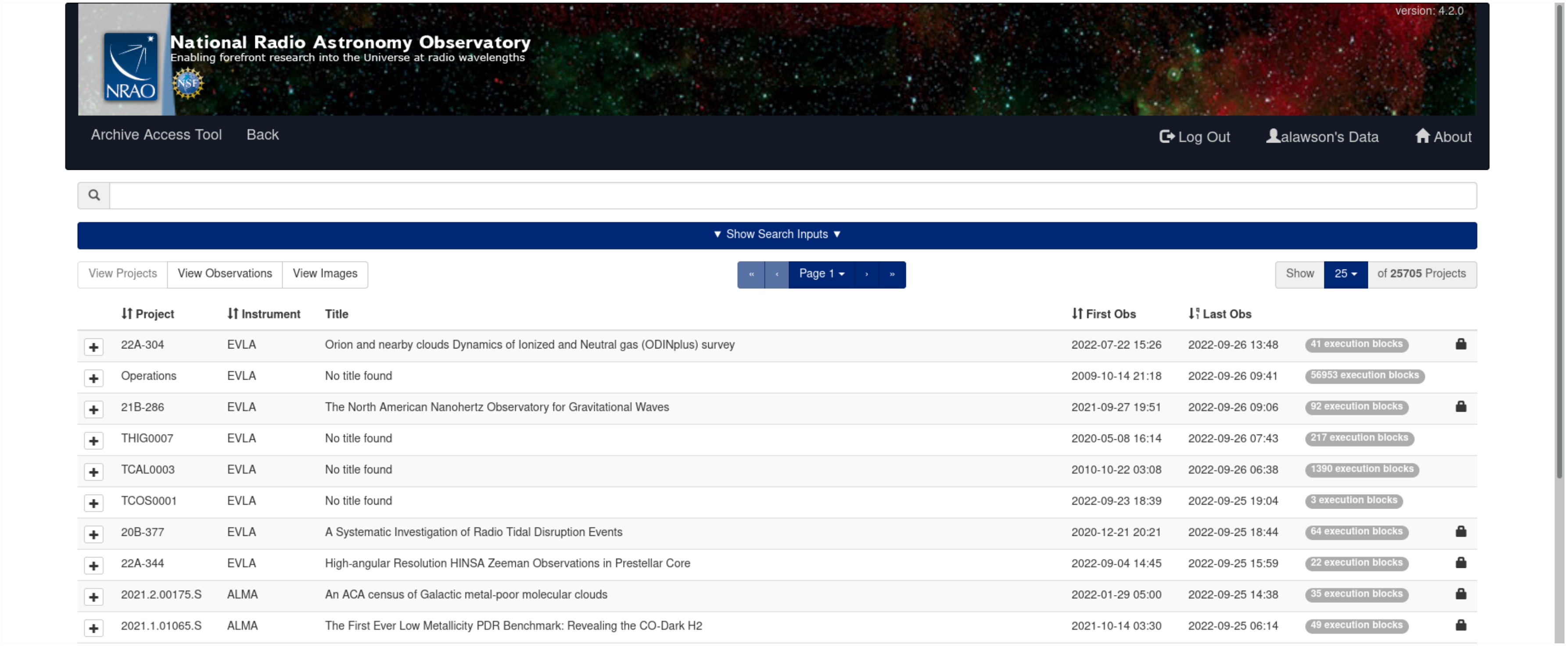
Practical Pipeline Applications

Aaron Lawson – NRAO 9th Data Reduction Workshop

Topics

- Getting a calibrated MS via the archive
- Running the pipeline yourself
- Using a flag template with the pipeline
- Avoiding specific reference antennas
- Setting UV limits
- Considerations for spectral line data
- Changing scan intents

Getting a Calibrated MS from the archive



	Project	Instrument	Title	First Obs	Last Obs		
+	22A-304	EVLA	Orion and nearby clouds Dynamics of Ionized and Neutral gas (ODINplus) survey	2022-07-22 15:26	2022-09-26 13:48	41 execution blocks	🔒
+	Operations	EVLA	No title found	2009-10-14 21:18	2022-09-26 09:41	56953 execution blocks	
+	21B-286	EVLA	The North American Nanohertz Observatory for Gravitational Waves	2021-09-27 19:51	2022-09-26 09:06	92 execution blocks	🔒
+	THIG0007	EVLA	No title found	2020-05-08 16:14	2022-09-26 07:43	217 execution blocks	
+	TCAL0003	EVLA	No title found	2010-10-22 03:08	2022-09-26 06:38	1390 execution blocks	
+	TCOS0001	EVLA	No title found	2022-09-23 18:39	2022-09-25 19:04	3 execution blocks	
+	20B-377	EVLA	A Systematic Investigation of Radio Tidal Disruption Events	2020-12-21 20:21	2022-09-25 18:44	64 execution blocks	🔒
+	22A-344	EVLA	High-angular Resolution HINSA Zeeman Observations in Prestellar Core	2022-09-04 14:45	2022-09-25 15:59	22 execution blocks	🔒
+	2021.2.00175.S	ALMA	An ACA census of Galactic metal-poor molecular clouds	2022-01-29 05:00	2022-09-25 14:38	35 execution blocks	🔒
+	2021.1.01065.S	ALMA	The First Ever Low Metallicity PDR Benchmark: Revealing the CO-Dark H2	2021-10-14 03:30	2022-09-25 06:14	49 execution blocks	🔒

- Fount of all knowledge: https://data.nrao.edu
  - Raw data
  - Calibrated data
  - Pipeline products
  - Images (Coming soon)

Restoring a Calibration manually

- Why?
  - Archive doesn’t have all versions of CASA
  - CASA 4.3 ~ 4.5
    - Pipeline products from this time used file extension “.tar.gz”
    - Modern restore process expects “.tgz”
    - Fix is to manually rename the “.tar.gz” files to “.tgz”
  - CASA 4.5.0 ~ 4.5.3 (fixed in 4.7)
    - hif_restoredata errors due to data shape mismatch (restorescript used ocorrmode=’ca’ while pipeline used ocorrmode=’co’)
    - Fix is to manually create a MS with ocorrmode=’co’ and use that as the pipeline starting point
    - https://science.nrao.edu/facilities/vla/data-processing/pipeline/KI_CIPL_453

Restoring a Calibration manually

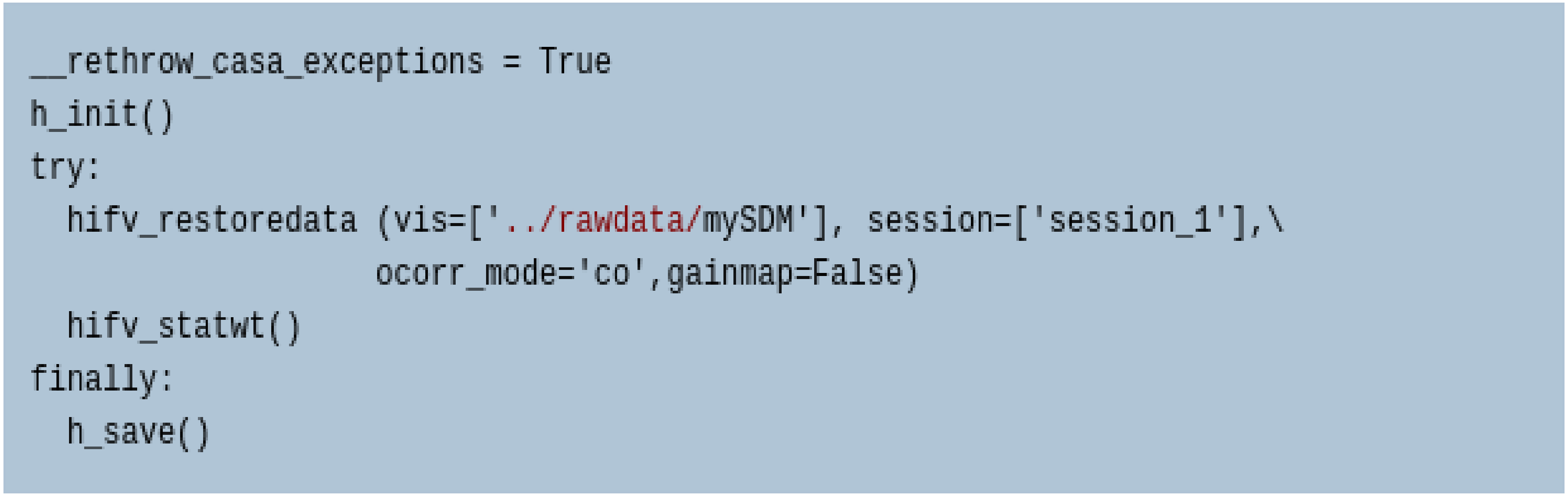
- Ingredients
  - Your CASA+pipeline version of choice
  - A SMD-BDF
    - Can substitute a fresh MS in a pinch
  - Several pipeline products
    - Unknown.session_1.caltables.tgz
    - <mySDM>.ms.flagversions.tgz
    - <mySDM>.ms.calapply.txt
    - Unknown.pipeline_manifest.xml
    - casa_piperestorescript.py

