JIVE
Joint Institute for VLBI
ERIC

VLBI (Amplitude) Calibration

(and other a-priori calibration)

Mark Kettenis, JIVE

OPTICON RadioNet Pilot

CASA VLBI workshop

Measurement Equation
(RIME)

- Formulated by: Hamaker, Bregman & Sault, 1996, A&AS, **117**, 137
- Reformulated in: Smirnov, 2011, A&AS, **527**, A106
- Mathematical basis for calibration of a radio interferometer
- Fully incorporates polarization

Electric field at the source: $\mathbf{e} = \begin{pmatrix} e_r \\ e_l \end{pmatrix}$

Recorded voltages of feeds at telescope: $\mathbf{v} = \mathbf{J}\mathbf{e}$ with (2x2) Jones matrix $\mathbf{J}$

Visibility matrix produced by the correlator: $\mathrm{V}_{pq} = 2\langle \mathbf{v}_p \mathbf{v}_q^H \rangle$

Measurement equation: $\mathrm{V}_{pq} = 2\langle \mathbf{J}_p (\mathbf{e}_p \mathbf{e}_q^H) \mathbf{J}_q^H \rangle = \mathbf{J}_p \mathbf{B} \mathrm{J}_q^H$ with brightness matrix $\mathrm{B} = \begin{pmatrix} I+Q & U+iV \\ U-iV & I-Q \end{pmatrix}$

Goal is to determine $\mathbf{J}_p$ for all antennas p.

Measurement Equation
continued

$\mathbf{J}_p = \mathbf{B}_p\mathbf{G}_p\mathbf{D}_p\mathbf{E}_p\mathbf{P}_p\mathbf{K}_p\mathbf{T}_p$

- $\mathbf{T}_p$ Polarization-independent multiplicative effects introduced by the troposphere, such as opacity and path-length variation.
- $\mathbf{K}_p$ Delay (this is VLBI!)
- $\mathbf{P}_p$ Parallactic angle, which describes the orientation of the polarization coordinates on the plane of the sky. This term varies according to the type of the antenna mount.
- $\mathbf{E}_p$ Effects introduced by properties of the optical components of the telescopes, such as the collecting area's dependence on elevation.
- $\mathbf{D}_p$ Instrumental polarization response. "D-terms" describe the polarization leakage between feeds.
- $\mathbf{G}_p$ Electronic gain response due to components in the signal path between the feed and the correlator.
- $\mathbf{B}_p$ Bandpass (frequency-dependent) response, such as that introduced by spectral filters in the electronic transmission system.

CASA always applies these in the same (physically correct) order!

CASA calibration

SN table

- CASA calibration tables represent Jones matrices
  - Have an identity
  - Contain real or complex parameters that are used to calculate elements
    Complex gain: $\mathbf{G} = \begin{pmatrix} g_r & 0 \\ 0 & g_l \end{pmatrix}$ is described by tow complex paramaters.
  - Can be given arbitrary (meaningful) names
- **Always explicitly specify calibration tables to be applied!**
  - There is no equivalent of an AIPS CL table

CASA calibration

continued

- Calibration tables are specified with task parameters:
  - `gaintable = [`*`caltable1`*`, `*`caltable2`*`]`
  - `gainfield = [`*`field1`*`, `*`field2`*`]` e.g. `'3C84', 'J1023+43'`
    (*`field1`* applies to *`caltable1`*, *`field2`* to *`caltable2`*)
  - `interp = [interp1, interp2]` e.g. `'linear', 'nearest'`
    (*`interp1`* applies to *`caltable1`*, *`interp2`* to *`caltable2`*)
  - `parangle = True` or `False` (default)
- Per-scan interpolation modes:
  - `'linearperscan', 'nearestperscan'`

new in CASA 6.5.5

- Data without calibration solutions is automatically flagged!
  - Can be bypassed when applying the final calibration
- Data is aggressively flagged if it is partly flagged:
  - `corrdepflags = True` or `False` (default); `True` prevents flagging both pols if one is flagged

