SNR of Massive Black Hole Binaries in LISA: Test of LISACode and PhenomC

Mauricio Diaz-Ortiz Jr.*
University of Florida Department of Physics
Max Planck Institute for Gravitational Physics
Leibniz Universität Hannover Institute for Gravitational Physics
(Dated: August 1, 2017)

With the selection of LISA as ESAs L3 mission, tests of current analysis tools is necessary as the consortium beings to move towards engineering readiness. Two packages, LISACode and PhenomC, are tested with respects to their SNR calculators. A MCMC was preformed with LISACode as the waveforms used within it are sky position dependent. The average of the SNR values from the MCMC were taken and compared with L3 mission requirements and the sky-averaged SNR values produced by PhenomC.

1. INTRODUCTION: GRAVITIATIONAL WAVES

A new age of astrophysics was ushered in with the direct detection of a gravitational wave in 2015 by the Laser Interferometer Gravitational Wave Observatory, LIGO. This instrument provides a means to probe portions of space which appear dark to us, due to limitations of electromagnetic observations, through the observation of slight length changes associated with the passage of a gravitational wave. These waves were predicted by Einstein's Theory of General Relativity which developed the geometric notion of spacetime by combining the three physical dimensions and time. Spacetime acts like a sheet that is curved by the presence of mass and energy. As mass is accelerated, like non-symmetrical rotation or two objects in-spiraling towards each other, quadrupole gravitational radiation is emitted perturbing spacetime - analogous to ripples in water. As these ripples propagate they stretch and squeeze their medium orthogonal to their direction of travel. The change in length, strain, that is a result of the compression and expansion of spacetime is incredibly small requiring rather sensitive instrumentation for detection, such as laser interferometers.

1.1. Laser Interferometers

A laser interferometer operates by interfering two sources of light to produce an interference pattern which can then be analyzed. Currently, the Michelson setup is used, which allows for the conversion of distance changes within its arms to measurable phase changes at a photodetector. In a Michelson interferometer, a laser is propagated to a beam splitter which splits the beam into two nonparallel paths, referred to as the arms of the interferometer. The beam then travels the length of the arms to a mirror that directs the beam back towards the beam splitter where the two beams then interfere. As a gravitational wave passes through the detector the length of one arm is lengthened while the other is shortened resulting in a change in the expected signal that is read at the photodetector. The LIGO detectors are able to detect gravitational waves in the frequency band of 10Hz to 10Khz, which is a culmination of the arm lengths as well as sources of noise. Gravity gradient noise, the result of non-uniformity in the local gravitational field, and seismic noise prevent these detectors from detecting any signaal below 10Hz. However, below this band lies sources rich in Astrophysical information such as white dwarf binaries, neutron star binaries, stellar origin black holes, these sources in various combinations, and more [1]. As a means of combating this hard limit space-based interferometers have been proposed such as the Laser Interferometer Space Antenna, LISA.

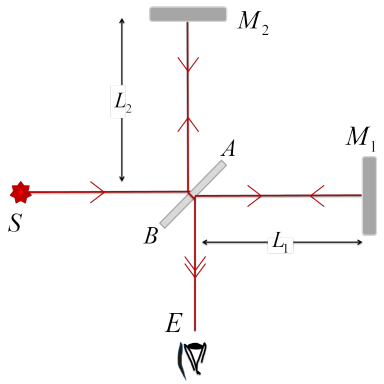

FIG. 1: Basic Michelson interferometer, where the eye in this diagram represents the photodetector.

1.2. Space Based Detector: LISA

The LISA mission will be comprised of three spacecraft that form an equilateral triangle where the vertices are said spacecraft, with sides 2.5Gm in length. These spacecraft will exchange laser light in order to detect, as with the land based detector, a shift in the phase that is associated with the passage of a gravitational wave within the

*Electronic address: mdiazort@ufl.edu

$10^{-4} - 1$ Hz range. The entire constellation will be trailing the Earth by about 19-23 and has a proposed mission length of four years with the possibility of running for ten [1]. Like ground based detectors, LISA has its own issues

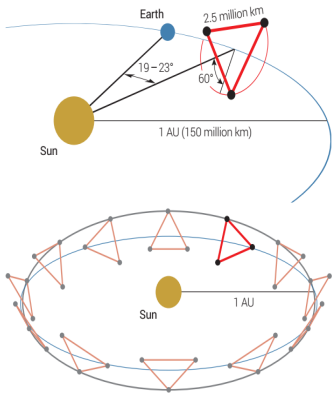

FIG. 2: LISA geometry and location relative to the Earth and Sun.

with noise, in particular laser phase noise. This noise is inherently suppressed in the ground based detectors, where arms lengths are equal resulting in the cancellation of the noise at the photodetector. However, it is virtually impossible to maintain equal arm length for a non-rigid space based detector that is trailing the earth. The use of heterodyne interferometry, mix frequency, and many compensatory systems are used but a post-processing algorithm called time-delay interferometry brings the noise down to acceptable levels[4].

1.3. Time-Delay Interfrometry

Time-delay interferometry, TDI, boils down to rather simple algebra. We start with signal received, as a data stream, in a single arm of LISA[2]

$$y_i(t) = C(t-2cL_i) - C(t) + h_i(t) + n_i(t), \ \ for\ i = 1, 2. \quad (1)$$

Where C is the laser phase noise, $h(t)$ is the gravitational wave singal, L_i is the arm length, and $n_i(t)$ is any other source of noise[tinto]. Now if we take the difference of the two arms,

$$\begin{aligned} y_1(t) - y_2(t) = C(t - 2L_1) - C(t - 2L_2) \\ + h_1(t) - h_2(t) + n_1(t) - n_2(t). \end{aligned} \quad (2)$$

we can see how the phase noise enters the stream. It is apparent that we can remove C with some finesse. So we can define the combination,

$$X(t) = [y_1(t) - y_2(t)] - [y_1(t - 2cL_2) - y_2(t - 2cL_1)], \quad (3)$$

and can see that the phase noise is removed from the combination of the data streams,

$$\begin{aligned} X(t) = h_1(t) - h_2(t) - h_1(t - 2cL_2) + h_2(t - 2cL_1) \\ + n_1(t) - n_1(t - 2cL_2) - n_2(t) - n_2(t - 2cL_1). \end{aligned} \quad (4)$$

There are other combiniations other than X, each of which represeting a different geometric path of the signal. However, as far as we are concerened X is sufficient for now. The two software packages that will be discussed later take advantage of this method.