Restoring a Calibration manually

- Restoration
  - rawdata
    - <mySDM>
  - products
    - unknown.session_1.caltables.tgz
    - <mySDM>.ms.flagversions.tgz
    - <mySDM>.ms.calapply.txt
    - unknown.pipeline_manifest.xml
  - working
    - casa_piperestorescript.py

Restoring a Calibration manually

- Edit the casa_piperestorescript.py to point to your <mySDM>



```
__rethrow_casa_exceptions = True
h_init()
try:
  hifv_restoredata (vis=['../rawdata/mySDM'], session=['session_1'],\
                   ocorr_mode='co',gainmap=False)
  hifv_statwt()
finally:
  h_save()
```

Restoring a Calibration manually

- Start-up your CASA+pipeline version
  - <your casa+pipeline> --pipeline
  - If on NRAO servers casa-pipe starts current production version
- Start the restore script
  - execfile(‘casa_piperestorescript.py’)
- Let cook until Corrected Data column is golden brown

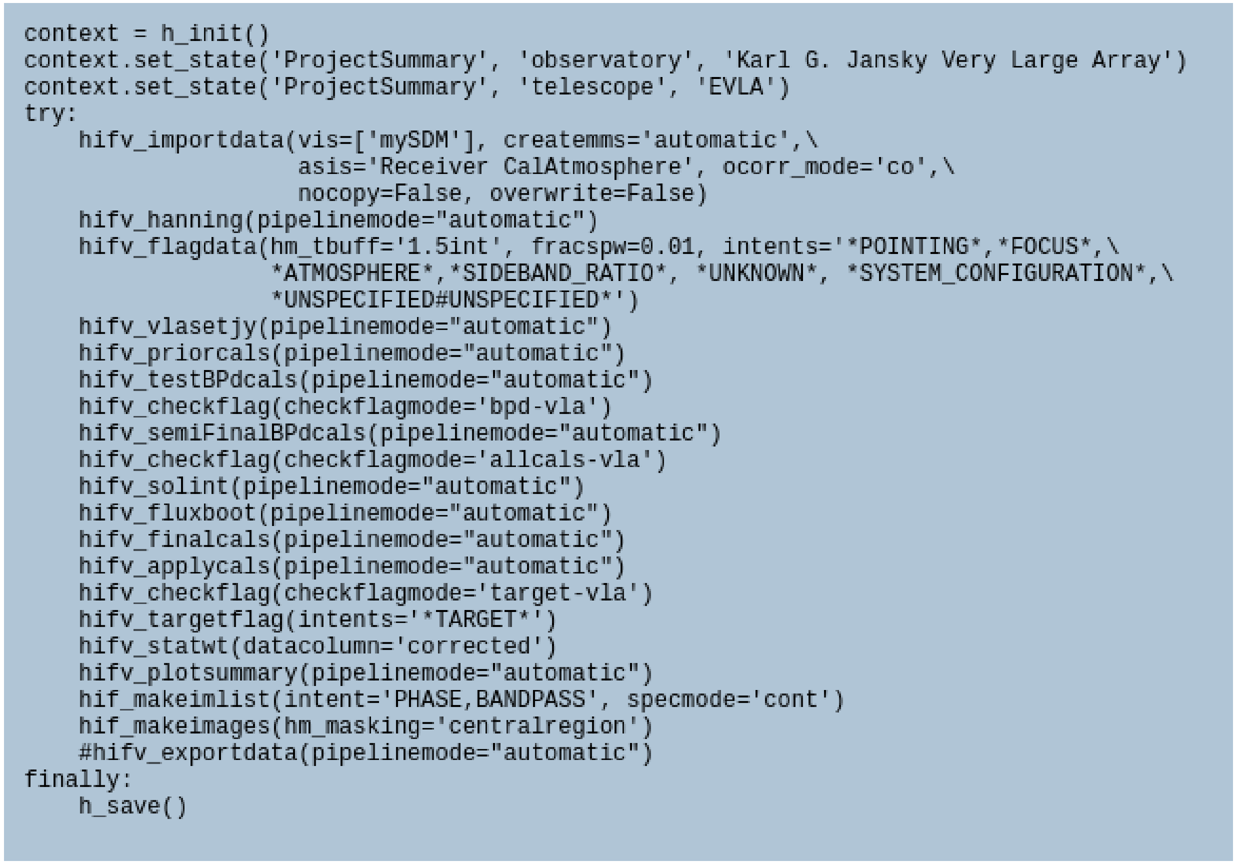
Starting CASA with the pipeline tasks

- Only a few specific versions of CASA actually have the pipeline built into them
  - Usually those versions have “pipeline” in the installation name, e.g. “6.1.2-7-pipeline-2020.1.0.40”
- Need to specifically tell CASA to load the pipeline tasks in order to use them
  - “<your casa> --pipeline”
    - If you try “<your casa> --pipeline” on a version without the pipeline built into it you’ll see a warning message telling you that CASA “could not import pipeline”
  - “casa-pipe”
    - Works if you’re using NRAO’s computing cluster
    - Starts whatever version of the pipeline we are currently using for production

CASA+pipeline downloads available now at*: https://casa.nrao.edu/casa_obtaining.shtml

*Prices may vary and are usually paid in your machine’s RAM

The casa_pipescript.py file

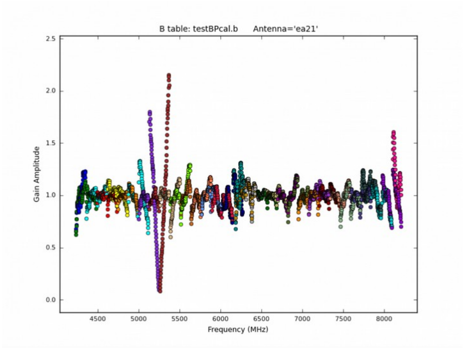


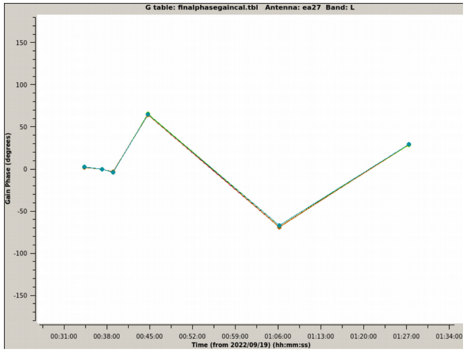
```
context = h_init()
context.set_state('ProjectSummary', 'observatory', 'Karl G. Jansky Very Large Array')
context.set_state('ProjectSummary', 'telescope', 'EVLA')
try:
    hifv_importdata(vis=['mySDM'], createmms='automatic',\
                    asis='Receiver CalAtmosphere', ocorr_mode='co',\
                    nocopy=False, overwrite=False)
    hifv_hanning(pipelinemode="automatic")
    hifv_flagdata(hm_tbuff='1.5int', fracspw=0.01, intents='*POINTING*,*FOCUS*,\
                  *ATMOSPHERE*,*SIDEBAND_RATIO*, *UNKNOWN*, *SYSTEM_CONFIGURATION*,\
                  *UNSPECIFIED#UNSPECIFIED*')
    hifv_vlasetjy(pipelinemode="automatic")
    hifv_priorcals(pipelinemode="automatic")
    hifv_testBPdcals(pipelinemode="automatic")
    hifv_checkflag(checkflagmode='bpd-vla')
    hifv_semiFinalBPdcals(pipelinemode="automatic")
    hifv_checkflag(checkflagmode='allcals-vla')
    hifv_solint(pipelinemode="automatic")
    hifv_fluxboot(pipelinemode="automatic")
    hifv_finalcals(pipelinemode="automatic")
    hifv_applycals(pipelinemode="automatic")
    hifv_checkflag(checkflagmode='target-vla')
    hifv_targetflag(intents='*TARGET*')
    hifv_statwt(datacolumn='corrected')
    hifv_plotsummary(pipelinemode="automatic")
    hif_makeimlist(intent='PHASE,BANDPASS', specmode='cont')
    hif_makeimages(hm_masking='centralregion')
    #hifv_exportdata(pipelinemode="automatic")
finally:
    h_save()
```

