VLA Data Reduction II: *Calibration*

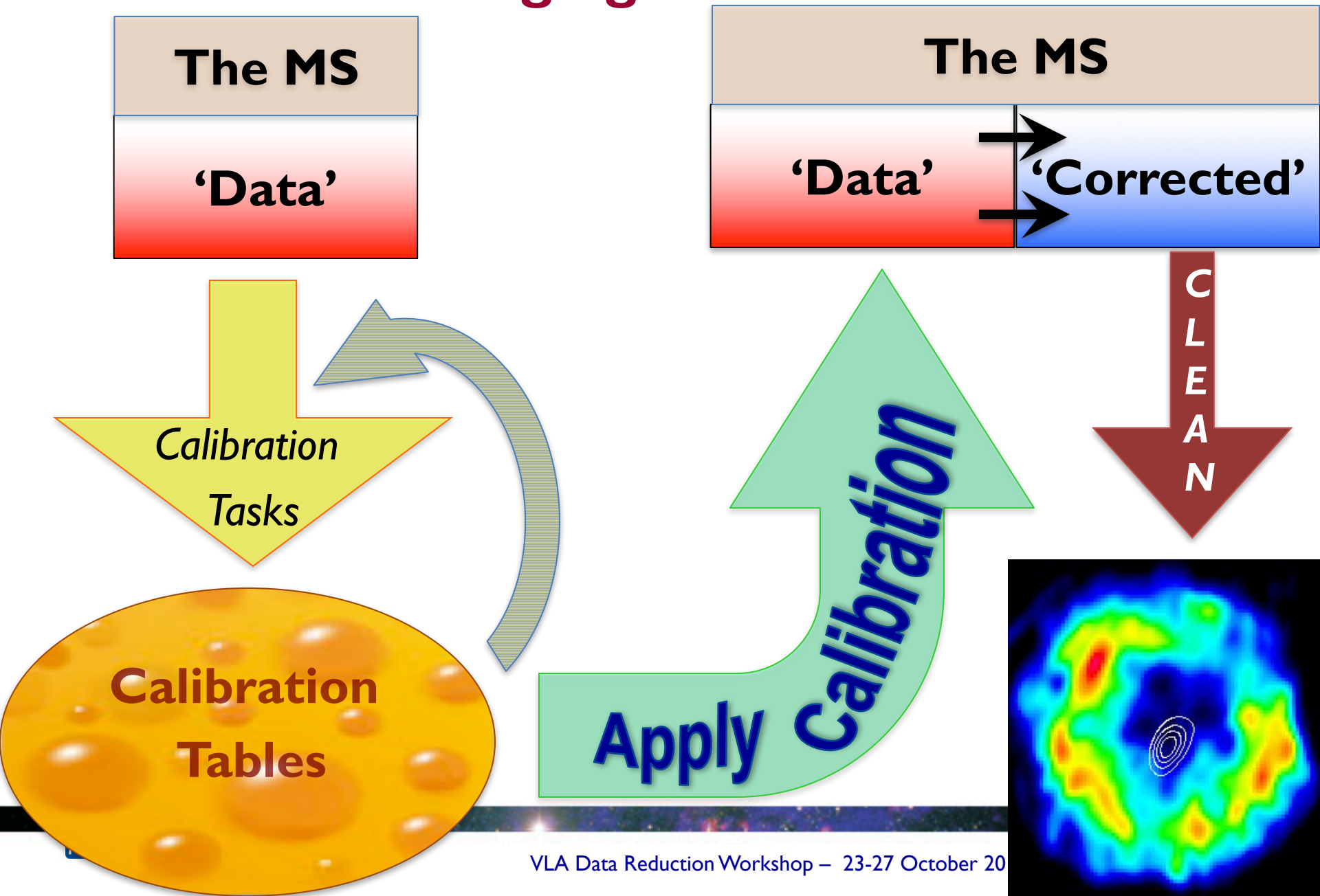
Amy Kimball, *NRAO*

The MS structure

‘Data’ column Raw Data	‘Corrected’ Column Calibrated Data	‘Model’ Column (optional) FT of source model

- When you load your data from the archive, your MS will only have the ‘Data’ column.
- The other two columns can be created by various means.
- The creation of the other two columns → MS *triples* in size.
- The ‘Model’ Column is optional.
  - If not created → MS *doubles* in size.
  - Models can be “attached” to the MS, FT-ed and used when needed (replacing the need for the ‘Model’ column).

Calibration & Imaging Flow



Calibration

- Correcting antenna positions
- Gain Curves
- Opacity (HF) and Ionospheric (LF) corrections
- Requantizer gain calibration
- Setting the flux density scale
- Delay calibration
- Initial Phase only calibration (HF)
- Bandpass calibration
- Complex gain calibration
- Polarization Calibration
- Setting the flux scales of the secondary calibrators

Prior Calibration

gencal

- *gencal* is a task for various types of corrections:

  'amp' = amplitude correction
  'ph' = phase correction
  'sbd' = single-band delay
  'mbd' = multi-band delay
  'antpos' = ITRF antenna position corrections
  'antposvla' = VLA-centric antenna pos. corrections
  'swpow' = EVLA switched-power gains
  'rq' = EVLA requantizer gains
  'swp/rq' = EVLA switched power gains/req. gains
  'opac' = Tropospheric opacity
  'gc' = Gain curve (zenith-angle-dependent gain)
  'eff' = Antenna efficiency (sqrt(K/Jy))
  'gceff' = Gain curve and efficiency
  'tecim' = Total electron content for ionospheric corrections

Antenna Positions: *gencal*

- Correct baselines after antenna moves
  - operator's log reports recent antenna moves
- Use the task *gencal* to produce a calibration table that will include the antenna position corrections
  - (check whether table was needed/created)

```
caltype  = 'antpos'
caltable = 'antpos.cal'
```

- Baseline correction related information is at:
  - http://www.vla.nrao.edu/astro/archive/baselines/

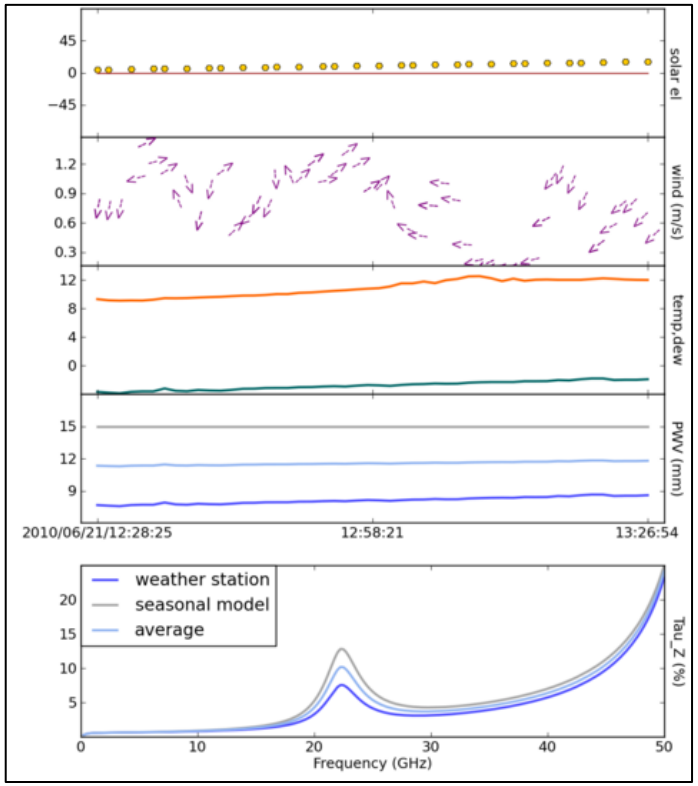
Gain Curves: *gencal*

- Large antennas have a forward gain that changes with elevation.
- Gain curves describe how each antenna behaves as a function of elevation, for each receiver band.
- The polynomial coefficients for the VLA are available directly from the CASA data repository.
- Especially important for higher frequencies.
- In *gencal*, set

```
caltype  = 'gc'
caltable = 'gaincurve.cal'
```