2. SENSTITVITY CURVES

The performance of a detector is represented by a sensitivity curve, like the one shown in figure 3, which shows the output of instrument noise versus frequency, where a signal that is above this noise floor will be detectable. These curves take into consideration the total noise generated by subsystems, electronics, the environment, orbit, and more to produce a noise curve. A transfer function is then found by calculating the isotropic power averaged over the antenna pattern of the detector,as well as all propagation vectors and polarizations of a signal in the detector. For those who are curious rather detailed derivations can be found in [2] and [3] specifically for space based gravitational wave interferometers of equal and non-equal arm lengths. In figure 3 we can see the

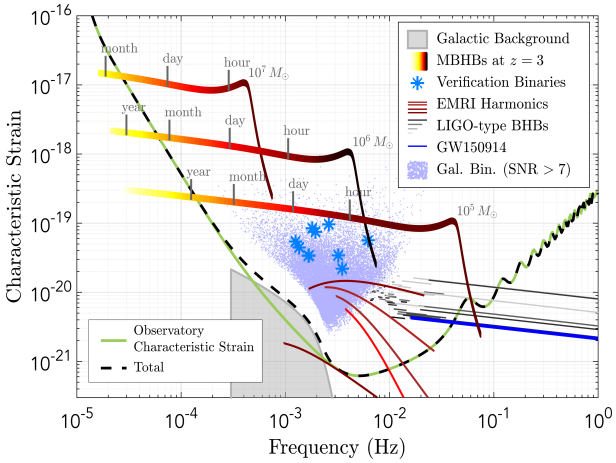

FIG. 3: Current sensitivity curve of LISA from the L3 mission proposal. This curve includes the signals that would be expected for certain sources.

expected signal from three massive black hole binaries at red shift 3, $z = 3$, with total masses of 10^5,10^6,and 10^7 $M_\odot$. Where we can see the characteristic increase of frequency and SNR of coalescing binaries. These masses will be important as they were used during our tests, as will be discussed in the procedure.

3. SIGNAL-TO-NOISE RATIO

The signal-to-noise ratio serves as a measure of detection confidence, a representation of how loud a signal is in the detector. The general expression for this quantity is given by

$$\rho^2 = 4\Re \int_0^\infty \frac{s^*(f)\hat{t}(f)}{S_n(f)} df, \tag{5}$$

Where $s(f)$ is the gravitational wave signal and $\hat{t}(f)$ is a template of said signal. To see thorough derivations of this equation see [5] or citemoorecurve. For our tests we are concerned with the sky-averaged SNR. In order to arrive at this we must average over the pattern function, inclination of the source with respects to the detector, and consider the angle between the interferometer arms. The expected sky averaged SNR for MBHB of mass ratio 0.2 of various red shifts are given by figure, these values serve as the expected values that the results of the test are compared against.

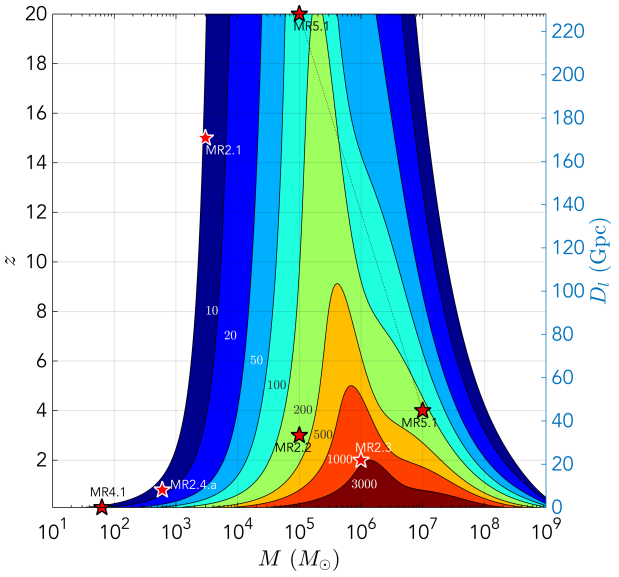

FIG. 4: Contour plot of constant SNR of MBHB with mass ratio 0.2 for various total masses versus red shift and luminosity distance.

3.1. Sky-Averaged SNR for Coalescing Binaries

As we are concerned with the SNR of coalescing MBHB let us derive the sky-averaged SNR for a coalescing binary like [11]. The restricted post-Newtonian wave form of a coalescing binary is given by

$$\tilde{h}(f) = \sqrt{\frac{5}{6}} \frac{\mathcal{M}^{\frac{5}{6}} f^{\frac{-7}{6}}}{2\pi^{\frac{2}{3}} D_L} e^{i\psi} Q, \tag{6}$$

where $\mathcal{M}$ is the chirp mass, D_L is the luminosity distance, and

$$Q = \frac{1+\cos^2(\iota)}{2} F_+ + i\cos(\iota) F_\times \tag{7}$$

Taking the angular average of the modulus of Q returns a factor of $\frac{2}{5}$, and then we multiply by $\frac{\sqrt{3}}{2}$ taking into account the 60° orientation of LISA's interferometer arms, thus equation 6 becomes

$$\tilde{h}(f) = \frac{\sqrt{10}}{20} \frac{\mathcal{M}^{\frac{5}{6}} f^{\frac{-7}{6}}}{2\pi^{\frac{2}{3}} D_L} e^{i\psi}. \tag{8}$$

In equation 5 $S_n(f)$ is non-sky-averaged, as such we use the relation[berti]

$$S_n^{NSA} = \frac{3}{20} S_n^{SA}. \tag{9}$$

Putting everything together our result is

$$\rho = \sqrt{\frac{2}{3}} \frac{\mathcal{M}^{\frac{5}{6}}}{\pi^{\frac{2}{3}} D_L} \left(\int_{f_{in}}^{f_{fin}} \frac{f^{\frac{-7}{3}}}{S_n^{SA}(f)} df\right)^{\frac{1}{2}}. \tag{10}$$

Equation 10 allows for a curve, that can be used as a benchmark of BH code ,to be generated such as figure 5 below. This curve will provides a means of checking how the software preforms with the physics of the event.

