Recent developments in optical interferometry data standards

John S. Young, William D. Cotton, Wolfgang Gässler, Rafael Millan-Gabet, John D. Monnier, et al.

John S. Young, William D. Cotton, Wolfgang Gässler, Rafael Millan-Gabet, John D. Monnier, Thomas A. Pauls, Isabelle Percheron, "Recent developments in optical interferometry data standards," Proc. SPIE 7013, Optical and Infrared Interferometry, 70134H (28 July 2008); doi: 10.1117/12.788955

Event: SPIE Astronomical Telescopes + Instrumentation, 2008, Marseille, France

Recent developments in optical interferometry data standards

John S. Young[a], William D. Cotton[b], Wolfgang Gässler[c], Rafael Millan-Gabet[d], John D. Monnier[e], Thomas A. Pauls[f] and Isabelle Percheron[g]

[a]Astrophysics Group, Cavendish Laboratory, JJ Thomson Avenue, CB3 0HE, UK;
[b]NRAO, 520 Edgemont Road, Charlottesville, VA USA 22903;
[c] Max-Planck-Institut für Astronomie, Königstuhl 17, D-69117 Heidelberg, Germany;
[d] Michelson Science Center, California Institute of Technology, 770 South Wilson Avenue, Pasadena, CA USA 91125;
[e]University of Michigan, Department of Astronomy, Ann Arbor, MI USA 48109
[f]Naval Research Laboratory, Code 7210, 4555 Overlook Avenue SW, Washington, DC USA 20375-5351;
[g] ESO, Karl-Schwarzschild-Strasse 2, D-85748 Garching bei München, Germany

ABSTRACT

A working group on interferometry data standards has been established within IAU Commission 54 (Optical/Infrared Interferometry). The working group includes members representing the major optical interferometry projects worldwide, and aims to enhance existing standards and develop new ones to satisfy the broad interests of the optical interferometry community. We present the initial work of the group to enhance the OIFITS data exchange standard, and outline the software packages and libraries now available which implement the standard.

Keywords: data formats, data exchange, FITS, OIFITS, OI-FITS, interferometry

1. INTRODUCTION

The OI Exchange Format ("OIFITS" for short)[1,2] is a data interchange standard for optical/infrared interferometry. The exchange format is a registered convention (see `http://fits.gsfc.nasa.gov/registry/oifits.html`) of the Flexible Image Transport System (FITS).[3] The standard has two main functions: (i) to facilitate combining data from multiple interferometers (see Ref. 4 for a recent example), and (ii) to promote the development of software packages that can work with reduced, calibrated data from any interferometer.

Current interferometer projects supporting the standard include CHARA, NPOI, VLTI, PTI, and the Keck Interferometer.

The OIFITS standard was developed under the auspices of the IAU working group on optical interferometry which was in existence from 2001 to 2006. In 2006 IAU Commission 54 on Optical/Infrared Interferometry was created, within which a working group on interferometry data standards was formed, with a primary mission to maintain and enhance the existing standard for the benefit of the community. The membership of the working group (which is identical to the author list of this paper) includes representatives from all of the major optical/IR interferometer projects worldwide.

In this paper, we use "optical interferometry" to mean interferometry at both optical and near/mid-infrared wavelengths.

Further author information: (Send correspondence to J.S.Y.)
J.S.Y.: E-mail: jsy1001@cam.ac.uk, Telephone: 44 1223 337365

Optical and Infrared Interferometry, edited by Markus Schöller, William C. Danchi, Françoise Delplancke
Proc. of SPIE Vol. 7013, 70134H, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.788955

Table 1. Summary of FITS binary tables defined by the OI Exchange Format (OIFITS). The content of the tables is defined the OIFITS format definition, Ref. 1. The optional keyword "ARRNAME" in a data table identifies the OI_ARRAY table giving the coordinates of the telescope stations used to obtain the data. Similarly the compulsory "INSNAME" keyword specifies the OI_WAVELENGTH table containing the wavebands for the interferometric data. All data tables refer to entries within a single OI_TARGET table.

Table	No. present	Cross-references	Content
OI_ARRAY	0+	-	Telescope coordinates
OI_TARGET	1	-	Target coordinates
OI_WAVELENGTH	1+	-	Observation wavebands
OI_VIS, OI_VIS2, OI_T3	1+	→ OI_TARGET, ARRNAME (opt.) → OI_ARRAY, INSNAME → OI_WAVELENGTH	Complex V, V^2, bispectra

2. CURRENT OIFITS FEATURES

In this section we briefly summarise the main features of the current OI Exchange standard. The reader is referred to Ref. 1 for the formal definition of the standard. OIFITS files contain several of the tables listed in Table 1, each being a FITS binary table containing a set of keywords and data columns which are specified by the OI standard.

