"Advanced" data reduction for the AMBER instrument

Florentin Millour, Bruno Valat, Romain Petrov, Martin Vannier

HAL Id: hal-00291883
https://hal.science/hal-00291883

"Advanced" data reduction for the AMBER instrument[*]

Florentin Millour[a], Bruno Valat[b], Romain G. Petrov[b], Martin Vannier[b]

[a]Max-Planck Institut for Radioastronomy, Auf dem hügel, 69, 53121, Bonn, Germany;
[b]Fizeau Laboratory, Nice university, Parc Valrose, Nice, France.

ABSTRACT

The amdlib AMBER data reduction software is meant to produce AMBER data products from the raw data files that are sent to the PIs of different proposals or that can be found in the ESO data archive. The way defined by ESO to calibrate the data is to calibrate one science data file with a calibration one, observed as close in time as possible. Therefore, this scheme does not take into account instrumental drifts, atmospheric variations or visibility-loss corrections, in the current AMBER data processing software, amdlib.

In this article, we present our approach to complement this default calibration scheme, to perform the final steps of data reduction, and to produce fully calibrated AMBER data products. These additional steps include: an overnight view of the data structure and data quality, the production of night transfer functions from the calibration stars observed during the night, the correction of additional effects not taken into account in the standard AMBER data reduction software such as the so-called "jitter" effect and the visibility spectral coherence loss, and finally, the production of fully calibrated data products.

All these new features are beeing implemented in the modular pipeline script `amdlibPipeline`, written to complement the amdlib software.

Keywords: AMBER, VLTI, Interferometry, data processing, calibration

1. INTRODUCTION

The AMBER data reduction software: `amdlib`,[1] features a new type of algorithm, the so-called P2VM algorithm,[2] which performs a direct fit of the observed fringes using a template fringe pattern recorded during the instrument calibration and produces a series of observables; namely, the squared visibility, the closure phase, and the differential phase.[3] The standard calibration plan handles the AMBER signal from the raw data to raw averaged visibilities OI DATA files. Though a simple calibration script (`amdlibDivideOiData`) is included in the `amdlib` software, a complete calibration procedure is not yet provided.

In this document, we present our approach to produce calibrated OI data files. It consists of several recipes, just like for the standard `amdlib` software, to organize, evaluate, plot, and finally, calibrate the data. This philosophy allows one to switch easily from one recipe to another if necessary. This leads to the modular structure of the example pipeline script `amdlibPipeline`[†], written in the scientific language `yorick`, to perform the final steps of data reduction after an `amdlib` data reduction session.

This paper is organized into three different sections:

- A section about file management and coping with different types of calibrations (different DITs, different modes, use of BCD),

- The rise of advanced calibration methods: "coherence length" and "jitter" effects correction,

- How to cope with calibration stars and transfer function, as well as a template for data calibration.

---

2. DATA MANAGEMENT

The AMBER user generally faces a first difficulty when receiving the ESO DVDs: one gets a series of files sorted by night or by observing block number, with a name reflecting the observing time: "AMBER.yyyy-mm-ddThh:mm:ss.sss.fits". The `amdlib` software, or even the ESO tool `gasgano`, already present some tools to ease the sorting and use of the AMBER raw data files: colouring the different files types, or "automatic" data reduction scripts. Here we propose to add some more comfort to the user to evaluate information on the overall observing night. Therefore, this section is meant to answer the following questions: How many stars and calibrators were observed during a night ? What is the UV coverage for the science stars ? Was the observing night a success or not ?

2.1 Find a star in a night directory

The small script `amdlibListStars` rapidly gives the information of which stars were observed during a night and whether they are tagged as "SCIENCE" or "CALIB". It makes use of the log file produced by the standard `amdlib` software and just compiles it into a short stars list. An example of the type of result one can get from this script is given below:

```
> amdlibListStars
 Choose the product directory (OI_DATA)

 The file/directory chosen is:
    /disk/DATA/2007-05-16/
 HD124454 CALIB
 Alpha_Arae SCIENCE
 HD164371 CALIB
 hd-test
 HD-93813
```

Here one can see that the actually observed stars are HD 124454, α Arae, and HD 164371. The star α Arae is the science star, whereas HD 124454 and HD 164371 are two calibrators. The other two targets: hd-test and HD 93813 were meant to be observed, but the observations were not executed (no `CALIB` or `SCIENCE` tag associated).

