ALMA Calibration Workshop

Lab #1: Basic Calibration

December 1, 2011

Overview

The goal of these exercises is to complete the initial calibration of an ALMA data set. There are two example data sets contained in this lab, and you have time to complete only one of them.

Section 1 (the "Four Quasar" data set) is designed for the novice, who needs to get his/her feet wet with ALMA calibration. Feel free to passover this section if you have previous experience, and go directly to section 2.

Section 2 (the "TW Hydra" data set) is designed for the astronomer with some prior experience with mm/submillimeter intereferometric analysis. Feel free to self select which set of data you would like to work on.

Initial Computer Set-up

Regardless of which data set you choose, there are a few things you need to do to get started.

1. Login using the name and password on the little slip of paper.
2. To bring up a terminal, simply right-click on the desktop. A menu will appear.
3. In your home directory, open a text file with your favorite editor, such as `vim myscript.py` or `emacs myscript.py &`. This will become an executable script with `casapy` commands. You will keep a record of all of your tasks in the script.
4. Be prepared to write every command in the text file, and then copy and paste them into the CASA shell.
5. Decide now which data to work with. If you are unsure, work with the "four quasar" data.
6. Copy over the measurement set directory to your home directory, just in case you need to start over, using a `cp -r` command from the shell.
7. Start up CASA by typing `casapy` in a terminal window.

Now, you are ready to continue with either §1 or §2.

1 The Four Quasars

The goal is to calibrate ALMA data from one quasar, using the other three as bandpass and gain calibrators.

Summary of Data This data-set consists of a single measurement set (MS) of successive scans on four bright quasars:

1924-292 The brightest quasar, which we will use as the flux and bandpass calibrator

923-210 & 1911-201 The medium bright quasars, which we will use as gain calibrators

1939-154 The faintest quasar, which will be our "science" target.

These data consists of four spectral windows in Band 6. However, given the large amount of time it can take to inspect each spectral window, we recommend selecting just one for this calibration exercise.

1.1 Getting the data and starting CASA

- Go to your working directory, where you should find a directory containing the example measurement set:
  `FourQuasarsBand6_Calibration`
- If you have not done so already, copy this to a new directory.
- cd to the working directory and look at its contents. You should see a directory with a single measurement set, and a Python script. *
- From the shell, start `casapy`. You should see a *logger* window appear that shows your messages.

1.2 Getting used to iPython

CASA is wrapped inside an iPython shell, and the CASA commands are for the most part simply Python functions. This section is designed to give you a very quick overview of Python, so you can use it to your advantage.

*The corrections based on antenna position and water vapor radiometer data have already been applied (hence the "wvr.ms" extension).

First notice that inside of CASA, common shell commands such as `ls` and `pwd` work as typed, but uncommon ones such as `df` do not. Any shell command can be executed with a `!`, such as `!df`.

Next, to get help on any object you can type the object's name, followed by a `?`. For example, type `listobs?`, and you will learn that it lists properties of a measurement set and sends the output to the logger. You also will notice that it takes a keyword argument `vis` for the visibility file.

iPython uses tab completion extensively, so hitting the tab key will complete or list commands. Moreover, Python connects commands with a `.`, so hitting tab after a `.` lists the available commands.

There are many modules that are already imported into CASA, and these can be investigated by typing `help` followed by `modules` at the help prompt. You can `import` them, and then find out about any one of these by typing its name followed by a dot and tab. For example try `import os` and `os.<tab>`. Now typing `temp = os.listdir(os.curdir)` makes a Python list of the contents of the current working directory and puts it in the object `temp`. By typing `print(temp)` you can see it.

Now let us set a variable with our measurement set filename. We could simply set it via `msfile = 'uid___A002_X219601_X4cd.wvr.ms'`, or we could have set it via `msfile = temp[0]`.

Python is a scripting language, so we recommend keeping a Python file with all the commands in a text file that you could execute later. That way you can simply redo all of your analysis by simply issuing the command `execfile('<your file>.py')` or `run <your file>.py`. The `#` is the comment character in Python , and Python is very picky about indentation so start every line with either a `#` or a Python command. Included in the data directory is a Python script that can be looked at when you are done.

1.3 Initial inspection and flagging

First we wish to inspect our data, and flag some initial data that we do not want to use.

