TOOLS FOR MODULATED NOISE STUDY

June 4, 2019

Barbara Patricelli
Dipartimento di Fisica dell'Università di Pisa & INFN Sez. Pisa

Giancarlo Cella
INFN Sez. Pisa

Contents

1 MONET: MOdulated NoisE Tool **2**

1.1 General description . . . 2

1.2 The script: monet.py . . . 2

1.2.1 How the script works . . . 3

1.2.2 How to use the code . . . 3

1.2.3 Outputs . . . 5

1.3 Where to find the code . . . 8

2 BRETON: BicoheRence EvaluaTiON tool **9**

2.1 General description . . . 9

2.2 The program: bico . . . 10

2.2.1 How the program works . . . 10

2.2.2 How to use the code . . . 10

2.2.2.1 Options . . . 10

2.2.2.2 Example . . . 12

2.2.3 Outputs . . . 12

2.2.3.1 Bicoherence data . . . 12

2.2.3.2 Summary data . . . 12

2.3 Where to find the code . . . 12

Chapter 1

MONET: MOdulated NoisE Tool

1.1 General description

MONET (MOdulated NoisE Tool) is a tool designed to investigate the side-bands observed around several lines in the interferometer noise spectrum. The main hypothesis at the basis of this tool is that the side-bands are due to some coupling of a carrier signal with the low-frequency (up to a few Hz) part of signal from an auxiliary channel.

1.2 The script: monet.py

MONET is a python script (monet.py) that looks for coherence between one main channel (e.g. dark fringe signal) and a combination of other existing channels, built under the above mentioned hypothesis.

1.2.1 How the script works

1. A main, reference signal is chosen (e.g., the LSC_DARM signal); we call it ch1.
2. A new signal is built in the following way:
   - A carrier signal is chosen (it can be a real signal or a simulated signal, i.e. a sinusoid with a chosen frequency); we call it ch2
   - A modulator signal is constructed by applying a low-band pass filter to a given, chosen signal from an auxiliary channel (ch3); we call it $\mathrm{ch3}_{\mathrm{low}}$.
   - A new signal (we call it ss) is constructed combining the carrier signal with the modulator signal; the signals are combined doing the product in the time domain: $\mathrm{ss}=\mathrm{ch2}*\mathrm{ch3}_{\mathrm{low}}$
3. The coherence between ch1 and ss is estimated in the overall frequency band of interest. Coherence values above a fixed threshold are stored in a table, together with the name of the corresponding modulator channels.

1.2.2 How to use the code

A short help can be obtained on the command line as:

```
$ python monet.py --help
```

The input parameters are the following:

- -a (--channel1)

  It is the main channel for the analysis (ch1); the default is LSC_DARM
- -c (--CarrierFrequency)

  Frequency of the carrier signal to be simulated. If 0 (this is the default value), a “true” signal (channel2) is used instead of a simulated signal
- -b --channel2

  Channel to be used as carrier if CarrierFrequency==0; the default is: ENV_CEB_UPS_VOLT_R
- -e (--channellist)

  It is a text file containing a list of channels to be investigated as ch3; the default file name is “ChannelList.txt”.

- -g (- -gps)

  It is the initial gps time of the data to be analyzed; the default value is 1213989318 (June 25, 2018).

- -d (- -dt)

  It is the interval of time to be analyzed, starting from gps; the default value is 2400 s.

- -z (- -timeslices)

  It represents the sub-intervals of time over which the averages are done when estimating the PSD; it determines the frequency resolution. The default value is 40 s.

- -r (- -resampling)

  It is the re-sampling frequency, i.e. all the channels are decimated accordingly to this frequency; the default value is 1000 Hz. Please note that the pipeline produces the output up to a frequency f=resampling/2.

- -t (- -threshold)

  It is the coherence threshold applied; the default value is 0.5.

- -f (- -freq list)

  It is an ascii file with a list of frequencies to be investigated in more details. Specifically, the ASD of the main channel, together with the noise projection based on the coherence values, are plotted in frequency intervals centered on the frequencies in the list and with overall amplitude Δf=4 Hz. The default file name is “frequencies.txt”.

- -n (- -TableFreqResolution)

  It is the frequency resolution of the final table in which the coherence values above the threshold are reported. The default value is 0.1 Hz.

- -o (- -outputDir)

  It is the working directory; all the directories created as output (see Fig. 1.1) are saved here.

