	JMMC-MAN-2500-0001 Revision 0.2 Date: 18/07/2014

JMMC

WISARD USER MANUAL

Authors:
L. Mugnier <> — ONERA
S. Meimon <> — ONERA
M. Vannier <> — Laboratoire Lagrange
A. Domiciano de Souza <> — Laboratoire Lagrange
JMMC Image Reconstruction team <> — JMMC

Author: Institute:	Signature: Date:
Approved by: Institute:	Signature: Date:
Released by: Institute:	Signature: Date:

Change record

Revision	Date	Authors	Sections/Pages affected
	Remarks		
1.0	10/07/2014		all

Table des matières

Liste des tableaux

Table des figures

1 Introduction

1.1 Object

This describes WISARD (Weak-phase Interferometric Sample Alternating Reconstruction Device), the image reconstruction software developed by ONERA in WP 2.5 of the JRA 4, within the framework of the FP 6. Together with the software archive, it constitutes ONERA's contribution to the outputs of WP 2.5 (Image Reconstruction Tools). The present version is an update from the original ONERA's document, after some updates and improvements were made on the software by M. Vannier, A. Domiciano de Souza, and the JMMC.

WISARD is a software for the reconstruction of images from interferometric data, such as the ones that can be recorded at ESO-VLTI or other optical/IR interferometers.

It is based on (and quite thoroughly described in) the PhD thesis work of S. Meimon [1] and in references [2, 3, 4]. Ref. [2] is the first presentation of the method, Ref. [3] describes the noise model, and Ref. [4] is an overview of WISARD with reconstructions from both simulated and experimental data.

The WISARD software described here is a complete rewrite (from scratch) by S. Meimon and L. Mugnier of earlier programs of the aforementioned thesis. This rewrite has bought us a dramatic gain in speed and in code maintainability as well as the elimination of quite a few bugs. This code and its documentation are distributed in the hope that it will be useful, but on an *as is* basis, without any warranty of any kind. More precisely, WISARD is a free software, licensed under the CeCILL-B license version 1, which can be found at http ://www.cecill.info.

WISARD is written in the commercial language IDL of ITT Visual Information Solutions [1]. It is compatible with the IDL Virtual Machine, so that it can be run without purchasing an IDL license. Additionally, WISARD should also run with GDL, the GNU Data Language, however only in its last version and on some platforms. GDL is available at http ://gnudatalanguage.sourceforge.net/.

Chapters 2 and 3 contain the instructions for installing WISARD on your computer and the description on the input data format constraints. Chapter 4 documents the use of WISARD and contains an example of reconstruction from interferometric data. The data comes from the Imaging Beauty Contest 2004 organized by Peter Lawson for the IAU [5]. The example batch file is part of the distribution and can be used and modified for self-study.

Chapter 5 contains an in-depth description of the method implemented by the WISARD code. For more details on the method, the interested reader should consult [1, 2, 3, 4]. Ref. [1] is available on the French national multidisciplinary thesis server TEL at the following address : http ://tel.archives-ouvertes.fr/docs/00/05/44/98/PDF/these_finale.pdf. The two following references are also available on-line, at http ://laurent.mugnier.free.fr/publis/Meimon-JOSAA-05.pdf and http ://laurent.mugnier.free.fr/publis/Meimon-OL-05.pdf, respectively.

1.2 Reference documents

[1] JMMC-MAN-0000-0001, Revision 2.0, LaTeXdocument style manual

1.3 Abbreviations and acronyms

JMMC	Jean-Marie Mariotti Center
CeCILL	Ce[a] C[nrs] I[nria] L[ogiciel] L[ibre] free software license agreement

[1] http ://www.ittvis.com/idl/

EII	European Interferometry Initiative
FITS	Flexible Image Transport System
FP 6	Sixth Framework Programme of the European Union
GPL	GNU General Public License
IAU	International Astronomical Union
JRA 4	Join Research Activity 4
NA	Not Applicable
OI FITS	Optical Interferometry FITS
VLTI	Very Large Telescope Interferometer
WP	Work Package
PSD	Power Spectral Density
OLBI	Optical/IR Long Baseline Interferometry

2 WISARD 's package : installation and structure

2.1 Requirements

The WISARD software should run on all platforms where (a) IDL or GDL[2] is installed, and (b) the OptimPack optimization package of Éric Thiébaut (see Section 2.4) is installed. This means that WISARD can be installed and run on at least all Unix-like platforms.

2.2 Downloading

WISARD code, batches and documentation are available as a package, distributed on the JMMC web page : http ://www.jmmc.fr/wisard

2.3 Components of the unpacked software

WISARD is distributed in a compressed archive (`.tgz`). Unpacking it is straightforward and will create a `WISARD-<version>/` directory under which is the distribution. From the shell prompt, type :

```
tar xvfz WISARD-<version>.tgz
```

[2]However, only for the latest GDL versions and on some platforms.

The directories created under `WISARD-VERSION/` and the nature of their contents are the following :

`./doc/`	This documentation of WISARD .
`./lib/`	WISARD , its sub-routines, and routines used by them :
`./lib/wisardlib/`	the WISARD main routine and its sub-routines. These routines are under the CeCILL-B license.
`./lib/extralib/`	Several extra routines : – the routine to translate OIFITS data into the data format directly used by wisard ; – routines used by the former code to read the OIFITS format, developed by J. D. Monnier (see http ://dept.astro.lsa.umich.edu/~monnier/oi_data/) ; – some routines from NASA's astro library, used in turn by the former routines, available from http ://idlastro.gsfc.nasa.gov/.
`./lib/oneralib/`	ONERA routines necessary to WISARD . These routines are *not* part of WISARD but used and distributed by permission of their respective authors. These routines are under the CeCILL-C license.
`./lib/optimpacklib/`	OptimPack library for IDL (`OptimPack_IDL*.so`) and its IDL frontend routines (`op_*.pro`), by É. Thiébaut. This library is *not* part of WISARD but is used by WISARD and distributed here by special permission of the author.
`./optimpack/`	OptimPack sources (`*.c *.h`) and `Makefile` (see Section 2.4), plus a copy of the IDL frontend routines (`op_*.pro`). Once again, this library is *not* part of WISARD but is used by WISARD and distributed here by special permission of the author. These routines are under the GPL.
`./pro/`	example batch file for WISARD . A good starting point for self-study, cf Section 4.2.
`./inputdata/`	input data for the example batch file.

2.4 Dependencies

WISARD heavily uses Éric Thiébaut's neat OptimPack software as its minimization engine. So WISARD needs the OptimPack library for IDL (`OptimPack_IDL*.so`) and its IDL frontend (`op_*.pro` files) to be installed. Although OptimPack is *not* part of WISARD , we have included the OptimPack IDL frontend and the pre-compiled OptimPack library for IDL (for Linux, Mac/OS and Solaris, on 32 and 64-bit architectures, and for Windows) in the `lib/optimpacklib/` directory of the WISARD distribution, with their author's permission. So if your computer runs on one of the architecture mentioned above, you should try

these and it is likely to work immediately.
Should these pre-compiled librairies not work for you, we have also included all the necessary sources (`*.c *.h` and `Makefile`) under the `optimpack/` directory. A `make` command from the `optimpack/idl` directory should build the library for your architecture. Then simply copy the resulting `OptimPack_IDL*.so` files in the `lib/optimpacklib/` directory.

2.5 Installing WISARD

Once you have unpacked the compressed archive (Sect. 2.3), and compiled the OptimPack library for IDL if necessary (Sect. 2.4), no particular installation is required. We will have, however, to set for every new session (or hard-code) the path to WISARD and to launch the initialization batch, as explained in Sect. 4.2.

3 Field of application and constraints on the input data format

The main routine of WISARD is `wisard.pro`. The input data for `wisard.pro` is typically recorded by an optical interferometer measuring, the simultaneous interferences of at least three telescopes. It consists in squared visibilities and closure phases, the error bars on these quantities, and the spatial frequencies corresponding to each data point.
WISARD does not use directly the interferometric data contained in the OIFITS file(s) given as input by the user. It first reads and translates them into a data structure readable by the reconstruction code itself. This WISARD structure, detailed in Sect. 5.2.3, is then used for making the myopic and convexified data models (see chapter 5). Due to the constraints from these latter steps (matrix structure, relationship between phase closure triplets, and visibility baselines for estimating the phase,...), some strong requirements apply on the OIFITS data used as an input :

- Input data format should strictly follow the standard OIFITS (Pauls et al., 2005). The structure of the data is given in Section 4.3 and more precisions are available in chapter 5.
- Within each file, there should be coherence between closure triplets and visibility baselines. That is, any visibility baseline should be associated with at least one phase closure triplet (i.e. the triplet should include that baseline). If closures and visibilities were observed simultaneously, this is implicitly done using OIFITS format. On the other hand, redundancy on phase closures is possible.
- Several tables of visibility and closure data in the same file are possible. In that case, each visibility table should be associated with closure phase table, successively in their order within the OIFITS structure.
- If several data tables are used within a file, they might refer to either a single wavelength table, or to different ones. In the latter case, it is assumed that the successive wavelength tables are associated with the visibility and closure data tables in the same order.
- There should be some unicity on the global dimensions between the tables of a file or between different files used together as an input. This means that different visibility tables should contain the same number of baselines or the same number of times of observation (assuming the same number of spectral channels were used). The data from a new table can therefore be merged to the previous one in a matrix expression required by WISARD . This, of course, forbids in general to use a set of heterogeneous data observed using various instruments or various number of baselines. **IS IT OK NOW ? It therefore represents a strong limitation of the software, which might be relaxed in the future if enough efforts is put to re-write in depth some parts of WISARD .**

4 User manual

4.1 General presentation

4.1.1 A broad overview

The WISARD reconstruction method is a regularized reconstruction [6, 7], where the solution (reconstructed object) is defined as the one that minimizes a compound criterion (or "metric") with two terms : (1) the likelihood term that measures the fit of the reconstruction to the data and (2) the regularization or penalty term that measures the compatibility of the reconstruction with a given prior knowledge on the true object. The prior knowledge is, on the one hand, the hard constraint of positivity of the reconstruction, and, on the other hand, a somewhat fuzzy knowledge that the object to be reconstructed has some kind of smoothness (or piece-wise smoothness, or global smoothness apart from some spikes, etc). In WISARD , several regularization terms are available to embody this prior knowledge on the solution :

– a smooth solution is obtained by a *quadratic* regularization, which uses the object's PSD (Power Spectral Density) as input. This regularization is described in [8] and implemented in `j_prior_gauss.pro`.
– a piecewise smooth prior is obtained by a *linear-quadratic* (also called L_1-L_2) regularization introduced in [9, 10]. This prior is called *edge-preserving* as it allows sharp edges in the object if the data is compatible with them, contrarily to a quadratic regularization.
– a variant of this regularization has been designed to grant the solution with some smoothness while allowing spikes ; it is a pixel-independent (or white) linear-quadratic regularization called *spike-preserving* in short. Both the edge-preserving and the spike-preserving priors are implemented in `j_prior_l1l2.pro`.
– a so-called *soft support* regularization, proposed by L. Mugnier and E. Thiébaut, is also available starting with version 3.x of WISARD . It is described in [11] and implemented in `j_prior_support.pro`.

