LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note	**LIGO-T11XXXXX–vX**	2015/07/06

Report II — LIGO — SURF — 2015 : Quantization Noise in Digital Control Systems

Ayush Pandey

California Institute of Technology
LIGO Project, MS 18-34
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project, Room NW22-295
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352
Phone (509) 372-8106
Fax (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189
E-mail: info@ligo.caltech.edu

http://www.ligo.caltech.edu/

1 Quantization Noise Analysis in Digital Filters: Introduction and Background

In any system, when a reference input signal is given, the output produced is dependent on the system parameters. This is usually undesirable. To get the desired response, a controller is used to control the output by using the sensor data as feedback.
In digital systems, the signals are a sequence of binary numbers. Once the actual output from the system (plant) is measured by a sensor and is made available in digital format (using Analog to Digital Converters(ADCs)) all comes down to the controller (also called a filter in digital controls) to perform mathematical operations to the digital signal such that the desired response is achieved. It is these mathematical operations that determine the so-called structure of the filter. Usually, some additions and multiplications are performed in a particular order so as to achieve a filter design. Now, as is obvious, which mathematical operations to perform? and in which order? can be of infinite types. This leads to infinite types of filter structures for a given transfer function. Each type has its own merits and demerits. These merits and demerits are measured with respect to some physical quantities such as the cost of the filter, time the filter takes to complete all its mathematical operations, the closeness or the accuracy with which the output is achieved and so on. One physical quantity closely related to the accuracy in the output is called quantization error. The more the error, the worse the output response. Quantization error occurs due to various reasons and has been described in detail in part I of this report [1].
This report describes in detail the work done to analyze this quantization noise in the digital filters at the Laser Interferometer Gravitational Wave Observatory (LIGO). More details about the LIGO project and about the Quantization noise as a bothering noise source in the gravitational wave measurement can be found in [1].
The report is divided into four major parts. This is the first part giving a background and literature review of the work being executed. The following part discusses problems with the quantization noise analysis tool that was already existent and also describes the improvements made. The 40m Caltech's prototype interferometer has been considered first and then the LIGO sites are considered. The final part summarizes the results and conclusion along with a short discussion on the work that shall follow in this project.

1.1 Quantization Noise Measurement

The obvious and the best way to measure the quantization noise is to subtract the digital output (the signal having the quantization noise) from the ideal output. Ideal output is an output which would be the output from the system if the digital system was able to perform its calculations using infinite precision. Practically, such an ideal output is never possible and hence perfect calculation of quantization noise is not possible. Though it is possible to estimate the quantization noise to a level of approximation which can be considered as almost ideal. The way to do is to subtract the output from another output which is calculated using a higher level of precision than the original output. For example: A digital filter produces output at 16 bits, i.e. the precision of its numbers is 16 bits. To estimate the quantization noise for this digital filter, the output of the filter needs to be calculated at a precision better than 16 bits. So, for example if the output is calculated at 32 bits then the difference

between the two outputs would be the quantization noise that was present in the original output. This is the exactly the process by which quantization noise is being measured in this project.
LIGO digital controller works at IEEE double precision format [2] which has typically 64 bits to hold a number. So, the aim here is to calculate the same output using the long double precision which is better than double for most [3] compilers, and then subtracting the two outputs, which would result in the quantization noise occurring in double precision filter implementation.

1.2 Signal to Noise Ratio (SNR) Dependence on Digital Filter Structure

Two brilliant references describing how quantization noise effects change with changing filter structures are:-

1. Widrow Book on Quantization noise [4]
2. A book on Digital Signal Processing by Oppenheim and Schafer [5]

While the Oppenheim book takes a path of explaining via very basic digital signal processing concepts, the Widrow book covers the floating point quantization noise analysis which the former does not. In this project, floating point quantization is of prime concern as all calculation done in LIGO digital controller are in floating point. There are various advantages and disadvantages of having a floating point digital system which is covered widely in the literature. The SNR dependence goes to the level of - order in which the mathematical operations are performed and also on the order in which various sections of a higher order filter are arranged.
Though this project does not explicitly focus on analyzing different digital filter forms and compare them for quantization noise since it has already been done for the LIGO digital control system and the results are summarized in [6]. The direct form II is most widely acceptable form of a digital filter which provides the best noise results trading off with the complexity of the filter. But, as [6] shows that for a slight compromise on complexity (one more addition operation) a much better noise performance can be achieved.
This project takes a peek at and tries to prove the theoretical facts presented in [4], [5] and [6] .

1.3 Foton Software

The foton is a software designed at LIGO which enables one to design a filter according to the specifications. It is a graphical user interface (GUI) which provides for various types of design methods such as the zero-pole-gain (ZPK) method or design using bode plots. Any user can easily design a filter using this GUI. After the design of a filter, its coefficients are written into a text file in a specific format with a specific name. These text files can be read by the foton software as well. Hence, whenever the need arises, a filter can be looked at (either the ZPK form or its bode plots) using the foton software by mentioning the file name in which the filter bank name exists. In this way, the thousands of digital filters have all been documented in these text files using the foton software.

The foton filter design procedure and GUI is a lot like MATLAB's filter design tool in the signal processing toolbox. To access various filters for analysis in MATLAB or in Python or C, the text files can be parsed. Some codes which are used often are already available.
The foton software provides a complete analysis of a digital filter. For any given filter, its step response, impulse response and ramp response can be visualized. Also, various other parameters such as pole and zero location, transfer function etc. are available. For designing a filter, some common filter designs such as the butterworth, chebyshev and elliptical filter designs are available for use and modifications.
This project reads data from the foton text files for the coefficients as well as reading all the digital filters simultaneously for the new software tool development.

1.4 Frequency Domain analysis of Quantization Noise

The discussion of noise analysis in frequency domain was initiated in the [7] and [1]. This section is continued after what was already discussed. All signals in the digital signal are sampled at a particular frequency fs which is mentioned in the foton file description of each filter. Noise is random but its characteristics can be pinned down on proper analysis in frequency domain. It could be a signal having a constant magnitude throughout the frequency scale (white noise) or it could be gussian or might have some other properties. All these or most of these properties are reflected on proper frequency domain analysis of the signals. Fourier analysis reveals the frequency components a signal is made up of [5]. The power spectral density (PSD) [9] gives a major piece of information as the SNR can be calculated by taking the ratio of PSDs of the output and the noise signals. The approach to calculate PSD involves taking the Fourier transform, then squaring the magnitude of result. There are various ways which affect the analysis and are described in detail in [8].
The quantization noise measured (estimated) by the subtraction procedure mentioned above is the raw data which is of no use if proper inferences are not taken out of it. It is only possible when the frequency spectrum of the quantization noise is calculated. This is done in a similar way as described in [8]. The power spectrum density for the noise, the input and the output is calculated using Welch's method [10] and plotted on the same log log plot. This project performs a rigorous frequency domain analysis to calculate the noise level and also the SNR. These results are plotted on a log-log scale for better visualization of the properties.

2 Quantization Noise Measurement: Implementation on 40m prototype Interferometer

This section describes how the quantization noise is measured and analyzed for the digital filters at the Caltech's 40m prototype Interferometer.