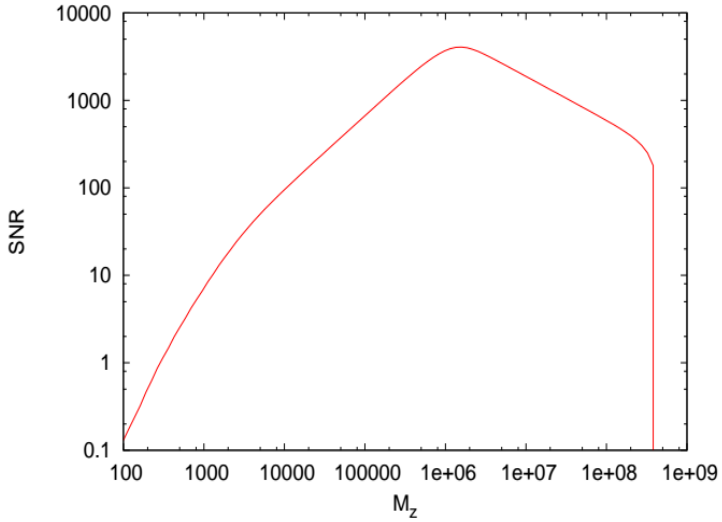

FIG. 5: This curve represents the SNR of various equal mass coalescing IMBHB MBHB using the classic LISA configuration, a similar curve will be built using data later.

4. PACKAGES

Three software packages were used during the performance analysis of LISACode and PhenomC. The two just mentioned and Docker.

4.1. Docker

Docker is a platform for deploying system images in order to reduce issues of cross platform compatibility. This is similar to GitHub in terms of functionality. It allows for one to upload a system image, referred to as a container, onto a server. From said server other users can pull the image and run it using the Docker interface, or through a terminal. The LISA data processing center maintains a container which is a system image of Ubuntu 14.04 LTS with LAL simulator, and LISACode installed.

4.2. LISACode

As there is no true simulator of LISA, at the engineering level, LISACode attempts to bridge the gap between the basic principles of LISA and a true end-to-end simulator [7]. This is done by considering as close to possible the sources that influence LISA's sensitivity as well as the TDI generators. The output of LISA is a time series of the phase meter which takes into consideration the constellation geometry, the response that is expected from the geometry, as well as internal and external noise sources. With the release of the L3 proposal the configuration files that reflect this proposal have been released for use within LISACode. The sensitivity of this configuration file can be seen in figure 7. LISACode is a C++ application, but some python scripts are included to facilitate its use. One of which is makeTDI.lisacode2.py, which generates the TDI observerables and requires the designation of the configuration file mentioned previously. As well makeEOBNRForLC2.py generates an effective one body Newtonian relativity waveform of the massive black hole binaries for injection into LISACode. This script requires the specification of the masses of the individual black holes, luminosity distance, coalescence time,observation time, and sky location. The coalescence time has a default value of $t_c = 604800s$ abd oberseva-

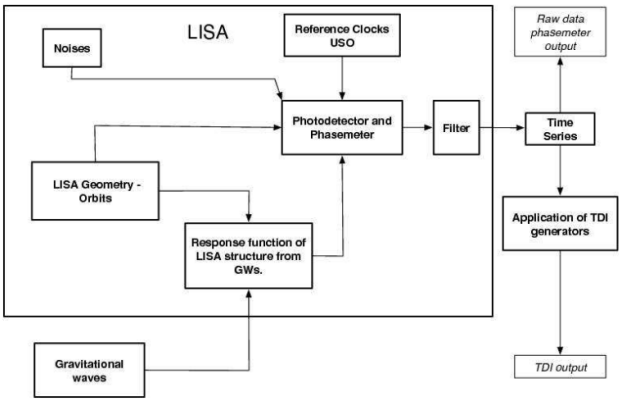

FIG. 6: LISACode block diagram, it is key to note that the waveforms are injected LISACode.

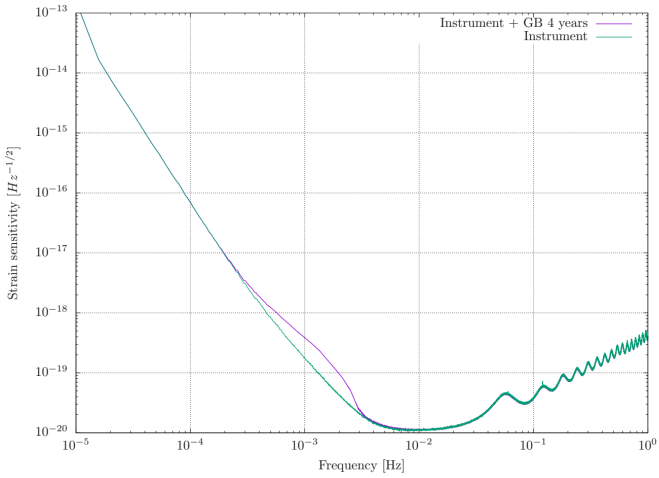

FIG. 7: L3 sensitivity curve that is used within LISACode currently, this curve shows the projected change in sensitivity with the inclusion of Galatic Binary noise.

tion time has a default value of $t_{obs} = 31557600s$,The sky location entails ecliptic latitude, ecliptic longitude, inclination, and polarization. It is with this script that we preform the MCMC.

4.3. PhenomC

PhenomC uses the phenomenological waveform of MBHB that is derived in [8]. The use of this program requires similar parameters as makeEOBNRforLC2, but it does not require the sky location. As well this can run locally, however does make use of some LISACode functionality. It is not included within the Docker image, as such one can either import it into the image with a locally mounted drive, or install LISACode.