The main input parameters for WISARD are the data of course, the Field-Of-View `FOV` for the reconstruction, the minimum number of pixel `NP_MIN` the user wants in this FOV, and keywords specific to the desired regularization. For a quadratic regularization, use keyword `PSD`, and possibly keyword `MEAN_O`. For an edge-preserving regularization, use keywords `DELTA` and `SCALE`. For a spike-preserving regularization, use keywords `DELTA`, `SCALE`, and `WHITE=1`. For a soft support regularization, use keywords `MU_SUPPORT` and either `FWHM` or `MEAN_O`. See 4.3 for the full list of keywords and details on how to set them.
Additionally, some information is displayed while the iterative minimization is in progress, under the following form :
`ITER/NBITER; Convergence=x. Criterion=jtotal = jdata + jprior ,`
where `ITER` is the number of iterations performed so far, `NBITER` is the maximum allowed number of iterations, `x` is the relative variation (normalized difference between the previous iteration and the current one) of the convergence criterion value `jtotal` to be minmized, wihch is itself the sum of `jdata`, the current value of the likelihood term (distance from the model to the data, i.e. data penalty) and `jprior`, the current value from the regularization term (prior penalty).
The on-line documentation on WISARD or any of its sub-routines can be obtained by typing
`doc_library, '<routine_name>'` at the IDL prompt
(for example : `doc_library, 'wisard'`).

4.1.2 WISARD reconstruction main parameters

We present below the main parameters for a reconstruction in a simple case. Note that launching WISARD should be preceded by the preliminary steps detailed in the next section. Assuming this, a typical call to perform a reconstruction with WISARD is the following :

```
reconstruction = WISARD(data,  $
                GUESS = guess, FOV = fov, NP_MIN = np_min, $
                SCALE = scale, DELTA = delta, /WHITE, /DISPLAY)
```

The output, `reconstruction`, is the image reconstructed by WISARD (an IDL 2D matrix). The following inputs are useful or necessary :

- `data` : interferometric data, as described Sect. 5.2.3 ;
- `GUESS` : object guess. Generally, if when dealing with less than 6 telescope interferometer data, one should first perform a parametric reconstruction (a.k.a. model fitting) before using WISARD and use the resulting image as a guess for WISARD . With very rich data (dense uv plane coverage), the object guess is not necessary and is derived from the dirty map.
- `FOV` : desired field of view for the reconstruction. WISARD unit is radians, but in the example batches we use a conversion factor allowing to give `FOV` in milliarcseconds. If a previous parametric reconstruction (model fitting) has been performed, it should provide a reasonable idea for the field of view `FOV`, which should be around 2 to 3 times the size of the object obtained by model fitting.
- `NP_MIN` : the minimum number of pixels in the FOV. You may first run WISARD with this parameter set to 0 (`NP_MIN=0`) ; in that case the minimum number of pixel to fulfill the Shannon-Nyquist criterion will be determined automatically, and will be contained as an output in (`NP_MIN`) after the first reconstruction by WISARD İf the grid seems then too coarse, you can increase the number of pixels. Do so gradually, because it strongly impacts the computation time.
- `scale` and `delta` : regularization parameters. We suggest, by default, to perform the reconstruction, with the following values :
  - `scale = 1D/(NP_min)^2;` (or `scale = 1D;` if the number of pixel is set automatically with `NP_MIN=0`)
  - `delta = 1D;`

  For a finer adjustment of the regularization parameters, see indications in Section 4.3.
- `/DISPLAY` : setting this keyword will make the reconstruction slower, but much easier to follow, thanks to graphs showing the fit to the visibilities along the way.

For a more advanced use of WISARD , please consult Sect. 4.3 for a complete list of parameters.

4.2 Step-by-step use of WISARD

An example of use of WISARD for several regularizations types and levels is the **`wisard_batch.pro`** file, located in the `pro/` directory of the distribution. **Studying this example, running it and modifying its parameters should be a good starting point for building your own reconstruction know-how**.
The data used here are from the Imaging Beauty Contest 2004 organized by Peter Lawson for the IAU [5]. With the latest version of the software distributed here, you will obtain reconstructions notably better than the ones we obtained with WISARD at the time [12], and at least as good as the best one of [5].
Various examples of reconstructions using data sets from other Imaging Beauty Contests (from 2006 to 2012) are given in batch **`examples_IBC.pro`**, also located in **`/pro`** directory.
Below are described the *essential steps* for running WISARD on IDL

Launch IDL or IDLDE (or GDL) : It is launched either from the executable icon or from the shell (`> idl` at the prompt). Instead of the sole command-line prompt interaction, IDL also offers an IDLDE, and interactive graphical environment which you might prefer. To launch it, type `idlde` instead of `idl`. For gdl, use : `> gdl` at the prompt. In the following, `gdl` should exhibit the same behavior as `idl`.

Initialize path and batch : The paths should be updated so that IDL can "see" all the WISARD components, its . So, at the beginning of a new IDL/wisard session, one should :

1. Set to variable `path_to_Wisard` the path to your Wisard directory (where are directories pro, doc, lib, inputdata, ...)
   `path_to_wisard='/home/my_home/my_full_path/wisard'` (replace here by your actual path to wisard)
2. Move to the 'pro' directory of Wisard [3] : `cd, path_to_wisard+'/pro'`
3. Execute the initialization batch by typing '`.r wisard_init`', or using the "Run" button from idlde. Among other things, this adds the paths to the whole Wisard tree, and checks whether codes are actually visible. If a message indicates that they are not, you have to check the path and repeat from step 1).

Copyright license : Read the license, and if this is fine for you, you can go on :

```
dummy = wisard(/copyright)
```

Translate your OIFITS data into /wisard structure data : It is necessary to pre-process you data file, which is in OIFITS format, into a structure suitable for input into WISARD :

```
datafilename = '../inputdata/dataImagingBeautyContest2004.oifits'
data = wisard_oifits2data(datafilename)
```

In practice, `datafilename` might contain a list of several file names, all of which will be read and merged into a single Wisard data structure. Also, this pre-processing step can accept a few options (described further below), mostly for selecting only a part of the spectral channels contained in the data file.

Choose the parameters for the reconstruction : The size parameters to be set for the reconstruction are the Field-Of-View (FOV) and the grid size. The default FOV is the inverse of the minimum spatial frequency present in the data, which is often too small (if the data lacks low frequencies). For a 32×32 reconstructed object and a FOV of 18 milli-arcseconds (for instance), type :

```
NP_min = 32L
onemas = 1d-3*(!DPi/180D)/3600D ; one mas in radian
fov = 18.0*onemas
```

Choose also the convergence threshold used to stop the iterations. By default, one waits until the criterion no longer evolves (the default threshold being given by machine precision and of the order of 10^{-7}). For experimenting with the code, you may choose a larger value, such as 10^{-6} :

```
threshold = 1d-6
```

You may also give an initial guess (if you have performed other reconstructions, or if you want to see how stable the reconstruction is with respect to the initial guess). By default (if the guess is 0 or absent, the so-called *dirty map* is computed by WISARD on the appropriate grid, thresholded to positive values, and used as an initial guess :

```
guess = 0
```

[3] Moving explicitely to that directory at every new IDL start may be avoided if you set the initial IDL path to be `/(your_path)/WISARD/pro`, and write this command in the `/.bashrc` file (if bash is the shell you use), or in a dedicated `/.idlenv` file. E.g. (for bash) : `export IDL_STARTUP= /(your_path)/WISARD/pro`

Choose regularization and launch WISARD reconstruction(s) : Here are a few hints for choosing the mathematical regularization, and how to launch WISARD . For a first reconstruction, you can use a Gaussian or PSD-based regularization. The PSD must be a 2-D map in Fourier space containing the assumed PSD of the object to be reconstructed. For example, for a disk-like object you should take a PSD with a $1/f^3$ dependence on the spatial frequency (if the square of the FT of a disk has an envelope that goes to zero with this dependence). You should threshold this PSD so that it remains finite at 0 and so that its dynamic range remains reasonable (as a large dynamic range for the PSD may hinder the minimization) :

```
distance = double(shift(dist(NP_min), NP_min/2, NP_min/2))
PSD = 1D/((distance^3  > 1D) < 1d6)
```

Run WISARD , displaying both the object being reconstructed (in window 0) and the fit of the reconstructed visibilities to the data (in window 1) :

```
x_psd = WISARD(data,  $
               FOV = fov, NP_MIN = np_min, $
               GUESS = guess, THRESHOLD = threshold, $
               PSD = psd, $
               AUX_OUTPUT = aux_output_psd, /DISPLAY )
```

The obtained reconstruction shows the correct overall shape of the reconstructed object, on a dark and quite noiseless background, but without the central peak of the original object (see[5]).
A global factor multiplying the PSD acts as the inverse of a regularization parameter. In other words, for more regularization (to get a smoother object), you should divide the PSD by, *e.g.*, a factor 10 (or more) *i.e.*, assume a weaker object ; and for less regularization (to get an object with more details), you should multiply the PSD by, *e.g.*, a factor 10 (or more) *i.e.*, assume a stronger object. If you set `PSD = 100D*psd` for instance, you will get an under-regularized solution very similar to that obtained without a regularization : quite noisy, but with the emergence of the central peak present in the original object.
Let's now try the white L1-L2 (spike-preserving) regularization. Set `WHITE=1` and choose `SCALE` of the order of the object's mean value ($\sim$1/NP^2 since the reconstructed object is of unit sum). For `DELTA`$\gg 1$, one obtains a quadratic (L2) regularization and a reconstruction similar to a PSD-based one. For the desired spike-preserving regularization you can set :

```
white = 1B;
scale = 1D/(NP_min)^2; for white L1-L2, scale ~ avg_object_level = 1/NP^2
delta = 1D;
```

Run it :

```
x_l1l2white = WISARD(data,  $
                     FOV = fov, NP_MIN = np_min, $
                     GUESS = 0, THRESHOLD = threshold, $
                     SCALE = scale, DELTA = delta, WHITE = white, $
                     AUX_OUTPUT = aux_output_l1l2white, /DISPLAY)
```

The result is much smoother than an under-regularized solution *and* recovers fine the central peak of the original object.
Figure 1 shows three reconstructions : an under-regularized PSD-based reconstruction (with the above PSD multiplied by 100), the PSD-based reconstruction run above, and the white L1-L2 reconstruction just obtained. See the example file `wisard_batch.pro` for reconstructions using the soft support regularization.