From a functional point of view, the main features of OIFITS are:

- All commonly-used observables in optical interferometry are supported:
  - Complex visibility (for phase-referenced or differential phase data)
  - Squared visibility
  - Bispectrum (triple product amplitude and closure phase)
- Multiple data tables of each type permitted; facilitates combining data from different interferometers
- Compact representation of spectrally-dispersed data
- Definitive *uv*-plane coordinates are kept with the data
- Provision for including the information (array and target coordinates) needed to re-calculate or interpolate between the *uv*-plane coordinates.
- Supports flagging of individual data points

3. ENHANCEMENTS UNDER DISCUSSION

The working group is currently discussing a number of *potential* enhancements to the OIFITS standard. We outline these and provide brief justifications in this paper. We would welcome any further proposals or suggestions of priorities from the community; these may be addressed to any member of the working group.

The working group would like to stress that discussions are still at a preliminary stage. It is likely that only a subset of these proposals will be incorporated into the next revision of the data exchange standard. We do not currently anticipate introducing any changes which are backwards-incompatible.

3.1 Suggestions for new features

These are listed in no particular order.

Support for nulling data Following the successful commissioning of the Keck Nuller,[5] there may be a case for adding table(s) for nuller data to OIFITS, in order to facilitate combining such data with e.g. squared visibility or closure phase measurements from other interferometers.

Calibration tables These would provide information that would allow the data to be recalibrated and/or indicate which errors are correlated. This proposal is discussed in more detail below.

Closure amplitudes Closure amplitude will be a data product of the Michiagan Infrared Combiner[6] on the CHARA array (and in principle can be measured by any interferometer with at least four collectors).

Hour angle Addition of an (optional) hour angle column to the data tables (OI_VIS, OI_VIS2 and OI_T3) would facilitate making plots of the data against hour angle (which is more useful than plotting against Universal Time since data from multiple nights are often combined for analysis). The reader should note that it is not always possible to calculate hour angle from the information given in an OIFITS file, since the OI_ARRAY table is optional.

Ability to flag triple product amplitude and phase independently This would address a shortcoming of the current format, which only allows the entire triple product vector to be flagged, or for a NULL value to be supplied for the amplitude.

Spectrum table Photometrically-calibrated (or uncalibrated) spectra are a common data product of interferometers that use spectral dispersion.

OI_VIS metadata The OI_VIS table stores complex visibilities whose phases may be meaningful in either a relative (for differential phase data) or absolute (for phase-referenced data) sense. Additional keyword(s) that indicate which case applies and the nature of any phase reference would help ensure that the data are used appropriately.

The format of any calibration tables (which would be the equivalent of AIPS SN or CL tables), or extensions to existing tables to include calibration information, could be chosen to satisfy *some or all* of the following use-cases:

- Permit the calibration to be undone without ambiguity (e.g. any interpolation between calibrator observations should be accounted for), so that a completely new calibration could be applied.
- The same calibration to be re-applied, but with a different diameter and error bar for one or more calibrator stars.
- Allow the calibrator observations to be inspected/fit to in order for suspicions about a particular calibrator to be followed up.
- Support calibration of all the OI observables, not just squared visibility.

The working group on calibrator stars within IAU Commission 54, led by Christian Hummel, has mooted the idea that calibrator star selection tools using a defined list of possible calibrators could store the list (including coordinates, diameter, limb-darkening and their uncertainties) in an extended version of the OIFITS OI_TARGET table. By itself, this would not be sufficient for all of the use cases outlined above. We expect these issues to be resolved by discussions between the two working groups.

4. SOFTWARE

Software libraries providing functions to read and write the OIFITS format are available for a number of programming languages. We provide an up-to-date description of these in the following sections.

4.1 OIFITSlib

OIFITSlib is a C library for input/output, merging, filtering and checking of optical/IR interferometry datasets in the OIFITS exchange format.

The table-level input/output code in OIFITSlib is derived from the previously-released "OIFITS example software in C",[2] and provides the same Application Programming Interface (API). OIFITSlib provides a file-level API built on top of the table-level code, containing functions to read and write an entire OIFITS file. The file-level API should be able to read any valid OIFITS file.

Command-line utilities `oifits-merge`, `oifits-filter` and `oifits-check` are also provided - these provide simple user interfaces to OIFITSlib routines.

4.1.1 oimerge

This module implements merging of a list of OIFITS datasets into a single dataset.