5. PROCEDURE

The test of LISACode was done using a bash script that was included in a locally mounted drive with Docker. Along with this script, some updated version of makeEOBNR and makeTDI were included. The masses that were tested are as stated before 10^5, 10^6, and 10^7 $M_\odot$. However, these were done with a mass ratio of 0.2, to coincide with figure 3.The procedure of this test are as follows:

1. makeTDI was initiated, this was only done once as it was unnecessary to produce the TDI after each iteration, given that the geometry does not change.

2. The Monte Carlo was preformed by randomizing the sky position which was done through a python script that chose values from a particular set or from formula given by [9]

- Ecliptic latitude: $\sin(\beta)$ using $\beta \in (\frac{\pi}{2}, \frac{3\pi}{2})$
- Ecliptic longitude: $\lambda \in (0, 2\pi)$
- Orbital plane angles: $\theta_L \in (0, \pi)$ and $\phi_L \in (0, 2\pi)$
- inclination:

$$\cos(\iota) = \cos(\theta_L)\sin(\beta) + \sin(\theta_L)\cos(\beta) + cos(\phi_L - \lambda) \quad (11)$$

- Polarisation:

$$\psi = \arctan\left(\frac{\sin(\beta)\cos(\lambda - \phi_L) - \cos(\theta_L)\sin(\beta)}{\sin(\theta_L)\sin(\lambda - \phi_L)}\right) \quad (12)$$

3. With these randomized parameters the EOBNR waveforms were generated using them.
4. Using the EOBNR waveform and TDI the SNR was calculated used the LC2SNR function of LISACode.

This was done for 500 points, after all SNRs had been calculated the average and standard deviation of the set was taken using the numpy python package. The formula given in equations 11 and 12 can be found in [10] and [9]. The tests of PhenomC were far less involved as all that was required to be done was calculate the SNRs, which are already averaged, then compare them with the expected values from figure 3.

6. RESULTS

The results are split into three sections. In section i, results that were gathered while use an improper configuration file in Docker is presented first, this data was taken at red shift 3. Next, in section ii I will present results that used the proper L3 configuration files, that are more consistent with what was expected, taken at redshift 1. Finally, in section ii the results of PhenomC will be discussed.

6.1. Section I

We can see the results of the MCMC in the graphs of SNR versus sky location in figures 8, 9, and 10. As one may expect there is relatively little structure in the inclination and ecliptic latitude, as these are truly randomized and represent a slice of a 5D space. There are modulations that can be observed within the graphs of ecliptic longitude and polarization, but again these are expected as they are calculated using the other randomized parameters using sinusoidal functions.

Despite these expected behaviors one may notice that all the points, for the most part, are localized around an SNR of 2300 for all 500 points, which should not be the case. Looking at figure 3, at red shift 1 we should

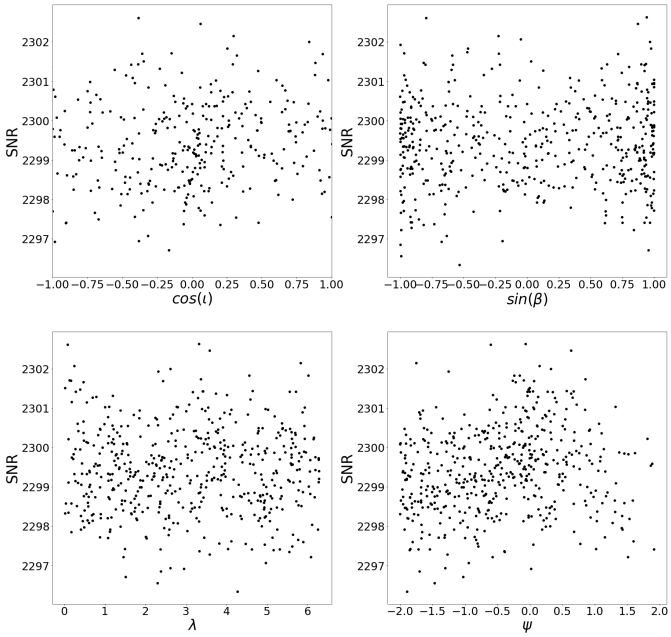

FIG. 8: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^5 M_\odot$

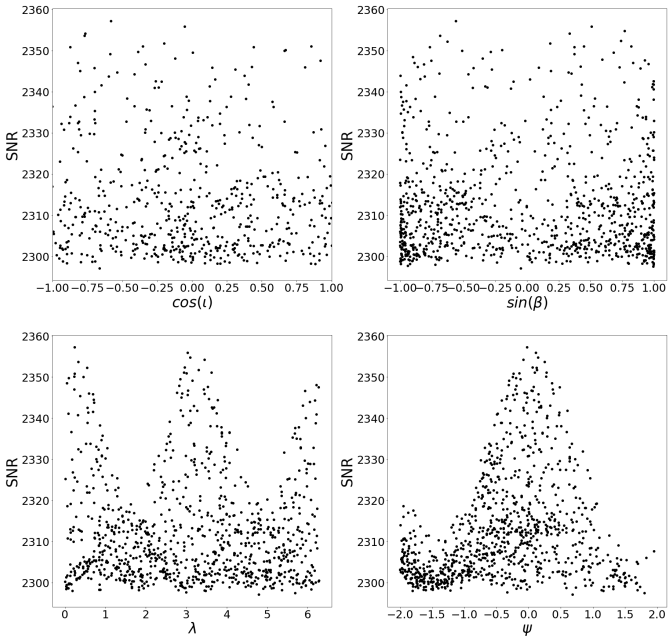

FIG. 9: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^6 M_\odot$, the modulation structure in λ and ψ are especially visible here.

expect the SNR from 10^5 $M_\odot$ to 10^6 $M_\odot$ to change by approximately 2000, but in both cases it remains around 2300. After some investigation the source of these results stems from a Docker side error that occurred when a recent image of the elisa/lisacode container had been updated. During the update Docker had thrown an error preventing the newest image from replacing the last.

