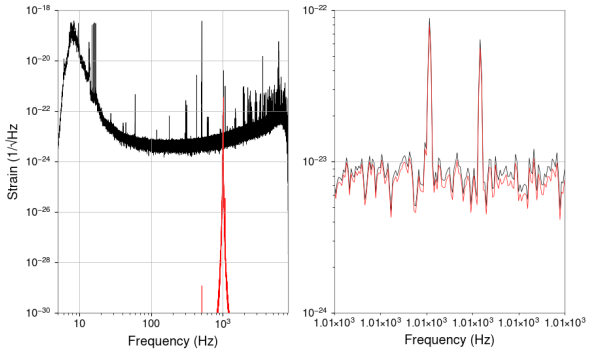

FIG. 15. Comparing the Filtered Data to the Raw Data. The left graph clearly shows how sharp the filter is in limiting the data to only the frequency band we want. The bandpass filter for the data focuses on the range from 1000 Hz to 1020 Hz. This way, most of the frequencies we do not need are filtered out. Using this filtered data, iWave can just focus on the 1012 Hz to 1013 Hz frequency band since the two violin mode lines are there. However, as the right graph shows, this filter also results in a loss of power compared to the raw data.

VII. CHALLENGES

The current challenge is the inaccuracies with iWave's amplitude estimation output. This challenge has impacted progress on Section VI the most since the iWave input needs to be subtracted from the raw/filtered data to clean the noise lines. iWave's amplitude estimate is an order of magnitude smaller than the data's peak. When subtracting the iWave output results of Section VI 1 (Figure 13) from the data, there is barely a difference in the peak at 120 Hz as seen in Figure 19 and Figure 20. Figure 18 is the subtraction of the iWave output for the Section VI 2 violin mode line tracking (Figure 16) from the raw data. There is barely a change in the peaks be-

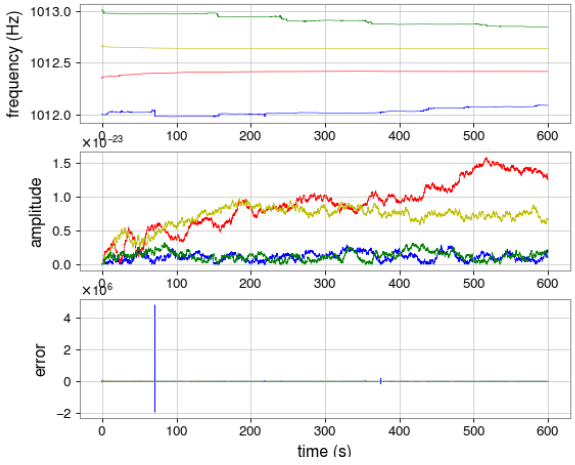

FIG. 16. iWave Output. Throughout the time series, the iWave trackers stay relatively constant in terms of the frequency estimates as seen in the top graph. The two middle lines are the violin mode lines we are trying to track and iWave locks on to them both. The amplitude estimates for the trackers (seen in the middle graph) seem to continuously increase. However, the greatest amplitude is around 1.5×10^{-22}, which is about an order of magnitude smaller than the power of the lines in the raw/filtered data. The error signal throughout the entire time series seems larger than ideal, however no rms has been calculated for this test.

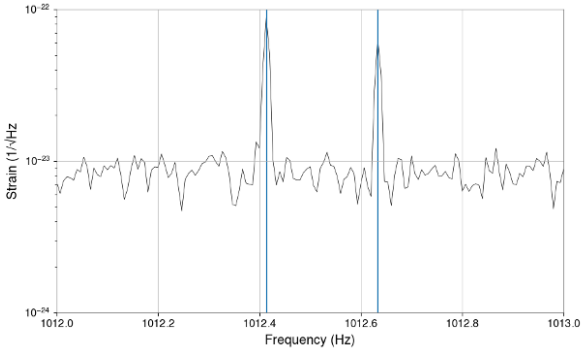

FIG. 17. Checking Frequencies. The two overlaid blue lines are vertical lines of the values from iWave's frequency output once it locks onto the two violin mode lines. The black line is the raw data. As shown in this figure, iWave's frequency estimates match up with the two peaks (the two violin mode lines) in the raw data.

cause of this order of magnitude difference.

Scaling up the input to iWave and re-scaling the output was one solution that was tried, however, this solution was unsuccessful in improving the amplitude estimation.

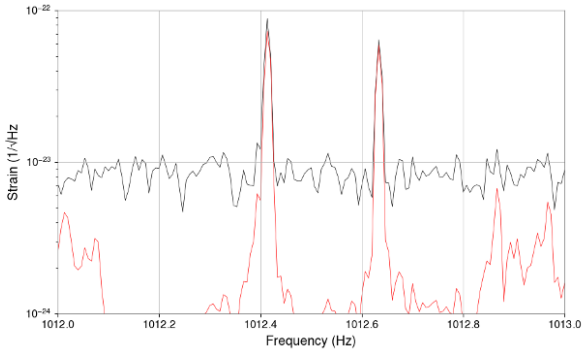

FIG. 18. Subtracting iWave Output from Raw Data. In this graph, the black line is the raw data and the red line is $data - d_{out}$. While tracking these two lines, the iWave output does not just contain the two lines, however also seems to contain all of the noise. This leads to more data being subtracted than what we want. Another problem is that the red line is not much different amplitude-wise at the two lines than the raw data. This is because of iWave estimating the amplitude being 1.5×10^{-22} instead of the 9×10^{-22} that the amplitude is around.

