Obit Development Memo 29
EVLA Continuum Scripts:
Outline of Data Reduction and Heuristics

Bill Cotton
November 13, 2019

1 Introduction

1.1 Scope

The scope of the present version of the EVLA continuum scripts is to perform standard calibration and editing of EVLA data and produce continuum, possibly wideband, images of target sources. Logs, reports and numerous diagnostic plots help evaluate the results of the processing. If default processing parameters are adequate, the scripts will start from EVLA archive ASDM/BDF files and result in FITS images, calibrated data, reports, plots etc. The scripting is also capable of being highly tuned to a particular project and can be rerun in whole or part with user specified parameters.

1.2 Software

The EVLA Obit scripts are:

- Written in python, and
- Use Obit and AIPS tasks to do the data processing, and
- Use AIPS data structures for intermediate data, and
- Write FITS images and (AIPS FITAB format) calibrated datasets.

AIPS (http://www.aips.nrao.edu/index.shtml) and
Obit (http://www.cv.nrao.edu/ bcotton/Obit.html) are installed on all NRAO Linux computers and available for installation via download to non-NRAO computers. The EVLA scripts are in the $OBIT/python directory with the template parameter script in $OBIT/share/scripts. A binary Linux distribution is available.

2 Execution

Several steps are needed to execute the EVLA scripts.

2.1 Generate parameter scripts

The processing is guided by values in python parameter scripts. These scripts can be created and initialized by information gleaned from the EVLA archive ALMA Science Data Model (ASDM) files using routine EVLACal.EVLAPrepare (see section 6). This will create one or more parameter files, each of which needs to be processed separately.

Alternatively, the parameter file can be derived manually using the template file $OBIT/share/scripts/EVLAContTemplateParm.py and making the substitutions described in the file.

2.2 Modify parameter scripts

If the default values in the automatically generated parameter script(s) are not appropriate, they can be changed. The details of each processing step and the parameters used are described in Section 6. Default parameters and control switches can be overridden in the parameter scripts. Additional calibrator model information can be entered as described in section 3. The end of the parameter script contains switches to turn on and off various stages.

2.3 AIPS and Obit setup scripts

A script needs to be created giving the details of the AIPS and Obit installations. This script is described in detail in Section 4.

2.4 Execute scripts

Each of the parameter scripts can be executed from the Unix shell by

```
> ObitTalk EVLAContPipe.py AIPSSetup.py \
     EVLAContParm_myProject_Cfg0_Nch64.py
```

where EVLAContParm_myProject_Cfg0_Nch64.py is the name of your parameter script.

This procedure should start from an archive data set and result in a set of calibrated data, images, reports, logs and various diagnostic plots, see Section 7 for details.

3 Calibrator models

The standard EVLA flux density calibrators are all resolved in the more extended configurations and an accurate, wideband source model at an appropriate frequency is needed for good calibration. Calibrator source models using the Obit CLEAN components with spectra are distributed in $OBIT/share/data and should be copied to the first FITS directory and gunzipped.

The EVLA calibration scripts operate on arrays of calibrator dict structures with the following entries:

- `Source:` Source name as given in the SU table.
- `CalName:` Calibrator model Cleaned AIPS map name
- `CalClass:` Calibrator model Cleaned AIPS map class
- `CalSeq:` Calibrator model Cleaned AIPS map seq
- `CalDisk:` Calibrator model Cleaned AIPS map disk
- `CalNfield:` Calibrator model No. maps to use for model
- `CalCCVer:` Calibrator model CC file version
- `CalBComp:` Calibrator model First CLEAN comp to use, 1/field
- `CalEComp:` Calibrator model Last CLEAN comp to use, 0=>all
- `CalCmethod:` Calibrator model Modeling method, 'DFT','GRID',' '
- `CalCmodel:` Calibrator model Model type: 'COMP','IMAG'
- `CalFlux:` Calibrator model Lowest CC component used
- `CalModelSI:` Calibrator Spectral index
- `CalModelFlux:` Parameterized model flux density (Jy)
- `CalModelPos:` Parameterized model position offset (asec)
- `CalModelParm:` Parameterized model parameters (maj, min, pa, type)
- `useSetJy:` Standard spectrum (from SetJy) flux density used.

These dicts are created in the parameter script by routine EVLACal.EVLACalModel for the various types of calibrators. EVLACal.EVLAStdModel is then used to fill in the details about calibrators it knows about and can find in the first FITS directory. Information not known to these scripts may be entered into the calibrator dict structure in the parameter script.

4 AIPS and Obit Setup

These scripts use data in AIPS format and some AIPS tasks; the location of the AIPS data directories and other details as well as the Obit initialization are given in the AIPSSetup.py file. An example is given in $OBIT/share/scripts/AIPSSetup.py. The items that need to be specified are:

- `adirs`
  A list of the AIPS data directories as a tuple, the first element is the URL of the ObitTalkServer or None for local disk. The second element is the directory path.

- `fdirs`
  A list of the FITS data directories as a tuple, the first element is the URL of the ObitTalkServer or None for local disk. The second element is the directory path.

- `user`
  The AIPS user number to be used.

- `AIPS_ROOT`
  The root of the AIPS system directories. An environment variable of this name is set by the AIPS startup scripts. Python None will default to your AIPS setup.