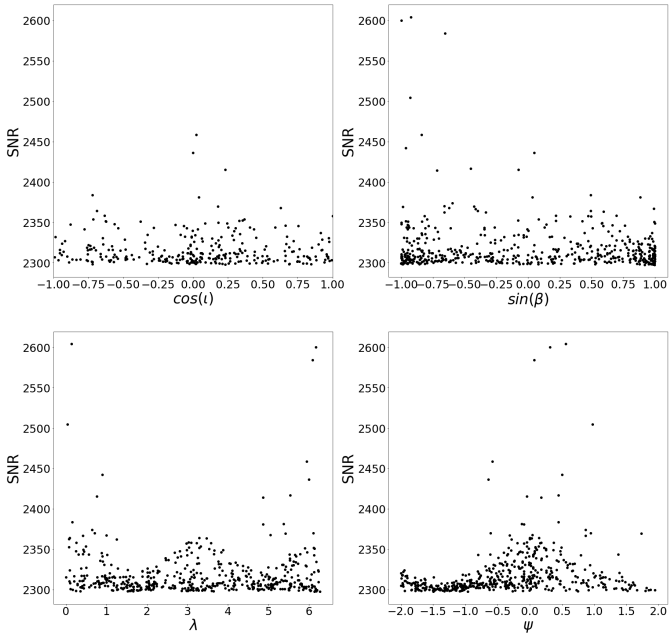

FIG. 10: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^7 M_\odot$

This resulted in all of the SNRs being computed using an eLISA configuration file from 2011. However, all of the LISACode files that are maintained on the GitHub are updated and the only differences between the pertinent files on the older image and the current are the configuration files, which are located in a sub directory, and the line within makeTDI that designates the instrument configuration that is being used. This issue was resolved by placing the necessary configuration files within the locally mounted directory along with the more up to date version of makeTDI and updating the bash script to place these files in there necessary locations. That being said, a point of interest may be despite the eLISA configuration being used while calculating the SNR values for MBHBs they do not coincide with the expected values for these sources as can be seen from figure 6.1. They are quite far off and may be indicative of potential issues with the code itself, as the change is geometry would shift the SNR values but the overall response should be comparable. Although, the results from section ii do not necessarily point to this possibility it may be worth future investigation.

6.2. Section II

For the secondary run of data collection the same overall structure of the graphs remain as mentioned previously; however, we see a large variability in the SNRs. The data was further plotted in a histogram to see the overall distribution, see figure 17. As can be seen the distribution of points is non-normal. This is due either to lack of data points or is related to how the SNR is calcu-

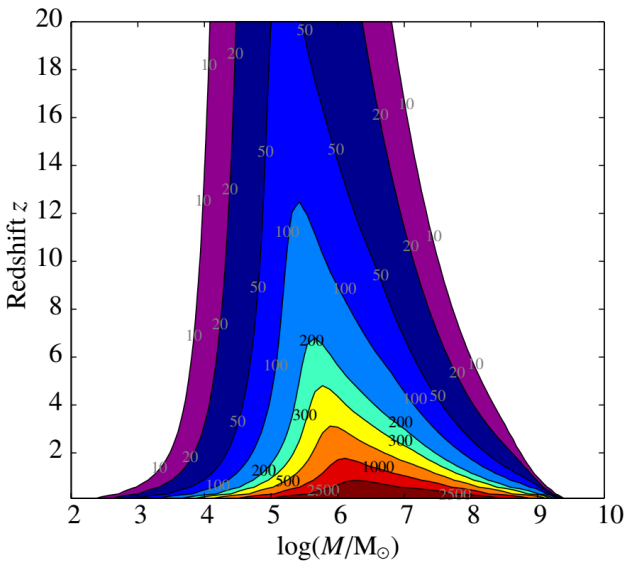

FIG. 11: Similar image to figure 3, however this represents the eLISA configuration.

lated. The SNRs that LISACode returns is the $\sqrt{\rho^2}$. In

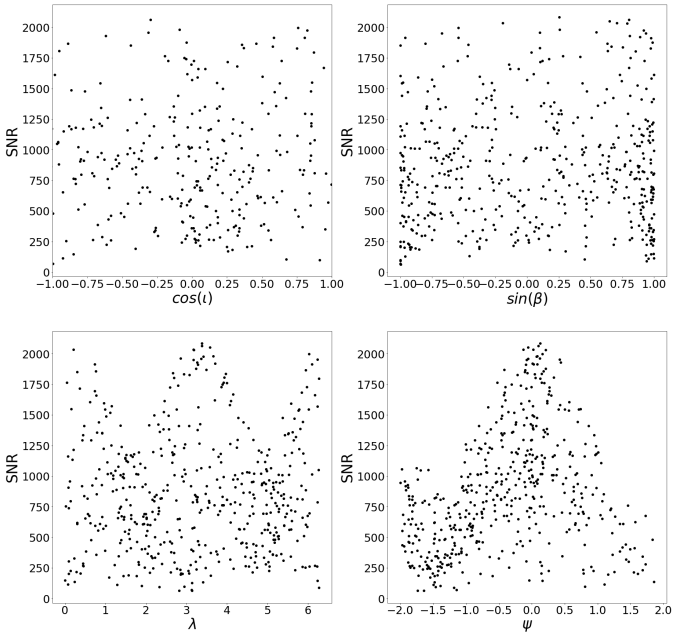

FIG. 12: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^5 M_\odot$

order to analyze the data which was right skewed I took the square root of the data set, which made it roughly normal then found the average and standard deviation using numpy. Using these values, I compared them with the square root of the expected values from figure 3. The results of this can be seen in the table below. Despite being within the expected range the standard deviation is quite high and will require further investigation.

