Post-Correlation Steps

Main Authors: Benito Marcote & Jay Blanchard

Date: 19 March 2020

Version: 4.4

This guide summarizes the steps to be performed on the data of an experiment after correlation (from when you receive the Yellow Folder until you distribute the data to the PI).

Bullet points:

- Check the quality of the data and convert them into an appropriate format.
- Run the EVN pipeline.
- Archive the data and contact the PI.

Note: Additionally to this guide, you **should always print the Checklist** (available in the Wiki) to do not miss any step during the post-processing of your experiment. And leave the paper inside the Yellow Folder once you are done with it.

General information

Computers used along the procedure

- *ccs*: where the correlator control files are located.
- *eee*: where all the information and results for the experiment are saved. This computer replaces the old *eee*, which still contains information for relatively old experiments.
- *jop83*: where the EVN pipeline and archive live.
- *vlbeer*: the ftp server where all stations put their local files (*log*, *antab*, *uvflg*,...) containing the information for each experiment.

The Wiki contains this document and other information:
http://www.jive.eu/jivewiki/doku.php?id=evn:supportscientists
A summary of tools mentioned in this guide exists here:
http://www.jive.eu/jivewiki/doku.php?id=supportsci:tools

Definitions in this document

- Anything between {} means text or a word that needs to be replaced according to your experiment.
- *exp* indicates the experiment name in lowercase (e.g. eg054g).
- *EXP* indicates the experiment name in uppercase (e.g. EG054G).

V*x files

These files (.vex/.vax/.vix/.vexfile) are *almost* the same file. They are all vex format. They contain the full information for an experiment. Initially created from SCHED, they are used by the stations during the observations, and by the correlator to produce the interferometric data. The differences between the slightly different extensions are:

- **vex**: Is the file created by SCHED (i.e. the one updated to *vlbeer* to be loaded by the stations).
- **vax**: Same file, but now the clock section has been added, and also the specific epoch. It is meant for rapid correlation control (e.g. for FTPs in NMEs).
- **vexfile**: Is the output of log2vex (it can contain some minor changes with respect to the *vax* file).
- **vix**: is the version that went through the correlator for production correlation (e.g. can have the clock section updated with respect to the previous one, and changes requested by PIs such as new source coordinates).

Get the data and inspect

The data, as recorded from the correlator, need to be converted to a standard format (Measurement Set, or MS). This process uses a *lis file* (for normal experiments `{exp}.lis`) to automate and document this task. The data will then be inspected with *jplotter*, and standard plots will be produced and archived to help PIs about the success of the observation. Standard issues will be addressed in this part. Finally, we will convert the data to *IDIFITS* files (readable by AIPS), which will be archived and pipelined.

CCS

Log in to *ccs* to get the initial files

```
ssh -Y jops@ccs
cd /ccs/expr/{EXP}/
```

NOTE: There is typically only one .vix file per project. In the case of e-EVN experiments, multiple experiments may have been observed within a single run, and thus under the same vix file. This will have the name of the very first experiment conducted in the session.

Create the *lis* file **by using one of the two following ways**:

```
showlog_new {EXP}
```

Which is the new version of `showlog` (to be decommissioned soon).

In the window, check the runs which are "PRODUCTION" (there probably will be "CLOCK SEARCH" runs too). If there are "UNKNOWN" status (only in e-EVN), we can mark them as "GOOD" going to 'Inspect Data' with the bottom-right options. Then, switch to 'ExportFile', see that all the scans we want have a "+" symbol and then click again in 'ExportFile' and save the lis file in the *{EXP}* directory. **Note:** in case of multi phase center or pulsar binning

experiments, the *save* button will show a tentative name for the output lis files as `expname*{key}.lis`. Do not change the `{key}` part, as that will be automatically changed to include the name of each phase center, or pulsar bin, producing as many lis files as centers (or bins) have been correlated.