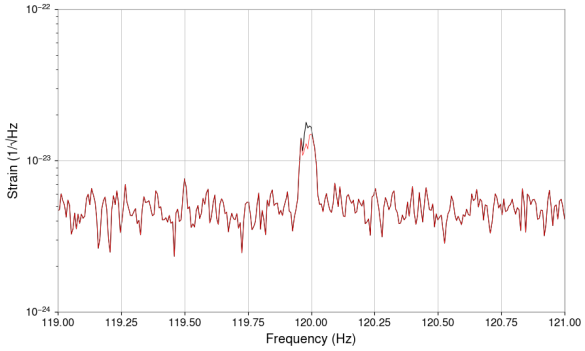

FIG. 19. Subtracting iWave Output from Raw Data. In this graph, the black line is the raw data and the red line is $data - d_{out}$. The iWave output only contains the line tracked which is why it is the only place where the red line is noticeably different than the black line. However, because of the amplitude estimation problem, the line is not cleaned enough.

VIII. DISCUSSION/NEXT STEPS

Based on the simulations in Section V, we can track simulated signals in Gaussian noise to around the same level as other currently used methods for tracking post-merger remnant. However, this is only a preliminary search with a set $\tau_{\rm gw}$ value, set timescale, and having the signal only be in Gaussian noise. To test the limits of iWave and how it performs tracking longer/shorter signals, signals with different signal $\tau_{\rm gw}$ and timescale values should be tested. Doing a comprehensive study on iWave's sensitivity to different types of GW signals and which signals it tracks better would help when defining

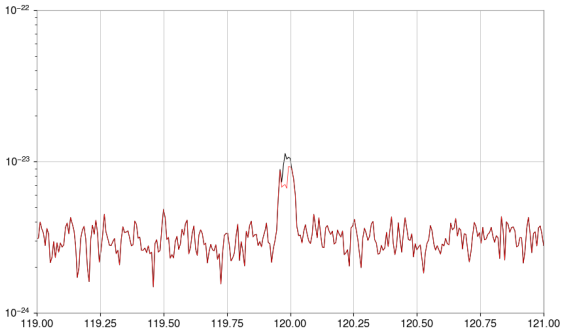

FIG. 20. Subtracting iWave Output from Filtered Data. This was mainly done to see if there was a difference between subtracting the output from the filtered data as compared to the raw data in case of any phase shifts introduced by the filter. There is not much of a difference between this figure and Figure 19 besides the overall power being less in this figure because of the decrease in power introduced by the filter.

the versatility of iWave. Also, because of the preference of a breaking index that simulates a signal whose energy loss is dominated by GW emission, the breaking index should be set to 5 instead of 3.

There are many studies about GW170817, however there does not exist any current methods that can confidently detect postmerger remnant from the distance that GW170817 happened at. iWave is no different, having its limit be around 7.0 Mpc as per Figure 10 while GW170817 is at 40.0 Mpc. However, this tool could prove useful in future BNS merger detections for understanding the postmerger remnant. We are currently only testing iWave with one detector at a time, so we hope that iWave's sensitivity will improve with inputs from both detectors.

Once iWave's amplitude estimation output is fixed, I would like to track and remove some more noise lines featured on the list of spectral lines in each detector over each observing run. Through tracking and hopefully successfully canceling out these lines, we will be able to tell which types of lines iWave is best at cleaning within the LIGO data.

The ultimate goal of this project is to implement iWave into the cluster to be a noise line filtering and signal tracking software for LIGO data. We are currently only testing iWave with one detector at a time, so we hope that iWave's sensitivity will improve with inputs from both detectors.

[1] Sources and types of gravitational waves.

[2] B P Abbott et al. A guide to LIGO–virgo detector noise and extraction of transient gravitational-wave signals. *Classical and Quantum Gravity*, 37(5):055002, feb 2020.

[3] B P Abbott and others. Observation of gravitational waves from a binary black hole merger. *Phys. Rev. Lett.*, 116:061102, Feb 2016.

[4] B P Abbott and others. Gw170817: Observation of gravitational waves from a binary neutron star inspiral. *Phys. Rev. Lett.*, 119:161101, Oct 2017.

[5] B P Abbott and others. Gwtc-1: A gravitational-wave transient catalog of compact binary mergers observed by ligo and virgo during the first and second observing runs. *Phys. Rev. X*, 9:031040, Sep 2019.

[6] B P Abbott and others. Gw190412: Observation of a binary-black-hole coalescence with asymmetric masses, 2020.

[7] B P Abbott and others. GW190425: Observation of a compact binary coalescence with total mass $\sim$ 3.4 m $\odot$. *The Astrophysical Journal*, 892(1):L3, mar 2020.

[8] B P Abbott and others. GW190814: Gravitational waves from the coalescence of a 23 solar mass black hole with a 2.6 solar mass compact object. *The Astrophysical Journal*, 896(2):L44, jun 2020.

[9] Joe Bayley, Chris Messenger, and Graham Woan. Generalized application of the viterbi algorithm to searches for continuous gravitational-wave signals. *Phys. Rev. D*, 100:023006, Jul 2019.

[10] Paul D. Lasky, Cristiano Leris, Antonia Rowlinson, and Kostas Glampedakis. The Braking Index of Millisecond Magnetars. *The Astrophysical Journal Letters*, jul 2017.

[11] Ling Sun and Andrew Melatos. Application of hidden markov model tracking to the search for long-duration transient gravitational waves from the remnant of the binary neutron star merger gw170817. *Phys. Rev. D*, 99:123003, Jun 2019.

[12] S. Suvorova, L. Sun, A. Melatos, W. Moran, and R. J. Evans. Hidden markov model tracking of continuous gravitational waves from a neutron star with wandering spin. *Phys. Rev. D*, 93:123009, Jun 2016.