Opacity Corrections (HF): *plotweather*

- Atmospheric optical depth corrections.
- Important for high frequency observations (>15 GHz).
- *Plotweather* is the task to estimate opacities and to make weather plots.
  - ➢ Uses weather statistics and/or seasonal models.
  - ➢ `tau_val = plotweather( vis='my.ms',doPlot=True, plotName='Weather.png')`
  - ➢ Gives one value per spw



Opacity Corrections (HF): *gencal*

- After *plotweather* use *gencal* to make a calibration table using the derived opacities:

```
caltype   = 'opac'
caltable  = 'opacity.cal'
parameter =  tau_val
spw       = '<match to tau_val spws>'
```

Ionosphere Correction: *gencal*

Needed for lower frequency observations (P, L, and S-bands)

- Important for polarimetry
- Under testing

1) Import the TEC image

```
from recipes import tec_maps
tec_image, tec_rms_image, tec_graph =
  tec_maps.create(vis='my.ms',doplot=True)
```

2) Run gencal

```
caltype   = 'tecim'
caltable  = 'tecim.cal'
infile    =  tec_image
```

Requantizer gains: *gencal*

- Optimizes the digital power within each spectral window.
- Required for 3-bit data.
- Strongly recommended for P-band 8-bit data.
- In *gencal*, set

```
caltype  = 'rq'
caltable = 'requant_gains.cal'
```

Setting the flux density scale: *setjy*

- Calculates the absolute flux density as a function of frequency (and time):
  - for standard flux density calibrators (e.g., Perley-Butler 2013)
  - and for Solar System objects (e.g., Butler-JPL-Horizons 2012)
- If provided, attaches a model record to the MS

```
vis                 =    'my.ms'
field               =    '<field name or #>'
spw                 =    ''
scalebychan         =    True
standard            =    'Perley-Butler 2013'
      model         =    '<source/band model name>'
      listmodels    =    False
usescratch          =    False
```

Setting the flux density scale: *setjy*

- Calculates the absolute flux density as a function of frequency (and time):
  - for standard flux density calibrators (e.g., Perley-Butler 2013)
  - and for Solar System objects (e.g., Butler-JPL-Horizons 2012)
- If provided, attaches a model record to the MS

```
vis              =   'my.ms'
field            =   '<field name or #>'
spw              =   ''
scalebychan      =   True
standard         =   'Perley-Butler 2013'
      model      =   '<source/band model name>'
      listmodels =   False
usescratch       =   False
```

Calibration: *setjy*

- `listmodels`
  - If `True`, the task will only list the available primary calibrator models (3C138, 3C147, 3C286, 3C48; at L, S, C, X, U, K, A, Q bands --- P band not yet available).
  - If `False`, the task will calculate the flux density.

- `usescratch`
  - If `True`, the 'Model' column will be created. This will increase the size of the MS.
  - If `False`, the model is simply attached to the MS. When needed, it will be FT-ed and used.

Setting the flux density scale: *setjy*

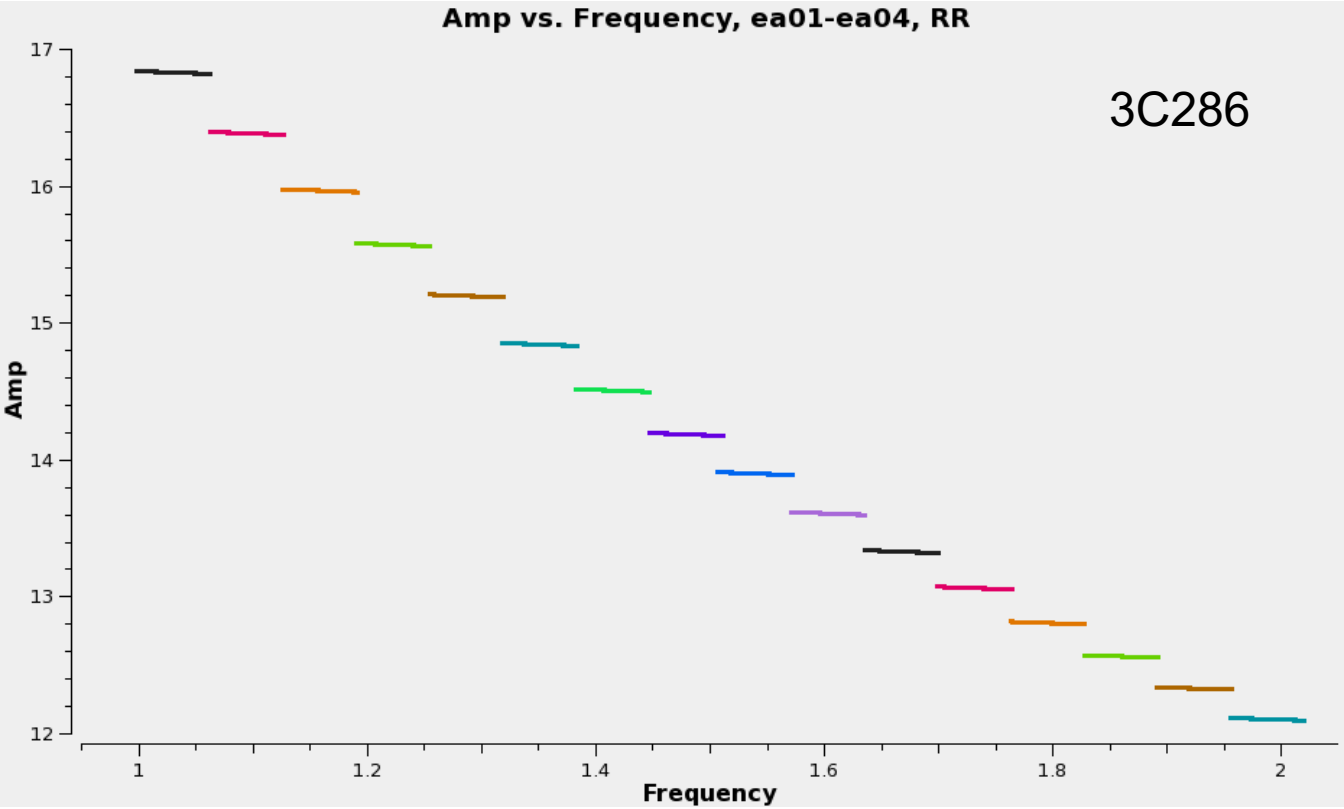
- Calculates the absolute flux density as a function of frequency (and time):
  - for standard flux density calibrators (e.g., Perley-Butler 2013)
  - and for Solar System objects (e.g., Butler-JPL-Horizons 2012)
- If provided, attaches a model record to the MS

```
vis                 =    'my.ms'
field               =    '<field name or #>'
spw                 =    ''
scalebychan         =    True
standard            =    'Perley-Butler 2013'
      model         =    '<source/band model name>'
      listmodels    =    False
usescratch          =    False
```