- Use `listobs` to get basic information about the data, including targets observed, time range, spectral setup, etc. It's best to choose just one spectral window to look at during this exercise. `listobs` uses the `vis` keywords to set the measurement set directory.

- Use `plotants` to plot the positions of the antennas in use when these data were taken. Choose a reference antenna close to the center of the array, and make a note of it.

- Flag data that we already know we want to exclude using `flagdata`. The two types of data we will initially flag are:
  - *shadowed data*, where one antenna blocks the line of sight of another
  - *autocorrelation data*, as opposed to the cross-correlation data.

  To do this, you will need to use the following keywords:

  `vis`: measurement set directory.

  `flagbackup`: set to `False` to save time

  `mode`: such as '`shadow`' or '`manualflag`'

  `autocorr`: set to `True` to flag autocorrelations

- Save the current flagging state using `flagmanager`. This can come in handy if you need to revert back to this state later. Useful keywords are:

  `vis`: measurement set directory.

  `mode`: for example '`save`' or '`restore`'

  `versionname`: a name for the flag version

1.4 Interactive inspection and flagging

The goal of the next few steps is to inspect the phase and amplitude versus frequency and time for antenna-based issues, and to flag bad data along the way using the `flagdata` command. You want to check that amplitude and phase vary smoothly with time and frequency. You will use the `plotms` GUI in CASA for these steps. Take notes of any unusual behavior, such as sudden jumps, large gradients (e.g. phase wraps), unusual amounts of scatter, etc. Make sure you are careful not to plot too much data at once, so each time you call `plotms` use its keywords.

You may also wish to follow-up on a hunch that something may be bad. For example, if you notice that one baseline appears bad, you can plot all the baselines involving that antenna. Perhaps the antenna is bad, or perhaps it is only the baseline? Perhaps it is bad for all sources? Perhaps it is bad in both XX and YY, or perhaps it is bad in only one correlation?

Finally you will want to look at the full data set to check for any more bad data we may have missed, and to make sure that the "science target" is flanked by gain calibrator observations.

1. Inspecting Your Data

   (a) Plot amplitude versus time for each of the baselines involving the reference antenna you chose in section 1.3 above with `plotms`. You may find it useful to cycle through the set of baselines that include the reference antenna, so you can better isolate potential problems. Also, you may want to average over a large number of spectral channels, or choose just one channel, to save time in plotting. For example, you may want to set the following `plotms` keywords:

      `vis`: measurement set directory.

      `xaxis`: x-axis quantity, such as `'time'`

      `yaxis`: y-axis quantity, such as `'amp'`

      `averagedata`: a boolean variable such as `True`

      `avgchannel`: tells it to average over these channels. Set it to a big number, such as `'100'`.

      `iteraxis`: tells it to iterate over an axis, such as `'baseline'`

      `coloraxis`: tells it to color the points. If we color them by `'corr'`, it will color code the XX and YY correlations.

      `antenna`: the baseline to pick. For example, if you pick all baselines associated with antenna 11, you would set it to `'DV11&*'`.

      `spw`: the spectral window(s), such as `'1:10~50'`

      `field`: the object we are looking at, such as `'1924-292'` [†]

      `ydatacolumn`: set to `'data'` as opposed to `'corrected'`.

   (b) Plot phase versus time, in much the same way.

   (c) Plot amplitude versus frequency. You may want to average over time (`avgtime='1E8'`), and to allow averaging of scans (`avgscan=True`), to save time in plotting.

   (d) Plot phase versus frequency.

   (e) Plot amplitude versus time, for all fields at once.

   (f) Plot amplitude versus frequency for all fields at once.

2. Flagging Your Data

[†]It can be useful to look at all calibrators; however, if you are running short on time, you can just focus on 1924-292.

(a) Use the `flagdata` task to flag bad data identified in steps 1a-1d below. You may choose to do this interactively, where you flag data as you progress through different plots, or wait until you've looked through all of the plots. `flagdata` can be used to manually flag data based on most criteria, using the following additional keywords:

`mode`: set to `'manualflag'`

`antenna`: the antenna or baseline to pick. For example, if you pick all baselines associated with antenna 11, you would set it to `'DV11'`.

`correlation`: the polarization correlation to flag (e.g. `XX` or `YY`) the default is both correlations.

`spw`: the spectral window, such as `'1'`, or `'0:0~5;61~63, 3:0~5;61~63'`.