- `AIPS_VERSION`
  The AIPS version. An environment variable of this name is set by the AIPS startup scripts. Python None will default to your AIPS setup.

- `DA00`
  The AIPS DA00 directory (TDD000004; file needed). An environment variable of this name is set by the AIPS startup scripts. Python None will default to your AIPS setup.

- `OBIT_EXEC`
  The root directory of your Obit directories. Python None will default to your system installation on NRAO Linux machines.

- `noScrat`
  A list of AIPS disks to avoid for scratch files, max. 10.

- `nThreads`
  The maximum number of threads allowed to be used. This generally should not be more than the number of cores available.

- `disk`
  The AIPS disk number to use for temporary storage of the data and images.

An example AIPSSetup.py file follows, items which may need to be modified are marked by `<====`.:

```
#   <====  Define AIPS  and FITS disks
global AIPS_ROOT, AIPS_VERSION, DA00, OBIT_EXEC
adirs = [(None, "/export/data_1/GOLLUM_1"),
         (None, "/export/data_1/GOLLUM_2"),
         (None, "/export/data_1/GOLLUM_3"),
         (None, "/export/data_1/GOLLUM_4"),
         (None, "/export/data_2/GOLLUM_5"),
         (None, "/export/data_2/GOLLUM_6"),
         (None, "/export/data_2/GOLLUM_7"),
         (None, "/export/data_2/GOLLUM_8")]
fdirs = [(None, "/export/users/aips/FITS")]

############################## Initialize OBIT ####################
global noScrat, nThreads, disk, err, user
err     = OErr.OErr()
user    = 104                       # <==== set user number
ObitSys = OSystem.OSystem ("Script", 1, user, 0, [" "], \
                           0, [" "], True, False, err)
OErr.printErrMsg(err, "Error with Obit startup")
# Setup AIPS
AIPS.userno = user
AIPS_ROOT    = "/home/AIPS/"        # <==== set root of AIPS
AIPS_VERSION = "31DEC19/"           # <==== set AIPS version
DA00         = "/home/AIPS/DA00/"   # <==== set AIPS DA00 directory
```

```
#  <====  Define OBIT_EXEC for access to Obit Software
OBIT_EXEC    = "/export/data_1/obit/ObitInstall/ObitSystem/Obit/"
# setup environment
ObitTalkUtil.SetEnviron(AIPS_ROOT=AIPS_ROOT, AIPS_VERSION=AIPS_VERSION, \
                        OBIT_EXEC=OBIT_EXEC, DA00=DA00, ARCH="LNX63", \
                        aipsdirs=adirs, fitsdirs=fdirs)
# List directories
ObitTalkUtil.ListAIPSDirs()
ObitTalkUtil.ListFITSDirs()
noScrat     = []                    # <==== AIPS disks to avoid
nThreads    = 6                     # <====  Number of threads allowed
disk        = 1                     # <====  AIPS disk number
```

5 The Process Overview

The scripted processing uses the following processes. Many of the default processing parameters are frequency dependent and may be overridden and the various steps may be turned on or off.

The general approach to calibration and editing is to first apply editing steps which can be applied to uncalibrated data to remove the most serious RFI and equiptment failures. Then an initial pass at calibration is done and a pass at the editing needing calibrated data. Calibration aids in the editing as calibrator data with no detections are effectively removed and calibration with deviant amplitude solutions are also removed. Once the first pass at editing and calibration is completed, the initial calibration tables are deleted and the calibration repeated. This procedure removes the bulk of the RFI infected and other bad data.

Diagnostic plots at various stages of the processing are generated. These include plots of calibration results as well as sample spectra.

The calibrated data are then imaged, using wideband imaging if appropriate. Images, calibrated data and calibration tables are saved to FITS files and a number of source dependent diagnostic plots are generated. Finally, an HTML report is generated allowing easy examination and access to the various products. A processing log is kept containing most details of the processing.

Following is a summary of the processing. Details and parameters which may be modified are described in a section 6.

1. Generation of parameter scripts from ASDM

2. Data converted to AIPS format
3. Hanning if necessary
4. Clear previous calibration
5. Copy initial FG table
6. Flag end channels
7. Apply Special Editing
8. Quack
9. Shadow Flagging
10. Initial Time domain flagging
11. Initial Frequency domain flagging
12. Initial RMS flagging of calibrators
13. Amplitude calibration based on switched power measurements
14. Parallactic angle correction
15. Find reference antenna
16. Plot raw sample spectra
17. Delay calibration
18. Bandpass calibration
19. Amp & phase Calibration
20. Flagging of calibrated data
21. Recalibration
    (a) Switched power cal
    (b) Parallactic angle correction
    (c) Delay calibration
    (d) Bandpass calibration
    (e) Amp & phase Calibration

(f) Flagging of calibrated data

22. Calibrate and average data
23. Cross Pol clipping
24. R-L delay calibration
25. Instrumental polarization calibration
26. R-L phase calibration
27. VPol clipping
28. Plot final calibrated spectra
29. Image targets
30. Generate source report
31. Save images, calibrated data
32. Contour plots of images
33. source UV diagnostic plots
34. Generate HTML summary
35. Cleanup AIPS directories

6 Script Stage Details

Details of the various processing steps and the parameters are described in the following. Processing parameters are stored in a python dict named parms and may be specified in the parameter script as

```
parms["parameter"] = value
```

Tables in the following give the parameter name, default value and a description.