2.1 Previous Implementation in MATLAB

The existing tool based on MATLAB was developed by Denis as a part of his PhD dissertation, [11]. The noise estimation algorithm works as described in the quantization noise

measurement section above. For a given input, filter output is calculated by using two different mantissa size and hence getting two different outputs for the same input varying in the precision level. In this implementation, the two different precisions used were single precision (float in MATLAB) and double precision. The quantization noise would be obtained on subtracting the two outputs. This quantization noise would be the noise that would occur if all the filters in the digital control system were realized using single precision. Since, all digital control systems are implemented in double precision in the LIGO setup, single precision noise is of no use. To obtain double precision noise, in this code, an extrapolation is done to estimate the approximate quantization noise occurring in digital filters implemented in double precision. The approximation method has been described by Denis in [13]. An empirically obtained factor equal to 10^{-7} is multiplied to the quantization noise values obtained for single precision.

2.2 Data Acquisition

To acquire input signal data for the filters the code downloads data from the 40m servers for 32 seconds and moves on towards noise estimation after some checks which determine whether the data downloaded is correct or not. The IIR filter second order sections (sos) is obtained from the foton filter file for the asked filter bank. The foton file contains the information about the coefficients of the filter and also the sampling frequency. Along with some other required parameters describing the gain, the offset and the saturation limit, the code then calculates the output and hence the noise using the algorithm described above.

2.3 Limitations of the Implementation

Here is a list of limitations of the noise estimation code as given in [12] and described in [11]:

1. The double precision noise is not actually calculated, but only approximated empirically:
   This is the major limitation of the code given by Denis, though the empirical approximation is intuitive but the noise analysis cannot be called cent percent accurate. To obtain noise in double precision, ideally, as described previously, one has to calculate the output in a precision which is better than double precision and then subtract the two outputs. This implementation doesn't do that, due to limitations of the MATLAB platform.

2. Signal To Noise Ratio: The code [12] doesn't facilitate SNR calculation for the noise estimated.

3. Tedious job if one wants to check many filters: Since, the channel names are complex and there are thousands of digital filters in the control system, it is a very tedious job to manually keep calling this function to check a digital filter with all its parameters.

4. Not all channels are turned on at all times and hence may return all zeros when requested for data. This implementation downloads the data for 32 seconds for every channel which is a long time and even so when the data being returned is of no use.

2.4 Ways to improve the measurement accuracy

1. Writing the complete code in Python or C :
   Though it seems like a good option since C or python would provide for long double precision [3] and hence the noise can be calculated with accuracy for double precision implementation of digital filters. But it is not a feasible option to go forward with because of two main reasons:

   (a) MATLAB's interface providing easy handling of matrices, arrays and plotting power spectrum densities among various other advantages that come with MATLAB.

   (b) The existing background code for various things such as foton file parsing etc. (in MATLAB) could be used 'as-is' and hence would be a great time saver.

2. Using MATLAB's multiple precision toolbox :
   A new toolbox was released for versions later than MATLAB 7 called the multiple precision toolbox [14]. This toolbox defines a multiple precision (mp) class under which objects with arbitrary precision could be defined. The MP toolbox also provides support for a host of MATLAB's functions so that all kinds of calculations could be carried out on MP objects just the way it is done for normal variables (float, double etc.). A big advantage that MP toolbox provides is that calculations in arbitrary precision could be done. But, this comes at a cost which is that the MP toolbox is not a part of Mathworks installation and needs to be setup separately for every machine. The MP toolbox depends on the MPFR and GMP C/C++ libraries. So, inorder for a machine to utilize the MATLAB's MP toolbox, the user needs to make sure all dependencies are present. These efforts required from the user's side make the overall process of analyzing quantization noise for digital filters cumbersome. Also, it has to be kept in mind that the software tool being developed is easy to use and is not prone to software changes/updates in future.

3. MEX-function to interface a part of code in C with MATLAB: This is the method already in use and a long double array in C cannot be taken to MATLAB using this option and hence the method is not feasible to calculate output in higher precision.

4. Binary File Interface between MATLAB and C: On analyzing and even trying all the methods mentioned above, finally it came down to develop two independent codes in C language and in MATLAB. Communication between the two is achieved by files which are used to write and read the data for the outputs, noise and the input signals. Text files could be used for this purpose but were not because for the size of data being communicated, the size of the text files was exceeding 25 MegaBytes of disk space. This contributed majorly in slowing down the whole process. Hence, binary files were used.
   This method was chosen above all the others mentioned above because it successfully achieves the task of calculating output using higher precision, while simultaneously providing for a standalone application.
   MATLAB writes the input signal array, filter coefficients containing sos matrix, the filter order, the gain along with other variables required for noise calculation. The C executable is called from MATLAB which then reads the file written by MATLAB.

On receiving all the input variables, the C code calculates the low noise form filter coefficients and then the outputs for both kinds of filters (DF2 and LNF). Even the noise calculation is done in the C code by subtracting the outputs of long double and double precision. The calculated noise and the outputs are written into another binary file by the C code after which MATLAB continues its execution where it reads the file written by the C code and goes on to plot the input, output and the noise.

2.5 Improvements completed in the new implementation

1. Accurate Noise Calculation: The procedure used has been described in previous sections in detail to achieve a better noise approximation. This is the most significant improvement in the code as now the noise is estimated systematically rather than empirically as done previously.

2. SNR Warning: Various level of warning messages have been set. The code calculates SNR from the noise power and the output power. This SNR is then checked, if its at a dangerous level or not. If the SNR is above the limit which is currently set at 100, a message saying "Filter is alright" is displayed on the MATLAB console. For SNR values less than 100, different warning messages are displayed depending on the seriousness.

3. Speed : The existing code downloaded data for the input signal for 32 seconds time interval for all filters. In the Digital Controller, many filters are fed with zero input and are switched on. These filters need not be checked hence the improved version of the code first checks if the data is non zero or not by downloading a small chunk of it (for eg. for 1 second) and if it is non zero then moves ahead with downloading 32 seconds of data, otherwise it displays a message to the console saying that the filter module is switched off. This helps in saving a lot of time in the long run of checking many filters.

4. SNR Plot: A plot is displayed showing the SNR distribution along the frequency scale for the low noise form of the digital filer.

3 Software Tool

One complete software tool was developed by incorporating all the changes and improvements described above.

3.1 Automatic Quantization Noise Analyzer Tool

The software tool is a MATLAB function which can be called without giving any arguments. Once called, it starts checking all the filters one by one on its own until the end of the list. So, basically analyzing all the digital filters is just one click away. The list that this function looks into is a list of filter bank module names mentioned in the Foton's description of the

filters. The automatic filter analyzer function constructs the channel name from the foton file name and the filter bank name given in the file. After constructing the channel name string, it checks whether this channel actually exists or not. This check is done against a list of channels downloaded from the server. The function developed for a single filter is then called upon with the arguments constructed automatically. Finally, the power spectrum density is plotted of the output against the noise. The software tool then moves on to check other filters.
A problem quite common during this analysis is the ambiguity in the channel name. There are two ways in which channel names have been defined, and hence both need to be checked otherwise the Invalid channel name error occurs to halt the execution. The code takes in account this duly and checks for both kinds of channel names. Not only this, the automatic filter analyzer tool continues operation even when any error occurs (recording the error in a log file). It has been implemented in this way because even on matching the channel name against a list of channel names downloaded from the server, there may still be many channels which might not exist because of continuous updates in the foton file names and their descriptions.