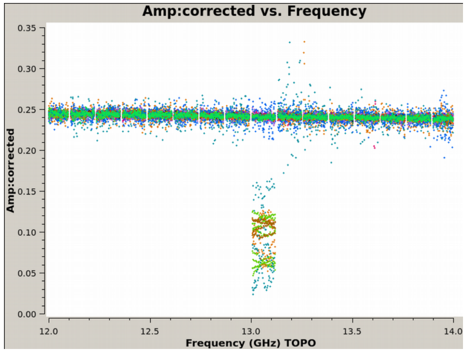
- Can be run with “execfile(‘casa_pipescript.py’)” inside of CASA
- Can also tell CASA to execute it (or any other script) on startup with: “<your casa> --pipeline -c /path/to/casa_pipescript.py” in your terminal

Template script at: https://science.nrao.edu/facilities/vla/data-processing/pipeline#section-16

The flag template



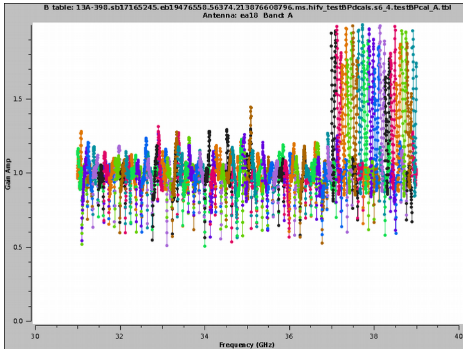


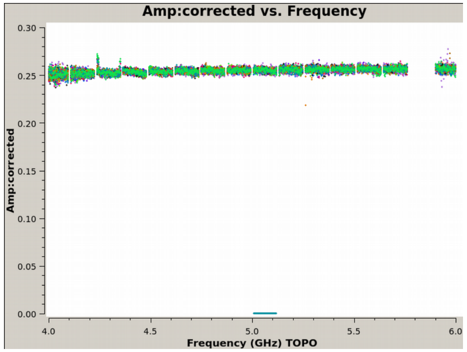


DTS – two different kinds

Phase jumps

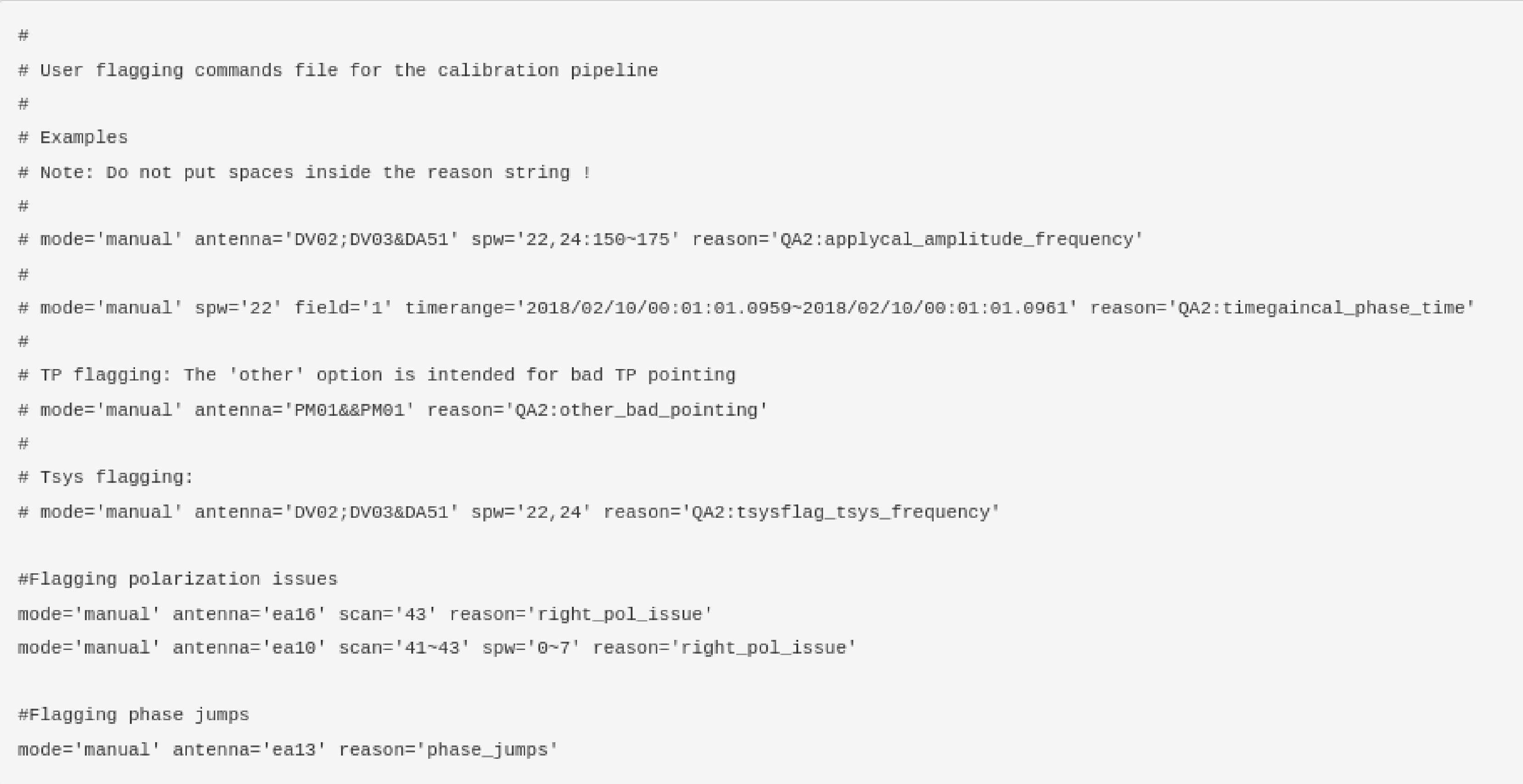
Baseline board issue





NRAO's 18th SIW Error Recognition talk:
http://www.aoc.nrao.edu/events/synthesis/2022/slides/Kepley_Error_Recognition_2022_final.pdf

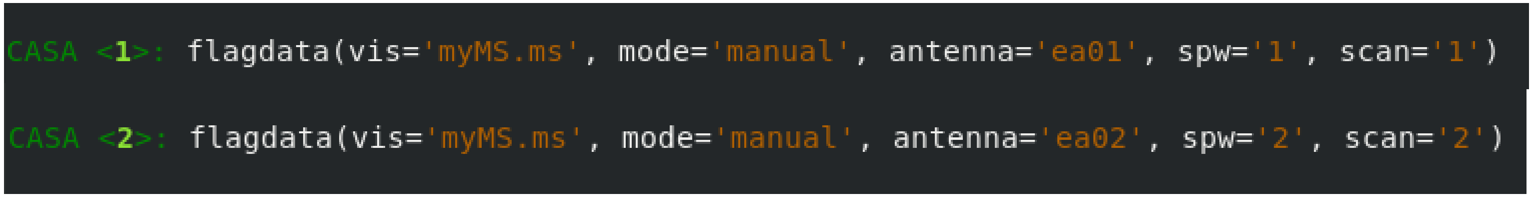
The flag template



```
#
# User flagging commands file for the calibration pipeline
#
# Examples
# Note: Do not put spaces inside the reason string !
#
# mode='manual' antenna='DV02;DV03&DA51' spw='22,24:150~175' reason='QA2:applycal_amplitude_frequency'
#
# mode='manual' spw='22' field='1' timerange='2018/02/10/00:01:01.0959~2018/02/10/00:01:01.0961' reason='QA2:timegaincal_phase_time'
#
# TP flagging: The 'other' option is intended for bad TP pointing
# mode='manual' antenna='PM01&&PM01' reason='QA2:other_bad_pointing'
#
# Tsys flagging:
# mode='manual' antenna='DV02;DV03&DA51' spw='22,24' reason='QA2:tsysflag_tsys_frequency'

#Flagging polarization issues
mode='manual' antenna='ea16' scan='43' reason='right_pol_issue'
mode='manual' antenna='ea10' scan='41~43' spw='0~7' reason='right_pol_issue'

#Flagging phase jumps
mode='manual' antenna='ea13' reason='phase_jumps'
```

- Lines that start with “#” are ignored
- Pipeline will make one for you when start, but it will be “empty”
- Syntax for actual commands is unique, but similar to flagdata calls

The flag template



```
CASA <1>: flagdata(vis='myMS.ms', mode='manual', antenna='ea01', spw='1', scan='1')

CASA <2>: flagdata(vis='myMS.ms', mode='manual', antenna='ea02', spw='2', scan='2')
```

