IDRS

Data Reduction Software for the IOTA-3T Interferometer

Software Manual

Version 2005-09-18

(for Software Version 0.7)

Stefan Kraus[1,2]

`skraus@mpifr-bonn.mpg.de`

[1]Max-Planck Institute for Radio Astronomy (MPIfR), Auf dem Hügel 69, D-53121 Bonn, Germany

[2]Department of Astronomy, University of Massachusetts, Amherst MA 01002, USA

Contents

1. Introduction

The IOTA data reduction software (IOTA-DRS, IDRS) was developed as part of a Master thesis project (Kraus 2003) at the University of Massachusetts at Amherst, and significantly improved at the Max Planck Institute for Radio Astronomy at Bonn and is now available to all users of the IOTA facility. In this manual I provide a brief overview of the functionality of this data reduction software.

The software provided allows the user to extract visibility and closure phases from Michelson interferograms. For the visibility estimation, two algorithm are offered. One of them is based on the Power Spectrum, whereas the other uses the continuous wavelet transform (CWT). Closure phases are extracted using the Fourier transformed of the scan. The program is simple to use and has been tested and optimized for IOTA scans obtained on broad-band filters (e.g., J, H, K band). A reduction of narrow-band data should be possible after adopting the available parameters, even if the software has not been tested yet for this kind of data.

Naturally, it is the primary interest of the observer to have a "first glance" at the data already during the duration of an observation run. This may help to detect technical problems of the system or to evaluate how strong a source of unknown extension is resolved by the interferometer. A mode is offered which allows the automatic reduction parallel to data acquisition. In this case, the software checks continuously for completed data files, reduces them, and displays the raw and calibrated visibilities and closure phases on the screen. Since this process is completely autonomous, the user can still concentrate on the observation itself.

IDRS produces output files which are consistent with the IAU data exchange standard for optical/IR interferometry based on FITS[1] and can therefore easily be used for further processing.

[1]For a definition of this standard see http://www.mrao.cam.ac.uk/∼jsy1001/exchange

2. Copyright Notes, Warranty, and Complementary Software

All IOTA observers are invited to make use of IDRS. If you intend to publish data which was reduced with IDRS, I ask for collaboration. This software is part of a more complex software package comprising the following tools:

SimMap: Simulates the uv-coverage which is obtained by observing with IOTA on various telescope configurations. Assuming an arbitrary source-brightness distribution, the visibilities and closure phases are simulated. This software may also be used to fit measured visibilities and closure phases to model images, e.g. generated by radiative transfer codes.

ModelBin: Fits binary-star models to the IDRS output file. The fits include the position of the stars, their uniform-disk diameters, and the flux ratio.

Map: Produces aperture-synthesis maps from the IDRS output file and allows the user to analyze them. Implemented mapping strategies: Conventional Hybrid Mapping, Difference Mapping, and the Building Block Method.

Please feel free to contact the author if you are interested in making use of this complementary software. Data reduced with this software may be found in Kraus et al. (2005). Generated Aperture Synthesis Maps can be found in Monnier et al. (2004) and Kraus et al. (2005).

Please note that the software is distributed without any warranties as to performance or any other warranties, whether expressed or implied. No warranty of fitness for a particular purpose is offered. The user must assume the entire risk of using the provided code.

In order to make the installation procedure as convenient as possible, all software libraries which are needed for a successfull installation of IDRS were included into this package:

PGPLOT by Tim Pearson

FFTW by Matteo Frigo and Steven G. Johnson

CFITSIO by William D. Pence, HEASARC, NASA/GSFC

OIFITS-library by John Young

Comments and suggestions for future improvements of IDRS are very welcome (please send email to skraus@mpifr-bonn.mpg.de).

I appreciate helpful discussions with John D. Monnier. The initializing master thesis project was advised by Peter F. Schloerb.

3. Recommended Observation Strategy

In order to acquired all data needed for proper data reduction, we recommend the following observation strategy:

- Before and directly after each target star observation, at least one calibrator star of known stellar diameter should be observed. Between the calibrator and target observations, the instrumental configuration should not be changed.
- When data acquisition on a star (either calibrator or target) is completed, a set of "matrix files" should be taken. For these measurements two or all of the three beams are shuttered out. It is important to verify that during matrix file acquisition, the star trackers continue to track the object. In case one or all of the matrix files were not acquired sucessfully, the observer should simply repeat the acquisition. In the follow, I use the notation that M1 denotes the file for which the shutter of telescope A was open only. Accordingly, M2 and M3 denotes the files with shutter of telescope B and telescope C open. For the last matrix file (M4, "darkframe"), all shutters are closed such that only the thermal background is measured.

4. Brief Description of the Algorithms Implemented

4.1. Visibility Estimation

A major observable for Michelson interferometers is the fringe contrast (Visibility V), which can be measured from the White Light Fringe at zero optical path delay (OPD). In order to form a proper interferogram out of the measured raw data, certain pre-processing steps must be performed. IDRS offers two alternative algorithms as described in sections 4.1.1 and 4.1.2, whereas the second one should be used if possible.

To extract visibilities from these pre-processed interferograms, again two different algorithms are implemented in IDRS. A short description of both algorithms is presented in chapters 4.1.3 and 4.1.4.

4.1.1. Pre-proc. Algorithm 1: Fringe Contrast without Photom. Calibration

For each baseline, the IONIC3 beam combiner installed at IOTA provides two signals (let's call them I_1, I_2 for the baseline formed between telescope A and B). Due to the instrumental design, these raw signals have complementary phases. To remove common-mode noise fluctuations, a simple procedure is to subtract and normalize the signals, e.g.:

$$I_{12} = \frac{I_1(\tau) - I_2(\tau)}{I_1(\tau) + I_2(\tau)} \tag{1}$$

This simple procedure can also be used to reduce IOTA scans, for which only on one of the baselines, a fringe was recorded. However, this method does not take potential differences/imbalances in the photometry between in combined beams into account.

4.1.2. Pre-proc. Algorithm 2: Photometric Calibration (requires 3T mode)

In cases where fringes could be acquired simultaneously on all three baselines, the photometric information can be deduced from the interferometric signals. This is done using a matrix inversion (similar to the procedure for asymmetric beam combination presented by Monnier (2001)). Before performing the matrix inversion, we smooth the signal in order to avoid numerical instabilities in the determined photometric signals. This is done by convolution with a gaussian of user-defined width.

The obtained three photometric signals are used to form three photometrically corrected interferograms by applying the correction presented by du Foresto (et al.1997). For example, considering the signals I_1 and I_2 measured between telescope A and B, the correction is applied using the equation

$$I_{1,\mathrm{corr}}(\tau) = \frac{I(\tau) - \kappa_{1A} P_1(\tau) - \kappa_{1B} P_2(\tau)}{2\langle\sqrt{P_1(\tau)P_2(\tau)}\rangle\sqrt{\kappa_{1A}\kappa_{1B}}} \tag{2}$$

$$I_{2,\mathrm{corr}}(\tau) = \frac{I(\tau) - \kappa_{2A} P_1(\tau) - \kappa_{2B} P_2(\tau)}{2\langle\sqrt{P_1(\tau)P_2(\tau)}\rangle\sqrt{\kappa_{2A}\kappa_{2B}}} \tag{3}$$

where κ_{ij} denotes the components of the transfer matrix which links the measured interferometric intensities with the photometric output from the various telescopes.

Again, to avoid numerical instabilities, we perform a smoothing of the product of the photometric signals (denotes by $\langle\cdots\rangle$ in the equation above).

The corrected signals are subtracted in order to obtain the final interferogram:

$$I_{12,corr} = I_{1,\mathrm{corr}}(\tau) - I_{2,\mathrm{corr}}(\tau) \tag{4}$$

4.1.3. Algorithm 1: Power Spectrum

According to PARSEVAL's theorem the power of a signal is the same whether computed in time or frequency space. Since we may choose an arbitrary normalization, we can define the power spectrum as the absolute square of the fourier transform $\mathcal{P}(\nu) := |\mathcal{F}(\nu)|^2$. In practice, the power spectrum is calculated with a fast Fourier transform (FFT) and the relation $\mathcal{P}(\nu) := \Re(\mathcal{F}(\nu))^2 + \Im(\mathcal{F}(\nu))^2$. To avoid aliasing, the data is multiplied by a window function (the Welch window; Flannery (et al.1992)) before the computation.

