CHARA Online Database Portal v0.2

Jeremy Jones

August 2021

1 Introduction

I have built an online portal for users to see what data are available in the CHARA Archive and to provide some metadata on these observations.

The metadata available on the database portal are either taken directly from or derived from quantities taken from the headers of the FITS files for data from all beam combiners except for VEGA. In the case of VEGA data, the metadata are taken from the .info files for each observation. All of this header information is stored in an SQLite database file with one table for each beam combiner. The process of extracting the metadata and storing them in the SQLite database is described in Section 2.

Some of the metadata from each beam combiner are taken from the SQLite database, put into standard formats[1], and added to a new table (all_chara) in the SQLite database file. In this step, I also calculate some metadata that is missing for some beam combiners (e.g., PAVO data do not have azimuth, elevation, or hour angle listed in the header information of the FITS data files, so I calculate these based on the target's right ascension and declination, and the date and time of observation). See Section 3 for a description of what attributes are included and if or how they are altered from their original format and if/how they are derived.

A Flask webapp reads the SQLite database file and displays the metadata based on input from the user. This webapp is hosted on a virtual machine on the CHARA Server in Atlanta and is described in Section 4.

All of the software for this project is accessible at `https://gitlab.chara.gsu.edu/jeremyjones/chara_online_database_portal` and the portal itself can currently be accessed at `http://chara.gsu.edu/observers/database` or more directly, at `http://131.96.55.87:8080`.

[1]For example, UT Time is labeled as CCUTTIME in the Classic and CLIMB data with a format of HH MM SS.SSS whereas MIRC data have the UT Time labeled as UTC_OBS with a format of HH:MM:SS.SSS. For the standardized "all_chara" table, I use the label UT_Time with a format of HH:MM:SS.

2 Extracting Metadata

The python script, generate_db.py, located in the "backend" directory of the gitlab repository is used to extract the FITS header information (or its equivalent) from the raw data in the archive. This script creates an SQLite database file with a table for each beam combiner. It then generates a list of files from which to extract the metadata for the database. Based on the attributes in the FITS headers (or .info files for VEGA data), the script adds column names and types to each table. It then populates each table with the metadata in the files listed in the file list.

2.1 File List

The generate_db.py script creates a list of files to use for extracting metadata to the database. For the Classic, CLIMB, FLUOR[2], and JouFLU beam combiners, this file list is simply a list of the data files, because for these beam combiners, one data file is equivalent to one observation. There should only be one entry in the database for each observation. The PAVO, MIRC, MIRC-X, and VEGA beam combiners store their data differently from the other beam combiners, so it is necessary to create separate procedures to designate what data files should be used for metadata extraction.

2.1.1 PAVO File List

PAVO data has $\gtrsim$700 files for each observation. It is unfeasible to read through so many files for header information, so for these data the function generate_db.make_pavo_file_list() reads through all of the PAVO headstrip files in the archive. PAVO headstrip files are files that have extracted a small amount of information from the PAVO data files' headers. Headstrip files are used in the data reduction process. They do not explicitly state when one observation begins and ends, so the make_pavo_file_list() function notes the file numbers of the first file in the data sequence for each observation. This is shown in the headstrip files as when all three shutters were open. The function returns the list of PAVO data files that have the noted file numbers.

2.1.2 MIRC & MIRC-X File Lists

MIRC(-X) data also have a large number of files for each observation ($\sim$100-200 files). The functions make_mirc_file_list() and make_mircx_file_list() read the *.mirclog and *_mircx_log.txt files and note the file numbers of the first file with the data type set to "DATA" (i.e. the beginning of the data sequence) for each observation. These functions return lists of MIRC and MIRC-X data files that have the noted file numbers.

[2]For extracting metadata to be used in the database, I am using the FITS version of the FLUOR data in the archive. These data have been converted from their original ASCII format using the ascii2fits script written by Theo ten Brummelaar.

2.1.3 VEGA File List

For VEGA data, the header information that is relevant for this project is saved in separate files from the raw data. These ASCII *.info files are what the generate_db script reads for extracting the relevant metadata. The key to these info files was provided by Denis Mourard and attribute titles in the database are based on this key.

