Internal Code Review Document:
Search for Gravitational Wave Radiation Associated with the Pulsating Tail of the SGR 1806-20 Hyperflare of Dec. 27th, 2004 Using the LIGO Detectors

Sharmila Kamat
Columbia Experimental Gravity Group
Columbia University

The review of the Matlab code used in the analysis "Search for Gravitational Wave radiation associated with the pulsating tail of the SGR 1806-20 hyperflare of Dec. 27th, 2004 using the LIGO detectors" is undertaken. The review examines the rationale behind the code, various subroutines associated with the code, identifies checks at different stages of the analysis and comments on whether the goals of the analysis is achieved.

I. INTRODUCTION

This document is a review of the algorithm spearheaded by Luca Matone of the Columbia Experimental Gravity Group for the analysis of the Astrowatch data coinciding with the Dec. 27, 2004 hyperflare of the Soft Gamma-ray Repeater SGR1806-20. The version of the code covered by this document is the same, which was distributed to the external reviewers. The analysis focuses on the search for gravitational waves (GW) associated with the Quasiperiodic Oscillations (QPO) observed in the pulsating tail of the SGR1806-20 hyperflare event. This review aims at ensuring that there are no conceptual errors in the search algorithm, that the code executes the search as it is designed to do, free of software errors and bugs, and that it is user friendly and appropriately commented.

II. THE ASTROPHYSICAL CONTEXT

The code analyzes the Astrowatch data taken by the 4 km LIGO Hanford detector (H1) on December 27, 2004. An external trigger received from the GCN network provides information on the electromagnetic signature observed from this event. Follow-up analysis of satellite data (RHESSI and RXTE) revealed the existence of Quasiperiodic oscillations (QPO) in the electromagnetic spectrum at different periods following the flare. The analysis uses the time and frequency information available from the satellite observations to search for gravitational wave signals in the same frequency bands where QPOs were observed in the electromagnetic spectrum. In the absence of a GW signal the analysis sets upper limits on the possible emission of a gravitational waves associated with the QPOs of the neutron star during the event.

The analysis identifies the on-source and off-source data segments. Depending on the QPO being analyzed, the QPOs were observed for periods lasting tens to hundreds of seconds. For the purpose of this review, the QPO associated with the RXTE satellite with a period of 50 s was considered. In this case, an off-source data of approximately 2 hours was used. The analysis examines the off-source data with a view to apply the same code on the signal region, and subsequently looks at the signal region as well. In the analysis

for each QPO frequency the off-source (or background) region is divided into the same time duration segments as the duration of the satellite observation for the given QPO frequency. For example, as in the case of the QPO observed at 92.5 Hz, the data segments chosen were of 50s to match the length of the on-source data to be analyzed The algorithm computes the excess power in the QPO frequency band observed in the H1: LSC-AS_Q channel for the corresponding time segments and uses these information to estimate the sensitivity of the search.

III. CLAIMS OF THE CODE DOCUMENTATION

Documentation associated with the code can be found as follows:

1. A readme file that explains the code structure in terms of routines to be used and gives a brief summary of its installation. Instructions on how to reproduce the on-source results shown on October 3, 2006 are also specified.
2. The APS poster
   (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/G060193-00-Z.pdf)
3. The plenary talk at the LIGO August 2006 Collaboration meeting.
   (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/G060405-00.pdf)
4. The talk given at the Bursts Meeting of October 3, 2006
   (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/SGR_QPO_analysis_update_06oct03.html)
5. A Technical document T060052 that needs to be updated.
6. The SGR1806-20 QPO analysis website:
   (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/SGR1806_QPO.html)

The documentation attached to the code makes the following statements: The code determines an upper limit on GW, the gravitational wave strain incident on the Hanford detector, from the magnetar associated with a QPO of a particular frequency, bandwidth and duration. Though designed to probe the Dec 27, 2004 flare of SGR1806-20, it has been constructed general enough to study other astrophysical events where emission of a quasiperiodic type waveform is expected. The simple and effective analysis retrieves both on and off source gravitational wave channel data; band pass filters it at the desired frequency and bandwidth, optionally removes short transient signals unrelated to any quasiperiodic behavior and calculates the relative excess power associated with the frequency band of interest for the period of analysis.

IV. THE SEARCH ALGORITHM

The search algorithm is given in a flowchart as given in Figure 1, provided by L. Matone as part of the plenary talk for the August 2006 LSC meeting, and summarized in the following way: Configuration Files are created using MakeConfiguration Files for the analysis of a particular QPO. The files include initial parameters such as the frequency, bandwidth and duration of the QPO, the times associated with the on-source and off-source data, veto window and thresholds for vetoing fast signals and glitches, and injection of various types of waveforms in the off-source analysis. Depending on whether it is the signal or background, Astrowatch data is retrieved for the period of interest. Injections are applied to it if it is off-source. The data is conditioned, calibrated and filtered, and fast signals are vetoed. The excess power distribution for the frequency

bands is computed. If the off-source data is being analyzed, the median value of the excess power of the background region is taken as the reference. For the signal region, it is the excess power in the on-source segment. The search sensitivity is found by injecting various types of waveforms and finding the injected power for which the signal is 90% of the time greater than the reference value. The upper limits are determined based on the Feldman-Cousins analysis method by considering any possible excess observed in the signal region.

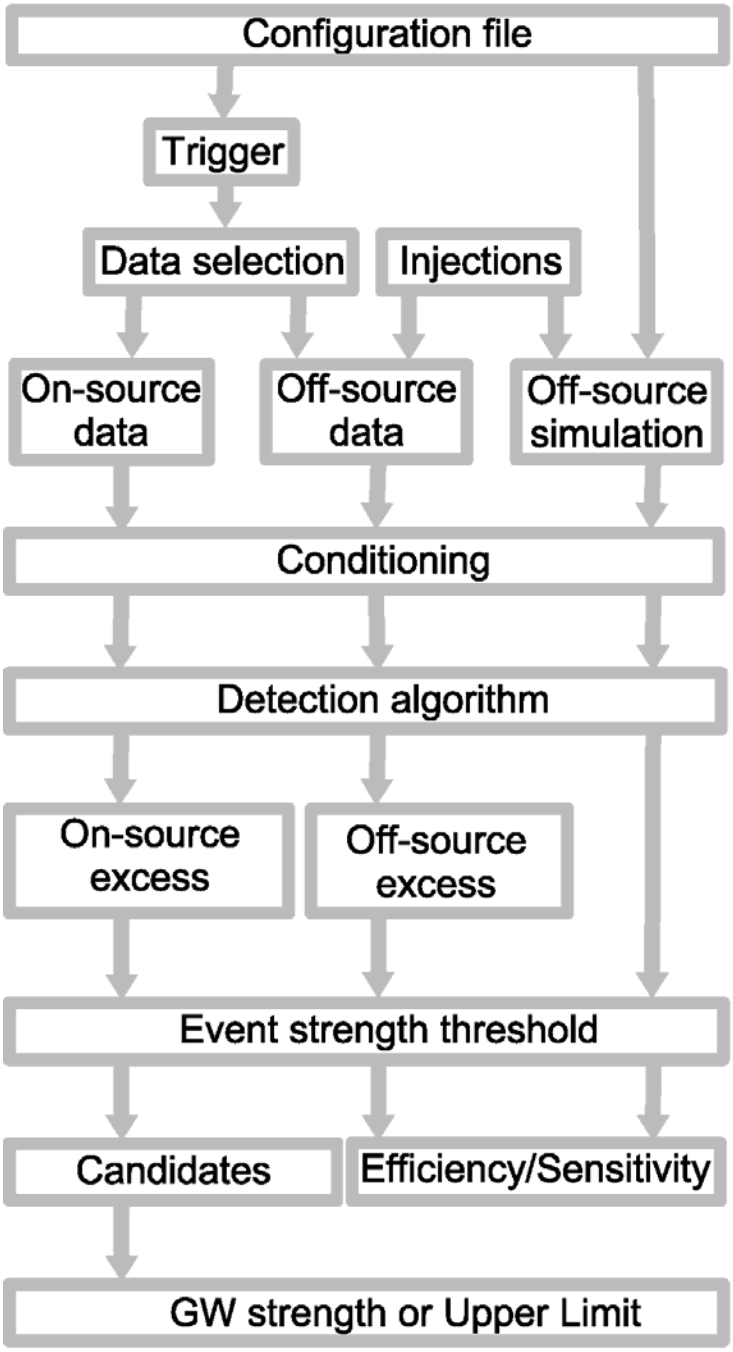


Figure 1 Analysis flowchart

The search algorithm proceeds as follows:

1. MakeConfiguration scripts set every parameter and determine the appropriate software environment for the investigation based on the associated software libraries. The Configuration files produced by these scripts are loaded by every function of the code to ensure uniform behavior. The MakeConfiguration script uniquely determines the behavior of the analysis and fully ensures traceability, repeatability, readability and absolute user friendliness. The configuration file contains every parameter of the investigation, such as the frequency, bandwidth and duration of the QPO event, the buffer time to remove filter transients, injection parameters and duration of the data segments for the analysis. Every investigation of the analysis is fully documented by the MakeConfiguration file and the filter/function libraries and repeatable any time. Further traceability is provided through the DEBUG flag, which initiates frequent and comprehensive dump of internal variables multiple times per function. All parameters used for the to-be-published analysis were the results of far reaching optimization procedures.
2. Access data from the H1: LSC-AS_Q channel (sampling frequency of 16384 Hz) from the only interferometer, H1, operating during the time of the event. Data access relies on the 'frgetvect' utility provided by Benoit Mours/VIRGO.
3. Software injection of a comprehensive and widely varying astrophysically-motivated and "ad-hoc" waveforms were applied to the off-source data prior to the filtering, calibration and the application of the vetoes to study the sensitivity and waveform independence of the analysis. Injections were switched off during background and on-source studies.
4. The architecture is open and flexible, while preserving the ease of traceability and self documentation.
5. The data is band-pass filtered at the QPO frequency for the chosen bandwidth (e.g. 10 Hz) as determined/documented within the Configuration File. Band pass filters were pre-created, verified by hand and they were archived within the Filter library and frozen by the analyst. The IIR digital time domain filters were of the order of 50th – 60th and they were always described / used through their "second-order-section" (SOS) representation to avoid numerical precision pitfalls. Each filtering step use the basic zero phase method described and recommended within the literature. (Digital Filters by Hamming, ISBN 0-486-6508-4, pg 252).
6. The data is calibrated in units of strain for given counts for a given frequency. For this, a response transfer function H1response_788218239.txt valid for this specific segment of Astrowatch data, provided by Michael Landry on behalf of the calibration group, was used. The contents describe the response function of the detector at the time when the flare occurred is used to do the requisite conversion.
7. The analysis next applies an optional veto to remove data segments corresponding to burst-like "glitches" with short durations that cannot be attributed to a quasiperiodic oscillation as dictated by the satellite observations. The veto is applied as follows:
   The MakeConfiguration File sets a choice of veto thresholds and the duration of the time segments (i.e. time resolution of transient search) to be vetoed. Glitches and fast transients that are clearly not associated with quasiperiodic gravitational waves are

then flagged and a temporary descriptor file is saved for traceability. The vetoes are applied to both data types with and without injections to remove such transient signals. The threshold values are found by generating the RMS distribution of the gravitational wave signal for the entire background data and then selecting the value of the threshold that removes fast signals. There is a trade-off between the choice of a safe threshold and the percent of data rejected. In general the amount of the fractional data removed was measurable in the few-percent region having negligible effect on the determination of the average QPO power.

8. Following the application of the vetoes, the power is computed for the off-source data in 250ms segments around f_o (e.g. 92.5Hz) with the set bandwidth (e.g. 10Hz). The same was repeated for the adjacent bands (e.g. centered at 102.5Hz and 82.5Hz). The total excess power was computed via the quadrature sum of the vetoed time segment RMS data included within the desired region (e.g. 50s long) based on both the difference and ratio algorithm. During the background tests and parameter optimization tests the ratio algorithm vas found consistently inferior, therefore the final analysis was executed based on the difference algorithm.
9. The reference to estimate the search sensitivity is provided. In the event of the off-source sensitivity estimate analysis (i.e. first presented at the APS poster publication), this is the median value of the excess power of the background region. For the on-source data (i.e. to-be-PRD publication), it is the excess power in the on-source segment. The final upper limits are constructed via the Feldman-Cousins method and based on the background distribution models with measured parameters and the excess observed within the signal region.
10. The search sensitivity and waveform independence is also determined via injecting different types of waveforms in the data and finding the value of the injected power for which the resulting signal is 90% of the time greater than the median of the background.

IV. THE CODE ROUTINES

Here is a summary of the libraries used and their functions:

1. Calibration Library:

NoCalibrator.m: Returns data as is with no calibration.

DummyCalibrator.m : This is a dummy function that is not being used and has been added so that the code could, in future, seamlessly incorporate other options as needed.

SingleBandCalibrator.m: Converts the data in units of strain/count given the input Calibration file. This is also called SimpleCalibrator.m. This is a relic from past versions of the code and is no longer being used in the new version of the code.

ThreeBandCalibrator.m: Performs the calibration for the three frequency bands of interest, i.e. the QPO band that is chosen and its two neighboring bands. This is the only calibration being used in the present analysis.

2. Data Conditioning Library:

Currently only a dummy routine **DataConditioning.m** exists in this library. Not in use and has been added so that the code could, in future, seamlessly incorporate other options as needed.

3. Filter Library:

SingleBandFilter.m: Filters the input data at QPO frequency of choice at a chosen bandwidth. For the 92.5Hz QPO, this was done at 92.5Hz for a bandwidth of 10Hz. This is from an older version of the code and is no longer being used.

ThreeBandFilter.m: Using the Filter Design and Analysis tool fdatool in Matlab, band pass filters are generated for 2Hz, 5Hz and 10Hz bandwidths by choosing the appropriate frequency. For example in the case of the RXTE observation of a QPO at 92.5Hz, band pass filters were created with width of 10Hz at 92.5Hz and additional 10Hz width band-pass filters were created for the neighboring bands of 82.5Hz and 102.5Hz frequencies. All of the filters are saved in the FilterLibrary / subdirectory.

Filter Files: The .mat files contain the coefficients while the .fda files contain the parameters used in the fdatool function. The ThreeBandFilter.m uses these coefficients in the Filter Library files to filter the input data for the three relevant frequency bands centered at the QPO frequency and at the neighboring bands.