Calibration: *setjy*

The `scalebychan` parameter

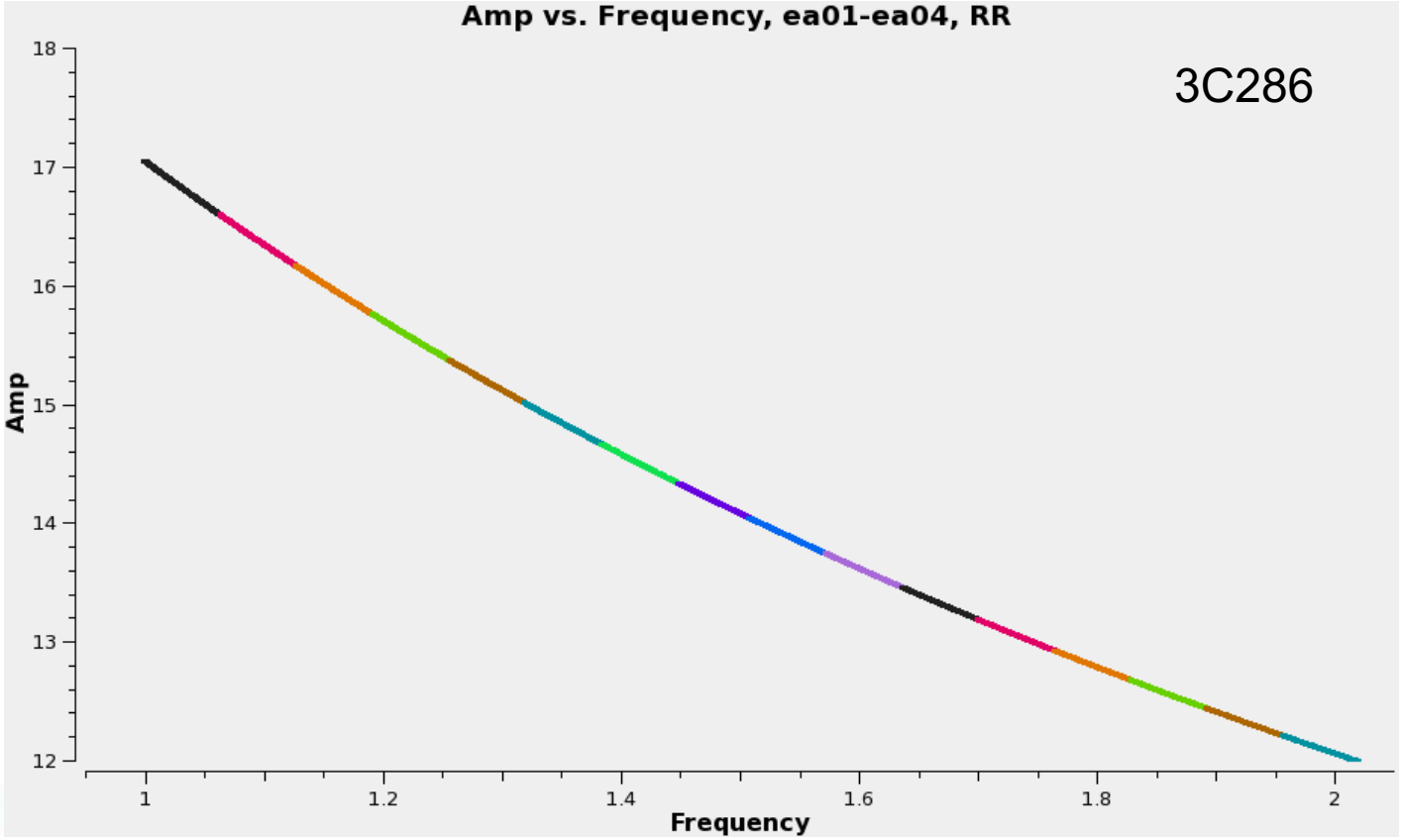
- When **`False`**: The values will be per spectral window.
- **Do not set it to False!**



Calibration: *setjy*

The `scalebychan` parameter

- When **`True`**: The values will be per spectral channel



Setting the flux density scale: *setjy*

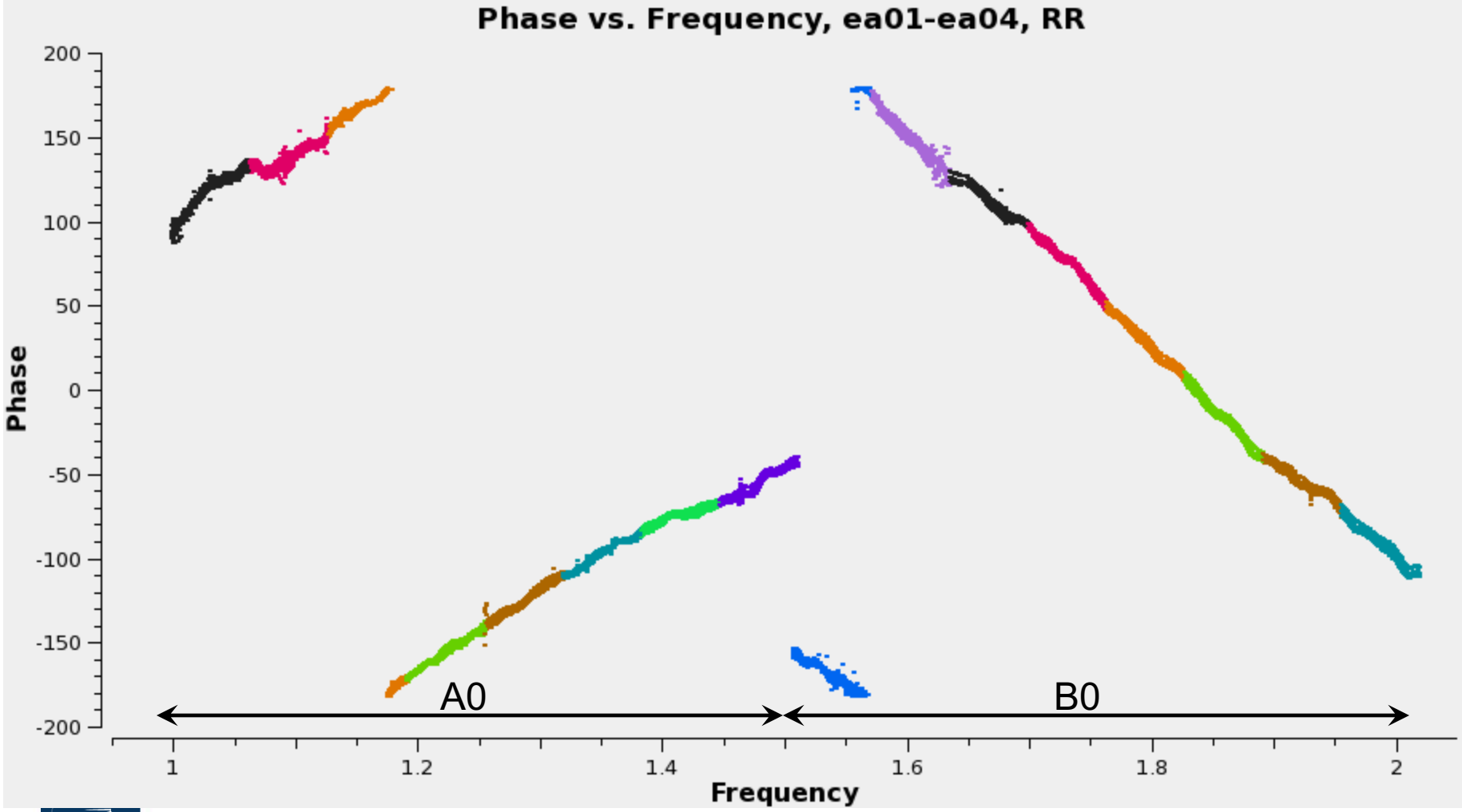
- User can also provide flux density values instead of letting the task calculate them (manual mode)

```
standard        =   'manual'
    fluxdensity     = [1, 0, 0, 0]
    spix            = [alpha,beta]
    reffreq         = '1GHz'
```