Becomes:

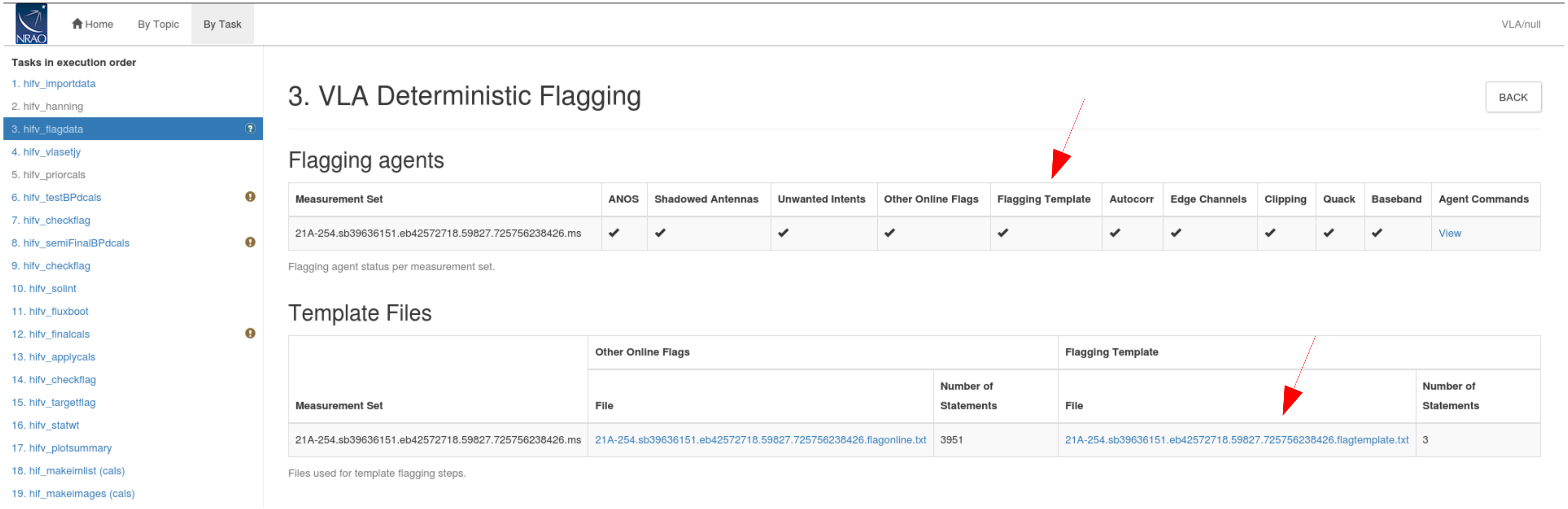
mode=’manual’ antenna=’ea01’ spw=’1’ scan=’1’ reason=’1_is_the_loneliest_number’

mode=’manual’ antenna=’ea02’ spw=’2’ scan=’2’ reason=’2_can_be_as_bad_as_1’

- “flagdata( )” is dropped
- “,” are dropped
- No need to reference “vis”
- Each command must be on its own line
- The “reason” parameter is required and cannot include whitespace characters

The flag template

- Default name is <mySDM>.flagtemplate.txt
  - If named the default way and in the directory the pipeline is running, then the file is picked up automatically
  - Can be edited to anything, but you will need to add the “flagtemplate=’newName.txt’” parameter to the “hifv_flagdata” task in the casa_pipescript.py file to pick it up
- Can see that it was used in the “hifv_flagdata” task



Tasks in execution order

1. hifv_importdata
2. hifv_hanning
3. hifv_flagdata
4. hifv_vlasetjy
5. hifv_priorcals
6. hifv_testBPdcals
7. hifv_checkflag
8. hifv_semiFinalBPdcals
9. hifv_checkflag
10. hifv_solint
11. hifv_fluxboot
12. hifv_finalcals
13. hifv_applycals
14. hifv_checkflag
15. hifv_targetflag
16. hifv_statwt
17. hifv_plotsummary
18. hif_makeimlist (cals)
19. hif_makeimages (cals)

3. VLA Deterministic Flagging

Flagging agents

Measurement Set	ANOS	Shadowed Antennas	Unwanted Intents	Other Online Flags	Flagging Template	Autocorr	Edge Channels	Clipping	Quack	Baseband	Agent Commands
21A-254.sb39636151.eb42572718.59827.725756238426.ms	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	View

Flagging agent status per measurement set.

Template Files

	Other Online Flags		Flagging Template	
Measurement Set	File	Number of Statements	File	Number of Statements
21A-254.sb39636151.eb42572718.59827.725756238426.ms	21A-254.sb39636151.eb42572718.59827.725756238426.flagonline.txt	3951	21A-254.sb39636151.eb42572718.59827.725756238426.flagtemplate.txt	3

Files used for template flagging steps.

Avoiding a specific reference antenna

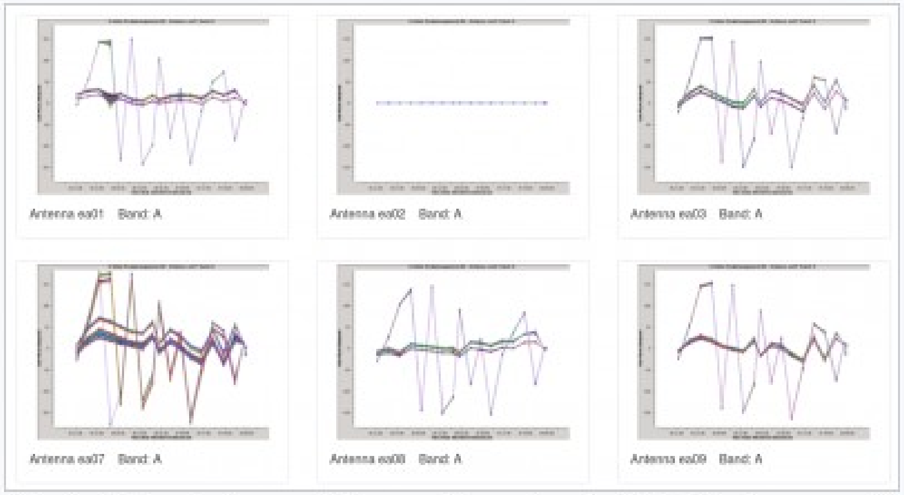


Fig. 7a. Plots of phase solutions vs time showing that all antennas have inherited ea02's phase issue when it is used as the reference antenna.

