

# LIGO SURF Final Report

## Spectral Line Monitoring Tool

Evan Anders

Mentors: Greg Mendell and Rick Savage

August 23, 2013

### Abstract

We have created a ‘Spectral Line Monitoring Tool’ (SLM) in the Python programming language which tracks amplitude, phase, and power spectral density at specified frequencies in LIGO data channels. The program uses Fast Fourier Transforms to cast time domain data into the frequency domain to determine characteristics of sinusoidal variations within the data. The uncertainty of SLM’s amplitude measurements for real LIGO data has been studied and an equation for such measurements has been derived using the parameters of Gaussian noise distributions. After development, SLM was utilized to investigate the strain caused by photon calibration lines in LIGO data from the Sixth Science Run.

## 1 Introduction

### 1.1 Gravitational Waves and LIGO

Much as accelerating charges radiate electromagnetic waves as they cause changes in the surrounding electromagnetic field, accelerating masses cause changes in the surrounding gravitational field which radiate throughout the universe as gravitational waves. These gravitational waveforms stretch and squeeze the fabric of spacetime, as shown in Fig. 1, as they pass through it at the speed of light. This stretching or squeezing results in some “strain” on an object, defined as  $h(t) = \Delta L/L$ , where  $L$  is the length of the dimension of the object which is being altered. Unlike electromagnetic waves which are easily scattered or dampened, gravitational waves can pass through most objects undampened making them an extremely powerful means of observing significant astrophysical phenomena such as supernovas or the motion of neutron stars and black holes in binary systems. Unfortunately, even the most powerful gravitational waveforms which should be detectable on earth would be extremely weak. Typical sources of gravitational waves are expected to be many mega-parsecs away and will produce strains on the order of  $10^{-21}$  or less on Earth [1].

The small magnitude of these disturbances makes their detection require extremely sensitive instruments. The Laser Interferometer Gravitational Wave Observatory (LIGO) has developed multiple instances of such an instrument: power-recycled Fabry-Perot Michelson interferometers with 4 km-long arm cavities located at two observatory sites, ‘H1’ in Hanford, WA and ‘L1’ in Livingston, LA [2]. These interferometers are currently undergoing upgrades in order to increase their sensitivity to  $10^{-19}$  m/ $\sqrt{\text{Hz}}$  at 10 Hz and  $2 \times 10^{-20}$  m/ $\sqrt{\text{Hz}}$  at 100 Hz [3]. LIGO’s interferometers are used alongside another large-scale Michelson interferometer, Virgo, in order to both detect and determine the source location of gravitational waves. Due to the exceptionally small magnitude of the gravitational wave strain, detections of the same waveform on multiple interferometers is

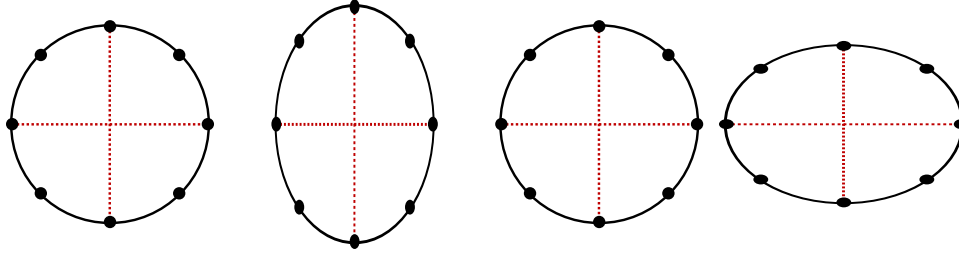


Figure 1: The + gravitational wave polarization is shown; a wave moving into the plane of the paper would cause space to stretch along one axis perpendicular to the motion while it shrinks along the other axis and vice versa. Another polarization, the  $\times$  polarization, is not shown above but causes the same effect as the + polarization rotated by an angle of  $45^\circ$ .

necessary to make any statements of their validity; additionally, the use of multiple interferometers makes it possible to triangulate the location of incoming gravitational waves, something which cannot be accomplished with only one detector [4].

## 1.2 LIGO Data and the Discrete Fourier Transform

LIGO data is gathered by thousands of different channels, often at very high sampling frequencies (16384 Hz). DARM\_ERR, a channel which gives information about differential arm length in terms of counts, contains such sampled data (see Fig. 2). During LIGO science runs, these thousands of channels are continually sampling thousands of data points every second. In short, LIGO produces a staggering amount of data. Such data must be processed in order to draw any meaning out of it and in order to reduce its quantity to a manageable quantity of points.

One such signal processing method is the use of a Discrete Fourier Transform, defined as

$$\tilde{x}_k = \sum_{j=0}^{N-1} x_j e^{-2\pi i j k / N}, \quad (1)$$

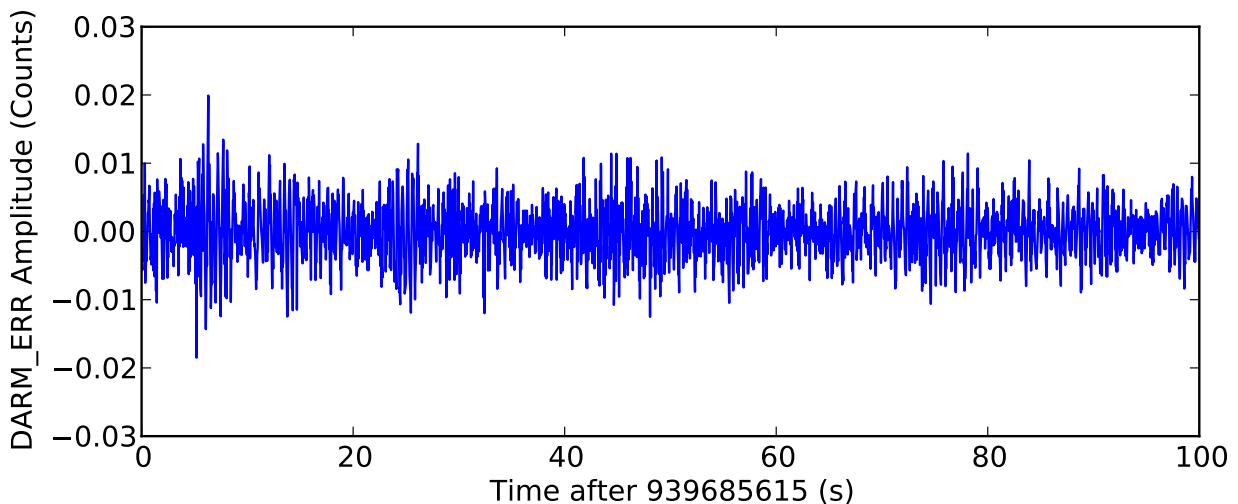


Figure 2: Raw LIGO data from the DARM\_ERR channel during S6; time on the x-axis is expressed in seconds.

where  $k$  is an index in the frequency domain,  $j$  is an index in the time domain,  $x_j$  is a time-domain data point,  $\tilde{x}_k$  is a frequency-domain complex data point,  $i = \sqrt{-1}$ , and  $N$  is the number of data points total. The DFT can alternatively be found by the Fast Fourier Transform (FFT) algorithm, which returns the same data at a much quicker speed. Taking an FFT of a time-domain data set casts the data into a series of complex data points which describe underlying signals in the frequency domain. Knowledge which can be gained through the application of the FFT is limited by a few things: the sampling frequency ( $f_s$ ) and the time over which the FFT is taken ( $T$ ). FFTs only give information about frequencies below some ‘‘Nyquist frequency,’’  $f_{\text{Nyq}} = f_s/2$ . As a result, the ‘‘useful band’’ of data in an FFT refers to the data in the frequency range of  $[0 \text{ Hz}, f_{\text{Nyq}}]$ . The time over which the FFT is taken limits the resolution of frequencies, such that  $\Delta f = 1/T$ . If a frequency in a wave is divisible by  $\Delta f$ , then a strong, ‘‘bin-centered’’ spectral line will be seen in a corresponding frequency bin in the FFT data. Otherwise, if a signal is not bin-centered, there will be some leakage into all other bins of the FFT, particularly in nearby frequency bins. When searching for a particular frequency in a data set, it is necessary to ensure that sampling frequency and FFT timeframes are sufficiently large so as to allow knowledge about desired frequency signals.