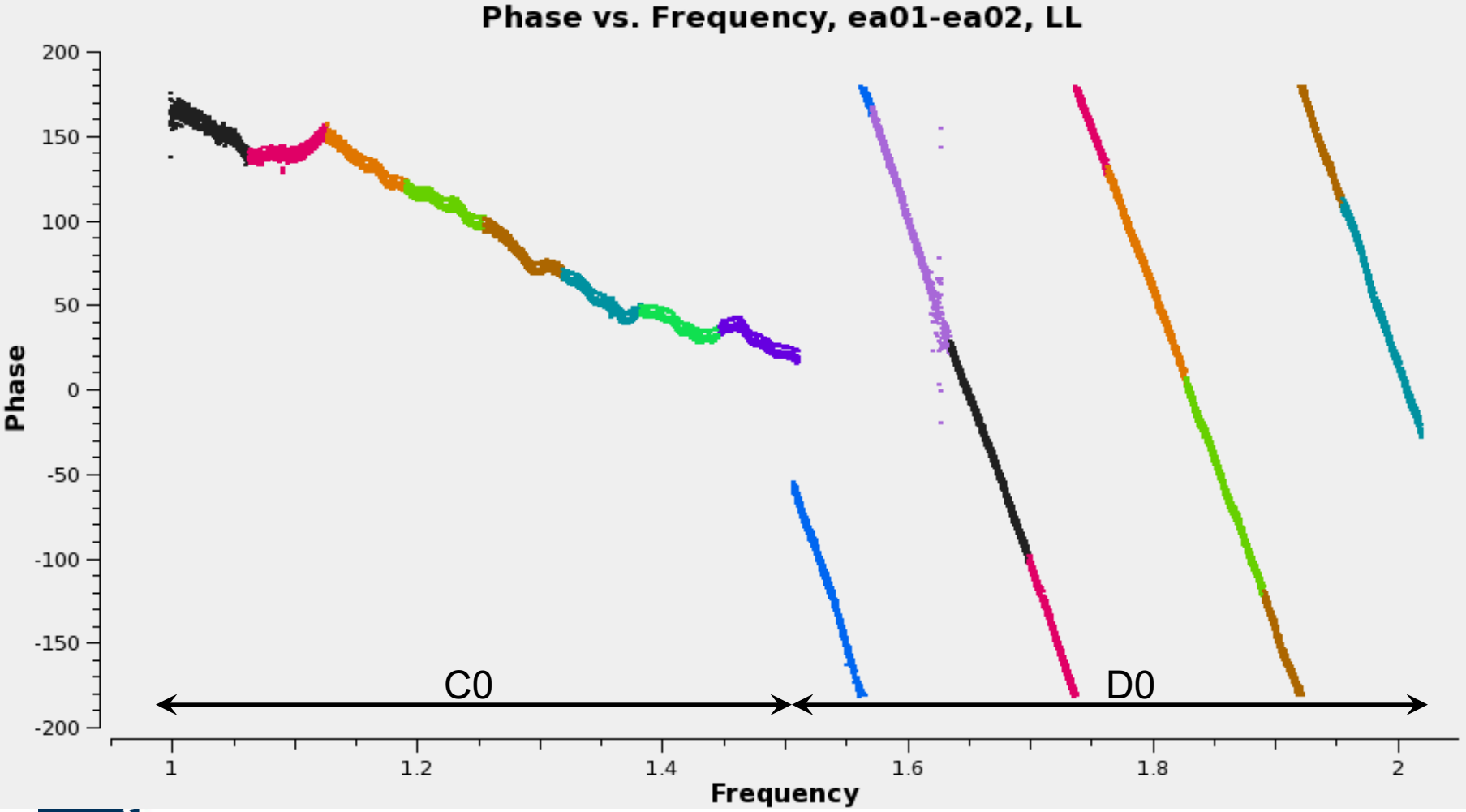
Can also provide

`polindex`: coefficients for frequency dependence of linear polarization fraction
`polangle`: coefficients for frequency dependence of polarization angle
`rotmeas`: rotation measure (rad/m^2)

Delays


Phase vs. Frequency, ea01-ea04, RR
Phase
Frequency
200
150
100
50
0
-50
-100
-150
-200
1
1.2
1.4
1.6
1.8
2
A0
B0

Delays


Phase vs. Frequency, ea01-ea02, LL
Phase
Frequency
200
150
100
50
0
-50
-100
-150
-200
1
1.2
1.4
1.6
1.8
2
C0
D0

Delay Calibration: *gaincal*

```
vis          =    'my.ms'
caltable     =    'delays.cal'
field        =    '<field name or #>'
solint       =    '60s'
refant       =    'ea??'
gaintype     =    'K'
gaintable    =    '[previous cal tables]'
```

- Choose 1 min of data on a strong source (through `selectdata` → `timerange`).
- `refant` should have baselines to all the antennas in the selected time range.
- This is not a Global Fringe Fitting; it solves for antenna based single-band delays or multi-band delays.

A note on the new parameter *docallib*

In current version of CASA, make sure

```
docallib = False
```

- `docallib` refers to a "calibration library", a new portable interface for describing ensembles of calibration replacing `gaintable, gainfield,` etc… parameters.
- Will enable on-the-fly calibration in various tasks.
- Will provide increased capability and flexibility.
- Can be used now but remains limited in some aspects.
- New features, additional flexibility, and broader deployment in more tasks will be offered in later CASA releases.

Before Bandpass Calibration

- Bandpass calibration is not only needed for spectral-line observations, but also for continuum.
- Before calibrating the bandpass, do phase-only calibration on the bandpass calibrator (to be applied when calibrating the bandpass).
  - Prevents decorrelation when vector averaging.
  - Critical for high frequency observations.
  - Can also be used in low frequency observations.

Initial Phase only calibration: *gaincal*

- Run *gaincal* on the bandpass calibrator using:
  - a short solution interval, and
  - a few channels per spw (free of RFI).
- This table should only be used while calibrating the bandpass.
- In *gaincal*, set
  - `caltable = 'bpphase.gcal'`
  - `field = '<bandpass_cal_name_or_#>'`
  - `calmode = 'p'`
  - `gaintype = 'G'`
  - `gaintable = '[various calibration tables]'`
  - `solint = '<a short time interval>'`
  - `spw = 'x~y:n~m'`

