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

Technical Note **LIGO-T1900398-v1** 2019/08/10

Analisys of the correlations between the non stationary noise and the auxiliary channels in LIGO

Luca D'Onofrio[1], Gabriele Vajente[2]

[1]Universita' di Napoli Federico II, Complesso Universitario di Monte S.Angelo, I-80126 Napoli, Italy
[2]LIGO, California Institute of Technology, Pasadena, CA 91125, USA

Distribution of this document:

LIGO Scientific Collaboration

Draft

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 NW17-161
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 Non stationary noise in LIGO

Data from the LIGO detectors typically contain non Gaussian and non stationary noise which arise due to instrumental and environmental conditions. This non stationary noise implies a time variations of the LIGO detectors sensitivity.

Operating the LIGO detectors at the sensitivity needed for gravitational-wave detection requires a large number of control loops and sensors to precisely control and monitor the instrumentation and environment. The channels that continually record information from those sensors allow to analyze the problematic variations in the sensitivity of the detector. However, due to the complexity of the system, the number of auxiliary sensor channels is around hundreds of thousands. The LIGO auxiliary channels include microphones, seismometers, magnetometers, photo-diodes, voltage monitors, feedback loop signals etc.

Searching through such a large amount of data to determine which sensors are most highly correlated with variations of the noise in the considered frequency range is an important and difficult challenge. Characterizing these correlations from the data and improving the background of LIGO's searches is crucial for reducing non-stationarity in the detectors and increasing the statistical significance of gravitational-wave candidate events.

2 Computation of Band-Limited Root Mean Square

To perform quantitative analysis on the noise floor variations over time, it is useful to compute the Band-Limited Root Mean Square (BLRMS). If $x(t)$ is a time series[1], the mean power density of $x(t)$, also called root mean square value (RMS), is defined as [1]:

$$RMS = \lim_{T\to\infty} \sqrt{\frac{1}{2T}\int_{-T}^{+T} x^2(t)dt} \tag{1}$$

that is linked to the auto-correlation function at zero delay. The power spectral density (PSD) is defined as the Fourier transform of the auto-correlation function R_{xx}:

$$PSD(\omega) = \int_{-\infty}^{+\infty} e^{-i\omega\tau} R_{xx}(\tau)d\tau \tag{2}$$

where

$$R_{xx}(\tau) = \lim_{T\to\infty} \frac{1}{2T}\int_{-T}^{+T} x(t+\tau)x(t)dt\,. \tag{3}$$

The physical units of the power spectrum are $[x]^2/Hz$ where $[x]$ is the physical unit of the considered time series. The BLRMS values of a signal in a frequency band $[\omega_1, \omega_2]$ is simply the integral of the PSD in that band:

$$BLRMS = \sqrt{\int_{\omega_1}^{\omega_2} PSD(\omega)d\omega}\,. \tag{4}$$

[1] We are considering ergodic process. A stochastic process is said to be ergodic if its statistical properties can be deduced from a single, sufficiently long sample of the process.

Table 1: Frequency bands for the computation of the BLRMS for the Ligo Hanford detector.

Starting frequency [Hz]	End frequency [Hz]	Starting frequency [Hz]	End frequency [Hz]
10	13	96	116
23	26	111	114
26	29	116	174
28	31	174	186
31	44	186	286
44	56	316	400
46	47	440	495
47	49	521	583
56	62	630	823
62	86	870	970
86	96	1120	1400
88	90		

Using the PSD of LIGO's detectors, if the computation of the BLRMS is performed periodically it is possible to track the time evolution of the noise.

In this way, considering n frequency bands and a time T as the sampling rate to compute the BLRMS, the results are n new signals which give the noise power in each band as a function of time [2]. It is possible to obtain these signals at a larger sampling rate using overlapping time intervals for the computations. This is useful to have a better frequency resolution.

Once the n frequency bands have been selected, the time evolution of the BLRMS can be analyzed. In this work the chosen sampling rate is $T = 2.5$s while the chosen bands are in Table 2. For the computation of the PSD, we use the fast Fourier transform (FFT) with a time window of 5 seconds, scale it in units of PSD, integrate to get the BLRMS and then we move forward by 2.5 seconds and repeat the computation [3].

In time evolution of BLRMS, *glitches* could be observed. A glitch is a high-amplitude, short-duration excursion of unknown origin in the calibrated strain signal, which is what we use to compute the BLRMS. To identify glitches, for each point we computed the median value and a dispersion (a sort of standard deviation of the data, but using median instead of mean) in a chosen time range. In this work we chose a time range of 40 seconds. We used the median value and not the mean value because the first is less dependent to the value of the glitch. It is possible to put a threshold (median plus three times the dispersion) on the BLRMS value to identify glitches and reject them using an interpolated value (see Figure 2).

Using the same procedure, it is possible to identify *lines*, that are strong sinusoidal contribution at a given frequency in the PSD. The BLRMS of a band containing a line might be completely dominated by the power in the line itself [1]. Therefore every fluctuation of the noise floor is hidden by the line amplitude.

