IRAM Memo 2014-?

From CASA to GILDAS
I - GILDAS Data Format Version 2

S. Guilloteau[1], E. Chapillon[1,2], F. Gueth[2], G. Dumas[2],
A. Lopez Sepulcre[2], K.T. Wong[2]
1. LAB (Bordeaux) 2. IRAM (Grenoble)

January 5, 2018– version 2.2

Abstract

With the advent of ALMA, IRAM users may prefer at some point to handle their ALMA data in GILDAS rather than in CASA[1], vice versa to handle IRAM data into CASA. This document describes the different ways to do so, and what are the benefits and limitations of the process. This document only covers **calibrated** data, either UV data or images. There is (currently) no plan to handle uncalibrated ALMA data into GILDAS.

Software versions used for latest tests : GILDAS Oct17 and CASA 4.7.0, CASA 4.7.2, CASA 5.0.0, CASA 5.1.1 You should not use a version of GILDAS older than June 2015. Using the latest version of GILDAS is recommended.

The file ∼ /`.gag.dico` should include the following lines:

```
gildas_headers 2
```

Memory manadgment could be controlled with the variables

```
space_gildas  4096
space_mapping 4096
```

Related documents: *Mapping documentation*

[1] `http://casa.nrao.edu/docs/userman/UserMan.html`

Contents

1 Introduction

ALMA data is delivered from the science archive in `ASDM` format. IRAM raw data format are `ipb` and `class` for the PdBI and 30m respectively. Data must be calibrated with the corresponding software (CASA for ALMA, GILDAS for IRAM). Calibration leads to a calibrated measurement set for ALMA data, and UV tables and CLASS tables for IRAM data (respectively interferometry and single dish data).
The exchange format between GILDAS and CASA are UVFITS for tables and FITS for images.

2 Design differences

2.1 Imaging Philosophy and Data Architecture

CASA intends to solve the *Measurement Equation*, whatever the complexity of this process. It is an all-in-one package for this purpose, where calibration and imaging are deeply intermixed and use a unified data format. As a result, a CASA Measurement Set is a complex architecture encompassing relations between many components stored as Tables in a directory-like tree. It can handle calibrated data, calibration tables, multisource data sets, raw data in the same architecture, allowing to retain all information to process complex images, such as multi-frequency synthesis of polarized emission observed in a mosaic of fields.

On the contrary, GILDAS is designed to break the process in largely independent steps. For IRAM data, calibration is done in one program (CLIC for interferometry or CLASS for single-dish), and imaging in another (MAPPING), with an intermediate step consisting of extracting from the calibrated data the limited set of information needed for imaging. These are written as UV Tables for interferometry or CLASS Tables for single-dish. The Gildas Data Format stores a limited number of informations in a single binary data file with a compact binary header, and is only suited for calibrated data.

GILDAS does not handle polarization transparently at the current time: polarization states must be imaged independently.

2.2 Frequency and Velocity scales

In the ALMA use of CASA, all observations are kept in the Observatory frequency frame, and only converted to a celestial reference frame (such

as the Local Standard of Rest, LSRK) at later stages, during imaging if requested.

GILDAS is more analysis oriented, and contains a dual interpretation of the frequency axis. This axis can either be interpreted with a Velocity scale, usually in the LSRK frame relative to a spectral line rest frequency, or as a Rest Frequency, with the astronomical source velocity specified. The choice of representation depends on the astronomer's science objective: astronomical object study (in which case the Velocity representation is more appropriate) or chemical composition study (in which case the Rest Frequency representation is preferred).

3 From CASA to GILDAS

3.1 Transfering UV data from CASA to GILDAS

The basic idea of data transfer at this stage is to transfer a whole spectral window to GILDAS, and use the tools available in GILDAS for extracting channels, resampling, subtracting continuum, etc... rather than doing that in CASA.
Warning!: Imaging and deconvolution should be done using the commands rather than the procedures because the laters paramters are tuned for NOEMA, not optimized for ALMA.

3.1.1 In CASA

The steps in CASA involve

1. Extracting the data to be exported
    - Separating spectral windows into independent MS
      When your calibrated.ms dataset is the result of a concatenation of several execution blocs, your need to split the several spectral window corresponding to the same freqeuncy range (see the listobs of the .ms file)
    - Separating different sources into independent MS
    - Getting rid of flagged data
2. Setting the velocity reference frame and correcting for Doppler motions
3. Exporting to UVFITS

The CASA commands to be use are: *mstransform* or [*split* + *cvel*], and *exportuvfits*.

Example:

- 1. For CASA relases up to 4.6.0

```
split(vis='calibrated.ms',outputvis='source0-spw2.ms',
      field='0', spw='2', keepflags=F)
```

  `keepflags = F` is necessary to get rid of the flagged data, which are otherwise written into the UVFITS file (see below). Selecting a source (field) and a spectral window is necessary to create the corresponding UV table in GILDAS.