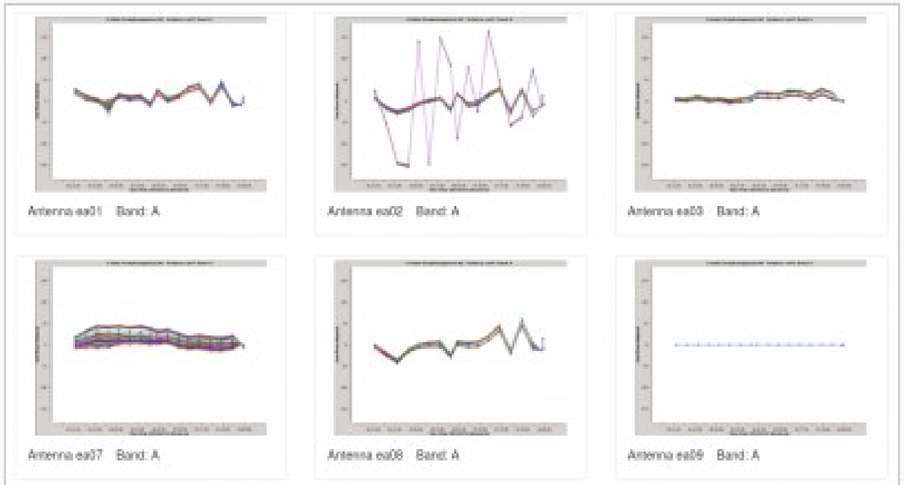
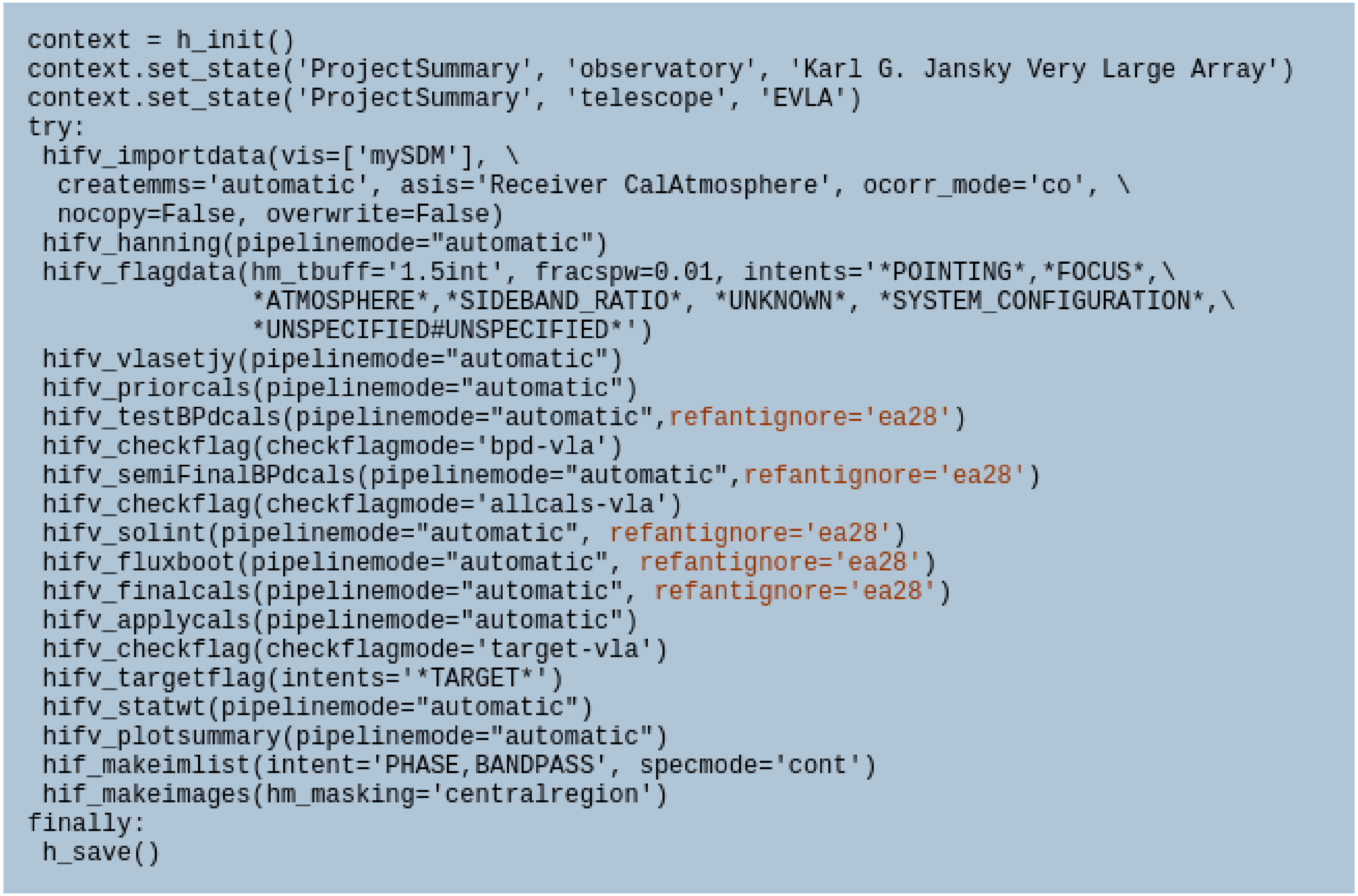


Fig 7b. Plots of phase solutions vs time showing ea02 has a phase issue. Here ea09 is used as the reference antenna.

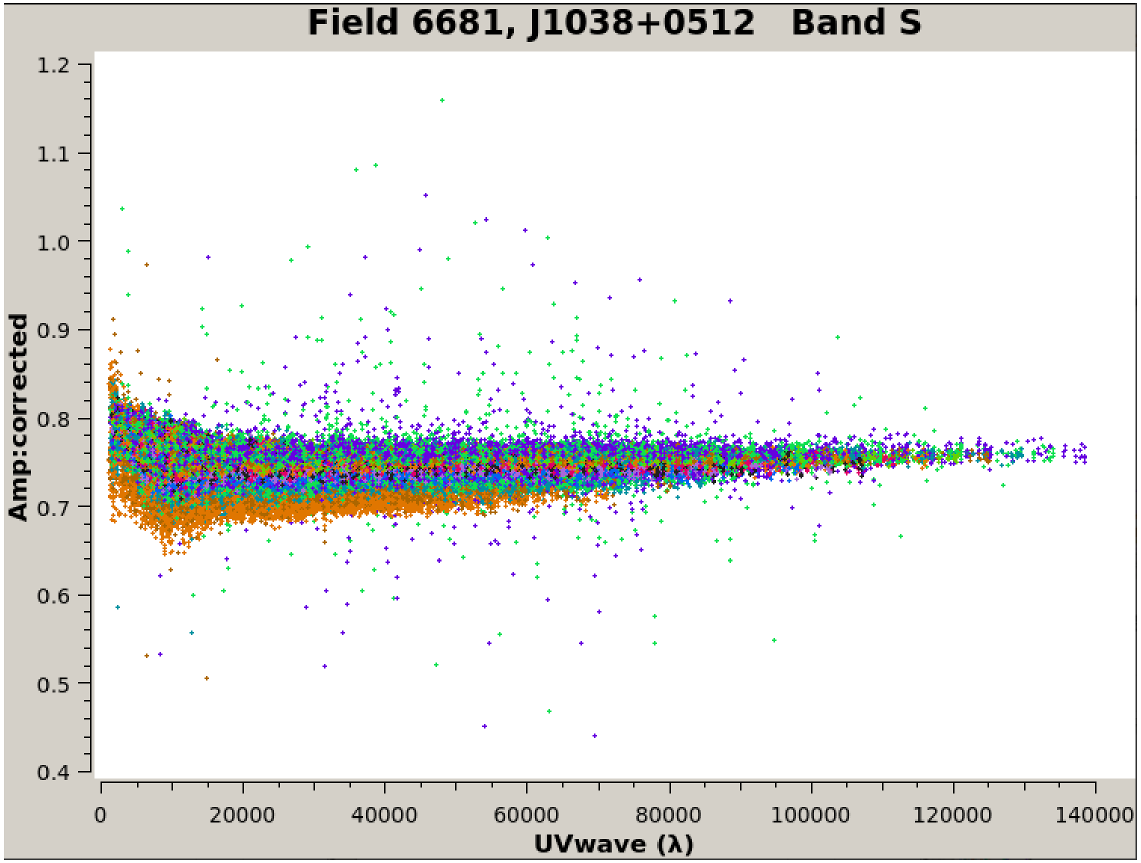
- Pipeline’s choice of reference antenna is mostly based off flagging percent and distance to the center of the array, so it will sometimes choose poorly

Avoiding a specific reference antenna



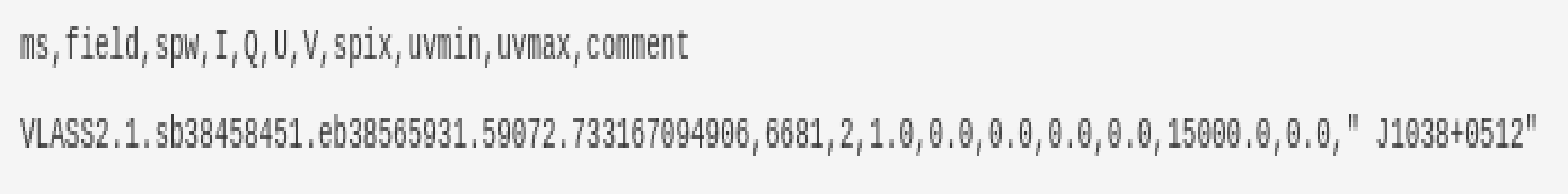
```
context = h_init()
context.set_state('ProjectSummary', 'observatory', 'Karl G. Jansky Very Large Array')
context.set_state('ProjectSummary', 'telescope', 'EVLA')
try:
 hifv_importdata(vis=['mySDM'], \
  createmms='automatic', asis='Receiver CalAtmosphere', ocorr_mode='co', \
  nocopy=False, overwrite=False)
 hifv_hanning(pipelinemode="automatic")
 hifv_flagdata(hm_tbuff='1.5int', fracspw=0.01, intents='*POINTING*,*FOCUS*,\
               *ATMOSPHERE*,*SIDEBAND_RATIO*, *UNKNOWN*, *SYSTEM_CONFIGURATION*,\
               *UNSPECIFIED#UNSPECIFIED*')
 hifv_vlasetjy(pipelinemode="automatic")
 hifv_priorcals(pipelinemode="automatic")
 hifv_testBPdcals(pipelinemode="automatic",refantignore='ea28')
 hifv_checkflag(checkflagmode='bpd-vla')
 hifv_semiFinalBPdcals(pipelinemode="automatic",refantignore='ea28')
 hifv_checkflag(checkflagmode='allcals-vla')
 hifv_solint(pipelinemode="automatic", refantignore='ea28')
 hifv_fluxboot(pipelinemode="automatic", refantignore='ea28')
 hifv_finalcals(pipelinemode="automatic", refantignore='ea28')
 hifv_applycals(pipelinemode="automatic")
 hifv_checkflag(checkflagmode='target-vla')
 hifv_targetflag(intents='*TARGET*')
 hifv_statwt(pipelinemode="automatic")
 hifv_plotsummary(pipelinemode="automatic")
 hif_makeimlist(intent='PHASE,BANDPASS', specmode='cont')
 hif_makeimages(hm_masking='centralregion')
finally:
 h_save()
```

Setting UV limits



- Calibrators can’t always be point sources, but the pipeline will treat them as point sources.
- We can use UV-minimums and UV-maximums to help deal with non point source calibrators.