The other method does not use a graphical interface, and directly writes a *lis* file by executing:

```
make_lis -e {exp} -p {profile} -s {exp}.lis
```

Where `-s` defines the name of the output file, and `-p` allows you to specify the jobs to be selected in the lis file by profile (e.g. only the production ones. *prod*, or in case of spectral line experiments it could be *prod_cont* or *prod_line*). This option is optional and if not specified, it will take all production jobs, only ignoring the clock search ones. **In case of multi phase center experiments**, you may add `-m SRC` with the phase center source name that will contain all calibrators' data. By default these data will be included in all phase centers.

Quickly check all scans that are present in the lis file by executing:

```
checklis {exp}.lis
```

This will tell you if there are problems with the files to process, like duplicated or missing scans.

eee

Log in to *eee* and get the data:

```
ssh -Y jops@eee
cd /data0/{your_username}/{EXP} # create the directory if does not exist
```

(**Recommended**): Create a *processing.log* file to keep track of all steps that you are doing in this experiment. This file contains the steps to be performed so you do not miss any of them, and you can record all the issues spotted in the experiment. You can create the template of this file by typing:

```
create_processing_log.py [-e {eEVN}] [-jss {your_surname}] {exp}
```

In case of e-EVN experiments, you need to specify the experiment name assigned to the observations. You can also specify who are you with the `-jss` flag.

Copy the PI letter and the experiment SUM file from the *jops* home directory (use any other computer if jop83 is not reachable):

```
scp jops@jop83:piletters/{exp}.piletter .
scp jops@jop83:piletters/{exp}.expsum .
```

The PI letter contains all information that will be sent to the PI by email and needs to be modified to provide a good feedback about the experiment.

The *expsum* file contains the information about the PI, the correlation setup, and the source information (which sources were observed, which ones are calibrators or targets, and which ones must remain protected).

NOTE: If these two files (piletter and expsum) has not been created yet (Bob typically do it once the experiment is correlated), you can do it manually by running:

```
ssh -Y jops@jop83

jexp    # This is a GUI program
```

- Select your experiment name from the pull-down menu on the left, just above the Save/Update button.
- Click the "Completed" button (after changing the date just to the left, if necessary).
- Click "Save/Update".
- Click "Exit".

Copy the *vix* and *lis* files from *ccs*:

```
scp jops@ccs:/ccs/expr/{EXP}/{exp}.vix {exp}.vix
scp jops@ccs:/ccs/expr/{EXP}/{exp}.lis {exp}.lis # Or all *.lis files
ln -s {exp}.vix {EXP}.vix
```

Open the *lis file* and modify any incorrect entries. Check the first line to be sure that j2ms2 will read the correct *vix* file and will save all the data into the correct place (typically *{exp}.ms*). This may need to be changed for spectral line, multi phase center, or pulsar binning experiments, among others. These types of experiments may require multiple *lis files*, which will create different *measurement sets* (e.g. one for a continuum pass and another one for the spectral line pass). You might also change the reference station, which is used to define the frequency setup in the experiment.

Once the *lis file* is ready, get the actual data for this experiment (the correlator output for the jobs specified in the lis file):

```
getdata.pl -proj {EXP} -lis {exp}.lis
```

Convert the data into a MS file:

```
j2ms2 -v {exp}.lis
```

NOTE: A developer version of j2ms2 (and of tConvert) exists but should only be used if a new feature is required and only with great care.

```
/eee/test/bin/j2ms2 -v {exp}.lis
```

For now this dev version must be used for e.g. experiments with multiple FREQIDs.

In case of multi-phase center or pulsar binning experiments

Different measurement sets (MS) must be created for the different phase centers (applicable to pulsar bin data too, or spectral line experiments with more than one pass). The new version of j2ms2 manages the different sources or freq. configuration and the output MS will only include the correct metadata.