  If, and only if the CORRECTED column of the measurement set is empty (i.e no action was performed on the .split.cal file, like e.g. flux equalization), you should force *split* to read the DATA column with the option: `datacolumn='DATA'`.

- 2. For CASA relases up to 4.6.0

```
cvel(vis='source0-spw2.ms', outputvis='source0-spw2-cvel.ms',
     outframe='LSRK', restfreq='345795MHz')
```

  The arguments `outframe='LSRK'` and `restfreq = ...` are necessary to select the LSR velocity frame and a rest frequency for the line to be imaged. Please be careful that this rest frequency has no default value within CASA.

- 1+2. For CASA relases 4.7.0 and later ones

```
mstransform(vis='calibrated.ms', combinespws=True, spw='2',
     field='0', outputvis='source0-spw2-cvel.ms',
     regridms=True, outframe='LSRK', restfreq='345795MHz',
     keepflags=False, datacolumn='DATA')
```

  `mstransform` is a much powerfull and memory saving task than `cvel`. The arguments `combinespws=True` will combine all the input spectral window in one.
  `regridms = True`, `outframe='LSRK'` and `restfreq = ...` are necessary to select the LSR velocity frame and a rest frequency for the line to be imaged. Please be careful that this rest frequency has no default value within CASA. By default, this command works on the "corrected" column. You should use `datacolumn='DATA'` unless you applied a calibration to your *.ms dataset. `keepflags=False` could be done at this stage if not done in `split`.

- 3. `exportuvfits(vis='source0-spw2-cvel.ms', fitsfile='source0-spw2.uvfits', multisource=F)`
  Export a UVFITS file, with one source only (`multisource=F`).

  Note that there is an additional test in the command exportuvfits in CASA 4.7.0 and CASA 4.7.2 (probably in CASA 4.7.1 too) that test for a receptor angle and prevent exportuvfits to finish correctly. A workaround is to perform `split` and `mstransform` in CASA 4.7.*, then `exportuvfits` in CASA 4.6.0.

3.1.2 In GILDAS (short recipe)

In practice, all required steps 1 to 5 (described below) are encapsulated by the `@ fits_to_uvt` script, called as
`@ fits_to_uvt FitsFile UVTable`
File names should be without extension.

The script also handles some nasty details, like the source coordinates being hidden in different places in the UVFITS file depending on the CASA version.

You can modify the velocity axis by
`@ fits_to_uvt FitsFile UVTable FREQUENCY Freq VELOCITY Value LINE Name`
Where"Freq" is the rest frequency in MHz, "Value" is the source velocity in km/s and "Name" is the line name (for bookkeeping only).

Example:
`@ fits_to_uvt source0-spw2 source0-spw2 FREQUENCY 345795 VELOCITY 10 LINE test`

3.1.3 In GILDAS (details of actions)

The steps in GILDAS involve:

1. Converting UVFITS to UVT
2. Extracting/collapsing the polarization information
3. Adjusting the weights to properly estimate the noise
4. Identify and flag bad data

5. Setting the frequency and velocity references

This can be done in MAPPING, using commands or tasks

- Step 1
  `fits 'name'.uvfits to 'name'.uvt /style CASA`
  will create a GILDAS UV table from the UVFITS file. Polarization information is handled properly at this stage.

- Step 2 to 4
  Although they can be perfromed through the three tasks mentioned below, Steps 2 to 4 are best handled by a single task called `uv_casa`, which saves 2 intermediate files (and thus 2 read/write of large files).

- Step 2
  `run uv_splitpolar`
  will extract the required polarization information (H, V, Total intensity, etc). The desired polarization state should be set to `NONE` to optimize signal to noise ratio for unpolarized sources, to `I` if polarization is a concern.

- Step 3
  Depending on the CASA version, the weights are not handled in the same way in the UVFITS file. E.g. in Casa 3.4, they are approximately correct, in Casa 4.1, they are off by a large factor (perhaps the number of channels?). Task `uv_noise` utilizes the many channels available (3840 per spectral baseband) to compute a statistic per visibility, and adjust the weight through a median scaling factor.

- Step 4
  CASA `exportuvfits` unfortunately writes flagged data, unless these have been removed from the measurement set by `split`. The convention for flagged data is not described in the resulting UVFITS file. Task `uv_noise` will also flag data with "unusual" weights, i.e. those deviating from the median by more than some factor (user specified, default 3).
  Task `uv_trim` will remove the flagged data from the UV table, saving space.

- Step 5
  At this stage, one could start usual imaging using the standard MAPPING commands `READ UV 'name'; UV_MAP`. Commands `MODIFY FREQUENCY`

`Value` and `MODIFY VELOCITY Value` will set the desired correspondence between Velocity and Frequency axis.
Since the procedures (e.g. `GO UVMAP; GO IMAGE...`) are fine-tuned for NOEMA and not for ALMA, the default parameters are not always the best and we recommend to use the commands instead (`READ UV; UVMAP; CLEAN...`) to do the imagery. The visualization procedures (e.g. `GO VIEW; GO NICE...`) are fine.