Since the aim is to analyze the non stationary noise floor on long time scales (approximately days), glitches and lines are not relevant in this context. Using an algorithm similar to what used for glitches, lines can be removed and BLRMS can be computed over frequencies excluding the lines.

The first step toward finding correlation between the noise variation and auxiliary channels is to compute the time evolution of the BLRMS for a certain frequency band without glitches and lines.

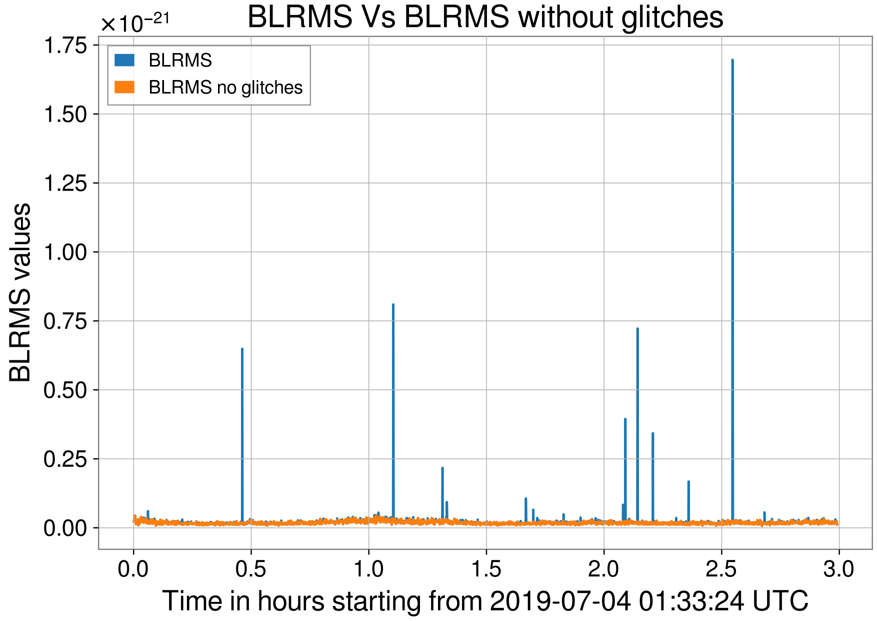

Figure 1: Plot of the BLRMS time evolution in the frequency band $47-49$ Hz. The blue line shows the BLRMS time evolution loading from the available data while the orange line shows the same evolution excluding glitches using the described procedure.

This has already been computed with the code in [3] and available in [2]. The next step is the use of statistical methods to verify potential correlations, in particular applying a multi-dimensional linear regression.

3 Multi-dimensional regression

The values at n different times y_i of the BLRMS in a given band can be viewed as the noisy version of a linear combination of the k auxiliary channels measured at the same times $x_{1\cdots n,1\cdots k}$ [4] where the first index runs over time samples and the second index over all auxiliary channels,

$$y_i = \beta_1 x_{i,1} + \cdots + \beta_k x_{i,k} + \varepsilon_i\,, \tag{5}$$

that can be rewritten as:

$$y = X\beta + \varepsilon \tag{6}$$

where y and ε are arrays of order $n \times 1$, X is a matrix of order $n \times k$ while β is an array of order $k \times 1$. The unknown coefficients β have to be estimated and ε is a source of additional noise, assumed to be normally distributed and considered as coming from the measurement of the y or also from the auxiliary channels.

The best estimation of the coefficients is obtained by minimization of the sum of squared errors S:

$$S = (y - \beta X)^T (y - \beta X) = \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{k} \beta_j x_{i,j} \right)^2 \tag{7}$$

and if all the auxiliary signals are linearly independent, there is a unique solution. The value of β that minimizes S is found by differentiating the function S with respect to β and setting the result to zero. This gives the condition,

$$\frac{\partial S}{\partial \beta} = 2\beta^T X^T X - 2y^T X = 0\,, \tag{8}$$

and so, for the coefficients β [5]:

$$\beta = (X^T X)^{-1} X^T y\,. \tag{9}$$

The equation (9) is sometimes called *”normal equation”*.

The advantage of this method to estimate the coefficients is that it is guaranteed to find the optimal solution analytically. This method is also useful to understand which amount of the total noise non-stationarity can actually be predicted from a given set of auxiliary channels.

However, if the datasets is large, it can be computationally too expensive to invert the matrix in this formula or simply, the matrix $X^T X$ may be singular (non-invertible). To face this problem, we compute the pseudo-inverse of the matrix X. The pseudo-inverse is equal to the matrix $(X^T X)^{-1} X^T$ if the columns of X are linearly independent. Otherwise it gives one of the possible solutions of the least square problem.