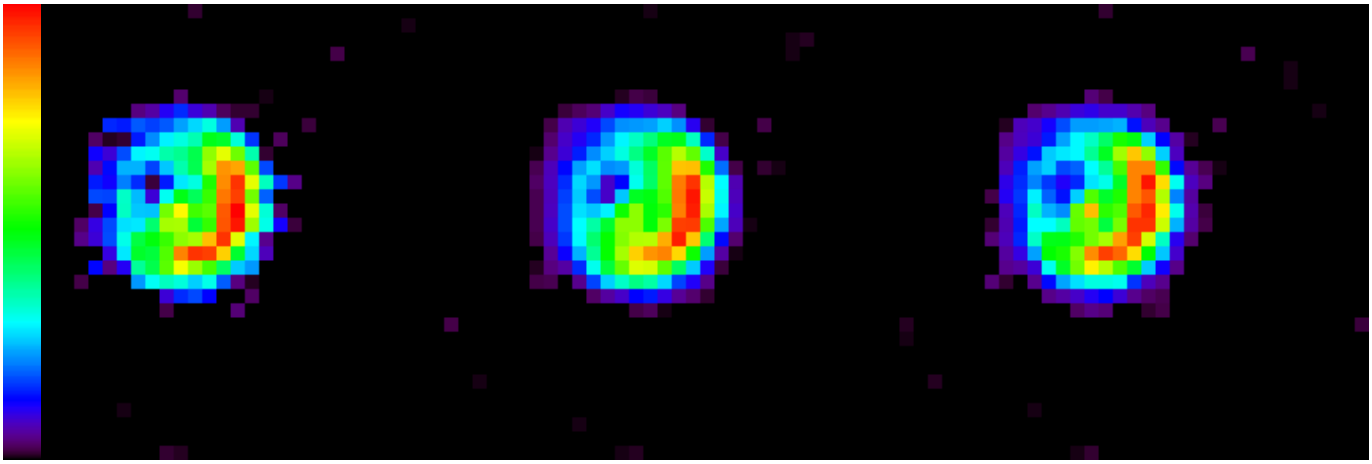

FIG. 1 – Three reconstructions obtained with WISARD for different regularizations. *left* : under-regularized PSD-based (central peak present, but noisy image) ; *center* : correctly regularized PSD-based (smooth image, but no clear central peak) ; *center* : white L1-L2 (smooth image and central peak present). The false-color table is given on the left side.

Figure 2 shows the plots of WISARD displayed during the white L1-L2 reconstruction. These plots aim at showing the quality of the fit of the reconstruction to the data : the red crosses show the modulus of the reconstructed visibilities (FT of the object at the measured frequencies), while the green squares show the "convexified myopic" measured visibilities. This latter quantity represents the original visibility data, corrected following the equations in the Design Report (Sect. 5). They should be roughly equal to the measured visibilities, unless these (and/or the measured closure phases) are associated with extremely large error bars, in which case you may find between discrepancies between the values of the displayed green squares and your visibilities contained in the OIFITS. The blue line shows the difference of the two, normalized by 10 times the standard deviation of the visibilities. In other words, when the blue line is at a level of 0.1 it means that the difference between the reconstructed and the measured visibilities is at one standard deviation (which means the reconstruction fits well the data).

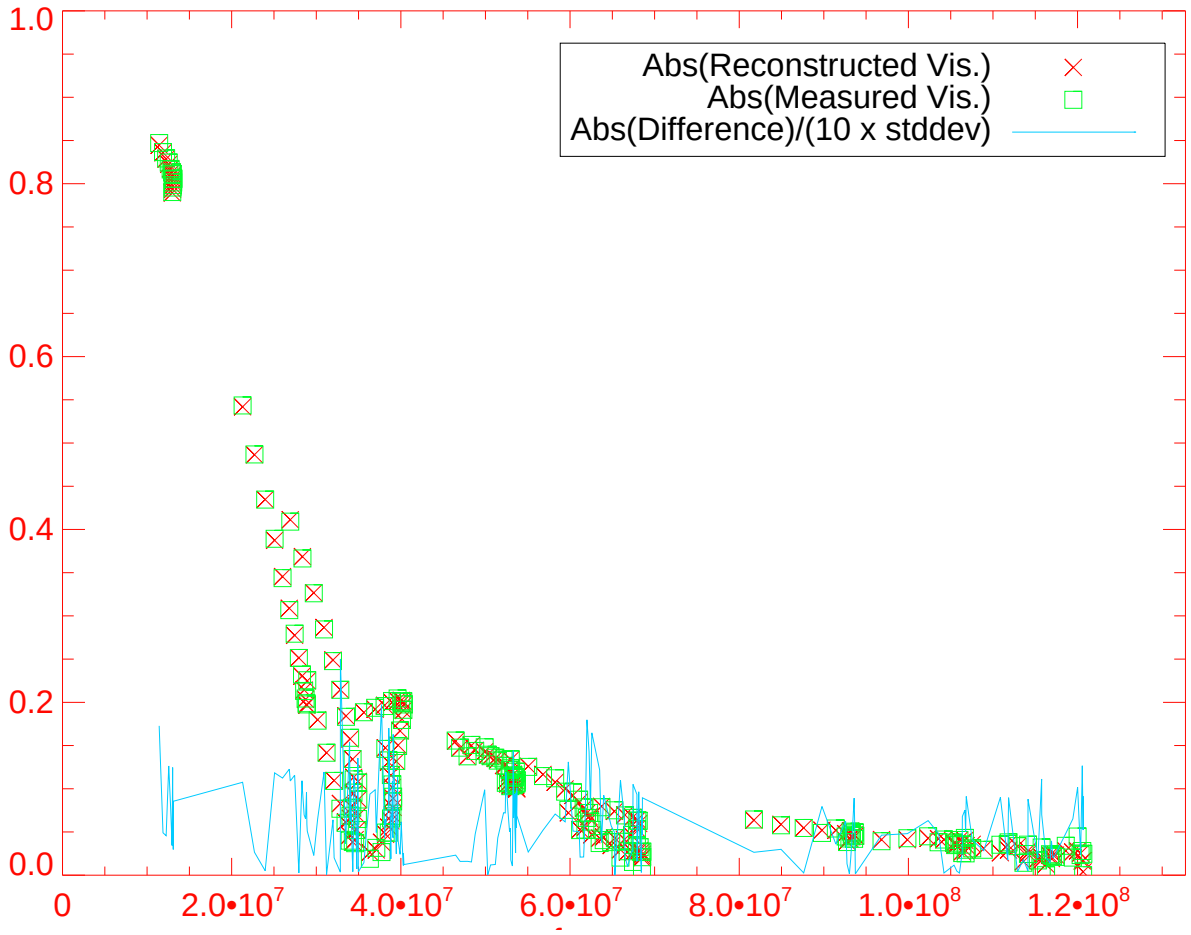

FIG. 2 – Typical plot displayed by WISARD at convergence (see Sect. 4.2).

4.3 List of parameters and accepted keywords

Call to `wisard_oifits2data` : The only positional parameter is `filename`. All the other parameters are passed as keywords. The notation `/KEYWORD` in the following table means `KEYWORD=1`, as is customary with IDL.

`filename`	(input)	Name(s) of OIFITS input data for the reconstruction. `data` might be either a single or an array of string variable(s), depending on the number of files (e.g. ["a.fits","b.fits"]). Each data file should comply with the OIFITS standard, i.e be normalized and contain at least the following extensions : `OI_ARRAY, OI_WAVELENGTH, OI_VIS2, OI_T3.`
`NUMBEROFTELS`$= N$	(optional input)	force the data to be read as a N simultaneous telescopes, in case WISARD does not grasp it correctly.
`/FORCE_DIFFERENT_TIMES`	(optional input)	indicate that the observation sets were done at different times. Useful if time stamps are not correct in the OIFITS files.
`/EXPLODE_TRIPLETS`	(optional input)	do not keep array configuration (all the triplets pertaining to the same observational time) when creating the out structure. This desperate move enables all the data to be fed in WISARD whatever the disparity of the configurations in the files(s) but at the expense of having much less phase information (all the data is "observed" by a 3-telescope interferometer).
`SYNCHRONICITY`$= value$	(optional input)	$value$ (in seconds) is the maxim admissible time difference between observables (e.g. V^2 and corresponding phase closures T3) in order to consider that they have been observed simultaneously. In general, the use of SYNCHRONICITY is not necessary, but for same data sets it can be useful. To be used with caution.
`/SELECT_BEST_DATA`	(optional input)	activate this flag when all the phase closures are present in the data so that only the best (higher signal-to-noise ratio) independent phase closures will be selected from the whole available set.
`VERBOSE=[0 to 4]`	(optional input)	displays different information.
`/HELP`	(optional input)	prints the on-line documentation and exits.

Once the data is read, it is possible (as an option) to select specific spectral channels for the image reconstruction using the routine **`wisard_select_wlen`** :

`data`	(input)	Name(s) of OIFITS input data for the reconstruction as previously described.
`wlen_range`	(input)	The wavelength interval(s). For example, to define one interval wlen_range should be defined as [[1.5,1.6]], and for two one different intervals [[1.5,1.6],[1.75,1.85]] (here the wavelengths are given in microns). If necessary, a list of wavelengths in the input data can be obtained by printing the array data.wlen.

After calling `wisard_select_wlen` a new data subset is created for the required wavelengths. For example :

```
data_subset=wisard_select_wlen(data, [[1.5,1.6],[1.75,1.85]])
```

In this example, the structure data_subset should replace the structure data if one wants to work on the wavelength-selected observations.

Call to `wisard` : The only positional parameter is `data`. All the other parameters are passed as keywords. Same comment on the use of `/KEYWORD` as above.

`data`	(input)	interferometric input data for the reconstruction. `data` is a vector of structures, one structure per time of measurement, described in Sect 5.2.3. The zero frequency should not be present in the data, and the visibility data should be normalized (as they are by convention in OIFITS files).
`FOV`	(input)	Field-Of-View of the reconstructed image, in units consistent with the data. More precisely the unit for FOV must be the inverse of the unit of the arrays of frequencies (`FREQS_U` and `FREQS_V`) of the data. Usually `FREQS_U` and `FREQS_V` are in rd^{-1} so `FOV` should be in rd (radians).
`NP_MIN`	(input)	MINimum width (Number of Points) of the reconstructed image. The Number of Points of the reconstructed object may be greater, depending on `FOV`, `OVERSAMPLING` factor, and frequencies present in the `data`. See routine `WISARD_MAKE_H` for details.
`OVERSAMPLING`	(optional input)	oversampling factor for the reconstructed image. By default, `OVERSAMPLING`=1 and the maximum spatial frequency of the reconstructed object is the maximum frequency of the data. See routine `WISARD_MAKE_H` for details.
`GUESS`	(optional input)	initial guess for the reconstructed image (or dirty map if not present). This guess is processed in the following way : resampled if necessary to the correct size, thresholded to positive values and normalized to a unit sum. This resulting processed guess is available on exit as a field of `AUX_OUTPUT`.
`NBITER`	(optional input)	maximum NumBer of ITERations for the reconstruction, 500 by default. For a better control of the reconstruction, one should rather use `THRESHOLD` below and not lower this value.
`THRESHOLD`	(optional input)	convergence threshold to be used as a stopping criterion for the iterations. By default set to the machine precision in simple precision (approximately 1.19×10^{-07}), although computations are performed in double precision. For tests and a quick-look of the results, set THRESHOLD to $\sim 10^{-6}$.