1. Generation of parameter scripts from ASDM
   This step is performed from ObitTalk to generate the parameter script(s).

```
>>> import EVLACal
>>> ASDMRoot = "/export/myData/12A-999.sb9332941.eb9360588.../"
>>> EVLACal.EVLAPrepare(ASDMRoot, err, project="myData")
```

This will parse the ASDM indicated and generate a parameter script for each configuration/number of channels combinations needed to process all data. Note: this will also include configurations used for calibration purposes only, such as offset pointing, so use Obit task ASDMList to see which configurations have useful data. For each configuration/number of channel, a parameter script with name of the form EVLAContParm_<project>_Cfg<config>_Nch<no channels>.py will be generated. The template can be in the current working directory or the default in $OBIT/share/scripts.

ASDMRoot		Root directory of ASDM/BDF data
project	??	Project name(+session should be 12 or fewer used as AIPS file name
session	??	session code
template	EVLAContTemplateParm.py	name of the parameter template file
parmFile		Name of desired parameter file, generated if not given

2. General control parameters
These parameters control the naming of files, whether UV data is used in compressed form and script debugging control.

project	??	Project name
session	??	session code, generated from configuration and no. channels
band	??	Observing band code, derived from ASDM frequencies
Compress	False	Use compressed UV data?
check	False	Only check script, don't execute tasks
debug	False	run tasks debug

3. Data converted to AIPS format
The bulk of the processing uses AIPS format UV data and images. The ASDM/BDF data is first converted to an AIPS data file using Obit/BDFIn. The details of the AIPS configuration are given in the

AIPSSetup.py file provided to the processing script. The output data file name is the Project+Session and the class is "UV"+band+"Raw" if Hanning is to be used, otherwise "UVDa"+band.

doLoadArchive	True	Load AIPS data from archive?
archRoot	??	User specified to create parameter script
selConfig	??	Frequency configuration, generated from ASDM
seq	1	AIPS sequence number to use
selBand	??	Data band-code, derived from ASDM
selChBW	-1	Selected channel bandwidth, -1 = All
selChan	??	Number of spectral channels, derived from ASDM
selNIF	??	Number of spectral windows (IFs), derived from ASDM
calInt	??	Calibration table interval (min), EVLA configuration dependent A: 0.3, B:0.45, C:1.0, D:2.0
doSwPwr	False	Make EVLA Switched power corr? now done later

4. Hanning
   At lower frequencies and compact configurations, RFI signals are frequently sufficiently strong and narrow to cause serious "Gibbs" ringing due to the truncation of the lag spectra. Hanning smoothing can be used to suppress this effect.

doHann		Apply Hanning smoothing? EVLA configuration and frequency dependent. A:False, B:$\nu < 8$ GHz, C:$\nu < 8$ GHz, D:$\nu < 8$ GHz
doDescm	True	If True, drop every other channel after smoothing

5. Clear previous calibration
   If the script is restarted it is frequently desirable that previous attempts at calibration and editing be removed.

doClearTab	True	Clear cal/edit tables?
doClearGain	True	Clear SN and CL tables > 1?
doClearFlag	True	Clear FG tables > 1?
doClearBP	True	Clear BP tables?

6. Copy initial FG table
   To allow restarting of the flagging, the on-line flags which are in FG table 1 are copied to table 2 and new flags added there. This should be turned off if the script is restarted except at the beginning.

doCopyFG	True	Copy FG 1 to FG 2?

7. Flag end channels
   The first and last few channels in each IF are flagged if FG table 2. This can be turned off by setting BChDrop and EChDrop to 0.

BChDrop	??	Number of channels to flag at the beginning If $\nu < 8$ GHz, max(2, 6.*(nchan/64) If $\nu > 8$ GHz, 3
EChDrop	??	Number of channels to flag at the end If $\nu < 8$ GHz, max(2, 4.*(nchan/64) If $\nu > 8$ GHz, 2

8. Apply Special Editing
   If some data are known to be bad, e.g. no receiver, then this information can be passed to the script. if doEditList is True, each entry is a python dict with the following:

   - `timer:` The affected time range as a pair of strings of the form day/hour:min:sec.
   - `Ant:`A baseline specification as a pair of antenna numbers, if the second is zero, then all baselines to the first antenna number is flagged. If the first is zero, then all antennas are flagged
   - `IFs:` Range (1-rel) of IFs (spectral windows) to flag. If the second is zero then all IFs higher than the first are flagged.
   - `Chans:` Range (1-rel) of channels to flag. If the second is zero then all channels higher than the first are flagged.
   - `Stokes:` Array of flags, 1=>flag, 0 => not flag; in order RR, LL, RL, LR (or XX, YY, XY, YX).
   - `"Reason:` Up to 24 characters giving reason.

   an example:

```
parms["doEditList"]  = True        # Edit using editList?
parms["editList"] = [
    {"timer":("0/00:00:0.0","5/00:00:0.0"),"Ant":[1,0],
     "IFs":[1,0],"Chans":[1,0],  "Stokes":'1111',"Reason":"No Rcvr"}
]
```

doEditList	False	Edit using editList?
editFG	2	Table to apply edit list to
editList	[ ]	List of data to flag