It is also easy to show that in some cases, standard linear regression can fail. We used simulated signals to analyze these cases. Chosen signals are random time series with 3600 points that corresponds to 60 hours for an auxiliary channel with sampling time of 1 minute. Using a law pass filter, we removed the high frequencies from these signals in order to reproduce the slow time evolution of the auxiliary channels. Choosing randomly ten coefficients, we defined a new signal combining linearly the first ten signals. A standard linear regression can be performed to compare the estimated coefficients to the real ones. If some background noise is added to the input signals, the regression coefficients are close to the real ones, but some other random coefficients also can get large. In addition, we simulated another random signal with the same characteristics of the input signals. In order to check for possible correlations with the input signals, we performed a linear regression and a good recontruction can be obtained if the number of signals is large enough. This is the most spectacular failure since we know there is no correlation at all. Linear regression is able to find a combination that does not exist due to the large number of signals.

Because of the great number of auxiliary channels in the LIGO detector, this problem of ”overfitting” must be considered.

4 Auxiliary channels

To achieve the required sensitivity for detecting gravitational waves, it is necessary to identify, control and eventually remove environmental disturbances. Therefore, the LIGO interferometers are equipped with thousands of sensors, the *auxiliary channels*, to monitor instrumental and environmental conditions.

The LIGO channels shown in this work have long and complicated names, using a series of abbreviations to specify the exact location, subsystem, and type of sensors being recorded, according to a convention described in [6]. Channel names follow this standard:

Table 2: Principal subsystems of LIGO detectors.

ASC	alignment sensing and control
LSC	length sensing and control
PEM	physical environment monitor
PSL	prestabilized laser
SEI	seismic isolation
SUS	suspension
IMC	input mode cleaner
ISI	internal seismic isolation
HPI	hidraulic external pre-isolation
SQZ	squeezed light

- Characters 1 and 2 show the IFO (interferometer) the channel belongs to (H1 for LIGO Hanford, L1 for LIGO Livingston, V1 for Virgo, G1 for GEO600).
- Character 3 = : (colon)
- Characters 4,5,6 show which subsystem the channel belongs to.
- Character 7 = - (hyphen)
- The rest of the name describes the channel

For example: H1:ASC−AS_D_DC_YAW_OUTPUT is the channel which samples the DC signal (total power) for the D photodiode yaw rotation at the antisymmetric port of the Hanford 4km interferometer. In the table 2, there is a list of subsystems of LIGO detectors used in this work.

5 Overfitting problems

In order to find a correlation between the BLRMS time evolution and the auxiliary channels' time series using linear regression as a first step, we considered the data segment from 1243393026 GPS time to 1243509653 GPS time (around 40 hours) in the frequency band 47-49 Hz. We chose this band because there is a bump that we would like to characterize [7].

In this first part, only the ASC subsystem (Alignment Sensing and Control) channels are selected. For this subsystem there are 2331 channels. Since the auxiliary channels time series are fast signals, we chose to consider the minute trends of these channels. These trends are time series with sampling time of 1 minute. For a particular channel, the corresponding trend is obtained computing the mean and the root mean square over 60 seconds of the channel time series. From each mean value M_i and mean root square value RMS_i, it is possible to compute the minute trend of the standard deviation SD_i:

$$SD_i = \sqrt{(RMS_i)^2 - M_i^2}\,. \tag{10}$$

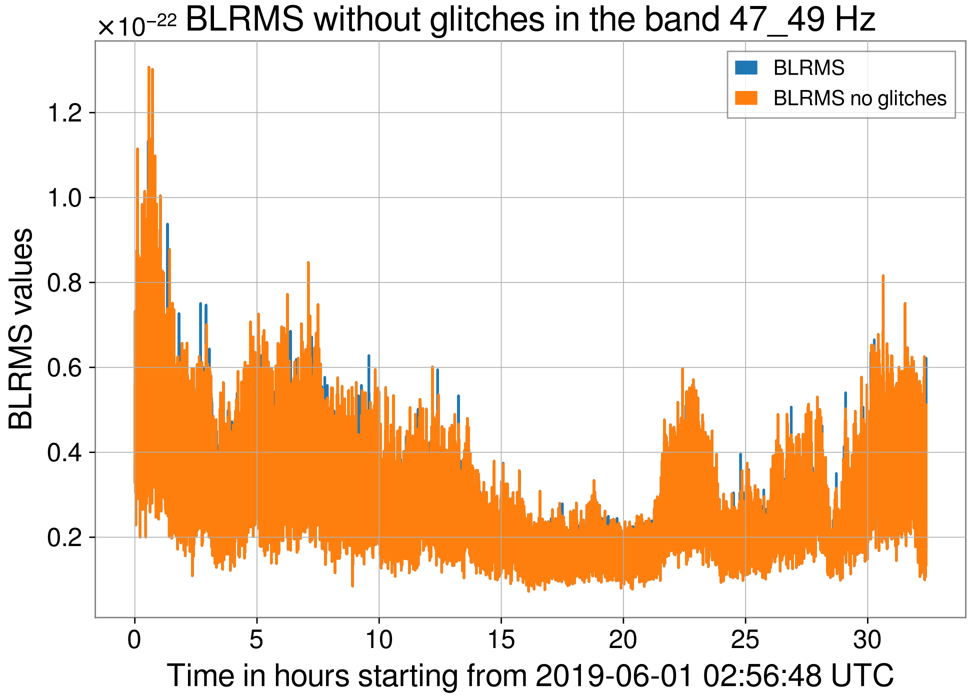

Figure 2: Plot of the BLRMS time evolution without glitches. The blue line shows the BLRMS time evolution after the procedure to remove glitches while the orange line shows the same evolution excluding remaining glitches using again the same procedure with a time range of 40 seconds. In this case, there are no remaining glitches and the two plots are almost coincident.