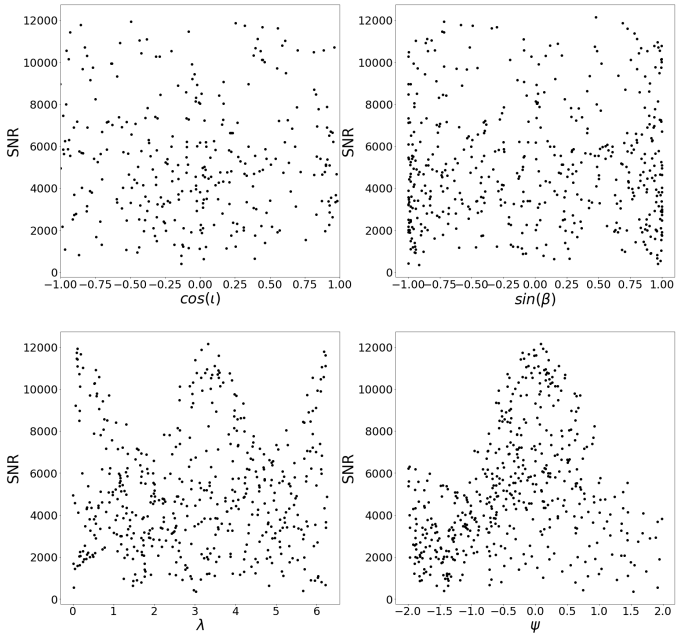

FIG. 13: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^6 M_\odot$

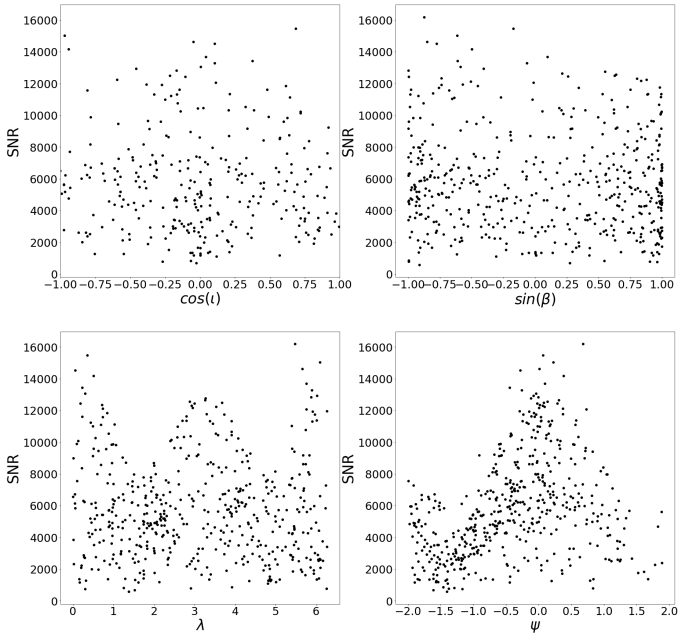

FIG. 14: SNR vs sky location for MBHB with mass ratio 0.2 and total mass $10^7 M_\odot$

$M_\odot$	*Expected SNR*	*Expected* $\sqrt{SNR}$	$\sqrt{SNR}$
10^5	1000	31.62	28.27±8.45
10^6	3000	54.77	68.37±20.40
10^7	3000	54.77	73.50±20.77

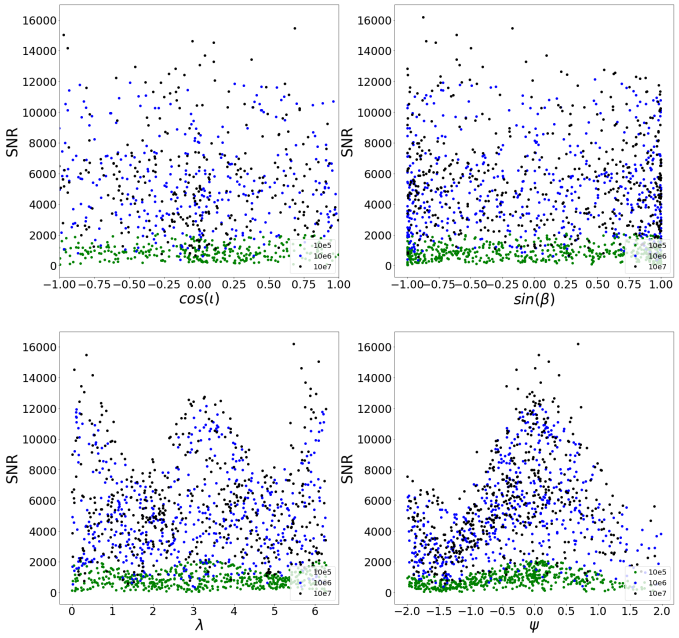

FIG. 15: Overplot of all data.

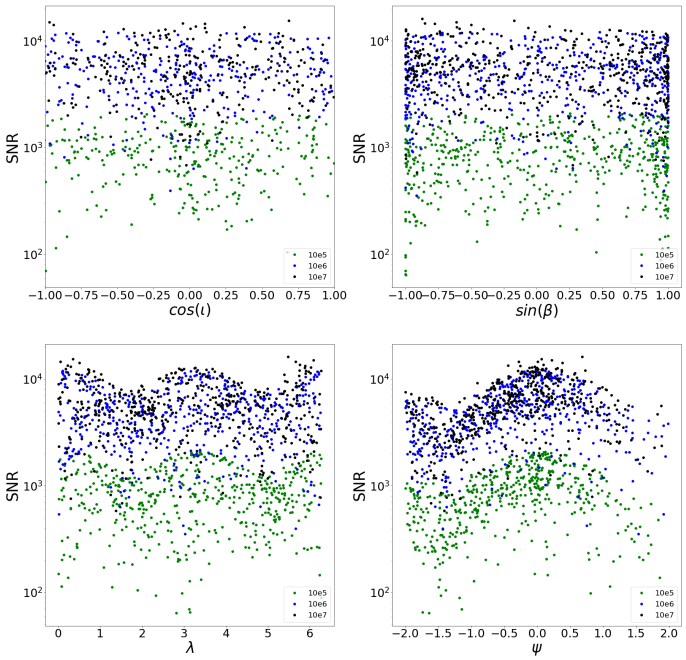

FIG. 16: Semilog overplot of all data.