Unfortunately, LIGO noise manipulates data signals in strange ways. Perturbations in LIGO data leak out into all frequency bins, making even those data sets which have been Fourier Transformed show grossly inaccurate results. As such, it is necessary to window LIGO data in order to reduce leakage, allowing for more accurate calculation of frequency amplitudes and phases. Two of the data windows used by LIGO scientists, the Tukey window and the Hann window, are pictured in Fig. 3. The Tukey window is defined as [5]

$$W_{\text{Tuk}} = \begin{cases} 0.5 \left[ 1 + \cos \left( \pi \left( \frac{2j}{\alpha(N-1)} - 1 \right) \right) \right], & 0 \leq j \leq \frac{\alpha(N-1)}{2} \\ 1.0, & \frac{\alpha(N-1)}{2} \leq j \leq (N-1) \left( 1 - \frac{\alpha}{2} \right) \\ 0.5 \left[ 1.0 + \cos \left( \pi \left( \frac{2j}{\alpha(N-1)} - \frac{2}{\alpha} + 1 \right) \right) \right], & (N-1) \left( 1 - \frac{\alpha}{2} \right) \leq j \leq (N-1) \end{cases}, \quad (2)$$

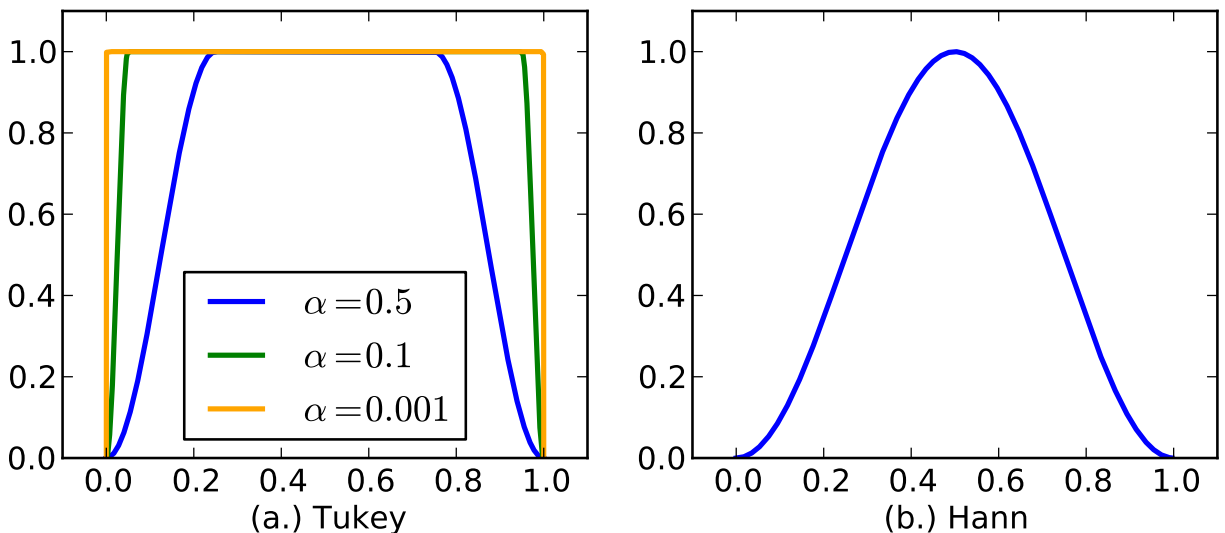


Figure 3: The Tukey Window for various values of  $\alpha$  is shown in (a). As  $\alpha$  approaches 0, the Tukey Window becomes the rectangular window, only minimally affecting the value of data. The Hann window, pictured in (b), drastically changes data values but achieves much greater accuracy in amplitude and PSD calculations after correction factors are applied.

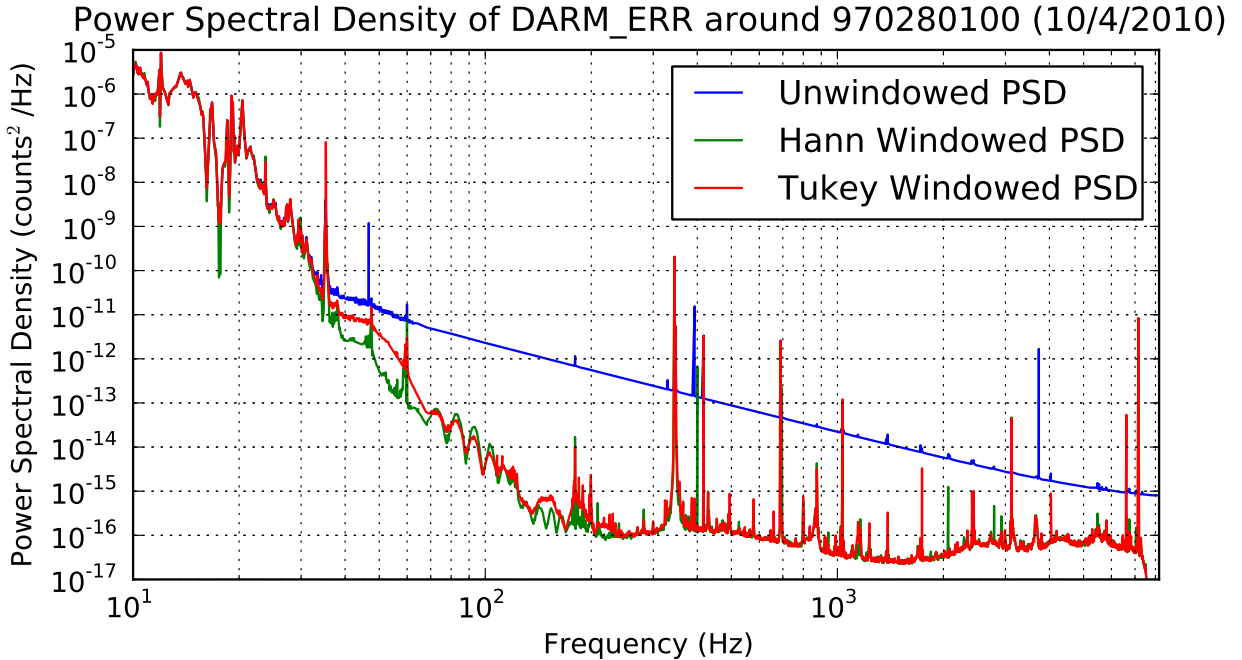


Figure 4: Power Spectral Densities for raw LIGO data were calculated using no window and then Tukey ( $\alpha = 0.001$ ) and Hann windows. The Hann window PSD is corrected by a factor of  $8/3$  for accuracy (see section 4.5.3 for details). Note the gap between Tukey and Hann Windowing below  $\sim 200$  Hz, making the Tukey window with  $\alpha = 0.001$  an ineffective means of gathering information for low-frequency LIGO data.

where  $\alpha$  is a user-defined parameter and  $N$  is the number of data points in the sample being windowed. For LIGO purposes, the Tukey window is calculated for a value of  $\alpha = 0.001$ . This small alpha value makes the Tukey window very similar to a rectangular window (which is to say, data which has not been windowed), while still impressively removing leakage from higher frequency bins. Unfortunately, at lower frequencies, data which has gone through the process of low- $\alpha$  Tukey windowing still fails to achieve optimal frequency resolution, as shown in Fig. 4. A more robust, anti-aliasing window must be used: the Hann window, defined as [5]

$$W_{Hann} = 0.5 \left[ 1.0 + \cos \left( \pi \frac{2j}{N-1} \right) \right], \quad 0 \leq j \leq N-1. \quad (3)$$

While eliminating a great degree of bin spillage, the Hann window causes buried sinusoids to show up within not only their own frequency bin but also the two frequency bins adjoining it (see section 4.5.1). Despite splitting important signals into three bins, the Hann window stops frequencies from spilling much further. Unfortunately, in addition to removing unwanted noise, applying a Hann window modifies the amplitude of FFT data, such that correction factors must be applied when calculating the amplitude, standard deviation, and power spectral density of a set of data; discussion on the derivation of such correction factors can be found in section 4.