We use standard deviation because it is the root mean square of a signal's variation about the mean, rather than about zero.
Possible glitches in the auxiliary channels' time series could negatively influence the linear regression. To identify glitches in the mean and the standard deviation time evolution of each channel, we applied the method used for the BLRMS values.

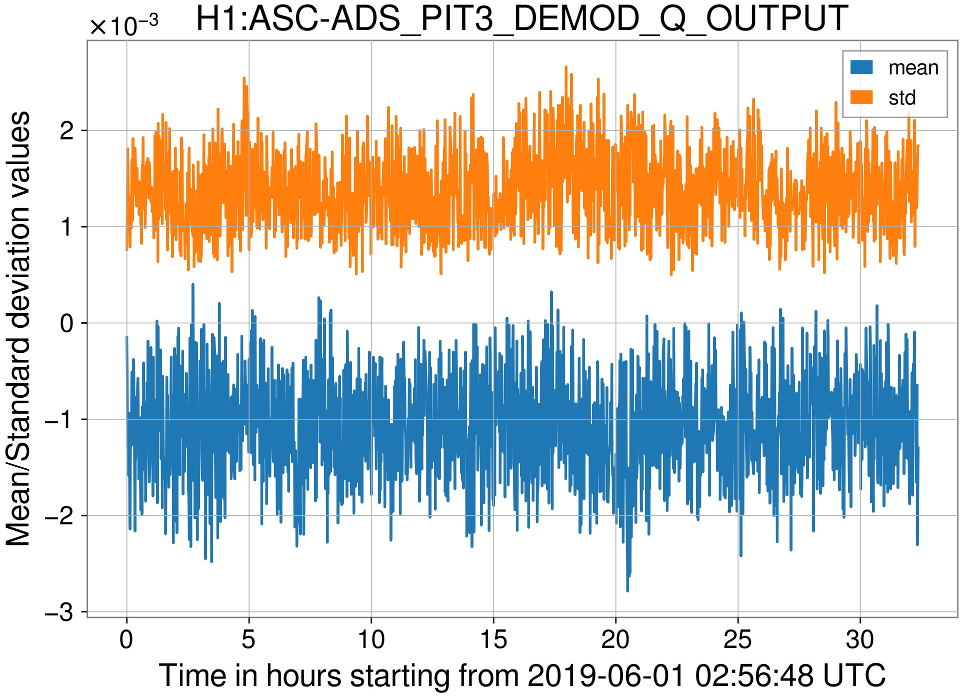

Figure 3: Plot of mean and standard deviation for the channel H1:ASC−ADS_PIT3_DEMOD_Q_OUTPUT, chosen for example, in the selected time range.

Since the sampling time for the auxiliary channels is 1 minute, to perform linear regression we need to re-sample the BLRMS data to the times of the auxiliary channels. We performed this resample using the average of the BLRMS in an interval of size 60 seconds centered around the considered auxiliary times.
Using the re-sampled BLRMS data and the auxiliary channels' time series, it is possible to reconstruct the y values and the matrix X of the equation 9. To estimate the β coefficients we used the function *pinv()* from the numpy python package [8] that calculates the pseudo-inverse of a matrix using its singular-value decomposition and including all large singular values.
In order to check for possible overfitting problems we separated the BLRMS dataset in a training set and in a test set of equal dimension. Then, we standardize the time series values by removing the mean and scaling to unit variance. For each channel and for the BLRMS, the mean and the variance are computed in the training set.
In the training set the β coefficients are estimated (see Figure 5) and used to predict the BLRMS values. In this way it is possible to compare the predicted values with the training and with the test set and plot the residuals. The residual plot, showed in the Figure 6 and 7, is the plot of the difference between the BLRMS values in the test set and the predicted values. Ideally, the residual should follow the same distribution as the added noise (ε in equation 6) which in real life situations