3.2 Testing at the 40m Interferometer Prototype

The tool developed was run to check all the digital filters implemented in the 40m Interferometer at Caltech. Results and plots are given later in the conclusions section.

4 Moving Towards Advanced LIGO

The Advanced LIGO was a big upgrade over the initial LIGO (iLIGO). Majority of hardware was replaced as it grew old and obsolete. Though the DAQ software remained the same, the CDS hardware was completely changed and upgraded. However, the basic architectural philosophy remained the same. A complete description of changes can be found in [15].

4.1 The Advanced LIGO Digital Control System: an Upgrade

The direct form 2 (df2) filter structure used earlier in the iLIGO was replaced by the much better,low noise form described in [6]. The filter designs were heavily altered with and more and more complex filter structures were implemented to incorporate various other changes in the hardware and to improve performance. With the complexity of the filter designs comes the risk of various noises coming into picture. Hence, a complete quantization noise analysis, among all others, is the need of the hour. This project aims to analyze each and every digital filter at Advanced LIGO for quantization noise to ensure that no signal is being limited by the quantization noise effects.
While most digital filters are simple gain filters, some of them are integrators and other common filters but some of them are quite complex. This project aims to check all digital filters.

4.2 Software Tool for Remote Analysis of Digital Filters at the Advanced LIGO sites

The two Advanced LIGO sites are at Hanford, Washington and Livingston, Louisiana. Working with digital filters implemented on the computer softwares at the sites from Caltech requires remote access which is provided by the sites, with some limitations. The nds servers provide access to the data for the filter channels. A simple guide to install and use nds servers is [16]. This guide can be used to run the software tool developed in this project independently on every machine with LIGO access. A readme is also available at [17] which provides step by step procedure to run the software tool at the sites remotely. The tool has been developed so that the steps to get it set up and running are simple and a few.
Two separate MATLAB scripts have been developed, one for Livingston and other for Hanford site. It can be called without any arguments and takes all the names pf channel, filter banks etc from the foton software text files. The code ignores the errors the errors so as to continue the analysis for all the filters. In the end, a complete log file is generated which has the record of all the errors and the filter banks which gave those errors so that the problem could be looked into. This software tool works very efficiently and takes a small time to analyze all filters because it downloads the data and the complete list of channels beforehand and matches the channel name constructed from the file to check if it actually exists at the given time. If not, it ignores and moves on. The data download is also fast because of the nds2 server functionality which downloads the data for the channel to a computer once and it remains in the cache for 24 hours making the analysis fast. Even after these efforts, the code takes a few hours (more than 4-5) to completely check all filters at a site since there are thousands of filters present (more than 5000).
The base code is the same with changes incorporated for it to run at the sites. Hence, PSD's for all digital filters' input signal, output signal for both DF2 and LNF and noise for both DF2 and LNF are plotted and saved as vector graphic files in the given directory.

5 Results and Conclusion

All results are mentioned in the following list:

1. Most of the filters are "safe". "Safe" here means that the SNR was constantly above the limit set for the same at 100 in the code. The quantization noise levels for most of the filters are orders of magnitude below the output PSD level. This type of response was observed for the complete frequency range except at very high frequency (twice of Nyquist frequency) where the output of the filter is usually rolled off to lower magnitudes. Some assorted results for some of the filters from both aLIGO sites are shown in the figure 1 and the ones following it. Figure 5 shows the SNR distribution of most of the filters analyzed. This type of SNR was common for most of the filters realized in the Low Noise Form.