Initial Phase only calibration

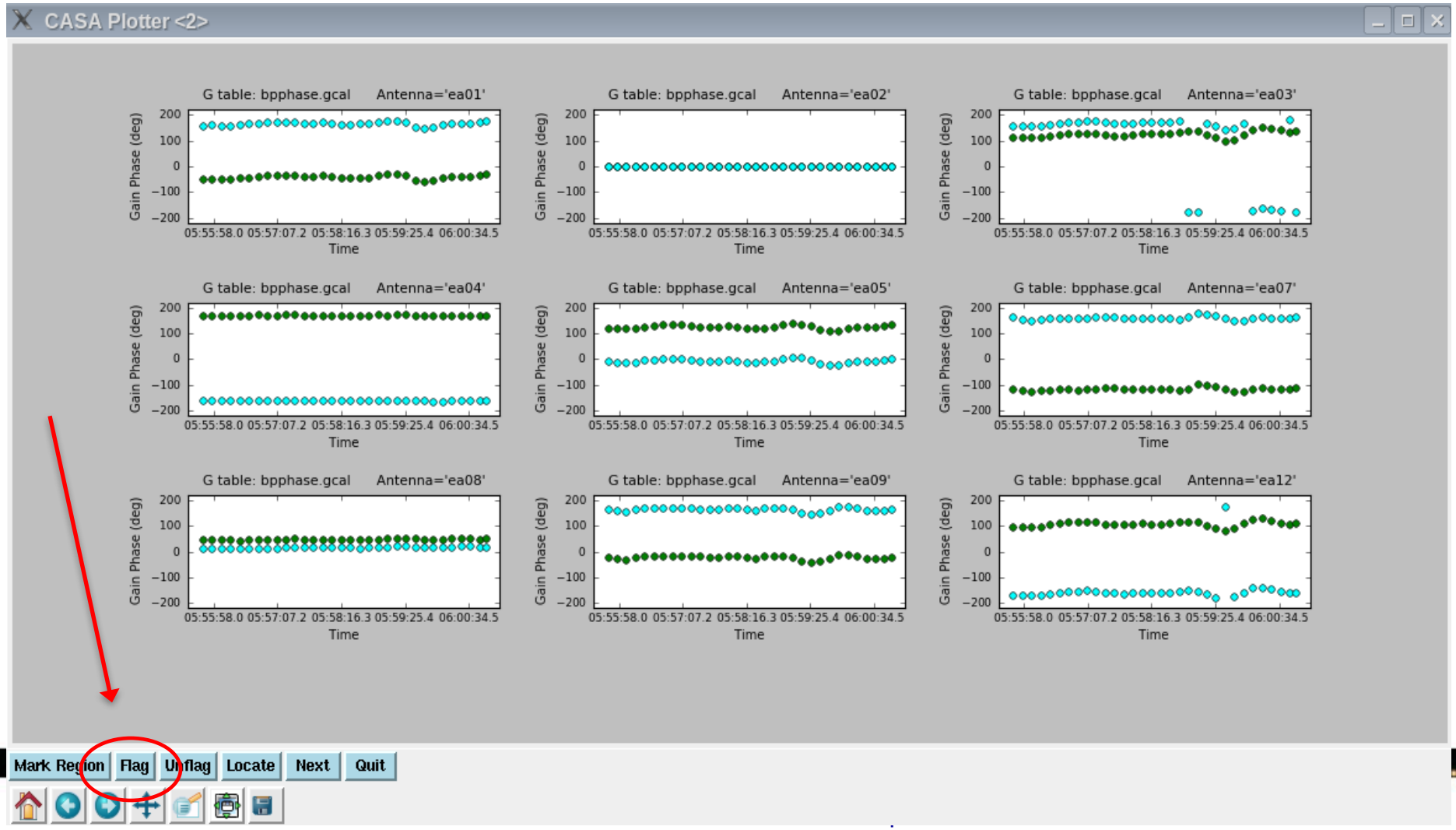
Plotting the solutions: *plotcal*

- *plotcal* is a multi-purpose plotter for calibration results
- To plot the phase calibration results:

```
caltable        = 'bpphase.gcal'
xaxis           = 'time'
yaxis           = 'phase'
spw             = '1'
subplot         =  331
iteration       = 'antenna'
plotrange       =  [0, 0, -200, 200]
```