4. Waveform Library:

This contains Matlab scripts to produce a range of waveforms used as injection inputs. Currently the studied waveforms are sine Gaussians (SG), amplitude and phase modulated SGs, exponential decay waveform and white noise bursts. Here is a short description of all the waveform routines:

- AmplitudeModulation.m: generates the time series of an amplitude modulated waveform whose initial parameters such as injection frequency and amplitude, modulation frequency, depth and phase are specified in the MakeConfiguration file.
- PhaseModulation.m: generates the time series of a phase modulated waveform whose initial parameters such as injection frequency and amplitude, modulation frequency, depth and phase are specified in the MakeConfiguration file.
- SineGaussian.m: generates the time series of a Sine Gaussian whose initial parameters such as injection frequency, duration and amplitude, Q factor and injection time are specified in the MakeConfiguration file.
- Sinusoid_ExponentialDecay.m: It generates the time series of a ring down wave (an exponentially decaying sine wave) whose initial parameters such as injection frequency, duration, phase and amplitude as well as decay time are specified in the MakeConfiguration file. This waveform was not used in the analysis.
- WhiteNoiseBurst.m: generates the time series of a white noise burst whose initial parameters are specified in the MakeConfiguration file.

5. MakeConfiguration files:

The input parameters are:

- Gravitational wave channel and sampling frequency: The gravitational wave channel is the data from the H1 interferometer while the sampling frequency is 16384 Hz, the rate at which the channels are sampled.
- Choice of background or signal region based on the observation times of the on and off-source data as provided by satellite observations. For example, based on the RXTE observation of a QPO, the on source data was taken 170s after the flare while the off-source data was taken 400s after the flare and lasted until the end of the lock.
- Duration of the QPO as based on astrophysical observations. As in the case of the RXTE observation, the QPO lasted 50s.
- Frequency and bandwidth of GW data to be analyzed (in this study it is determined by information from astrophysical observations on the QPOs); For example, in the case of the RXTE observation, the QPO had a frequency of 92.5 Hz. A 10Hz bandwidth was considered for this analysis.
- Frequency and bandwidth of the adjacent bands to be studied for computing the excess power; In the case of the RXTE observations, as the QPO was at 92.5Hz, adjacent bands at 82.5Hz and 102.5Hz were considered.
- Buffer Time. This is the time allowed to ensure that all filter transients are removed. For this analysis, a buffer time of 10 seconds was deemed sufficient.
- Data flag that allows testing of the code by replacing LIGO data with white Gaussian noise.
- Veto window and veto thresholds. The veto window is defined as the segment duration under which the RMS is measured. The veto threshold is set as a level and power is checked to be above or below that threshold. If it is above the threshold, then the segment of that particular veto window is tagged. For the 92.5Hz QPO, veto window segments were chosen to be 125ms long and the veto threshold set at 2 sigma.
- Injection parameters such as the duration, frequency and amplitude of a waveform such as a Sine Gaussian that is injected into the background distribution.
- Selects the band pass filter that chooses, for example, the 92.5Hz filter with a 10Hz bandwidth, algorithms and calibration files to be used.
- Algorithm: The difference algorithm takes the difference in power between the power in the frequency band of interest and the average of the power in the two neighboring bands given by (P_{qpo} - P_{avg}).
- The amplitude of the injected waveform is found when the resulting signal is greater than the reference 90% of the time where 90% = (1- Miss Probability), where MissProbability = 10% as set in the Configuration File. The Target Precision is a value also set in the Configuration File and gives the level of precision desired between the reference value and the signal.

Certain representative MakeConfiguration Files have been included such as:

- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_2sigma_AM_100m Hz_1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_2sigma_PM_100mH z_10.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_2sigma_RandomNoi se_40s_1Hz.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_2sigma_SG_Q1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_3sigma_SG_Q1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_125ms_4sigma_SG_Q1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_1s_2sigma_SG_Q1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_1s_3sigma_SG_Q1e6.m
- MakeConfiguration_OffSource_92.5Hz_10Hz_50s_diff_1s_4sigma_SG_Q1e6.m
- One representative example:

MakeConfiguration_OffSource_92.5Hz_10Hz_diff_125ms_2sigma_SG_Q1e6.m, illustrates the naming scheme: first whether it is signal or background, then the QPO frequency, followed by the bandwidth, followed by the choice of algorithm (ratio or difference), duration of time segments vetoed (such as 1s or 125 ms), choice of threshold for vetoing fast signals (such as 2σ, 4σ, etc.), choice of injected waveform (such as Sine Gaussian) and choice of the Q value of the injected waveform. In the case of phase and amplitude modulated waveforms or white Gaussian noise, the corresponding labels of AM, PM or Random Noise are given in place of SG. These files can be used to reproduce the results for the off-source study.

In addition to these files, there are also files such as MakeConfiguration_EXAMPLE_RandomNoise.m and MakeConfiguration_EXAMPLE_SG.m which aid the user in setting the parameters.

6. Evaluation of the background excess power distribution:

- **getData.m**: Uses **frgetvect** to get the gravitational channel data for a given time period at the QPO frequency and bandwidth. A buffer time allows for selection of data with a sufficient extra time at its ends to avoid transients due to filtering. Relies on **GenerateWaveform.m** to generate the injected waveform of particular amplitude if required, **FilterData.m** and **CalibrateData.m** to filter and calibrate the data. Also uses **DataQuality.m** which uses **genRMS.m** and **genVeto.m**
- **GenerateWaveform.m** generates a waveform and calculates the energy and power associated with it using **calcEnergy.m** and **calcPower.m** and reverse calibrates it.
- **FilterData.m:** Band passes the data at the required QPO and neighboring frequencies using the filters in the Filter Library. For example for the RXTE observation of a QPO at 92.5 Hz, the data is band passed at 82.5Hz, 92.5Hz and 102.5Hz at bandwidths of 10 Hz.
- **ConditionData.m:** Allows for conditioning data. Currently a blank routine.
- **CalibrateData.m:** Calibrates the input GWchannel data in units of strain. This is based on the calibration file H1response_788218239.txt.

- **CalculateRMS.m**: Uses **getData.m** to get the LIGO data (or random white noise for the case of the Monte Carlo) and **genRMS.m** to generate the RMS distribution for the GW channel data for time segments of a specified duration and frequency.
- **GenerateDistribution.m**: Uses **CalculateRMS.m** to generate the RMS distribution for the three bands for the given time segment and duration.
- **ExcessPower.m**: Uses **GenerateDistribution.m**, **DetermineExcess.m,** and **GeneratebandAvg.m** Generates the RMS distribution of the gravitational wave signal for a set of frequency bands (band in question and two neighboring bands) and subsequently the power in the three bands. The average of the power in the two neighboring bands P_{avg} is computed. The excess is determined through the difference algorithm which selects the excess as $P_{qpo} - P_{avg}$. For each segment the mean of the excess power is computed for a given number of tiles (sub-segments) ignoring the NaN (vetoed) values.
- **EstimateBG.m**: Uses the ExcessPower.m routine to generate the excess power as specified earlier.

7. <u>Estimating the sensitivity of the search:</u>

- **ExcessPower.m**: Specified earlier
- **EstimateReference.m:** selects choice of reference, median if off source data, signal region if on-source.
- **CriteriaResponse.m**: Finds the probability distribution function (PDF) and cumulative distribution function (CDF) of the excess power in the backgrounds with and without injections and applies a two-Gaussian fit to the two distributions using **Frankenfit.m** and **FrankenGauss.m.** The difference between the reference and the set value of the Miss Probability (set as 10% as a choice in the Configuration File) is determined and compared to Target Precision as also set in the Configuration File. If greater, it goes through a second iteration after increasing the amplitude of the injection by 10%. If the target precision is still not met, the routine initiates a procedure to find the necessary injection amplitude to satisfy the criteria. The sensitivity of the search is determined in terms of the injected waveform h_{rss}.
- **Frankenfit.m** and **FrankenGauss.m:** This is a custom-created fit to model the data, which has been created only for this application. In essence, it fits a Gaussian to one part of the distribution and another Gaussian to the other part of the distribution. The advantage of this custom fit is that it models the tails of the distributions to a reasonable extent.
- **DetermineAmplitude.m:** The function **DetermineAmplitude.m** determines the amplitude of the injection by linearly fitting the data contained in the file FileNamefit which contains past values of the injection amplitude and the value of DeltaP which is the difference between the reference and the response. The objective is to find the injection amplitude A to minimize DeltaP.
- **EstimateUpperLimit.m:** This uses **ExcessPower.m, EstimateReference.m, DetermineAmplitude.m** and **CriteriaResponse.m** as specified above to determine the sensitivity of the search.