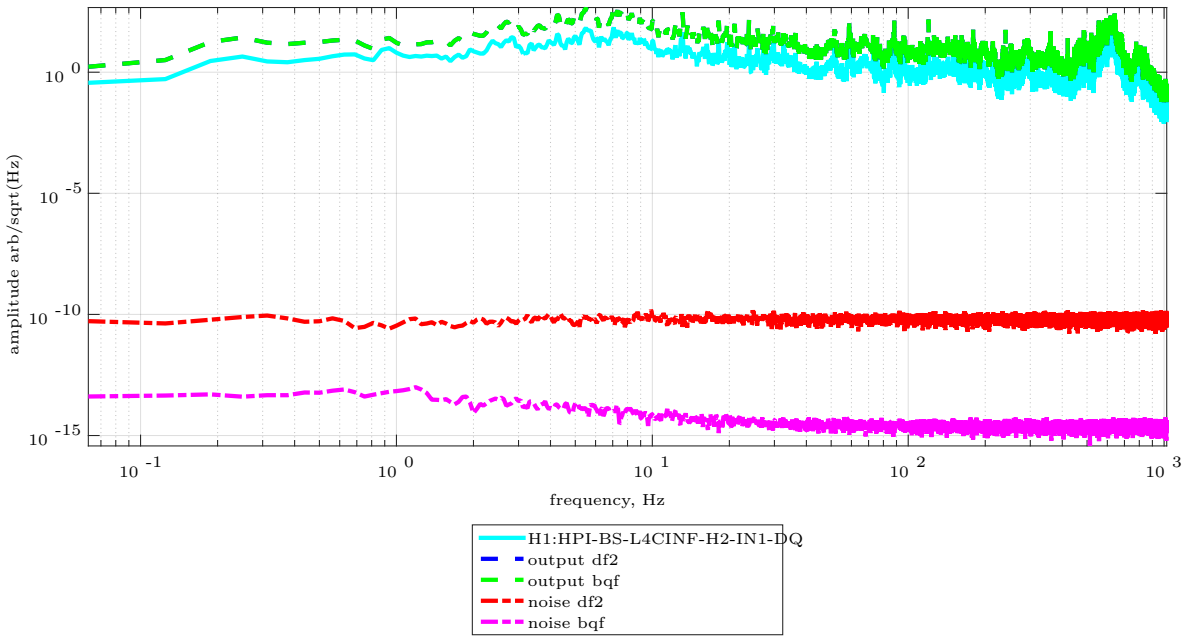

Figure 1: Analysis for Hanford HPI BS filter

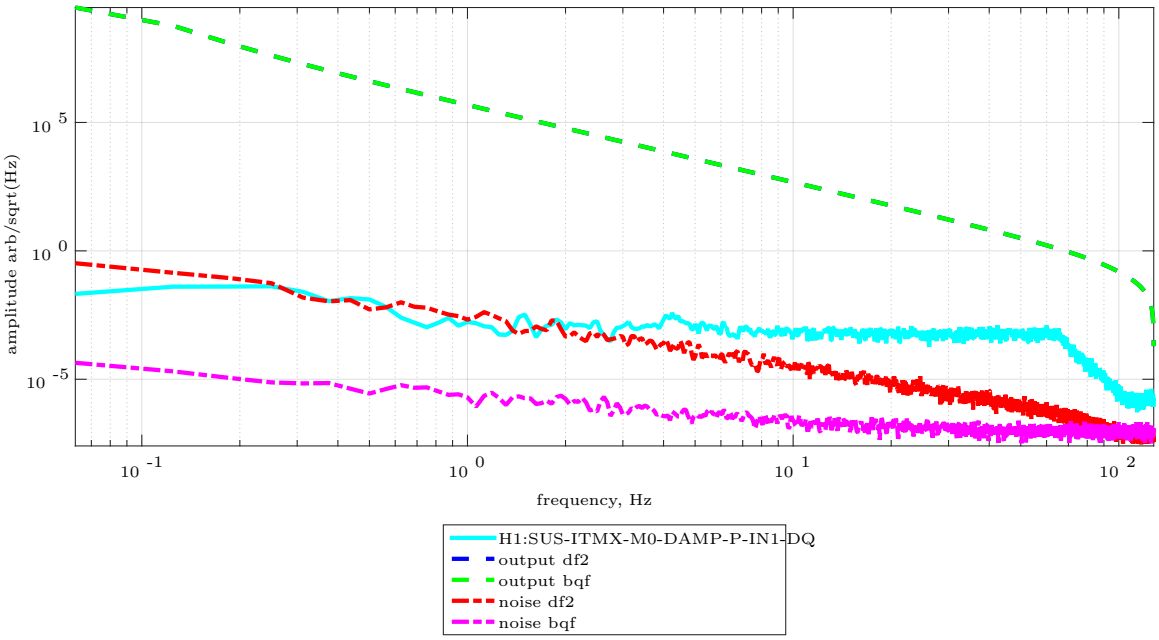

Figure 2: Analysis for Hanford SUS ITMX filter

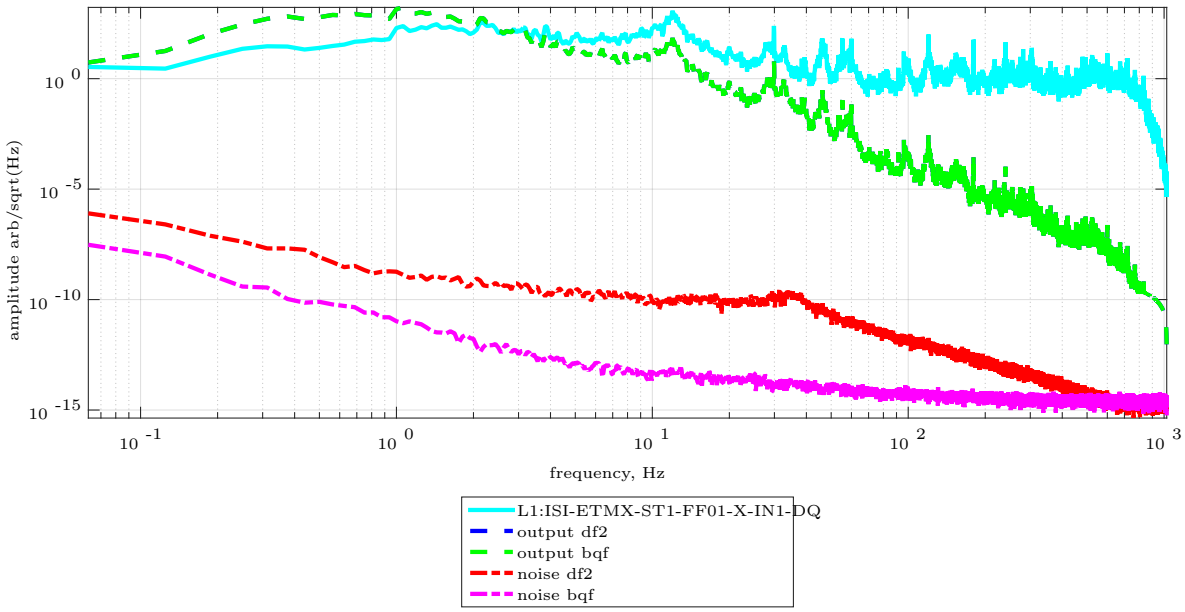

Figure 3: Analysis for Livingston ISI ETMX filter

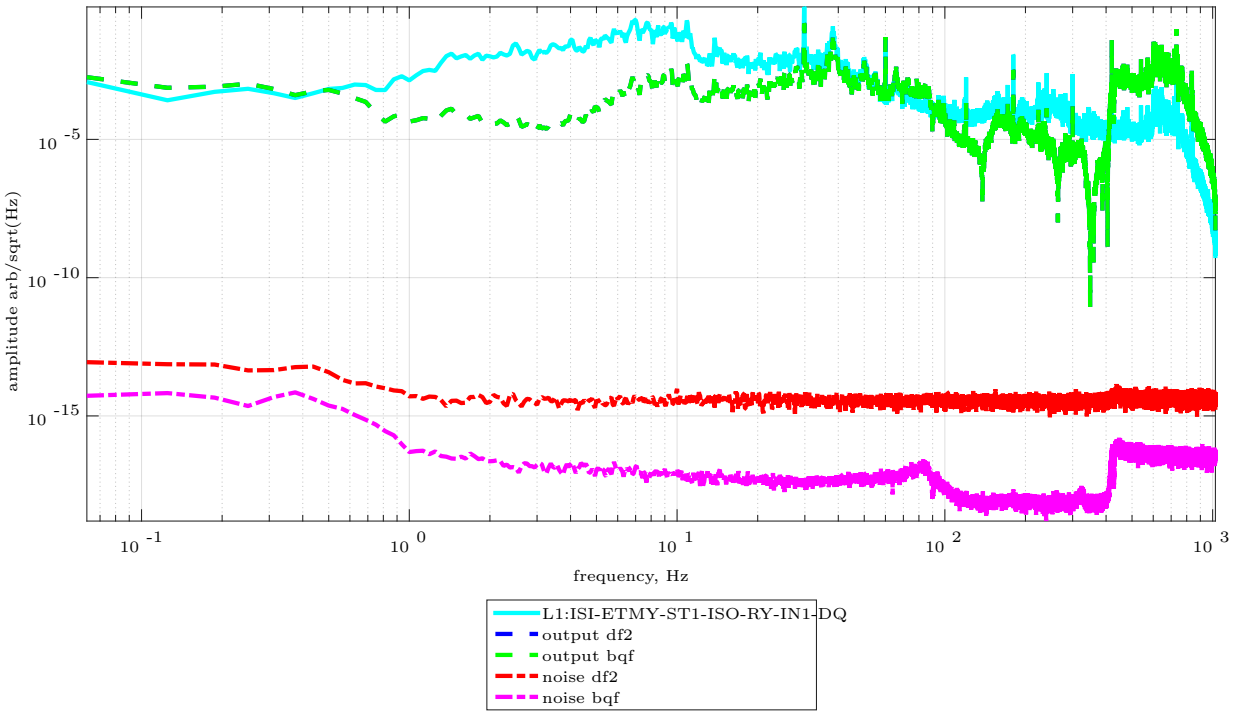

Figure 4: Analysis for Livingston ISI ETMY filter

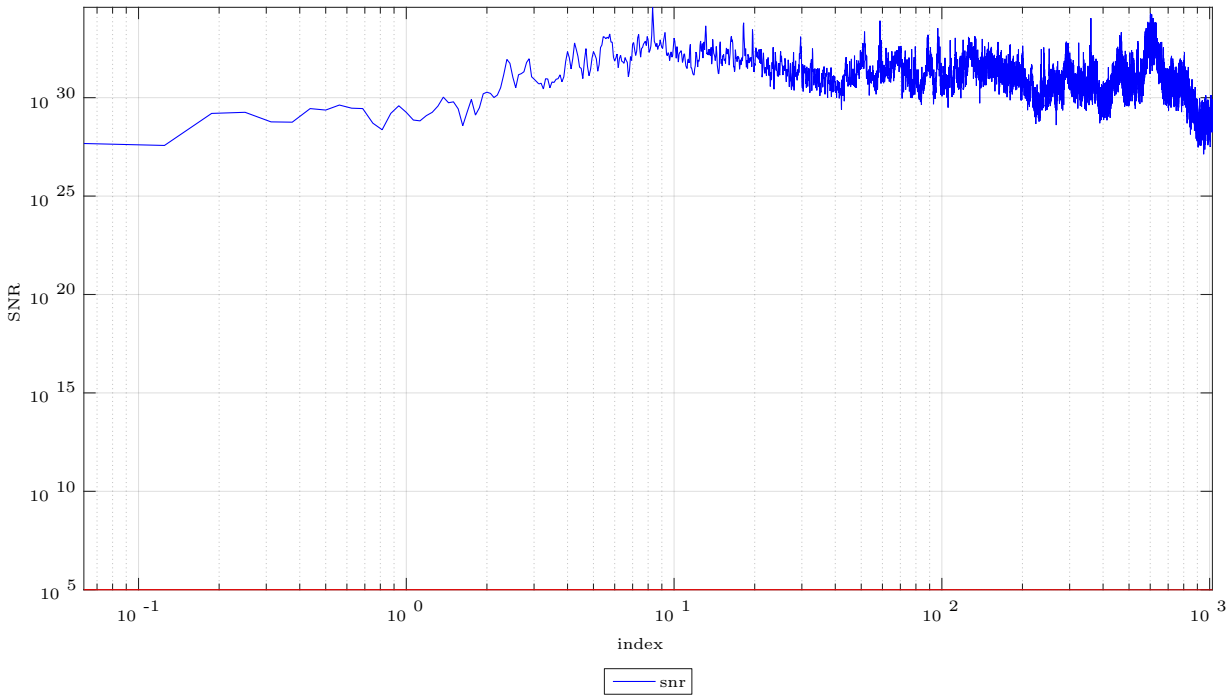

Figure 5: SNR distribution for HPI ITMY filter

2. The DF2 filter shows alarming results for some of the filters. For a few filters, the DF2 quantization noise level is even above the output for some frequency band. Though this is an alarming observation, but this does not affect the aLIGO digital control system at all because all digital filters have been implemented in the low noise formand the DF2 form has been changed long back. This result further strengthens and proves the results showed by [6] and others. For these filters, the low noise form quantization noise level is considerably lower and the filter is safe. Some results are shown in the figure 6 and the ones following it.