In the same philosophy, the script `amdlibListAllStars` does the same, but on a series of night data directories, and produces a résumé file `starList.txt` listing the stars and giving the different directory where we find them.

2.2 Check file tags and fix them

For some old data, or during a technical night where all the stars are tagged as "CALIB", it may be convenient to tag again the corresponding files to "SCIENCE". The script called `amdlibConvertScienceToCalib` is made for that purpose and applies new tags to all the files linked to a given star. It makes use of the precedent function and fits files read/write routines.

2.3 Sorting files by integration time, instrument mode, etc.

This is often convenient to sort the data files with specific properties such as the instrument modes, whether the fringe tracker was used or not, etc. The function `sortAMBERFiles` allows one to sort the files by creating sub-directories and producing symbolic links to the original files. By default, it will sort the files by star name, but it is often convenient to sort the files by exposure time instead (as the transfer functions at different exposure times do not match). In table 1, one can find the relevant fits keywords for sorting the files, with a short description of the sorting criteria. For example, if one wants to sort files by exposure time, the following command should be typed in yorick:

```
> sortAMBERFiles, keyword="ESO DET DIT", keyDec=1
```
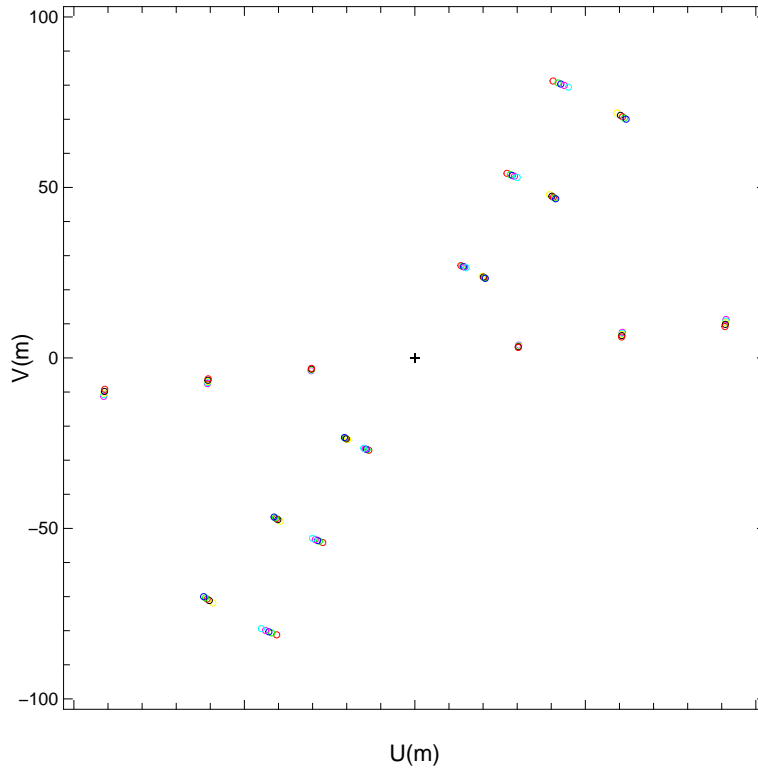
Figure 1. UV coverage of a star observed in 2007 in a full night. Here, 3 consecutive shots were recorded with the large aligned triplet.

In case one works with compressed files (the suffix of the files is then `.fits.gz` instead of `.fits`), the symbolic links will not work with the amdlib software. Alternatively, one can move the files instead of creating symbolic links using the option `symbolic=0`.

Table 1. Typical ESO FITS keywords usable for sorting the AMBER files.

FITS keyword	Short description
ESO OBS TARG NAME	Star name
ESO DET DIT	Detector Integration Time (exposure time)
DATE-OBS	Date of observation
ESO INS MODE	Instrument mode: whether 2 or 3 telescopes, spectral resolution, etc.
ESO DEL FT STATUS	FINITO (fringe tracker) status: ON or OFF.
ESO INS OPTI7 NAME	BCD (Beam Commuting Device) status: IN or OUT

2.4 UV coverage during a night

An interesting feature is to know what type of UV coverage was observed for a given star in an observing night. The small script `amdlibPlotUV_Night` allows one to plot the UV coverage for all the science stars present in a data directory (see Fig. 1). One option also allows one to plot the calibration stars UV coverage.