## 2 The Spectral Line Monitoring Tool

The Spectral Line Monitoring tool (SLM tool) was written in the Python Programming language, relying extensively on the Python NumPy [6] and Pylal.Frutils [7] programming modules.

### 2.1 SLM Tool Input

The SLM tool is run with command-line input. A sample of such input is as follows:

```
python slm_tool.py
-F 'H1_RDS_R_L1,H1:LSC-DARM_ERR,393.1aups,400.2au;H1_RDS_R_L1,H1:LSC-ETMX_CAL,400.2ap'
-s 962362000 -d 86400 -t 60 -P 10 -p 8 -N example -H True
```

In the above example, the tool is given a semi-colon delimited list of channels, where relevant information about each channel is comma-delimited and comes in the following format:

FRAMETYPE, CHANNELNAME, FREQ1, FREQ2, . . . , FREQN.

Where ‘FREQN’ is the Nth frequency being tracked where N is the number of frequencies being tracked. Each frequency is followed by a string of characters which can contain any combination of ‘a’ (amplitude), ‘u’ (uncertainty), ‘p’ (phase), and ‘s’ (power spectral density). The monitor will calculate only attributes specified for frequencies; in the above example, for the DARM\_ERR channel, all four attributes are calculated for 393.1 Hz, but only amplitude and uncertainty are calculated for 400.2 Hz, and the 400.2 Hz line in ETMX\_CAL is only tracking amplitude and phase. Next, a GPS start time (-s) is provided, and then either a GPS end time (-e) or a duration of the run (-d) in seconds must be provided. In the above example, the monitor is beginning computations at 962362000 and running for 86400 seconds, resulting in a GPS end time of 962448400. After this, a time over which FFTs are taken (-t) must be provided, which in this case is 60 seconds. The following input, -P, tells the SLM tool how many FFTs over which to take averages to calculate power spectral densities, which in this case is 10 (so  $10 * 60 = 600$  seconds of data are used for each PSD calculation). The final three inputs refer to the structure of the output file. The first, -p, determines the precision with which numbers are written to the output file, in this case eight decimal places of scientific notation will be recorded. The next input -N is a string which is attached to the name of the output file to identify it as well as the start time, interferometer symbol, and duration—in this case the output file will be H-example-962362000-86400.txt. Finally, the -H option determines whether or not a header string describing columns should be placed at the top of an output file—for runs in parallel over many different times, it is wise to enable this option for the first file but disable it for all other such that data can be easily compressed to one file.

### 2.2 SLM Tool Output

For any given run, the SLM tool will output two files, which take the form:

```
H-nampiece-920000000-86400.txt and
H-logfile-nampiece-920000000-86400.txt.
```

The first of these files is a data file which contains all relevant information computed by the SLM tool. This file will contain amplitudes and phases in the format seen in Fig. 5.

The second type of output file—the logfile—contains all of the information about the current run of the SLM tool: namely, which frequencies were tracked and what characteristics of those frequencies were tracked. An example of the format of such files can be seen in Fig. 6.

```

GPS TIME   STAT   A 393.1LSC-DARM U 393.1LSC-DARM P 393.1LSC-DARM PSD 393.1LSC-DA A 400.2LSC-DARM U 400.2LSC-DARM A 400.2LSC-ETMX P 400.2LSC-ETMX
0962362000 65535 +7.05442920e-07 +1.59337395e-09 -2.59787984e-01 +0.00000000e+00 +1.83102672e-07 +1.58268859e-09 +1.69994054e+03 +1.00565895e+00
0962362060 65535 +7.02050731e-07 +1.91981765e-09 -2.60399480e-01 +0.00000000e+00 +1.83175930e-07 +1.98871770e-09 +1.69994205e+03 +1.00566430e+00
0962362120 65535 +7.01802364e-07 +1.58560908e-09 -2.59339826e-01 +0.00000000e+00 +1.83734400e-07 +1.66378628e-09 +1.69992048e+03 +1.00566326e+00
0962362180 65535 +6.98966906e-07 +1.87188874e-09 -2.62357245e-01 +0.00000000e+00 +1.83588873e-07 +1.89259693e-09 +1.70000662e+03 +1.00566115e+00
0962362240 65535 +7.03060866e-07 +2.10098708e-09 -2.57105037e-01 +0.00000000e+00 +1.84634413e-07 +1.77134362e-09 +1.70027536e+03 +1.00566300e+00
0962362300 65535 +7.06319474e-07 +1.55742856e-09 -2.58981704e-01 +0.00000000e+00 +1.85122915e-07 +1.91030349e-09 +1.70039498e+03 +1.00566587e+00
0962362360 65535 +7.08651453e-07 +2.17810056e-09 -2.58488718e-01 +0.00000000e+00 +1.86081637e-07 +1.43134968e-09 +1.70067973e+03 +1.00565649e+00
0962362420 65535 +7.00770002e-07 +1.08854106e-09 -2.62169757e-01 +0.00000000e+00 +1.83683821e-07 +1.73715360e-09 +1.70095587e+03 +1.00566422e+00
0962362480 65535 +7.03333011e-07 +1.97908079e-09 -2.63013205e-01 +0.00000000e+00 +1.86327518e-07 +1.45887650e-09 +1.70086132e+03 +1.00565865e+00
0962362540 65535 +7.00189739e-07 +1.69633600e-09 -2.62881611e-01 +4.94299649e-12 +1.83045977e-07 +1.75337351e-09 +1.70065275e+03 +1.00566149e+00
0962362600 65535 +6.97474006e-07 +1.82101297e-09 -2.61703578e-01 +0.00000000e+00 +1.84705574e-07 +1.01003483e-09 +1.70072024e+03 +1.00566360e+00
0962362660 65535 +7.01931188e-07 +1.87943566e-09 -2.64831815e-01 +0.00000000e+00 +1.83117434e-07 +1.72270412e-09 +1.70105878e+03 +1.00566001e+00
0962362720 65535 +7.10474749e-07 +1.73658093e-09 -2.58777765e-01 +0.00000000e+00 +1.85930902e-07 +1.52657283e-09 +1.70135437e+03 +1.00566213e+00
0962362780 65535 +7.12087172e-07 +1.86897366e-09 -2.54869853e-01 +0.00000000e+00 +1.85686570e-07 +1.54860848e-09 +1.70095251e+03 +1.00565265e+00
0962362840 65535 +7.07168709e-07 +1.69763737e-09 -2.60747495e-01 +4.96820313e-12 +1.88451755e-07 +1.99675378e-09 +1.69957506e+03 +1.00565248e+00
0962362900 65535 +7.07316084e-07 +1.86632352e-09 -2.54601493e-01 +0.00000000e+00 +1.85084190e-07 +1.34939182e-09 +1.69838441e+03 +1.00566008e+00
0962362960 65535 +7.07778266e-07 +1.72643963e-09 -2.55274623e-01 +0.00000000e+00 +1.86382470e-07 +1.77797931e-09 +1.69832581e+03 +1.00565839e+00
0962363020 65535 +7.03660836e-07 +1.62854800e-09 -2.61635725e-01 +0.00000000e+00 +1.86057438e-07 +1.76450161e-09 +1.69852677e+03 +1.00565705e+00
0962363080 65535 +7.07205871e-07 +1.99157820e-09 -2.56048664e-01 +0.00000000e+00 +1.83481476e-07 +2.05666357e-09 +1.69860767e+03 +1.00565854e+00
0962363140 65535 +7.00167401e-07 +1.72644841e-09 -2.61637018e-01 +4.97786668e-12 +1.83753925e-07 +1.67072755e-09 +1.69857720e+03 +1.00565820e+00
0962363200 65535 +6.99371522e-07 +1.89544838e-09 -2.67844388e-01 +0.00000000e+00 +1.85505006e-07 +1.61143893e-09 +1.69846318e+03 +1.00565258e+00

```

Figure 5: Sample output of the SLM tool is shown. This run calculated amplitude, uncertainty, phase, and power spectral density of 393.1 Hz as well as amplitude and uncertainty of 400.2 Hz in DARM\_ERR; amplitude and phase of 400.2 Hz in ETMX\_CAL were also tracked. Output is shown with precision set to 8 and the header bool set to true.