Inspect the data (Jplotter)

Use jplotter (or jiveplotter) to make standard plots:

```
standardplots -weight [-scan {scan_no}] {exp}.ms {refant} {calsrcs}
```

Where `scan_no` *optionally* specifies a reference scan to produce the auto and cross correlation plots instead of picking them automatically, `refant` is the reference antenna to be used in the plots, and `calsrcs` is the name of the calibrator source (one or more) to be picked in the scan selection. Run with `-h` for help. It will create the following plots that you must look at (e.g. with *gv*):

- *{exp}-weight.ps*: Weights vs time per antenna. Check if all antennas show weights of ~1 (as expected) or lower ones. Determine a good weight threshold to flag the data in following steps (typically 0.9 unless some station shows lower weights most of the times).
- *{exp}-auto-{n}.ps* (typically *n=1,2*, for two different scans): Autocorrelations (amplitude vs channel) for each antenna. Check that all antennas show good bandpasses (ideally flat across the band). Note that observations at L band may show RFI.
- *{exp}-cross-{n}.ps*: Cross-correlations (amplitude and phase versus channel) for each baseline to the reference antenna. Check that fringes are found (you see clear amplitudes and phases) and the polarizations are OK for all antennas (e.g. no swapped polarizations, and similar signal in both of them).
- *{exp}-ampphase-{n}.ps*: Amplitude and phase versus time (where *n=0* refers to the whole time range and *n=1* is a zoom during the first observing hour) for all baselines to the reference antenna. You can check how is the behavior of each antenna along the whole experiment (e.g. there may be times where an antenna dropped, or lost fringes).

In case *standardplots* does not produce the desired output run everything manually (required at the moment):

```
> jplotter

ms {exp}.ms
r
# Shows the frequency setup, stations, sources, and times

# Plot of the weights vs time
bl auto;fq */p;sort bl sb;pt wt;ckey sb;ptsz 4;pl
save {exp}-weight.ps

# Plot amp and phase vs time
bl Ef* -auto;fq 5/p;ch 0.1*last:0.9*last;avc vector;nxy 1 4;pt anptime;ckey
src;y local;ptsz 2;time none;pl
or:
bl Ef* -auto;fq 4/p;ch 0.1*last:0.9*last;avc vector;nxy 1 4;pt anptime;ckey
p['RR']=2 p['LL']=3;y local; ptsz 2;sort bl;

save {exp}-ampphase-1.ps

# For example another plot but only showing 1 h of data
time 2017/06/21/05:00:00 to +1h
or
time $start to +2h

save {exp}-ampphase-2.ps


# auto-corr plots
indexr
listr
# shows you all the scans. Pick one containing a strong source
scan 1;pt ampchan;bl auto;fq */p;ch none;avt vector;avc none;ckey p['RR']=2
p['LL']=3;sort bl sb;new sb false;multi true;y 0 1.6;nxy 1 4;pl

save {exp}-auto-1.ps

# Pick other scan and save it as -2.ps
scan 99;pl
save {exp}-auto-2.ps

# cross-corr plots
scan 1;bl Ys* -auto;pt anpchan;fq *;ckey p['RR']=2 p['LL']=3 p['RL']=4
p['LR']=5;y local;draw lines points;ptsz 2;multi true;new sb false;sort
bl;nxy 2 3; filter phase: p in [LL,RR];pl
save {exp}-cross-1.ps

# Pick other scan and save it as -2.ps
scan 99;pl
```

```
save {exp}-cross-2.ps

exit
```

Fixing the MS files

In case Yebes (or Hobart from the LBA) is in the array, run this script to fix its mount type in the MS file:

```
ysfocus.py {exp}.ms
```

Flag weights below the specified threshold (typically 0.9, but check the weight plot from before):

```
flag_weights.py {exp}.ms {threshold}
```

Take note of how much data have been flagged, and add this information to the PI letter when required.

In the cases where a station has swapped polarizations, it needs to be fixed by using:

```
polswap.py {exp}.ms {antenna}
```

Optionally you can provide a time range for which the swap must be applied, by using the optional parameters `-t1 STARTTIME` and/or `-t2 ENDTIME`.

In the cases where a station recorded 1-bit data instead of the usual 2-bits (at the moment it should only happen in RadioAstron observations), then the data must be scaled to the usual 2 bits to correct for quantization losses. In that case use:

```
scale1bit.py {exp}.ms {antenna}
```

Optionally you can also scale the weights with the parameter `-w`.

NOTE: in case of one (or multiple) stations recording linear polarizations, you need to convert that into circular ones. However, this will be done after creating the IDI FITS files.