9. Quack
   Data at the beginning and end of each scan can be flagged using Obit/Quack.

doQuack	True	Quack data?
quackBegDrop	0.1	Time to drop from start of each scan (min)
quackEndDrop	0.0	Time to drop from end of each scan (min)
quackReason	"Quack"	Reason string

10. Shadow Flagging
    In the more compact EVLA configurations, some antennas may shadow others at times. The affected data may be flagged using Obit/UVFlag.

doShad		Do shadow flagging? Configuration Dependent A:False, B:False, C:True, D:True
shadBl	25.0	Minimum shadowing baseline (m)

11. Initial Time domain flagging
    Obit task MednFlag can be used to flag data by amplitudes deviant from a running median by more than a specified amount. This is performed independently on each data stream (baseline, channel, IF, poln). At this point that data are uncalibrated.

doMedn	True	Median editing?
mednSigma	10.0	Sigma clipping level
mednTimeWind	1.0	Window width (min) for median flagging
mednAvgTime	10.0/60.	Averaging time (min)
mednAvgFreq	0	1=>avg mednChAvg chans, 2=>avg all chan, 3=>avg chan and IFs
mednChAvg	1	Number of channels to average

12. Initial Frequency domain flagging
    The uncalibrated data can be examined for impulsive signals in frequency by comparison with a running median in each spectrum and

deviant data are flagged using Obit task AutoFlag. Since bandpass corrections have not been determined and applied at this stage, structure in the instrumental bandpass will increase the apparent RMS in the baseline reducing the sensitivity of this test. Note: this should NOT be used for data expected to have prominent spectral features.

doFD1	True	Do initial frequency domain flagging?
FD1widMW	31	Width of the initial FD median window
FD1maxRes	5.0	Clipping level in sigma
FD1TimeAvg	1.0	time averaging (min). for initial FD flagging

13. Initial RMS flagging of calibrators
Calibrators are expected to be simple and have significant SNR so can be edited by having an RMS/average amplitude of less than some amount. Discrepant calibrator data can be flagged in this step using Obit task AutoFlag.

doRMSAvg	True	Edit calibrators by RMS/Avg?
RMSAvg	3.0	Max RMS/Avg for time domain RMS filtering
RMSTimeAvg	1.0	Time averaging (min)

14. Switched power calibration
Amplitude calibration is derived from switched power measurements from the SY table and written to an SN table which is applied to the previous CL table generating a new CL table. The SY table can be clipped by deviations from a smoothed time stream and/or then smoothed before conversion to an SN table. Flagging can be determined from the derived amplitudes in the SN table relative to a running median window filter. Solution plots are written into file parms["project"]+"_"+parms["session"]+"_"+parms["band"]+"SYCal.ps".

Switched power calibration is most critical for the lower frequencies (X band and below) which are subject to strong RFI capable of changing the gain of the receivers. Very strongly affected spectral windows (SW) can have their SY calibration replaced by another, better behaved, SW using "SYSWUse". Note as of this writing (Feb 2016) SY calibration is not thought to work (says RP) for the 3 bit samplers (those with 64 SWs).

doSYCal	True	Do switched power calibration?
SYSWUse	None	A list of SW (1-rel) substitutions "None" means SWs are all used for themselves
SYcalInt	0.1	Calibration interval(min) in SN table
SYsmoFunc	"MWF"	SY smoothing function 'MWF', "BOX", "GAUS"
SYsmoTime	0.002778	SY smooth time in hrs
SYclipSmo	0.0833	smooth time for clip in hrs
SYclipParm	5	clip level (sigma) wrt running median
doSYEdit	True	Edit/flag on the basis of SY solutions
SYSigma	10.0	Multiple of median RMS about median gain to clip/flag
SYEditFG	2	FG table for editing
doSNPlot	True	Plot calibration solutions?

15. Parallactic angle correction
    Phases are corrected for the effects of the parallactic angle using Obit task CLCor. The initial CL table is copied to CL 2 and modified.

doPACor	True	Make parallactic angle correction?

16. Reference antenna
    The choice of reference antenna is of some importance but nothing in the ASDM helps in this choice. In addition, at least early EVLA data may have data for antennas with no receiver and such antennas are unsuitable for use as reference antenna. If the reference antenna is unspecified (0), this step runs Obit task Calib on the bandpass calibrator(s) (assumed to give good fringes) using the middle half of each spectrum. The resultant SN table is then examined for the antenna with the maximal amount of valid solutions and with the highest average SNR; this antenna is used as the reference antenna. Once a reference antenna is determined its value is saved in a pickle file and will be recovered on subsequent runs.

refAnt		Reference antenna, if <= 0 then determine
BPCals		Determined from the ASDM
bpsolint1		Bandpass first solution interval, configuration and frequency dependent

17. Plot Raw spectra
    At this point plots of sample spectra can be made to display residual RFI and other problematic data,

doRawSpecPlot	True	Plot diagnostic raw spectra?
plotSource		Default is first bandpass calibrator
plotTime		List of start and end time in days. Default is first bandpass calibrator scan
refAnt		Reference ant., baselines to refAnt are plotted