8. Creating the vetoes:

- **createVeto.m:** This routine identifies thresholds in order to tag segments associated with the presence of fast signals. It is independent of the main code and is used prior to its running. The function takes as input a set of RMS values (for either the band of interest or the adjacent bands) calculated in an arbitrary time period, usually 125ms or 1s long periods. The Configuration File contains all the parameters that generated the RMS data stream. The createVeto.m routine then uses the function sigmaFind.m to calculate the standard deviation of the underlying distribution and determines the threshold corresponding to the thr1 cut (thr1 is in units of sigma and is one of the function's input parameters) expressed in h_{rms} [strain] units. At the same time, all of the RMS segments which are above threshold are dumped into a file. The information provided can then be used in a MakeConfiguration File which generates the Configuration Files required for the running of the code.
- **sigmaFind.m :** For a given time series, this routine returns the standard deviation of the underlying distribution by iteratively fitting it with a gamma distribution while neglecting Nsigma outliers. This information is then passed on to the createVeto.m function.
- **DataQuality_Auto.m:** This routine applies vetoes to the data by setting the thresholds as specified in the MakeConfiguration Files. Segments identified with the presence of a fast signal are tagged with NaNs. The routine uses **genVeto.m** for identifying outliers
- **DataQuality_Files.m:** This routine applies vetoes to the data by making use of veto files specified in the MakeConfiguration Files which identify the segments associated with fast glitches.
- **DataQuality_NoVeto.m:** This routine can be specified in the MakeConfiguration Files when no vetoes are to be applied to the data.
- **genVeto.m:** This routine applies vetoes to the data by identifying outliers as specified earlier.

V. TESTS CARRIED OUT FOR THE SEARCH CODE ROUTINES

The checks on the search code routines are aimed at ensuring:

- That the data is processed correctly.
- Inputs such as parameter files and calibration files are applied correctly.
- If some part of information is not supplied, code should fail.
- Loops are implemented appropriately.
- Ran the entire code to ensure repeatability of results.

Checks on the Calibration Library:

Data was given as input and the output checked for the **SingleCalibration.m** and **ThreeBandCalibration.m** to ensure that it was properly converted based on the LIGO Calibration Response File H1response_788218239.txt. Note that this constant calibration factor is applied over the whole two-hour interval. It was assumed in the code that the same calibration factors are valid over the period of interest and that there is need to use the provision available to propagate the calibration to particular times.

Checks on the Filter Library:

Here we checked for the bandwidth of the filters and a proper attenuation of the 60Hz line (and their harmonics).

To check for the bandwidths we used the filters created for the 92.5Hz QPO:

BP_IIR_Butter_92.5Hz_2Hz.mat
BP_IIR_Butter_92.5Hz_5Hz.mat
BP_IIR_Butter_92.5Hz_10Hz.mat
BP_IIR_Butter_82.5Hz_10Hz.mat
BP_IIR_Butter_102.5Hz_10Hz.mat

(only the 10Hz wide filters were used in the SGR analysis). We generated white gaussian noise of variance 1 cnt (amplitude spectral density of sqrt(2/Fs) = 1.105e-2 cts/rHz and shown by the blue curve of Figure 2) and we fed it to 92.5Hz 10Hz wide filter. The resulting filtered time series should have an RMS of sqrt(10Hz) * 1.105e-2 cts/rHz = 3.49e-2 cts. The measured RMS from the filtered (green curve in Figure 2) signal was found to be 3.491e-2 cts/rHz, in good agreement with expectations. Furthermore we numerically calculate the RMS of the input signal in the band of interest (red crosses in Figure 2) and we measured 3.489e-2 cts/rHz, again in very good agreement with the filter output. These results are shown in Figure 2 and Figure 3. For the other filters we found a similar agreement.

At 60Hz and harmonics, there is an attenuation of more than 9 orders of magnitude.

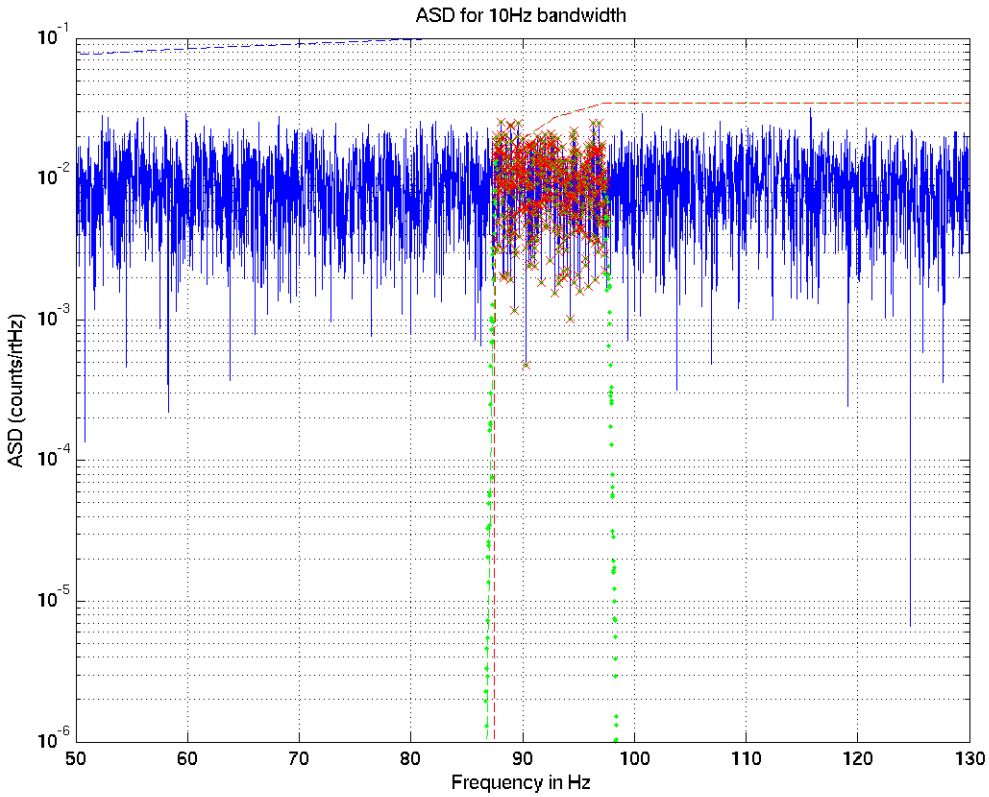


Figure 2: ASD of unfiltered (blue), filtered (green) and expected (red) values for white noise band-passed with 10 Hz bandwidth