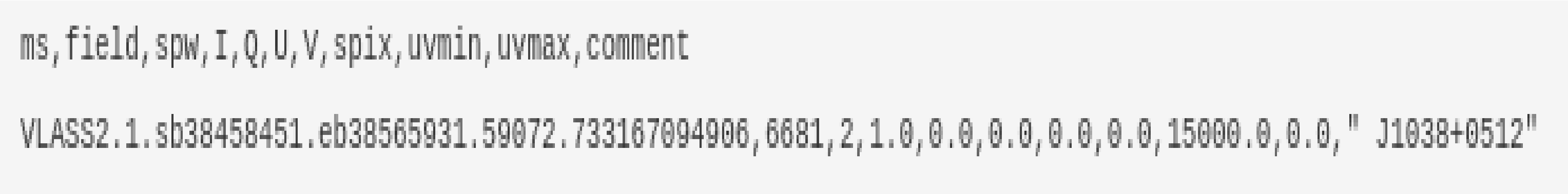
Setting UV limits



```
ms,field,spw,I,Q,U,V,spix,uvmin,uvmax,comment

VLASS2.1.sb38458451.eb38565931.59072.733167094906,6681,2,1.0,0.0,0.0,0.0,0.0,15000.0,0.0," J1038+0512"
```

- A flux.csv file can be used to set UV-minimums and UV-maximums
- If named “flux.csv” and placed in the pipeline’s working directory it will be picked up automatically
- Only need to list one of the spectral windows in the observation
  - All other spectral windows will inherit the uvmin/uvmax
  - Cannot use a specific uvmin/uvmax per spectral window

Setting UV limits



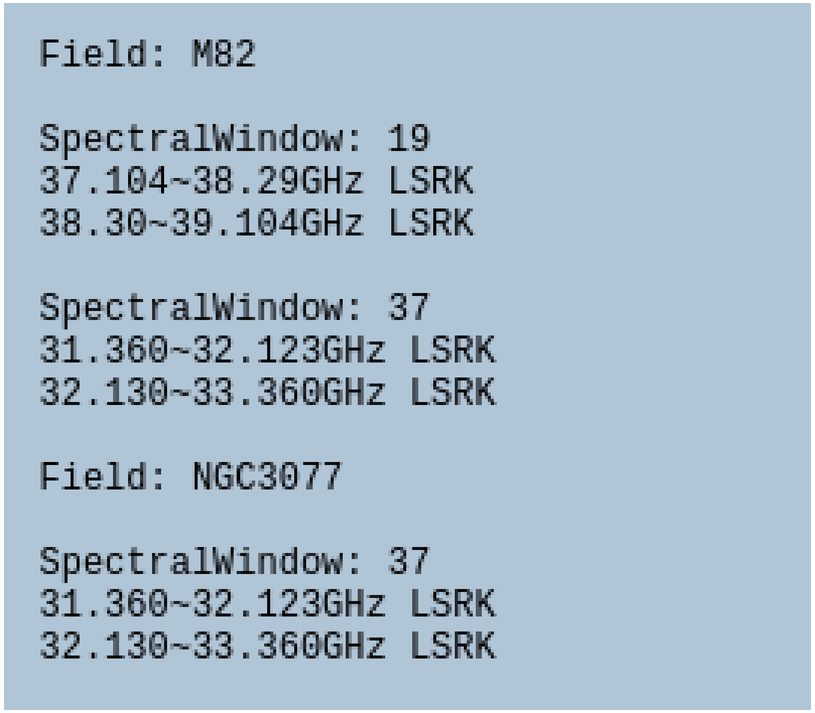
```
ms,field,spw,I,Q,U,V,spix,uvmin,uvmax,comment

VLASS2.1.sb38458451.eb38565931.59072.733167094906,6681,2,1.0,0.0,0.0,0.0,0.0,15000.0,0.0," J1038+0512"
```

- The Stokes parameters (I, Q, U, V) are ignored for VLA so we can just set a nominal I, Q, U, V of 1.0, 0.0, 0.0, 0.0
- The uvmin/uvmax parameters are given in units of lambda
- Having uvmax = 0.0 causes uvmax to be ignored (no upper bound)
- If to much data is cut off via the uvmin/uvmax you'll notice increased flagging due to the gaincal solver not having enough data to work with

Spectral Line Considerations

- The pipeline isn't optimized for spectral line data
- Without extra precautions it is likely to flag any strong lines as 'RFI'
- Even if the line survives it is likely to be down weighted by the statwt task during hifv_statwt
  - statwt works based of the RMS noise of the visibilities; strong lines will naturally have higher RMS
- The pipeline by default applies Hanning Smoothing to the data (may or may not be desirable)
- The pipeline will automatically flag the edge channels for each spectral window (hopefully your line isn't there…)

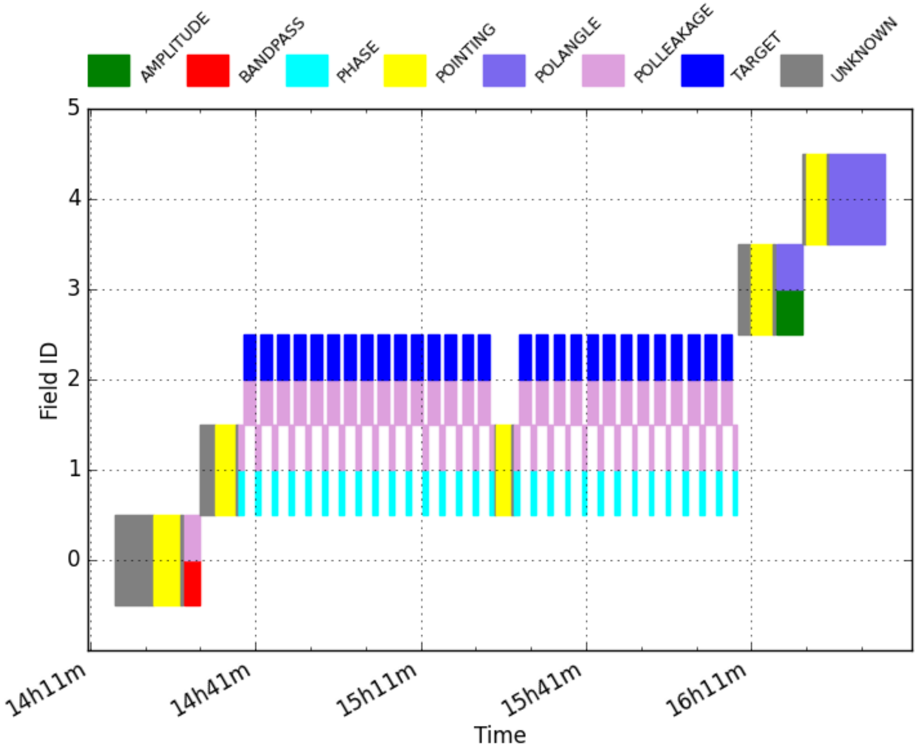
Spectral Line Considerations – cont.dat



```
Field: M82

SpectralWindow: 19
37.104~38.29GHz LSRK
38.30~39.104GHz LSRK

SpectralWindow: 37
31.360~32.123GHz LSRK
32.130~33.360GHz LSRK

Field: NGC3077

SpectralWindow: 37
31.360~32.123GHz LSRK
32.130~33.360GHz LSRK
```

- Data defined here will be considered as continuum and will undergo RFI flagging and statwt
  - Data not defined here will not undergo RFI flagging or statwt
- Needs to be named “cont.dat” and placed in the pipeline’s working directory to be picked up
- Unfortunately no opposite behavior (specline.dat)