2.2 Metadata Extraction

For data from beam combiners where each observation corresponds to a single, uncompressed OIFITS file (e.g., Classic, CLIMB, FLUOR, and JouFLU) metadata are extracted using the astropy package's FITS reader to read the headers of the files listed in the file lists for each combiner (see Section 2.1). For data from PAVO, MIRC, and MIRC-X, where data are stored in multiple, compressed files for each observation, I use the procedure described in Sections 2.1.1 and 2.1.2 to determine from which files metadata should be extracted, uncompress those files in a temporary directory, and read the uncompressed files using the astropy FITS reader. VEGA *.info files, in ASCII format, are read using the python inbuilt csv module's reader. The key to these info files was provided by Denis Mourard and attribute titles in the database are based on this key.

2.3 The SQLite Database

When the metadata is extracted, it is stored in an SQLite database with a separate table for each beam combiner.

3 Standardizing Metadata

I standardize and combine the metadata from all different beam combiners into a single table, the All CHARA table, in the SQLite database using the generate_db_summary.py script, located in the "backend" directory of the gitlab repository. Each beam combiner stores useful information in slightly different ways. Section 3.1 lists the attributes that are standardized, describes what each attribute name represents, and how they are standardized for each beam combiner.

3.1 Standardized Attributes

3.1.1 Star_HD

The name of the star (or other object) that is the subject of the observation. Only observations with a non-null star name are saved to the All CHARA database table.

For Classic and CLIMB data, the CCSTARHD attribute is simply renamed to be Star_HD. No values are altered.

For FLUOR data, the FLSTARHD attribute is simply renamed to be Star_HD. No values are altered.

In JouFLU data, the star name is saved in one of three attributes: CCSTARHD, FLSTARHD, or JFSTARHD. For each observation, only one of these three is used, so the star name is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, the HD attribute is simply renamed to be Star_HD. No values are altered.

For PAVO data, the PVSTARHD attribute is simply renamed to be Star_HD. No values are altered.

For VEGA data, the star_name attribute is simply renamed to be Star_HD. No values are altered.

3.1.2 Object_Type

The type of object (e.g., target, calibrator, etc.) being observed.

For Classic and CLIMB data, the CCOBJTYP attribute is renamed to be Object_Type with no values being altered.

For FLUOR data, the FLOBJTYP attribute is renamed to be Object_Type with no values being altered.

In JouFLU data, the object type is saved in one of three attributes: CCOBJTYP, FLOBJTYP, or JFOBJTYP. For each observation, only one of these three is used, so the object type is taken from the attribute that was used for each observation.

No object type is saved in MIRC or PAVO data.

For MIRC-X data, the object type is saved in one of two attributes: OBJTYPE and OBJECT_TYPE. Typically, one of these attributes will be null for a given observation and the other attribute is used. Cases where neither are null are flagged for manual review.

For VEGA data, object type is determined by looking at the info file's name. If 'CAL' is in the name, 'CAL' is returned as object type. If 'D_R' or 'D_B' is in the name, 'SPEC' is returned as object type. Otherwise, the object type is assumed to be 'OBJ'.

3.1.3 UT_Date

The UT Date of the observation. All dates are confirmed to be in the "YYYY-MM-DD" format. If any dates are not in that format, they are converted to that format using the parser.parse() function of the python module, dateutil. Dates which cannot be parsed (typically because there is no actual date entered) are flagged for manual review.

For Classic and CLIMB data, dates are taken from the CCUTDATE attribute.

For FLUOR data, the UT date is not recorded in the FITS version of the data, so it is taken from the file name, which does include it, and put into the appropriate format.

In JouFLU data, the UT date is saved in one of three attributes: CCUTDATE, FLUTDATE, or JFUTDATE. For each observation, only one of these three is used, so the UT date is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, dates are taken from the DATE_OBS attribute.

For PAVO data, dates are taken from the PVUTDATE attribute.

For VEGA data, the UT date is converted from the julian date, provided in the jul_date attribute.

3.1.4 UT_Time

The UT Time of the observation. All times are converted from whatever format they are in to the "HH:MM:SS" format. As with the UT_Date attribute, times which cannot be parsed are flagged for manual review.

For Classic and CLIMB data, times are taken from the CCUTTIME attribute.

For FLUOR data, times are taken from the FLUTTIME attribute.

In JouFLU data, the UT time is saved in one of three attributes: CCUTTIME, FLUTTIME, or JFUTTIME. For each observation, only one of these three is used, so the UT time is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, times are taken from the UTC_OBS attribute.