3. ADVANCED DATA PROCESSING

The amdlib software lacks two important steps in the data processing scheme; namely, the coherence loss correction and the so-called "jitter" correction. We present here the developments we made to correct these two effects and the implications they have on the data processing.

3.1 Correcting the visibilities from coherence length losses

This step is fundamental for low spectral resolution data, as the coherence length L_c is smaller than the maximum atmospheric optical path difference (OPD) stroke, with typical values of $30\mu m$ and $35\text{-}40\mu m$, respectively. As shown in Millour et al.,[4] the probable shape of the spectral channels of AMBER is a Gaussian curve of FWHM L_c. A first idea is to correct instantaneous squared visibilities from this visibility loss using the frame-by-frame OPD measurement.

$$\rho(\delta) = exp\left[-2\ln(2)\left(\frac{\delta}{L_c}\right)^2\right] \tag{1}$$

However, the very important thing to notice here is that the OPD value, δ, given in equation 1 is an *absolute* value of relative to the white fringe. Instead, the OPD measurement in AMBER is always made relative to the P2VM reference. Therefore, this P2VM OPD effect has to be taken into account as it can strongly affect the OPD measurements (see Fig. 5). Moreover, since the AMBER fibers are not exactly of equal length, a chromatic OPD is induced in the fringe pattern. This leads to curved fringes and, thus, an OPD measurement dependent on λ. Therefore, one needs a λ-dependent OPD measurement on the P2VM to correct for it.
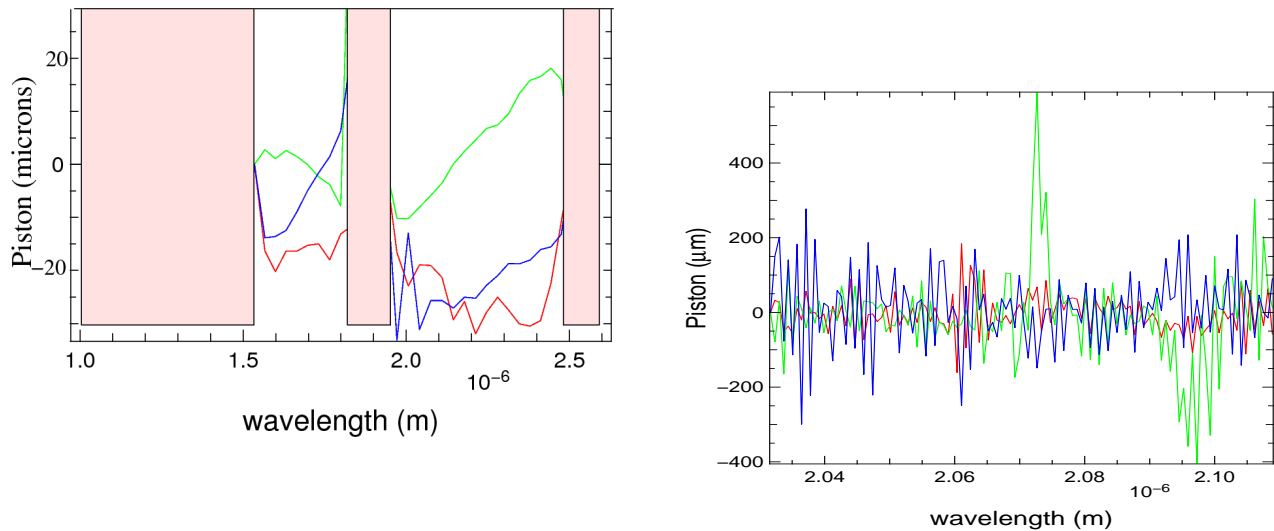

Figure 2. *Left*: Example of a non-zero OPD on a P2VM. The masked zones correspond to the inter-bands and the J band where the OPD cannot be computed. *Right*: Example of a γ^2 Vel observation, where the object's phase can introduce chromatic "OPD" as high as $500\mu m$, which is no longer negligible for this medium resolution data (same as the one of Fig. 5).