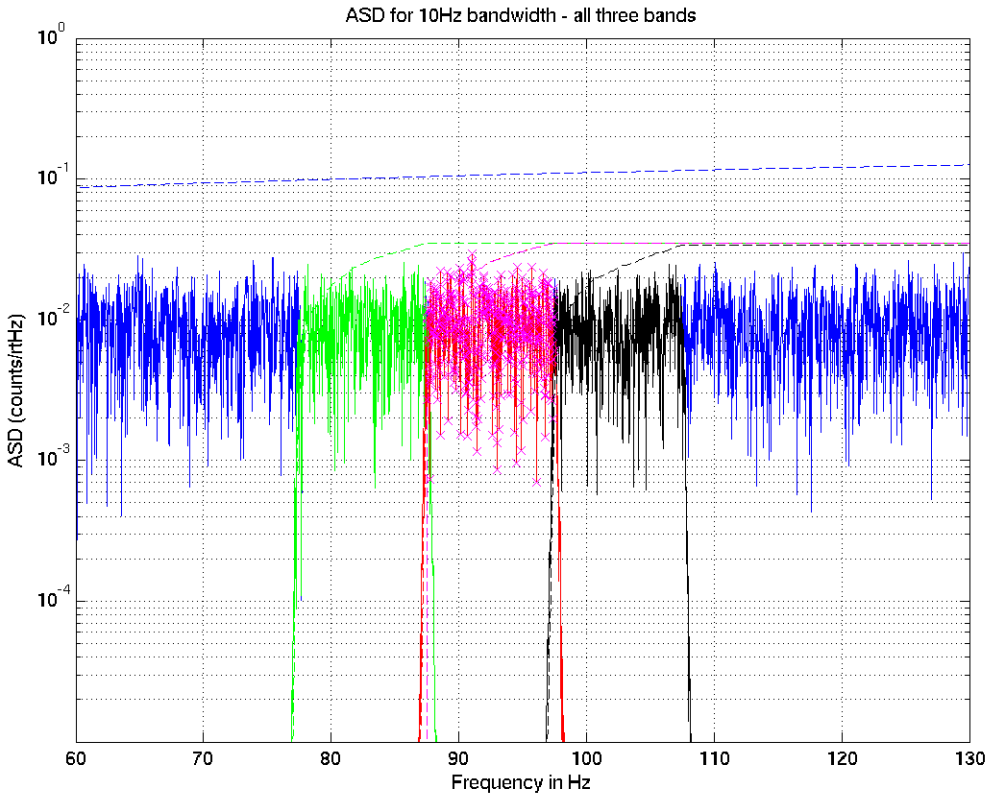


Figure 3: ASD of unfiltered (blue), filtered (green for 82.5Hz, magenta for 92.5 and black for 102.5) and expected (red crosses for 92.5Hz) values for white noise band-passed with 10 Hz bandwidth.

The Buffer Time which allows for the removal of filter transients is set to 10 seconds (BuffTime in the MakeConfiguration file). This seems to be a reasonable choice, according to the fdatool impulse response calculator.

Checks on the Waveform Library:

It was confirmed that the **GenerateWaveform.m** routine generates the waveform to be injected based on the options available such as amplitude or phase modulated waveforms, sine waves and sine Gaussians and white noise burst waveforms. In each case, I generated the injection given the parameters specified in the Configuration File and ensuring that the time series of the injected waveform is produced correctly. For example, in the case of an injection of an amplitude modulated waveform, a time series of such a waveform is generated for a given modulation frequency and amplitude. The parameters were changed such as modulation amplitude and it was verified that the generated waveforms reflected the changes.

The routines **calcEnergy.m** and **calcPower.m** were confirmed to work by feeding in a simple input distribution and ensuring that the output calculates the energy and power of

the distribution as defined in the routines. To call these routines, it is necessary to use the Matlab command *addpath(genpath(pwd))* to ensure all routines in the Waveform Library are explicitly seen.

MakeConfiguration files

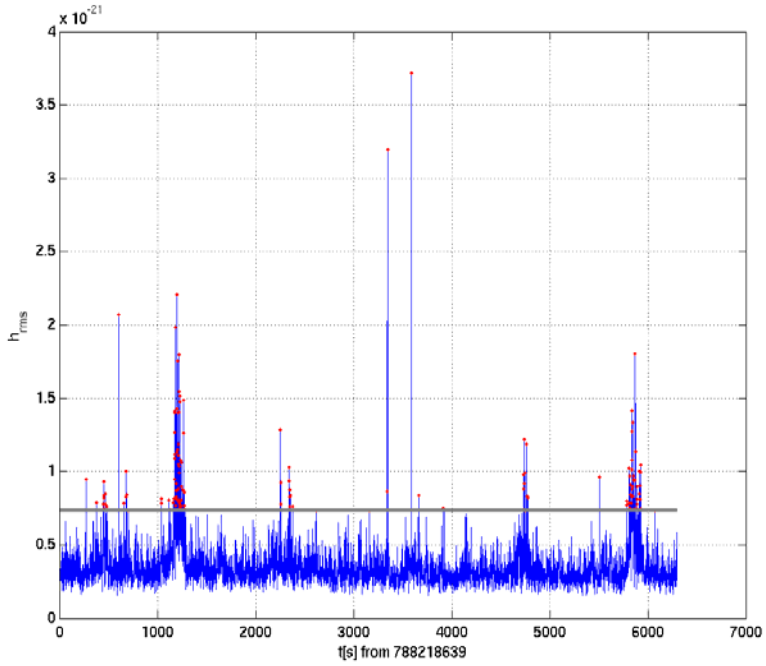
It was determined that the code uses the right configuration file with the correct parameters within as generated by the MakeConfiguration routine. It selects the specified time interval and period, selects white noise for random data, the right vetoes, and selects the specified injected waveform, filters and analysis algorithm. For each one of the options, I physically checked that this was the case. For example, for the injections, I checked that the background data was generated with and without injections. I made sure white noise was generated when the Monte Carlo flag was set, that the times chosen coincided with the intervals set in the MakeConfiguration files and that the right filters were chosen based on choice of QPO frequency in this file. Data was fetched using the frgetvect function and then compared with the data fetched using getData.

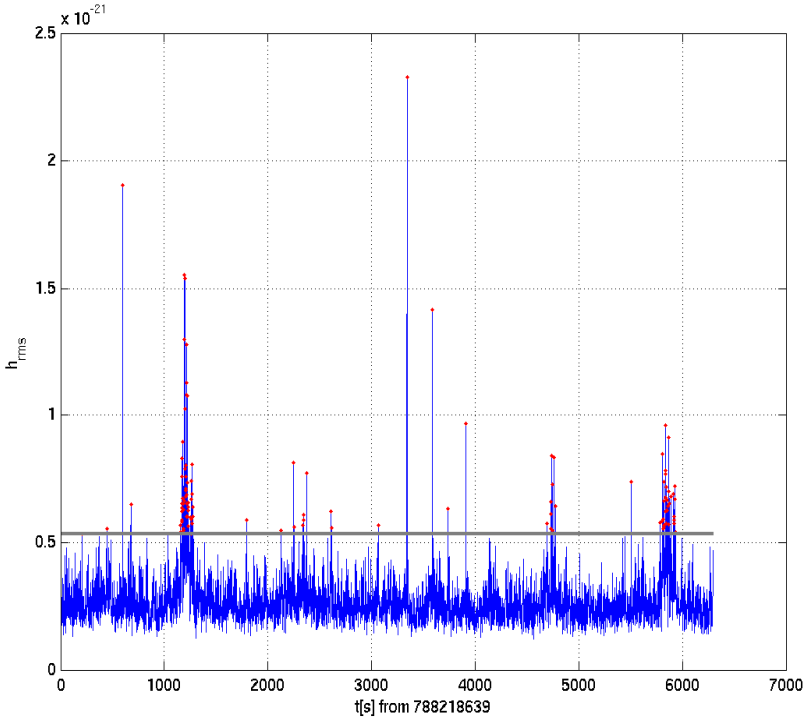
The version of the code has comments explaining choice of all parameters. A README file provided as part of the code also gives instructions on running the code. Example MakeConfiguration Files for adding different injected waveforms are provided. For generating Random Noise, it is MakeConfiguration_EXAMPLE_RandomNoise.m

