‘USER’S MANUAL’ OF MLE OF FOURIER SPECTRA

20 July 1998, Version 1.0

THIERRY APPOURCHAUX
Space Science Department
ESTEC, NL-2200 AG Noordwijk, The Netherlands

Warning

Writing ‘user’s manual’ is not something I am used to. There is certainly room for improvement. If you find something unclear, please send me an e-mail to thierrya@so.estec.esa.nl. I will be happy to post a newer and clearer version.

1. What is the purpose of this software?

The software provided will allow you to fit Fourier spectra of p modes. The mathematical background of the MLE of Fourier spectra can be found in Appourchaux et al (1998) (Part I, A&A Sup. Series or on http: virgo.so.estec.esa.nl).

2. How do I install the software?

You use the following unix command to uncompress and extract the archive:

- gunzip gong.tar.gz
- tar -xvf gong.tar

This will create the following files:

- Data file: specsohogongtri1
- Evaluation routines:
    - -crossspectrum.pro
    - -crossnoise.pro
    - -routines_gong.pro
- Fitting routines:
    - gongamp.pro
    - routines_gong.pro
    - fitplot_gong.pro
    - MLEfit_gong.pro
- Other useful files:
    - -tvplot.pro
    - -ct.pro
    - -legendre.pro
    - -compile_gong.pro

Each file is described in the section 'What are these files?'

3. How do I fit the data?

The fitting of Fourier p-mode spectra is based on the knowledge of 2 very important matrices:

- the leakage matrix,
- the noise covariance matrix.

Both matrix can be estimated directly from the data. The usual recipe for fitting the data is the following:

1. Estimate the leakage matrix using the *crossspectrum* routine,
2. Estimate the noise covariance matrix using the *crossnoise* routine,
3. Fit the data using *gongamp*

3.1. Step 1: Using the *crossspectrum* routine

This routine can help you to visualize the covariance matrix of the data (See Appourchaux et al, 1998, Part II, A&A Sup. Series or on http: virgo.so.estec.esa.nl) and to get a first estimate of the leakage matrix. An example is given hereafter:

```
IDL> .run tvplot
% Compiled module: TVFRAMEMOD.
% Compiled module: CONGRID.
% Compiled module: MINMAX.
% Compiled module: RESIZE.
IDL> .run crossspectrum
% Compiled module: CROSSSPECTRUM.
% Compiled module: ECHELLEPLOTPILE.
```

```
IDL> crossspectrum,1,'','gong'
Which one to plot: Real (1), Imaginary (2) or Real+Imaginary (3)=3
3.00000
% READFITS: File may contain FITS extensions
% READFITS: Now reading 259200 by 3 by 2 array
```

The need for sign correction can be checked by looking at the Real (1) and the Imaginary (2) before looking at the sum (3), check it out! This one here is specific to the LOI!

```
Sign correction (1 if Yes)=1
Sign correction being done... Clean data from m (or l) leaks? (1 if Yes)=1
Use LOI leakage matrix? (1 if yes)0
Leakage matrix computation valid for GONG only!
Redefine l=1 leakage matrix? (1 is yes)0
1.0000000 0.0000000 -0.55389500
0.0000000 1.0000000 0.0000000
-0.55389500 0.0000000 1.0000000
Start cleaning
End cleaning
Resolution= 0.0321502
-636.277 2403.57
resolution 0.0321502
resolution 0.0321502
```

Examples of these diagrams are given in Figure 1. The routine above can also help you also to clean the data from m leaks, and then visualize if