For the same reason, because the atmosphere also induces chromatic effects, one needs to take into account the chromatic OPD on the target itself to correct for the curved fringes in the white fringe position estimation. Figure 2 illustrates these two problems for two extreme cases:

- for a low resolution P2VM, where the chromatic P2VM OPD can be as high as $20\mu m$ due to a non-optimal OPD alignment of the AMBER instrument,

- for a medium resolution data-set, where the OPD effect coming from the object's phase itself (peaks in the green base) can produce an off-centering of the white light fringe compared to the usual OPD computation.

The function `amdlibCorrectCoherenceLength` is a proposition for such an implementation: it uses a `RAW` OI data file, containing instantaneous squared visibilities and OPD measurements as input, and produces a raw OI data file called `COR`, where the visibilities have been corrected according to equation 1, using an OPD

measurement both on the associated P2VM and on the curved fringes information contained in the differential phase of the object. Then, the usual steps of `amdlib` (frame selection and data averaging) can be proceed on these files.

3.2 Correcting the visibilities from the "jitter" effect

Another origin of visibility loss is the fringe motion during one single exposure time, usually called "jitter" because it reflects the fringe smearing onto the instrument detector during one frame integration.

Based on a proposition from Malbet et al.,[5] we developed an estimator based on the assumption that the jitter effect comes only from a regular phase shift between two consecutive frames. To do so, we compute a cross-product of the raw complex coherent flux between two consecutive frames:

$$W_{ij}^{12} = C_{ij}(\lambda, t_1) \times C_{ij}^*(\lambda, t_2) \tag{2}$$

The phase of this cross-product gives an estimate of the phase difference between these two frames. An OPD is then computed on this cross-product, which is the difference of OPD between the two frames. Then, a correlation plot between the OPD, OPD difference, and squared visibility is built (see Fig. 3). In the OPD direction, the correlation is "well known" and can be computed according to equation 1. In the "jitter" direction, we find that the squared visibility decrease can be empirically described with a Gaussian-like shape. Therefore, one can describe the squared visibility as a function of the coherence loss and of the jitter as a 2-D Gaussian curve. Therefore, one can use this now known dependence to correct the visibilities for jitter and coherence losses, and compare it with uncorrected data. This is shown in Fig. 3 on the right, where the original squared visibility histogram, which has a log-normal distribution, becomes a Gaussian histogram after correction. This example test has been repeated on many data sets and always shows the same improvement of the visibilities histogram.

Moreover, we find that the transfer function (i.e. the squared visibility vs time, for one star) is more stable after this jitter correction than if only the coherence length correction is applied to the data. Therefore, we can claim here that we have found a way of estimating and correcting the jitter effect, which was a burden for low spectral resolution AMBER data until now. The next step for now is to implement this method in a robust way into the standard AMBER data processing software.

4. CALIBRATING THE DATA

Data calibration can be done in several ways, using classical optical interferometry calibration techniques (like in Perrin 2003[6]) or speckle data calibration. The aim of this section is not to explain these different calibration techniques, but to propose a data handling framework where all types of calibrations could be done more easily.

4.1 Finding/registering the diameter of calibration stars

This step consists in browsing all the calibration stars observed during a night and finding their associated angular diameters to correct in a further step their observed visibility from the expected one. Usually, it is left to the user to do such step, but it is not as easy as one might think. In several cases (browsing archival data, etc.), one does not have direct access to the calibration stars diameters. Moreover, many catalog databases (reported in table 2) and calibrator-seeking software (`ASPRO`[‡] (based on the work of Bonneau et al.[7]), `CalVin`[§] or `getCal`[¶]) exist, which make the whole calibration diameter finding step complicated.

The approach used here is to produce a local database containing sufficient information, taken from the fits file header and used to identify the stars. This information is namely the star name used during the observation and its coordinates. This allows one to query the different catalogs listed in table 2 from CDS and fill-in missing information in the database, used afterwards to correct the calibrator visibilities and produce a transfer function.