Update the PI letter

Open the PI letter (*{exp}.piletter*) and put all relevant information on it:

- Change the acknowledge section removing the trailing letter of the experiment name if it exists (there is a comment in the file reminding you about it).
- Write down the weight threshold that has been used.
- Add how much data have been flagged in `flag_weights.py`.
- Per station, quote the issues that may have appeared during observation, correlation or during the previous steps of post-processing.
- Note that after pipelining the experiment you may update this file again if new issues have raised from the data.

Creating FITS IDI files

Convert the data to IDI files and archive them:

```
tConvert {exp}.ms {exp}_1_1.IDI
```

The numbers in the IDI file are as `{exp}_{corr}_{pass}.IDI` where:

- *corr* (1..*n*) refers to different correlations, e.g. continuum and spectral line, or different phase centers, pulsar bins.
- *pass* (1..*n*, typically only 1) refers to the number of correlation passes, appropriate for experiments using 2-heads or very many lags/subbands that cannot be correlated in one go, but generally can be 'glued' back together using VBGLU in AIPS.

Converting linear polarizations into circulars (if required)

In some experiments, you will notice that one (or multiple) stations recorded data with linear polarizations, instead of circular ones. EVN observations are always conducted with circular polarization, thus the data from these stations must be converted.

Currently there is a program (still under development) based on CASA that can convert the different polarizations: **PolConvert**. It is being written by Iván Martí-Vidal (and contributions from Mark Kettenis) mainly meant for ALMA data within EHT observations. This program basically derive the required polarization conversion based on data recorded in a bright source (e.g. one of the scans on a fringe-finder), and then applies the conversion to the rest of the dataset. Currently, it only works with the FITS IDI files (so data from the Measurement Set files cannot be corrected).

The way this program is called will change in a (near) future to make it much more user-friendly, but currently these are the steps that need to be done (and afterwards I detail how to do them):

- You need to find a good time range to be used by PolConvert to compute the required conversion. This means that you need to pick a scan (or part of it, typically with a few

- minutes of data is enough) on a strong source, where both a reference (good behaving) station and the one that needs to be fixed worked properly.
- Then you need to find in which IDI FITS file that scan has been recorded.
- Prepare the input script to run PolConvert with all required parameters.
- Run PolConvert. It will show you different plots with the data before correction and the data after correction. The last plots should clearly show the expected behavior for the resulting circular polarizations (i.e. high amplitudes for the direct hands between the two stations, RR and LL, and almost no signal for the crossed polarizations, RL and LR). It will write new FITS IDI files with the corrected data (you also have the option to overwrite the data, but this approach is safer in case PolConvert could not fix the data).
- If you produced new FITS IDI and you have checked that the conversion worked properly, then you can delete the original IDI files and rename the new ones with the classical name.

For the first couple of steps, we have created a script that directly tell you the FITS IDI file that contains the data that you want to use for the conversion. You just need to run:

```
find_idi_with_time.py  {IDI_files}  {YYYY/mm/dd/HH:MM}
```

where the first argument contains all IDI files that need to be searched for (e.g. type *{exp}_1_1.IDI** to search along all these files), and the second argument specifies the time stamp that will be searched in the different files (e.g. put a time contained in the data that will be used for the calibration.

This script will tell you the FITS IDI file that contains that time stamp.

Then you can copy one of the input files as template to your current directory:

```
cp ~/polconvert/PolConvert_calibrate_EVN.py .
```

Open this file and modify the first part, where all parameters are defined:

- **REFANT**: the number of the antenna to be used as reference (in the FITS IDI order which is the same as in the MS file plus one, and you should already know at least from the *standardplots* output).
- **LINANT**: the station to be fixed (the one that recorded linear polarization). In case of more than one you will need to run the script more than once.
- **REF_IDI**: type the name of the FITS IDI file that contains the data to be used to compute the required conversion (e.g. the one that `find_idi_with_time.py` told you).
- **CALRANGE**: specify the time range to be used to compute the required conversion (in AIPS format).
- **NCHAN, NIF**: number of channels and number of IFs in the data.
- **NITER**: number of iterations to run. By default it is two.
- **ALL_IDIs**: you need to set a list with all FITS IDI files that require the conversion (typically all FITS IDI files produced in this experiment).
- **SUFFIX**: The suffix that will be written for the output (corrected) FITS IDI files. If empty then it will overwrite the files.