POSITIVITY	(optional input)	Positivity constraint for the reconstruction. It is set to true (1) by default, which is the commonly used value. Set it explicitly to 0 if for some reason (e.g. tests) you do not want to use the positivity constraint in the reconstruction.
PSD	(optional input)	2-D map of size NP_MIN×NP_MIN containing the PSD for the (quadratic) regularization of the reconstruction. See routine J_PRIOR_GAUSS and the example file for details.
MEAN_O	(optional input)	2-D map of size NP_MIN×NP_MIN containing the mean object to be used for regularization of the reconstruction. MEAN_O is used both for PSD regularization and for white L1-L2 regularization (i.e., when DELTA, SCALE and WHITE are set). It is also used for soft support regularization if FWHM is not given ; in this case it must be a 2D map with strictly positive values.
DELTA	(optional input)	scalar factor for L1-L2 regularization, used to set the threshold between quadratic (L2) and linear (L1) regularization. See routine J_PRIOR_L1L2 and the example file for some more details. See [10] for a complete description of the L1-L2 regularization. The PDF of this paper is on-line at : http ://laurent.mugnier.free.fr/publis/Mugnier-JOSAA-04.pdf.
SCALE	(optional input)	scalar SCALE factor for L1-L2 regularization. Should be of the order of the average object value if WHITE is set, and of the order of the RMS object's gradient value if WHITE is not set. See J_PRIOR_L1L2 and example file for some more details, and the paper of [10] cited above for a complete explanation. An alternative, more empirical way to set the scale is to consider that it also determines the balance between the jdata and jprior terms of the minimization criterion. So if, after an initial reconstruction, these happen to be too largely unbalanced, you may (for instance) decrease/increase SCALE in order to increase/decrease proportionally jprior.

WHITE	(optional input)	flag to switch between edge-preserving regularization (WHITE=0, default) and spike-preserving regularization (WHITE=1). In the latter case the regularization is performed independently on each pixel value, hence the flag name.
MU_SUPPORT	(optional input)	global factor for regularization by "soft support". See J_PRIOR_SUPPORT for more details. See [11] for a complete description of the soft support regularization.
LIBRARY	(optional input)	full path of the OptimPack library (see FMIN_OP for details), if necessary. Under Unix systems the OptimPack library should be found automatically.
AUX_OUTPUT	(optional output)	structure containing various optional auxiliary outputs, for diagnostic purposes. The details of this structure is given in the on-line documentation of the wisard routine.
/DISPLAY	(optional input)	display the reconstructed object and the fit to the visibilities during the execution of WISARD .
/VERSION	(optional input)	prints version number before execution.
/HELP	(optional input)	prints the on-line documentation and exits.
/COPYRIGHT	(optional input)	prints information about copyright and exits.

5 WISARD Design Report

Optical/IR long baseline interferometry (OLBI) with a few 10m-class telescopes can reach the angular resolution that a 100m-class telescope would provide under only under operation conditions close to the diffraction-limit.
The interferograms of current instruments are affected by turbulence, which corrupts completely the recorded object's phase at each baseline. Thus, one is led to measure quantities that are less (or not) affected by turbulence (phase closures) and/or that are corrected from turbulence and instrumental effects (calibrated squared visibilities).
In order to cope with the missing phase information, we introduce phase calibration parameters to be estimated jointly with the observed object and we propose WISARD (for Weak-phase Interferometric Sample Alternating Reconstruction Device) to perform this estimation. This algorithm combines, within a Bayesian framework, an alternating estimation of the object and phase parameters (in the spirit of self-calibration algorithms proposed by radio-astronomers [13]), a recently developped noise model suited to OLBI data [2], and various regularization criteria [8, 10, 11] to deal with the sparsity of the data typical of OLBI.

5.1 Structure of WISARD

WISARD is made of the following major blocks :
– a first block wisard_data2mdata.pro computes (so-called "myopic") complex pseudo-data from the input data, and error bars on these pseudo-data. These complex data have the following properties :
  – the phases measurements and error bars are computed from and compatible with the measured phase closures and their error bars ;
  – the amplitudes measurements and error bars are computed from and compatible with the measured squared visibilities and their error bars.
– a convexification block wisard_mdata2cmdata.pro computes a gaussian approximation of the pseudo visibility data model. This approximation is optimal in the sense of a Kullback Leibler distance (see Appendix C) ;

– a third block `wisard_set_regul.pro` proposes an adapted prior term, to be chosen among a list of regularizations, such as positivity, PSD quadratic regularization, linear-quadratic (also called L_1–L_2) regularization, etc... The algorithmic structure of WISARD makes it easy to add other kinds of prior terms to the code ;
– a self-calibration block performs the minimization. It alternates :
  – a minimization of the criterion w. r. t. the object, which consists essentially in a minimization of a convex criterion under positivity constraint ;
  – an optimization of the aberrations for the current object. It is accelerated by performing in parallel several optimizations of a subset of the aberrations, instead of one global optimization.

The algorithm structure of WISARD is shown in Fig. 5.1.

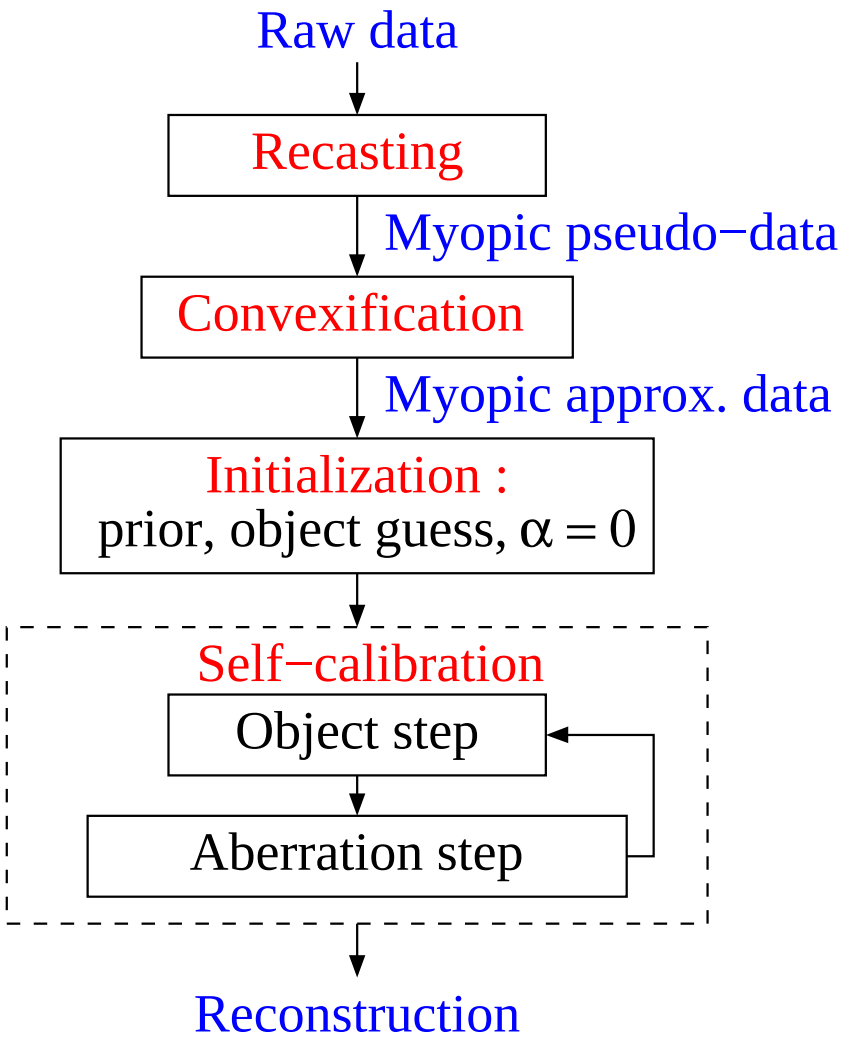

FIG. 3 – WISARD algorithm loop.

5.2 Interferometric observables

Consider a model 2-telescope interferometer, with two identical apertures (T_1, T_2) located at positions $\overrightarrow{OT}_1$ and $\overrightarrow{OT}_2$. We denote $\mathcal{P}$ the plane normal to the pointing direction, and $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$ the projection of $\overrightarrow{OT}_1$ and $\overrightarrow{OT}_2$ onto $\mathcal{P}$. The *baseline* $\boldsymbol{u}_{12}$ is defined by the projection of the displacement $\overrightarrow{T_1T_2}$ onto $\mathcal{P}$:

$$\boldsymbol{u}_{12} \triangleq \boldsymbol{r}_2 - \boldsymbol{r}_1$$

Because of the Earth rotation, the pointing direction changes during an observing night, so the baseline is time dependant too :

$$\boldsymbol{u}_{12}(t) \triangleq \boldsymbol{r}_2(t) - \boldsymbol{r}_1(t) \tag{1}$$

Let us consider now a N_t-telescope interferometer. There are as many baselines as ways to choose 2 telescopes among N_t :

$$N_b = \binom{N_\mathrm{t}}{2} = \frac{N_\mathrm{t}(N_\mathrm{t}-1)}{2} \tag{2}$$

For $N_\mathrm{t} = 4$, the $N_b = 6$ baselines are :

$$\begin{cases} \boldsymbol{u}_{12}(t) = \boldsymbol{r}_2(t) - \boldsymbol{r}_1(t) \\ \boldsymbol{u}_{13}(t) = \boldsymbol{r}_3(t) - \boldsymbol{r}_1(t) \\ \boldsymbol{u}_{14}(t) = \boldsymbol{r}_4(t) - \boldsymbol{r}_1(t) \\ \boldsymbol{u}_{23}(t) = \boldsymbol{r}_3(t) - \boldsymbol{r}_2(t) \\ \boldsymbol{u}_{24}(t) = \boldsymbol{r}_4(t) - \boldsymbol{r}_2(t) \\ \boldsymbol{u}_{34}(t) = \boldsymbol{r}_4(t) - \boldsymbol{r}_3(t) \end{cases}$$

or in a matrix formulation :

$$\boldsymbol{u}(t) = \boldsymbol{B} \cdot \boldsymbol{r}(t), \tag{3}$$

where the $N_b \times N_\mathrm{t}$ operator $\boldsymbol{B}$ is the *baseline operator* (see appendix A).

5.2.1 Ideal interferometric data

We consider a monochromatic source with a wavelength λ, and denote $x(\boldsymbol{\xi})$ the brightness distribution, $\boldsymbol{\xi}$ being angular coordinates. For each baseline, it is possible to access to the following short exposure data :
– a phase $\boldsymbol{\phi}^{\mathrm{data}}(t)$
– a modulus $\boldsymbol{a}^{\mathrm{data}}(t)$
The complex vector $\boldsymbol{y}^{\mathrm{data}}(t) = \boldsymbol{a}^{\mathrm{data}}(t)\mathrm{e}^{\mathrm{i}\boldsymbol{\phi}^{\mathrm{data}}(t)}$ is called the *complex visibility vector*. According to the Van Cittert-Zernike theorem [14], complex visibilities are *ideally* linked to the Fourier Transform (FT) of $x(\boldsymbol{\xi})$ at the 2D spatial frequency

$$\boldsymbol{\nu}(t) \triangleq \frac{\boldsymbol{u}(t)}{\lambda}, \tag{4}$$

through ${y_{ij}}^{\mathrm{data}}(t) = \mathrm{FT}\left[x(\boldsymbol{\xi})\right](\boldsymbol{\nu}_{ij}(t))$, *i.e.*, :

$$\begin{cases} {\phi_{ij}}^{\mathrm{data}}(t) = \phi_{ij}(\boldsymbol{x},t) \triangleq \arg \mathrm{FT}\left[x(\boldsymbol{\xi})\right](\boldsymbol{\nu}_{ij}(t)) \\ {a_{ij}}^{\mathrm{data}}(t) = s_{ij}(\boldsymbol{x},t) \triangleq \left|\mathrm{FT}\left[x(\boldsymbol{\xi})\right](\boldsymbol{\nu}_{ij}(t))\right| \end{cases} \tag{5}$$

or in a matrix formulation :

$$\begin{cases} \boldsymbol{\phi}^{\mathrm{data}}(t) = \boldsymbol{\phi}(\boldsymbol{x},t) \\ \boldsymbol{a}^{\mathrm{data}}(t) = \boldsymbol{a}(\boldsymbol{x},t) \end{cases} \tag{6}$$

In what follows, we will only consider a discretized version $\boldsymbol{x}$ of $x(\boldsymbol{\xi})$. The Fourier Transform then corresponds to a matrix product by a Fourier operator $\boldsymbol{H}$:

$$\boldsymbol{y}(\boldsymbol{x},t) = \boldsymbol{H}(t)\boldsymbol{x} \tag{7}$$

For convenience, we also define :

$$\begin{cases} \boldsymbol{\phi}(\boldsymbol{x},t) \triangleq \arg\left(\boldsymbol{H}(t)\boldsymbol{x}\right) \\ \boldsymbol{a}(\boldsymbol{x},t) \triangleq \left|\boldsymbol{H}(t)\boldsymbol{x}\right| \\ \boldsymbol{s}(\boldsymbol{x},t) \triangleq \left|\boldsymbol{H}(t)\boldsymbol{x}\right|^2 \end{cases} \tag{8}$$

The operators $|.|$ and $\arg$ are point-to-point operators.