Data Formats

- MeasurementSet (v2)
  Native data format of CASA; MS for short
- UV-FITS
  What AIPS writes
- FITS-IDI
  Produced by the SFXC (EVN) and DiFX (VLBA, LBA, …) correlators

All thee formats can contain metadata such as gain curves and T_{sys}

VLBI amplitude calibration

- VLBI observations typically use 2-bit sampling
- For maximum efficiency the 4 states should be sampled ~17% in the “high” state and 33% in the “low” state.
- This is done by automatically adjusting the gain on a (relatively) short timescale
- As a consequence all amplitude information is lost in the correlated visibilities
- Standard CASA amplitude calibration using the setjy/fluxscale tasks does not work:
  - Calibrater sources are either variable (in time) or resolved at VLBI scales!

VLBI amplitude calibration
continued

- System equivalent flux density

  $$\text{SEFD} = \frac{T_{sys}}{G}$$

  where *G* is the antenna gain

  $$G = \text{DPFU} \times g(el)$$

- DPFU: “degrees per flux unit”, conversion factor from temperature scale (K) to flux density scale (Jansky)

- g(el): gain curve; correcting for deformation under gravity of the dish (normalized)

- Flux density on a particular baseline

  $$S_{i,j} = \sqrt{\text{SEFD}_i \cdot \text{SEFD}_j} \cdot r_{c,i,j}$$

  where $r_{c,i,j}$ is the **normalised** correlation coefficient

VLBI amplitude calibration

T_{sys} measurement methods

- Classic noise diode:
  - Noise diode get fired in gap just before start of scan
  - Backend tracks total power; this is then used to extrapolate T_{sys}
  - Must flag data when noise diode is on
- “Continuous Cal”
  - Noise diode is turned on and off at a rate of (typically) 80 Hz
  - Can track T_{sys} directly
  - Lower power of noise diode means data does not have to be flagged
- Chopper or Hot/Cold load
  - Places an object twith properties similar to a blackbody of known temperature in front of the receiver
  - Typically used for mm-VLBI

$$T_{sys} = \frac{P_{off}}{P_{on} - P_{off}} T_{inject}$$

Preparing your data

ANTAB

scripts at https://github.com/jive-vlbi/casa-vlbi

- Attach gain curves and DPFU from ANTAB files
    - Using append_gc.py script:

      ```
      casa —-no-gui -c append_gc.py antabfile idifile
      ```

    - Using casavlbitools python module:

      ```
      import casavlbitools.fitsidi
      casavlbitools.fitsidi.append_tsys(antabfile, idifiles)
      ```

- Attach T_{sys} measurements from ANTAB files
    - Using append_tsys.py script:

      ```
      casa —-no-gui —c append_tsys.py antabfile idifiles…
      ```

    - Using casavlbitools python module:

      ```
      import casavlbitools.fitsidi
      casavlbitools.fitsidi.append_tsys(antabfile, idifiles)
      ```

    - Provide the names of **all** FITS-IDI files here)!

VLBA data already includes this metadata

new EVN data also (from mid 2022)

Preparing your data
Some installation notes

- On a Mac, create symlinks for CASA:
  - run `!create-symlinks` command from the CASA prompt
- Set the PYTHONPATH environment variable:
  - `export PYTHONPATH=$PYTHONPATH:/path/to/casa-vlbi`
- Scripts will emit a (harmless) waring message:
  - `PyFITS is deprecated, please use astropy.io.fits`
- Scripts may appear in a future CASA release
  - Including tools to add ANTAB information directly to MS!