A very nice property of power spectra is that in averaging, real signals add constructively in the power spectrum while it decreases noise. Thus this method is useful, especially for the measurement of the fringe power in cases of low SNR.

The width of the fringe peak in the power spectrum is given by the bandwidth but it might also be broadened by atmospheric piston. In addition, there is a noise background which has a positive, but quite constant, slope in most of the cases. To fit the fringe peak the background is first estimated by measuring the lowest power $P_l(\nu_l)$ in a window on the lower-frequency end of the covered frequency range. The same is done for P_r, measured at ν_r in a window on the high-frequency end of the power spectrum.

This averaged background level with constant slope is subtracted such that mainly the signal remains:

$$\mathcal{P}_{\mathrm{bgsub}}(\nu) = \mathcal{P}(\nu) - \left(\mathcal{P}_l + \frac{\mathcal{P}_l - \mathcal{P}_r}{\nu_l - \nu_r}(\nu - \nu_l)\right) \tag{5}$$

Finally the signal above this background is integrated and used as an estimation for the Raw Visibility.

4.1.4. Algorithm 2: Continous Wavelet Transform

Another approach, which is implemented in IDRS to estimate visibilities makes use the continous wavelet transformed (CWT) of the measured interferograms. For the decomposition, a *mother wavelet* $\psi(\eta)$ must be choosen, for which we use the Morlet function (a sine wave modulated with a Gaussian). This function shows similarities to the analytic fringe function (a sine wave modulated with a sinc) and is computational efficient to implement. The Morlet function is given by

$$\psi(\eta) = \pi^{-1/4} e^{ik_0\eta - \eta^2/2} \tag{6}$$

where k_0 is the wavenumber, which have to be adjusted such that the number of fluctuations within ψ roughly fits the number of fluctuations within a typical fringe package (we choose $k_0 = 6$).

Now the CWT is defined as a convolution of the signal $I_{\rm red}(\tau)$ with the complex conjugate of a *wavelet* (which is a translated and dilated/contracted version of the mother wavelet)

$$\mathbb{W}(\tau, s) = \frac{1}{\sqrt{s}} \int\limits_{-\infty}^{\infty} I_{\rm red}(t) \psi^* \left(\frac{\tau - t}{s} \right) dt \tag{7}$$

By varying s and τ one obtains a two-dimensional image with real and imaginary part. Finally, the wavelet power spectrum is given by $\mathcal{P}_W(\tau, s) := |\mathbb{W}(\tau, s)|^2$.
IDRS computes the CWT using the algorithm by Torrence & Compo (2004) which makes use of the FFT to accelerate the computation.

The CWT of the sample data, taken under bad atmospheric conditions with a low signal-to-noise ratio (SNR) can be seen in figure 2. Atmospheric piston as well as instrumental noise causes the numerous noise features beside the significant fringe power peak. To isolate the fringe itself, we apply a filter which removes all signal below some significance level (e. g. 35% scaled to the peak in $\mathcal{P}_W$). Since this filtering may not clip only the fringe peak but also additional strong noise features, an additional routine is applied which removes all those areas that are separated from the partition with the highest intensity in it. The power of the remaining peak is used as an estimation for the raw visibility μ^2. Since piston may also cause to spread the fringe peak, the extension of this remaining partition along the OPD and scale axis is used as a rejection criteria. Other potential rejection criteria are the size of the area over with the power is spread and the location of the peak on the spatial and scale axis.

In chapter 6.4, it will be explained how those parameter may be adopted in order to optimize the performance of IDRS.

4.1.5. Calibrating Visibilities

To calibrate the measured raw visibility μ for the instrumental efficiency, calibrator observations are used to determine the transfer function T. In the ideal case, these calibrators are point sources with a similar spectral energy distribution as the target. The calibrated visibility V is then given by

$$V^2 = \frac{\mu^2}{T^2}. \tag{8}$$

The software provides two different methods for determining the transfer function. In general, it is recommendable to interpolation the transfer function for an observation at a particular time t between neighboring groups of calibrators.

In a first step the software groups all contiguous measurements of one calibrator, and the average $\langle T \rangle$ is calculated for each group. To calibrate a fringe visibility $\mu(t)$ the transfer function is interpolated linearly between neighboring calibrator groups at t_l and t_r (with $t_l < t < t_r$, as illustrated in Figure 1). With the averaged transfer functions of the two groups $\langle T_l(t_l) \rangle$ and $\langle T_r(t_r) \rangle$, the interpolation equation becomes

$$T(t) = \langle T_l(t_l) \rangle + \frac{\langle T_l(t_l) \rangle - \langle T_r(t_r) \rangle}{t_l - t_r}(t - t_l). \tag{9}$$

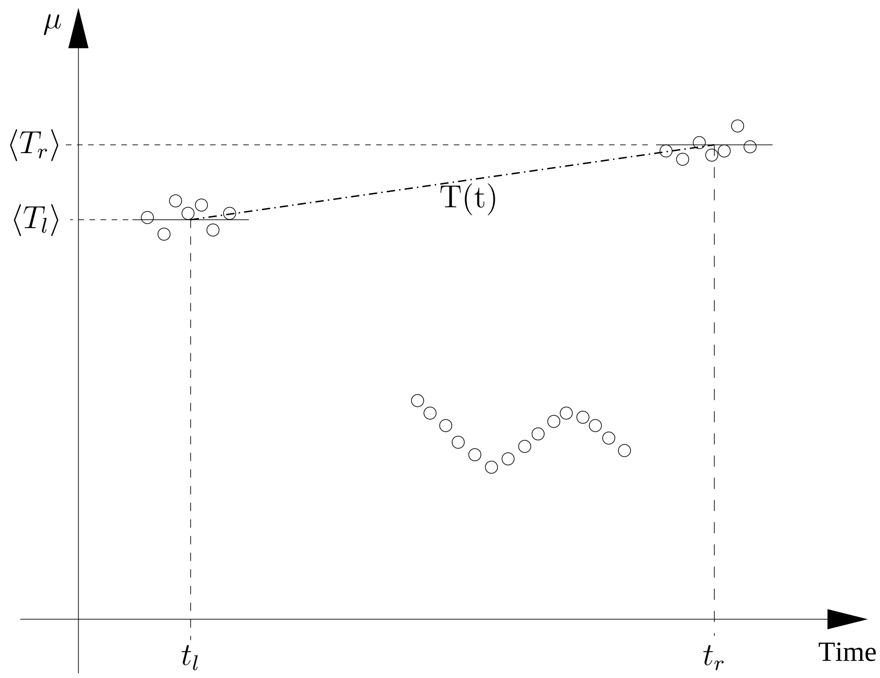

Fig. 1.— Linear interpolation between neighboring calibrator groups at t_l and t_r provides the transfer function $T(t)$.

Sometimes, directly neighboring calibrator groups might be not available. Then, the assumption can be made that T is quite stable over the night. Thus, the fringe visibility of all calibrators for that night can be averaged to obtain $\langle T \rangle$.