1.2.3 Outputs

The outputs are organized in a directory structure (see fig. 1.1):

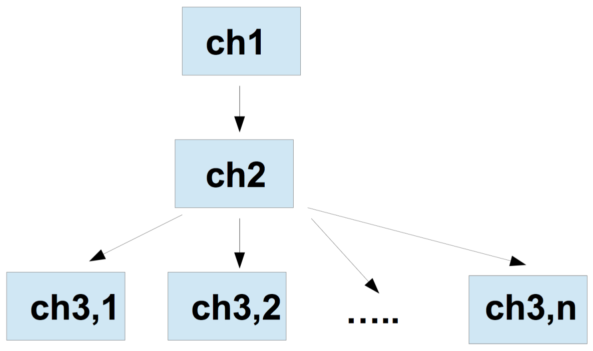


Figure 1.1: Output directory scheme

- ch1: Main directory. Its name indicates the main channel, the gps time and the time length of the segment; e.g.: LSC_DARM-gps_1235862060-dt_1200

- ch2: Secondary directory. Its name indicates the carrier channel used; e.g.: ENV_CEB_UPS_VOLT_R or Sin-50Hz.

- ch3,i: Directories for the modulator channels.

The content of the directories is the following.

- In ch2 there are three files: a plot (Coherence.png) and two ascii tables (Ranked-coherence.txt and Ranked-coherence-LowRes.txt).

  In Coherence.png (see fig. 1.2) the ASD of the main channel is plotted; superimposed on the ASD there are several red points, that mark the frequencies at which the coherence is above the fixed threshold.

  In the two ascii files (see fig. 1.3) there are three columns: in the first one there are the frequency bins; in the second column the coherence values above the threshold are reported and in the third column there are the corresponding modulator channels; the coherence values are ordered from the highest to the lowest. The two files differ from the frequency resolution. The file "Ranked-coherence.txt" is the high frequency resolution table and contains coherence values corresponding to frequency bins whose width is fixed by the "z" parameter; the frequency resolution is the same

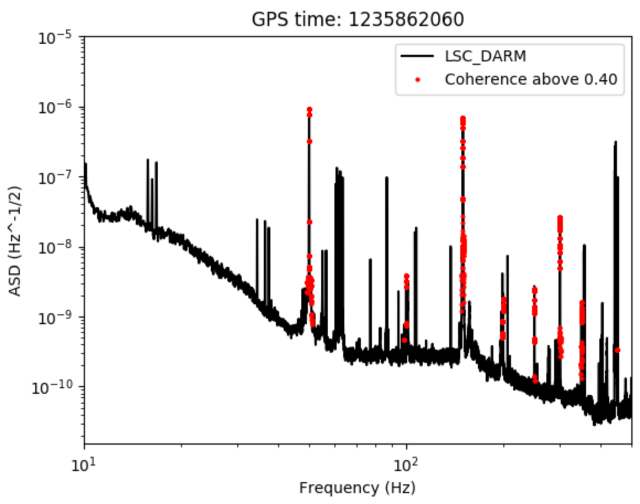
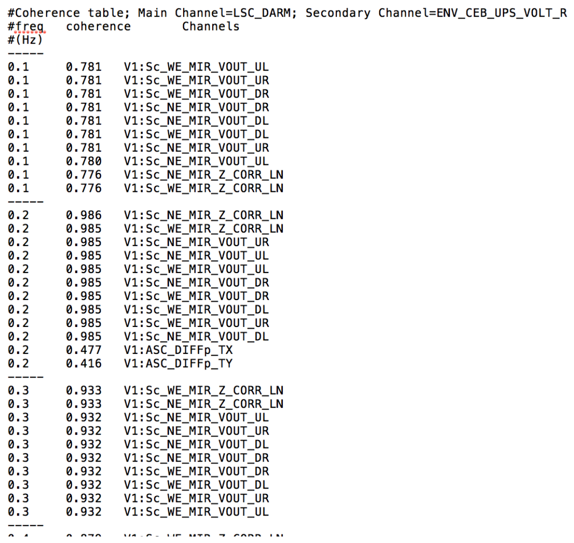


Figure 1.2: ASD of the main channel (LSC_DARM), together with red points marking the frequencies at which the coherence is above the chosen threshold.