[‡]`http://www.mariotti.fr/aspro_page.htm`

[§]`http://www.eso.org/observing/etc/`

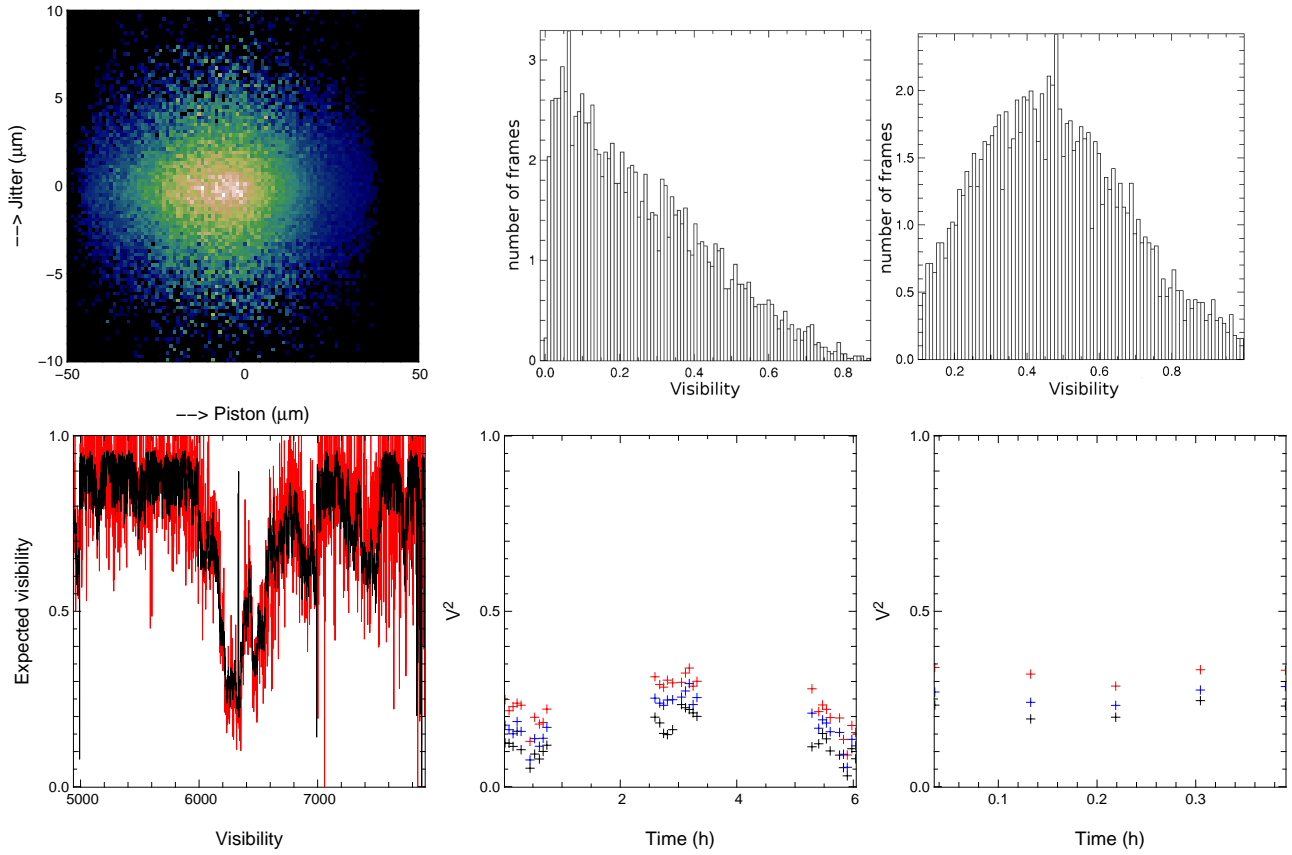[¶]`http://msc.caltech.edu/software/getCal/`

Figure 3. *Top-left:* Correlation plot showing the squared visibility as a function of our jitter estimator (vertical direction) and OPD (horizontal direction). One can see that in both directions, the squared visibility can be empirically described by a 2-D Gaussian-like surface. *Top-right:* Squared visibility histograms without (left) and with (right) jitter and OPD loss correction. The histogram switches from a log-normal distribution to a Gaussian distribution, which is much easier to reduce than before. *Bottom-left:* a time-sequence of squared visibilities (in red) compared to the expected visibility loss from jitter and piston variations. *Bottom-middle:* example of a 10hrs time-sequence for uncorrected V^2 (standard amdlib output without any frame selection, black crosses), with the coherence length corrected (in blue) and with the coherence length + jitter corrected (in red, upper curve). The typical scatter of visibilities is divided by 2 in this case (30% to 15% relative error). *Bottom-right:* a shorter time-sequence on very good quality data, showing a much higher improvement: from 16% relative error to less than 4%.