For PAVO data, times are taken from the PVUTTIME attribute.

For VEGA data, times are taken from the ut_time attribute.

3.1.5 Telescope_Used

There are six attributes (one for each currently active telescope, e.g., S1_Used, S2_Used, etc.) that indicate if the associated telescope was used in a given observation. The values in these attributes is either 0 (telescope not used) or 1 (telescope used).

For Classic data, this is determined by using the CCBEAMS attribute to determine which beams were used and the appropriate CC_BEAMX attributes, which show what telescopes were used on those beams. If CCBEAMS does not list the beams that were used, I assume that beams 5 and 6 are used.

For CLIMB 1 data, this is determined by looking at the CC_BEAM1, CC_BEAM2, and CC_BEAM3 attributes, which show what telescopes were used on those beams.

For CLIMB 2 data, this is determined by looking at the CC_BEAM4, CC_BEAM5, and CC_BEAM6 attributes, which show what telescopes were used on those beams.

For FLUOR data, this is determined by looking at the FL_BEAM5 and FL_BEAM6 attributes, which show what telescopes were used on those beams.

For JouFLU data, this is determined by looking at the CC_BEAM5, CC_BEAM6, FL_BEAM5, FL_BEAM6, JF_BEAM5, and JF_BEAM6 attributes, which show what telescopes were used on those beams.

For MIRC and MIRC-X data, this is determined by looking at the N_TELS, TEL_KEY, and BEAMORDX (X=0,1,2,3,4,5) attributes.

For PAVO data, this is determined by looking at the PV_BEAM1, PV_BEAM2, and PV_BEAM3 attributes, which show what telescopes were used on those beams. There is no easy way to distinguish between PAVO being run in 2-telescope mode vs. 3-telescope mode with only the header information. So the database treats these data as if all three beams are active.

For VEGA data, this is determined by looking at the T1, T2, T3, and T4 attributes, which show what telescopes were used.

3.1.6 Beam_X

There are six attributes (one for each beam, e.g., Beam_1, Beam_2, etc.) that indicate what telescope was active (if any) on each beam.

For Classic data, the Beam_X attribute corresponds to the CC_BEAMX attribute.

For CLIMB 1 data, the Beam_X attribute corresponds to the CC_BEAMX attribute, but only for X = 1,2,3. For beams 4, 5, and 6, which are not used by CLIMB 1, database entries are shown as "None".

For CLIMB 2 data, the Beam_X attribute corresponds to the CC_BEAMX attribute, but only for X = 4,5,6. For beams 1, 2, and 3, which are not used by CLIMB 2, database entries are shown as "None".

For FLUOR data, the Beam_X attribute corresponds to the FL_BEAMX attribute for X = 5 or 6. For beams 1, 2, 3, and 4, which are not used by FLUOR, database entries are shown as "None".

In JouFLU data, the telescope used by a given beam is saved in one of three attributes: CC_BEAMX, FL_BEAMX, or JF_BEAMX for X = 5 or 6. For each observation, only one of these three is used, so the telescope that is used by Beam_X is taken from the attribute that was used for each observation. Database entries for beams 1, 2, 3, and 4, which are not used by JouFLU, are shown as "None".

For MIRC and MIRC-X data, the Beam_X attribute corresponds to the BEAMORDY (where Y = X − 1) attribute.

For PAVO data, the Beam_X attributes corresponds to the PV_BEAMX attribute, but only for X = 1,2,3. For beams 4, 5, and 6, which are not used by PAVO, database entries are shown as "None".

For VEGA data, the Beam_X attribute corresponds to the BEAMX attribute, but only for X = 1,2,3,4. For beams 5 and 6, which are not used by VEGA, database entries are shown as "None".

3.1.7 Azimuth

The Azimuth at which the observation was taken.

For Classic and CLIMB data, the CC_AZ attribute is renamed to be Azimuth with no values being altered.

For FLUOR data, the FL_AZ attribute is renamed to be Azimuth with no values being altered.

In JouFLU data, the azimuth is saved in one of three attributes: CC_AZ, FL_AZ, or JF_AZ. For each observation, only one of these three is used, so the azimuth is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, the AZ attribute is renamed to be Azimuth with no values being altered.

PAVO data do not store azimuth information, so for each observation, I use the astropy.coordinates python module to determine the azimuth based on the target's name, UT date, UT time, and the CHARA Array's terrestrial coordinates.