Once you have everything ready you can run PolConvert by typing:

```
casa -c PolConvert_calibrate_EVN.py
```

Check the plots that will be shown and verify that everything run properly. If you are happy with the results, then you can remove the original FITS IDI files and then rename the new ones to match the original ones. For example with:

```
rename 's/.POLCONVERT//' {exp}_1_1.IDI*.POLCONVERT
```

In case of spectral line experiments. The previous steps work when you need to apply the conversion to the same data set. In the cases when you need to transfer the conversion from a given data set (e.g. continuum pass) to a different data set (e.g. spectral line pass), you need to copy the input file `~/polconvert/PolConvert_transfer_EVN.py` to your current directory and modify/run it in a similar way as than in the previous steps.

Archiving the experiment

NOTE The password protection is not needed for NMEs or for those experiments where the PI has waved proprietary period.

```
date | md5sum | cut -b 1-12 # Quick way to create a "secure" password

touch {exp}_{password}.auth # It is recomended to have an empty file in the
directory with this info

archive -auth -e {exp}_{YYMMDD} -n {exp} -p {password}
archive -fits -e {exp}_{YYMMDD} *IDI*

# In case it is necessary (experiments not submitted through NorthStar).
Typically Bob takes care:
archive -abstract {abstract.txt} -e {exp}_{YYMMDD}

# Modify the PI letter accordingly and archive it together with the
standard plots:
gzip *ps
archive -stnd -e {exp}_{YYMMDD} {exp}.piletter *ps.gz
```

Note that the PI letter may need to be updated after checking the pipeline results.

Pipeline the data

Pipe (jop83)

The *pipe* user in *jop83* loads the two directories where you will work:

- $IN: where the inputs for the EVN pipeline are located (/jop83_0/pipe/in).
- $OUT: where the outputs for the EVN pipeline are located (/jop83_0/pipe/out).

```
ssh -Y pipe@jop83

mkdir $IN/{exp}
mkdir $OUT/{exp}

mkdir $IN/{username}/{exp}
cd !$
```

This last folder under your *username* path is recommended to download all antab and log files from *vlbeer* and fix them before running the pipeline. Copy all the ANTAB, LOG and UVFLG files to this folder from *vlbeer*:

```
sftp evn@vlbeer.ira.inaf.it
cd vlbi_arch/{monthYY}
mget {exp}*.log {exp}*.antabfs {exp}*.uvflgfs
```

Note: There is an alias on pipe called *sftpvlbeer* that runs the sftp command.

ANTAB files

If some antennas have not uploaded the *antabfs* file to vlbeer yet, request it by email. You may need to create a nominal file sometimes. Note that some special stations (such as Arecibo or the KVN) can send files with a slightly different names.

Once you have all existing *antabfs* files with the correct name in your working directory, you can run the graphical tool to check, improve, or create the missing ANTAB files. Just run:

```
# Being at $IN/{username}/{exp}.  It will understand you are processing
{exp}
antab_editor.py

# Optional parameters
antab_editor.py -e {EXPERIMENT} -l -a EXP2 [EXP3 ...]
# -l (or --spectralline) for spectral line experiments
# -a EXP2 [EXP3 ...] for check the FITS IDI files from other experiments
when loading the
# current one. Useful for e-EVN runs, when an ANTAB file is produced for
multiple experiments
# observed with the same vexfile named for EXPERIMENT (exp).
```

It will open a window where you can look at the gain curves (check if they are plausible), edit the ANTAB files from all telescopes, or create nominal ones in case some are missing. Once you are done, in the first tab you can save the individual (corrected) ANTAB files for each telescope (they will be stored with the extension `.editor` to distinguish them from the original ones), or directly save the combined `{exp}.antab` file.

Additional remarks:

- `antab_editor.py` will read the existing FITS IDI files to verify that you have Tsys measurements for, at least, each scan. It will take a bit but the information will be stored so doesn't need to be read the following times the program is open.
- For spectral line experiments you need to use the `-l` (or `--spectralline`) option. This will produce the two expected ANTAB files by default when creating the final file. One for the continuum part and another one for the spectral line one.
- For e-EVN experiments, bullet point #1 could miss the data from the additional experiments that were observed in the same run. You can consider the files from these ones by using the `-a` option.