5.2.2 Data model

The long exposure observables considered in this report are squared visibilities $\boldsymbol{s}^{\text{data}}(t)$ and closure phases $\boldsymbol{\beta}^{\text{data}}(t)$. These observable are affected by a noise, and we suppose in this report that only second degree statistics are provided, and that the closure phase measurement set and the squared modulus measurement set are two uncorrelated sets of variables.
The noise distribution we can make out of these statistics is then gaussian :

$$\begin{cases} \boldsymbol{s}^{\text{data}}(t) = \boldsymbol{a}^2(\boldsymbol{x},t) + \boldsymbol{s}^{\text{noise}}(t), \quad \boldsymbol{s}^{\text{noise}}(t) \sim \mathcal{N}\left(0, \boldsymbol{R}_{\boldsymbol{s}(t)}\right) \\ \boldsymbol{\beta}^{\text{data}}(t) = \boldsymbol{C}\boldsymbol{\phi}(\boldsymbol{x},t) + \boldsymbol{\beta}^{\text{noise}}(t), \quad \boldsymbol{\beta}^{\text{noise}}(t) \sim \mathcal{N}\left(0, \boldsymbol{R}_{\boldsymbol{\beta}(t)}\right) \end{cases} \tag{9}$$

$\boldsymbol{R}_{\boldsymbol{s}(t)}$ is the covariance matrix of the distribution. The algebraic structure of the *closure* operator $\boldsymbol{C}$ is described in appendix A page 24.
The covariance matrices $\boldsymbol{R}_{\boldsymbol{s}(t)}$ and $\boldsymbol{R}_{\boldsymbol{\beta}(t)}$ are implicitly supposed diagonal.

5.2.3 Data format used in WISARD

The WISARD data format is generated from the OIFITS data input (see requirements on OIFITS format in Sect. **??**). It is a vector of structures, with one structure instance for each time of measurement. Each of them contains the following fields :
– `VIS2` : the squared visibilities for each baseline and each spectral channel,
– `VIS2ERR` : the standard deviation on the squared visibilities
– `CLOT` : the closure phases for each triplet of telescopes involving telescope 1, and each spectral channel
– `CLOTERR` : the standard deviation on the closure phases
– `FREQS_U` : first coordinate of the spatial frequencies involving telescope 1
– `FREQS_V` : second coordinate of the spatial frequencies involving telescope 1

For example, at each moment t, for a 4-telescope array, the fields contain :

field	#	notation
`VIS2` :	6	$s^{\text{data}}_{12}(t), s^{\text{data}}_{13}(t), s^{\text{data}}_{14}(t), s^{\text{data}}_{23}(t), s^{\text{data}}_{24}(t), s^{\text{data}}_{34}(t)$
`VIS2ERR` :	6	$\sigma_{s_{12}(t)}, \sigma_{s_{13}(t)}, \sigma_{s_{14}(t)}, \sigma_{s_{23}(t)}, \sigma_{s_{24}(t)}, \sigma_{s_{34}(t)}$
`CLOT` :	3	$\beta^{\text{data}}_{123}(t), \beta^{\text{data}}_{124}(t), \beta^{\text{data}}_{134}(t)$
`CLOTERR` :	3	$\sigma_{\beta_{123}(t)}, \sigma_{\beta_{124}(t)}, \sigma_{\beta_{134}(t)}$
`FREQS_U` :	3	$u_{12}(t), u_{13}(t), u_{14}(t)$
`FREQS_V` :	3	$v_{12}(t), v_{13}(t), v_{14}(t)$

5.3 From input data to a myopic model

This section describes the code `wisard_data2mdata.pro`.
We want to recast the data model in a myopic one, with visibility phase and modulus pseudo-data. The corresponding format is described below. The myopic data format is a vector of structures, one structure for each time of measurement, with the following fields :
– `VISAMP` : the visibility moduli for each baseline
– `VISAMPERR` : the standard deviation on the visibility moduli

– `VISPHI` : the visibility phases for each baseline
– `VISPHIERR` : the standard deviation on the visibility phases
– `FREQS_U` : first coordinate of every spatial frequency
– `FREQS_V` : second coordinate of every spatial frequency

For example, at each moment t, for a 4-telescope array, the fields contain :

field	#	notation
`VISAMP` :	6	$a_{12}(t)^{\text{data}}, a_{13}(t)^{\text{data}}, a_{14}(t)^{\text{data}}, a_{23}(t)^{\text{data}}, a_{24}(t)^{\text{data}}, a_{34}(t)^{\text{data}}$
`VISAMPERR` :	6	$\sigma_{a_{12}(t)}, \sigma_{a_{13}(t)}, \sigma_{a_{14}(t)}, \sigma_{a_{23}(t)}, \sigma_{a_{24}(t)}, \sigma_{a_{34}(t)}$
`VISPHI` :	6	$\phi_{12}(t)^{\text{data}}, \phi_{13}(t)^{\text{data}}, \phi_{14}(t)^{\text{data}}, \phi_{23}(t)^{\text{data}}, \phi_{24}(t)^{\text{data}}, \phi_{34}(t)^{\text{data}}$
`VISPHIERR` :	6	$\sigma_{\phi_{12}(t)}, \sigma_{\phi_{13}(t)}, \sigma_{\phi_{14}(t)}, \sigma_{\phi_{23}(t)}, \sigma_{\phi_{24}(t)}, \sigma_{\phi_{34}(t)}$
`FREQS_U` :	6	$u_{12}(t), u_{13}(t), u_{14}(t), u_{23}(t), u_{24}(t), u_{34}(t)$
`FREQS_V` :	6	$v_{12}(t), v_{13}(t), v_{14}(t), v_{23}(t), v_{24}(t), v_{34}(t)$

5.3.1 Visibility modulus pseudo data

Each $\left\{a_{ij}^{\text{data}}(t), \sigma_{a_{ij}(t)}\right\}$ is computed from input data $\left\{s_{ij}^{\text{data}}(t), \sigma_{s_{ij}(t)}\right\}$ acording to :

– in every case, $a_{ij}^{\text{data}}(t) = \sqrt{|s_{ij}^{\text{data}}(t)|}$;
– if $s_{ij}^{\text{data}}(t) > \sigma_{s_{ij}(t)}$ then $\sigma_{a_{ij}(t)} = \frac{\sigma_{s_{ij}(t)}}{2\sqrt{|s_{ij}^{\text{data}}(t)|}}$;
– else $\sigma_{a_{ij}(t)} = \frac{\sqrt{\sigma_{s_{ij}(t)}}}{2}$.

See appendix B for justification of these formulae.

5.3.2 Visibility phase pseudo data

For each instant t, the vector $\boldsymbol{\phi}^{\text{data}}(t)$ - containing the $\phi_{ij}^{\text{data}}(t)$ - and the vector $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(t)$ - containing the $\sigma_{\phi_{ij}(t)}$ - is computed from the vector $\boldsymbol{\beta}^{\text{data}}(t)$ - containing the input data $\beta_{1ij}^{\text{data}}(t)$ - and the vector $\boldsymbol{\sigma}_{\beta}(t)$ -containing the input data $\sigma_{\beta_{1ij}(t)}$ - acording to :

– $\boldsymbol{\phi}^{\text{data}}(t) = \boldsymbol{C}^{\dagger}\boldsymbol{\beta}^{\text{data}}(t)$;
– $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(t) = 3 \cdot \boldsymbol{C}^{\dagger}\text{Diag}\left\{\boldsymbol{\sigma}_{\beta}(t)\right\}\boldsymbol{C}^{\dagger,\text{T}} \otimes \mathbf{Id}$.

where $\otimes$ denotes a term-to-term product, and $\boldsymbol{C}^{\dagger}$ is defined in appendix A. Let R be a diagonal matrix, the vector of the diagonal components being r. For any matrix M with compatible dimensions, we have :

$$\begin{aligned}\left\{MRM^{\text{T}} \otimes \mathbf{Id}\right\}_{ij} &= \delta_{ij}\sum_k\sum_l R_{ik}\Delta_{kl}M_{jl} \\ &= \delta_{ij}\sum_k\sum_l R_{ik}\Delta_{kl}M_{il} \\ &= \delta_{ij}\sum_k M_{ik}^2 r_k\end{aligned}$$

So we can write :

$$\boldsymbol{\sigma}_{\boldsymbol{\phi}}(t) = 3\left(\boldsymbol{C}^{\dagger} \otimes \boldsymbol{C}^{\dagger}\right)\boldsymbol{\sigma}_{\beta}(t)$$

This is how the computation of $\boldsymbol{\sigma}_{\boldsymbol{\phi}}(t)$ is coded in `wisard_data2mdata.pro`.

5.4 From a myopic model to a myopic convexified model

This section describes the code `wisard_mdata2cmdata.pro`.
The convexified myopic data format is a vector of structures, one structure for each time of measurement, with the following fields :

- `VIS` : the complex visibilities for each baseline ;
- `W_RAD` : radial weight on the visibility moduli, to be used in criterion minimized (see sect. 5.5) ;
- `W_TAN` : tangential weight on the visibility moduli, to be used in criterion minimized (see sect. 5.5) ;
- `FREQS_U` : first coordinate of each spatial frequency ;
- `FREQS_V` : second coordinate of each spatial frequency.