To simplify the implementation, OI_ARRAY tables are not copied into the output dataset (these are not required by the OIFITS standard). Target records with the same target name are merged (without checking that the coordinates etc. are identical), as are duplicate OI_WAVELENGTH tables.

4.1.2 oifilter

This module implements a filter for OIFITS data, intended for use in selecting a subset of the data in a OIFITS file for analysis. The following filtering criteria are available:

- ARRNAME value
- INSNAME value
- TARGET_ID value
- MJD range
- Wavelength range
- Projected baseline range
- Signal-to-noise range
- Data type

4.1.3 oicheck

This module provides a set of checks that a OIFITS dataset conforms to the published standard. The list of checks is not exhaustive, but goes beyond the level of conformity needed for the file to be read successfully by OIFITSlib.

Separate functions are provided to check different aspects of the dataset, so that an application can perform only those checks that are relevant to the tasks it performs. Functions are provided to verify that:

- Targets listed in OI_TARGET table have unique identifier strings (TARGET column)
- Entries for all targets mentioned in data tables (OI_VIS, OI_VIS2, OI_T3) are present in the OI_TARGET table
- Entries for all array elements (stations) mentioned in data tables are present in the appropriate OI_ARRAY table, if one is referenced
- All error bars in data tables are positive (some non-conforming files use negative errors to indicate flagged data)
- Triple product amplitude values are normalised (i.e. do not exceed unity within uncertainties)
- Wavelength values within each OI_WAVELENGTH table are in ascending order

4.1.4 Python interface

A python interface to OIFITSlib is also provided (created using SWIG). This module provides full read-only and limited read-write access to OIFITS data, by means of an OiFits class. OiFits instances have a comprehensive set of data attributes (mirroring those in the OIFITSlib C API) plus a few methods.

The python interface to OIFITSlib may be used as an alternative to the native python OIFITS library described in Sec. 4.3. The former provides an interface to the merging, filtering, and checking functionality in OIFITSlib, whereas the latter offers full read-write access to the data, using native python data types.

The OiFits constructor is used to read an OIFITS file from disk:

```
>>> import oifits
>>> o = oifits.OiFits('bigtest2.fits')
```

The resulting OiFits instance may be passed to functions in the oicheck, oimerge, and oifilter modules. Functions in the latter two modules create a new OiFits instance, which may be written to an OIFITS file using the write() method:

```
>>> o.write('junk.oifits')
```

The interface uses various sorts of mappings from the C API. Some involve special types provided by SWIG, while other use native python types. Please refer to the documentation provided with the software for full details.

4.1.5 oimerge

This module provides a simple interface to the OIFITSlib merge functionality. Two or more oifits.OiFits instances must be supplied, and a new OiFits instance containing the merged dataset is returned.

```
>>> import oifits, oimerge
>>> o1 = oifits.OiFits('testdata.fits')
>>> o2 = oifits.OiFits('bigtest2.fits')
>>> merged = oimerge.merge(o1, o2)
```

4.1.6 oicheck

This module provides an interface to the OIFITSlib dataset checking functions. A number of functions check_XXX() are provided, each of which performs one of the available checks (see above).

```
>>> import oifits
>>> from oicheck import *
>>> checks = [check_unique_targets, check_targets_present,
...           check_elements_present, check_flagging, check_t3amp,
...           check_waveorder]
>>> o = oifits.OiFits('bigtest2.fits')
>>> for c in checks:
...     level, result = c(o)
...     if level > 0:
...         print result
```

4.1.7 oifilter

This module provides an object-oriented interface to the OIFITSlib dataset filtering functionality. To filter a oifits.OiFits instance, create an instance of the Filter class, make the appropriate assignments to its data attributes, than pass the OiFits instance to its apply() method (which returns a filtered version of the dataset).

```
    >>> import oifits, oifilter
    >>> o = oifits.OiFits('testdata.fits')
    >>> f = oifilter.Filter()
    >>> f.accept_t3amp = 0
    >>> f.accept_t3phi = 0
    >>> filtered = f.apply(o)
```

4.2 IDL Routines

An IDL library designed to read and write optical interferometry data conforming to the OIFITS standard has been written by J.D.M., and is available from `http://www.astro.lsa.umich.edu/~monnier/oi_data/`.

All features of the OIFITS format are supported in v1.2, including the ability to read and write multiple OI_ARRAY, OI_WAVELENGTH, OI_VIS, OI_VIS2, and OI_T3 binary tables. Merging data from different epochs and arrays is very straightforward, using the supplied `merge_oidata.pro`.