18. Delay calibration
    Parallel hand group delays are solved for using the list of calibrator models in DCals. Obit task Calib solves for the delays which are then smoothed and applied to all sources in a new CL table using Obit task CLCal. Solutions can be plotted as well as sample spectra applying the delay calibration.

doDelayCal	True	Determine/apply delays?
DCals		The list of delay calibrators are determined from the ASDM, all amplitude, phase and bandpass calibrator. The list of models is determined from the parameter script using standard calibrator models.
delayBChan		first channel to use in delay solutions max(2, 0.05*nchan)
delayEChan		highest channel to use in delay solutions min(nchan-2, nchan-0.05*nchan
delaySolInt		Solution interval (min), config. dependent A:2 sec, B: 5 sec, C:10 sec, D:15 sec.
refAnts	[refAnt]	Delay reference ant., baselines to refAnt are plotted
gainUVRange	[0.0,0.0]	Range of baseline used in kilolambda, zeros=all
doTwo	True	Use two baseline combinations in delay cal
delayZeroPhs	True	Zero phase in Delay solutions?
doSNPlot	True	Plot calibration solutions?
doSpecPlot	True	Plot diagnostic calibrated spectra?
plotSource		Default is first bandpass calibrator
plotTime		List of start and end time in days.

19. Bandpass calibration
Bandpass calibration uses Obit task BPass and calibrator model list BPCals. BPass does a two pass calibration, the first doing a phase only calibration to straighten out the phases followed by a longer amplitude and phase calibration using blocks of channels. The resultant solutions are then combined into a BP table

doBPCal	True	Determine/apply bandpass calibration?
BPCals		The list of bandpass calibrators is determined from ASDM, The list of models is determined from the parameter script using standard calibrator models.
bpsolint1		BPass phase correction solution in min, frequency dependent: $\nu < 1$ GHz: 10 sec, L band:15 sec, S, C, Ku, K, Ka 10 sec, Q band 5 sec.
bpsolint2	10.0	BPass bandpass solution interval (min)
bpsolMode	'A&P'	Bandpass type 'A&P', 'P', 'P!A'
bpBChan1	1	Low freq. channel, initial cal
bpEChan1	0	Highest freq channel, initial cal, 0=>all
bpBChan2	1	Low freq. channel for BP cal
bpEChan2	0	Highest freq channel for BP cal, 0=>all
bpChWid2	1	Number of channels in running mean BP soln
specIndex	-0.7	Spectral index of BP Cal
bpDoCenter1	None	Fraction of channels in 1st, overrides bpBChan1, bpEChan1
bpdoAuto	False	Use autocorrelations rather than cross?
bpUVRange	[0.0,0.0]	UV range for bandpass cal zeroes=> all
refAnt		BP reference ant., baselines to refAnt are plotted
doSpecPlot	True	Plot diagnostic calibrated spectra?
doBPPlot	True	Plot fitted bandpasses?
plotSource		Default is first bandpass calibrator
plotTime		List of start and end time in days.
doAmpEdit	True	Edit/flag on the basis of amplitude solutions
ampSigma	10.0	Multiple of median RMS about median gain to clip/flag

20. Amp & phase Calibration
Standard flux density calibrators have their flux densities entered into the SU table using Obit task SetJy, other calibrators have their flux density entries set to 1.0. All the amplitude and phase calibrators have Obit/Calib run using their models and doing amplitude and phase solutions. Solutions are then median window smoothed using Obit/SNSmo to time solSmo clipping really wild points. Obit task GetJy then solves for the flux densities for non flux density calibrators and adjusts the SU and SN tables. If doAmpEdit is True, solutions in each IF (spectral window) more than ampSigma from the mean are flagged both in the SN table and in FG table ampEditFG. Finally solutions are applied to the previous CL table to create a new CL table. Calibrator sources are "self-calibrated" and targets use the smoothed solutions. Solution plots are written into file
parms["project"]+"_"+parms["session"]+"_"+parms["band"]+"APCal.ps".

doAmpPhaseCal	True	Do amplitude and phase calibration?
ACals		The list of amplitude calibrators defaulted from ASDM, The list of models is determined from the parameter script using standard calibrator models.
PCals		The list of phase calibrators defaulted from ASDM
refAnt		Reference antenna
solInt		Solution interval (min), config. dependent: A: 2 sec, B: 5 sec, , C: 10 sec, D: 15 sec.
ampBChan		first channel to use in A&P solutions max(2, 0.05*nchan)
ampEChan		highest channel to use in A&P solutions min(nchan-2, nchan-0.05*nchan)
gainUVRange	[0.0,0.0]	Range of baseline used in kilolambda, zeros=all
solSmo	0.0	Smoothing interval for Amps (min)
ampScalar	False	Ampscalar solutions?
doAmpEdit	True	Edit/flag on the basis of amplitude solutions
ampSigma	20.0	Multiple of median RMS about median gain to clip/flag
ampEditFG	2	FG table for editing
doSNPlot	True	Plot calibration solutions?