## 2.3 SLM Tool Graphing Feature

In addition to calculating FFTs and returning data, an additional Python script can be run to graph data produced by the SLM tool. This program is run from command line input, similar to the SLM tool, taking the following form:

```
python slm_grapher.py -s 957955600 -d 86400 -t 60 -g a -N pcallines -I H -n 100
```

This program processes information regarding time in the same manner as the `slm.tool.py` program, such that `-s`, `-e`, `-d`, and `-t` are equivalent entries for GPS start time, GPS end time, time duration, and FFT timestep, respectively. The `-N` entry is also the same as it was for the prior program, giving the file nampiece. The `-g` entry is the graphing entry and can be any letter in the set (a, p, s), referring to amplitude, phase, and power spectral density, respectively. All calculations of that type in the given run will be plotted. The `-I` refers to the interferometer for which data is being plotted ('H' or 'L'). The final entry, `-n`, is optional and allows the user to reduce the number of data points plotted through taking averages over time (of only science-mode data). In this case, the user only wants 100 data points plotted. However, the total number of data points is  $86400/60 = 1440$ , and  $1440/100 = 14.4$ . As such, the program takes averages of 14 points in order

```

[1] -
FRAMETYPE:  H1_RDS_R_L1
CHANNELNAME: H1:LSC-DARM_ERR
FREQUENCY:  400.2
AMP:        True
UNCRT:     True
PHASE:     False
PSD:       False

```

Figure 6: Information regarding the second frequency (indexing begins at 0 and counts up) in the SLM run referred to in the caption of Fig. 5. For this run, The amplitude and amplitude uncertainty of 400.2 Hz in DARM\_ERR are calculated.

to produce each point; unfortunately, this still leaves  $0.4 * 100 = 40$  data points unaccounted for; in order to fix this problem, the monitor creates a final data point which averages those 40 “left over” points, plotting 101 points total. It is advisable that the user selects a number of data points to plot by which the total number of data in the file is divisible.

## 2.4 SLM Tool Under-the-hood

The SLM tool uses the `AutoqueryingFrameCache` object from the `Pylal.Frutils` module to fetch LIGO data frame segments. Once the program has the data, the program applies a Hann window and performs an FFT. At this point, the necessary operations for calculation of amplitude, uncertainty, phase, or power spectral density are carried out and information is saved to the output data file.

For each line in which an amplitude must be calculated, the program calculates the value for the amplitude using

$$A = 2 \left( \frac{2|\tilde{x}_k|}{N} \right), \quad (4)$$

where  $|\tilde{x}_k|$  is the magnitude of the Fourier Transformed data in the bin  $k$ , which corresponds to the frequency at which the line is being calculated. The program uses an extra factor of 2 in front of the amplitude calculation in order to correct for the process of Hann windowing. For information about how Eq. (4) is derived or how the correction factor of 2 is found, see sections 4.2 and 4.5.2, respectively.

Phase calculations of the SLM are simple, and take the form of

$$\phi = \tan^{-1} \left( \frac{\Im[\tilde{x}_k]}{\Re[\tilde{x}_k]} \right), \quad (5)$$

where  $\phi$  is the phase,  $\tilde{x}_k$  is the value of the FFT in frequency bin  $k$ , and  $\Re$  and  $\Im$  give the real and imaginary parts of the FFT, respectively.

The SLM calculates power spectral density (PSD) such that

$$S_n(f_k) = \frac{8}{3} \left( \frac{2\langle |\tilde{x}_k|^2 \rangle \Delta t^2}{T} \right), \quad (6)$$

where  $S_n(f_k)$  is the PSD in frequency bin  $k$ ,  $\Delta t = 1/f_s$  (where  $f_s$  is the sampling frequency of the data),  $T$  is the time of the FFT, and  $\langle |\tilde{x}_k|^2 \rangle$  is the average of the squared magnitude of the FFT bin taken over time, meaning that multiple FFTs must be taken before a PSD calculation can be completed. The value of  $8/3$  is applied to correct for Hann windowing the data, as derived in section 4.5.3. It is important to note that this correction factor is only valid for noise; if the SLM tool calculates PSD values for bins containing signals with sufficiently large signal-to-noise ratios, it will find a value which is  $2/3$  of that bin’s true value.

Amplitude uncertainty calculations of the SLM tool take the form of

$$\sigma_A = 2 \sqrt{\frac{2\langle |\tilde{n}|^2 \rangle}{N^2}} \quad (7)$$

where  $|\tilde{n}|$  is the magnitude of the FFT in a frequency bin corresponding to noise surrounding a signal and  $\langle |\tilde{n}|^2 \rangle$  is the average of the squares of ten such bins above and below the signal. As the Hann window causes a signal to spill into the two most nearby frequency bins, this average is taken for ten bins above and below the frequency being tracked, starting two bins away from the signal both above and below the frequency of interest. While the power spectral density is different from

bin to bin, background noise within a few frequency bins is close enough to equivalent that this is a relatively good approximation. A correction factor of 2 is applied to this calculation due to the process of Hann windowing. It is important to note at this point that if the program were to correct the power spectral density (by a factor of  $8/3$ ) before the calculation of uncertainty, then the correction factor of uncertainty would become  $\sqrt{3/2}$  rather than 2. For more information about the derivation of Eq. (7) or the correction factor, see sections 4.3 and 4.5.4, respectively.

## 2.5 Application: S6 Photon Calibration

After production of the SLM tool, the program was applied to data from S6 in order to test its validity. Measurements of LIGO Photon calibration (Pcal) output power were used to determine the size of displacement which would be caused by the photon calibrator's beams. For information on specifics of how the test mass displacement was calculated, see section 4.1.

A plot of the comparison between calculated Pcal strain (based on laser output) and measured Pcal strain (from the `H1:LDAS-STRAIN` data channel) can be seen in Fig. 7, where the mean of the absolute value of the difference between the two channels differs by roughly 3.88%, well within the error range of the strain channel's calculation during the S6 run.

While results are not perfect, they satisfactorily prove that the SLM tool can retrieve valid amplitude values to track LIGO calibration frequencies.

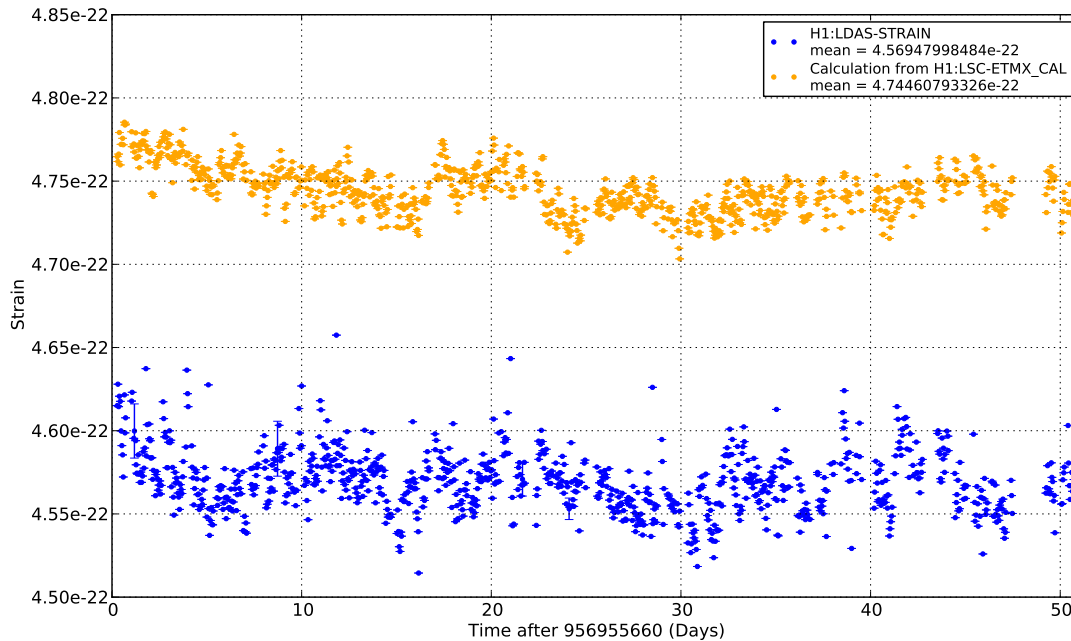


Figure 7: Values for the strain on LIGO caused by photon calibration and strain measured by the `H1:LDAS-STRAIN` channel are shown above. 1224 data points are shown (one data point per hour) containing information from LIGO data between 5/3/2010 and 6/23/2010; only those points for which the interferometer was in science mode are included. The mean percent difference between the two data sets is roughly 3.88%; the ratio between the two traces is stable over the 51 days.