Changing scan intents

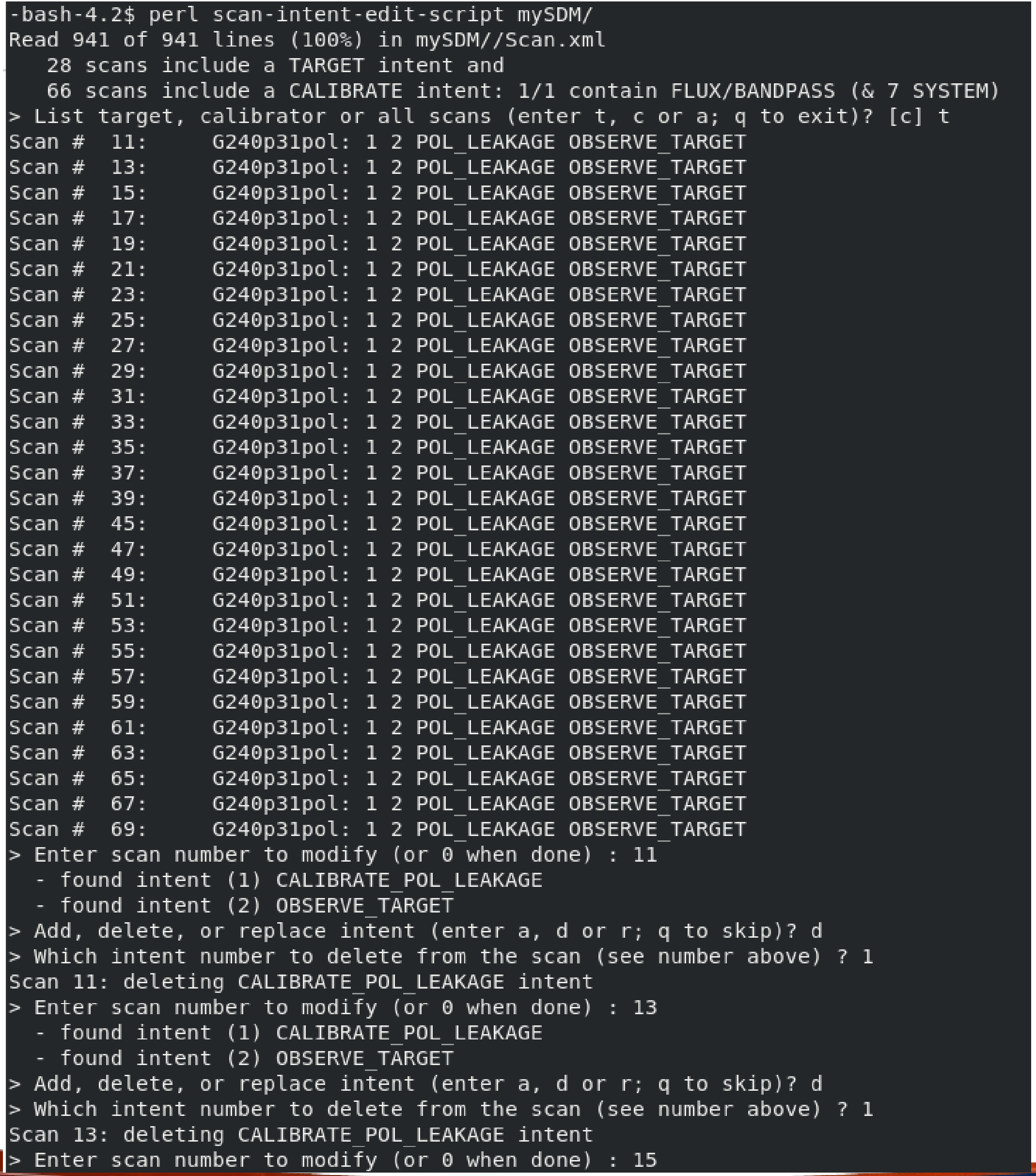


- Scan intents are set when you make your observation in the OPT (observation preparation tool)
- Pipeline uses these scan intents to know how it should treat each field
- Setting the intents in unexpected ways can have undesirable effects
  - But luckily we can change these after the observation

Changing scan intents

- An easy(-ish) way of editing the intents is by using a Perl script provided by NRAO
  - Description: https://science.nrao.edu/facilities/vla/data-processing/pipeline#section-32
  - Perl script: https://science.nrao.edu/facilities/vla/data-processing/pipeline/scan-intent-edit-script
- Script wants an SDM as input and then guides you through on how to add/delete/replace scan intents

Changing scan intents



```
-bash-4.2$ perl scan-intent-edit-script mySDM/
Read 941 of 941 lines (100%) in mySDM//Scan.xml
   28 scans include a TARGET intent and
   66 scans include a CALIBRATE intent: 1/1 contain FLUX/BANDPASS (& 7 SYSTEM)
> List target, calibrator or all scans (enter t, c or a; q to exit)? [c] t
Scan #  11:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  13:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  15:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  17:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  19:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  21:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  23:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  25:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  27:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  29:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  31:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  33:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  35:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  37:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  39:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  45:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  47:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  49:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  51:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  53:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  55:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  57:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  59:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  61:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  63:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  65:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  67:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
Scan #  69:     G240p31pol: 1 2 POL_LEAKAGE OBSERVE_TARGET
> Enter scan number to modify (or 0 when done) : 11
  - found intent (1) CALIBRATE_POL_LEAKAGE
  - found intent (2) OBSERVE_TARGET
> Add, delete, or replace intent (enter a, d or r; q to skip)? d
> Which intent number to delete from the scan (see number above) ? 1
Scan 11: deleting CALIBRATE_POL_LEAKAGE intent
> Enter scan number to modify (or 0 when done) : 13
  - found intent (1) CALIBRATE_POL_LEAKAGE
  - found intent (2) OBSERVE_TARGET
> Add, delete, or replace intent (enter a, d or r; q to skip)? d
> Which intent number to delete from the scan (see number above) ? 1
Scan 13: deleting CALIBRATE_POL_LEAKAGE intent
> Enter scan number to modify (or 0 when done) : 15
```

Start with “perl <scriptName> <SDM>”

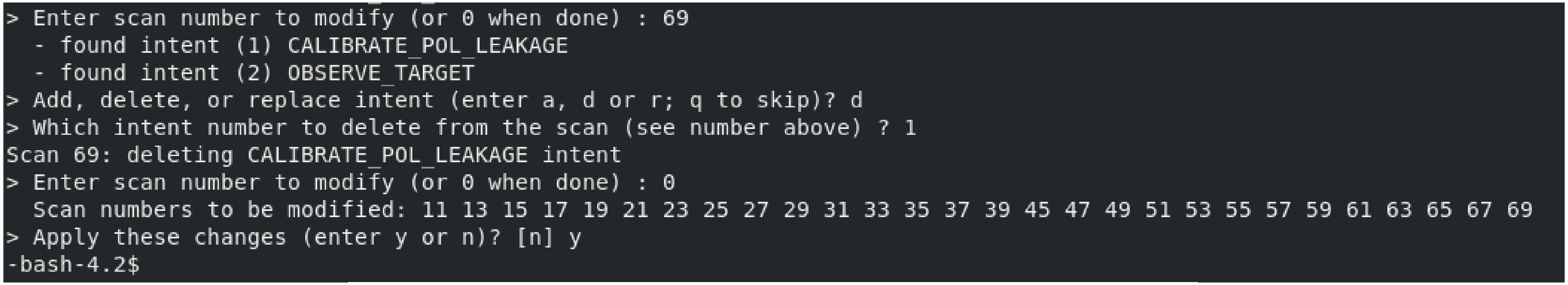
Asks you if you want to see target/calibrator/all scans

Lists your selection

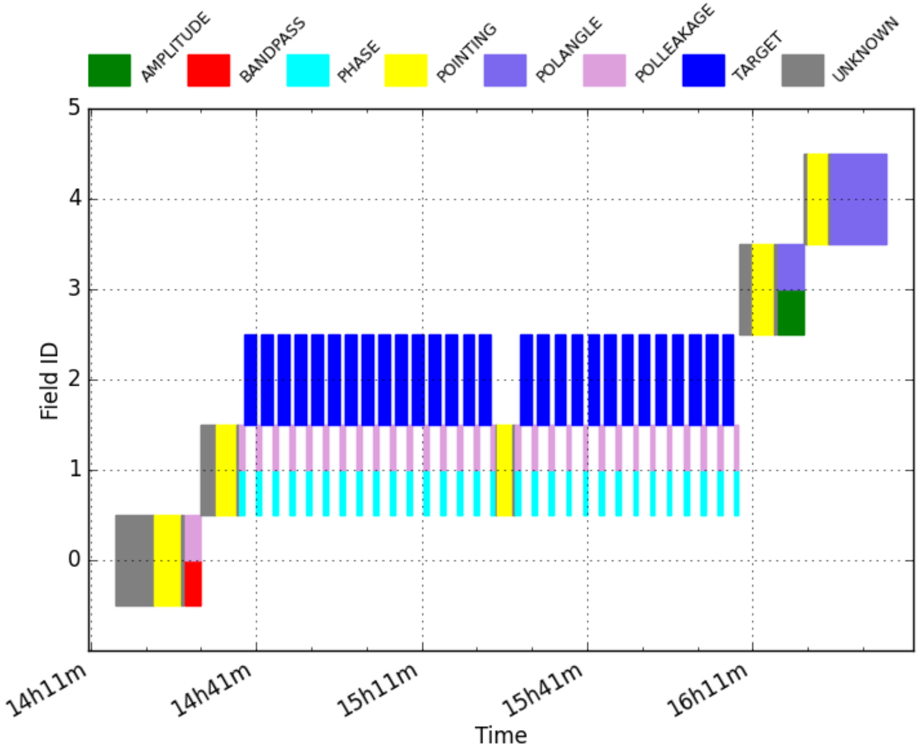
Which scan in the list do you want to modify?