(b) Use `flagmanager` to store the current flagging state.

Once you have finished inspecting and flagging your data, refer to Table 1 to compare your answers with some of the things we found.

1.5 Bandpass calibration

Since 1924-292 is much brighter than the other quasars, we will use this one as our bandpass calibrator. Typically, the bandpass calibrator is observed only once in an MS, usually at the beginning; in this case, 1924-292 is observed much more often.

1. Use `gaincal` to solve for the variation of phase as a function of time on very short timescales. Average over a small fraction of the total bandpass to avoid phase versus frequency effects.

2. Inspect how well the solution from 1 improves the phase versus time solution. To do this, first use `applycal` to apply this calibration table to the bandpass calibrator. Then, use `plotms` to plot phase versus time. It is good to plot all baselines, iterating over each antenna. Note that `applycal` only overwrites the "corrected" column, and leaves the "data" column alone. This means that you can switch between these columns in the `plotms` GUI by setting `ydatacolumn`, to see the before and after effects of applying this table.

3. Use bandpass to solve for the frequency response of each antenna. Average all data in time, allowing combination of all scans and field

IDs for 1924-292. Apply the phase versus time calibration determined in step 1 on-the-fly.

4. Use `plotms` to look at the calibration table created in step 3. Look at both phase versus frequency and amplitude versus frequency. As in step 2, first use `applycal` to apply the calibration table to the corrected data column, then use `plotms` to compare the before and after plots.

1.6 Amplitude and phase calibration

1. Use `setjy` to set the absolute flux density scale for the flux calibrator. 1924-292 does not have a known model in `setjy`, so you will need to enter its flux density manually. It is well described by a point source, and its flux density is well approximated by a flat spectrum with a flux of 8.1 Jy in Band 6, and we do not care about polarization, so you can zero Q, U and V.

2. Use `gaincal` to derive a short-interval solution for phase variations for observations of all calibrators. Apply the bandpass solution on-the-fly.

3. Use `gaincal` to derive the long timescale phase variation calibration table for all calibrators. Do not average over scans or fields. Apply the bandpass solution on-the-fly. This is what we will use to correct the source later on.

4. Use `gaincal` to derive the long timescale amplitude variation calibration tables for all calibrators. Apply the bandpass solution, as well as the short timescale phase solutions determined in step 2, on-the-fly.

5. We now want to make a new version of the amplitude solution, which includes the absolute flux scale. To do this, use the `fluxscale` task to set the flux of the two phase calibrators with reference to the flux calibrator.

6. Apply the calibrations derived above to the data. It is best to do this for each field separately. For each field, you will apply three tables for (1) bandpass (i.e. frequency), and (2) phase and (3) amplitude time solutions.

7. Inspect the "corrected" column, looking at plots of phase and amplitude versus time and frequency for all fields. At this point, it is best

not to average over time or frequency, since this is likely the final inspection of the data, and you want to check for any further oddities or badness.

8. If any more data stand out as bad, flag them, and repeat §1.5 and §1.6 if you think it may influence the calibration tables.

1.7 Cleaning Up

To save and get a quick look at our data:

- Use `split` to save the calibrated data in a new MS. You can leave out unwanted spectral windows, fields, etc.
- Use `clean` with niter=0 to produce a dirty image of the continuum. Use `viewer` to look at the images.

2 TWHydra Band 3

2.1 Introduction

This section is designed for the more experienced user, so it will cover the same topics as §1, but quicker and with less explanation. While it is also designed to walk you through editing and calibrating the ALMA TWHydra Band 3 science verification data, it does not provide the casapy commands. However, the uncalibrated data referenced below does contain a casapy script that can be referenced when needed.

2.2 Startup

- The data may be already in a working directory on your computer, but if not, extract the TWHydra Band3 uncalibrated data via:
  `tar -xzf /export/lustre/SV/TWHyaBand3/TWHYA_BAND3_UnCalibratedMSAndTablesForReduction.tgz`
- List the contents of the directory.
  - The measurement sets have already had Tsys and WVR corrections applied. Plots of Tsys and WVR data are provided in subdirectories.
  - A Python script that edits and calibrates these data is also included.
    * You can run this script in totality using the command `execfile('twhya_band3_calibrate.py')` from the casapy prompt.
    * You can also use the script as a cheat sheet as you go though these exercises. If you get lost or have trouble figuring out how to set a given task's parameters, you can open the Python script to see how it was written.
- From the command line, start `casapy`.
  - Inspect the 4 measurement sets using task `listobs`. It can be useful to send the `listobs` output to a file for easy reference later.
  - Plot the antennas using `plotants`.