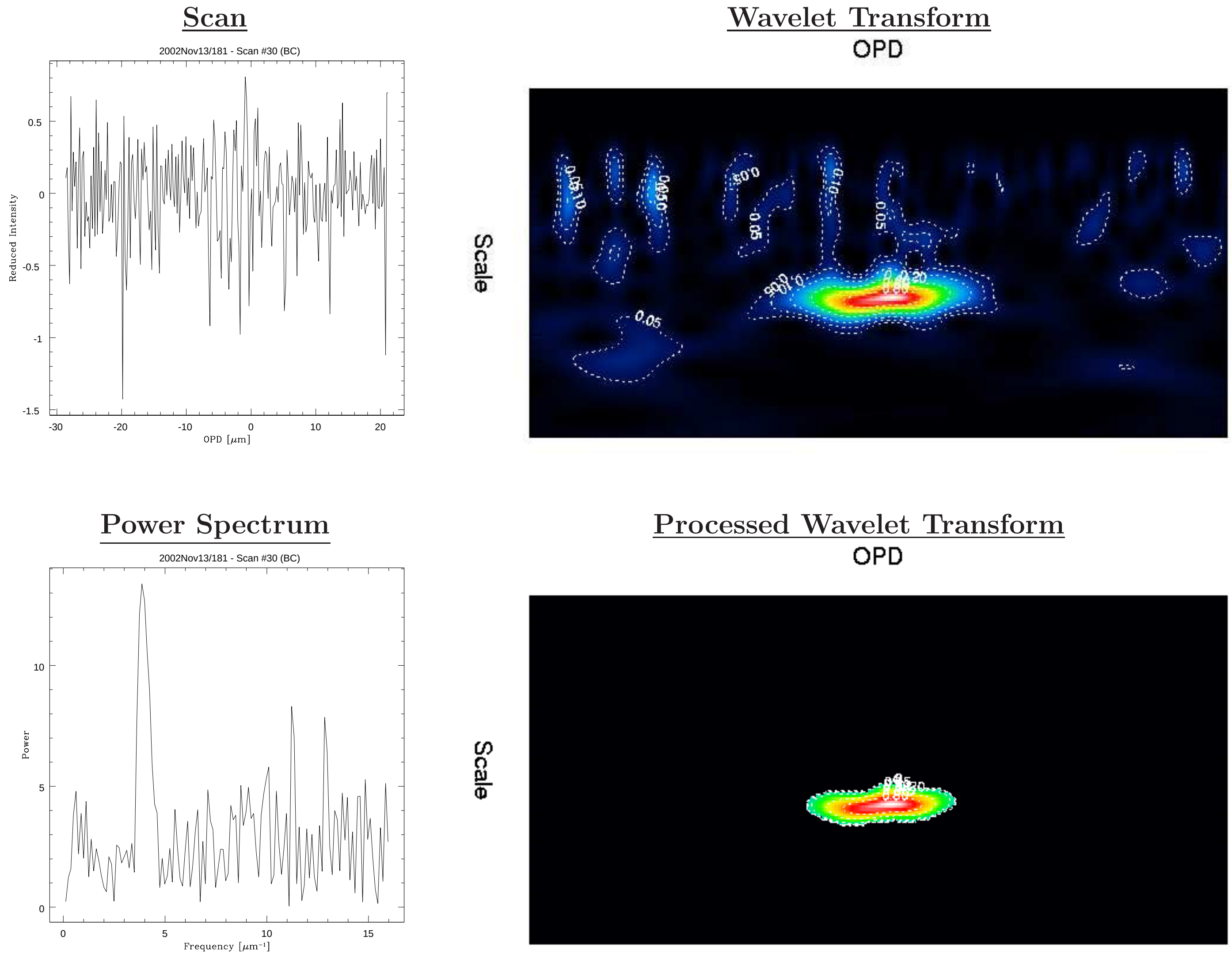

Fig. 2.— CWT of a scan obtained with IOTA on 2002 November 13. As the plot on the upper left (reduced intensity) and lower left (power spectrum) shows, the scan is strongly affected by atmospheric piston. Within the CWT (on the upper right) this piston appears as additive noise features. Performing background subtraction and a defragmentation algorithm, these features can be removed (final CWT in the lower right). Within the CWT plots, 5, 10, 20, 40, 60, and 80% contours (scaled to the signal peak) are shown.

4.2. Closure Phase Estimation

Closure phase measurement has to be made within a time interval shorter than the coherence time of the atmosphere t_0. Then, the closure phase Φ is given by

$$\Phi = \varphi_{AB} + \varphi_{BC} + \varphi_{CA}. \tag{10}$$

To estimate Φ, the software estimates the rough position of the fringes in all threebaselines using the *IOTA-Ames Fringe Tracker* (Wilson 2002) and average these to get a mean position within the scan which most likely contains high power in all three fringes. Then, a window is set around this averaged position. The size of this window can be chosen by the user to fulfill the above mentioned atmospheric condition. After calculating the FFT of all three windowed scans, the peak ν_{XY} within the power spectrum is found, and the phases φ_{AB}, φ_{BC}, φ_{CA} are calculated using

$$\varphi_{XY} = tan^{-1}\left[\Im(\mathcal{F}(I_{\rm red}(\nu_{XY})))/\Re(\mathcal{F}(I_{\rm red}(\nu_{XY})))\right], \tag{11}$$

where $\Im$ denotes the imaginary and $\Re$ the real part of a complex quantity, and $\mathcal{F}$ is the Fourier transform.

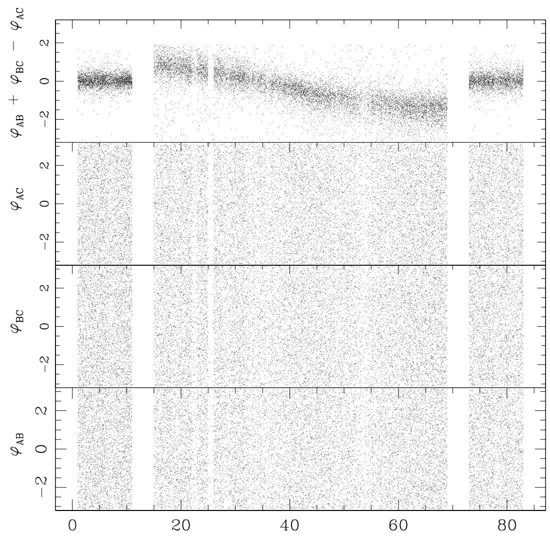

Fig. 3.— The individual phases and the closure phase of an unresolved calibrator (δ Aur, left: scan 0..13 and right: scan 72..85), and the resolved binary Capella in between. Data from IOTA, 2002Nov15/88..169

Figures 3 and 4 illustrate how the three influenced individual phases form the closure phase for one calibrator-target-calibrator sequence.

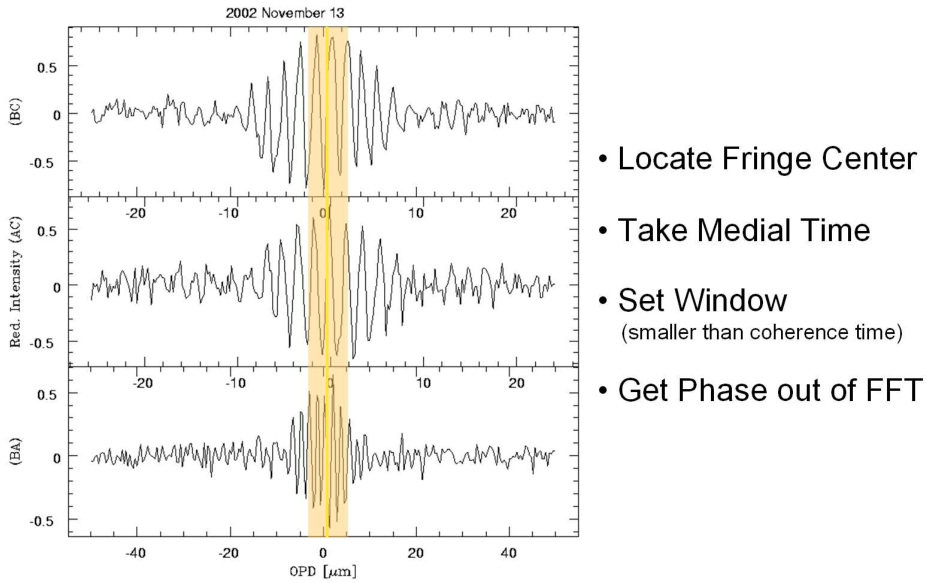

Fig. 4.— Illustration of the method used to extract closure phases.

To reject individual measurements of lower quality, a check is made of whether or not the frequency-closure relation is fulfilled

$$\nu_{closure} = \nu_{AB} + \nu_{BC} + \nu_{CA} = 0. \tag{12}$$

Finally, the individual closure phase estimations for all scans within a datafile are averaged. This is done on the complex plane to take the 2π periodicity of the phase into account.