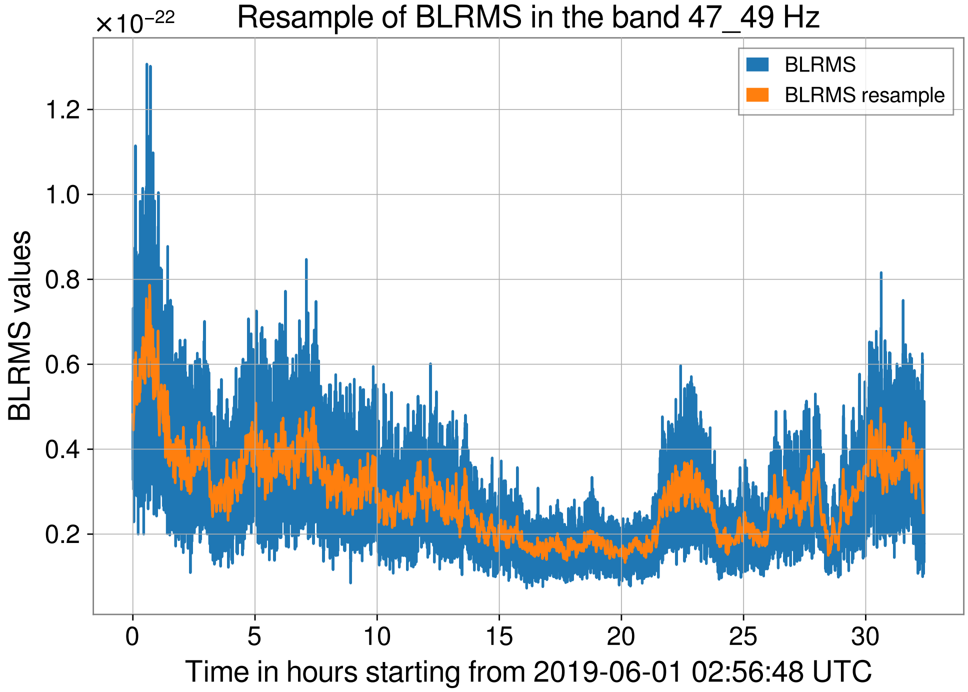

Figure 4: Plot of the re-sampled BLRMS. The blue line is equal to the orange line in Figure 2. The orange line shows the BLRMS time evolution with a sampling time of 60 seconds.

is not measurable.
In case of overfitting, it is expected that the predicted values fit almost exactly the training set and fail to predict the test set. In other words, overfitting means that the model captures the patterns in the training data, but fails to generalize well to unseen data [9]. If a model suffers from overfitting, we also say that the model has a high variance, which can be caused by having too many parameters that lead to a model that is too complex given the underlying data.
According to Figure 6 and 7, the linear regression exactly fits the training test but the prediction on the test set is totally wrong. There is no link between the BLRMS time evolution for this time range and this band but linear regression works perfectly on the training set anyway. The linear combination is the simplest model and the main problem is the great number of auxiliary channels used to fit the time evolution of the BLRMS. It is important to underline that, for now, we are just considering a small subset of the available auxiliary channels. Adding more channels would make the situation even worse.

6 Random Forest

In the last section we showed the overfitting problems of linear regression. In order to reduce these problems and to explore non parametric regressors[2], we chose Random Forest (RF).

[2] Machine learning algorithms can be grouped into parametric and non-parametric models. Using parametric models, we estimate parameters from the training dataset to learn a function that can classify new data points without

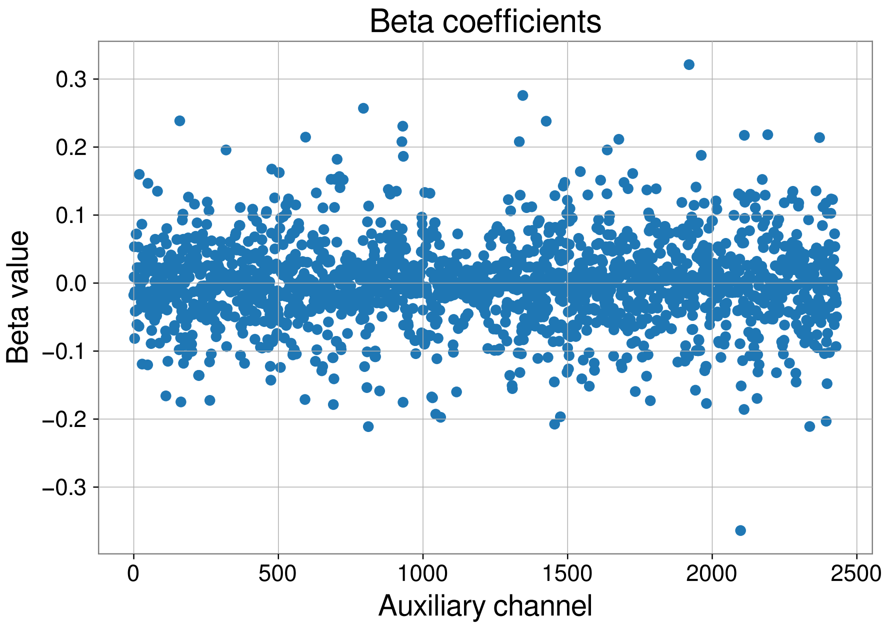

Figure 5: Plot of beta coefficients computed using linear regression for the training set. On the x axis, there is the index of ASC subsystem channel corresponding to the β value.