21. Flagging of calibrated data
    Calibrated data are then edited using Obit/AutoFlag. If recalibration is to be done, then this is only on the calibrators, else all sources. Data with amplitudes outside of a given range are flagged and data overly discrepant from a running median in frequency is flagged.

doAutoFlag	True	Autoflag editing after first pass calibration?
IClip	??	AutoFlag Stokes I clipping 200 Jy for $\nu > 1$ GHz, else 20000.
minAmp	1.0e-5	Minimum allowable amplitude
timeAvg	0.33	AutoFlag time averaging in min.
doAFFD	True	do AutoFlag frequency domain flag
FDmaxAmp	IClip[0]	Maximum average amplitude (Jy)
FDmaxV	VClip[0]	Maximum average VPol amp (Jy)
FDwidMW	31	Width of the median window
FDmaxRMS	[5.0,.1]	FDmaxRMS
FDmaxRes	6.0	Max. residual flux in sigma
FDmaxResBL	6.0	Max. baseline residual
FDbaseSel	[0,0,0,0]	Channels for baseline fit

22. Recalibration
    If doRecal is true then the previous calibration tables are deleted and the calibration redone using the flag table from the first pass.

    (a) Switched power calibration
        As before if doSYCal2 = True.

    (b) Parallactic angle correction
        As before if doDelayCal2 = True.

    (c) Delay calibration
        As before if doDelayCal2 = True.

    (d) Bandpass calibration
        As before if doBPCal2 = True.

    (e) Amp & phase Calibration
        As before if doAmpPhaseCal2 = True.

    (f) Flagging of calibrated data
        As before if doAutoFlag2 = True.

23. Calibrate and average data
The calibration and editing files are then applied with possible averaging in time and/or frequency. This uses Obit/Splat which writes a multi-source file.

doCalAvg	True	Calibrate and average?
avgClass	"UVAv"	+band, AIPS class of calibrated/averaged UV data
seq	1	AIPS sequence
CalAvgTime		Time for averaging calibrated UV data (min), config dependent: A: 1 sec, B:3 sec, C: 10 sec, D: 20 sec.
avgFreq	0	0 => no averaging, 1 =>avg chAvg chans, 2 =>avg all chan, 3 =>avg chan and IFs
chAvg	1	Number of channels to average
CABChan	1	First channel to copy
CAEChan	0	Highest channel to copy, 0 => all higher than CABChan
CABIF	1	First IF to copy
CAEIF	0	Highest IF to copy, 0 => all higher than CABIF
Compress	False	Write compressed UV data?

24. Cross Pol clipping
if XClip[0] is not None, cross polarized data with amplitudes > XClip[0] are flagged.

XClip	[5.0,0.05]	AutoFlag cross-pol clipping, None=> no flagging

25. R-L delay calibration (full polarization)
Determine right–left (X-Y) delays from a list of calibrators with R–L phase and rotation measure. The function EVLACal.EVLAPrepare will search for sources in the Field table whose positions are those of calibrators with known position angle (and presumably reasonable strength polarized components); 3C286 is an example. These calibrators are added to the R-L calibrator list (parms["RLDCal"] in the parameter script). If the first source in this list is not None, and the data contains the RL and LR correlations, the R-L delay calibration is performed.

doRLDelay	??	Do R-L delay calibration?
RLDCal	??	Array of triplets of (name, R-L phase (deg at 1 GHz), RM (rad/m**2)) for polarization angle calibrators
rlBChan	1	First (1-rel) channel number
rlEChan	0	Highest channel number. 0 => high in data.
rlUVRange	[0.0,0.0]	Range of baseline used in kilo wavelengths, zeros=all
rlCalCode	' '	Calibrator code
rlDoCal	2	Apply calibration table? positive=>calibrate
rlgainUse	0	CL/SN table to apply, 0 =>highest
rltimerange	[0.0,1000.0]	Time range of data (days)
rlflagVer	2	FG table version to apply
rlrefAnt	0	Reference antenna, defaults to refAnt

26. Instrumental polarization calibration (full polarization)
Determine instrumental polarization from a list of calibrators. The function EVLACal.EVLAPrepare sets this list to those in the Delay calibrator list. Parameter parms["doPolCal"] is set True if this list is not empty and the calibration performed if the data contains the RL and LR correlations. Calibration uses Obit task PCal which determines antenna and source polarization parameters on blocks of channels in a running window. The antenna parameters are the ellipticity and orientation of the feed; see Obit Development Memo 30 for details.

doPolCal	??	Determine instrumental polarization?
PCInsCals	??	Instrumental poln cals, name or list of names
PCCalPol		if non None then the list of source parameters as tuples in the order of calibrators in PCInsCals, (PPol, RLPhase, RM) PPol = fractional poln, < 0 => fit RLPhase = R-L phase difference in deg RM = Rotation measure
PCFixPoln	False	If True, don't solve for source polarization
PCAvgIF	False	If True, average IFs (shouldn't use for EVLA)
PCSolInt	2.0	Instrumental solution interval (min), 0 => scan average(?)
PCRefAnt	0	Reference antenna, 0 => absolute solution
PCSolType	" "	Solution type, "LM " (better), " " (faster)
PCChInc	5	Channel step in spectrum
PCChWid	5	Number of channels to average
doPol	False	Apply polarization cal in subsequent calibration?
doFitRL	False	Fit R-L (or X-Y) gain phase
PDVer	1	PD table to apply in subsequent polarization cal?