DEPRECATED (only the nominal could still be required under very special circunstances, or VLBA part)

Creating an *antab* file from the *log* file:

For those stations that do not have an *antabfs* file, there are different possibilities:

Try with the `antabfs.pl` script located in jops@jop83:/jop83_0/pipe/in/agudo/antabfs:

```
antabfs.pl {logfile}
```

In a non-interactive more, add `-batch`.

Go to *eee*, where you must have a copy of the log file, and then run:

```
antabfs.py /data0/tsys/rxg_files/{calANTBAND.rxg} {logfile}
```

which is the script that stations currently use (maintained by Yebes).

For stations with continuous Tsys measurements:

You can use either `antabfs.py`, as before, or use the data that went through the correlator with (also in *eee*:

```
tsys.py -r /data0/tsys/rxg_files/{calANTBAND.rxg} -l {exp}.lis -e {STATION}
> {exp}{station}.antabfs
```

Note that you can always use `-h` to get an explanation of the parameters to use. `-I -c` can be typically needed.

Nominal ANTAB files:

If it fails, then create an ANTAB file with nominal values (taken from the EVN Status Table) on *pipe*:

```
antabfs_nominal.py  -d  {DURATION_IN_HOURS}  -b  {BAND_IN_CM}  {ANTENNA}
{EXPERIMENT} {DOY/HH:MM}
```

Note that you can always use `-h` to get an explanation of the paramenters to use.

Interpolating ANTAB files:

Recently we have noticed that some stations (typically the Russians and Ur) produce antab files with measurements only once every several minutes. As all these measurements are typically recorded during calibrator scans or during gaps, all the other scans (likely on the targets) are typically flagged in the pipeline (the current AIPS Pipeline requires to have at least one Tsys measurement per scan, otherwise it is flagged). To avoid this issue, you can run the following script (in *pipe*) on those antabfs files to fix the issue. It will interpolate Tsys values from the given ones:

```
antabfs_interpolate.py [-p] {antab_file} {interval}
```

This script interpolates the measurements every `{interval}` seconds. Typical values are 15-30 s. The `-p` flag is optional but recommended. It will produce plots for each IF/polarization showing the original values and the interpolation that has been done. Check that the interpolation worked properly and ignored outliers. If no more options are provided, it will replace the original `{antab_file}`. But you may provide `-o NEWFILE` which will write the output into *NEWFILE*. Specific start and ending times can be also provided, see the help with `-h`.

In the case of globals (VLBA):

- Login to *eee*.
- Go to your experiment directory.
- You will need to copy the the *vlba_gains.key* file located in */data0/tsys/*.
- Then you can produce the antabfs file for each station through:

```
tsys.py -l {exp}.lis -i 15 -t /data0/tsys/{Freq.station-file} {StationCode}
>> {exp}.antabfs
```

Repeat it for each station (will be appended to the previous `{exp}.antabfs` file created). Or use different antab files for the different VLBA stations. **NOTE**: if you have the GBT, the script might not copy correctly the header (gain information) to the antabfs. Do it manually getting the information from the */data0/tsys/gbt_gains.key* file.

Log files from the VLBA web: http://www.vlba.nrao.edu/astro/VOBS/astronomy/ Download the *{exp}cal.vlba* file, which contains the flagtables, the antab, and the weather information. All these files should also be at *ccs* under /ccc/var/log2vex/logexp_data/{EXP}_{YYYYMMDD}/.

Checking ANTAB files:

Perform some sanity checking of the ANTAB information, by running:

```
antab_check.py [-h] [-n] [-s STATION]
```

By default it will take all uvflgfs/antabfs files present in the current directory. You can specify a particular station with `-s` . `-n` disables the option of making plots. `-h` to see the help and all the options.