Table 2. Available catalogs of stars giving access to apparent stellar diameters. ind. stands for indirect estimation method, LBSI stands for Long Baseline Stellar Interferometry and Lun. occ stands for Lunar Occultation.

Name	Nr of stars	Techniques used	Vizier name	Reference
ESO VLTI	191	LBSI	J/A+A/434/1201	Richichi et al., 2005, A&A, 434, 1201
Bordé	374	ind.	J/A+A/393/183	Bordé et al., 2002, A&A, 393, 183
Mérand	948	ind.	J/A+A/433/1155	Mérand et al., 2005, A&A, 433, 1155
CHARM	3248	LBSI, Lun. occ.	J/A+A/386/492	Richichi et al., 2002, A&A, 386, 492
CHARM2	8231	LBSI, Lun. occ., ind.	J/A+A/431/773	Richichi et al., 2005, A&A, 431, 773
CADARS	13570	LBSI, ind.	II/224/cadars	Pasinetti et al. 2001, A&A 367, 521

In our pipeline, this database is a text file which can be automatically produced with the function `amdlibGetCalibzSize`, given that an Internet connection is available. This script perform a series of queries on CDS in the available catalogs of interferometric calibrators using the vizquery $\parallel$ utility from CDS.

4.2 Computing a transfer function

This step aims at building the measured instrumental response to calibrate afterwards the science data. The function `amdlibComputeTransFunc` uses the precedent calibrators database to identify the stars from the fits keywords contained in the files' headers. The transfer function is simply computed accounting for a uniform disk (UD) shape for the stellar disk. As most of the catalogs give UD diameters, this is relevant for the further steps. As many catalogs are used, it happens sometimes that the diameter was measured several times. In this case, we take the average diameter and compare the diameter dispersion with the existing errors. The expected calibration stars visibility is then computed according to the following equation:

$$V^2 = \left| 2 \frac{J_1\left(\frac{\pi B \theta}{\lambda}\right)}{\frac{\pi B \theta}{\lambda}} \right|^2 \tag{3}$$

The averaged measured visibility of the star is divided by this expected visibility and the result is saved in a new OI FITS file. This allows us to keep this calibration step made for further analysis using any of the existing calibration method. The variance on visibility from the diameter error, if present, is also computed and added to the visibility variance, to take this calibration error into account in the transfer function.

4.3 Checking the transfer function

Checking the quality of the transfer function may be a difficult task since AMBER is a spectroscopic instrument. In principle, one would have to plot 2D-maps giving the transfer function at each wavelength for each time of the night. Because this may not be easy to plot, we deliberately propose two functions to plot the transfer function projected onto the time axis or onto the wavelength axis:

- `amdlibPlotTransFuncTime` plots wavelength-averaged visibilities as a function of time, to check the overnight stability of the instrumental+atmospheric transfer function. As default, the entire wavelength range available is averaged for all of the files, but one can choose a wavelength interval with the `wlenRange` option to get only the wavelengths of interest.

- `amdlibPlotTransFuncWlen` plots a projection of the transfer function on the wavelength axis. This plot is similar to the one which is often used for the MIDI instrument, except that here we get 3 baselines and a closure phase.

One can see the result in figure 4 for the first science night of AMBER (25/12/2004), showing that the process works fine, even for old data sets.

4.4 Calibrating the data

The last step of the interferometric data reduction is the calibration of the science data itself. As explained in Perrin 2006,[6] the calibration of visibilities needs an accurate time-resolved estimate of the instrument and atmosphere transfer function. This is not the case for the AMBER instrument, as the minimum time between a science star and a calibration star is 30 mn (due to operation strategies and overheads in the VLTI).