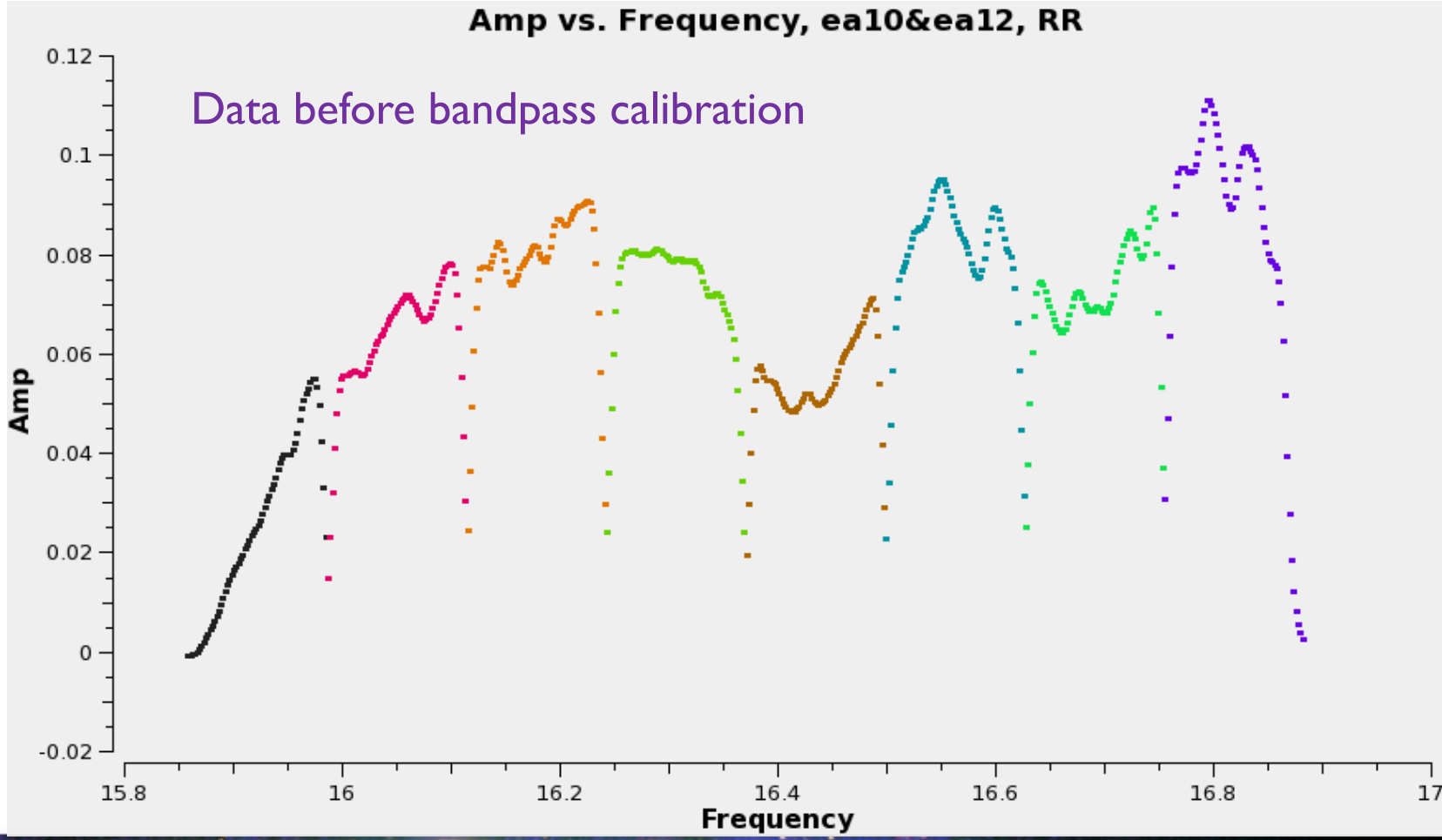
Initial Phase only calibration

Plotting the solutions: *plotcal*



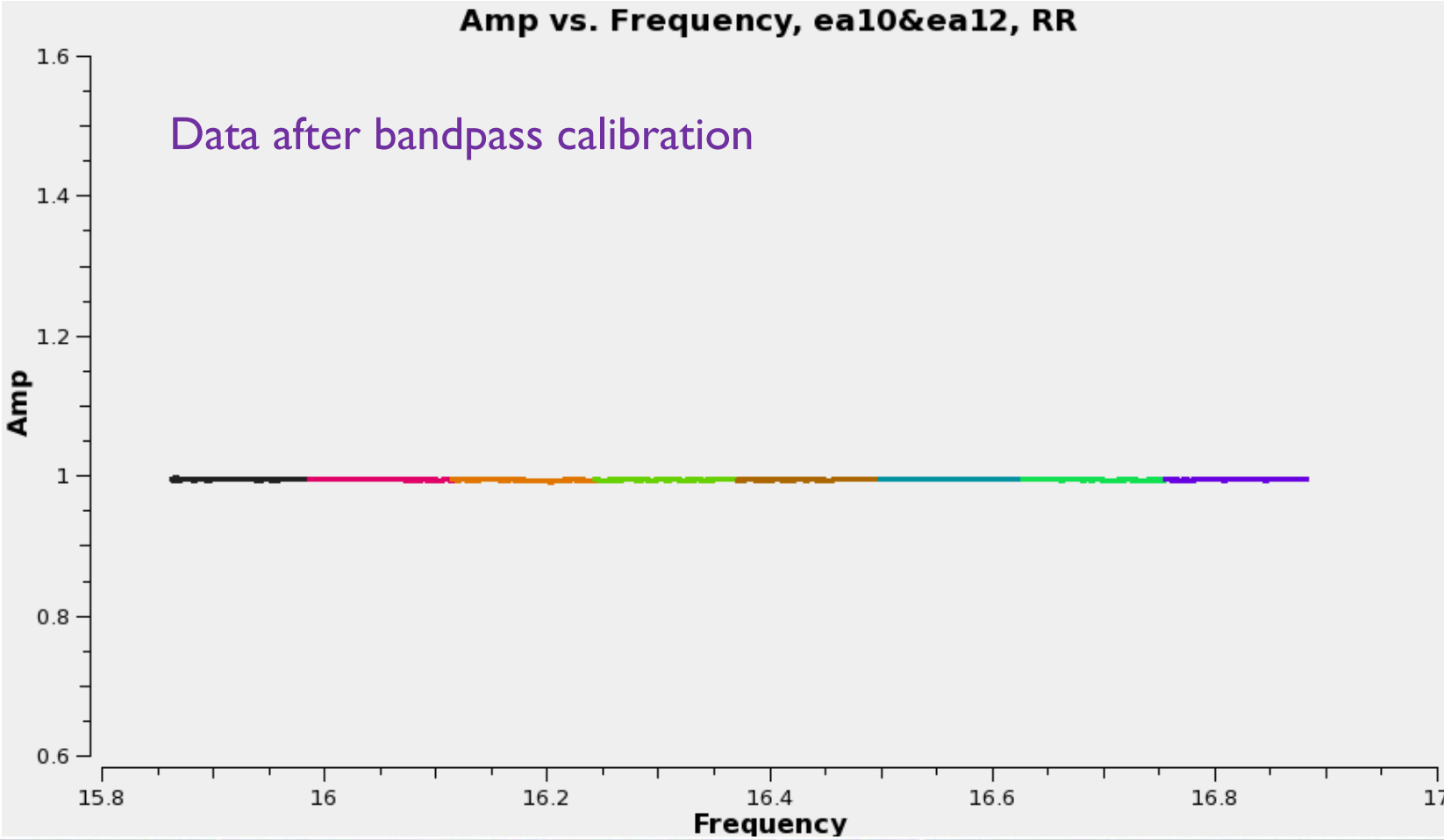
Bandpass Calibration: *bandpass*

- Needed for continuum observations too.



Bandpass Calibration: *bandpass*

- Needed for continuum observations too.



Bandpass Calibration: *bandpass*

```
caltable        =   'bandpass.bcal'
field           =   '<field name or #>'
solint          =     '?'   [time; optionally frequency]
refant          =   'ea??'
solnorm         =    False
bandtype        =   'B'   or   'BPOLY'
gaintable       = [various calibration tables]
```

- If using a source other than the flux calibrator, the spectral index (and spectral curvature) should be accounted for.
- CASA will report these while bootstrapping the flux densities, and store the numbers in a dictionary.
  - Use *setjy* to make use of these values.

Bandpass Calibration

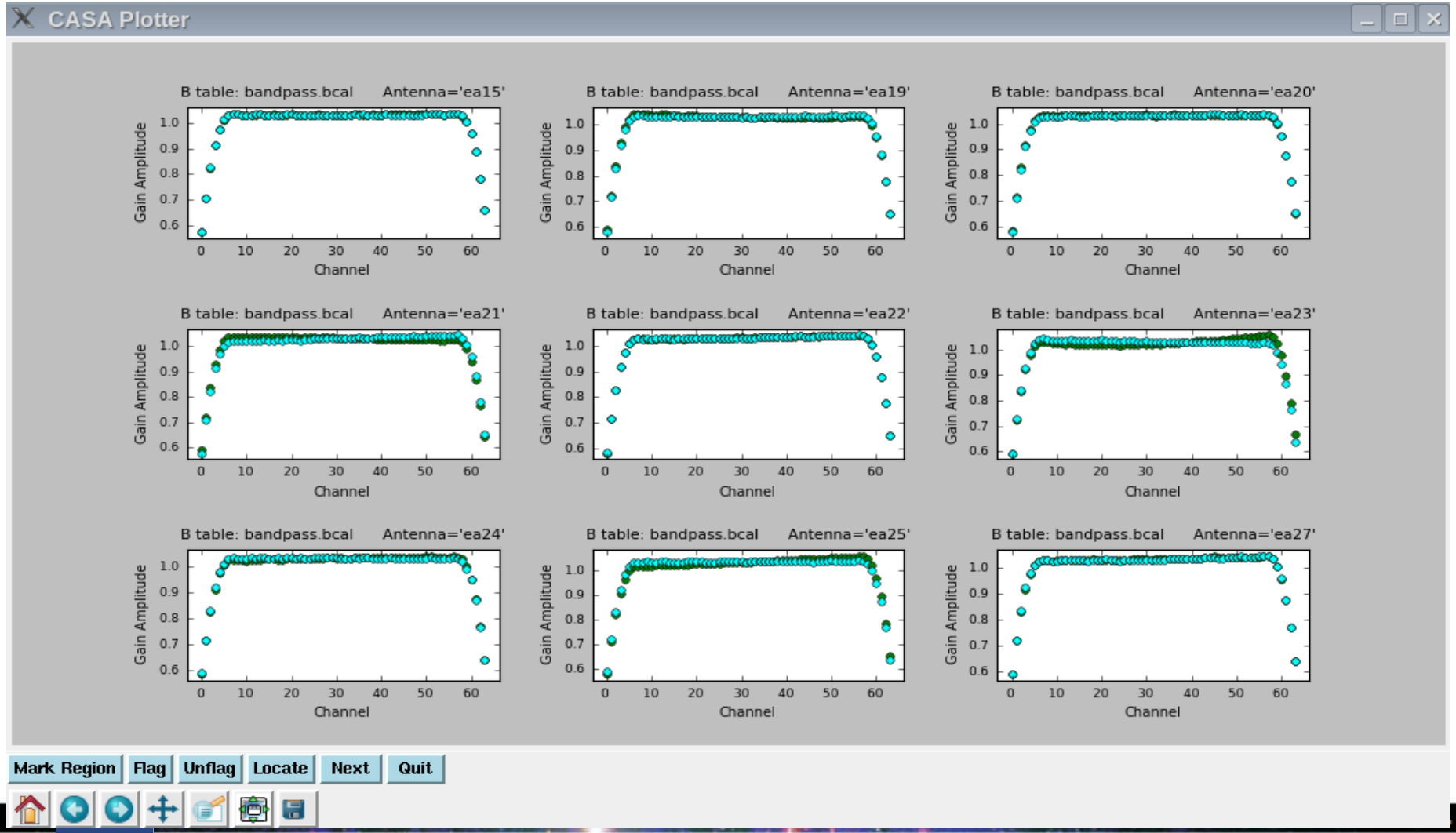
Plotting the solutions: *plotcal*

```
caltable        =   'bandpass.bcal'
xaxis           =   'chan'
yaxis           =   'amp' or 'phase'
spw             =   '1'
subplot         =    331
iteration       =   'antenna'
```