How do you want to modify it?

Changing scan intents



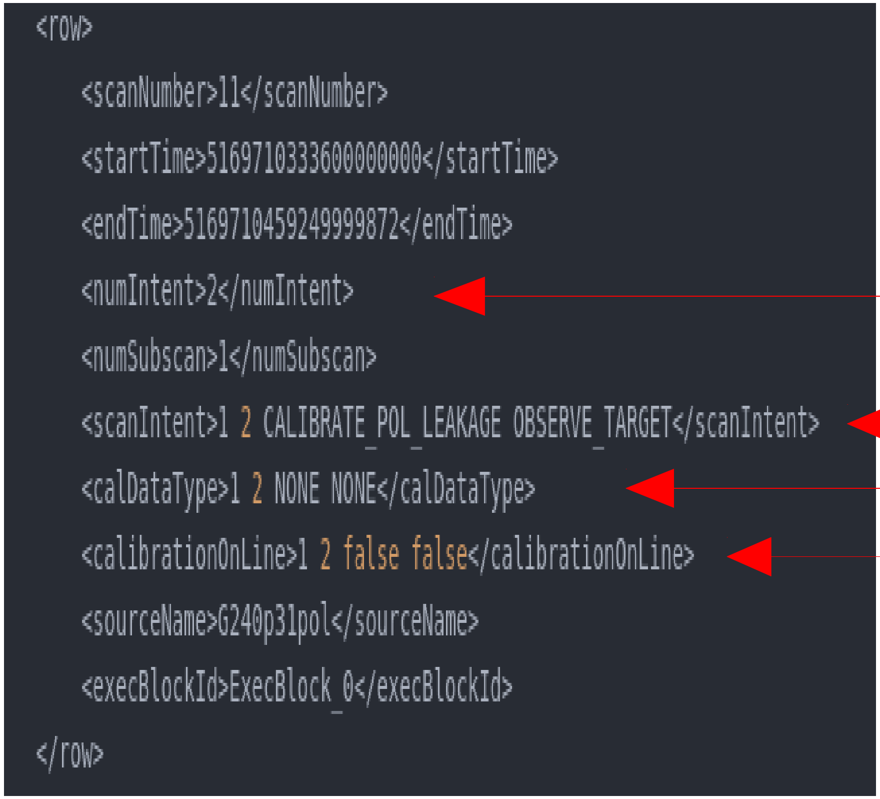
```
> Enter scan number to modify (or 0 when done) : 69
  - found intent (1) CALIBRATE_POL_LEAKAGE
  - found intent (2) OBSERVE_TARGET
> Add, delete, or replace intent (enter a, d or r; q to skip)? d
> Which intent number to delete from the scan (see number above) ? 1
Scan 69: deleting CALIBRATE_POL_LEAKAGE intent
> Enter scan number to modify (or 0 when done) : 0
  Scan numbers to be modified: 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 45 47 49 51 53 55 57 59 61 63 65 67 69
> Apply these changes (enter y or n)? [n] y
-bash-4.2$
```



Changing scan intents

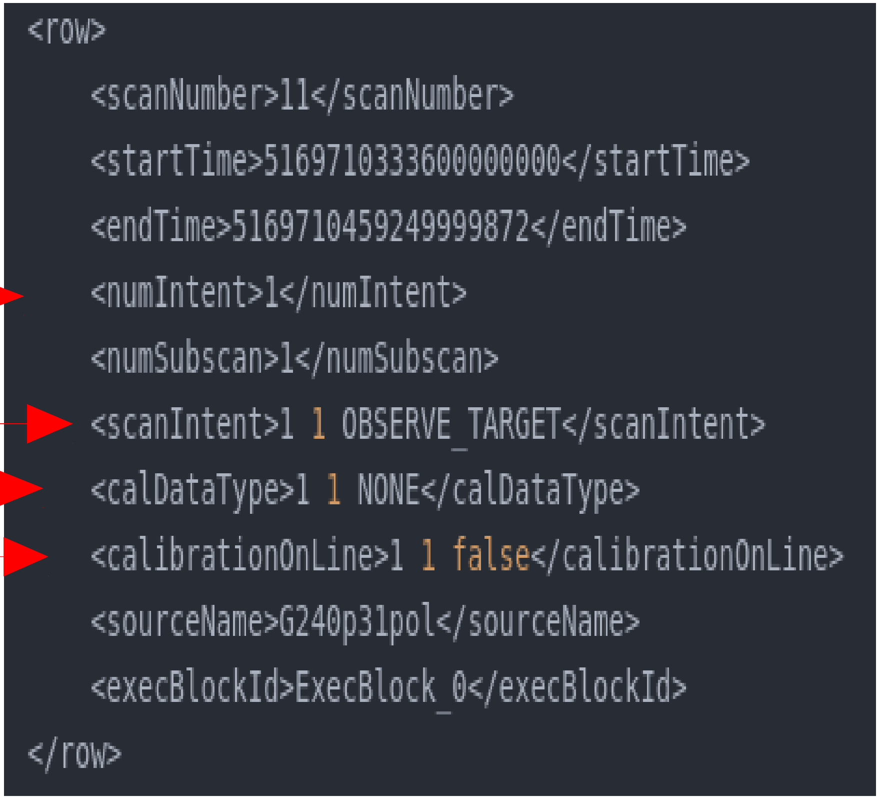
- Can edit the Scan.xml in the SDM directly
  - Probably wanna make a backup before you try this

Before



```
<row>
    <scanNumber>11</scanNumber>
    <startTime>5169710333600000000</startTime>
    <endTime>5169710459249999872</endTime>
    <numIntent>2</numIntent>
    <numSubscan>1</numSubscan>
    <scanIntent>1 2 CALIBRATE_POL_LEAKAGE OBSERVE_TARGET</scanIntent>
    <calDataType>1 2 NONE NONE</calDataType>
    <calibrationOnLine>1 2 false false</calibrationOnLine>
    <sourceName>G240p31pol</sourceName>
    <execBlockId>ExecBlock_0</execBlockId>
</row>
```

After



```
<row>
    <scanNumber>11</scanNumber>
    <startTime>5169710333600000000</startTime>
    <endTime>5169710459249999872</endTime>
    <numIntent>1</numIntent>
    <numSubscan>1</numSubscan>
    <scanIntent>1 1 OBSERVE_TARGET</scanIntent>
    <calDataType>1 1 NONE</calDataType>
    <calibrationOnLine>1 1 false</calibrationOnLine>
    <sourceName>G240p31pol</sourceName>
    <execBlockId>ExecBlock_0</execBlockId>
</row>
```

Links

- Archive:
  - https://data.nrao.edu
- Pipeline info:
  - https://science.nrao.edu/facilities/vla/data-processing/pipeline
- Scripted Pipeline:
  - https://science.nrao.edu/facilities/vla/data-processing/pipeline/scripted-pipeline
- Frequent VLA Problems:
  - https://casaguides.nrao.edu/index.php?title=Pipeline:_Frequent_VLA_problems
- Pipeline guide:
  - https://casaguides.nrao.edu/index.php?title=VLA-S-CASA_Pipeline-CASA6.2.1
- Helpdesk:
  - https://help.nrao.edu/

www.nrao.edu
science.nrao.edu
public.nrao.edu

The National Radio Astronomy Observatory is a facility of the National Science Foundation operated under cooperative agreement by Associated Universities, Inc.