27. R-L phase calibration (full polarization)
    Determine R-L phase bandpass for each channel using Obit/RLPass and calibrators in RLDCal. The function EVLACal.EVLAPrepare will search for sources in the Field table whose positions are those of calibrators with known position angle (and presumably reasonable strength polarized components); 3C286 is an example. These calibrators are added to the R-L calibrator list (parms["RLDCal"] in the parameter script). If the first source in this list is not None, and, the data contains the RL and LR correlations, the R-L phase calibration is performed.

    RLPass uses a two pass calibration, the first using Stokes I and a phase only self calibration to remove phase fluctuations; the second pass fits the R-L phase in each running block of channels. RLPass itself is run multiple times for better convergence. In each run corrections are applied to the old BP table and a new BP table is created. A spectral plot of the first RLDCal calibrator's RL and LR data are made in file parms["project"]+"_"+parms["session"]+"_"+parms["band"]+"RLSpec2.ps"

doRLCal	??	Determine R-L phases?
RLDCal	??	Array of triplets of (name, R-L phase (deg at 1 GHz), RM (rad/m**2)) for polarization BP calibrators If None then no R-L delay recalibration
rlBChan	1	First (1-rel) channel number
rlEChan	0	Highest channel number. 0 => high in data.
rlUVRange	[0.0,0.0]	Range of baseline used in kilo wavelengths, zeros=all
rlCalCode	' '	Calibrator code
rlDoCal	2	Apply calibration table? positive=>calibrate
rlgainUse	0	CL/SN table to apply, 0 =>highest
rltimerange	[0.0,1000.0]	Time range of data (days)
rlDoBand	1	If > 0 apply bandpass calibration
rlBPVer	0	BP table to apply, 0 =>highest
rlflagVer	2	FG table version to apply
rlrefAnt	0	Reference antenna, defaults to refAnt
rlChWid	3	Number of channels in running mean RL BP soln
rlsolint1	10./60	First solution interval (min), 0 => scan average
rlsolint2	10.0	Second solution interval (min)
doPol	False	Apply polarization cal?
PDVer	-1	PD table to apply?
doSpecPlot	True	If True make spectral plot.

28. VPol clipping
    If VClip is not None, data with circularly polarized amplitudes > VClip[0] are flagged.

VClip	[2.0,0.05]	AutoFlag cross-pol clipping, None=> no flagging

29. Plot final calibrated spectra
    At this point, plots of sample spectra can be made to display calibrated data.

doSpecPlot	True	Plot diagnostic spectra?
plotSource		Default is first bandpass calibrator
plotTime		List of start and end time in days.
refAnt		Reference ant., baselines to refAnt are plotted

30. Image targets
    All targets are imaged and deconvolved using Obit/Imager or Obit/MFImage if wideband imaging needed (fractional spanned bandwidth $\geq$ MBmaxFBW). Phase only and amp and phase calibration may be applied if sources exceed given thresholds. If wideband imaging is used, then the resultant images are cubes having planes:
    (a) Total intensity at reference frequency.
    (b) Spectral index at reference frequency
    (c) any higher order planes
    (d) One plane for each of the coarse frequency samples.

doImage	True	Image targets?
targets	[?]	Target list set from ASDM, empty=>all
seq	1	AIPS sequence for images
doPol	True	Apply polarization cal?
PDVer	1	PD table to apply
outIclass	"IClean"	Image AIPS class
Stokes	"I"	Stokes to image
Robust	0.0	Weighting robust parameter
FOV		Field of view radius in deg, average ν dependent:
		$\nu < 1$ GHz: FWHM, L,S,C,X,Ku band: 0.5×FWHM
		K,Ka,Q Band: 0.25×FWHM
Niter	500	Max number of CLEAN iterations
UVRange	[0.0,0.0]	Range of baseline used in kilo wavelengths, zeros=all
minFlux	0.0	Minimum CLEAN flux density (Jy)
minSNR	4.0	Minimum Allowed SNR in self cal
maxPSCLoop	1	Max. number of phase self cal loops
minFluxPSC	0.05	Min flux density peak for phase self cal
solPInt		Phase self cal solution interval (min), ν dependent
		$\nu < 1$ GHz, L,C,X,Ku,K,Ka: 0.25, Q band:0.10
solPMode	"P"	Phase solution for phase self cal
solPType"	" "	Solution type for phase self cal
maxASCLoop	1	Max. number of Amp+phase self cal loops
minFluxASC	0.5	Min flux density peak for amp+phase self cal
solAInt		amp+phase self cal solution interval (min), ν dependent
		$\nu < 1$ GHz, L,C,X,Ku,K,Ka,Q: 3.0
solAMode	"A&P"	Amp and phase self cal
solAType	" "	Solution type for Amp and phase self cal
refAnt		Reference ant., for self cal, def. determined by script
avgPol	True	Average poln in self cal?
avgIF	False	Average IF in self cal?
nTaper	0	Number of additional imaging multi-resolution tapers
Tapers	[20.0,0.0]	List of tapers in pixels
do3D	False	Make ref. pixel tangent to celest. sphere for each facet
noNeg	False	Allow negative components in self cal model?
BLFact	1.01	Baseline dependent time averaging for > 1.0?
BLchAvg	True	Baseline dependent frequency averaging?
doMB	??	Set in parameter script depending on spanned bandwidth
MBnorder	1	Order of wideband imaging
MBmaxFBW	0.05	max. MB fractional bandwidth
CleanRad	None	CLEAN radius about center or None=autoWin
PBCor	True	Make primary beam correction when finished
antSize	24.5	Antenna diameter for primary beam correction (m)