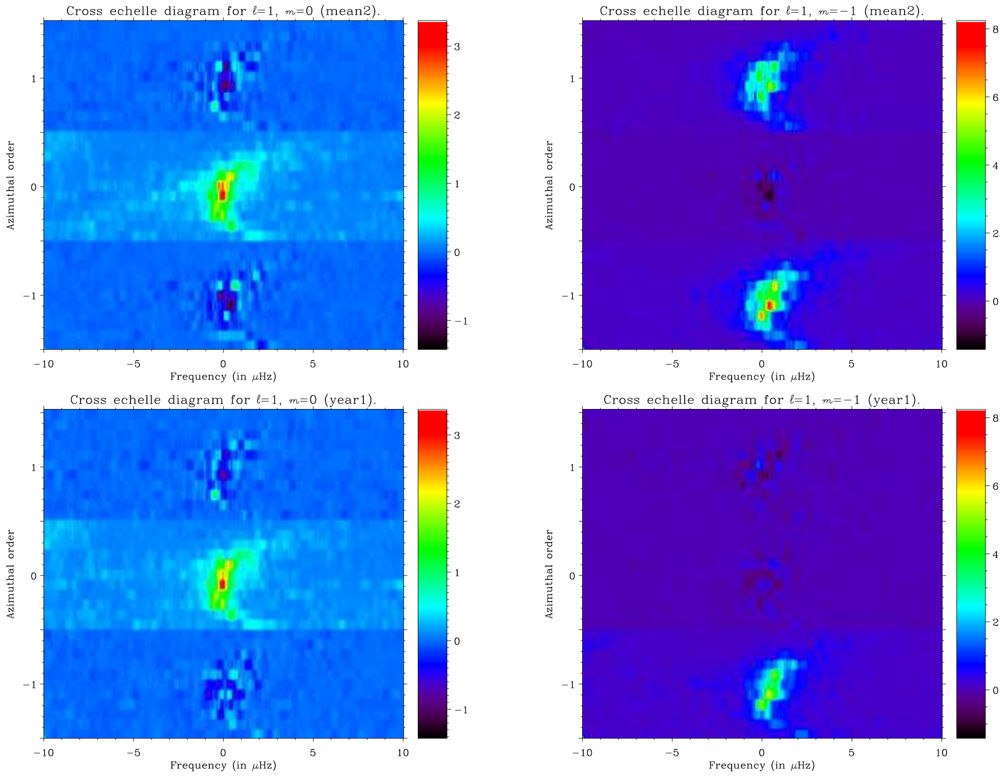

Figure 1. Cross echelle diagrams of uncleaned LOI data (top) and cleaned LOI data (bottom). Created using the routine *crossspectrum*.

the data are effectively cleaned (See Figure 1). The cleaning of the data is a tool for verifying to the first order that the computed leakage matrix is correct.

3.2. Step 2: Using the *crossnoise* routine

This routine can help you to visualize the correlation between the cross spectra, especially for the noise. You can visualize the so-called ratio matrix of the data (See Appourchaux et al, 1998, Part II, A&A Sup. Series or on http:
virgo.so.estec.esa.nl). An example is given hereafter:

```
IDL> .run crossnoise
% Compiled module: CROSSNOISE.
```

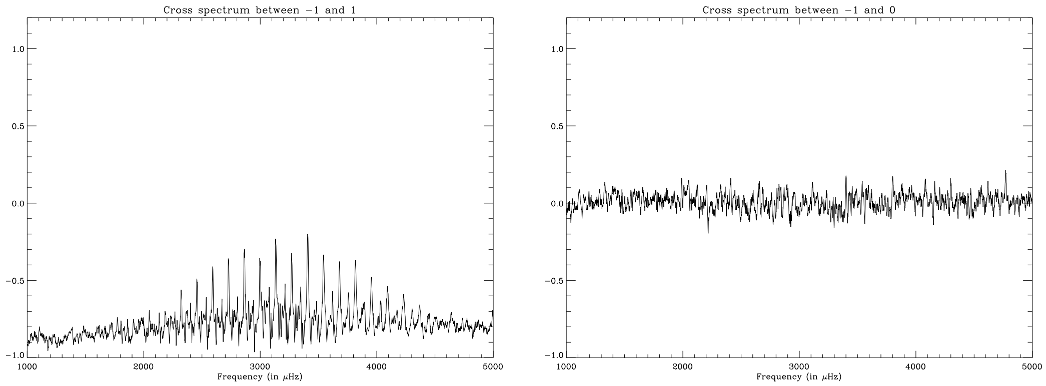

Figure 2. Cross spectra of uncleaned GONG data for $m = -1$ and $m' = -1$ (left), and for $m = -1$ and $m = 0$ (right).

IDL> crossnoise,1,”,’gong’,1000.,5000.
Which one to plot: Real (1), Imaginary (2) or Real+Imaginary (3)=3
3.00000 Between real-real(0) or real-imag(1)0

The need for sign correction can be checked by looking at the Real (1) and the Imaginary (2) before looking at the sum (3), check it out! This one here is specific to the LOI!
Sign correction (1 if Yes)=1
Sign correction being done...
Clean data from m (or l) leaks? (1 if Yes)=0

Resolution= 0.0321502
1.00000 0.00000 NaN NaN
0.00878854 0.0221489 0.0346441 0.481236
-0.774851 0.0249816 0.668368 6.65247
: 0
0.00793013 0.0140508 0.0260151 0.228410
1.00000 0.00000 NaN NaN
0.00326841 0.0137392 0.00221178 0.166712
Examples of these cross spectra ratios are given in Figure 2.