Importing your data

FITLD

- FITS-IDI data can be imported using the importfitsidi
    - A single FITS-IDI file:

```
importfitsidi(vis=ms, fitsidifiles=[fitsfile],
                  scanreindexgap_s=seconds)
```

15 seconds is good (matches FITLD)

- Multiple FITS-IDI files for a single observation:

```
importfitsidi(vis=ms, fitsidifiles=[fitsfile1, fitsfile2],
                  constobsid=True, scanreindexgap_s=seconds)
```

Use Python glob module for EVN data

```
import glob
fitsfiles =  sorted(glob.glob("N20C2_1_1.IDI*"))
```

- Applies digital corrections for DiFX correlator (VLBA & Co)
- Data is marked to be in a geocentric frame
  (**incorrect for EVN data correlated before 2012!**) (EVN & Co)
- Warnings about telescope diameter and scan numbers can be ignored

- UVFITS data can be imported using importuvfits

```
importuvfits(vis=ms, fitsfile=[fitsfile])
```

This does not import most of the VLBI metadata correctly!

Normalizing your data

AIPS ACCOR

- Fix correlation amplitudes based on autocorrelations (VLBA & Co)

```
accor(vis=ms, caltable=caltable)
```

- Generates G-type calibration table

- CASA data selection provides AIPS `ACSCL` functionality

Flagging your data

UVFLG

scripts at https://github.com/jive-vlbi/casa-vlbi

- Apply a-priori flagging (EVN & Co)

```
$ flag.py uvflgfile fitsfile > flagfile
```

```
flagcmd(vis=ms, inpmode='list',inpfile=flagfile)
```

- Apply a-priori flagging (VLBA)

```
flagcmd(vis=ms, inpmode='table')
```

- Additional (interactive) flagging can be done using `plotms`

A Diverse bunch

Amplitude calibration

Dr. Schorsch, https://creativecommons.org/licenses/by-sa/3.0/deed.en

ESO, https://creativecommons.org/licenses/by-sa/3.0/deed.en

Alessandro Cattani

JJ Harrison, https://creativecommons.org/licenses/by-sa/3.0/deed.en

Amplitude calibration

ANTAB

- Generate caltables for gain curves:

  `gencal(vis=ms, type='gc', caltable=gctable)`

- Generate caltables for T_{sys}:

  `gencal(vis=ms, type='tsys', caltable=tsystable, uniform=False)`

  Using uniform=False is important; without it your data will be (mostly) flagged!

- Generates G-type calibration tables
- To apply use:

  `gaintable=[gctable, tsystable]`

  In subsequent calibration tables.