4.2.1. Calibrating Closure Phases

The closure phase offset must be determined from calibrator observations. Therefore, IDRS averages the closure phases for each group of calibrators and interpolates linearly between neighboring groups. The interpolated offset is subtracted from the target closure phase.

5. The IDRS Software Package

5.1. System Requirements

It should be possible to install the IDRS software on each Linux and Sun system, although only a limited number of compiler/operation system combination could be tested yet.

Please ensure before the installation that the include and source files for the GNU C compiler and the X-window X11 are installed on your system. In case you are using SuSE Linux, please ensure with YaST that the following packages are installed: Beside ”gcc” and ”gcc-g77”, the X11 source files ”XFree86-devel” (SuSE-Linux 9.1) or ”xorg-x11-Mesa-devel” (SuSE 9.2) are needed.

5.2. Installation

To install the IDRS software, you should first download the most recent version from the following ftp site:
`ftp://ftp.mpifr-bonn.mpg.de/outgoing/skraus/idrs/`

Beside the installation file (e.g. `idrs-*.*.tar.gz`), I recommend to download also the file `testdata.tar.gz`, which contains sample data to test IDRS on your machine. Please download these files into a temporary directory and decompress/expand the installation file (but **not** the testdata archive), using e.g.

```
tar zxvf idrs*.tar.gz
```

Now, you must set the environment variable `IDRSROOT` to the path, in which you want to install the software. Depending on the shell you use, this might be done using e.g.

```
setenv IDRSROOT /home/user/idrs
```

or

```
export IDRSROOT=/home/user/idrs
```

In the following, we will give only the csh-shell commands (`setenv`). In case you are using the bash-shell, please ensure to use the appropriate syntax instead (`export`).

To proceed with the installation, please choose a operation system/compiler combination from Table 1 and run the bash shell script `install.sh` with the operation system name as first argument and the compiler as second argument, e.g.

```
./install.sh linux g77_gcc
```

This should start the complete installation routine, which will automatically install the required libraries PGPLOT, CFITSIO, FFTW, OIFITS, and finally create also the binaries `idrs` and `idrsplot`. After the installation script has finished successfully, you will find the binaries in the directory `$IDRSROOT/bin`, which I recommend to add to your PATH variable.

In addition, the PGPLOT software need information where the shared library for PGPLOT is installed. This can be done by setting the environment variables `LD_LIBRARY_PATH` and `PGPLOT_DIR`. Please set these variables in your ∼`/.cshrc` file (for bash-shell: ∼`/.bashrc`) by adding the following entries (please replace `$IDRSROOT` by the appropriate path):

```
setenv LD_LIBRARY_PATH $IDRSROOT/lib
setenv PGPLOT_DIR $IDRSROOT/src/pgplot
```

On some systems, `LD_LIBRARY_PATH` might be used already by other software installations. You might check this using the command `echo $IDRSROOT`. In case, the variable is used already, you might add the entry `setenv LD_LIBRARY_PATH $LD_LIBRARY_PATH:$IDRSROOT/lib` instead.

Table 1: Offered operation system/compiler combinations

Operation System	Compiler	Comments
linux	g77_gcc	Tested
sol2	f77_gcc	Tested (Solaris 2.x, SunOs 5.x)
sun4	f77_gcc	Not tested (SunOS 4.x)

IMPORTANT: BYTE SWAPPING

Currently, the interferograms are recorded at the IOTA site using a Solaris system. Reading this data with a PC requires to perform a byte-swap. The IDRS software will perform this byte swap automatically if the flag `SWAPBYTES` is set. The installation script sets this flag automatically when `linux` is specified as operating system. You might change this after the installation is finished by editing the file `$IDRSROOT/src/configure`. Setting the line `SWAPBYTES=''1''` enables byte-swapping. After any changes are made, you have to rerun `configure` and `make` in the `$IDRSROOT/src/` directory.

6. IDRS

6.1. Reducing the testdata

After the installation, the default [IniFile] `idrs.ini` is configured such that the path is already set to the testdata directory (find the observation log for this testdata in Table 2). Therefore you can start the reduction simply by changing into the `$IDRSROOT/bin` directory and typing the command

```
idrs all
```

This will start the data reduction using the parameters given in the default configuration file `idrs.cfg`:

Table 2: Observation Log for the testdata set, taken on 2003 March 24 on the target λ Vir and the calibrator star HD 126035. The following baseline configuration was used: Telescope A on station ne35, telescope B on se15, telescope C on ne10.

Date [UT]	Time [UT]	Hour Angle [h]	Source	Filter	No. of data file
starting data acquisition					(without matrix files)
03/24/03	07:24	-2.29	HD 126035	H	73-76
			M1-M4 HD 126035	H	77-80
	07:52	-1.76	λ Vir	H	81-104
			M1-M4 HD 126035	H	105-108
	08:20	-1.35	HD 126035	H	109-113
			M1-M4 HD 126035	H	114-117
	08:43	-0.90	λ Vir	H	118-143
			M1-M4 HD 126035	H	144-147
	09:16	-0.43	HD 126035	H	148-152
			M1-M4 HD 126035	H	153-156
			M1-M4 HD 126035	H	157-160
	09:37	0.00	λ Vir	H	161-183
			M1-M4 HD 126035	H	184-187
	10:05	0.40	HD 126035	H	188-191
			M1-M4 HD 126035	H	192-195
	10:13	0.61	λ Vir	H	196-212
			M1-M4 HD 126035	H	213-216
	10:35	0.90	HD 126035	H	217-222
			M1-M4 HD 126035	H	223-228

```
skraus@linux:~/idrs/bin> idrs all
Configuration File: idrs.cfg
Read Entries:
                                      PlotPC = 0
                                      PlotBG = 0
                                     PlotRaw = 0
                                    PlotProc = 0
                                   PlotHisto = 0
                                    PlotWait = 0
                         ReportDetailsScreen = 0
                         VisEstimationMethod = 1
                        VisCalibrationMethod = 1
                            VisSubBackground = 1
                      VisSubBackgroundSmooth = 0.97
                    VisPhotometricCorrection = 1
             VisPhotometricCorrectionSmooth1 = 0.9
             VisPhotometricCorrectionSmooth2 = 0.9
                 VisErrorAddCalibrationError = 0
                              VisErrorAddSys = 0.01
                                VisFilterRaw = 0
                       VisFilterRawThreshold = 6000
                   VisFilterScatteringGroups = 0
          VisFilterScatteringGroupsThreshold = 1.02
               VisFilterLargeErrorsThreshold = 1.0
                   VisFringeTrackerThreshold = 1.0
                           CWTPowerThreshold = 0.2
                            CWTSubBackground = 0
                         CWTBackgroundBorder = 25
                         CWTThresholdAreaMin = 150
                         CWTThresholdAreaMax = 1000
                      CWTThresholdIntPeakOPD = 20
                    CWTThresholdIntPeakScale = 35
             CWTThresholdSizePeakScaleCh0Ch1 = 100
                CWTThresholdSizePeakScaleCh2 = 100
                    CWTThresholdSignal2Noise = 0.0
                         CPCalibrationMethod = 1
                               CPWindowWidth = 32
                               CPErrorAddSys = 2.0
                CPFilterLargeErrorsThreshold = 120.0

Processing Inifile: idrs.ini
*** PROCESSING THE MATRIX FILES ***
Skip writing raw data (no file '/home/user/idrs/testdata.proc/idrs_raw.ini')


=============================================================
          NEW OBJECT: hd126035 -> ID #1
=============================================================

Please specify object type: TARGET ('T'), CALIBRATOR ('C') or RESOLVED CALIBRATOR ('R')?
Type 'T', 'C' or 'R' and press ENTER:
```

Now you have to specify that the star HD 126035 (this is the name you specified during data acquisition) shall be used as a calibrator star (press 'C' for unresolved calibrators or