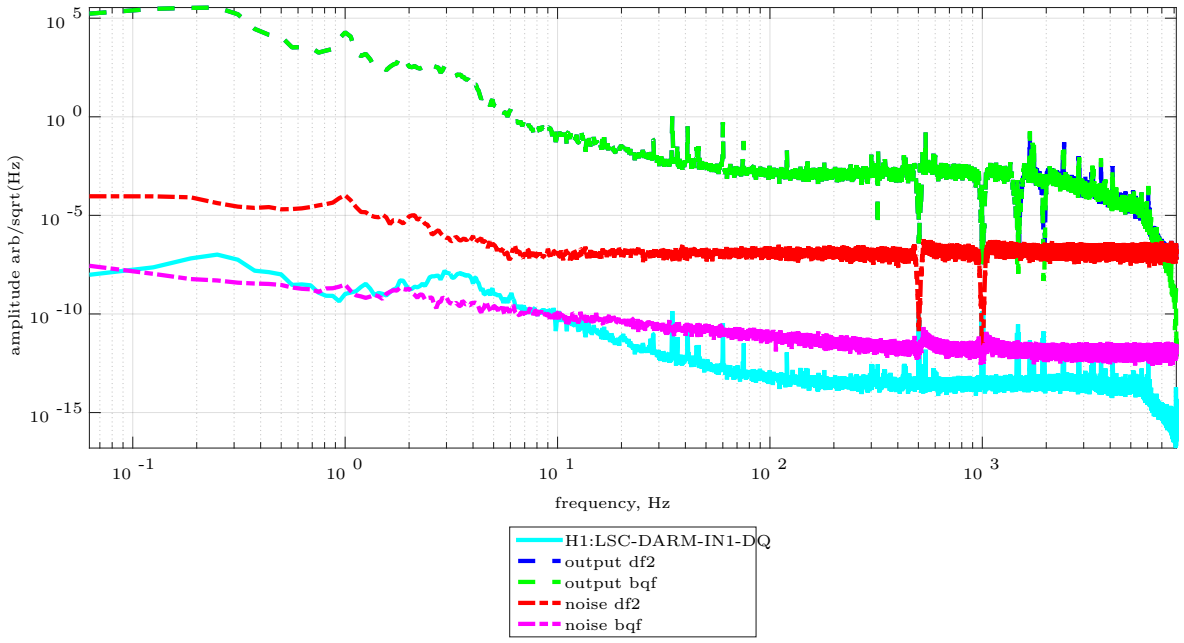

Figure 6: Analysis for Hanford LSC DARM filter

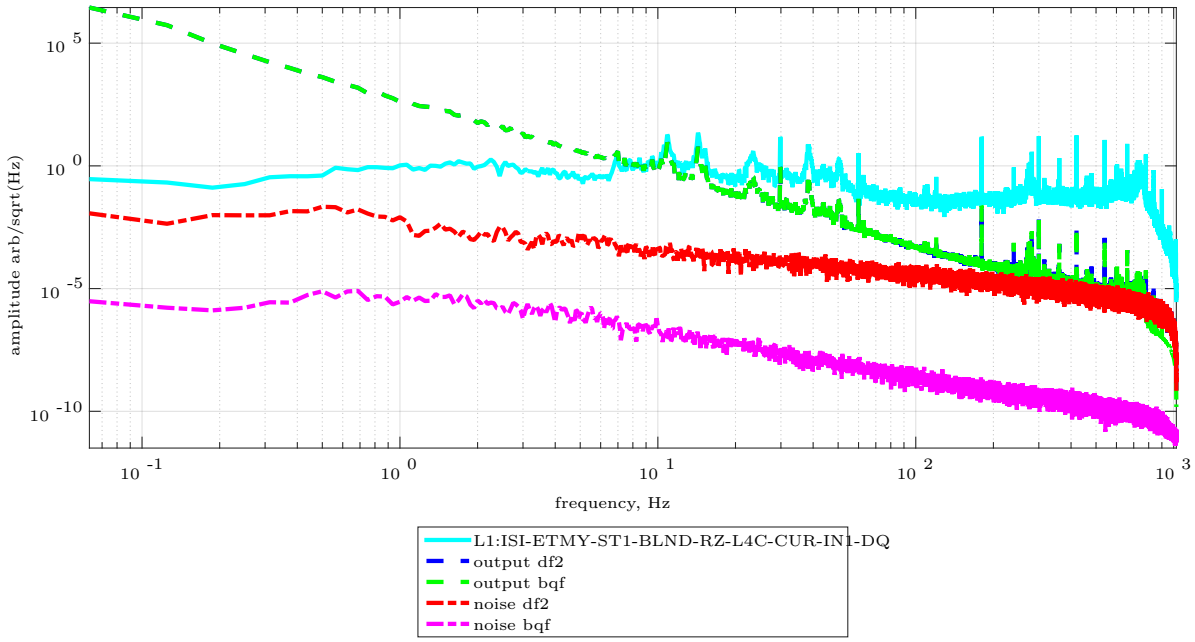

Figure 7: Analysis for Livingston ISI ETMY filter

3. For some filters, the $DF2$ performs equally well as the LNF. This result is shown in the figure 8 and the ones following it. The reason behind this observation is clear from the figures which show the input and output signal as well. The output for these kinds of filters are mostly representing filters which only perform a gain operation (i.e. a simple multiplication). If it is not a perfect gain filter, even then the PSD plot for the output shows that for such kinds of filters the variation is very less, if any. Since a pole causes a $-20dB/$decade slope and a zero results in $+20dB/$decade, these kinds of filters represent a pole and a zero placed close to each other.

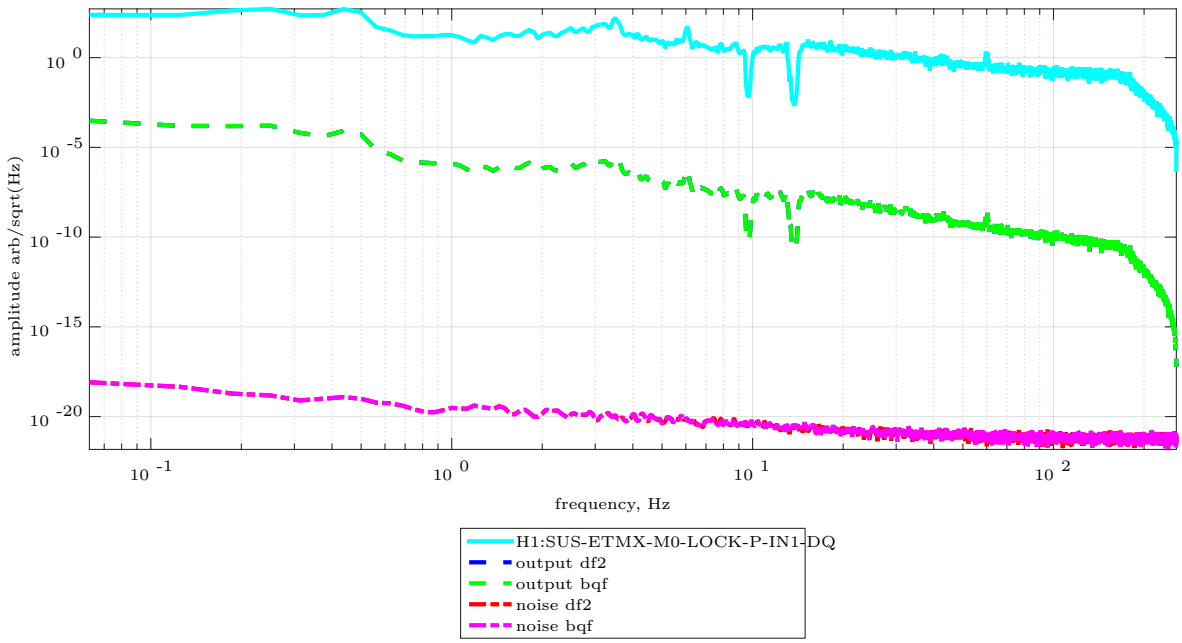

Figure 8: Analysis for Hanford SUS ETMX filter

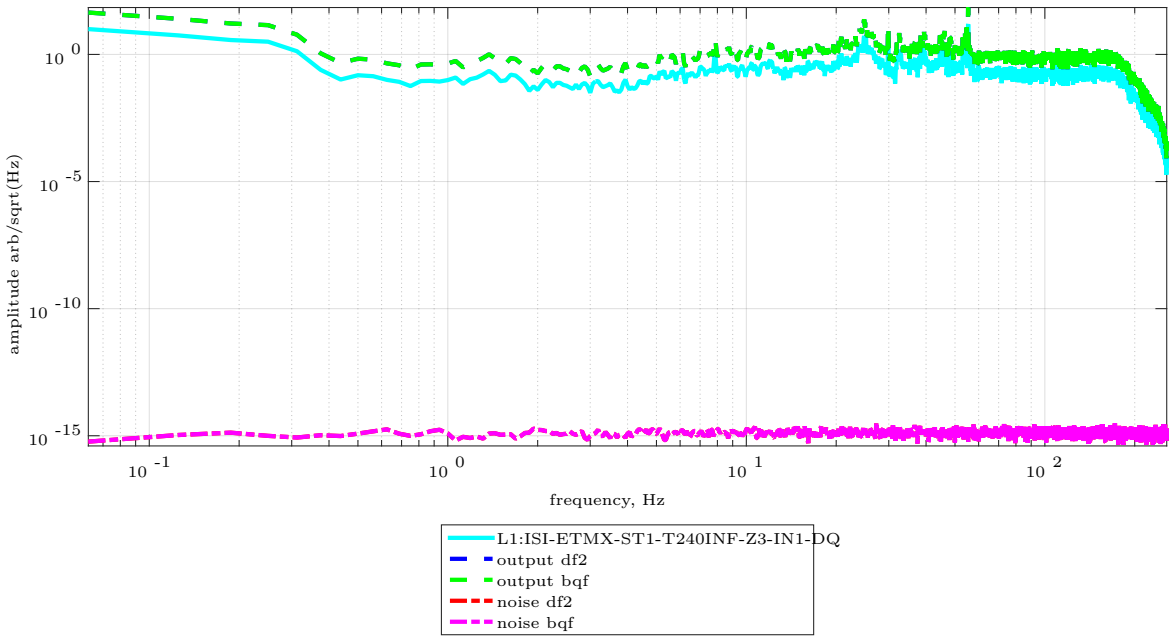

Figure 9: Analysis for Livingston ISI ETMX filter

4. There are filters that were observed that do not give any output even when the input is not zero and channel is valid. The gain for these filters is also not zero and hence on debugging the problem it was found that the foton file description of these filters describes the coefficients of these filters as all zero. The ZPK is given as all zero. This is suspected to be a bug, though it might be something intentional as well. There were many filters mainly of the type $BLND_*$ which showed this characteristic. The response for these filters is shown in the figure 10 and the figures following it. As is clear from the figure, only the input was plotted because the resultant output is zero and hence the noise.