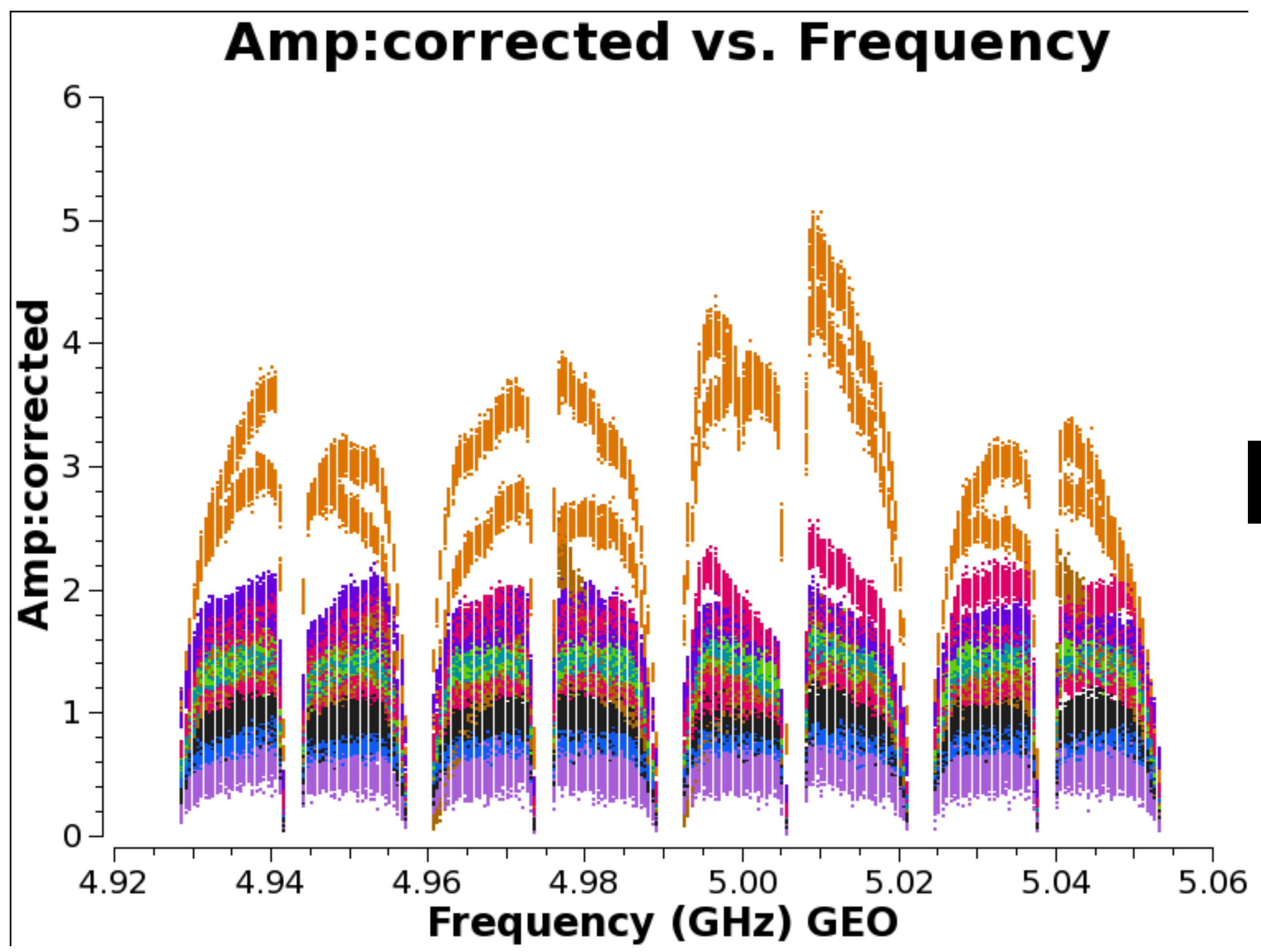
Bandpass calibration

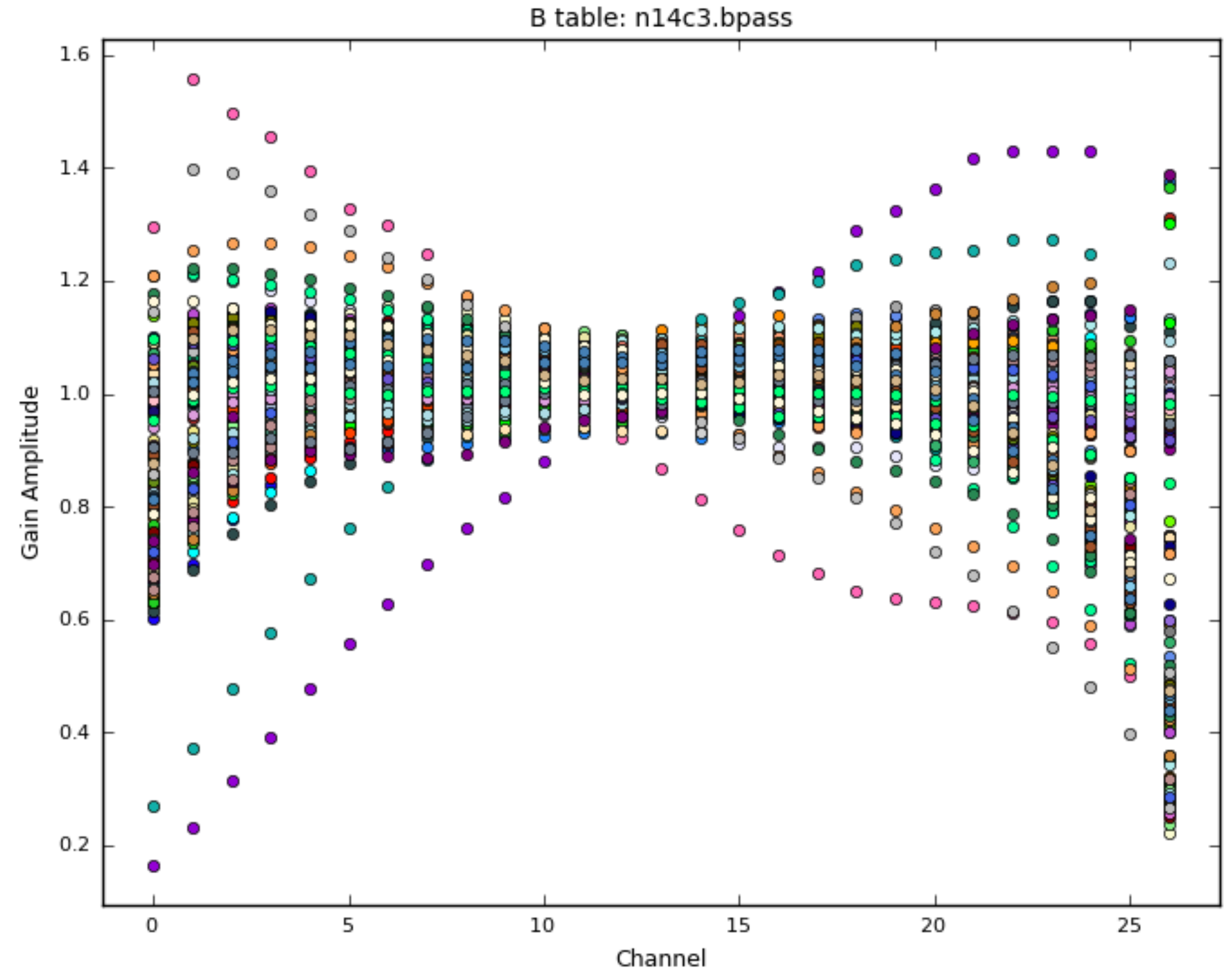