### 3 Conclusion and Findings

A first version of the SLM tool has been produced for the future tracking of calibration lines in LIGO data. This program has been created such that it has a high level of flexibility (it can track any number of amplitudes the user desires) and such that it can be easily expanded with extra features, such as the ability to calculate information about frequencies which are not bin-centered. The tool has been tested and proven accurate through use on old data from LIGO’s sixth science run.

In addition to the creation of this tool, we find the factor giving the uncertainty of amplitude in Hann windowed data. While the Hann window is exceptionally useful in increasing frequency resolution of noisy data, this window does increase the uncertainty of amplitude calculations by a factor of  $\sqrt{3/2}$ .

### 4 Methods

The following contains in-depth explanations of how Pcal strain is calculated and how the equations used by the SLM tool are derived. Sections are as follows:

- 4.1 – An explanation of how Pcal motion is derived from data sets
- 4.2 – Derivation of the uncorrected amplitude equation for a Fourier Transformed signal
- 4.3 – Derivation of the uncorrected amplitude uncertainty equation for a Fourier Transformed signal in roughly Gaussian noise
- 4.4 – A brief explanation of the uncorrected equation used for one-sided power spectral density calculations
- 4.5 – Various Hann window calculations, such as:
  - 4.5.1 – A derivation of the Fourier Transformed equation of the Hann window
  - 4.5.2 – A derivation of the correction factor which must be used on signal amplitudes in Hann windowed data
  - 4.5.3 – A derivation of the correction factor which must be used on power spectral densities of Hann windowed data
  - 4.5.4 – A derivation of the correction factor which must be used on amplitude uncertainties in Hann windowed data

#### 4.1 Calculation of Pcal Strain

During the S6 science run, LIGO Hanford’s photon calibrator produced a sinusoidally varying signal at a frequency  $f = 400.2$  Hz. This laser produced a beam with some measurable power which varied with an amplitude of  $P_m$ . As such, the time-varying portion of the power leaving the photon calibrator laser was equivalent to

$$P(t) = P_m \sin(2\pi ft). \tag{8}$$

A small portion of this beam was sent into an integrating sphere which contained a photodetector whose values were read back and transformed into the data contained in the H1:LSC-ETMX\_CAL channel in units of “counts.” The rest of the beam is then split into two roughly equal parts (one

called the ‘transmitted’ beam and one called the ‘reflected’ beam) which then passed through a viewport window (of some optical efficiency  $< 1$ ) and struck the end test mass (ETM) at two points at angles of  $\theta_{\text{trans}}$  and  $\theta_{\text{refl}}$ . The vast majority of the laser which struck the ETM reflected, imparting momentum to the ETM; as the two beams struck the ETM at sufficiently symmetrical positions, only motion in the axis of LIGO’s arm was induced (the torque imposed by one beam cancelled that caused by the other beam).

Assuming that the transmitted and reflected beams respectively had  $n$  and  $m$  photons strike the mirror every second, the quantity by which the momentum of the mirror changed (ignoring the restoring force of the ETM suspension) was equivalent to

$$\frac{dp}{dt} = \frac{hf}{c} (n \cos \theta_{\text{trans}} + m \cos \theta_{\text{refl}}) \quad (9)$$

where  $h$  is Planck’s constant and  $c$  is the speed of light. The amplitude of power incident upon the mirror can thus be related back to the power source by some equation

$$P = P_m (N_{\text{corr\_trans}} + N_{\text{corr\_refl}}) = (m + n)hf, \quad (10)$$

where  $N_{\text{corr}}$  is some correction factor which accounts for all losses which the laser power incurs on the way to the ETM. To be safe, we split the correction factor into two parts—one for the reflected beam and one for the transmitted beam. The change in momentum thus becomes

$$\frac{dp}{dt} = \frac{P_m \sin(2\pi ft)}{c} (N_{\text{corr\_trans}} \cos \theta_{\text{trans}} + N_{\text{corr\_refl}} \cos \theta_{\text{refl}}). \quad (11)$$

Assuming that the ETM has mass  $M$  and has some induced amplitude of displacement  $x$ , it is clear that

$$M\ddot{x} = \frac{P_m \sin(2\pi ft)}{c} (N_{\text{corr\_trans}} \cos \theta_{\text{trans}} + N_{\text{corr\_refl}} \cos \theta_{\text{refl}}). \quad (12)$$

Integrating to obtain  $x$ , we see that sinusoidal motion is induced. The amplitude of this motion,  $x_m$ , is found to be

$$x_m = -\frac{P_m}{Mc(2\pi f)^2} (N_{\text{corr\_trans}} \cos \theta_{\text{trans}} + N_{\text{corr\_refl}} \cos \theta_{\text{refl}}), \quad (13)$$

where the negative sign signifies that the motion of the mirror is in the opposite direction of the motion of the photons as it is being driven well above its resonant frequency [2].

In the process of determining the true power incident upon the ETM, the power of the transmitted and reflected beams must be measured with an integrating sphere referred to as the “working standard.” During this process, these beams are entirely directed into the integrating sphere and a measurement of a photodetector in the integrating sphere is recorded. A ratio between this measurement and the measurement of LSC-ETMX\_CAL (the small portion of light extracted from the beam after it leaves the laser) was taken and recorded at various times over the course of S6. This measurement is taken for both the transmitted and reflected beam, resulting in ratios of  $PMcal_r$  and  $PMcal_t$ , both of which are in volts\_WS/count (where volts\_WS refers to volts measured by the working standard). NIST calibrates the integrating spheres to some “gold standard” at which the voltage reading of the photodiode can be converted to power of light inside of the sphere (some value  $GS\_cal$ , in volts\_GS/Watt, where volts\_GS is volts measured at the gold standard). LIGO has measured a conversion factor between the “gold standard” and “working standard” ( $WS\_to\_GS$ ) in order to reconcile measurements and obtain power reading from measured volages. In addition to these conversion factors, the viewport window only allows a certain portion of the pcal light to

pass through it ( $OE\_viewport$ , the optical efficiency of the viewport) and a very small portion of the light passes through the ETM rather than reflecting off of it (that part which is reflected is the  $refl\_ratio$ ).

All of these important variables are contained in the following [8]:

1.  $ETMX\_CAL$  – The reading, in counts, of the H1:LSC-ETMX\_CAL channel.
2.  $PMcal\_r = 1.98626 \times 10^{-5}$  (in volts\_WS/count)
3.  $PMcal\_t = 2.00577 \times 10^{-5}$  (in volts\_WS/count)
4.  $WS\_to\_GS = 0.9985$  (volts\_GS/volts\_WS)
5.  $GS\_cal = 3.1955$  (volts\_GS/watts)
6.  $OE\_viewport = 0.888$  (Optical efficiency of viewport)
7.  $refl\_ratio = 0.9997$  (Portion of light which reflects off ETM)

Utilizing these variables, the correction factors can be found, such that

$$\begin{aligned} N_{corr-trans}P_m &= \frac{(WS\_to\_GS)(refl\_ratio)(ETMX\_CAL)(OE\_viewport)(PMcal\_t)}{(GS\_cal)} \\ N_{corr-refl}P_m &= \frac{(WS\_to\_GS)(refl\_ratio)(ETMX\_CAL)(OE\_viewport)(PMcal\_r)}{(GS\_cal)} \end{aligned} \quad (14)$$

As such, we can rewrite the equation for amplitude of induced displacement such that

$$x_m = \frac{(ETMX\_CAL)(OE\_viewport)(WS\_to\_GS)}{(GS\_cal)Mc(2\pi f)^2} ((PMcal\_t) \cos \theta_{trans} + (PMcal\_r) \cos \theta_{refl}). \quad (15)$$

The strain on LIGO's arm can be calculated such that

$$h(t) = \frac{\Delta L}{L}. \quad (16)$$

Using Eq. (15) along with the following values:

1.  $\theta_{refl} = 9.03299^\circ$
2.  $\theta_{trans} = 10.11154^\circ$
3.  $M = 10.338$  kg
4.  $c = 299,792,458$  m/s
5.  $L = 4000$  m

it is possible to solve for  $x_m$  from photon calibration. Using these results, it is easy to calculate the strain caused by photon calibration on LIGO. Measurements of this form, along with data points produced by the H1:LDAS-STRAIN channel are shown in Fig. 7.