'R' for calibrators with known diameter). Several lines later, you should specify that λ Vir shall be considered as science target. The data processing continues with the processing of the matrix files which were found within the data set.

```
*** PROCESSING MATRIXFILE 1: /home/user/idrs/testdata/2003-03-24/iota77.data (50 Scans/ch.)
   Number of used scans: 50 50 50 50 50 50 (File /home/user/idrs/testdata.proc/matrix77.m1)

*** PROCESSING MATRIXFILE 2: /home/user/idrs/testdata/2003-03-24/iota78.data (50 Scans/ch.)
   Number of used scans: 50 50 50 50 50 50 (File /home/user/idrs/testdata.proc/matrix78.m2)

*** PROCESSING MATRIXFILE 3: /home/user/idrs/testdata/2003-03-24/iota79.data (50 Scans/ch.)
   Number of used scans: 50 50 50 50 50 50 (File /home/user/idrs/testdata.proc/matrix79.m3)

*** PROCESSING MATRIXFILE 4: /home/user/idrs/testdata/2003-03-24/iota80.data (50 Scans/ch.)
   Number of used scans: 50 50 50 50 50 50 (File /home/user/idrs/testdata.proc/matrix80.m4)
Read matrixfile: /home/user/idrs/testdata.proc/matrixavg77.m1
Read matrixfile: /home/user/idrs/testdata.proc/matrixavg78.m2
Read matrixfile: /home/user/idrs/testdata.proc/matrixavg79.m3
Read matrixfile: /home/user/idrs/testdata.proc/matrixavg80.m4
   Matrix1: (1.839843 2.867187)    -42.289062  -59.078125     -48.699218  -34.851562
   Matrix2:    -45.488281  -35.363281  (2.121094 0.957031)    -40.824218  -50.167969
   Matrix3:    -21.058594  -29.945312     -30.867188  -20.781250  (0.597657 1.234375)
```

Following this, the raw visibilities and closure phases are computed

```
***  PROCESSING THE DATA FILES  ***

*** PROCESSING FILE: /home/user/idrs/testdata/2003-03-24/iota73.data (=#73; 200 Scans/ch.)
Read Matrix: /home/user/idrs/testdata.proc/phot80.dat
Channel 0:     Signal:    2.6020       - Peak at pixel #57   (38 scans used)
          Background:        0.0271    - left: 0.023387, right: 0.034822
        Fringe Power:        2.6020    - SNR:        3.20
Channel 1:     Signal:  11.2174        - Peak at pixel #30   (45 scans used)
          Background:        0.0680    - left: 0.061782, right: 0.073572
        Fringe Power:       11.2174    - SNR:        8.25
Channel 2:     Signal:    6.6509       - Peak at pixel #30   (44 scans used)
          Background:        0.0622    - left: 0.054695, right: 0.072680
        Fringe Power:        6.6509    - SNR:        5.35

...

***  COMPUTING CLOSURE PHASES  ***

*** PROCESSING FILE: /home/user/idrs/testdata/2003-03-24/iota73.data (=#73; 200 Scans/ch.)
Read Matrix: /home/user/idrs/testdata.proc/phot80.dat
Baseline-Def.: B-C A-C B-A: Sign convention -phi2  -phi1
 -Raw Closure Phase (111 phases used): Mean:     -38.22 deg.      Variance:      19.61 deg.

*** PROCESSING FILE: /home/user/idrs/testdata/2003-03-24/iota74.data (=#74; 200 Scans/ch.)
Read Matrix: /home/user/idrs/testdata.proc/phot80.dat
Baseline-Def.: B-C A-C B-A: Sign convention -phi2  -phi1
 -Raw Closure Phase (130 phases used): Mean:     -39.81 deg.      Variance:      20.55 deg.

...
```

```
...

Averaged Fringe Power of the calibrators during each night:
   Night #1:       2.72554        10.81530        5.80218

Scattering of the Fringe Power of the calibrators during each night:

   Night #1:       0.18914         0.30049        0.32540
Night 1: CP calibrator #0: -39.434596
Night 1: CP calibrator #1: -40.152667
Night 1: CP calibrator #2: -40.627256
Night 1: CP calibrator #3: -41.198356
Night 1: CP calibrator #4: -41.135279

Night 1: Subtract closure phase offset using the mean calibrator CP of -40.509633 deg
         (Scattering: 0.657465 DEG).

*** Filter groups with high scattering in V^2 or CP.
Group #  0 (ID  1, JD 222.81) scatters with 0.000020 0.000000 0.000048 from 73 to 81
Group #  1 (ID  1, JD 222.85) scatters with 0.000014 0.000035 0.000445 from 109 to 118
Group #  2 (ID  1, JD 222.89) scatters with 0.000078 0.000000 0.000091 from 148 to 161
Group #  3 (ID  1, JD 222.92) scatters with 0.000121 0.000449 0.000300 from 188 to 196
Group #  4 (ID  1, JD 222.94) scatters with 0.000019 0.000011 0.000834 from 217 to 222
Binning data...

*** Writing OI-FITS File !/home/user/idrs/testdata.proc/bispec.fits:
  - Found 1 telescope configuration(s):
         IOTA_ne35se15ne10
  - Found 3 target(s):
         12Del_Crt (ID 1)
         hd126035 (ID 2)
         LamVir (ID 3)
  - Found 1 spectral channel(s):
         IONIC3_1.650000  Wavel. 1.650000  Bandw. 0.300000

  Configuration: 2003-03-24   IOTA_ne35se15ne10   IONIC3_1.650000
     Writing 84 Vis2 measurement(s)

  Configuration: 2003-03-24   IOTA_ne35se15ne10   IONIC3_1.650000
     Writing 87 Vis2 measurement(s)

  Configuration: 2003-03-24   IOTA_ne35se15ne10   IONIC3_1.650000
     Writing 86 Vis2 measurement(s)

  Configuration: 2003-03-24   IOTA_ne35se15ne10   IONIC3_1.650000
     Writing 87 CP measurement(s)
```

The file `$IDRSROOT/testdata.proc/bispec.fits` will contain the calibrated visibilities of the calibrator and the target in the oifits format. To find these values for the target separated, please use the file `$IDRSROOT/testdata.proc/bispec_maintarget.fits`.

To have a first glance at the data, you may also use IDRSplot by typing
`idrsplot`

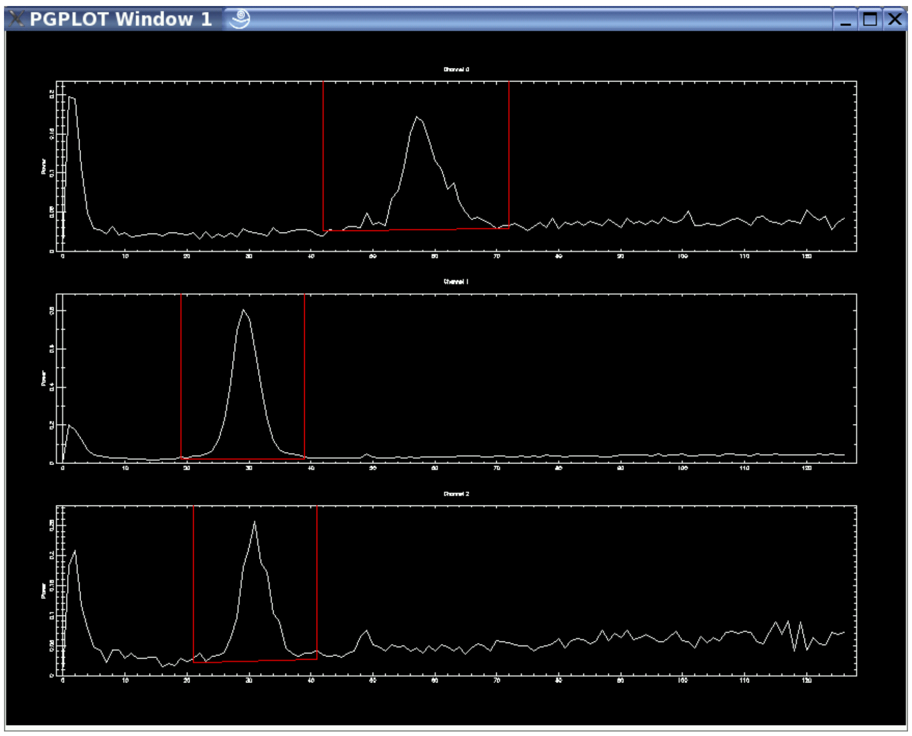

Fig. 5.— IDRS-Screenshot, which can be activated by setting the parameter `PlotProc` in [ConfigFile] to 1.