BPASS

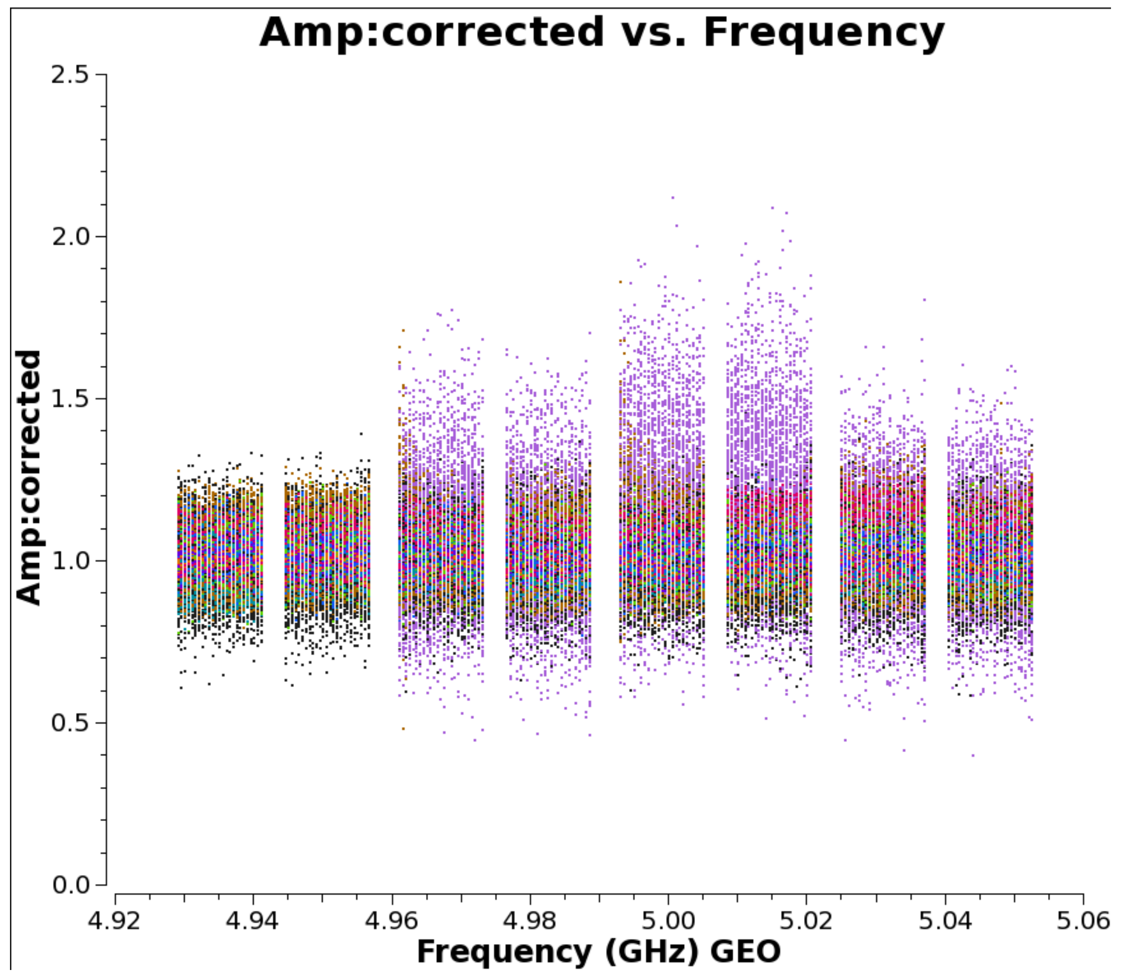
- Generate caltables for gain curves:

```
bandpass(vis=ms, field=field, refant=refant,
    gaintable=[…], solnorm=True, caltable=bptable)
```