## 4.2 Derivation of the SLM Amplitude Equation

Suppose you have some discretized sinusoidal signal,

$$x_j = A \cos(2\pi f t_j + \phi_0), \quad (17)$$

where  $A$  is the amplitude,  $f$  is the signal frequency,  $\phi_0$  is the phase, and  $x_j$  is the signal's value at some discrete time,  $t_j$ . Applying Euler's formula, Eq. (17) becomes

$$x_j = \frac{A}{2} \left[ e^{(2\pi i f t_j + \phi_0 i)} + e^{-(2\pi i f t_j + \phi_0 i)} \right] = \frac{A}{2} \left[ e^{i\phi_0} e^{2\pi i f t_j} + e^{-i\phi_0} e^{-2\pi i f t_j} \right], \quad (18)$$

where  $i = \sqrt{-1}$ . If  $f$  is a bin-centered frequency in the  $k$ th frequency bin after applying a DFT to the set of all  $x_j$  data points, we know that  $f = T/k$ , where  $T$  is the time of the DFT. Thus, substituting for  $f$  as well as  $t_j = j\Delta t$  where  $\Delta t = 1/f_s$ , the sampling frequency of the data, the value of the signal at any  $j$  point in time can be expressed as

$$x_j = \frac{A}{2} \left[ e^{i\phi_0} e^{2\pi i j k / N} + e^{-i\phi_0} e^{-2\pi i j k / N} \right], \quad (19)$$

where  $N = f_s T$ . Plugging Eq. (19) into the definition of the DFT and acknowledging that the only  $k$  of interest is that which contains the frequency of our signal ( $k = fT$ ), we obtain

$$\tilde{x}_k = \sum_{j=0}^{N-1} x_j e^{-2\pi i j k / N} = \frac{A}{2} \sum_{j=0}^{N-1} \left( e^{i\phi_0} e^{2\pi i j k / N} + e^{-i\phi_0} e^{-2\pi i j k / N} \right) e^{-2\pi i j k / N}. \quad (20)$$

Through distribution, this simplifies to

$$\tilde{x}_k = \frac{A}{2} \left( \sum_{j=0}^{N-1} e^{i\phi_0} + \sum_{j=0}^{N-1} e^{-i\phi_0} e^{-4\pi i j k / N} \right), \quad (21)$$

the first term of which is a trivial sum and the second term of which is a geometric series with  $r = e^{-4\pi i k / N}$ . The DFT of the signal becomes

$$\tilde{x}_k = \frac{AN}{2} e^{i\phi_0} + \frac{A}{2} e^{-i\phi_0} \left( \frac{1 - e^{-4\pi i k}}{1 - e^{-4\pi i k / N}} \right), \quad (22)$$

the second term of which is zero for  $k \neq 0$ . The remaining exponential in the first term is a complex number with a magnitude of 1; as such,  $|\tilde{x}_k| = AN/2$ . Solving for  $A$ , we find that

$$A = \frac{2|\tilde{x}_k|}{N}. \quad (23)$$

## 4.3 Derivation of the SLM Uncertainty Equation

The signal-to-noise ratio (SNR) of a signal found in the frequency bin  $k$  of some Fourier Transformed data is defined as

$$d^2 = \frac{4\Delta t^2}{T} \frac{|\tilde{h}_k|^2}{S_n(f_k)}, \quad (24)$$

where  $d$  is the SNR,  $\Delta t$  is the time resolution of the data set in the time domain,  $T$  is the time over which the Fourier Transform was taken,  $\tilde{h}_k$  is the value of the Fourier Transform of the signal

at index  $k$ , and  $S_n(f_k)$  is the power spectral density of the background noise at  $f_k$  [9]. When there is no signal involved, the normalized total power of the noise can be expressed as

$$\rho = \frac{4\Delta t^2}{TS_n(f_k)} (\Re^2[\tilde{n}_k] + \Im^2[\tilde{n}_k]), \quad (25)$$

where  $\Re^2[\tilde{n}_k]$  and  $\Im^2[\tilde{n}_k]$  are the squares of the real and imaginary parts of the Fourier Transform in bin  $k$ , respectively. The real and imaginary parts of  $\tilde{n}_k$ , depending only upon the value of Gaussian noise, are both Gaussian Distributed variables. As  $\rho$  depends on the addition of the squares of these two variables, it is clear that  $\rho$  in the frequency bin  $k$  for Gaussian noise should follow a chi-squared distribution of two degrees of freedom (one for the real and one for the imaginary part).

The addition of a signal changes this simple model slightly. The power now depends on both a signal and noise, and Eq. (25) becomes

$$\rho = \frac{4\Delta t^2}{T} \frac{|\tilde{h}_k + \tilde{n}_k|^2}{S_n(f_k)} = \frac{4\Delta t^2}{TS_n(f_k)} (\Re^2[\tilde{h}_k + \tilde{n}_k] + \Im^2[\tilde{h}_k + \tilde{n}_k]), \quad (26)$$

Assuming the amplitude of the signal does not measurably change, the real and imaginary parts of the Fourier Transform are now constructed from two parts: a constant factor (from the signal) as well as some deviation (the Gaussian Distributed noise variable). Whereas we could express  $\rho$  of pure Gaussian noise as a chi-squared distribution of two degrees of freedom, we can now express  $\rho$  of a signal imbedded in Gaussian noise as a *non-central* chi-squared distribution of two degrees of freedom, as the signal adds in a constant non-centrality parameter to both of the degrees of freedom making up the distribution.

A non-central chi-squared distribution is defined such that

$$\begin{aligned} \bar{\rho} &= \nu + \lambda \text{ and} \\ \sigma_\rho^2 &= 2(\nu + 2\lambda), \end{aligned} \quad (27)$$

where  $\bar{\rho}$  is the mean value of the distribution,  $\sigma_\rho^2$  is its variance,  $\nu$  is the number of degrees of freedom (in our case,  $\nu = 2$ ), and  $\lambda \approx d^2$ , the squared SNR. For sufficiently large SNR,  $\nu$  is negligible compared to  $\lambda$  and drops out of our equations. In such cases of large SNR, frequency bins will display the property  $|\tilde{h}_k| \gg |\tilde{n}_k|$ ; as a result, we use the approximation of  $|\tilde{n}_k| \approx 0$  in Eq. (26). Solving for  $|\tilde{h}_k|$  in terms of amplitude ( $A$ , as defined in Eq. (23)) and  $N$  (acknowledging that  $N = T/\Delta t$ ), we can solve for a simplified SNR,

$$\rho \approx \lambda = d^2 = \frac{A^2 T}{S_n(f_k)}, \quad (28)$$

Solving for the amplitude, we find that

$$A = \sqrt{\frac{\rho S_n(f_k)}{T}}. \quad (29)$$

Assuming negligible uncertainty of  $S_n$ , propagation of uncertainty shows that

$$\sigma_A = \left( \frac{\partial A}{\partial \rho} \right)_{\bar{\rho}} \sigma_\rho = \frac{1}{2} \sqrt{\frac{S_n(f_k)}{T \bar{\rho}}} \sigma_\rho \quad (30)$$

Due to the large value of the SNR,  $\sigma_\rho \approx 2\sqrt{\lambda} \approx 2\sqrt{\bar{\rho}}$ . As such, all dependence on the mean value (the value of the signal) disappears and the uncertainty of an amplitude in Gaussian noise can be approximated as

$$\sigma_A = \sqrt{\frac{S_n(f_k)}{T}}. \quad (31)$$

#### 4.4 Power Spectral Density Equation

While the exact derivation is outside of the scope of this report, the one-sided power spectral density of the frequency in bin  $k$ ,  $f_k$ , can be estimated as

$$S_n(f_k) = \frac{2\langle |\tilde{x}_k|^2 \rangle \Delta t^2}{T}, \quad (32)$$

where  $\Delta t$  is the time resolution of gathered data,  $T$  is the time of the DFT, and  $\langle |\tilde{x}_k|^2 \rangle$  is the square of the magnitude of the Fourier Transformed data in frequency bin  $k$ , averaged over time (multiple DFTs).