6.3. Section III

As stated earlier in the procedure, SNR was calculated for the same total masses used in the LISACode test. However, some additional masses were tested as well, as the run time of PhenomC incredibly small in comparison to LISACode. Since it was unnecessary to define the sky location the parameters that needed to be accounted for, other than the masses of the binary system, were

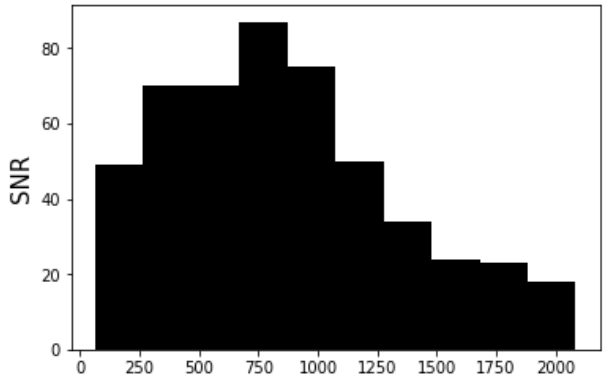

FIG. 17: Histogram of the SNR values for 10^5 $M_\odot$, the distribution in this case is not normal.

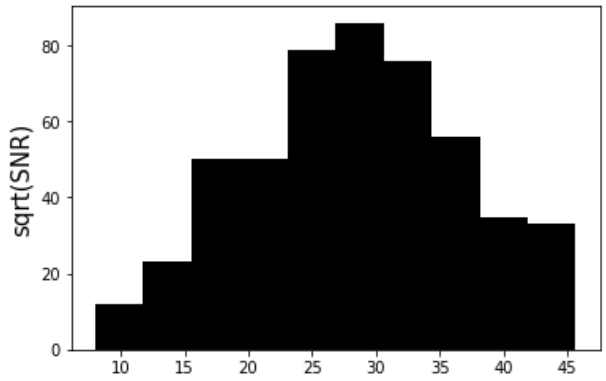

FIG. 18: Histogram of the SNR values after being square rooted for 10^5 $M_\odot$, this is still slightly skewed.

coalescence time, and observation time. These both were set to the default values for LISACode. The graph of the results from the TDI X and TDI A/E observables can be seem in figure 19. Looking at figure 5 we can see that the

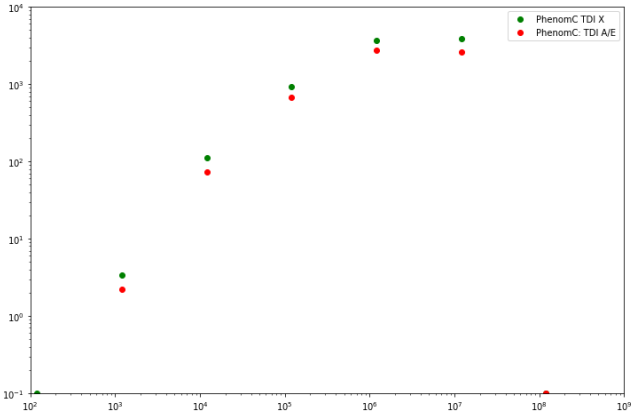

FIG. 19: Graph of PhenomC SNRs vs total mass, the overall behavior of the graph is comparable to figure 5.

behavior of L3 LISA with respects to MBHBs is what is expected. A key point to notice is that the overall SNR of the TDI A/E is higher than those of TDI X, this is due to the number of links associated with both. X has four, while A/E has six resulting in a higher overall sensitivity. Now we can square root the SNRs for 10^5, 10^6, and 10^7 $M_\odot$ then compare them with the values of LISACode.

$M_\odot$	$\sqrt{SNR_{Ex}}$	$\sqrt{SNR_{LC}}$	$\sqrt{SNR_{PC}}$
10^5	31.62	28.27±8.45	25.8
10^6	54.77	68.37±20.40	52.5
10^7	54.77	73.50±20.77	51.16

7. FUTURE WORK

From these results it can be seen that a lot of work still needs to be done in understanding the performance of LISACode. Calculation of SNRs at differing red shifts are necessary, as well as for varying mass ratios to truly explore its performance. Further statistical analysis of the behavior of the SNR vs sky position would certainly be interesting and may hold relevant information, as well as more in depth and knowledgeable statistical analysis of these results in general are needed. PhenomC seems to work well but fully constructing the SNR calibration curve in figure 19 may be of some use. The large variance in the SNR values of LISACode need to be explored considerably, especially in regards to consistency with physical behavior. Looking at why the eLISA configuration remains isolated around 2300 no matter the change in parameters is worth investigation as it may point to possible faults in the code or file itself.

8. ACKNOWLEDGEMENTS

I would like to extend my deepest gratitude to Sweta Shah for sharing her patience, knowledge, and some good laughs with me throughout the project. As well my thanks go to Martin Hewiston for his deep knowledge, and excitement as a whole. I have to extend my thanks to Janis, Dennis, Vitali, Oliver, Richard, Maya, Todd, and Collin for making this an exceptional experience. I also must thank Guido Mueller, Bernard Whiting, Kristin Nichola, Michaela Pickenpack, Andrew Miller, and all of those involved with making this program what it is, and the NSF for funding this oppurtinity.

[1] Pau Amaro-Seoane, et all, "Laser Interferometer Space Antenna,"arXiv:1702.00786 [astro-ph.IM]

[2] Shane L. Larson, William A. Hiscock , Ronald W. Hellings, "Sensitivity curves for spaceborne gravitational wave interferometers," Sep 1999,Phys.Rev. D62 (2000) 062001.

[3] Shane L. Larson (Caltech), Ronald W. Hellings, William A. Hiscock (Montana State U.), "Unequal arm space borne gravitational wave detectors", Jun 2002. 14 pp. Published in Phys.Rev. D66 (2002) 062001 DOI: 10.1103/PhysRevD.66.062001.

[4] Massimo Tinto and Sanjeev V. Dhurandhar,Time-Delay Interferometry,Living Rev. Relativity, 17, (2014),http://www.livingreviews.org/lrr-2014-6

[5] Michele Maggiore, "Gravitational Waves. Vol. 1: Theory and Experiments," Oct 2007 - 572 pages, Oxford Master Series in Physics, Oxford University Press (2007),ISBN: 9780198570745, 9780198520740 (paperback)

[6] C. J. Moore, R. H. Cole, and C. P. L. Berry, Gravitational-wave sensitivity curves, Classical and Quantum Gravity, vol. 32, no. 1, p. 015014, 2015.