Retrieving data and generating the RMS and hence excess power distribution for the background data

getData.m

- Checks done to ensure data flag correctly selects Gaussian white noise or LIGO data.
- LIGO Frame Data was retrieved for arbitrary GPS times with the same calibration parameters and checked that it was retrieved, filtered and calibrated correctly.
- Checked that data of the specified times was retrieved.
- The provision for allowing different types of injections into gravitational wave data was tested for different injection parameters.
- Ensured that data quality flags were applied correctly when the vetoes are determined by the threshold values in the MakeConfiguration Files or by using data quality veto files. Checked that NaNs are correctly assigned to vetoed periods. If no vetoes are applied, the MakeConfiguration file selects the DataQuality_NoVeto.m routine which passes the data unaltered.
- Tested that the choice of thresholds specified in the MakeConfiguration File for a combination of veto window and N sigma value was correct. For example, the thresholds for the 82.5 Hz, 92.5Hz and 102.5 Hz bands and 1s data segments the veto window was given to be [7.3269e-22 5.3416e-22 4.3899e-22]; I created the vetoes for 1s segments and found the thresholds as shown below in Figure 4, they match up to the ones specified in the configuration files.


x 10^-21
h_rms
t[s] from 788218639


x 10^-21
h_rms
t[s] from 788218639

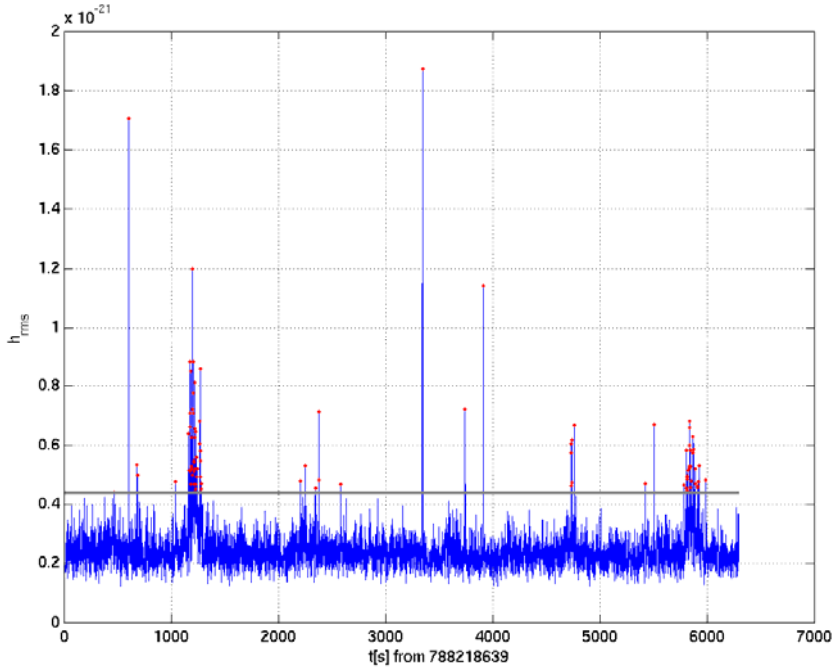


Figure 4: Four-sigma Thresholds to select 1 sec segments of data to be vetoed at 82.5Hz, 92.5Hz and 102.5Hz bands.

CalculateRMS.m:
The routine CalculateRMS.m is used to generate an RMS distribution of the input signal. It was tested against a small script which calculates the RMS values: for instance white Gaussian noise (ASD of 1e-22/rHz) was fed to the test script and the RMS of the time series is calculated each 250ms segments.

In the second case, white Gaussian noise is fed to CalculateRMS by setting the MonteCarlo='true' flag in the Make Configuration file (ASD of noise set to sigmaMonteCarlo=1e-22 strain/rHz). The RMS distribution computed by the test script and the distribution computed by CalculateRMS.m are then plotted as histograms to find their mean and their spread. The two RMS distributions are found to agree very well. This served as a sanity check of the part of the code that does the actual computation of the RMS.

The mean as computed from the distribution generated by CalculateRMS.m was found to be 3.19e-22 strain. This agrees well with the theoretical expectation of the RMS as sqrt(10)*1e-22 strain/rtHz.

The same procedure was executed in case of injections. In this case we also found good agreement.

The expected power and energy for an input of white Gaussian stationary noise was verified using a test script that computer the values in comparison with the output of this routine.

GenerateDistribution.m
The same tests were done as in CalculateRMS.m for the general case when the entire background data was considered. An RMS distribution of fake data in the form of white Gaussian noise was also generated.

DetermineExcess.m The correct evaluation of excess power was checked for both ratio and difference algorithms. For the difference algorithm, for the case of the 92.5Hz QPO, the average of power in the adjacent bands is consistently in excess of the band of interest. In this case, the excess power term is negative which arises as a consequence of the power being consistently higher in the average of the adjacent bands as compared to the band of interest. Since power spectrum of the noise is convex, it is expected that the average of the adjacent bands will exceed the power in the QPO band which is what is being observed.

GeneratebandAvg.m: Generated the structure band which contains the power in the three bands computed in 250ms long segments produced by my own MakeConfiguration File. This structure was sent as input to GeneratebandAvg.m to calculate the average power in 50 second long segments. I compared the resulting average manually with simply computing the average power. I found them to agree exactly.

ExcessPower.m: The routine uses **GenerateDistribution.m**, **DetermineExcess.m** and **GeneratebandAvg.m** whose checks are summarized above.

Another check I did was to give white Gaussian noise as input data by setting the MonteCarlo flag in the MakeConfiguration file. The ExcessPower.m routine generates the power distribution in QPO band and the two adjacent bands. Since we are feeding in white noise, we should expect the distribution of the power in the three bands to agree. I verified this by plotting a histogram of the three power distribution and find good agreement for the three means. As the ASD of noise is set to 1e-22 strain/rHz and the theoretical RMS is computed to be 3e-22 strain (as explained above), we expect the theoretical value of the mean of the power to be 9e-44 which agrees well with the means of the three distributions.

EstimateBG.m: This routine essentially takes the parameters such as Tstart and Tend, dtqpo, etc set in the MakeConfigurationFile and inputs them in the ExcessPower.m routine. For completeness, I ensured that the parameters were fed in correctly to ExcessPower.m

Estimating the sensitivity of the search:

EstimateReference.m :
Checked that median of background correctly selected for Background region or that the signal is selected for the On-Source region.

ExcessPower.m: The tests are summarized in the earlier section

DetermineAmplitude.m
The function **DetermineAmplitude.m** determines the amplitude of the injection by linearly fitting the data contained in the file FileNamefit which contains past values of the injection and the value of DeltaP which is the difference between the reference and the response. The objective is to find the injection amplitude A to minimize DeltaP. I checked that the data was fitted properly to a line and that the best estimate for the amplitude was given.

CriteriaResponse.m

- Checked that the right algorithm chosen for excess power based on the MakeConfiguration File.