#### 4.5 Derivation of Hann Window Correction Factors

##### 4.5.1 Frequency-domain expression of the Hann window

The Hann window, defined in Eq. (3), can be expressed in terms of complex exponentials, such that

$$W_j = \frac{1}{2} \left[ 1 - \frac{1}{2} \left( e^{2\pi i j / (N-1)} + e^{-2\pi i j / (N-1)} \right) \right], \quad (33)$$

where  $i = \sqrt{-1}$  and  $W_j$  is the  $j$ th element of the window. For sufficiently large data sets,  $N-1 \approx N$ . Taking the DFT of the Hann Window, we find that

$$\widetilde{W}_k = \sum_{j=0}^{N-1} W_j e^{-2\pi i j k / N} = \sum_{j=0}^{N-1} \left( \frac{1}{2} e^{-2\pi i j k / N} - \frac{1}{4} e^{-2\pi i j (k-1) / N} - \frac{1}{4} e^{-2\pi i j (k+1) / N} \right). \quad (34)$$

Splitting this into three different sums, each term proves to be a geometric series. All sums can thus be replaced such that

$$\widetilde{W}_k = \frac{1}{2} \left( \frac{1 - e^{-2\pi i k}}{1 - e^{-2\pi i k / N}} \right) - \frac{1}{4} \left( \frac{1 - e^{-2\pi i (k-1)}}{1 - e^{-2\pi i (k-1) / N}} \right) - \frac{1}{4} \left( \frac{1 - e^{-2\pi i (k+1)}}{1 - e^{-2\pi i (k+1) / N}} \right). \quad (35)$$

For most integers  $k$ , this expression is zero. However, there are undefined terms when  $k = 0, 1, -1$ . Through taking the Taylor Series expansion of  $e^{-ix}$  around  $x = 0$  or through acknowledging the orthogonality of the DFT basis functions, it can be found that

$$\widetilde{W}_k = \frac{N}{2} \delta_{k,0} - \frac{N}{4} \delta_{k,1} - \frac{N}{4} \delta_{k,-1}, \quad (36)$$

where  $\delta_{m,n}$  is the kronecker delta.

Application of a window to a set of data is multiplication in the time domain, such that

$$x_j^w = W_j x_j, \quad (37)$$

where  $x_j^w$  is the windowed data at time index  $j$ . According to the convolution theorem, multiplication in the time domain is equivalent to convolution in the frequency domain. Applying this knowledge, it is clear that

$$\tilde{x}_k^w = \widetilde{W}_k * \tilde{x}_k = \frac{1}{N} \sum_{k'=0}^{N-1} \tilde{x}_{k'} \widetilde{W}_{k-k'}, \quad (38)$$

which can be used alongside Eq. (36) in order to determine the expression for a Hann windowed set of data,

$$\tilde{x}_k^w = \frac{1}{2} \tilde{x}_k - \frac{1}{4} \tilde{x}_{k-1} - \frac{1}{4} \tilde{x}_{k+1}. \quad (39)$$

### 4.5.2 Amplitude correction factor

For a set of data with a bin-centered frequency contained in the bin  $k$ , it is clear that all other frequencies bins in an un-windowed set of data would be empty. As such, Eq. (39) simplifies to

$$\tilde{x}_k^w = \frac{1}{2}\tilde{x}_k. \quad (40)$$

When calculating the amplitude of a signal according to Eq. (4), the SLM tool corrects measured amplitudes by a factor of 2, such that

$$A = 2 \left( \frac{2|\tilde{x}_k^w|}{N} \right) \quad (41)$$

### 4.5.3 Power Spectral Density correction factor

If we have some unwindowed set of Gaussian distributed data points, we can refer to Parseval's theorem to find that

$$\frac{1}{N} \sum_{k=0}^{N-1} |\tilde{n}_k|^2 = \sum_{j=0}^{N-1} |n_j|^2 = N\sigma^2, \quad (42)$$

where  $\tilde{n}_k$  is the Fourier Transform at frequency bin  $k$  of the data set of Gaussian Distributed time-domain noise values,  $n_j$ , whose variance is  $\sigma^2$ . By definition, the variance of such Fourier Transformed data points is  $N\sigma^2$ .

Hann Windowing our data, we find Eq. (39). Assuming that the frequency bins  $\tilde{x}_k$ ,  $\tilde{x}_{k+1}$ , and  $\tilde{x}_{k-1}$  all contain Gaussian distributed noise, we can find the variance of hann-windowed data utilizing propagation of uncertainty, such that

$$\sigma_{\text{Hann}}^2 = \left( \frac{\partial \tilde{x}_k^w}{\partial \tilde{x}_k} \sigma_{\tilde{x}_k} \right)^2 + \left( \frac{\partial \tilde{x}_k^w}{\partial \tilde{x}_{k+1}} \sigma_{\tilde{x}_{k+1}} \right)^2 + \left( \frac{\partial \tilde{x}_k^w}{\partial \tilde{x}_{k-1}} \sigma_{\tilde{x}_{k-1}} \right)^2 = \left( \frac{1}{2} \sigma_{\tilde{x}_k} \right)^2 + \left( \frac{1}{4} \sigma_{\tilde{x}_{k+1}} \right)^2 + \left( \frac{1}{4} \sigma_{\tilde{x}_{k-1}} \right)^2. \quad (43)$$

But we already know that

$$\sigma_{\tilde{x}_k} = \sigma_{\tilde{x}_{k+1}} = \sigma_{\tilde{x}_{k-1}}. \quad (44)$$

As such,

$$\sigma_{\text{Hann}}^2 = \frac{3}{8} \sigma_{\tilde{x}_k}^2. \quad (45)$$

For a set of Gaussian distributed data points, averages taken over a range of frequencies are equivalent to averages of one frequency taken over time assuming that the noise distribution is constant over both time (static) and frequency (white). As such, we can apply the knowledge gained in Eq. (42) to Eq. (32) in order to see that the power spectral density of pure Gaussian noise can be simplified to

$$S_n(f_k) = \frac{2N\sigma^2\Delta t^2}{T} = \frac{2\sigma^2}{f_s}. \quad (46)$$

Applying the ratio of variances which we find in Eq. (45), the correction factor for the power spectral density of Gaussian noise can be found, such that

$$S_n(f_k) = \frac{8}{3} S_n^w(f_k), \quad (47)$$

where  $S_n^w(f_k)$  is the power spectral density in frequency bin  $k$ .

We have verified that the correction factor of 8/3 is applicable to Gaussian noise through a comparison of the ratio of Tukey windowed LIGO data ( $\alpha = 0.001$ ) and Hann windowed LIGO data, and found that the ratio is, on average, 8/3. A sample of the discovery of this ratio can be seen in Fig. 8.