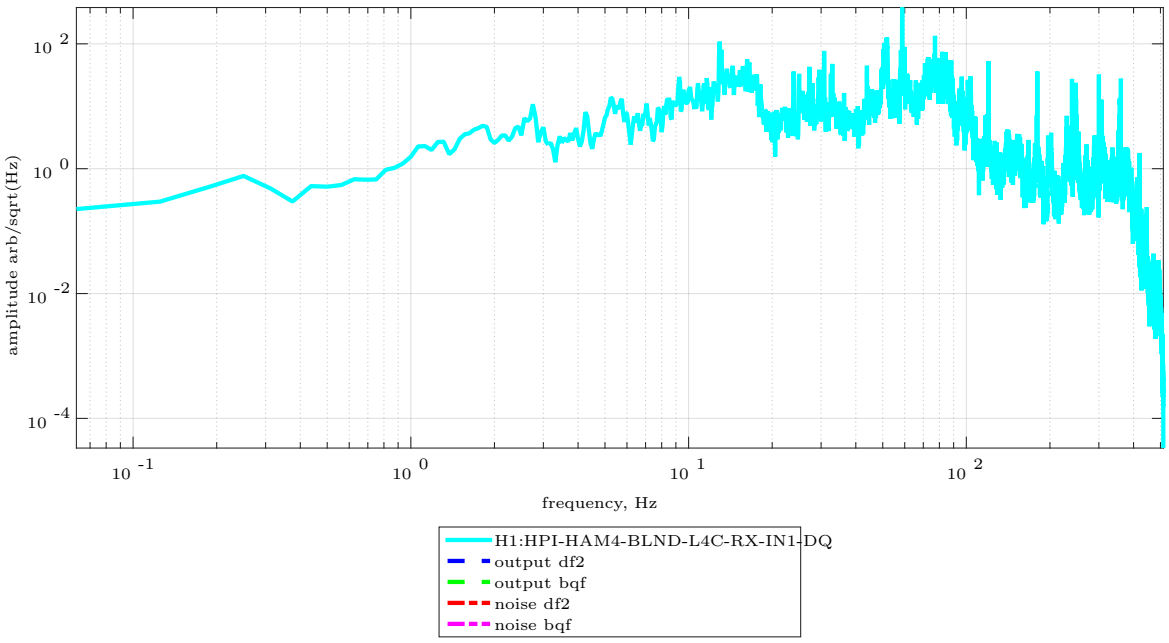

Figure 10: Analysis for Hanford HPI BLND filter

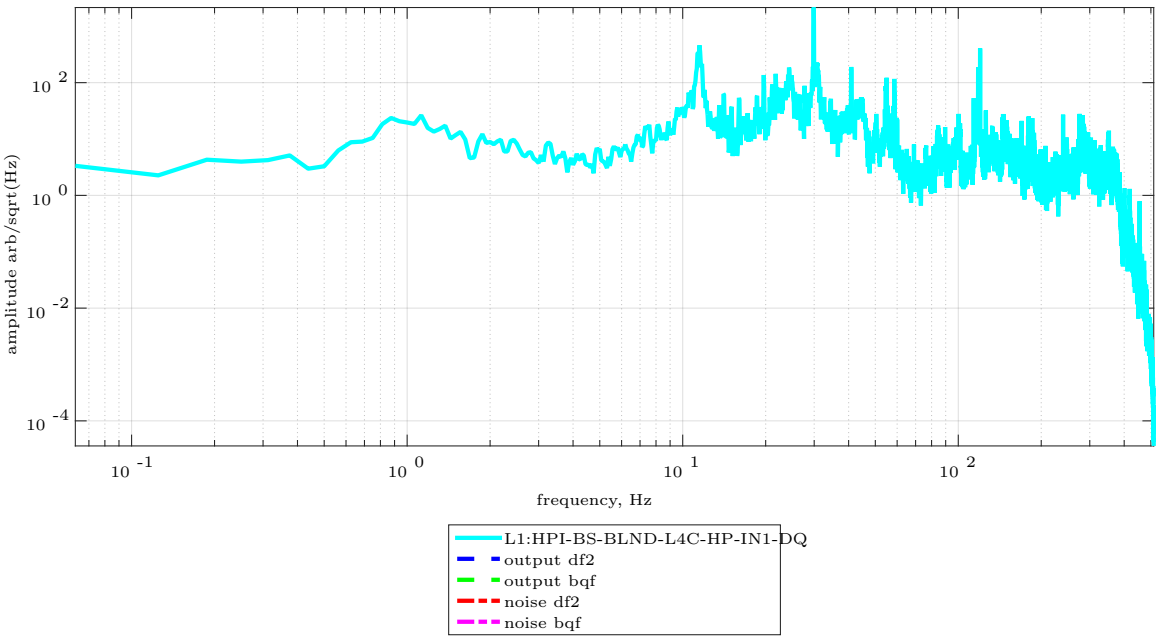

Figure 11: Analysis for Livingston HPI BLND filter

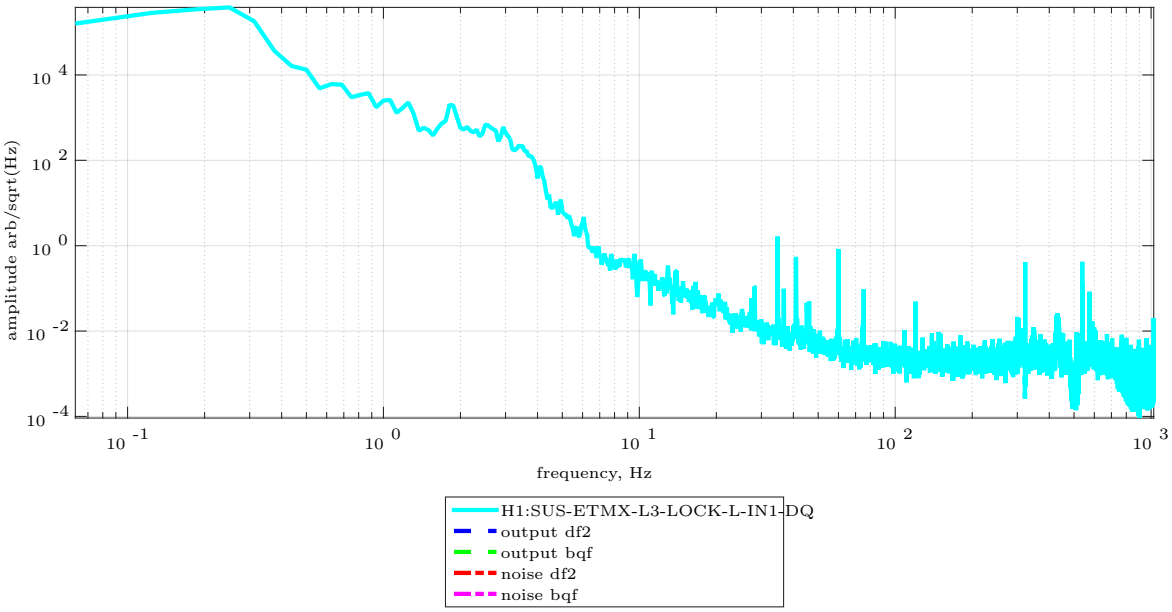

Figure 12: Analysis for Hanford SUS ETMX filter

5. A few filters do not perform any operation and hence their output is equal to the input. For these filters, the quantization noise is zero as shown in figure 13.