31. Generate report

doReport	True	Generate source report?
targets	[?]	Target list set from ASDM, empty=>all
seq	1	AIPS sequence for images
outIclass	"IClean"	Image AIPS class
Stokes	"I"	Stokes imaged

32. Save images, calibrated data
Images and calibrated/averaged data and calibration tables are written to FITS files. File names begin with parms["project"]+parms["session"]+parms["band"] followed by <source_name>+<Stokes>+"Clean.fits" for images and "Cal.uvtab" for calibrated data and "CalTab.uvtab" for calibration tables from the original data.

doSaveImg	True	Save target images to FITS?
targets	[?]	Target list set from ASDM, empty=>all
doSaveUV	True	Save calibrated UV data for AIPS/FITAB format?
doSaveTab	True	Save calibration tables for AIPS/FITAB format?

33. Contour plots of images
Contour plots are generated for target images. Plot names are parms["project"]+"_"+parms["session"]+"_"+parms["band"] followed by the source name and ".cntr.ps" which are also converted to jpeg with the suffix "jpg".

doKntrPlots	True	Generate contour plots?
targets	[?]	Target list set from ASDM, empty=>all

34. UV diagnostic plots
Plots of amplitude vs. baseline length, real vs. imaginary and UV coverage are generated. Plot names are parms["project"]+"_"+parms["session"]+"_"+parms["band"] followed by the source name and ".amp.ps", ".ri.ps", or ".uv.ps" which are also converted to jpeg with the suffix "jpg".

doDiagPlots	True	Make UV diagnostic plots per source?
targets	[?]	Target list set from ASDM, empty=>all

35. Generate HTML Summary
Generate an HTML page with source statistics and links to the various plots.

doHTML	True	Generate HTML reports?

36. Cleanup
AIPS data and image files are zapped.

doCleanup	True	Clean out AIPS directories?

7 The Products

- Calibrated (u,v) dataset with calibration and flagging tables in AIPS FITAB format – Tables from initial data and averaged visibilities per input dataset. These files are
parms["project"]+parms["session"]+parms["band"]+"Cal.uvtab" and
parms["project"]+parms["session"]+parms["band"]+"CalTab.uvtab".
NB: This is not "uvfits" format.

- FITS Images – for each target object in files
parms["project"]+"_"+parms["session"]+"_"+parms["band"]+
source_name+".IClean.fits".
If wideband imaging is used, then the resultant images are cubes having planes:

  1. Total intensity at reference frequency.
  2. Spectral index at reference frequency
  3. any higher order planes
  4. One plane for each of the coarse frequency samples.

- Diagnostic plots – calibration and several per source. The project plots have prefix parms["project"]+"_"+parms["session"]+"_"+parms["band"] and are

  - `RawSpec.ps:` AIPS/POSSM plots of sample spectra with initial editing but no calibration applied.
  - `SYCal.ps:` AIPS/SNPLT plots of switched power calibration.
  - `DelaySpec.ps:` AIPS/POSSM plots of sample spectra with initial editing and delay calibration applied. One set per pass through the calibration.

- `BPSpec.ps:` AIPS/POSSM plots of sample spectra with initial editing and delay and bandpass calibration applied. One set per pass through the calibration.
- `Spec.ps:` AIPS/POSSM plots of sample spectra with final editing and calibration applied.
- `RLSpec2.ps:` AIPS/POSSM plots of sample RL and LR spectra with final editing and calibration applied.
- `DelayCal.ps:` AIPS/SNPLT plots of delay calibration.
- `APCal.ps:` AIPS/SNPLT plots of amplitude and phase calibration.

The source plots have prefix parms["project"]+"_"+parms["session"]+"_"+parms["band"] and are

- `source_name.cntr.jpg:` Source image contour plot as jpeg
- `source_name.cntr.ps:` Source image contour plot as postscript
- `source_name.amp.jpg:` Source amp. vs baseline plot as jpeg
- `source_name.amp.pdf:` Source amp. vs baseline plot as pdf
- `source_name.amp.ps:` Source amp. vs baseline plot as postscript
- `source_name.ri.jpg:` Source real vs imaginary plot as jpeg
- `source_name.ri.pdf:` Source real vs imaginary plot as pdf
- `source_name.ri.ps:` Source real vs imaginary plot as postscript
- `source_name.uv.jpg:` Source uv coverage plot as jpeg
- `source_name.uv.pdf:` Source uv coverage plot as pdf
- `source_name.uv.ps:` Source uv coverage plot as postscript

- Reports and logs created during the process
  The logfile is
  parms["project"]+"_"+parms["session"]+"_"+parms["band"]+".log", and
  the HTML report is
  parms["project"]+"_"+parms["session"]+"_"+parms["band"]+".report.html".

The file set comprising all files and the meta-data are stored in a single directory.