of the output plots. The file "Ranked-coherence-LowRes.txt" contains coherence values averaged over frequency bins whose width is fixed by the "n" parameter; it is a lower resolution table, for a faster view of the results.



```
#Coherence table; Main Channel=LSC_DARM; Secondary Channel=ENV_CEB_UPS_VOLT_R
#freq   coherence       Channels
#(Hz)
-----
0.1     0.781   V1:Sc_WE_MIR_VOUT_UL
0.1     0.781   V1:Sc_WE_MIR_VOUT_UR
0.1     0.781   V1:Sc_WE_MIR_VOUT_DR
0.1     0.781   V1:Sc_NE_MIR_VOUT_DR
0.1     0.781   V1:Sc_NE_MIR_VOUT_DL
0.1     0.781   V1:Sc_WE_MIR_VOUT_DL
0.1     0.781   V1:Sc_NE_MIR_VOUT_UR
0.1     0.780   V1:Sc_NE_MIR_VOUT_UL
0.1     0.776   V1:Sc_NE_MIR_Z_CORR_LN
0.1     0.776   V1:Sc_WE_MIR_Z_CORR_LN
-----
0.2     0.986   V1:Sc_NE_MIR_Z_CORR_LN
0.2     0.985   V1:Sc_WE_MIR_Z_CORR_LN
0.2     0.985   V1:Sc_NE_MIR_VOUT_UR
0.2     0.985   V1:Sc_NE_MIR_VOUT_UL
0.2     0.985   V1:Sc_WE_MIR_VOUT_UL
0.2     0.985   V1:Sc_NE_MIR_VOUT_DR
0.2     0.985   V1:Sc_WE_MIR_VOUT_DR
0.2     0.985   V1:Sc_WE_MIR_VOUT_DL
0.2     0.985   V1:Sc_WE_MIR_VOUT_UR
0.2     0.985   V1:Sc_NE_MIR_VOUT_DL
0.2     0.477   V1:ASC_DIFFp_TX
0.2     0.416   V1:ASC_DIFFp_TY
-----
0.3     0.933   V1:Sc_WE_MIR_Z_CORR_LN
0.3     0.933   V1:Sc_NE_MIR_Z_CORR_LN
0.3     0.932   V1:Sc_NE_MIR_VOUT_UL
0.3     0.932   V1:Sc_NE_MIR_VOUT_UR
0.3     0.932   V1:Sc_NE_MIR_VOUT_DL
0.3     0.932   V1:Sc_NE_MIR_VOUT_DR
0.3     0.932   V1:Sc_WE_MIR_VOUT_DR
0.3     0.932   V1:Sc_WE_MIR_VOUT_DL
0.3     0.932   V1:Sc_WE_MIR_VOUT_UR
0.3     0.932   V1:Sc_WE_MIR_VOUT_UL
-----
```

Figure 1.3: Table with the coherence values.

- In the ch3,i directories, there are an ascii file (and a plot) with the coherence values associated with the modulator channel ch3,i estimated for the various frequency bins (see fig. 1.4); there are also several plots of the ASD "zoomed" around specific spectral lines (as from the frequency list file), together with the noise projection based on the coherence values (see fig. 1.5).

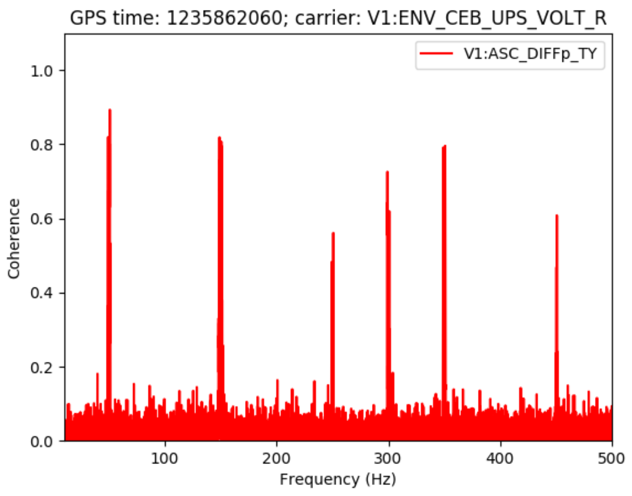


Figure 1.4: Coherence as a function of the frequency, for the modulator channel `V1:ASC_DIFFp_TY`

.

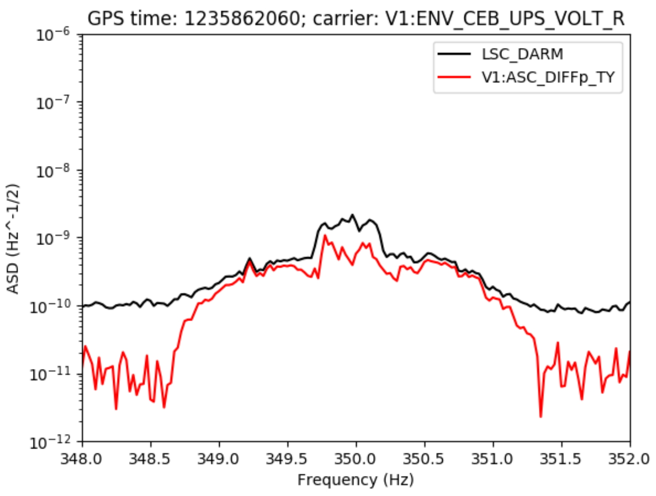


Figure 1.5: ASD around the 350 Hz spectral line (in black), together with the noise projection (in red), based on the coherence values obtained with the the modulator channel `V1:ASC_DIFFp_TY`.