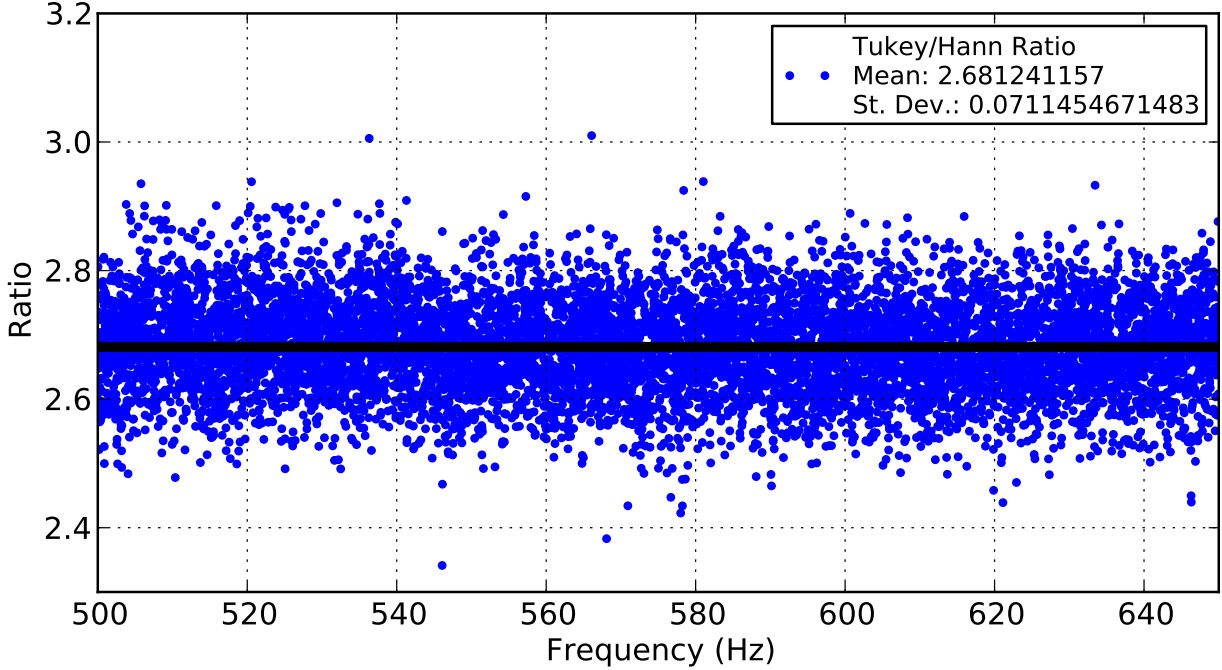


Figure 8: The ratio between Tukey windowed and Hann windowed data for a 1000-FFT average of PSD, using 60-second data chunks, starting on October 5, 2010 and only using science mode data.

#### 4.5.4 Uncertainty correction factor

Returning to our derivation of the uncertainty of the signal-to-noise ratio in a set of data, we modify Eq. (28), acknowledging that  $A^w = A/2$  and  $S_n^w(f_k) = 3S_n(f_k)/8$  where  $A^w$  and  $S_n^w(f_k)$  are the uncorrected amplitude and power spectral density of Hann windowed data. We find that

$$\lambda = d^2 = \frac{\left(\frac{A}{2}\right)^2 T}{\frac{3S_n^w(f_k)}{8}} = \frac{2}{3} \frac{A^2 T}{S_n(f_k)}. \quad (48)$$

Solving for  $A$ , we find that

$$A = \sqrt{\frac{3\lambda S_n(f_k)}{2T}}. \quad (49)$$

Through propagation of uncertainty, it can be determined that

$$\sigma_A = \sqrt{\frac{3}{2} \frac{S_n(f_k)}{T}} = 2\sqrt{\frac{S_n^w(f_k)}{T}}. \quad (50)$$

Rather than calculating the corrected power spectral density and then calculating the corrected amplitude uncertainty, the SLM tool simply calculates the uncorrected power spectral density of the windowed FFT data, then applies the correction factor of 2 at the end.

It is, however, important to note that the uncertainty of amplitude calculations of Hann Windowed data is greater than unwindowed data by a factor of  $\sqrt{3/2}$ . This factor is not readily found in most discussions of signal processing but has been proven here to be true for both ideal Gaussian noise and LIGO data, as seen in Fig. 9.



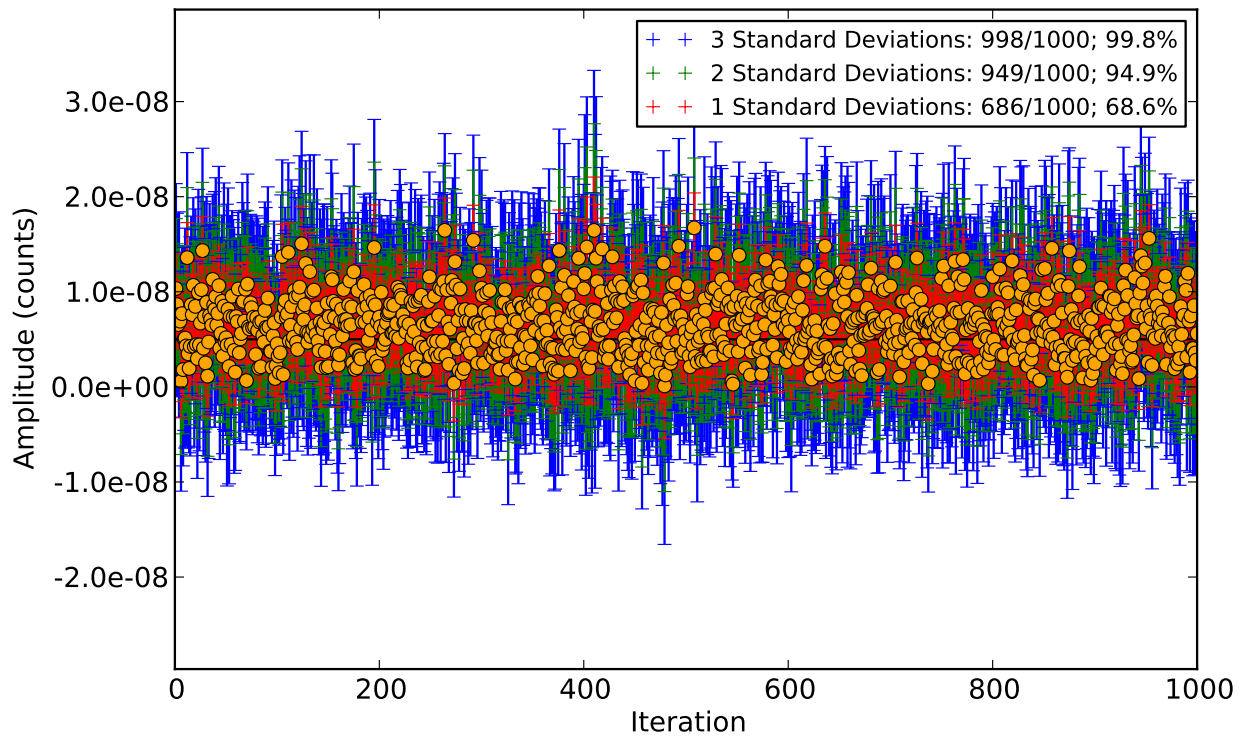


Figure 9: A signal with an amplitude of  $5 \times 10^{-9}$  and frequency of 100.3 Hz was injected into 1000 60-second data segments of science mode data taken from the `DARM_ERR` channel from 7/4-5/2010. The data was Hann windowed and Fourier Transformed, and then the proper equations were applied to calculate the amplitude and amplitude uncertainty using known correction factors. These results are approximately what would be expected for Gaussian noise.

## References

- [1] Bernard F. Schutz. *A first course in general relativity*. Cambridge University Press, 1985.
- [2] Alberto Sottile. *Characterization of the Photon Calibrator data from LIGO's sixth Science Run*. PhD thesis, University of Pisa, 2011. LIGO DCC: LIGO-P1100013-v1.
- [3] S. J. Waldman and the LIGO Scientific Collaboration. The advanced ligo gravitational wave detector. In *2010 Rencontres de Blois conference*, March 2011. doi: arXiv:1103.2728.
- [4] B. P. Abbott et al. Ligo: the laser interferometer gravitational-wave observatory. *Rep. Prog. Phys.*, 72, 2009. doi: 10.1088/0034-4885/72/7/076901.
- [5] Fredric J. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66(1):51–83, 1978. doi: 10.1109/PROC.1978.10837.
- [6] NumPy Developers. Numpy module, 2013. URL <http://www.numpy.org/>.
- [7] Patrick Brady and Nickolas Fotopoulos. *'Pylal.Frutils Module Documentation'*. URL <https://www.lsc-group.phys.uwm.edu/daswg/projects/pylal/doc/pylal.frutils-module.html>.
- [8] J. Berliner and R. Savage. *Calibration of H1 Coil Actuation Coefficient Via Photon Calibrator For the S6 Science Run*, December 2010. LIGO DCC: LIGO-T1000739.
- [9] Gregory Mendell and Michael Landry. *Stackslide and Hough Search SNR and Statistics*, January 2005. LIGO DCC: LIGO-T05003-00.