For example, at each moment t, for a 4-telescope array, the fields contain :

field	#	notation
`VIS` :	6	$y_{12}^{\text{data}}(t), y_{13}^{\text{data}}(t), y_{14}^{\text{data}}(t), y_{23}^{\text{data}}(t), y_{24}^{\text{data}}(t), y_{34}^{\text{data}}(t)$
`W_RAD` :	6	$w_{\text{rad},12(t)}, w_{\text{rad},13(t)}, w_{\text{rad},14(t)}, w_{\text{rad},23(t)}, w_{\text{rad},24(t)}, w_{\text{rad},34(t)}$
`W_TAN` :	6	$w_{\text{tan},12(t)}, w_{\text{tan},13(t)}, w_{\text{tan},14(t)}, w_{\text{tan},23(t)}, w_{\text{tan},24(t)}, w_{\text{tan},34(t)}$
`FREQS_U` :	6	$u_{12}(t), u_{13}(t), u_{14}(t), u_{23}(t), u_{24}(t), u_{34}(t)$
`FREQS_V` :	6	$v_{12}(t), v_{13}(t), v_{14}(t), v_{23}(t), v_{24}(t), v_{34}(t)$

Each $\left\{y_{ij}^{\text{data}}(t), w_{\text{rad},ij(t)}, w_{\text{tan},ij(t)}\right\}$ is computed from myopic data $\left\{\phi_{ij}^{\text{data}}(t), \sigma_{\phi_{ij}(t)}, a_{ij}^{\text{data}}(t), \sigma_{a_{ij}(t)}\right\}$ acording to :

$$
\begin{aligned}
y_{ij}^{\text{data}}(t) &\triangleq \left[a_{ij}^{\text{data}}(t) - \bar{y}_{ij}^{\text{data}}(t)\right] \exp \mathrm{i}\phi_{ij}^{\text{data}}(t) \qquad (10)\\
\bar{y}_{ij}^{\text{data}}(t) &= a_{ij}^{\text{data}}(t)\left[\mathrm{e}^{-\frac{\sigma_{\phi_{ij}(t)}^2}{2}} - 1\right]\\
\Rightarrow y_{ij}^{\text{data}}(t) &= a_{ij}^{\text{data}}(t) \exp \phi_{ij}^{\text{data}}(t)\left[2 - \mathrm{e}^{-\frac{\sigma_{\phi_{ij}(t)}^2}{2}}\right]\\
w_{\text{rad},ij(t)} &= \left[\sigma_{\text{rad},ij(t)}^2\right]^{-1}\\
&= \left[\frac{1 + \mathrm{e}^{-2\sigma_{\phi_{ij}(t)}^2}}{2} \cdot \sigma_{a_{ij}(t)}^2 + \frac{\left(1 - \mathrm{e}^{-\sigma_{\phi_{ij}(t)}^2}\right)^2}{2} \cdot {a_{ij}^{\text{data}}}^2(t)\right]^{-1}\\
w_{\text{tan},ij(t)} &= \left[\sigma_{\text{tan},ij(t)}^2\right]^{-1}\\
&= \left[\frac{1 - \mathrm{e}^{-2\sigma_{\phi_{ij}(t)}^2}}{2} \cdot \sigma_{a_{ij}(t)}^2 + \frac{1 - \mathrm{e}^{-2\sigma_{\phi_{ij}(t)}^2}}{2} \cdot a_{ij}^{\text{data}}(t)^2\right]^{-1}
\end{aligned}
$$

These equations are explained in appendix C.

5.5 The criterion minimized

We define :

$$\begin{aligned} z_{\mathrm{rad},ij}(t) &\triangleq \Re\mathrm{e}\left\{z\mathrm{e}^{-\mathrm{i}\phi_{ij}^{\mathrm{data}}(t)}\right\},\ \forall z\in\mathbb{C} \text{ (Cf. eq. 10)} \\ z_{\mathrm{tan},ij}(t) &\triangleq \Im\mathrm{m}\left\{z\mathrm{e}^{-\mathrm{i}\phi_{ij}^{\mathrm{data}}(t)}\right\},\ \forall z\in\mathbb{C} \text{ (Cf. eq. 10)} \\ \boldsymbol{y}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) &\triangleq \boldsymbol{y}(t)^{\mathrm{data}} - \boldsymbol{y}(\boldsymbol{x},t)\mathrm{e}^{\mathrm{i}\bar{\boldsymbol{B}}\boldsymbol{\alpha}(t)} \end{aligned} \tag{11}$$

and propose to use the following data-likelihood criterion :

$$\begin{aligned} \mathcal{J}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}) &\triangleq \sum_t \mathcal{J}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) \triangleq \sum_t\sum_{ij} \mathcal{J}_{ij}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) \\ \mathcal{J}_{ij}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) &\triangleq \frac{1}{2}w_{\mathrm{rad},ij(t)}\left[\boldsymbol{y}_{\mathrm{rad},ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right]^2 + \frac{1}{2}w_{\mathrm{tan},ij(t)}\left[\boldsymbol{y}_{\mathrm{tan},ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right]^2 \end{aligned} \tag{12}$$

5.5.1 Expression for the aberration step

From Eq. 11 , we have :

$$\boldsymbol{y}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) = \boldsymbol{y}(t)^{\mathrm{data}} - \boldsymbol{y}(\boldsymbol{x},t)\mathrm{e}^{\mathrm{i}\bar{\boldsymbol{B}}\boldsymbol{\alpha}(t)}$$

Before starting the aberration step, we compute all the elements useful for the minimization independent of $\boldsymbol{\alpha}$, i.e. $\boldsymbol{y}(\boldsymbol{x},t)$, $\boldsymbol{a}(\boldsymbol{x},t)$ and $\boldsymbol{\phi}(\boldsymbol{x},t)$. To compute the gradient in $\boldsymbol{\alpha}$ we also compute two other quantities, called wx1 and wx2 in the codes. All these quantities are arranged in the structure `weights_x`, which does not depend on the $\boldsymbol{\alpha}$.

5.5.2 Expression for the object step

From Eqs. 8 and 11, we gather :

$$\begin{aligned} \boldsymbol{y}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) &= \boldsymbol{y}(t)^{\mathrm{data}} - \boldsymbol{H}(t)\boldsymbol{x}\mathrm{Diag}\left\{\mathrm{e}^{\mathrm{i}\bar{\boldsymbol{B}}\boldsymbol{\alpha}(t)}\right\} \\ &= \boldsymbol{y}(t)^{\mathrm{data}} - \underbrace{\mathrm{Diag}\left\{\mathrm{e}^{\mathrm{i}\bar{\boldsymbol{B}}\boldsymbol{\alpha}(t)}\right\}\boldsymbol{H}(t)}_{\boldsymbol{P}\boldsymbol{H}(t)}\boldsymbol{x} \end{aligned}$$

Moreover, from Eqs. 11 and 12, we have :

$$\begin{aligned} \mathcal{J}_{ij}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) &\triangleq \frac{1}{2}w_{\mathrm{rad},ij(t)}\left[\boldsymbol{y}_{\mathrm{rad},ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right]^2 + \frac{1}{2}w_{\mathrm{tan},ij(t)}\left[\boldsymbol{y}_{\mathrm{tan},ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right]^2 \\ &= \frac{1}{2}w_{\mathrm{rad},ij(t)}\Re\mathrm{e}^2\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\mathrm{e}^{-\mathrm{i}\phi_{ij}^{\mathrm{data}}(t)}\right\} + \frac{1}{2}w_{\mathrm{tan},ij(t)}\Im\mathrm{m}^2\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\mathrm{e}^{-\mathrm{i}\phi_{ij}^{\mathrm{data}}(t)}\right\} \end{aligned}$$

We gather :

$$\begin{aligned} \mathcal{J}_{ij}^{\mathrm{data}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t) = &\frac{w_{11,ij(t)}}{2}\cdot\Re\mathrm{e}^2\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right\} \\ &+ w_{12,ij(t)}\Im\mathrm{m}\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right\}\Re\mathrm{e}\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right\} \\ &+ \frac{w_{22,ij(t)}}{2}\cdot\Im\mathrm{m}^2\left\{\boldsymbol{y}_{ij}^{\mathrm{res}}(\boldsymbol{x},\boldsymbol{\alpha}(t),t)\right\} \end{aligned} \tag{13}$$

with (for clarity, we omit here the ij and t indexes)

$$\begin{aligned} w_{11} &= w_{\mathrm{rad}}\cos^2\phi^{\mathrm{data}} + w_{\mathrm{tan}}\sin^2\phi^{\mathrm{data}} \\ w_{12} &= (w_{\mathrm{rad}} - w_{\mathrm{tan}})\sin\phi_{ij}^{\mathrm{data}}(t)\cos\phi^{\mathrm{data}} \\ w_{22} &= w_{\mathrm{rad}}\sin^2\phi^{\mathrm{data}} + w_{\mathrm{tan}}\cos^2\phi^{\mathrm{data}} \end{aligned} \tag{14}$$

It is from this expression that the criterion and its gradient w. r. t. the object are computed in WISARD .

A The closure and baseline operators

Let N_t be the number of telescopes of the interferometric array. We have the following definitions :

$$\boldsymbol{B}_2 \triangleq \left[\begin{array}{cc} -1 & 1 \end{array}\right] \tag{15}$$

$$\boldsymbol{B}_{N_\mathrm{t}} \triangleq \left[\begin{array}{c|c} -\mathbf{1}_{n-1} & \mathbf{Id} \\ \hline \boldsymbol{O} & \boldsymbol{B}_{n-1} \end{array}\right] \tag{16}$$

$$\bar{\boldsymbol{B}}_{N_\mathrm{t}} \triangleq \left[\begin{array}{c} \mathbf{Id} \\ \hline \boldsymbol{B}_{n-1} \end{array}\right] \tag{17}$$

$$\boldsymbol{C}_{N_\mathrm{t}} \triangleq \left[\begin{array}{c|c} -\boldsymbol{B}_{n-1} & \mathbf{Id} \end{array}\right] \tag{18}$$

$$\boldsymbol{C}^\dagger \triangleq \boldsymbol{C}^\mathrm{T}\left[\boldsymbol{C}\boldsymbol{C}^\mathrm{T}\right]^{-1} \tag{19}$$

for $N_\mathrm{t} \geq 3$.
It is easy to see that $\boldsymbol{C}\boldsymbol{B} = \mathbf{0}$.

B Square-root of a gaussian distribution

Let us suppose we measure the squared value s of a positive value a, with an additive noise system :

$$s^\mathrm{data} = a^2 + s^\mathrm{noise},$$

s^noise being 0 mean gaussian with the variance σ_s^2. Let $\hat{a}$ be the estimator of a from s^data defined by $\hat{a} = \begin{cases} \sqrt{s^\mathrm{data}}, & \text{if } s^\mathrm{data} > 0 \\ 0 \text{ else} \end{cases}$.
Although $\hat{a}$ is not gaussian vector, we will approximate its distribution by a gaussian one. In many cases, this approximation is valid, as shown in fig. 4.
However, if σ_s is greater than a few a^2, this estimator is biased. We have studied the behavior of the mean $< \hat{a} >$ and standard deviation $\sigma_{\hat{a}}$ of this estimator in function of σ_s/a^2 (see figs. 5 and 6).
We then propose the data model for a

$$a^\mathrm{data} = a + a^\mathrm{noise}$$

with :

$$a^\mathrm{data} = \begin{cases} 0 & \text{if } s^\mathrm{data} \leq 0 \\ \sqrt{\sigma_s/6}, & \text{if } 0 \leq s^\mathrm{data} \leq \sigma_s/6 \\ \sqrt{s^\mathrm{data}}, & \text{if } s^\mathrm{data} \geq \sigma_s/6 \end{cases}$$

$$\sigma_a = \begin{cases} \sqrt{\sigma_s}/2 & \text{if } s^\mathrm{data} \leq \sigma_s \\ \frac{\sigma_s}{2\sqrt{s^\mathrm{data}}}, & \text{if } s^\mathrm{data} \geq \sigma_s \end{cases}$$

We also decide to discard the data such that $s^\mathrm{data} + \sigma_s \leq 0$.