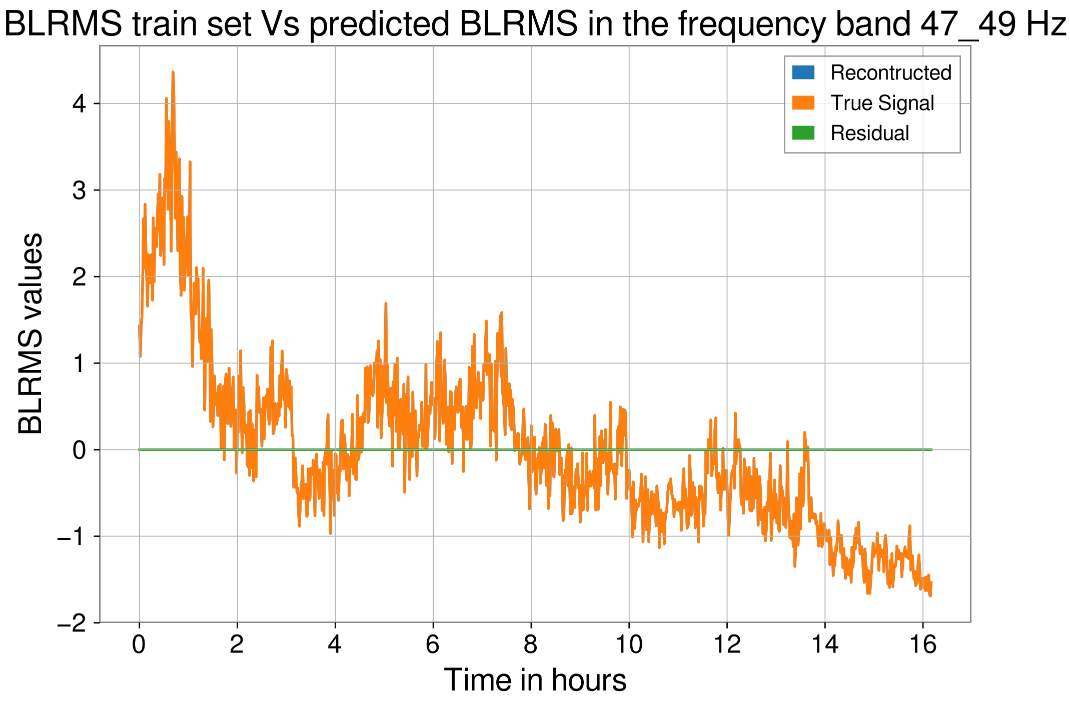

Figure 6: Plot of the predicted values and of the residuals for the train set.

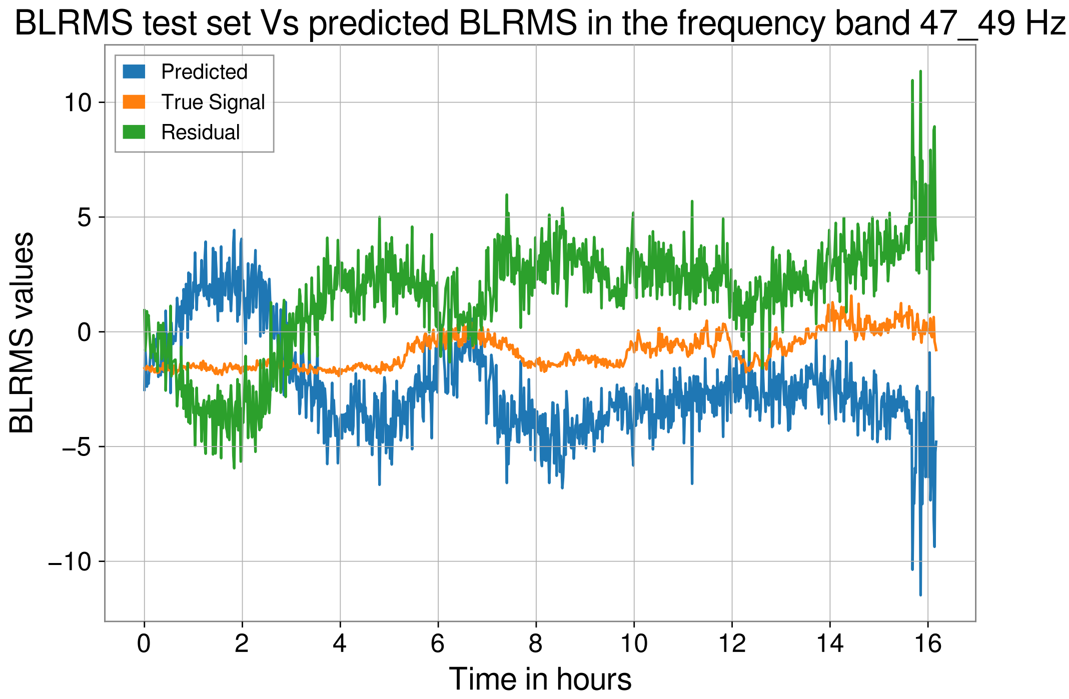

Figure 7: Plot of the predicted values and of the residuals for the test set.