Bandpass Calibration

Plotting the solutions: *plotcal*

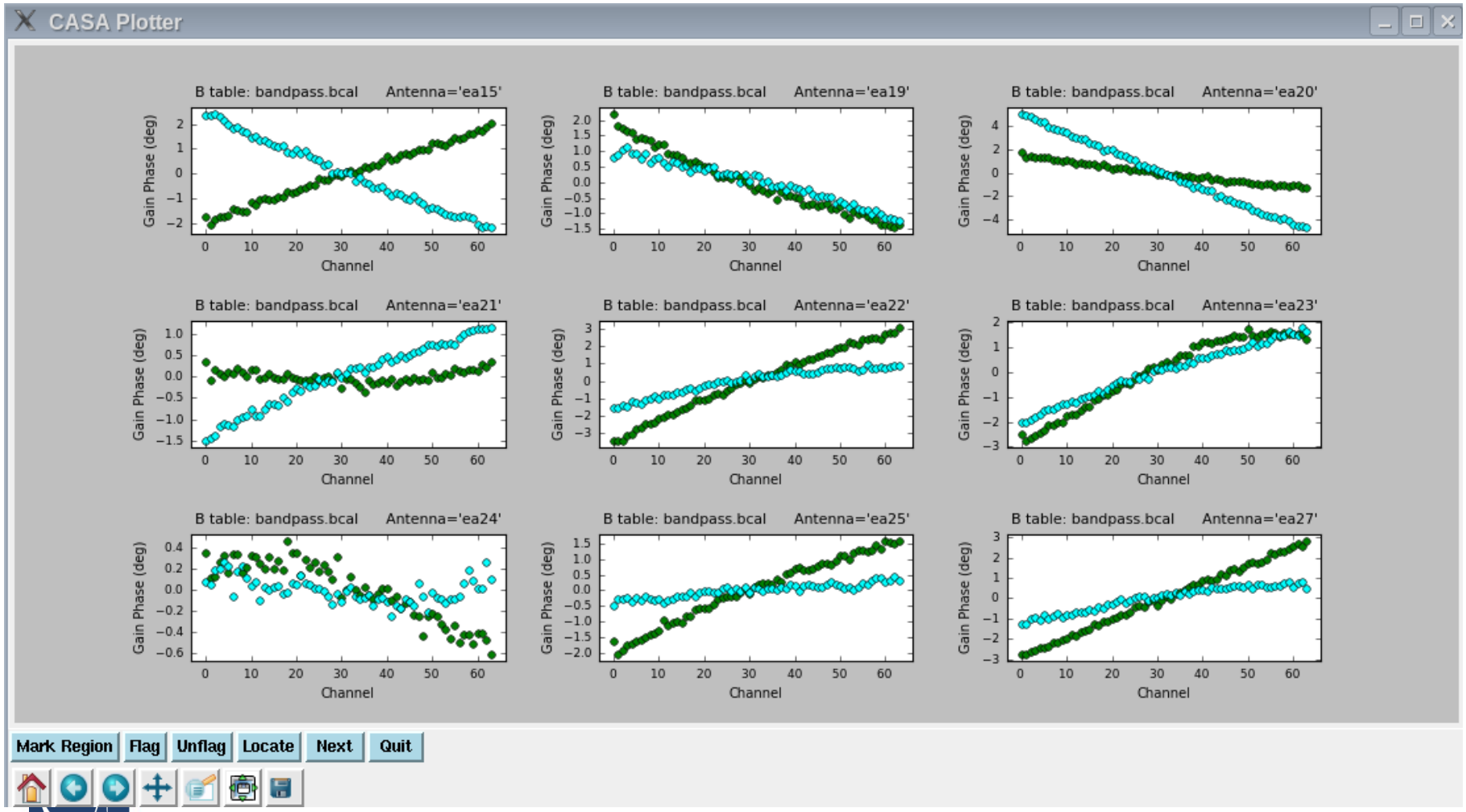
8 MHz spw data set at Ka-band



Bandpass Calibration

Plotting the solutions: *plotcal*

8 MHz spw data set at Ka-band (no delay calibration was performed prior to the bandpass calibration).



Complex Gain Calibration: *gaincal*, High Freq

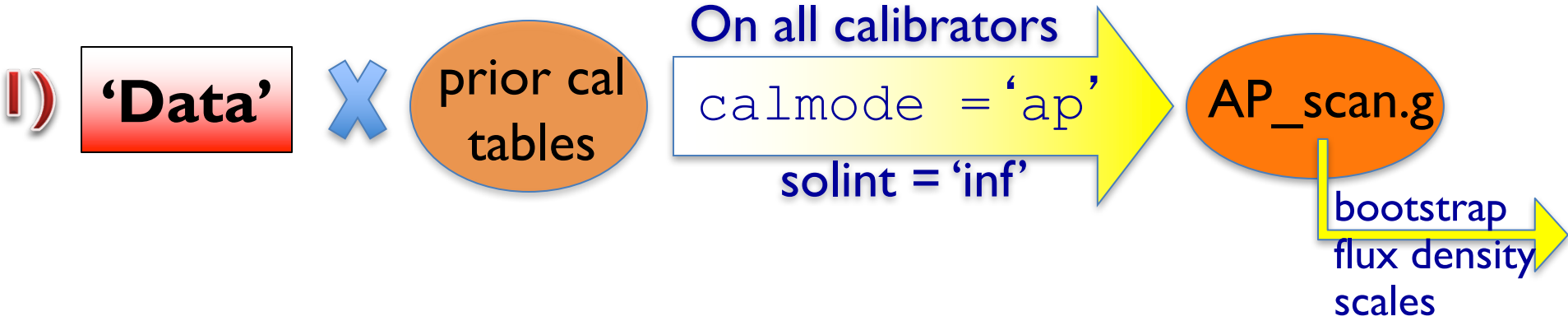
1) **'Data'** X prior cal tables → On all calibrators `calmode = 'p'` short solint → P_ShS.g

2) **'Data'** X prior cal tables, P_ShS.g → On all calibrators `calmode = 'ap'` solint = 'inf' → AP_scan.g → bootstrap flux density scales

3) **'Data'** X prior cal tables, AP_scan.g → On Complex gain calibrator `calmode = 'p'` solint = 'inf' → P_scan.g → calibrate target's phases

Examine the resulting tables with *plotcal*

Complex Gain Calibration: *gaincal*, Low Freq



- Examine the resulting table with *plotcal*.
- If the phases show rapid variations (e.g., due to ionosphere), use the method outlined for high frequencies.
- The VLA calibration pipeline uses the high-freq approach.

Polarization Calibration

- *Gaincal*
  - solving for the cross-hand delays
- *Polcal*
  1. solving for the leakage terms
  2. solving for the R-L polarization position angle

➢ For VLA observations, and particularly for wide bandwidth observations: have channel based solutions for the leakage terms and for the R-L polarization position angle.

➢ Both CASA and AIPS allow solving for these per spectral channel.

Polarization Calibration: *gaincal*

- For polarization calibration, you will typically observe
  - A source to calibrate the leakage terms (this can be a polarized or an unpolarized source), ***and***
  - A source with very well known polarization characteristics to calibrate the polarization position angle.
- Before running *polcal*, calibrate the cross hand delays (critical if your leakage calibrator is polarized):
  - Use the (polarized) position angle calibrator (use *setjy* to provide its Q and U values).
  - Run *gaincal* with `gaintype = 'KCROSS'`
  - Examine the resulting table with *plotcal*.
  - Apply the resulting table in subsequent steps.

Polarization Calibration: *polcal*

1. Solving for the leakage (D) terms (instrumental pol.)

- For an unpolarized calibrator (*Q=U=0*):
  - Use `poltype = 'Df'` to solve for the leakage terms (D) on a per-channel (f) basis.
    - Single scan
- For a polarized calibrator with unknown polarization:
  - Use `poltype = 'Df+QU'` to solve for channel-based leakage terms & apparent source polarization.
    - Several scans, and a good parallactic angle coverage

Examine the resulting tables with *plotcal*.

Polarization Calibration: *polcal*

2. Solving for the R-L polarization position angle

- To obtain an accurate polarization position angle, the R-L phase needs to be calibrated.
- In *polcal*, use `poltype = 'Xf'` for a frequency dependent polarization position angle calibration.
- Requires the use of a (polarized) source with known polarization angle (use *setjy* to set its Q and U values).
- Examine the resulting table with *plotcal*.