2.3 Working with Multiple Measurement Sets

Notice that we have multiple measurement sets (MSs), so one may think that it would make sense to concatenate all our measurement sets so we're only

working with one rather than four. While this can simplify data reduction, it can also limit our ability to analyze and calibrate our data. For this data set, you will see below that 2 of the MSs were contaminated by flux from Saturn. Toward the end of our calibration we must take an extra step to scale the flux density for these two MSs. If we concatenate now, we will not be able to scale the flux in this way.

In general, only concatenate MSs if you're confident that each MS contains data recorded under the same observational conditions and the same array configuration.

When working with multiple MSs, it is extremely convenient to use for loops over a list of the MS file names. This allows you to execute a task using the Python function syntax and substitute a different MS file name during each iteration of the loop.

2.4 Flagging - Round 1

The first step in obtaining good, calibrated data is editing the data and flagging any data that are corrupted or not useful. For most projects this means immediately flagging autocorrelations and any data from an antenna that was "shadowed" by another antenna. Do this now using the `flagdata` task, which you can learn about in the iPython shell as we discussed above in §1.2. You can use the `flagmanager` task to save and organize your flags.

Titan is the primary flux calibrator for this project. Saturn can sometimes contaminate Titan data. The difference in source size is evident when the data is plotted as a function of uv distance. Use task `plotms` to inspect each MS for a signal from Saturn. Two MSs will have it; two will not. When you find Saturn, flag those data using `plotms` or `flagdata`.

`plotms`: `plotms` can be operated in GUI mode. Just start it with `plotms()` and then use the GUI for everything else. This can sometimes take a long time, however, so you may want to plot only some data at a time, as described above is §1.4.

`flagdata`: Flagging can be accomplished from the casa prompt, using `flagdata`, or through the `plotms` GUI. For reproducibility, we recommend using `flagdata`. If you must start over, you can execute all your previously entered `flagdata` commands as a script.

Now we will visually inspect the data and flag any data that appear to be bad. Nominally, this involves plotting both amplitude and phase versus time and channel for each MS and looking at each of the two spectral

windows. However, given time limitations we recommend working with only one spectral window for this exercise, such as `spw=1`.

Here are some things you should see while inspecting these data:

Birdies: "Birdies" are strong, narrow-band noise spikes that span one or two channels. These can be seen most easily in the sources observed for long times, such as the phase calibrator or target. Look through the MSs and flag all birdies. Take care not to flag spectral lines. Note: we found 4 birdies in each SPW.

Saving Flags: Before we begin calibration it's a good idea to save your flags using task `flagmanager`. If anything goes wrong during calibration, you will save time by applying your saved flag table rather than re-executing a series of `flagdata` commands.

2.5 Calibration - Round 1

- To solve for calibration solutions, a reference antenna must be chosen. It's a good idea to choose an antenna close to the center of the array. You can use the task `plotants` to view the array configuration. It's also a good idea to choose an antenna that is well behaved (and not flagged) for most of the observation.
- Use `setjy` to determine the flux scale from Titan. This initializes the model column of the MS and inserts a model of the flux density for this source. We will use this later to compute the flux calibration for the other targets.
- Now solve for the bandpass calibration using the bandpass calibrator (3C279):
  - First, solve for phase solutions for each integration using task `gaincal`. This is a necessary step prior to bandpass calibration because time-based phase variations would cause the vector-averaged bandpass solutions to decorrelate. Average over a subset of the channels to avoid frequency-dependent variations.
  - Then, use task bandpass to solve for the gains as a function of frequency. Be sure to apply the phase calibration solutions when solving for the bandpass solutions on-the-fly.
- Now solve for the calibrator phase solutions *for each integration* while applying the bandpass solutions. (Later, when applying calibration solutions to the calibration sources, use these per-integration solutions.)