A RF is an ensemble of multiple Decision Trees trained via bagging method. RF fits a number of decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control overfitting. The sub-sample size is always the same as the original input sample size. As regressor, RF can be understood as the sum of piecewise linear functions in contrast to the global linear regression models that we discussed previously. RF usually has a better generalization performance than an individual decision tree. In facts, instead of searching for the very best feature when splitting a node RF searches the best feature among a random subset of features [10]. This results in a greater tree diversity which helps to decrease the model's variance. As impurity metric we used the Mean Squared Error (MSE). MSE is simply the averaged value of the sum of the squared errors:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - \hat{y}^{(i)})^2\,, \tag{11}$$

where N is the number of samples, $y^{(i)}$ is the original value and $\hat{y}^{(i)}$ is the estimated value [9]. In the case of linear regression where $\hat{y}^{(i)} = \sum_{j=1}^{k} \beta_j x_{i,j}$, MSE is equal to the quantity S defined in equation 7, except for a constant. RF tries to minimize the impurity metric, that is the squared error function, using iterative methods and ensamble learning. We set the number of trees and the maximum number of nodes for each tree using grid search. Grid search is a brute-force search paradigm where we specify a list of values[3] for different hyperparameters, and the computer evaluates the model performance for each combination of those to obtain the optimal combination of values

requiring the original training dataset anymore. In contrast, non-parametric models can't be characterized by a fixed set of parameters, and the number of parameters grows with the training data [9].

[3] In this work the list for the number of trees is [30,50,70,100,120,140,160,180,200,250,300,500] while for the number of nodes for each tree is [5,7,9,12,15].

from this set [9]. For the chosen set, we set the number of trees to 50 each limited by a maximum of 5 nodes.

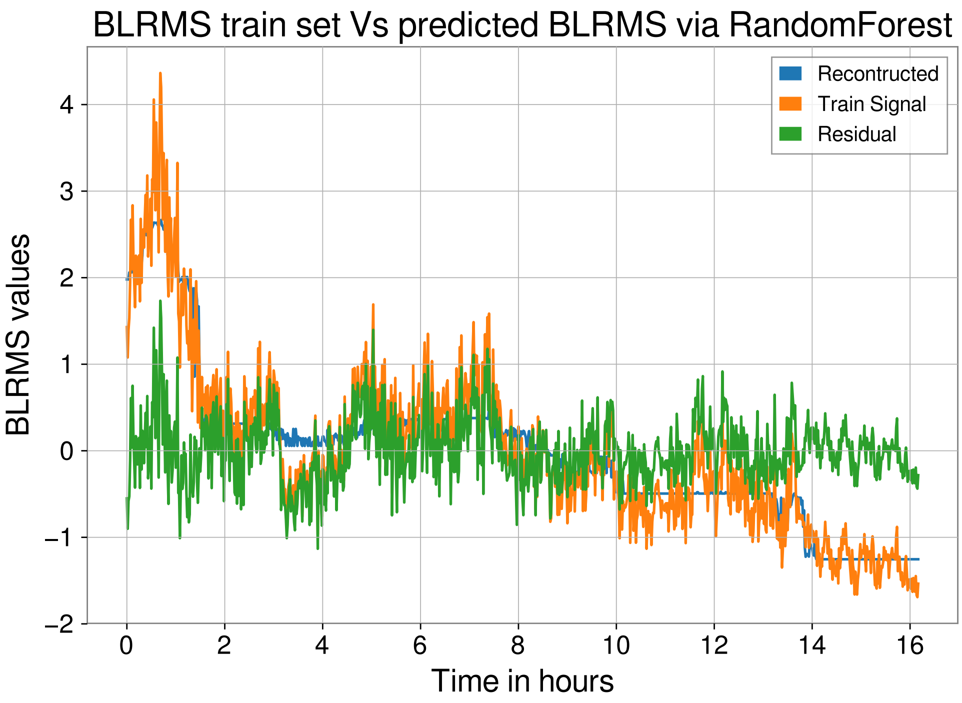

Figure 8: Plot of the predicted values and of the residuals for the train set via RandomForest.

The MSE is also useful to compare different regression models or for tuning their parameters via grid search and cross-validation.
In the case of RandomForest, the MSE value for the training set is $MSE_{train} = 0.15$ while for the test set is $MSE_{test} = 0.56$, while with linear regression MSE_{train} is practically zero and $MSE_{test} = 8.08$. Looking at the residual, it is clear that the RF does not even get close to a good prediction. The reason the MSE is lower in the RF is that the prediction is just smaller in amplitude, but not any closer to the real values.

7 Can we solve overfitting problems?

In general, RFs very often outperform linear regression but linear regression should be better than a RF when the underlying function is truly linear. The reason RF performs better on the test set is that we limited the number and the size of trees while linear regression is completely unconstrained. In both cases we need to solve overfitting problems because the prediction does not fit the expected values also for RF. First, we will try to implement a LASSO regression for a particular subsystem to reduce overfitting problems [11]. Then we have to enlarge the number of channels considering other subsystems and compare the three regressors (linear regression, RF, LASSO regression). Reducing the number of channels (like in the feature selection) for each subsystem, could be an idea to improve the regressors' predictions.