- The time taken for the routine to converge depends on the initial choice of the injection amplitude.

- For the sine-Gaussian with Q=1e6, the routine converges based on the initial amplitude. As the value of Q is decreased, the waveform tends to behave more like a burst than a sinusoid. For Q=6e2, the routine sees the injection as a burst and treats it as a fast signal and vetoes causing the routine to take a very long time to converge. This helps us understand how sensitive the pipeline is to recognizing burst like injections. For the RXTE 92.5Hz QPO frequency considering the 125ms - 2 sigma case with an injection of a sine-Gaussian with Q=1e6, the routine converged with one iteration while for the case of Q=6e2, it had a very hard time converging.

- It is to be noted here that the number of iterations and the time taken for convergence depends on the Target Precision and the initial choice of the amplitude of the injection. The choice of A is based often on prior knowledge based on having found convergence in earlier cases and noting the A value accordingly. As a simple check, I changed both A and later Target Precision for the case of the 92.5Hz QPO with an injection of a SineGaussian with Q=1e6 and as expected the convergence took many more iterations than the single iteration I see with the present values of A and Target Precision as used for generating the results for this QPO.

Frankenfit.m and FrankenGauss.m: Verified the fitting to the distribution of the excess power in the background data with and without injections. For example for the case of the 92.5Hz QPO from the RXTE observation for a sine Gaussian with a Q of 1e6, the fits to

the background with and without injection based on the custom-made Frankenfit appeared to track the tail of the distribution adequately.

Checking the optimization of the veto thresholds - CreateVeto.m :

The objective for the test is to find the data rejected for different values of the veto threshold. In the analysis, a 125 ms 2 sigma threshold cut was chosen for the vetoes (always with respect to the 92.5Hz QPO). This choice is based on the search sensitivity to a particular waveform as well as the amount of data rejected. My study was aimed at determining the amount of data rejected for each choice of the threshold and to hence determine the amount of data rejected for the cut used in the analysis.

For this study, I generated white noise and calculated RMS in segments of 1s and 125ms duration. Subsequently using createVeto.m, I found the thresholds corresponding to a range of 4sigma to 8sigma cuts. Figure 5 shows the application of veto thresholds 4sigma and 8 sigma to white Gaussian noise for 1s data segments. When a segment is tagged as having a fast signal, the entire segment (or either 1 s or 125 ms) is rejected. The data rejected by each of these cuts is then computed. Similar test were performed on LIGO data. The entire off-source GW channel data at 92.5Hz with a bandwidth of 10Hz is divided in 1s and 125 ms segments and the RMS distribution computed as before. As before, I found the thresholds corresponding to 4sigma to 8 sigma cuts and calculated the data rejected by these cuts.

Figure 6 shows the data rejection as a function of sigma for the union of the vetoes applied for three bands for both the 1s and 125ms segment case. As can be seen from the plot, the data rejected for a 2 sigma, 125 ms case (the cut used for the analysis) is about 10%.

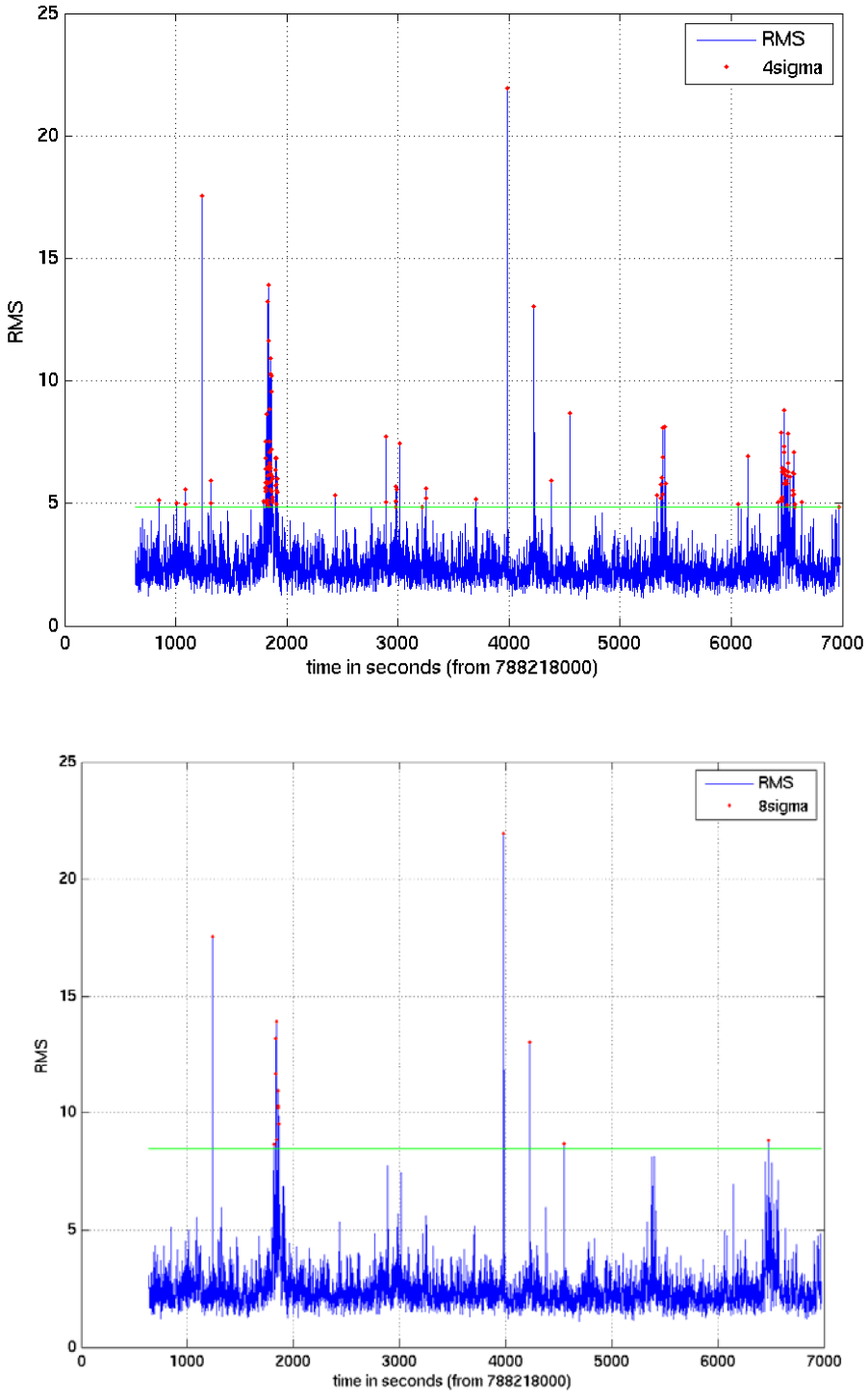


Figure 5: Rejecting 4σ and 8σ outliers for white Gaussian noise for frequency of 92.5Hz and a bandwidth of 1Hz.