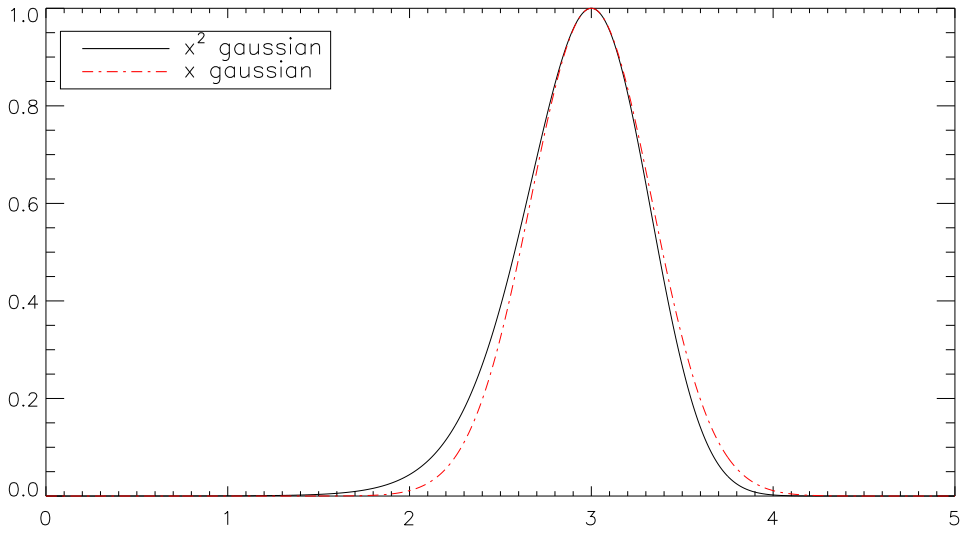

FIG. 4 – Comparison between a gaussian distribution of x and a gaussian distribution of x^2

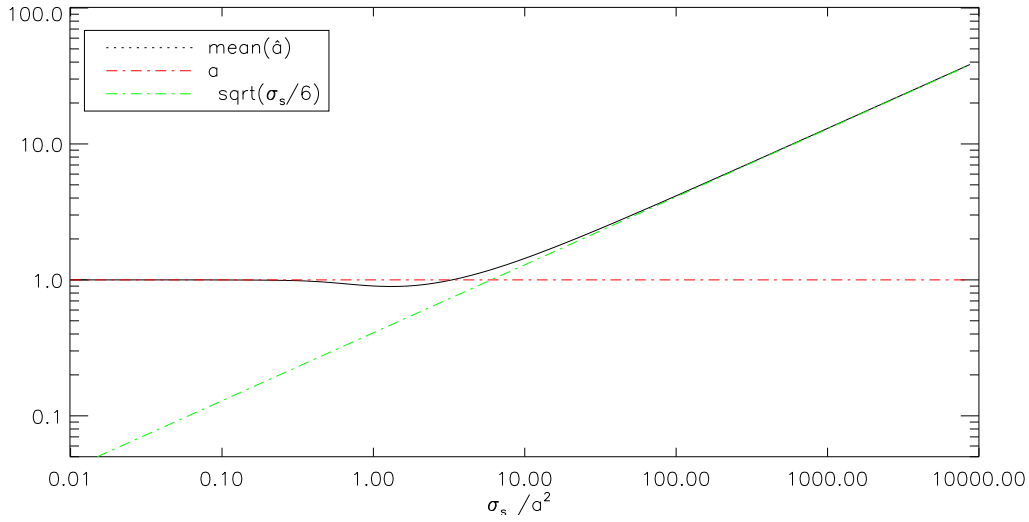

FIG. 5 – Mean of the estimator $\hat{a}$ in function of σ_s/a^2

C Cartesian gaussian approximation to a polar gaussian distribution

D General expression

With consider a polar distribution of a gaussian vector $\boldsymbol{y}$ of modulus $\boldsymbol{a}$ and phase $\boldsymbol{\phi}$:

$$\boldsymbol{\phi}^{\text{data}} = \bar{\boldsymbol{\phi}} + \boldsymbol{\phi}^{\text{noise}} \tag{20}$$

$$\boldsymbol{a}^{\text{data}} = \bar{\boldsymbol{a}} + \boldsymbol{a}^{\text{noise}} \tag{21}$$

where $\boldsymbol{\phi}^{\text{noise}}$ and $\boldsymbol{a}^{\text{noise}}$ are 0 mean real gaussian vectors, of covariance matrices $\boldsymbol{R_a}$ and $\boldsymbol{R_\phi}$ (the vectors $\boldsymbol{\phi}^{\text{noise}}$ and $\boldsymbol{a}^{\text{noise}}$ are supposed uncorrelated).

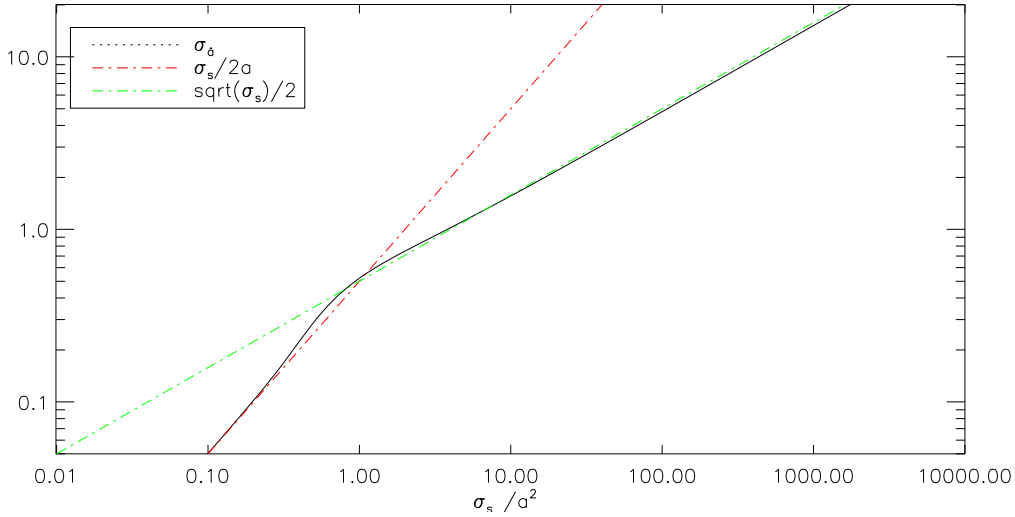

FIG. 6 – Standard deviation of the estimator $\hat{a}$ in function of σ_s/a^2

With the definitions

$$\begin{cases} \bar{\boldsymbol{y}} \triangleq \bar{\boldsymbol{a}} \exp \mathrm{i}\bar{\boldsymbol{\phi}} \\ \boldsymbol{y}^{\mathrm{noise}} \triangleq \boldsymbol{y}^{\mathrm{data}} - \bar{\boldsymbol{y}} \\ \boldsymbol{y}^{\mathrm{n}}_{\mathrm{rad}} \triangleq \Re\mathrm{e}\left\{\boldsymbol{y}^{\mathrm{noise}} \mathrm{e}^{-\mathrm{i}\bar{\boldsymbol{\phi}}}\right\} \\ \boldsymbol{y}^{\mathrm{n}}_{\mathrm{tan}} \triangleq \Im\mathrm{m}\left\{\boldsymbol{y}^{\mathrm{noise}} \mathrm{e}^{-\mathrm{i}\bar{\boldsymbol{\phi}}}\right\} \\ \bar{\bar{\boldsymbol{y}}}^{\mathrm{noise}} \triangleq \begin{bmatrix} \boldsymbol{y}^{\mathrm{n}}_{\mathrm{rad}} \\ \boldsymbol{y}^{\mathrm{n}}_{\mathrm{tan}} \end{bmatrix} \end{cases} \tag{22}$$

we gather :

$$\begin{cases} \boldsymbol{y}^{\mathrm{n}}_{\mathrm{rad}} = \left[\bar{\boldsymbol{a}} + \boldsymbol{a}^{\mathrm{noise}}\right] \cos \boldsymbol{\phi}^{\mathrm{noise}} - \bar{\boldsymbol{a}} \\ \boldsymbol{y}^{\mathrm{n}}_{\mathrm{tan}} = \left[\bar{\boldsymbol{a}} + \boldsymbol{a}^{\mathrm{noise}}\right] \sin \boldsymbol{\phi}^{\mathrm{noise}} \end{cases} \tag{23}$$

A complex vector is gaussian if and only if each of its components is gaussian. A complex is gaussian if and only if, in any cartesian basis, its two components are gaussian. So $\boldsymbol{y}$ is gaussian if and only if $\bar{\bar{\boldsymbol{y}}}^{\mathrm{noise}}$ is gaussian, which is not the case [2]. In what follows, we show how to optimally approximate the distribution of $\bar{\bar{\boldsymbol{y}}}^{\mathrm{noise}}$ by a gaussian distribution.

D.1 Gaussian Approximation

We caracterize our Cartesian additive gaussian approximation, *i.e.*, its mean $\left\langle \bar{\bar{\boldsymbol{y}}}^{\text{noise}} \right\rangle$ and covariance $\boldsymbol{R}_{\bar{\bar{\boldsymbol{y}}}^{\text{noise}}}$, by minimizing the Kullback-Leibler distance between the two noise distributions, which gives [2] :

$$\begin{cases} \left\langle \bar{\bar{\boldsymbol{y}}}^{\text{noise}} \right\rangle = E\left\{ \begin{bmatrix} \boldsymbol{y}^{\text{n}}_{\text{rad}} \\ \boldsymbol{y}^{\text{n}}_{\text{tan}} \end{bmatrix} \right\} = \begin{bmatrix} \bar{\boldsymbol{y}}^{\text{n}}_{\text{rad}} \\ \bar{\boldsymbol{y}}^{\text{n}}_{\text{tan}} \end{bmatrix} \\ \boldsymbol{R}_{\bar{\bar{\boldsymbol{y}}}^{\text{noise}}} = E\left\{ \begin{bmatrix} \bar{\boldsymbol{y}}^{\text{n}}_{\text{rad}} - \boldsymbol{y}^{\text{n}}_{\text{rad}} \\ \bar{\boldsymbol{y}}^{\text{n}}_{\text{tan}} - \boldsymbol{y}^{\text{n}}_{\text{tan}} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{y}}^{\text{n}}_{\text{rad}} - \boldsymbol{y}^{\text{n}}_{\text{rad}} \\ \bar{\boldsymbol{y}}^{\text{n}}_{\text{tan}} - \boldsymbol{y}^{\text{n}}_{\text{tan}} \end{bmatrix}^{\text{T}} \right\} \end{cases} \tag{24}$$

and we define

$$\boldsymbol{R}_{\bar{\bar{\boldsymbol{y}}}^{\text{noise}}} \triangleq \begin{bmatrix} \boldsymbol{R}_{\text{rad,rad}} & \boldsymbol{R}_{\text{rad,tan}} \\ \boldsymbol{R}^{\text{T}}_{\text{rad,tan}} & \boldsymbol{R}_{\text{tan,tan}} \end{bmatrix}$$