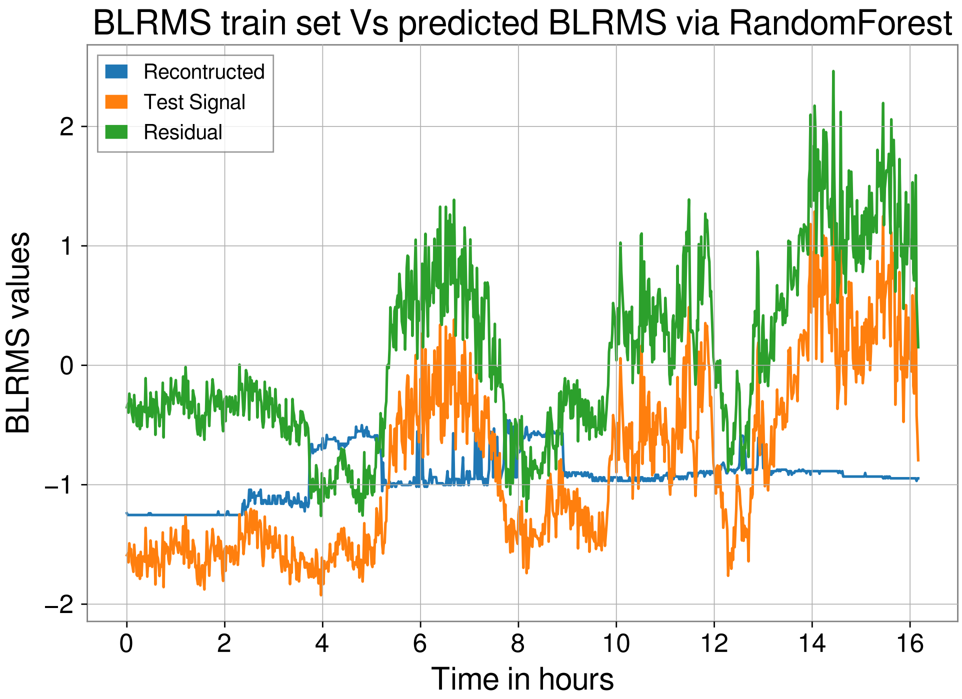

Figure 9: Plot of the predicted values and of the residuals for the training set via RandomForest.

8 Plan

In the Table 3, a general plan for the second half of the program is described.

Table 3: Planning

Weeks	Aims
week5	Analisys of other subsystems using linear regression and RandomForest. Application of the LASSO method to reduce overfitting problems.
week6	Find a criterion to select the ”important” channels for each subsystem.
week7	Study of the correlated auxiliary channels in the chosen frequency bands and test of possible idea to reduce overfitting using simulated data.
week8 - week10	Application of the proposed solutions to real data.

References

[1] G. Vajente, ”Analysis of sensitivity and noise sources for the Virgo gravitational wave interferometer”, Tesi di perfezionamento, Scuola Normale Superiore di Pisa, https://gwic.ligo.org/thesisprize/2008/VajenteThesis.pdf (2008)

[2] G. Vajente, ”Band-limited RMS”, aLIGO LHO Logbook - posted 14:55, Monday 22 April 2019 (48668), https://alog.ligo-wa.caltech.edu/aLOG/index.php?callRep=48668,

[3] G. Vajente, ”Compute band-limited RMS for LIGO O3 data”, https://git.ligo.org/gabriele-vajente/blrms (2019)

[4] G.Vajente, ”Study of dark-fringe noises slow non-stationarities during VSR1”, Virgo Internal note, VIR-009A-08 (2008)

[5] S. Pollock’s, ”The Classical Linear Regression Model”, Lectures at the University of Leicester, http://www.le.ac.uk/users/dsgp1/COURSES/MESOMET/ECMETXT/06mesmet.pdf

[6] D. Barker, ”aLIGO CDS Channel Naming Standards”, LIGO Scientific Collaboration, LIGO-T1000649-v1 https://dcc.ligo.org/public/0029/T990033/000/T990033-00.pdf (2011)

[7] aLIGO LHO Logbook, ”48Hz Peak in DARM”, https://alog.ligo-wa.caltech.edu/aLOG/index.php?callRep=50204, Wednesday 26 June 2019 (50204)

[8] NumPy v1.16 Manual, https://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.pinv.html

[9] S. Raschka, V. Mirjialili, ”Python Machine Learning”, Expert Insight (Second Edition 2017)

[10] A. Gron, ”Hands-On Machine Learning with Scikit-learn and TensorFlow”, O’Reilly, First Edition (2017)

[11] M. Walker et al., ”Identifying correlations between LIGO’s astronomical range and auxiliary sensors using lasso regression”, Classical and Quantum Gravity 35(22),0264-9381 (2018)