[7] A. Petiteau, G. Auger, H. Halloin, O. Jeannin, E. Plagnol, et al.. LISACode : A scientific simulator of LISA. Physical Review D, American Physical Society, 2008, 77 (023002), pp.11. .

[8] L. Santamara, F. Ohme, P. Ajith, B. Brgmann, N. Dorband, M. Hannam, S. Husa, P. Msta, D. Pollney, C. Reisswig, E. L. Robinson, J. Seiler, and B. Krishnan Phys. Rev. D 82, 064016 Published 13 September 2010

[9] Shah, S, "The synergy between gravitational wave and electromagnetic data of compact binaries," 2014, 168 p,Dissertation,Radboud Universiteit Nijmegen, 24 november 2014,9789462594098

[10] Cornish, N. J. and Rubbo, L. J. 2003, Phys. Rev. D, 67, 022001

[11] Emanuele Berti, Alessandra Buonanno, Clifford M. Will (Paris, Inst. Astrophys.), "Estimating spinning binary parameters and testing alternative theories of gravity with LISA, " Nov 2004. 30 pp. Phys.Rev. D71 (2005) 084025, DOI: 10.1103/PhysRevD.71.084025

Appendix A: Code

1. Bash Script

```
#This bash script performs a montecarlo simulation varying about sky location in order to con
#In order to run this must you be run in the Docker[sudo docker run -v /location/of/this/direc
#to run: bash /AVG_SNR_MC/Avg_snr.sh  when in said shell, you can edit the number of runs by c
#                                        Mauricio Diaz-Ortiz Jr.

#!/bin/env bash

#sudo docker run -v /home/morice/Desktop/AVG_SNR_MC_0/:/workspace/AVG_SNR_MC
-it elisadpc/lisacode:develop <---- use this to run docker container
header="AVG S/N Monte Carlo"
now=$(date +"%d/%m/%Y")
parameters="Longitude Latitude Polarisation Inclination SNR"
counter=0
mass1=10000000 #larger mass
mass2=2000000 #smaller mass
q=$(echo "scale=1; $mass2/$mass1" | bc -l)

#Places of a copy of makeEOBNRforLC.py, SkyPosition.py, LISATools.py, makeTDI-lisacode2.py,
 and ConfigNewLisa into their necessary locations.
cp /workspace/AVG_SNR_MC/makeEOBNRForLC.py /workspace/LISACode/Main/Exe
cp /workspace/AVG_SNR_MC/LISATools.py /workspace/LISACode/Main/Exe
cp /workspace/AVG_SNR_MC/SkyPosition.py /workspace
rm -rf ../usr/Cfg/ConfigNewLISA
cp -a /workspace/AVG_SNR_MC/ConfigNewLISA ../usr/Cfg/
rm -rf ../usr/Cfg/makeTDI-lisacode2.py
cp -a /workspace/AVG_SNR_MC/makeTDI-lisacode2.py ../usr/Cfg/

#Preps results.txt by providing relevant information
echo $header > results.txt
echo "Date: $now" >> results.txt
echo "Mass 1: $mass1" >> results.txt
echo "Mass 2: $mass2" >> results.txt
echo "Mass Ratio(q): $q" >> results.txt
echo "$parameters" >> results.txt

#generates TDI using defualt ESA config
python /usr/Cfg/makeTDI-lisacode2.py SNR_MC_tdi

#This loops does all the work in that it generates each EOBNR with the varied parameters
then uses the generated TDI
#and calculates the SNR all of which is output to results.txt
while [ $counter -lt 500 ]

do
        #variables that are varied for the Monte Carlo simulation, these will be varied
        simultaneously via SkyPosition.py.
        Longitude=$( python -c "import SkyPosition; print SkyPosition.longitude()" )
        Beta=$( python -c "import SkyPosition; print SkyPosition.beta()" )
        Latitude=$( python -c "import SkyPosition; print SkyPosition.latitude($Beta)" )
        Theta=$( python -c "import SkyPosition; print SkyPosition.theta()" )
        Phi=$( python -c "import SkyPosition; print SkyPosition.phi()" )
        Inclination=$( python -c "import SkyPosition;
        print SkyPosition.inclination($Theta,$Phi,$Beta,$Longitude)" )
```

```
        Polarisation=$( python -c "import SkyPosition;
         print SkyPosition.polarisation($Theta,$Phi,$Beta,$Longitude)" )

        export Longitude Latitude Inclination Polarisation mass1 mass2
        python LISACode/Main/Exe/makeEOBNRForLC.py --latEcl=$Latitude
        --lonEcl=$Longitude --pol=$Polarisation
        --inc=$Inclination --m1=$mass1 --m2=$mass2 --dist=6.617e9 --plot ASNR_MC
        /usr/bin/LC2SNR -dn ASNR_MC.xml SNR_MC_tdi.xml
        echo $Longitude $Latitude $Inclination $Polarisation
        $(awk '{if(NR==2) print $4}' SNRResults.txt) >> results.txt
        rm -f SNRResults.txt

        ((counter++))

done

cp results.txt /workspace/AVG_SNR_MC/
```

2. SkyPositions.py

```
#!/usr/bin/env python

import math, random

def beta():
    beta = random.uniform(math.pi/2,3*math.pi/2)
    return beta

def longitude():
    lon = random.uniform(0,2*math.pi)
    return lon

def latitude(beta):
    lat = math.sin(beta)
    return lat

def theta():
    theta = random.uniform(0,math.pi)
    return theta

def phi():
    phi = random.uniform(0,2*math.pi)
    return phi

def inclination(theta,phi,beta,lon):
    cos_in = math.cos(theta)*math.sin(beta)+math.sin(theta)*math.cos(beta)+math.cos(phi-lon)
    return cos_in

def polarisation(theta,phi,beta,lon):
    tan_psi = (math.sin(beta)*math.cos(lon-phi)-math.cos(theta)*math.sin(beta))
                        /(math.sin(theta)*math.sin(lon-phi))
    psi = math.atan(tan_psi)
    return psi
```