Each of the six allowed tables, OI_ARRAY, OI_TARGET, OI_WAVELENGTH, OI_VIS, OI_VIS2, and OI_T3, is converted from FITS into an IDL structure. While values are stored directly in the structure when possible, pointers must be used in order to accommodate the full standard which allows multiple OI_WAVELENGTH tables.

4.2.1 Example

In order to read in the tables from the `testdata.fits` file which is distributed by J.S.Y. (as part of OIFITSlib, see `http://www.mrao.cam.ac.uk/research/OAS/oi_data/oifits.html`) one merely executes the following IDL command:

```
IDL> READ_OIDATA, 'testdata.fits', oiarray,oitarget,oiwavelength,$
     oivis, oivis2,oit3, /inventory
        This file Satisfies the requirements of the OI_DATA format
        Inventory:
          OI_ARRAY:             1
          OI_TARGET:            1
          OI_WAVELENGTH:        1
          OI_VIS:               1
          OI_VIS2:              1
          OI_T3:                1
         Unknown Tables:        0
IDL> print,oivis2(0).time,*oivis2(0).vis2data,*oivis2(0).vis2err
         82810.000      0.67700000     0.064000000
IDL> print,string(*oivis(0).flag)
           F
```

Writing the data back is just as easy:

```
IDL> WRITE_OIDATA, 'testdata1.fits', oiarray,oitarget,oiwavelength,$
        oivis, oivis2,oit3
```

4.3 Python Library Module

A library coded in the Python language (see `http://www.python.org`) for reading and writing the Exchange Format is available from the OIFITS website. This makes use of the PyFITS library[7] from Space Telescope Science Institute (see `http://www.stsci.edu/resources/software_hardware/pyfits`).

The python library module is structured as a set of python classes, instances of which provide object-oriented access to data from an OIFITS file (which may be created in memory before the corresponding file exists on disc). The classes are as follows:

OI_FITS Instance stores data for entire OIFITS file

> The data attributes of the OI_FITS instance are instances of other classes, or lists/dictionaries of these. Each of the these "table classes" corresponds to one of the FITS binary tables defined in the standard:

Array OI_ARRAY table

Target OI_TARGET table

Wavelength OI_WAVELENGTH table

Vis OI_VIS table

Vis2 OI_VIS2 table

T3 OI_T3 table

The table classes typically have data attributes that are instances of lower-level classes, corresponding to rows of the FITS table. These "row classes" generally don't have any method attributes.

The entire FITS file corresponding to an OI_FITS instance may be read from/written to disk using the instance's FromHDUList() and ToHDUList() methods. These convert from/to a pyfits.HDUList instance (which represents data from a generic FITS file), as defined in the third-party pyfits module. Pyfits provides many functions that can operate on HDUList instances (see the pyfits manual).

A script to write an OIFITS file would be structured as follows:

```
import oi_fits
oi = oi_fits.OI_FITS()
# insert code to assign to data attributes
oi.ToHDUList().writeto(’output.fits’)
```

A script to read from an OIFITS file would be structured like this:

```
import oi_fits, pyfits
hlist = pyfits.open(’input.fits’)
oi = oi_fits.OI_FITS()
oi.FromHDUList(hlist)
hlist.close()
# insert code to do something with oi’s data attributes
```

Instances of table classes have FromTable() and ToTable() methods, which convert between the table class instance and a pyfits.BinTableHDU instance. These methods may be used when reading/writing individual tables, rather than the whole file.

A python script for copying selected tables from one OIFITS file to another, `oicopy`, is available from the OIFITS website. This script serves as an example of how to use the OIFITS python module.

4.4 Software Applications that support OIFITS

In this section, we provide an up-to-date list of freely-available software applications that support the OIFITS format.

4.4.1 Observation Preparation/Data Visualisation

The ASPRO software from JMMC (see `http://www.jmmc.fr/aspro_page.htm`) can export simulated data to an OIFITS file, and can read an OIFITS file and make various plots of the contents.

The equivalent package from MSC, called VMT ("Visibility Modelling Tool" – see `http://mscweb.ipac.caltech.edu/vmt/`) is expected to provide input/output of OIFITS data and a model-fitting capability in a future release.

4.4.2 Model-fitting

Full descriptions of the available packages can be obtained from their own websites:

- mfit - `http://www.mrao.cam.ac.uk/~jsy1001/mfit/`
- OYSTER - `http://www.eso.org/~chummel/oyster/oyster.html`

We expect that the LITpro model-fitting software[8] will be released to the community in future.