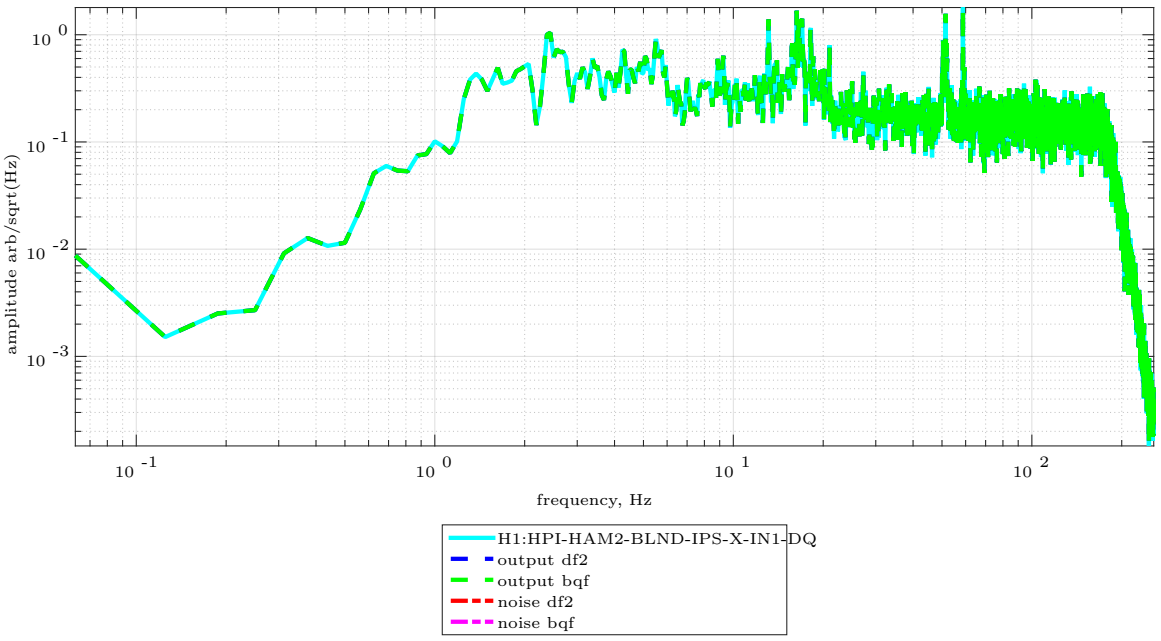

Figure 13: Filter HPI HAM2 BLND showing zero noise

5.1 Conclusion

The conclusion is favorable and on a positive note that there are no filters which have problems with quantization noise in the LNF structure of the filter.
The complete analysis is available at [18]. It has all the plots for the thousands of filters analyzed using the software tool.

6 Challenges Faced

Though the results and the task show that the process to achieve it would be simple, but it was no where near to being that as there were numerous challenges that came along the way . Some of them are listed here:

1. Channel Name Ambiguity: As previously mentioned, the channel names in the software tool are composed automatically without any input from the user. But in the process of creating this channel name from the filter bank names given in the foton file, there is an ambiguity. The problem that was faced with and dealt with successfully after some debugging was that the channel name could either contain a string from the file name or it may not for some others. The software tool finally developed checks for

which one is correct by matching the channel name from the downloaded list and then proceeds with further execution.

2. Remote Connection Problems: There are a host of problems that needed some debugging regarding the remote connection to the sites. These have been well documented and have been accounted for so that running the software tool in future won't be a hassle.

3. Handling the Long Double Precision: As MATLAB only deals in double precision length variables, there were a few problems in dealing with the data input and output for the C code.

4. Down sampled Channel Data Rates:On downloading the data for a particular channel, the frequency is different to that mentioned in the foton file. This fact is due to down sampling of data sometimes. Hence, a check was employed in the downloading signal code

5. Random Errors and Segmentation Faults: These are a part and parcel of any coding problem. And yeah, just like every other problem, these were quite challenging at times and needed heavy debugging to fix.

7 Limitations

Here is a list of the limitations that the current study has not taken care of:

1. Only Data Acquisition Channels: As all work is being done remotely, hence only the channels for which the data is stored on a hard disk is available. So, only the channels ending in _DQ are being analyzed.

2. Only filters having input recorded: Currently the code analyzes only the input channels. But as it turns out, there are lots of channels which have their outputs recorded but not their input. Hence, these channels are going unchecked. But, this problem is next on the list of things to do and hopefully this won't be a limitation in future.

3. User friendliness Tool Structure: Some steps are required to set up the software tool. Though, all these steps have been mentioned in the documentation, still there is always that odd chance where the user falls into a problem while setting up the tool to run. This limitation will be continuously looked into and improved upon.

4. Size of the software tool: Since, file handling is being used to implement noise estimation with accuracy using output calculation using higher precision, the size of the complete code has gone on to be more than 30 mega bytes of memory.

8 Goals for the coming weeks

The initial goal for the coming week would be to tackle the limitations which are possible to be tackled and eliminated. Following that, as planned in the [7] the last weeks of the project

would be given to the DAC quantization noise analysis and the possibility of implementation of noise shaping to the same to improve the noise performance. It is expected that the DAC might be limiting some signals due to its quantization noise, hence there exists the possibility to look into ways to mitigate that noise. Currently, the DACs do not take advantage of the noise shaping technique in the code which would push the quantization noise to higher (undesired) bands of frequency. The possibility of declaring the complete digital controller at aLIGO quantization noise free (or atleast unharmed) still remains and all efforts in that direction would be taken in the remaining time.

References

[1] Ayush Pandey, *Progress Report I : Quantization Noise in Digital Control Systems*

[2] Wikipedia: Double Precision https://en.wikipedia.org/wiki/Double-precision_floating-point_format

[3] Wikipedia: Long Double Precision https://en.wikipedia.org/wiki/Long_double

[4] Widrow and Kollar, *Quantization Noise Book*

[5] Oppenheim and Schafer, *Discrete-Time Signal Processing*

[6] Matthew Evans, *Digital Filter Noise*

[7] Ayush Pandey, *Quantization Noise in Digital Control Systems* https://dcc.ligo.org/LIGO-T1500243

[8] Jonah Kanner https://dcc.ligo.org/LIGO-G1500862

[9] Michael Cerna and Audrey F. Harvey (2000) *The Fundamentals of FFT-Based Signal Analysis and Measurement*

[10] Welch (1967), *The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms* , IEEE Transactions on Audio Electroacoustics, AU-15, 7073.

[11] Denis Martynov, *Lock Acquisition and Sensitivity Analysis of Advanced LIGO Interferometers* https://dcc.ligo.org/LIGO-P1500087

[12] Denis Martynov, *Checking Digital System* http://github.com/denismartynov/quantization

[13] Denis Martynov, http://nodus.ligo.caltech.edu:8080/40m/?mode=full&attach=1&reverse=0&npp=510&Author=den&subtext=digital

[14] Ben Barrowes *Multiple Precision Toolbox* http://www.mathworks.com/matlabcentral/fileexchange/6446-multiple-precision-toolbox-for-matlab

[15] Rolf Bork, https://dcc.ligo.org/DocDB/0001/T070059/001/T070059

[16] John Zweizig, https://www.lsc-group.phys.uwm.edu/daswg/projects/nds-client.html

[17] Software Tool Code, http://github.com/rxa254/DACdenoising

[18] Ayush Pandey, *A complete collection of all plots for all digital filters at Hanford and Livingston* https://drive.google.com/folderview?id=0BzjRW8WwGjzJfkE3cVFzczJVU0JpSkZUTm1DR0dpWF9BWFlNVTh3VGg3UG93dHRLTURPZWs&usp=sharing