3.3. Step 3: Using the *gongamp* main routine

3.3.1. *Before using it!*

Before using the main routine, you need to compile lot ot routines using *compile_gong* like:

```
IDL> @compile_gong
% Compiled module: CROSSTALKGONG.
% Compiled module: LORENTZIAN.
% Compiled module: LORENTZIAN_ASS.
% Compiled module: LORENTZIAN_RAK.
% Compiled module: FUNCTION_ROT.
% Compiled module: DMM.
% Compiled module: COMBI.
% Compiled module: P1.
% Compiled module: P2.
% Compiled module: P3.
% Compiled module: P4.
% Compiled module: P5.
% Compiled module: CROSSTALKGONG.
% Compiled module: LORENTZIAN.
% Compiled module: LORENTZIAN_ASS.
% Compiled module: LORENTZIAN_RAK.
% Compiled module: FUNCTION_ROT.
% Compiled module: DMM.
% Compiled module: COMBI.
% Compiled module: P1.
% Compiled module: P2.
% Compiled module: P3.
% Compiled module: P4.
% Compiled module: P5.
% Compiled module: MLEFIT.
% Compiled module: LIKELIHOODAMPL.
% Compiled module: DLIKELIHOODAMPL.
% Compiled module: LIKELIHOODAMPLFIXED.
% Compiled module: DLIKELIHOODAMPLFIXED.
% Compiled module: HESSIAN.
% Compiled module: DERIVEE2XY.
% Compiled module: MLEFIT.
% Compiled module: LIKELIHOODAMPL.
% Compiled module: DLIKELIHOODAMPL.
% Compiled module: LIKELIHOODAMPLFIXED.
% Compiled module: DLIKELIHOODAMPLFIXED.
```

% Compiled module: HESSIAN.
% Compiled module: DERIVEE2XY.
% Compiled module: FITPLOT_GONG.

3.3.2. *What does this routine do?*
This *gongamp* routine sets all the important starting parameters and which parameters to be fitted. There 4 important commons to be set up in this routine.

- crosstalk,ctnm1: set the leakage matrix for l
- nb_bruit, nbbruit: set the number of independent noise to be fitted (usually l+1)
- matbruit,croiseen: set the noise covariance matrix
- profile,profilename: routine name for the line profile

Here is an example of one should set the leakage matrix:

```
ctnm1=fltarr(2*degree+1,2*degree+1)

for i=0,2*degree do begin
ctnm1(i,i)=1.0
endfor
```

In this specific example, the leakage matrix is the identity matrix. Herafter, you have an example of how one should set the noise covariance matrix for $l = 1$ with `nbbruit=2`:

```
if (degree eq 1) then begin
 croiseen(0,*,*)=   [[1.,0.,0.3],$
                     [0.,0.,0.],$
                     [0.3,0.,1.]]

 croiseen(1,*,*)=   [[0.,0.,0.],$
                     [0.,1.,0.],$
                     [0.,0.,0.]]
endif
```

There is also some 'freedom' in choosing the line profile. But this should be taken with care because this line profile is used by the routine *crosstalkgong* which compute the covariance of the data (See Appourchaux et 1l, 198, Part I).

Note: If one wants to fit the data using its own covariance matrix, one should only write a routine similar to *crosstalkgong* and using its own line profile.

3.3.3. *How do I use it?*

You start the fitting process doing the following: IDL> gongamp
give the degree of the mode= 1

give the wid card of your file as a string=

give the starting point for the fit= 8

give tolerance for the Powell algorithm (1d-08 works well on a Sparc20...)=
0.000001

give the printer output as a string (press return to redirect to your monitor)=

window size (in microHz) 20.

Number of independent noise (3 is the usual for the LOI)2

Asymmetry (1 if yes)0

```
SIMPLE = T / Written by IDL: 19-Feb-1998 11:07:47.00
BITPIX = -32 /
NAXIS = 3 /
NAXIS1 = 259200 /
NAXIS2 = 3 /
NAXIS3 = 2 /
DATE = '19/02/98' /
END
1.00000 0.00000 0.00000
0.00000 1.00000 0.00000
0.00000 0.00000 1.00000
: 0
1.0000000 0.0000000
0.0000000 0.0000000
0.30000001 0.0000000

0.0000000 0.0000000
0.0000000 1.0000000
0.0000000 0.0000000

0.30000001 0.0000000
0.0000000 0.0000000
```

1.0000000 0.0000000
: 0
00 2.00000
001111 2.00000

****

Central frequency 2828.12

****
1.00000e-06
Window size= 19.9974
00011110000011111
001111
0.018193832 -0.085188953 0.44312925 0.013210602 0.0000000
0.0000000 0.0000000 -1.6466484 -1.6466484 -1.6466484
-2.4086836 -2.4086836 -100.00000 -100.00000 -100.00000
-100.00000 0.0000000
2 622 3 9 1866
Perform fit on the amplitude spectra
plot amplitude
plotted