If you like to continue experimenting with the testdata, you might want to change some parameters in the configuration file `idrs.cfg`. For example, by setting the value
`VisEstimationMethod 0`
the method used for visibility estimation can be switched from the power spectrum (value 1) to the wavelet transform (value 0) method. To accelerate the computation, you might also turn off the visualisation window by setting
`PlotProc 0`
To check the proper function of the background subtraction, you might want to set
`PlotBG 1`

In the next chapters, you will find a more detailed description of the available options.

6.2. Command Line Parameters

The program runs through a cycle, which consists of the visibility estimation, the closure phase estimation, and the final calibration of the reduced data followed by the filtering of bad scans.

The general command line to start IDRS are:
`idrs [option] -i [IniFile] -c [ConfigFile]`

If `-i` or `-c` are not set, the default settings `[IniFile]=idrs.ini` and `[ConfigFile]=idrs.cfg` are used.

With `[option]=all`, the whole data reduction cycle is performed. Optionally, other values may be used for [option] to skip some of these steps, assuming that the earlier cycles have already been processed:

`all`: Perform all data reduction steps.

`vis`: Starts the cycle with the visibility estimation.

`cp`: Starts the cycle with the closure phase estimation.

`final`: Performs only the final calibration and filtering of the reduced data.

6.3. Configuration Files

To use IDRS, the user must generate a text file which lists the location of the IOTA data files. The following sample initiation file `mar03.ini` may be used as an example:

```
ProcDir: /home/user/idrs/Data/Results/
Dir: /home/user/idrs/Data/2003Mar23/  Files: All       Scans: All
Dir: /home/user/idrs/Data/2003Mar24/  Files: 70 to 80  Scans: 50 to 100
```

Using this file, IDRS will process **all data files** in the directory `/home/user/idrs/Data/2003Mar23/`, and scans 50 to 100 within the files `iota70.data` to `iota80.data` in the directory `/home/user/idrs/Data/2003Mar24/`. The directory [ProcDir], which is specified in the first line of the file, denotes where temporary files and also the final results shall be saved. In case the directory does not exist already, it is created by IDRS.

Creating such an initiation file should already be sufficient for reducing interferograms with IDRS.

Since the software needs to know in addition which objects are target and which are calibrators, two supplementary files are needed. These files can be produced automatically while the program is running. In this case, the software will ask as soon as a non-specified object appears whether this should be used as target or calibrator star (type `'T'` or `'C'`).

Alternatively, the user may already specify this information in advance to ensure that the software runs without interruption: For this purpose, the configuration files `objid.ini` and `calib.ini` may be created in [ProcDir]. First, `objid.ini` correlates the object designations in the IOTA header files with the IDRS intern pure numerical object ID, for example

```
HD126035 -> ID #1
DelAur -> ID #2
BetCMi -> ID #3
LamVir -> ID #4
```

Then the file `calib.ini` specifies which of these objects serve as calibrators, to determine the transfer function:

```
Calibrators: 1 2
```

If it is known a priori that some of the calibrators are resolved by the interferometer, the stellar diameters may optionally be given within this file. Then, IDRS corrects the measured visibilities using the uniform-disk diameters given within `calib.ini` before calculating the transfer function:

```
Calibrators:                      1  2  3
ResolvedCalibrators:              2  3
ResolvedCalibratorsDiameters:     2.48  0.79
```

In this example, we assumed calibrator diameters of $D_{\delta Aur} = 2.48$ mas and $D_{\beta CMi} = 0.79$ mas.

6.4. Parameter Fine-tuning

The performance of the software may be optimized by fine-tuning various parameters in the configuration file [ConfigFile]. A case in which these adjustments may be required is when the bandwidth filters used are changed.

The parameters are arranged in four groups: General, Visibility Estimation, Vis: Continuous Wavelet Transform, Closure Phase Estimation.

A brief description of the specific function of all parameters is given in the default configuration file `idrs.cfg` and can be also found in Tables 3 to 7.

Table 3: Parameters in [ConfigFile]: Section **"General"**

Parameter	Valid values	Function
`PlotPC`	0, 1	Plot Photometric Correction 0=No Plot, 1=Plot
`PlotBG`	0, 1	Plot Background Subtraction 0=No Plot, 1=Plot
`PlotRaw`	0, 1	Plot Raw Scans and Power Spectra 0=No Plot, 1=Plot
`PlotProc`	0, 1	Plot Wavelet or Averaged Power Spectrum 0=No Plot, 1=Plot
`PlotHisto`	0, 1	Plot Histogram of Individual Raw Visibility Measurements (not in all modes) 0=No Plot, 1=Plot
`PlotWait`	0, 1	Makes the program hold after finishing each plot 0=Do not wait, 1=Wait
`ReportDetailsScreen`	0, 1	writes a lot of internal details on the screen 0=No Report, 1=Verbatim Mode

Table 4: Parameters in [ConfigFile]: Section **"Visibility Estimation"**

Parameter	Valid values	Function
`VisEstimationMethod`	0,1	Method to estimate visibility 0=wavelet spectral density spectrum 1=averaged power spectrum
`VisCalibrationMethod`	0,1	Method to determine calibrator level 0=average level for each night 1=interpolate between neighboring calibrator stars
`VisSubBackground`	0.0 ... 1.0	Subtract background from the raw data 0=Skip subtraction, 1=Perform subtraction
`VisSubBackgroundSmooth`	0.0 ... 1.0	Smoothing factor for the background signal 0.0=do not smooth, ..., 1.0=smooth strongly
`VisPhotometricCorrection`	0, 1	Applies Photometric Correction 0=No Corr., 1=Apply Corr.
`VisPhotometricCorrectionSmooth1`	0.0 ... 1.0	Smoothing factor for raw signal to determine the photometric signal 0.0=do not smooth, ..., 1.0=smooth strongly
`VisPhotometricCorrectionSmooth2`	0.0 ... 1.0	Smoothing factor for photometric signal to normalize interferogram 0.0=do not smooth, ..., 1.0=smooth strongly
`VisErrorAddCalibrationError`	0, 1	adds calibration error $(\Delta V^2)_{calib}$ (from the scattering of the calibrators) to the statistical error
`VisErrorAddSys`	0.0 ... 1.0	adds constant systematic error $(\Delta V^2)_{sys}$ e.g. 0.03 = 3%
`VisFilterRaw`	0, 1	filters all measurements with a squared raw visibility μ^2 above some threshold
`VisFilterRawThreshold`	> 0.0	Threshold for `VisFilterRaw` e.g. 4000.0

Table 5: Parameters in [ConfigFile]: Section **“Visibility Estimation”**

Parameter	Valid values	Function
`VisFilterScatteringGroups`	0, 1	filters all target-groups with a scattering in the squared raw visibility above some threshold
`VisFilterScatteringGroupsThreshold`	> 0.0	Threshold for `VisFilterScatteringGroups` e.g. 0.1 = 10%
`VisFilterLargeErrorsThreshold`	> 0.0	filters all measurements with an statistical error $(\Delta V^2)_{stat}$ above this threshold; e.g. 0.2 = 20%
`VisFringeTrackerThreshold`	$0.0 \ldots 1.0$	determines from which threshold the AMES-FringeTracker accepts fringe detection e.g. 0.4, $\longrightarrow$ Rejection Code -9

Table 6: Parameters in [ConfigFile]: Section **“Vis: Power Spectrum”**

Parameter	Valid values	Function
`PSBackgroundBorder`	≥ 0	Number of pixels border between the signal and noise-background region within the Power Spectrum

Table 7: Parameters in [ConfigFile]: Section **"Vis: Continuous Wavelet Transform"**