Here, we present a simple approach for the AMBER data calibration, that can be used as an example for More sophisticated calibration algorithms. The function `amdlibCalibrateOiDataFMi` calibrates the data using the very simple assumption that the transfer function is constant throughout the night and that the variability appears as a scattering of visibilities around its average. This assumption is thought to be relevant in the current

---

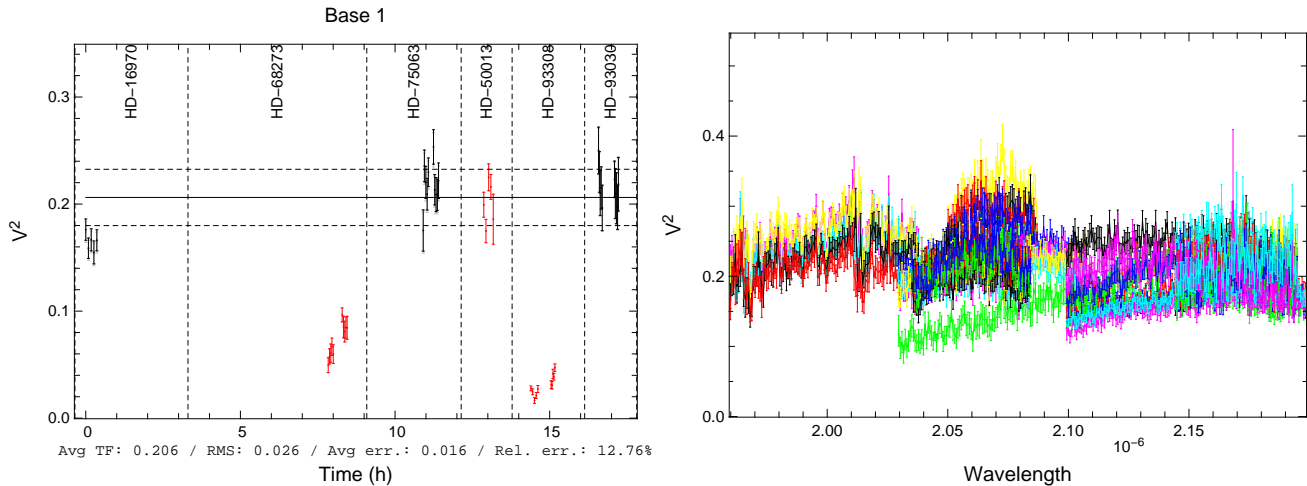$\parallel$ http://vizier.u-strasbg.fr/doc/vizquery.htx

Figure 4. Left: The transfer function of 25/12/2004 plotted by `amdlibPlotTransFuncTime`, in black, showing the stability over the night, to be compared with the science data published in several articles[8–10] in red. The star HD16970 was observed at low spectral resolution and pollutes the whole night transfer function. Right: The same transfer function plotted with `amdlibPlotTransFuncWlen`. Here the science data is not plotted as it would be more complicated to disentangle from the transfer function.

VLTI case, where large time intervals (more than 10mn) are used between calibration stars, preventing the interpolation of the transfer function with respect to time.

Therefore, this script loads the transfer function and the science-tagged files, computes an average transfer function and a scatter for each science data wavelength, and divides the latter by the first. This function provides calibrated OI fits files as a result that can be used for science. An example of the resulting calibrated observables plot is shown in figure 5 and can be compared with Millour et al.[9]
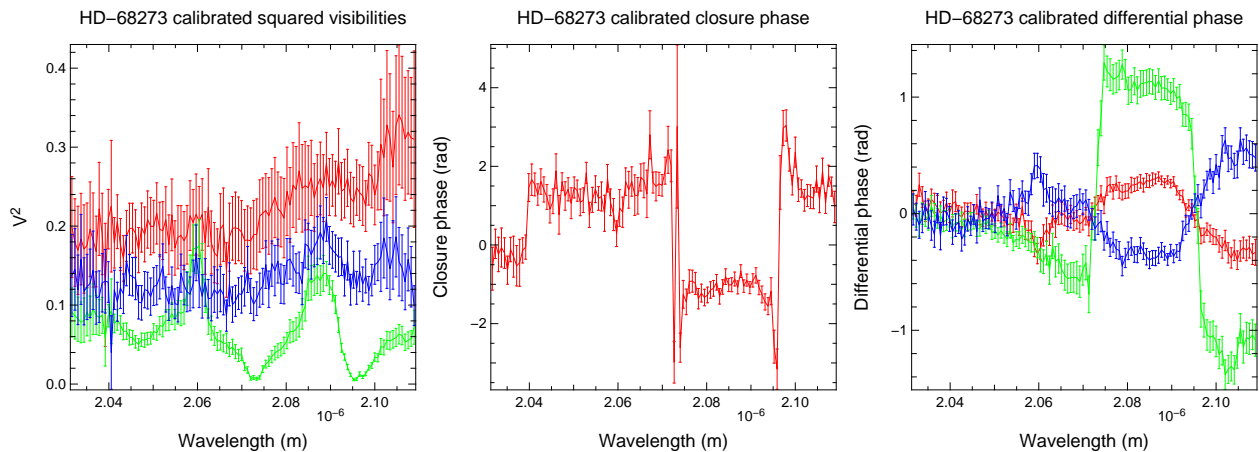