1.00000e-06
1 1
Fitting...
inalgo= 110
0 1 2 7 8 9 10 11
inonfixed= 8
3 4 5 6 12 13 14 15 16
ifixed= 9
cc is **************** 2 622 3 9 1866
Creating new commons...
likelihoodamplfixed dlikelihoodamplfixed
0.018193832 -0.085188953 0.44312925 -1.6466484 -1.6466484
-1.6466484 -2.4086836 -2.4086836
We use the Powell algorithm
N evaluation= 1169.00
Powell finished, Fmin= -5944.4580

-0.017063517 -0.091914740 0.39084839 -1.4794212 -0.84078381
-1.6857260 -2.4359077 -2.1992669

```
Number of iterations= 6 6

Fmin= -5944.4580
N evaluation= 1169.00
8
2 622 3 9 1866
Perform fit on the amplitude spectra
plot amplitude
plotted

1.00000e-06
1 1
Fitting...
inalgo= 110
0 1 2 3 7 8 9 10 11
inonfixed= 9
4 5 6 12 13 14 15 16
ifixed= 8
cc is **************** 2 622 3 9 1866
Creating new commons...
likelihoodamplfixed dlikelihoodamplfixed
-0.017063517 -0.091914740 0.39084839 0.013210602 -1.4794212
-0.84078381 -1.6857260 -2.4359077 -2.1992669
We use the Powell algorithm
N evaluation= 666.000
Powell finished, Fmin= -5947.0535

-0.012485522 -0.094410461 0.39943381 -0.047079569 -1.4741397
-0.85562870 -1.6795931 -2.4368353 -2.1975863

Number of iterations= 3 3

Fmin= -5947.0535
N evaluation= 666.000
9
Delta lmin= 2.5954570
-5947.0535
```

4. What are these files?

4.1. Data format: an example with *specsohogongtri1*

All spectra are saved as array of $N_{points} \times 2l+1 \times 2$, where N_{points} is the total number of data points of the times series divided by 2 (representing the frequency), the second index represents the azimuthal order m, the last index represents the real (0) and imaginary (1) parts of the Fourier spectra. The file is in a fits formate read with the following statement:

```
IDL> data=readfits('specsohogongtri1')
% READFITS: File may contain FITS extensions
% READFITS: Now reading 259200 by 3 by 2 array
```

There are 2 files:

- *specsohogong1*: the regular l=1 GONG data
- *specsohogongtri1*: the $l = 1$ GONG data cleaned from the $l = 6, 9$ aliases (See Appourchaux et al, 1998, Part II)

4.2. Evaluation routines

The routines *crosssepctrum* and *crossnoise* are descibed in Sections 3.1 and 3.2.

4.3. Fitting routines

4.3.1. *The* gongamp *routine*

This routine is described in Section 3.3.

4.3.2. *The* routines-gong.pro *file*

The *routine_gong.pro* file is a collection of useful routines that are sometimes better described in their header:

- crosstalkgong: compute the covariance matrix of the data to be fitted
- lorentzian: regular lorentzian profile
- lorentzian_ass: first version of Nigam's asymmetrical profile with proper normailzation
- lorentzian_rak: last version of Nigam's asymmetrical profile
- function_rot: compute the $d^{(l)}(\beta)$ matrix for a given β (or B) angle and l (See Edmonds, 1963, *Angular momentum in Quantum Mechanics*)
- dmm: compute the coefficients of the matrix $d^{(l)}(\beta)$ using Jacobi polynomials
- combi: compute C_r^n
- p1 to p5: compute Clebsh-Gordan polynomials

4.3.3. *The* fitplot-gong *routine*

This is the routine to be called for fitting the data. Most of its feature are described in the header and an example of calling it is given in the routine *gongamp*.

4.3.4. *The* MLEfit-gong.pro *file*

The *routine_gong.pro* file is a collection of useful routines that are sometimes better described in their header:

- MLEfit: this is a general routine which select which likelihood function to be used. In the case of Fourier spectra fitting only one routine is used. This routines is described also in the header.
- likelihoodampl: compute likelihood using a model of covariance of the data (never used in practice!)
- dlikelihoodampl: compute derivative of the above (never used in practice!)
- likelihoodamplfixed: compute likelihood using a model of covariance of the data when some parameters are fixed (always used!)
- dlikelihoodamplfixed: compute derivative of the above (alwyas used!)
- hessian: compute hessian for a given MLE function
- derivee2xy: compute second derivative with respect to x and y

4.4. Other useful files

Here is another collection of useful routines of files:

- tvplot.pro: collection of file making nicer plot than tvscl (See Figure 1)
- ct.pro: compute leakage matrix of GONG for a given l
- legendre.pro: compute Legendre polynomials
- compile_gong: macro command for compiling files