VEGA data do not store azimuth information, so for each observation, I use the astropy.coordinates python module to determine the azimuth based on the julian date, right ascension, declination, UT time, and the CHARA Array's terrestrial coordinates.

3.1.8 Elevation

The Elevation at which the observation was taken.

For Classic and CLIMB data, the CC_EL attribute is renamed to be Elevation with no values being altered.

For FLUOR data, the FL_EL attribute is renamed to be Elevation with no values being altered.

In JouFLU data, the elevation is saved in one of three attributes: CC_EL, FL_EL, or JF_EL. For each observation, only one of these three is used, so the elevation is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, the EL attribute is renamed to be Elevation with no values being altered.

PAVO data do not store elevation information, so for each observation, I use the astropy.coordinates python module to determine the elevation based on the target's name, UT date, UT time, and the CHARA Array's terrestrial coordinates.

VEGA data do not store elevation information, so for each observation, I use the astropy.coordinates python module to determine the elevation based on the julian date, right ascension, declination, UT time, and the CHARA Array's terrestrial coordinates.

3.1.9 Hour_Angle

The Hour Angle at which the observation was taken.

For Classic and CLIMB data, the CC_HA attribute is renamed to be Hour_Angle with no values being altered.

For FLUOR data, the FL_HA attribute is renamed to be Hour_Angle with no values being altered.

In JouFLU data, the hour angle is saved in one of three attributes: CC_HA, FL_HA, or JF_HA. For each observation, only one of these three is used, so the hour angle is taken from the attribute that was used for each observation.

For MIRC, MIRC-X, and VEGA data, the HA attribute is renamed to be Hour_Angle with no values being altered.

PAVO data do not store hour angle information, so for each observation, I use the astropy.coordinates python module to determine the hour angle based on the target's name, UT date, UT time, and the CHARA Array's terrestrial coordinates.

3.1.10 U_Coordinate

The U-coordinate(s) of the observation. If there are multiple U-coordinates, they are listed with the format "123.456 | 789.1011 | 121.314".

For Classic data, the CC_U attribute is renamed to be U_Coordinate with no values being altered.

For CLIMB data, values from the CC_U_12, CC_U_23, and CC_U_31 attributes are combined in the above format.

For FLUOR data, the FL_U attribute is renamed to be U_Coordinate with no values being altered.

In JouFLU data, the U-coordinate is saved in one of three attributes: CC_U, FL_U, or JF_U. For each observation, only one of these three is used, so the U-coordinate is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, values from all attributes with the following format are combined in the above format: U_X_Y where X and Y are two different telescopes.

No U-coordinate values are saved in PAVO or VEGA data.

3.1.11 V_Coordinate

The V-coordinate(s) of the observation. If there are multiple V-coordinates, they are listed with the format "123.456 | 789.1011 | 121.314".

For Classic data, the CC_V attribute is renamed to be V_Coordinate with no values being altered.

For CLIMB data, values from the CC_V_12, CC_V_23, and CC_V_31 attributes are combined in the format above.

For FLUOR data, the FL_V attribute is renamed to be V_Coordinate with no values being altered.

In JouFLU data, the V-coordinate is saved in one of three attributes: CC_V, FL_V, or JF_V. For each observation, only one of these three is used, so the V-coordinate is taken from the attribute that was used for each observation.

For MIRC and MIRC-X data, values from all attributes with the following format are combined in the above format: V_X_Y where X and Y are two different telescopes.

No V-coordinate values are saved in PAVO or VEGA data.

3.1.12 RA

The right ascension of the target being observed. All right ascension values are converted to decimal form for use in the target search feature of the CHARA Online Database Portal.

For Classic and CLIMB data, right ascension values are taken from the CC_RA attribute.

For FLUOR data, right ascension values are taken from the FL_RA attribute.

In JouFLU, right ascension is saved in one of three attributes: CC_RA, FL_RA, or JF_RA. For each observation, only one of these three is used, so the right ascension is taken from the attribute that was used for each observation.

For MIRC, MIRC-X, PAVO, and VEGA data, right ascension values are taken from the RA attribute.

3.1.13 DEC

The declination of the target being observed. All declination values are converted to decimal form for use in the target search feature of the CHARA Online Database Portal.

For Classic and CLIMB data, declination values are taken from the CC_DEC attribute.