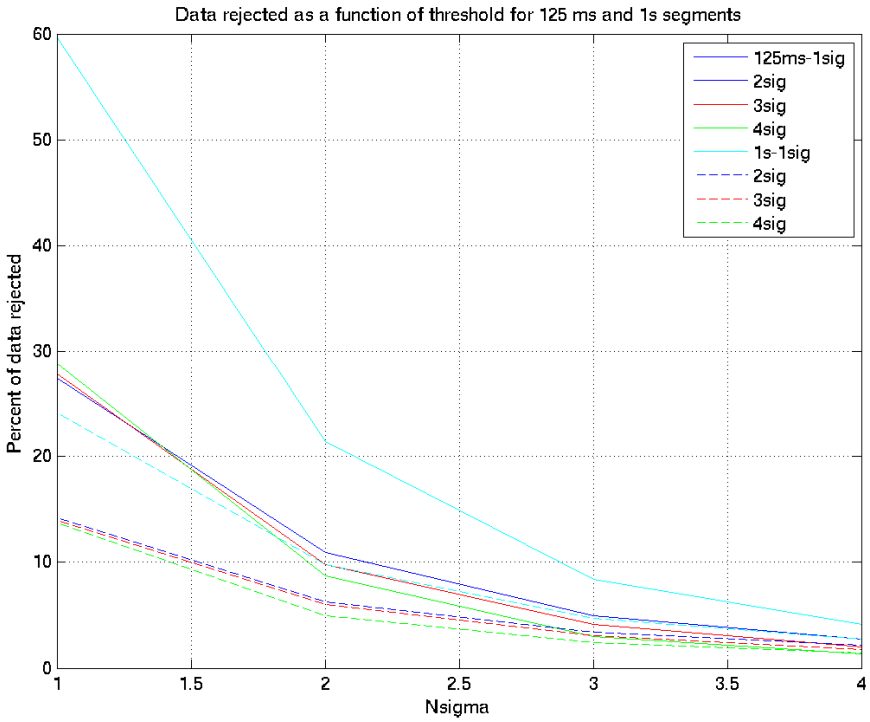


Figure 6 : Data rejected as a function of threshold for 125ms and 1s segments

DataQuality_Auto.m, DataQuality_NoVeto.m and DataQuality_Files.m:

The DataQuality_NoVeto.m check was done as follows: In the MakeConfiguration File, the DataQuality_NoVeto.m is selected. LIGO data for the background is taken as the input. The data before and after applying the vetoes was checked using the getData routine. The comparison showed they were identical with no segments are tagged as NaNs as is expected from this routine.

The DataQuality_Auto.m check was done as follows: The DataQuality_Auto.m was selected in the MakeConfiguration File. Next white Gaussian noise is given as the input data by selecting the MonteCarlo flag in the MakeConfiguration File. The thresholds are set for 2sigma for 125ms data segments for white Gaussian noise using createVeto.m. The output of getData.m, which is a time stream, is saved before and after the application of the vetoes.. The vetoed data is checked to ensure that the NaNs are applied correctly for the vetoed segments. The amount of data rejected for an input segment of 2000 seconds by applying the veto thresholds is computed and is found to be 10.3% for the union of the three bands.

Similarly LIGO data is given as input to getData.m and the thresholds relative to the 2sigma, 125 ms segment cut for the LIGO data applied to it. The data, which is the time stream output of the getData.m routine, is compared for the cases of before and after the

application of vetoes. The data rejected, also for a 2000 second segment, is found to be 10.7% for the union of the three bands in agreement with the values obtained using the createVeto.m routine as shown earlier.

Checking the veracity of the off-source results:

The results provided from the search analysis by L. Matone were verified against what the code produces given the same initial conditions. I did the following:

- To the skeletal MakeConfiguration file I fed the input parameters based on the routine being tested, i.e. the QPO frequency and duration based on the various observations,
- In the case of feeding white noise in place of LIGO data, I set the Monte Carlo flag to true.
- The choice of injection was set by selecting the right waveform from the waveform library and feeding in the right choice of parameters. I wanted to verify one of each of the sets of results for different injections, such as for the case of an Injection of a Sine Gaussian with Q = 1e6 into the off-source data for a veto window of 1s, a threshold at 4 sigma and Target Precision = 5e-46 set in the Make Configuration File, I arrive at a sensitivity of 5.9e-22 strain/√Hz. The results for 1s 4 sigma case give a mean value of 6.2e-22 strain/√Hz value obtained for the search analysis which is within 5% of this value.(see plots at http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/SearchSensitivity.html)
- For the case of an Injection of a Sine Gaussian with Q=1e4 into the off-source data for a veto window of 125ms and a threshold at 2 sigma set in the Make Configuration File, I arrive at a 4.85e-22 strain/√Hz while the values determined earlier ranged from 5.1 e-22 strain/√Hz which is within the fluctuations due to random error and the different choice of random seed. (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/G060405-00.pdf) .
- I similarly checked for the phase modulated waveforms: case of an injection of a phase modulated waveform with 100 mHz modulation frequency and 1 Hz_peak modulation depth for the veto window of 125ms with 2 sigma threshold. In this case, the values obtained are within 5% of the values quoted in the search analysis presented at the LSC meeting as given in (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/G060405-00.pdf) provided the Target Precision is selected correctly. The time taken to converge depends again on the choice of initial conditions.

ON SOURCE RESULTS:

For the on-source analysis checks, I did the checks in two different ways. For the case of say the 92.5Hz QPO seen for a 50 second duration, I determined the excess power in the on-source segment. I then applied a Sine-Gaussian injection with high-Q (1e6), I set the Reference parameter to SignalRegion and I calculated the resulting sensitivity.

I then checked this result against the one using the Feldman-Cousins statistics. In this particular case I found the two upper limits to agree within the experiment uncertainties. The fact that both approaches provide a similar result is reassuring.

I also manually reproduced backgrounds for the 626.5Hz QPO and I was able to reproduce the upper limits provided by L.Matone (see extended burst telecom of Oct.3rd, 2006).

Addressing the question of different timings of the 92.5Hz flare

- I looked at the 92.5Hz flare for three different sets of observed times, 170-220 s after the flare (coinciding with RXTE observations), 150-260s after the flare (coinciding with the RHESSI observations) and from the moment of the flare.
- For the QPO seen from 170 – 220 seconds after the flare, I generated the sensitivity of the search looking at the background data and arrived at a number 4.998e-22 which is within 5% of the value produced in the original results. (http://geco.phys.columbia.edu/~matone/DataAnalysis/QPO/SGR1806-20/doc/G060405-00.pdf) This is within the random fluctuations and is attributed to the choice of a different random seed as compared with my results and those generated for the search analysis.
- On performing the on-source analysis using the Feldman-Cousins functions (to account for the fact that sometimes the on-source value may fall in the rising edge of the distribution leading to attendant problems in choice of injection), I get the same value 5.52e-22 as do the results reported by L. Matone at the plenary talk.
- I then tried the same for the two other time durations. For the case of the RHESSI observation of a 92.7Hz QPO between 150-220 seconds after the flare, I get 3.5137e-22 strain/√Hz sensitivity using the Feldman Cousins lookup table methods and for the QPO from 0-260 seconds after the flare, I get 7.2715e-22 strain/√Hz. The value is identical with the ones reported.
- I verified the Feldman Cousins look up table given in the routine FeldmanCousins.m with the values quoted in 'Unified approach to the classical statistical analysis of small signals,' by G. H. Feldman and R.D.Cousins, Phys.Rev.D, 57, 3873 (1998) to ensure that the table was reproduced correctly. First I verified that the numbers in the table in the FeldmanCousins.m routine matched the numbers in the table in the PRD paper. Next I checked the function by checking if it provided the right upper limit for a given value of mean and standard deviation.

Conclusions: The review of the Matlab code used in the analysis "Search for Gravitational Wave radiation associated with the pulsating tail of the SGR 1806-20 hyperflare of Dec. 27th, 2004 using the LIGO detectors" was undertaken. The various subroutines were tested, the rationale behind the code examined and it was verified that the code executes the search as it is designed to do, that it is user friendly and appropriately commented.