Scale flux density: *fluxscale*

- Bootstraps the flux density scale of the secondary calibrators.
- Uses the scan-based ‘ap’ gain table

AP_scan.g

```
vis          =  'my.ms'
caltable     =  '<input ap table>'
reference    =  '<field # of flux cal>'
fitorder     =   1 or 2
fluxtable    =  '<output cal table>'
incremental  =   True or False
```

- Reports the flux density values per calibrator per spw.
- Fits across the spws of each calibrator to report a *spectral index and curvature* (can be supplied through *setjy* if needed).

Scale flux density: *fluxscale*

- Bootstraps the flux density scale of the secondary calibrators.
- Uses the scan-based ‘ap’ gain table

AP_scan.g

```
vis            =  'my.ms'
caltable       =  '<input ap table>'
reference      =  '<field # of flux cal>'
fitorder       =   1 or 2
fluxtable      =  '<output cal table>'
incremental    =   True or False
```

- Reports the flux density values per calibrator per spw.
- Fits across the spws of each calibrator to report a *spectral index and curvature* (can be supplied through *setjy* if needed).

Scale flux density: *fluxscale*

```
fluxtable          =  '<output cal table>'
incremental        =    True or False
```

- If `incremental = False`

  The `output table` replaces the input '`ap`' table.

- If `incremental = True`

  The `output table` contains only the scaling factors, and should be used alongside the input '`ap`' table.

Apply Calibration: *applycal*

```
field          =   '<field name or #>'
interp         =   nearest or linear
gaintable      =   [various calibration tables]
gainfield      =   fields corresponding to the
                       above tables
parang         =   False (True if polcal was run)
calwt          =   False
```

- Calibrate one field at a time (but targets with the same calibrators can be grouped together).
- Use the appropriate tables for each source.
- Make sure to match the gainfield entries with the gaintables.

Examine the calibrated data (the corrected column) with *plotms*.

Flag, if needed, and re-calibrate.

The VLA Calibration Pipeline

- Performs basic flagging and calibration using CASA.
- Primarily designed for Stokes `I` continuum data.
- To run successfully, the scan intents in the scheduling block must be set correctly.
- Information are at:

https://science.nrao.edu/facilities/vla/data-processing/pipeline

- (Many more details in later talks in this workshop.)

Split the target(s): *split*

- Split the target source(s) using the corrected column.
- Optionally:
  - apply time averaging
  - apply frequency averaging
  - choose certain antennas
  - choose spectral windows/channels
  - choose a certain UV range
  - choose particular scans
  - choose polarization
- The *split* out data will occupy the 'data' column in the output MS.
- Self-calibration can be performed if the target is strong enough.
- Self-calibrated data will be placed in the corrected column (upon running applycal).

Weights:

- The weights are initialized to be channel bandwidth and time dependent ($2\Delta\nu\Delta\tau$).
- Weight values are per spw.
- CASA infrastructure supports weight spectrum (by channel):
  - Not yet used in VLA data

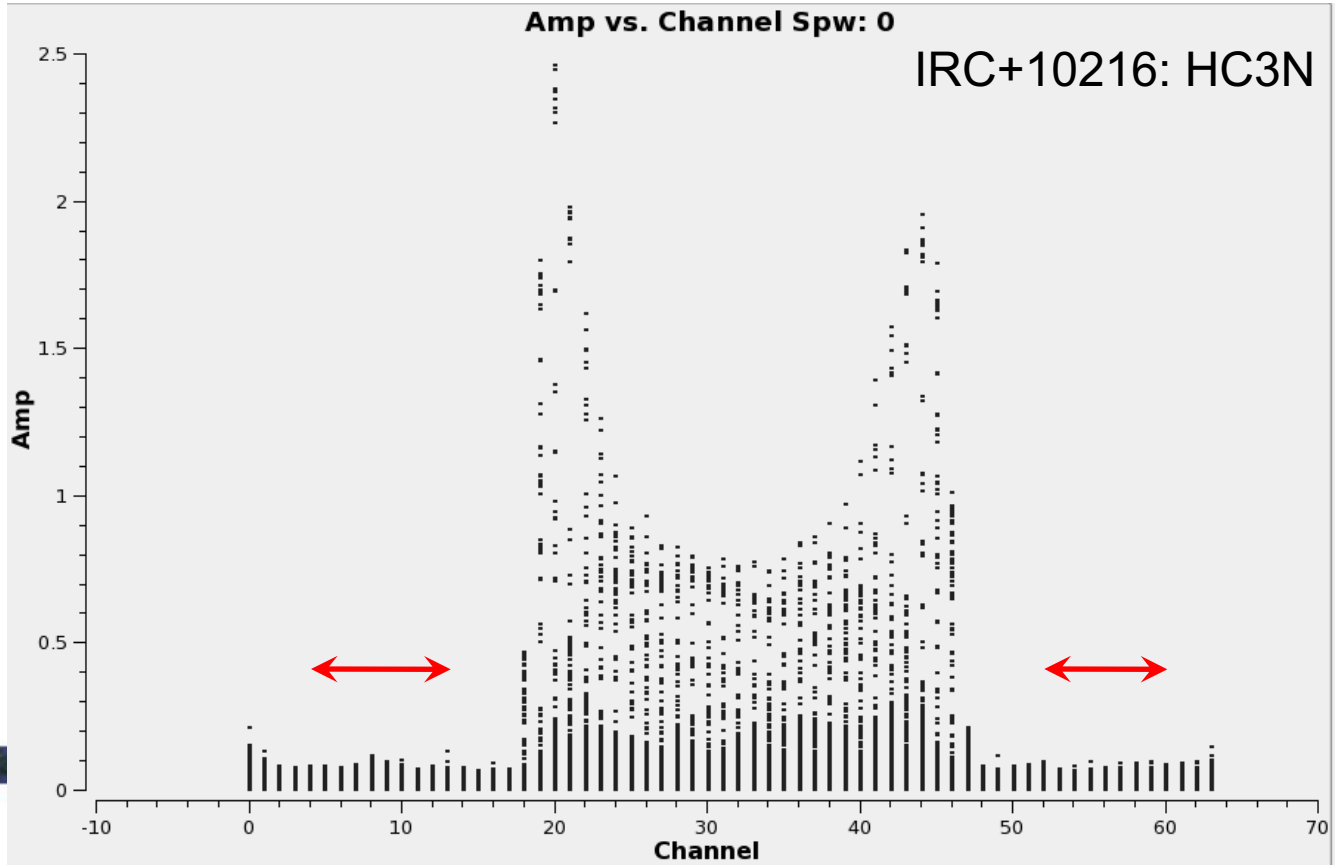
(Re)Weighting visibilities: *statwt*

- *statwt* re-weights the visibilities according to their scatter.
- Needed to down-weight underperforming antennas, or spw's affected by RFI.
- How/when to use it:
  - The data should be fully calibrated.
  - Recommended to *split* the data (source) of interest first (some time averaging might be helpful).
  - Spectral lines: use `fitspw` to exclude strong lines
- Channelized version of *statwt* is coming soon (*statwt2*)

Continuum Subtraction: *uvcontsub*

```
vis              = 'my.ms'
fitspw           = '0:4~13;52~60'
excludechans     =  False  or  True
want_cont        =      False
```

can choose multiple spws



Doppler Correction: *cvel* → *cvel2*

- The VLA offers Doppler setting, NOT Doppler Tracking
- The line of interest may shift over one or more channels during the observations.
- If adding different observing blocks, one can choose to first Doppler correct (*cvel2*) each block, concatenate (*concat*) and then image (*clean* or *tclean*).
- Imaging task *clean* can do this on-the-fly BUT:
  - *cvel2* must be run if one needs/wants to do self-calibration using a (narrow) strong spectral line.

The End of Part II