Figure 5. Left: Calibrated squared visibilities Middle: Calibrated closure phase Right: Calibrated differential phases

5. CONCLUSION

The standard default calibration scheme proposed for AMBER (calibrating one file with another) is obviously insufficient to get realistic error bars, and an overnight calibration quality estimate.

We presented a proposition for a user-friendly pipeline framework for the amdlib software. We are able, with the proposed scripts and an Internet connection, to produce science-grade data with realistic error bars in a semi-automated way, as it is already done for the first data reduction steps in `amdlib`.

This article is thought to be a starting point for further reflexions about AMBER data calibration. Further work could allow one to perform the AMBER data reduction in a fully automated way, allowing in the future the production of archived data products, in the same way as it is done with the ESO imaging and spectroscopic instruments (WFI, FEROS, etc.).

REFERENCES

[1] Tatulli, E., Millour, F., Chelli, A., and Duvert, G. a., "Interferometric data reduction with AMBER/VLTI. Principle, estimators, and illustration," *A&A* **464**, 29–42 (Mar. 2007).

[2] Millour, F., Tatulli, E., Chelli, A. E., Duvert, G., Zins, G., Acke, B., and Malbet, F., "Data reduction for the AMBER instrument," in [*New Frontiers in Stellar Interferometry.*], Traub., W. A., ed., **5491**, 1222, SPIE (Oct. 2004).

[3] Millour, F., Vannier, M., Petrov, R. G., Chesneau, O., Dessart, L., and Stee, P., "Differential interferometry with the AMBER/VLTI instrument: description, performances and illustration.," in [*Astronomy with High Contrast Imaging III: Instrumentation and data processing*], Aime, C., Ferrari, A., and Carbillet, M., eds., in press, EAS Publications Series (2006).

[4] Millour, F., Petrov, R., Malbet, F., Tatulli, E., Duvert, G., Zins, G., Altariba, E., Vannier, M., Hernandez, O., and Causi, G. L., "AMBER on the VLTI: data processing and calibration issues," *ArXiv e-prints* **705** (May 2007).

[5] Malbet, F., Berger, J.-P., Colavita, M. M., Koresko, C. D., Beichman, C., Boden, A. F., Kulkarni, S. R., Lane, B. F., Mobley, D. W., Pan, X. P., Shao, M., van Belle, G. T., and Wallace, J. K., "FU Orionis Resolved by Infrared Long-Baseline Interferometry at a 2 AU Scale," *ApJ* **507**, L149–L152 (Nov. 1998).

[6] Perrin, G., "The calibration of interferometric visibilities obtained with single-mode optical interferometers. Computation of error bars and correlations," *A&A* **400**, 1173–1181 (Mar. 2003).

[7] Bonneau, D., Clausse, J.-M., Delfosse, X., Mourard, D., Cetre, S., Chelli, A., Cruzalèbes, P., Duvert, G., and Zins, G., "SearchCal: a Virtual Observatory tool for searching calibrators in optical long baseline interferometry I: The bright object case," *A&A* **4469** (2006).

[8] Meilland, A., Millour, F., Stee, P., and al., D., "An asymmetry detected in the disk of κ Canis Majoris with AMBER/VLTI," *A&A* **464**, 73–79 (Mar. 2007).

[9] Millour, F., Petrov, R. G., Chesneau, O., Bonneau, D., and al., D., "Direct constraint on the distance of γ^2 Velorum from AMBER/VLTI observations," *A&A* **464**, 107–118 (Mar. 2007).

[10] Weigelt, G., Kraus, S., Driebe, T., and Petrov, R. G. a., "Near-infrared interferometry of η Carinae with spectral resolutions of 1 500 and 12 000 using AMBER/VLTI," *A&A* **464**, 87–106 (Mar. 2007).