1.3 Where to find the code

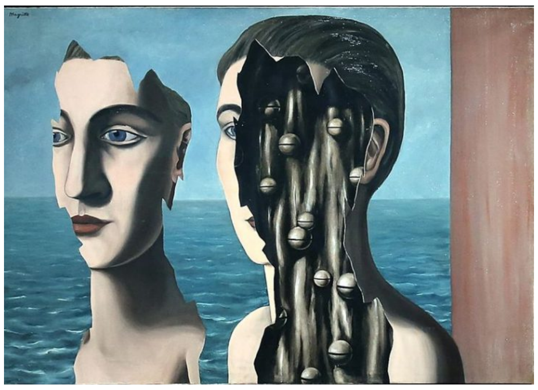
The tool is available in the SVN repository:

`https://svn.ego-gw.it/svn/advsw/DetcharScripts/trunk/scripts/MONET/`

In the MONET directory you can find the script (monet.py), together with two examples of input ascii files (the list of channels to be used as modulators, and the list of frequencies, see Sec. 1.2.2)

Chapter 2

BRETON: BicoheRence EvaluaTiON tool



2.1 General description

BRETON (BicoheRence EvaluaTiON tool) is a tool designed evaluate the bicoherence between three channels of data. The definition of the bicoherence used is the following:

$$b_{XYZ}(f_1, f_2) = \frac{\left|\sum_k \tilde{X}_k(f_1)\tilde{Y}_k(f_2)\tilde{Z}_k(f_1+f_2)^*\right|}{\sum_k |\tilde{X}_k(f_1)\tilde{Y}_k(f_2)\tilde{Z}_k(f_1+f_2)^*|} \tag{2.1}$$

where $\tilde{X}_k$, $\tilde{Y}_k$ and $\tilde{Z}_k$ are the Fourier transforms of the first, second and third channel evaluated over the k–th subsegment of a segment of data and the sum is over all the segments.

2.2 The program: bico

BRETON is a binary executable (bico) written in c, which evaluate the bicoherence between three channels. Currently all the channels must have the same sampling rate. The three channels can be the same one, or two of them can be the same, or they can be all different.

2.2.1 How the program works

1. The bicoherence is constructed using the data of three time series, which are read from three channels in a frame file
2. The first two channels (ch1 and ch2) are the reference ones. The bicoherence deals with the three channels in the same way, but the result is presented as a function of the frequencies of the reference ones.
3. A bidimensional array of data is generated. Row and columns correspond to the frequencies associated to ch1 and ch2, and the array data is the bicoherence estimate.
4. The array data are saved.
5. A file with a short report of the results is saved.

2.2.2 How to use the code

A short help can be obtained on the command line as:

```
$ bico -h
```

2.2.2.1 Options

The input parameters are the following:

- -c1 <ch1>

  <ch1> is the name of the first channel for the analysis.

  Default: There is no default value.

- -c2 <ch2>

<ch2> is the name of the second channel for the analysis.

Default: There is no default value.

- -c3 <ch3>

  <ch3> is the name of the third channel for the analysis.

  Default: There is no default value.

- -frame <framefile>

  <framefile> is the name of the frame file which contains the data.

  Default: There is no default value.

- -gpsstart <gps>

  <gps> is the starting gps time for the data to be used for bicoherence estimation.

  Default: There is no default value.

- -tlen <timelength>

  <timelength> is the length of the data segment used for each estimate of the bicoherence, in seconds. It determines the frequency resolution.

  Default: 1

- -nstat <statsamplings>

  <statsamplings> is the number of estimates of the bicoherence that are averaged.

  Default: 60s

- -prefix <prefixname>

  <suffix> is the suffix of the output files. The bicoherence array is saved in bico<suffix>, the summary file in summary<suffix>

  Default: .dat

- -outputdir <dirname>

  <dirname> is the name of the directory where the output files are saved.

  Default: .

2.2.2.2 Example

```
$ bico -frame test.gwf \
-c1 V1:ASC_B1p_QD1_V_56MHz_I \
-c2 V1:ASC_B1p_QD2_H_6MHz_I \
-c3 V1:ASC_B1p_QD2_H_6MHz_I \
-gpsstart 1185436000 \
-tlen 1 \
-nstat 90
```

2.2.3 Outputs

The results are saved in a directory which can be changed with the -outputdir option. A description of the output files follows.

2.2.3.1 Bicoherence data

This is an ASCII file with three columns, separated by spaces. The format of each row is the following:

$f_1 \quad f_2 \quad b_{XYZ}$

where f_1 and f_2 are the frequency of the first and the second channel, and b_{XYZ} is the corresponding bicoherence value.

2.2.3.2 Summary data

An ASCII file with the parameters of the run.

2.3 Where to find the code

The tool is available in the SVN repository:

`https://svn.ego-gw.it/svn/advsw/DetcharScripts/trunk/scripts/BRETON/`