Data Processing School :: Exercise 10

Source directory	/data/lofarschool/data/exercise 10
Contact person	Martin Bell

Context

This exercise will make use of a CS1 Jupiter observation which has been correctly converted into UVFITS format. The Parseltongue script used in this example prepares the data so that it is usable within AIPS. For more details on the conversion process and loading non UVFITS format into AIPS see below. If the user has experience with AIPS then they can simply log in and start following some of the suggested plotting and analysis stages below. If the user has never used AIPS before, the script contains 'hashed out' examples which can simply be 'un-hashed' and run from the command line.

To log in to the correct computer the instructions are as follows

```
ssh –X bell@dop168.astron.nl
```

Password will be given in the session. Then change directory to

```
>>cd /export/astron/bell
```

There should be 10 directories visible labelled Exercise_1-Exercise_10. Choose a directory and continue as follows.

To run this script the AIPS installation will have to be sourced by

```
>>  source /local /scripts/doAIPS
```

Then we have to set the environment variable so that AIPS and PT know where the data that is to be loaded is located.

```
>> setenv 'MYAREA' /Disk/Location/That/Your/File/is/in
>> pt lofar_load.py
```

To start AIPS

```
>> aips tv=local:'Number'
```

Where 'Number' is the exercise file location number you are working in e.g Exercise_5 …. aips tv=local:5

Without un-hashing the exercises section the code will simply load the data into AIPS.

Exercise 1: Un-hash the 'view antenna table' section and re-run the code. This should show a plot of the positions of the antennas. The projection is on the Earth's surface, examine it and make sure it makes sense. Can you estimate the latitude of the CS1 core from the graph?

Exercise 2: Log into AIPS and view the 'NX' table. Details about how to log into AIPS will be given in the actual session. Once logged in, at the command line type

```
>> pcat
>> getn 2
>> Task ‘LISTR’
>> optype = SCAN’
>> go
```

You should now be given an observation summary. How many channels are there? What is the operating frequency of the observation? How many visibilities does CS1 record per integration?

Exercise 3: Un-hash the ‘Plotting within AIPS’ section and view the following plots 1. UV plot 2. Time against Flux 3. Time against Phase

Exercise 4: Run the imaging section of the code. Take note how Parseltongue feeds the time ranges for which it wants to image over into a python function. View the three images (within AIPS). Also view the dirty beam, does it change?

```
>> pcat
>> getn 4 (or the catalogue number of the file you want to load)
>> tvlo
```

To clear the TV server, type

```
>> tvini
```

After finishing the exercises remove all of your data files from your AIPS number, as follows

```
>> getn NUMBER
>> zap
```

Repeat the step above for all the files in the AIPS catalogue, then to leave AIPS

```
>> kleenex.
```

Prerequisite

Notes on Conversion:

The script lofar_load.py will load a UVFITS file into AIPS which has been correctly converted from the Measurement Set format. The Measurement Set format is currently used by The Westorbork Synthesis Radio Telescope (WSRT) and LOFAR (and maybe others). The conversion can be done via the following two methods.

1. Casacore MS2UVFITS converter: This should write all data tables needed by AIPS. This is a standalone programme and can be used from the command line with the following syntax.

```
>> ms2uvfits ms=<infile> fitsfile=<outfile> multisource=F writesyscal=F
combinespw=F
```

2. Alternatively it can be converted using an AIPS++ installation as follows. At a command line type 'glish' then

```
>> myms.tofits
>>myms.tofits(fitsfile='L2007_01585_SB89.UVF',column='DATA',
spwid=1,start=100,nchan=1,width=1,writesyscal=F,multisource=F,combinespw=F)
```

Once conversion is finished the parseltongue script lofar_load.py can be run. Before hand just set the environment variable…

```
>> setenv 'MYAREA' /Disk/Location/That/Your/File/is/in
>> parseltongue lofar_load.py
```

Notes on Loading Data in AIPS:

The first part of this script makes the data usable in AIPS. The changes needed to be made are as follows

1. The data must be sorted in time order. It might be the case that the data is already sorted; however AIPS will need to see this in the UVFITS header. Sorting the data will do no harm in any case so this is run on ALL data. N.B Some telescopes might sort data in baseline order.
2. The data must be indexed. To list information about your observation an ‘NX’ table must exist. This is a snapshot of the observations you made and at what frequency and time etc. It is a radio astronomers ‘bread and butter’ of understanding what has gone on and where. The task ‘indxr’ will create this table and ‘listr’ will access it.
3. Redefine Stoke’s parameters. AIPS is designed for dual circular polarisation and LOFAR measures dual XX,YY so we have to redefine the Stoke’s parameters in the AIPS header. This only really allows us to work with ‘Stokes I’ (or total intensity) as the circular polarisation components will be incorrect. Therefore if you suspect any of the sources you are observing are intrinsically circularly polarised you will run into trouble. So stick to total intensity or perhaps go to the MIRIAD reduction package, it was designed for XX,YY polarised feeds and will better handle circular polarisation work. The changes we will make to the Stoke’s parameters essentially amounts to inserting a ‘-1’ into the AIPS header and this is done automatically in the script. See http://www.astron.nl/wsrt/WSRTsoft/cookbook.html for more details.

   ```
   Linear   ==  XX=I-Q, YY=I+Q, XY=-U+iV and YX=-U-iV
   Circular ==  RR=I+V, LL=I-V, RL=Q+iU  and LR=Q-iU
   Therefore (I,Q,U,V) >>>> (I,-V,-Q,U).
   ```