4.5 Image reconstruction

Of the image reconstruction codes that entered the two most recent IAU-sponsored imaging "beauty contests",[9,10] the following are now available to the community:

- MACIM[11] - `http://quoll.gps.caltech.edu/MACIM`
- BSMEM[12] - `http://www.mrao.cam.ac.uk/research/OAS/bsmem.html`
- A public release of MIRA is expected soon

REFERENCES

[1] Pauls, T. A., Young, J. S., Cotton, W. D., and Monnier, J. D., "A data exchange standard for optical (visible/IR) interferometry," *PASP* **117**, 1255 (2005).

[2] Pauls, T. A., Young, J. S., Cotton, W. D., and Monnier, J. D., "A data exchange standard for optical (visible/IR) interferometry," in [*New Frontiers in Stellar Interferometry*], Traub, W., Monnier, J. D., and Schöller, M., eds., *Proc. SPIE* **5491**, 1231 (2004).

[3] Hanisch, R. J., Farris, A., Greisen, E. W., Pence, W. D., Schlesinger, B. M., Teuben, P. J., Thompson, R. W., and Warnock, A., "Definition of the Flexible Image Transport System (FITS)," *Astronomy & Astrophysics* **376**, 359–380 (2001).

[4] Tannirkulam, A., Monnier, J. D., Millan-Gabet, R., Harries, T. J., Pedretti, E., ten Brummelaar, T. A., McAlister, H., Turner, N., Sturmann, J., and Sturmann, L., "Strong near-infrared emission interior to the dust sublimation radius of young stellar objects MWC 275 and AB Aurigae," *Astrophys. J.* **677**, L51–L54 (2008).

[5] Colavita, M. M., Serabyn, G., Wizinowich, P. L., and Akeson, R. L., "Nulling at the Keck interferometer," in [*Advances in Stellar Interferometry*], Monnier, J. D., Schöller, M., and Danchi, W. C., eds., *Proc. SPIE* **6268**, 626803 (2006).

[6] Monnier, J. D., Pedretti, E., Thureau, N., Berger, J.-P., Millan-Gabet, R., ten Brummelaar, T., McAlister, H., Sturmann, J., Sturmann, L., Muirhead, P., Tannirkulam, A., Webster, S., and Zhao, M., "Michigan Infrared Combiner (MIRC): commissioning results at the CHARA Array," in [*Advances in Stellar Interferometry*], Monnier, J. D., Schöller, M., and Danchi, W. C., eds., *Proc. SPIE* **6268**, 62681P (2006).

[7] Barrett, P. E. and Bridgman, W. T., "PyFITS, a Python FITS Module," in [*ASP Conf. Ser. 216: Astronomical Data Analysis Software and Systems IX*], 67 (2000).

[8] Tallon-Bosc, I., Tallon, M., Béchet, C., and Thiébaut, E. M., "LITpro: a model-fitting software for optical interferometry," in [*Optical and Infrared Interferometry*], Schöller, M., Danchi, W. C., and Delplancke, F., eds., *Proc. SPIE* **7013** (2008). Paper 7013-54, these proceedings.

[9] Lawson, P. R., Cotton, W., Hummel, C. A., Baron, F., Young, J. S., Meimon, S. C., Besnerais, G. L., Mugnier, L. M., Ireland, M. J., Monnier, J. D., Thibeaut, E. M., Zavala, R. T., Kraus, S., Hofmann, K.-H., Weigelt, G. P., Pott, J., and Hope, D. A., "2006 interferometry imaging beauty contest," in [*Advances in Stellar Interferometry*], Monnier, J. D., Schöller, M., and Danchi, W. C., eds., *Proc. SPIE* **6268**, 62681U (2006).

[10] Cotton, W. D., Monnier, J. D., Lawson, P. R., Zhao, M., Thiébaut, E. M., Hofmann, K.-H., Kraus, S., Weigelt, G. P., Mosoni, L., Baron, F., Young, J. S., Sridharan, R., Jaffe, W. J., Hummel, C. A., Pauls, T. A., Schmitt, H. R., and Tuthill, P. G., "The 2008 interferometry imaging beauty contest," in [*Optical and Infrared Interferometry*], Schöller, M., Danchi, W. C., and Delplancke, F., eds., *Proc. SPIE* **7013** (2008). Paper 7013-58, these proceedings.

[11] Ireland, M. J., Monnier, J. D., and Thureau, N., "Monte-Carlo imaging for optical interferometry," in [*Advances in Stellar Interferometry*], Monnier, J. D., Schöller, M., and Danchi, W. C., eds., *Proc. SPIE* **6268**, 62681T (2006).

[12] Buscher, D. F., "Direct maximum-entropy image reconstruction from the bispectrum," in [*Very High Angular Resolution Imaging*], Robertson, J. G. and Tango, W. J., eds., *IAU Symposium* **158**, 91 (1994).