3.1.4 Mosaics

For GILDAS version older than June 17 To transfer mosaics data from CASA into GILDAS, you should apply the methode described above, for each field and each spectral window. Once you have all the uv tables, you can proceed in mapping the usual way. Take care about the filename convention for UV Tables (sourcename-fieldid.uvt)

FOR GILDAS version after June 17 GILDAS now support uvt table with several fields. You should apply the methode described above putting all fields in a single ms (but still one ms per spectral window). To ease the split, you can use the target name in the "split" command. In such a case, it is recommended to add the option
`intent = ''OBSERVE_TARGET#ON_SOURCE''`
to select only the scientific observation and get rid of the technical ones (e.g. the Tsys measurment)

3.1.5 ACA dataset

CASA make no difference between ALMA and ACA (as it reads the antennas setup that include the antennas diameters), and thus the FITS and UVFITS "TELESCOPE" keyword is set to 'ALMA' in all case.

In gildas, ACA is known as an instrument, and you can modify the header of your dataset with :

```
read uv 'name'
specify telescope ACA
write uv 'name'
```

3.2 Transfering Image data from CASA to GILDAS

Images can be exported by CASA as FITS files, which can then readily be converted into GILDAS images. However, the default FITS files created

by CASA have a frequency axis labelled in FREQUENCY, while GILDAS usually works with this axis labelled in VELOCITY. The images must thus be created in CASA using the `velocity=True` argument:

```
exportfits(imagename='mymap.image', fitsimage='mymap.fits',
           velocity=True)
```

They can then be imported into GILDAS using the standard command:

```
FITS mymap.fits TO mymap.gdf
```

Images delivered by ALMA after the QA2 (Quality Assessment) step have however been procuced without the `velocity=True` argument and have thus a frequency axis that could be wrongly interpreted by a number of GILDAS tools. However, the standard scripts `GO BIT` and `GO VIEW` should normally work. You could modify the axis with the command:

```
sic\modify mymap.gdf /spec velo
```

Automatic swapping to Velocity axis is available in `GO VIEW`.

4 From GILDAS to CASA

This was tested under CASA 3.4, 4.1, 4.2 and 4.2.1.

NOEMA is a known observatory of CASA starting version 5.1.1. If you use an older version of CASA you should update your "data" repository (the same way to update the leak second) to get the updated list of instrument.

4.1 Transfering UV data from GILDAS to CASA

4.1.1 In GILDAS

Create a *uv* table, then use the command `FITS` or the task `GILDAS_FITS` to convert it to UVFITS. In both case, the option `/STYLE CASA` must be selected, to write the correct flavor of UVFITS.

```
fits mysource.uvfits from mysource.uvt /style casa
```

4.1.2 In CASA

Use the task `importuvfits` to create a Measurement Set (ms file) from the UVFITS file (reminder: there is no explicit *uv* table in CASA).

```
importuvfits(fitsfile='mysource.uvfits',vis='mysource.ms')
```

4.1.3 Mosaics

In GILDAS :

- Old fashion (GILDAS version later than XXX)
  Convert each *uv* table into a UVFITS file, as described above.
  ```
  for i 1 to Nfields
  fits mysource-'i'.uvfits from mysource-'i'.uvt style casa_uvfits
  next
  ```
- new way: Convert your multisfield *uv* table into a uvfits file `fits mysource.uvfits from mysource.uvt style casa_uvfits`

In CASA : If you have one file per field, each UVFITS file must be converted into a measurement set `importuvfits`) and these ms files must be concatenated into one single field (`concat`).

```
for i in range(Nfields)
   importuvfits(fitsfile='mysource-'+str(i)+'.uvfits',
                vis='mysource-'+str(i)+'.ms')
   concat(vis=['mysource-1.ms','mysource-2.ms','mysource-3.ms',...],
          concatvis='mysource.ms')
```

This could obvioulsy be scripted in a more general way, e.g. writting the list of ms files in a python dictionary that is then used as input for `concat`.

Else just import your unique uvfits file:

```
importuvfits(fitsfile='mysource.uvfits',vis='mysource.ms'
```

4.2 Transfering images from GILDAS to CASA

4.2.1 In GILDAS:

use the command `FITS` or the task `GILDAS_FITS` to convert a GILDAS data cube to FITS. There is no need to use options. `fits mymap.fits from mymap.gdf`

4.2.2 In CASA:

use the task `importfits` to create a CASA image from the FITS file.
`importfits(fitsimage='mymaps.fits',imagename='mymaps.image')`
Caution! With FITS files written by GILDAS versions < March 2013, CASA claims that the velocity frame of the data cubes is TOPO(-centric)

even though it is LSRK.