- Generates B-type calibration tables



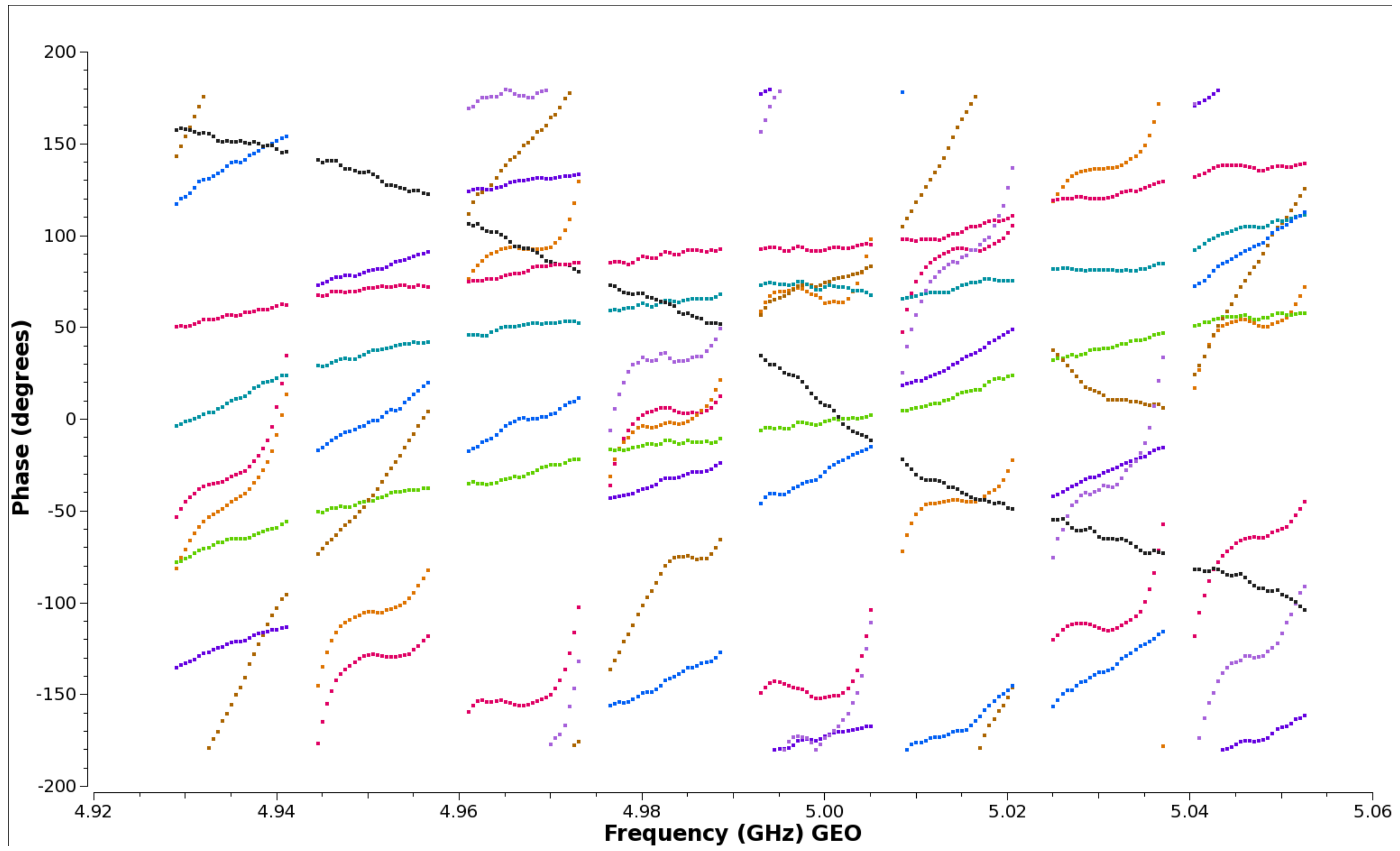


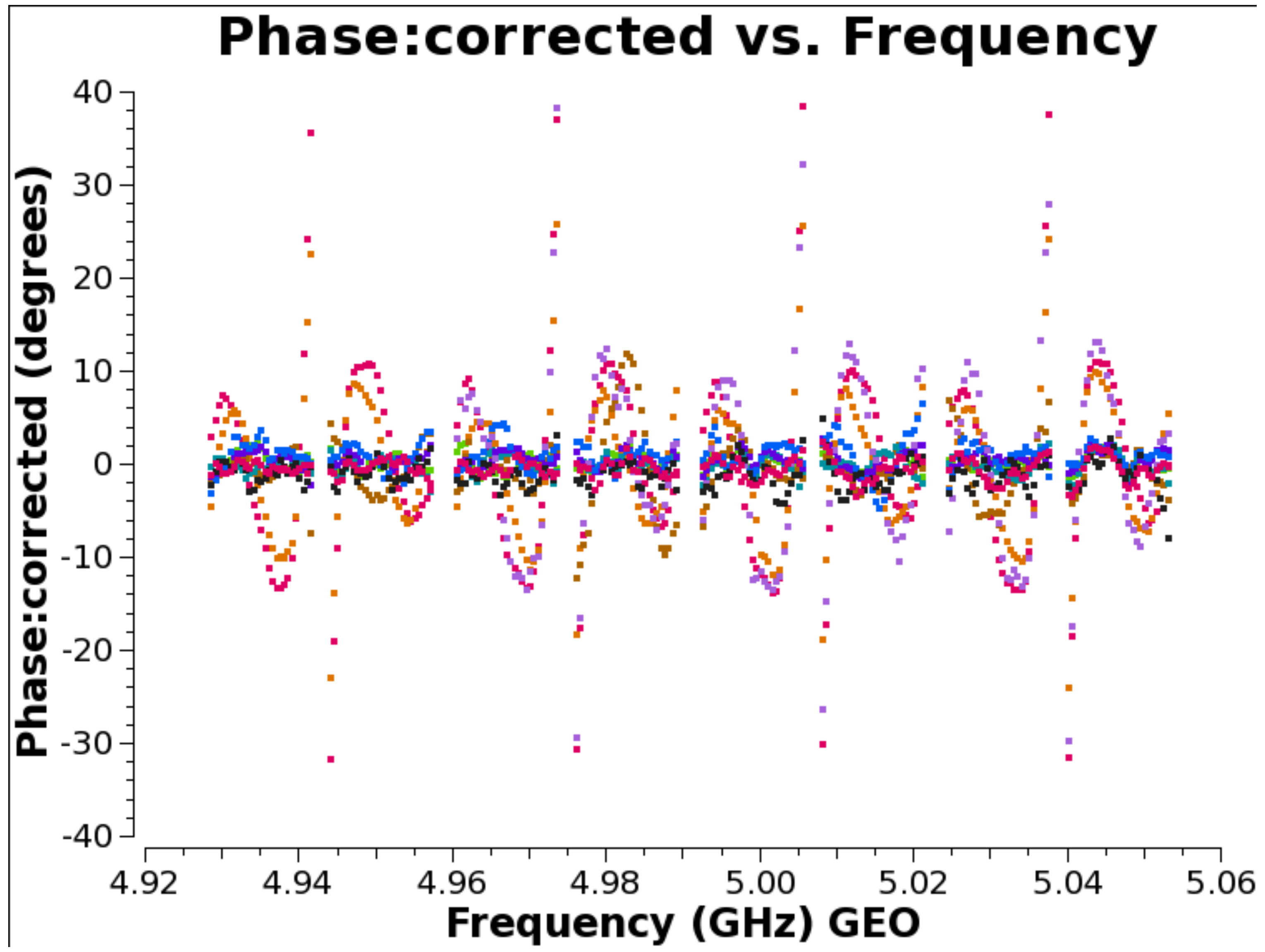


Fringe Fitting

FRING

- See lecture by Des Small on tuesday





Apply calibration

SPLIT

- Applying calibration to the whole MeasurementSet:

```
applycal(vis=ms, gaintable=[…], interp=[…], …)
```

  - Adds a CORRECTED_DATA colum; full copy of the data
- Split the MeasurementSet:

```
split(vis=ms, outputvis=splitms, field=field, …)
```

  - Supports averaging (time & frequency)
  - Needs to be run for each field you want to image
  - The `mstransform` task can also be used.
    - Ends up running the same code.

Tasks under development

EOP correction

CLCOR

- EOP (Earth Orientation Parameters) correction task
  - EOPs are used by correlator to calculate delays
  - EOPs have to measeured/modelled; final values available after a few weeks
  - predicted or non-final values may have been used during correlation
  - New task will make appropriate phase corrections
  - **Important for astrometry!**

Tasks under development

Ionospheric correction

TECOR

- Apply (dispersive) delay corrections based on TEC (Total Electron Content)
  - Uses TEC maps in IONEX format based on GPS measurements
  - IONEX files are automatically downloaded
  - **Important for low frequencies and wide bandwidths!**
  - **Mechanics are implemented but generate wrong corrections for VLBI**
  - Under investigation
- Generate caltables for ionospheric corrections:

```
from casatasks.private import tec_maps
tec_maps.create(vis=ms, doplot=False, imname=tecmap)

gencal(vis=ms, type='tecim', infile=tecmap, caltable=tectable)
```

- Generates G-type calibration table

Tutorial

- Amplitude calibration tutorial uses N14C3 dataset
- Deliberately explores some of the things that can go wrong!
- Will show you how applying the amplitude calibration will change the visibility weights of baselines.

JIVE

OPTICON RadioNet Pilot

THIS EVENT HAS RECEIVED FUNDING FROM THE EUROPEAN UNION'S HORIZON 2020 RESEARCH AND INNOVATION PROGRAMME UNDER GRANT AGREEMENT 101004719