Parameter	Valid values	Function
`CWTPowerThreshold`	$> 0.0 \ldots 1.0$	Significance level to cut the fringe region within the wavelet power spectrum (scaled to the peak in $\mathcal{P}_W$) e.g. 0.35 = 35%
`CWTSubBackground`	0, 1	perform background subtraction 0=Skip subtraction, 1=Perform subtraction
`CWTBackgroundBorder`	≥ 0	Number of pixels border between the signal and noise-background region within the CWT e.g. 10
`CWTThresholdAreaMin`	≥ 0	Minimum number of area pixels used in the CWT $\longrightarrow$ Rejection Code -1 e.g. 150
`CWTThresholdAreaMax`	≥ 0	Maximum number of area pixels used in the CWT $\longrightarrow$ Rejection Code -1 e.g. 800 (> `CWTThresholdAreaMin`)
`CWTThresholdIntPeakOPD`	≥ 0	The fringe peak in the CWT (OPD axis) must be this number of pixels away from the border $\longrightarrow$ Rejection Code -2 e.g. 20
`CWTThresholdIntPeakScale`	> 0	The fringe peak in the CWT (scale axis) must be this number of pixels away from the border $\longrightarrow$ Rejection Code -4 e.g. 35
`CWTThresholdSizePeakScaleCh0Ch1`	> 0	Extension of the fringe peak in the CWT must be below this number for Channel 0, 1=Baseline BC, AC $\longrightarrow$ Rejection Code -3 e.g. 60
`CWTThresholdSizePeakScaleCh2`	> 0	Extension of the fringe peak in the CWT must be below this number for Channel 2=Baseline BA $\longrightarrow$ Rejection Code -3 e.g. 30
`CWTThresholdSignal2Noise`	$0.0 \ldots 1.0$	If the power within the CWT noise region gets too strong, reject $\longrightarrow$ Rejection Code -5 e.g. 30 (= reject if SNR < 30)

Table 8: Parameters in [ConfigFile]: Section **"Closure-Phase Estimation"**

Parameter	Valid values	Function
`CPErrorAddSys`	0.0 . . . 180.0	adds constant systematic error [degree] e.g. 2.0
`CPWindowWidth`	0 . . . 128	width of the window set around the position of the fringe to determine the phase within the coherence time of the atmosphere
`CPFilterLargeErrorsThreshold`	0.0 . . . 360.0	filters all CP measurements with an error above this [degree] e.g. 60.0
`CPFreqClosureThreshold`	0 . . . 256	filters all CP measurements for which the measured fringe frequencies do not close within the given border [pixel] e.g. 0 (increase to weaken criterium)

6.5. The Output Files

The calibrated squared visibilities and closure phases, are stored in two formats. The file `bispec.fits` follows the OI-FITS data exchange format as defined by the IAU Working Group. Besides some special software which is available to process OI-FITS files (e.g. *mfit*, *OYSTER*), the data tables may also be inspected using the fits viewer *fv*.

In addition, the data is given as an ASCII table (`bispec.dat`). From the large number of columns, the following ones are most relevant to the user:

Column 1: Row number (integer)
Column 2: MJD = JD-2452500
Column 3: Object ID (integer)
Column 4: V^2_{AB}, Baseline AB
Column 5: Error V^2_{AB}
Column 7: V^2_{AC}, Baseline AC
Column 8: Error V^2_{AC}
Column 10: V^2_{BC}, Baseline BC
Column 11: Error V^2_{BC}
Column 13: u, Baseline AB [cm]
Column 14: v, Baseline AB [cm]
Column 15: u, Baseline AC [cm]
Column 16: v, Baseline AC [cm]
Column 17: u, Baseline BC [cm]
Column 18: v, Baseline BC [cm]
Column 19: Closure Phase Φ [rad]
Column 20: Error Φ [rad]

All values $V^2 < 0$ and $\Phi < -\pi$ must be considered as invalid. If not otherwise noted, the values are floating numbers. The same data format is also used for the file `amp.dat`, which contains the unfiltered, uncalibrated raw visibilities and closure phases including the instrumental phase offset. The files `bispec.dat` and `amp.dat` can be plotted using the tool IDRSPlot.

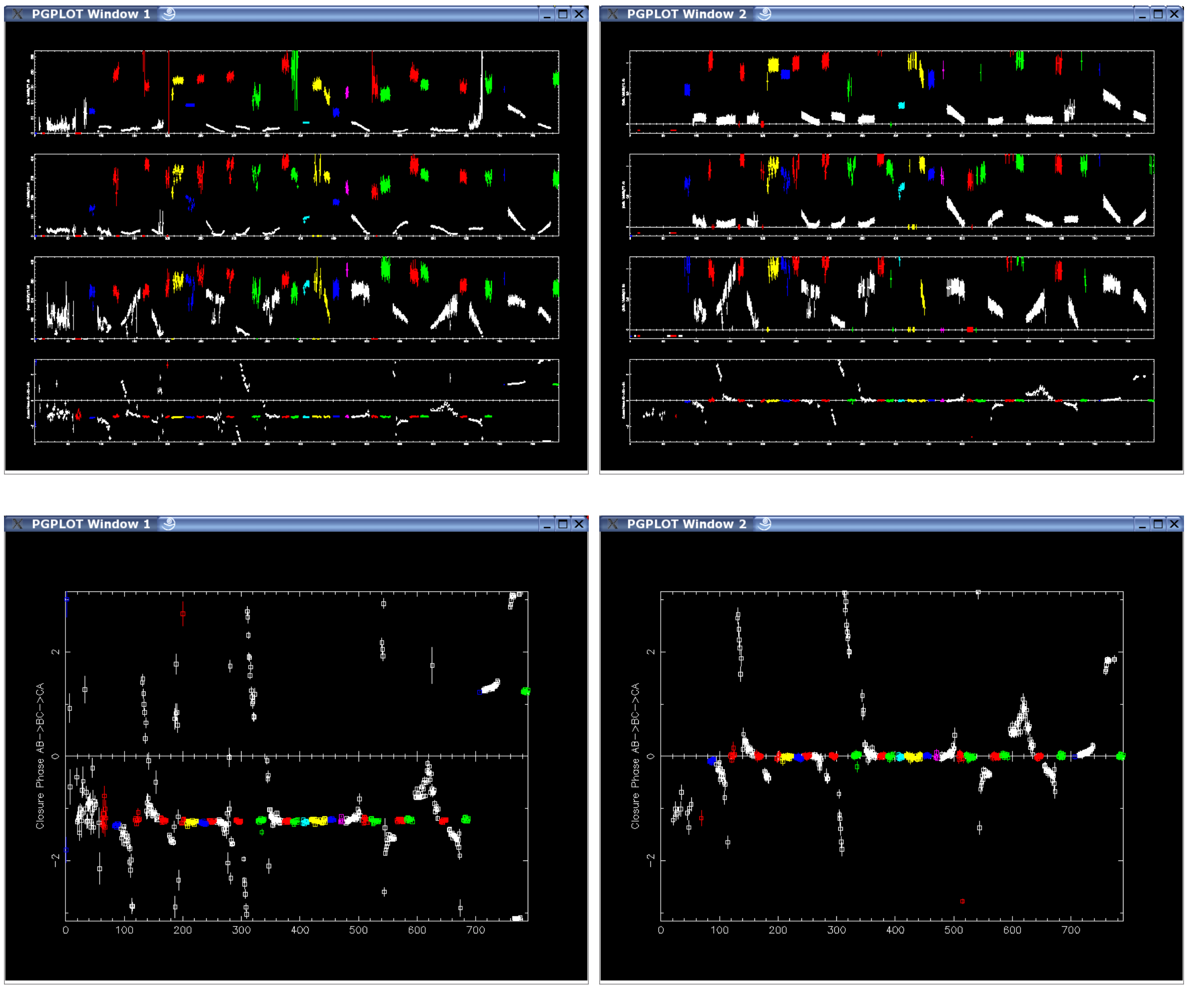

Fig. 6.— Plotting raw visibilities, calibrated visibilities, and closure phases with IDRSPlot.