- Now solve for the calibrator phase solutions *for each scan* while applying the bandpass solutions. (Later, when applying calibration solutions to the target, you can use these per-scan calibration solutions.)
- Now solve for the calibrator amplitude solutions over each scan while applying the bandpass and per-integration phase solutions.
- Titan was our primary flux calibrator, so we need to bootstrap this calibration to the other calibrators using the task `fluxscale`. This will scale the flux by comparing the data and model columns of the flux density calibrator and assuming the same flux offset for the other two calibrators.
- Apply the calibration tables to each source. To do this use task `applycal`. Use parameters `gaintable` and `gainfield` to select the table and fields that should be applied to each source. For example, 3C279 was used to solve for bandpass solutions, so that source should always be used when applying the bandpass table. Note that `applycal` will flag all visibilities for which no calibration solutions exist.

2.6 Flagging - Round 2

Now that a 1st round of calibration has been applied, you should inspect the corrected data. It is not uncommon for data problems hidden before calibration to become visible after calibration. As before, use `plotms` to inspect the amplitude and phase as a function of time and channel. However, this time it is important to plot the corrected column rather than the data column. Flag any bad data that you see. When you are finished save your flags using `flagmanager`.

2.7 Calibration - Round 2

If any additional data was flagged during the 2nd round of flagging, it's a good idea to repeat the bandpass, phase, and amplitude calibration. To do this simply repeat the appropriate steps from round 1 of calibration. If you wish to overwrite the round 1 calibration tables, you will need to move the existing tables out of the way.

At this point we have performed 2 rounds of flagging and 2 rounds of calibration. If necessary, you could continue to iterate between flagging and calibration until the calibrated data look good. However, remember that `applycal` will flag any data for which no calibration solutions exist; if you

begin losing data unexpectedly, this could be the culprit. The solution to this problem is to either reset your flags to a previous state using `flagmanager` or to rerun all `flagdata` commands on a fresh data set, calibrate, and apply calibration solutions.

2.8 More Advanced Flux Scaling (optional)

The output from `fluxscale` was sent to the logger. Inspect the flux densities for 3C279 and J1037-295 for each measurement set. Ideally, each measurement set should produce the same flux values. You will notice that the flux densities for X149.ms and X30f.ms are high and low respective to the average over the 4 measurement sets. These are the same two data sets that were contaminated by Saturn. The flux densities may be affected by some remaining flux from Saturn, or by a higher noise level since we flagged more of the data. To resolve this problem we will use the average flux densities determined for X1e9.ms and X23.ms and apply them to all four measurement sets. This can be done using task `setjy` with input parameter `fluxdensity`. Note that `setjy` writes to the MS model column. We can now ignore the absolute flux calibration table created by `fluxscale`.

Now that the absolute flux has been determined for each of the three calibrators, we want to solve again for the amplitude over each scan. While doing this, apply the bandpass calibration and per-integration phase calibration. When finished, apply calibration tables to each source using `applycal`. Use parameters `gaintable` and `gainfield` to select the table and fields that should be applied to each source.

2.9 Finishing Up

To finish up, we want to export the calibrated data for our target. First concatenate all 4 measurement sets using task `concat`. Then, split out the TWHydra calibrated visibilities using task `split`. Note that what was the `'corrected'` column (i.e. the calibrated visibilities) is copied into the `data` column of the newly created measurement set.

With that accomplished, you're ready to image TWHydra tomorrow. If you do not finish however, you will be provided with new calibrated data tomorrow.

If you have extra time on your hands, feel free to run the script contained in the original tarball on a fresh data set and compare the output TWHydra MS with the MS you generated here. They should be about the same.

Table 1: Inspecting and Flagging Example Answer Key

Field	Spec. Win.	Corr.	Antenna(s)	Problem
all	all	YY	DV05	amp is very low at all freq, phase varies strongly
all	all	both	DV13	low amp, phase wrap in freq
all	2	YY	DV01	low amp
all	2	XX	DV01	high amp
all	all	both	all	edge channels, 0-5 and 61-63
*all	0	both	all	dip at 229.55 GHz
*all	1	both	all	dip at 231.27 GHz
*all	2	both	all	dip at 244.15 GHz
all	0	YY	DV12	lots of scatter at $>$ 230 GHz, does not improve after calibration
all	all	YY	DV10&PM01	brightest source shows low points for these baselines
1939-154	all	both	all	flag last scan #72, since we'll use this a "science" target
All unstarred should be flagged outright.				
* Atmospheric features. Flag affected channels only after bandpass solution is applied.				