For a 0 mean gaussian vector $\boldsymbol{\phi}^{\text{noise}}$ of covariance matrix $\boldsymbol{R}_{\boldsymbol{\phi}}$,

$$\begin{aligned} E\left\{\sin \phi_i^{\text{noise}}\right\} &= 0 \\ E\left\{\cos \phi_i^{\text{noise}}\right\} &= \exp -\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}}}{2} \\ E\left\{\sin \phi_i^{\text{noise}} \sin \phi_j^{\text{noise}}\right\} &= \sinh \boldsymbol{R}_{\boldsymbol{\phi}_{ij}} \cdot \exp -\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}} + \boldsymbol{R}_{\boldsymbol{\phi}_{jj}}}{2} \\ E\left\{\cos \phi_i^{\text{noise}} \cos \phi_j^{\text{noise}}\right\} &= \cosh \boldsymbol{R}_{\boldsymbol{\phi}_{ij}} \cdot \exp -\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}} + \boldsymbol{R}_{\boldsymbol{\phi}_{jj}}}{2} \\ E\left\{\cos \phi_i^{\text{noise}} \sin \phi_j^{\text{noise}}\right\} &= 0 \end{aligned} \tag{25}$$

By combining Eqs. 24, 22, 23 and 25, we obtain :

$$\begin{aligned} E\left\{\boldsymbol{y}^{\text{n}}_{\text{rad}\,i}\right\} &= \bar{a}_i \left[\text{e}^{-\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}}}{2}} - 1 \right] \\ E\left\{\boldsymbol{y}^{\text{n}}_{\text{tan}\,i}\right\} &= 0 \\ \left[\boldsymbol{R}_{\text{rad,rad}}\right]_{ij} &= \left[\bar{a}_i \bar{a}_j \left(\cosh \boldsymbol{R}_{\boldsymbol{\phi}_{ij}} - 1 \right) + \boldsymbol{R}_{a_{ij}} \cosh \boldsymbol{R}_{\boldsymbol{\phi}_{ij}} \right] \cdot \text{e}^{-\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}} + \boldsymbol{R}_{\boldsymbol{\phi}_{jj}}}{2}} \\ \left[\boldsymbol{R}_{\text{rad,tan}}\right]_{ij} &= 0 \\ \left[\boldsymbol{R}_{\text{tan,tan}}\right]_{ij} &= \left(\bar{a}_i \bar{a}_j + \boldsymbol{R}_{a_{ij}} \right) \sinh \boldsymbol{R}_{\boldsymbol{\phi}_{ij}} \cdot \text{e}^{-\frac{\boldsymbol{R}_{\boldsymbol{\phi}_{ii}} + \boldsymbol{R}_{\boldsymbol{\phi}_{jj}}}{2}} \end{aligned} \tag{26}$$

D.2 The scalar case

Now, we make the additional assumption that both $\boldsymbol{\phi}^{\text{noise}}$ and $\boldsymbol{a}^{\text{noise}}$ are uncorrelated, i.e.

$$\begin{cases} \boldsymbol{R}_{\boldsymbol{a}} = \text{Diag}\left\{\sigma^2_{a,i}\right\} \\ \boldsymbol{R}_{\boldsymbol{\phi}} = \text{Diag}\left\{\sigma^2_{\phi,i}\right\} \end{cases}$$

We obtain :

$$\begin{cases} \boldsymbol{R}_{\mathrm{rad,rad}} = \mathrm{Diag}\left\{\sigma^2_{rad,i}\right\} \\ \boldsymbol{R}_{\mathrm{tan,tan}} = \mathrm{Diag}\left\{\sigma^2_{tan,i}\right\} \\ \boldsymbol{R}_{\mathrm{rad,tan}} = 0 \end{cases}$$

with

$$\begin{aligned} \sigma^2_{rad,i} &= \frac{\bar{a}_i^2}{2}\left(1-\mathrm{e}^{-\sigma^2_{\phi,i}}\right)^2 + \frac{\sigma^2_{a,i}}{2}\left(1+\mathrm{e}^{-2\sigma^2_{\phi,i}}\right) \\ \sigma^2_{tan,i} &= \frac{\bar{a}_i^2}{2}\left(1-\mathrm{e}^{-2\sigma^2_{\phi,i}}\right) + \frac{\sigma^2_{a,i}}{2}\left(1-\mathrm{e}^{-2\sigma^2_{\phi,i}}\right) \end{aligned} \tag{27}$$

In this case, we can plot for one complex visibility the true noise distribution - i.e. a gaussian noise in phase and modulus - and our gaussian approximation (see fig. 7)

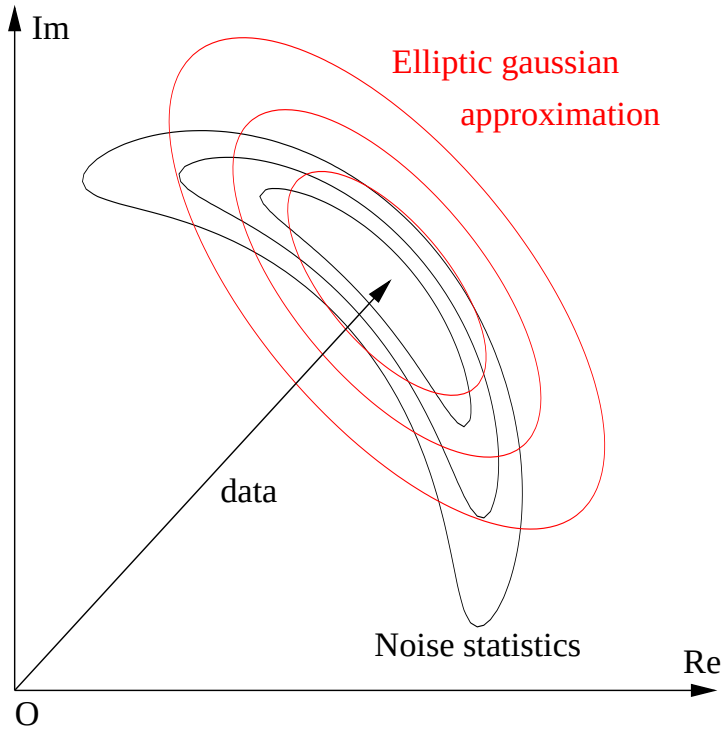

FIG. 7 – Polar gaussian distribution contour plot and its cartesian gaussian approximation.

E Acknowledgements

We WISARD hereby express their gratitude to the following people, as WISARD uses some of their routines :
– Éric Thiébaut (from CRAL) : WISARD heavily uses his (neat) OptimPack software as its minimization engine.
– Frédéric Cassaing, Jean-Marc Conan and Jean-François Sauvage : WISARD uses a few routines by these ONERA scientists. These routines (and some others by Laurent Mugnier) are distributed along with WISARD , with these authors' permissions, in the `lib/oneralib/` directory.
– John D. Monnier for his `OIFITS` reading routines, as well as the authors and maintainers of the NASA astronomical library, for their routines used by John's ones. The `OIFITS` reading routines are available at http ://dept.astro.lsa.umich.edu/~monnier/oi_data/, while NASA's astronomical library can be found at http ://idlastro.gsfc.nasa.gov/. Their routines are included in the `lib/extralib/` directory of the WISARD distribution.

Références

[1] S. Meimon, *Reconstruction d'images astronomiques en interférométrie optique*, Thèse de doctorat, Université Paris Sud (2005). 1.1

[2] S. Meimon, L. M. Mugnier et G. Le Besnerais, *A convex approximation of the likelihood in optical interferometry*, J. Opt. Soc. Am. A (novembre 2005). 1.1, 5, D, D.1

[3] S. Meimon, L. M. Mugnier et G. Le Besnerais, *Reconstruction method for weak-phase optical interferometry*, Opt. Lett., 30 (14), pp. 1809–1811 (juillet 2005). 1.1

[4] L. M. Mugnier, G. Le Besnerais et S. Meimon, *Inversion in Optical Imaging through Atmospheric Turbulence*, Dans *Bayesian Approach to Inverse Problems*, sous la direction de J. Idier, Digital Signal and Image Processing Series, chap. 10, pp. 243–283. ISTE / John Wiley, London (2008). 1.1

[5] P. R. Lawson, W. D. Cotton, C. A. Hummel, J. D. Monnier, M. Zhao, J. S. Young, H. Thorsteinsson, S. C. Meimon, L. Mugnier, G. Le Besnerais, E. Thiébaut et P. G. Tuthill, *An interferometric imaging beauty contest*, Dans *New frontiers in stellar interferometry*, sous la direction de W. A. Traub, vol. 5491, pp. 886–899. Proc. Soc. Photo-Opt. Instrum. Eng. (2004), Conference date : juin 2004, Glasgow, UK. 1.1, 4.2, 4.2

[6] G. Demoment, *Image Reconstruction and Restoration : Overview of Common Estimation Structures and Problems*, IEEE Trans. Acoust. Speech Signal Process., 37 (12), pp. 2024–2036 (décembre 1989). 4.1.1

[7] J. Idier, rédacteur en chef, *Approche bayésienne pour les problèmes inverses*, Hermès, Paris (2001). 4.1.1

[8] J.-M. Conan, L. M. Mugnier, T. Fusco, V. Michau et G. Rousset, *Myopic Deconvolution of Adaptive Optics Images by use of Object and Point Spread Function Power Spectra*, Appl. Opt., 37 (21), pp. 4614–4622 (juillet 1998). 4.1.1, 5

[9] L. M. Mugnier, C. Robert, J.-M. Conan, V. Michau et S. Salem, *Myopic deconvolution from wavefront sensing*, J. Opt. Soc. Am. A, 18, pp. 862–872 (avril 2001). 4.1.1

[10] L. M. Mugnier, T. Fusco et J.-M. Conan, *MISTRAL : a Myopic Edge-Preserving Image Restoration Method, with Application to Astronomical Adaptive-Optics-Corrected Long-Exposure Images.*, J. Opt. Soc. Am. A, 21 (10), pp. 1841–1854 (octobre 2004). 4.1.1, 4.3, 5

[11] G. Le Besnerais, S. Lacour, L. M. Mugnier, E. Thiébaut, G. Perrin et S. Meimon, *Advanced Imaging Methods for Long-Baseline Optical Interferometry*, IEEE Journal of Selected Topics in Signal Processing, 2 (octobre 2008). 4.1.1, 4.3, 5

[12] S. C. Meimon, L. M. Mugnier et G. Le Besnerais, *A novel method of reconstruction for weak-phase optical interferometry*, Dans *New frontiers in stellar interferometry*, sous la direction de W. A. Traub, vol. 5491, pp. 909–919. Proc. Soc. Photo-Opt. Instrum. Eng. (2004), Conference date : juin 2004, Glasgow, UK. 4.2

[13] T. J. Cornwell et P. N. Wilkinson, *A new method for making maps with unstable radio interferometers*, Mon. Not. R. Astr. Soc., 196, pp. 1067–1086 (1981). 5

[14] J. W. Goodman, *Statistical optics*, John Wiley & Sons, New York (1985). 5.2.1