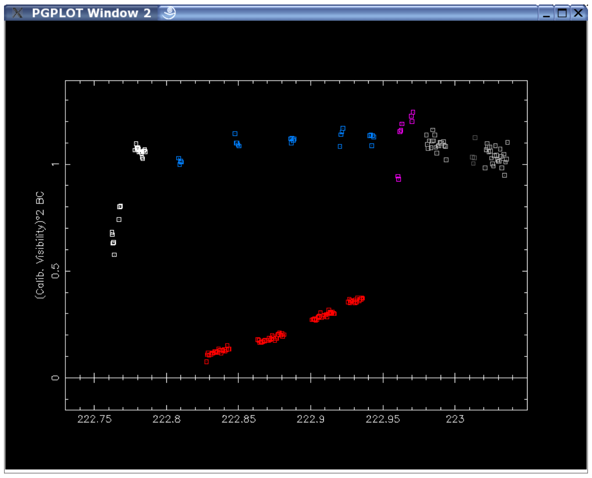

Fig. 7.— IDRSPlot can also display the channels separately. In this plot, the julian date is plotted on the abscissa instead of the file number.

7. IDRSPlot: A Plotting Tool

To plot the visibilities and closure phases stored by IDRS, a small tool, IDRSPlot, is provided. Using PGPLOT, this tool will plot V^2 for all three baselines and Φ on an X-window or any other device.

The simplest way to run this plotting tool is to type
`idrsplot`
within the [ProcDir] you want to investigate. There are several optional arguments, from which the most noteworthy are (to obtain a more complete list, type `idrsplot -help`):

-dev (pgplot_device): Allows to chose between all PGPLOT-device installed (default: "/xwindow", to generate e.g. a postscript file, use "file.ps/CPS").

-wait (sec): Defines, with which interval idrsplot should refresh the plot.

-onlyraw: Plots only the raw visibilities and the uncalibrated closure phase.

-onlycalib: Plots only the calibrated visibilities and closure phase.

-iraw (filename), -icalib (filename), -itrans (filename): Allows to specify manually, which files are shown by `idrsplot`. By default, `-iraw` is set to `ampcorr.dat`, `-icalib` is set to `bispec.dat`, and `-itrans` is set to `transf.dat` within the current directory.

By default, the software opens two windows, the first one showing uncalibrated raw visibilities and closure phases including error bars together with the transfer function computed by `idrs`. This plot allows the user to check whether the obtained transfer function (plotted in the same color as the target object) looks reasonable and might reflect whether the used calibrator stars and the parameters choosen in [ConfigFile] (e.g. `VisCalibrationMethod`) are reasonable. The second window displays the final, calibrated observables V^2 and Φ.

Based on the object ID (specified in `objid.ini`), different objects will be plotted with different colors. Since the number of colors available is limited by PGPLOT, there is no guarantee that for a large number of objects (normally > 15) the colors will be unique.

The plot options can be changed interactively by just typing one of the following characters:

'a': Plot all channels (V_{AB}^2, V_{AC}^2, V_{BC}^2, Φ)

'1': Plot only V_{AB}^2

'2': Plot only V_{AC}^2

'3': Plot only V_{BC}^2

'c': Plot only Φ

'j': Switch abscissa quantity between julian date and file number

'l': Change the abscissa limits manually

'q': Quit program

8. IDRSObs: Real-Time Data Reduction

To have a quick look at the data already during the data acquisition process, the shell script IDRSObs may be used with the command line
`./idrsobs.sh [option] [DataDir]`

[DataDir] is the directory where the data acquisition program writes the data files, and [option] may be "`start`" or an integer number giving the file number from which the script should start processing the data files.
So, the script may be started with
`./idrsobs.sh start /home/user/acquisition/data/`

To ensure that the script will not disturb the data acquisition process, not all data files are processed at once: assuming that $N = 100$ data files exist in [DataDir] (starting from `iota1.data` to `iota100.data`), IDRSObs will process $N-1$ files and wait with the reduction of `iota100.data` until a file with number $N+1$ (`iota101.data`) exists.

After starting the script, no further user interaction is required. Parallel to the data reduction process, the script starts IDRSPlot to display the extracted visibilities and closure phases on the screen.

To quit the script, the user can press ENTER. In this case, the script will process the last file in the directory and exit to the command prompt.

9. General Remarks and FAQs

In order to validate the obtained results we strongly recommend...

– to check carefully whether the background subtraction works properly (use `PlotBG`, maybe combined with `PlotWait`) for the given data. In case the background subtraction does not look optimal, it is abvisable to vary the parameters listed in Tables 6 and 7.

– to check carefully whether the photometric correction works properly (use `PlotPC`, maybe combined with `PlotWait`).

– to try various methods for background subtraction (change `VisSubBackground` and `VisSubBackgroundSmooth`) in order to examine, how strong the obtained results depend on such details.

- to try various values for the strength of smoothing applied to the data for the photometric correction (change `VisPhotometricCorrectionSmooth1` and `VisPhotometricCorrectionSmooth2`).
- to compare the results obtained with the power spectrum (`VisEstimationMethod=1`) and wavelet transform (`VisEstimationMethod=0`) data reduction procedures.
- to check how strong the visibilities/phases obtained for various calibrators observed during the same night vary by using the plotting tool `idrsplot`.
- to compare the results obtained with different calibration procedures (change `VisCalibrationMethod`).
- to check for very faint objects, whether using the preselection procedure which removes interferograms without fringe (`VisFringeTrackerSelection`) introduces biases. In these cases we recommend to vary the detection threshold (`VisFringeTrackerThreshold`) or to disable this feature.

Acquisition of one (or all) matrix files failed, but was repeated. How does the software manages this?

In the first reduction step, IDRS processes all matrix files which are located in the data directory. In case, the flux ratio between channels containing light and shuttered channels is suspicious low (set by the value `matrix_threshold` in the file `idrs.c`), scans in the matrix files are automatically rejected (see output "Number of used scans:" on the screen). In case an alternative matrix file is available, this will be used automatically.

Due to technical problems, fringes could only be acquired on two baselines. Can the data still be reduced?

Without simultaneous acquisition of fringes on all baselines, the photometric information cannot be reconstructed. Nevertheless, data reduction is possible by disabling the photometric correction (`VisPhotometricCorrection=0`). Especially for faint sources, the errors introduced by skipping the photometric correction might be not neglectable.

REFERENCES

Alonso, A., Arribas, S., & Martínez-Roger, C., 1999, *A&AS* 140, 261

Coudé du Foresto, V., Ridgway, S., Mariotti, J. M.: 1997, *å* 121, 379

Flannery, B., et al., 1992, "Numerical Recipes in C", Cambridge University Press

Kraus, S., 2003, Master Thesis, University of Massachusetts at Amherst

Kraus, S., Schloerb, F. P., Traub, W. A., Carleton, N. P., Lacasse, M., Pearlman, M., Monnier, J. D., Millan-Gabet, R., Berger, J.-P., Haguenauer, P., Perraut, K., Kern, P., Malbet, F. and Labeye, P.: 2005, *AJ* 130, 246

Monnier, J. D.: 2001, *PASP* 113, 639

Monnier, J. D., Traub, W. A., Schloerb, F. P., Millan-Gabet, R., Berger, J.-P., Pedretti, E., Carleton, N. P., Kraus, S., Lacasse, M. G., Brewer, M., Ragland, S., Ahearn, A., Coldwell, C., Haguenauer, P., Kern, P., Labeye, P., Lagny, L., Malbet, F., Malin, D., Maymounkov, P., Morel, S., Papaliolios, C., Perraut, K., Pearlman, M., Porro, I. L., Schanen, I., Souccar, K., Torres, G. and Wallace, G.: 2004, *ApJ* 602, L57

Torrence, C., Compo, G. P., 1998, "A Practical Guide to Wavelet Analysis", American Meteorological Society

Wilson, E., 2002, "IOTA-Ames Fringe Tracking Status"

This preprint was prepared with the AAS LaTeX macros v5.2.