A careful look at all these files is mandatory. Pay attention to the frequency ranges (FREQ) and column indexes from the header of the files, and look at suspicious numbers.

uvflg files

If there are *uvflgfs* files missing, create them from the *log* files in *pipe*:

```
uvflgall.sh

# Or individually:
uvflg.pl {exp}{ant}.log
```

Check that all the *uvflgfs* files have roughly the same size. If there is one much smaller, this is likely because there was a problem with the log file interpretation.

DEPRECATED (uvflgall.sh does this automatically) Sometimes it is due to the number located around the 7th row: *'flagr,0'*. Change that number with one normally used in the other stations (400, 700, …, measured in ms). Create again the *uvflgfs* file.

Concatenate all the *antabfs* and *uvflgfs* files into `{exp}.antab` and `{exp}.uvflg`, respectively (e.g. by typing *cat {exp}* antabfs > {exp}.antab* and *cat {exp}* uvflgfs > {exp}.uvflg*).

In case of spectral line experiments, copy the same uvflg table as {exp}_2.uvflg and {exp}_1.uvflg. Copy also the antab file, and leave only the IF (for both polarizations) which is selected for the line pass in the INDEX line for each station.

Run the EVN pipeline

Prepare the `{exp}.inp.txt` file (based on an older one for example) in the *$IN/{exp}* folder. You can copy the template from a previous experiment or from *$IN/template.inp*.

Run the EVN pipeline.

```
EVN.py {exp}.inp.txt
```

Numbers for the tmask parameter:

1. Load and sort the data
2. A-priori data flagging
3. Plot the raw data
4. Amplitude calibration and parallactic angle correction
5. Fringe-fitting

6. Bandpass calibration
7. Plot the results after ampcal, fringe fitting and bandpass.
8. Split
9. Create multi files and make dirty maps and first clean maps.
10. Continue mapping
11. Plot the final data
12. Calculate the antenna sensitivities
13. Save useful data and plot final map

All the output files are saved in the `$OUT/{exp}` directory.

If the pipeline breaks at the beginning (check *$OUT/{exp}/AIPS.LOG* and the ParselTongue output), a typical reason is that for some antab files, the given frequency range is smaller than the actual recorded frequency range. That implies missing data for some channel/IF. Check the values given for each antenna (the log file from the pipeline will tell you what is happening).

Provide the feedback for the experiment

There are a couple of text files to be created before archiving the pipeline: the *{exp}.tasav.txt* file in the *$IN/{exp}* directory, and the *{exp}.comment* file in the *$OUT/{exp}* directory.

Traditionally, both files have been created by copying them from previous experiments and manually modifying them. Now you can use the following script to generate both of them:

```
comment_tasav_file.py '{exp}'
```

where the script can be run from any location, and `{exp}` must be the experiment name pass (i.e. {exp}, or likely {exp}_1 for continuum passes, or {exp}_2 for spectral line passes). Note that the name is case insensitive, you can write {exp} or {EXP} and get the same behavior. **Please check the comment file carefully and check the values written there (specially the frequency setup). It could provide the wrong values in a few particular cases.** By default it will write the files in the appropriate directory (under $IN for the tasav file, and under $OUT for the comment file).

Then run the feedback script:

```
feedback.pl -exp '{exp}' -jss '{your_username}'
```

If necessary, add the `-source 'source1 source2 ...'` option.

Check the *html* file in the browser.

Check the EVN Pipeline output

Open the generated *html* file from the EVN Pipeline and check the full output. Among other plots, please consider checking in detail the following sections:

- *Plots of the uncalibrated amplitude and phase against frequency channel*. Do you see fringes for all stations (at least for the fringe finders, or phase calibrator?
- *Telescope sensitivities*. Check that there is Tsys information for all participating stations, for all time ranges, and all observed subbands.
- *Fringe-fit (delay and rate) solutions*. Check that all stations got consistent solutions.
- *Telescope bandpasses*. Check that the bandpasses are correct (roughly flat across the band).
- *Calibrated amplitude and phase against time*. Do all stations have data? Did you miss any of them?
- *Calibrated amplitude and phase against frequency*. Do you see flat phases? And amplitudes?
- *Statistical summary* (from the amplitude corrections applied to the fringe finders). Do all stations have corrections around one with low median errors? If some stations exhibit large correction factors you may want to take a closer look to the data and the ANTAB files to fix it.
- *Telescope sensitivities*. How are the gains applied to all stations/IFs along the time? Did some stations drop data?