For FLUOR data, declination values are taken from the FL_DEC attribute.

In JouFLU, declination is saved in one of three attributes: CC_RA, FL_RA, or JF_RA. For each observation, only one of these three is used, so the declination is taken from the attribute that was used for each observation.

For MIRC, MIRC-X, PAVO, and VEGA data, declination values are taken from the DEC attribute.

3.1.14 PI

The primary investigator for the observation. In future versions of the CHARA Online Database Portal, the PI name will likely be generated by comparison with historic observing schedules.

For Classic and CLIMB data, the CCPINAME attribute is renamed to be PI with no values being altered.

No PI names are saved in FLUOR, MIRC, or VEGA data.

For JouFLU data, the FLPINAME attribute is renamed to be PI with no values being altered.

For MIRC-X data, the PI_NAME attribute is renamed to be PI with no values being altered.

For PAVO data, the PVPINAME attribute is renamed to be PI with no values being altered.

3.1.15 Program

The program ID for the observation. In future versions of the database, the program name will likely be generated by comparison with historic observing schedules.

For Classic and CLIMB data, the CCPGNAME attribute is renamed to be Program with no values being altered.

No program IDs are saved in FLUOR, MIRC, MIRC-X, PAVO, or VEGA data.

For JouFLU data, the FLPGNAME is renamed to be Program with no values being altered.

3.1.16 Wavelength

The wavelength at which the observation was taken.

For Classic data, the CCLAMBDA attribute is renamed to be Wavelength with no values being altered.

For CLIMB data, two wavelength values are listed in the data associated with attributes CCLAMB1 and CCLAMB2. CCLAMB1 is renamed to be Wavelength. Instances where CCLAMB1 differs from CCLAMB2 are flagged for manual review.

For FLUOR and JouFLU data, the FLLAMD0 attribute is renamed to be Wavelength with no values being altered.

For MIRC and MIRC-X data, the WAVELEN attribute is renamed to be Wavelength with no values being altered.

For VEGA data, the wavelength attribute is renamed to be Wavelength with no values being altered.

3.1.17 File_Name

The full path name to the data file from which the metadata is taken.

3.1.18 Combiner

The name of the beam combiner.

4 The CHARA Online Database Portal

The chara_database_portal flask webapp can be found in the "frontend" directory of the gitlab repository. This app reads the metadata stored in the SQLite database file (named CHARA.db) and displays it based on users' requests. I discuss the different pages available in this webapp and their uses in Sections 4.1 - 4.3.

4.1 The Index Page

The index page of this app (the page that appears when a user enters the url) includes a list of features that the database portal includes and a radio button list of database tables that the user can use to choose which database table they wish to view. The options on this list are the All CHARA table (which is the table that is recommended to users) and a database for each of the individual beam combiners.

4.2 All CHARA Page

The All CHARA page displays standardized metadata in the combined table of the database (see Section 3) based on user input. An SQL query is created based on the user's inputs that determines what metadata is displayed.

The user can search this database table by star/object name. The input object name does not have to match object names in the database. When the user inputs an object name into the "Star Name" field, the webapp uses the astropy coordinates modules to determine the right ascension and declination of the input target. When the user submits this search, the generated SQL query will contain a constraint that will only show entries in the database where the right ascension and declination are each within the search distance (0.5 degrees by default) from the values determined by astropy. This search distance can be altered using the "Search Distance" field.

The "Start Date" and "End Date" fields allow the user to constrain their search by date.

The "Columns to Disply" field allows the user to select which attributes they wish to display. If the user does not select any attributes, Star_HD, UT_Date, UT_Time, RA, DEC, and Combiner are chosen by default. For detailed descriptions of these attributes, see Section 3.1.

The "Select Beam Combiner(s)" field allows the user to constrain their search by beam combiner.

The "Select PI(s)" field allows the user to constrain their search by the listed principle investigator.

4.3 Individual Beam Combiner Pages

The individual beam combiner pages have limited search functionality, but every attribute that is saved in the headers is available.

The user can search by star/object name using the "Star Name" field. However with these pages, unlike with the All CHARA page, the object name must be an exact match to what is in the database.

The "Start Date" and "End Date" fields allow the user to constrain their search by date.

The "Display" field allows the user to choose which attributes to display.

Because these database tables have not been cleaned, there are many entries which do not have valid star names or other information is missing.