If some antenna has a big gain correction (you can also verify that by running 'gscale' in Difmap on one of the fringe-finders datasets), then modify it in the *antab* file: Search for "GAIN {ANT}" and some lines below change the "FT=" value to the square of the obtained correction.

Update the PI letter (*{exp}.piletter* in *eee*) if you notice issues that should be mentioned. And archive again the file.

Archive the pipeline results

Update the authentification in the webpage from: http://archive.jive.nl/scripts/pipe/admin.php Do not allow free access for the sources as specified by the PI in the *{exp}.expsum* file (copied previously in the working directory in *eee*).

Archive both directories, `$IN` and `$OUT` (you need to be the *jops* user):

```
su jops
cd $OUT/{exp}
archive -pipe -e {exp}_{YYMMDD}

cd $IN/{exp}
archive -pipe -e {exp}_{YYMMDD}

exit
```

Copy the gain corrections from each station done by the pipeline to the database. In *pipe* run:

```
cd $OUT/{exp}/
ampcal.sh
```

In case of e-EVN experiments, you need to add the option `-e {exp_name}`, where *exp_name* is the official experiment name of the e-EVN run (basically the name of the first observed experiment).

Post-processing steps

Send the emails to contact PIs

- PI letter to the PI and *jops@jive.eu* (this can be done just after archiving the results and before running the pipeline, but we typically wait until running the pipeline to have the full picture of what happened during the observations).

- Pipe letter to the PI only containing the credentials to access the data. You can generate this file by using (in eee):

  ```
  pipelet.py [-h] {exp} {jss}
  ```

  where jss is the surname of the Support Scientist in charge of this experiment (i.e. you). It takes the credentials to access the data from the `username_password.auth` file that should be in the current directory. Otherwise you can pass the information through different paramenters (see help with `-h`).

Add the PI letter information to the database

Import the PI letter comments to the database to keep track of all the important problems in the session:

In the directory where the `{exp}.piletter` is located (in *eee*), run:

```
parsePIletter.py -s SESSION {exp}.piletter
```

where SESSION is the session name in the format xxxYY (e.g. oct17). For regular EVN sessions, use only Feb, Jun, or Oct. As usual, `-h` will show you the help.

Tracking of station issues per session

It is convenient to **report the major issues that you have seen** in the JIVE Wiki to track the issues related to particular stations that are happening in one EVN session or along the time. Please, go to the Wiki list of problems:

http://www.jive.eu/jivewiki/doku.php?id=evn:supportscientists#problem_list_per_session

And add a summary of important issues you have identified in stations from this experiment (note that the problems are listed per session and per band).

Additionally, please also report all problems you have spotted concerning the ANTAB files in this other page of the JIVE Wiki:

http://www.jive.nl/jivewiki/doku.php?id=evn:supportscientists:antabs

The idea is to keep track of both, issues inside the ANTAB files and stations that do not send ANTAB files or do not upload them after observations.

House-keeping

After an experiment has been distributed and the usual period of two weeks waiting for feedback from the PI has expired, then you can remove all the files created in AIPS to free disk space (as all those results are already archived). Same approach for all IDI files located in *eee*. They can be safely removed as a copy of them lies in the EVN Data Archive. Measurement sets should be retained until backed up to tape (by Bob).

NME Report

If you are post-processing a NME, then you need to write the NME Report.

This is the website in the Wiki where all the NME Reports are stored: http://www.jive.eu/jivewiki/doku.php?id=evntog:nme_reports

There you have a LaTeX template and you can upload it to the corresponding place.

However, I created a script `create_nme_report_template.py` to produce the LaTeX template with the correct structure for your NME (e.g. the table will show all the stations that participated). This will save you time modifying the table (just run with a `-h` to know the necessary inputs).

If you do not have this script, download it from the Wiki webpage: http://www.jive.nl/jivewiki/doku.php?id=evn:supportscientists

Finally, send an email to EVNTech (evntech@jive.eu) informing everyone that the Report has been uploaded.