Methods of VGOS correlation and post-processing

Dan Hoak (for the Haystack Geodesy team)

TOW Correlator Workshop May 8-9 2025

v1.1 updated Aug 22 2025

MIT
HAYSTACK
OBSERVATORY

IVS

Outline

Correlation setup & initial fourfit control file
Correlation prepassing
- Phasecal tones
- Fringe detection and sampler delays
- Clock tuning

Production correlation
ffres2pcp
fourphase
lxy fringe fit
Proxy cable cal
Data quality checks
- Phasecal behavior
- Bad channels

Other resources:
VGOS Data Processing Manual
https://www.haystack.mit.edu/haystack-observatory-postprocessing-system-hops/

TOW2023 presentations
https://www.haystack.mit.edu/conference-2/tow2023/tow-2023-12th-ivs-technical-operations-workshop-notebook/

Correlation setup

To correlate a VGOS session, you will need:

- VDIF files with the recorded data
- Station logs (downloaded from CDDIS)
- Station STOP messages (from CDDIS or IVS-Ops mailing list)
- The skd file for the session (downloaded from CDDIS)
- EOP estimates for +/-2 days around the session (from IERS bulletins)
- VEX and V2D files from a previous session (most of the information in these files can be re-used)
- An initial fourfit control file to enable fringe-fitting

CDDIS Master schedule (each session page will have START and STOP messages, station logs, and skd schedule file):
https://ivscc.gsfc.nasa.gov/sessions/

IERS Bulletins – use Bulletin A, latest versions (look for FINAL or RAPID estimates):
https://www.iers.org/IERS/EN/Publications/Bulletins/bulletins.html

IERS Bulletin A has Predictions, Rapid, and Final values; use the best available.
The “geteop.pl” script is ok, but I prefer to check the bulletins.

Correlation setup

VEX files contain information related to the VLBI experiment:

- EOPs
- Clock models
- Site positions
- Antenna details
- Channel frequency definitions of the VDIF data
- Polarization assignment
- Schedule
- Source positions

V2D files contain correlator-specific information:

- Frequency resolution and integration time settings (nChan, tInt, etc)
- DiFX packet size (subintNS; can be optimized for correlator architecture)
- List of compute nodes and playback units
- Locations of VDIF files for each station
- VDIF format for each station
- Scans or sources you want to correlate

If everything goes well, it can take several hours to mount the data, setup the VEX and V2D files, and start correlating. Each session has something unique so it's difficult to automate.

VEX file setup

VEX files have the following sections (order does not matter):

GLOBAL
EXPER
MODE
TAPELOG
STATION
ANTENNA
BBC
DAS
FREQ
IF
SITE
TRACKS

These sections essentially never change, aside from adding new stations, updating station positions, updating date ranges, editing session name, etc.

EOP
CLOCK
SCHED
SOURCE

These sections need to be redone for every session

VEX file setup

VEX files have the following sections (order does not matter):

GLOBAL
EXPER
MODE
TAPELOG
STATION
ANTENNA
BBC
DAS
FREQ
IF
SITE
TRACKS
EOP
CLOCK
SCHED
SOURCE

MODE defines which FREQ, IF, BBC, TRACK definitions apply to each station

SITE positions should be updated about once a year!
(This should be the responsibility of the schedulers...)

CLOCK models come from the station logs and peculiar offsets

SCHED and SOURCE definitions come from the skd schedule file

SCHED and SOURCE sections

Once you download the skd file from CDDIS, run sked to generate a VEX file:

```
$ sked vr2502.skd
? vwc vr2502.vex
? ab
```

This VEX file will be formatted for S/X, so only the SCHED, SOURCE, and SITES sections are useful. I copy/paste these into a VGOS VEX file from a previous session.

From the station logs and STOP messages, if you know that a station missed scans or collected bad data, comment out that station from the schedule:

```
 scan 071-0228a;
    start = 2025y071d02h28m16s;
    mode = VGOS;
    source = 1102-242;
    station = Gs :    0 sec :    30 sec :     0 ft : 1A : &n : 1;
    station = Hv :    0 sec :    30 sec :     0 ft : 1A : &ccw : 1;
*   station = Mg :    0 sec :    30 sec :     0 ft : 1A : &n : 1;
*   station = Sa :    0 sec :    30 sec :     0 ft : 1A : &n : 1;
    station = Wf :    0 sec :    30 sec :     0 ft : 1A : &n : 1;
  endscan;
```

This will save time in the correlation and fringe-fitting!

EOPs and SITE positions

CALC will interpolate the Earth orientation around the time of the session to build the delay model. Use 5 days of EOPs, with the session in the middle (3rd) day. From the IERS bulletin:

IERS Final Values	MJD	x "	y "	UT1-UTC s
25 2 13	60719	0.0828	0.3130	0.04707
25 2 14	60720	0.0826	0.3140	0.04675
25 2 15	60721	0.0824	0.3149	0.04631
25 2 16	60722	0.0817	0.3157	0.04592
25 2 17	60723	0.0811	0.3170	0.04562

This is the day of the session (← 25 2 15)

Thoughts on site positions:

```
def KOKEE12M;
   site_type = fixed;
  *site_position = -5543831.745 m : -2054585.590 m :  2387828.974 m; * from skd
   site_position = -5543831.806 m : -2054585.298 m :  2387829.119 m; * 2025 update
```

- The site positions from the skd file are ok, but can be out of date (2020?)
- Kokee moves by 5cm/yr!
- We should try to keep the site positions accurate to 30mm (100psec)
- If you are onboarding a new station, **convert the ECEF xyz to lat/lon and check with google earth that the coordinates land on a telescope**

CLOCKs

Use the fmout script [1] and the station log to estimate the initial clock model:

```
$ $ python fmout.py -r 500 vo4143ow.log
vo4143ow  gps-maser                -126.074 -6.219e-13 2024y144d00h14m00s 0.004
```

Use the ‘-r’ flag to reject large outliers.
Clock rates greater than 10^-12 sec/sec may indicate a maser issue.
This leads to a clock model in VEX:

```
def Ow;
clock_early = 2024y143d00h00m :-126.074 usec : 2024y143d12h00m :-6.219e-13 ;
enddef;
```

Then add the station peculiar offset [2] to refine the clock offset:

```
def Ow;
clock_early = 2024y143d00h00m :-109.763 usec : 2024y143d12h00m :-6.219e-13 ;
enddef;
```

Clock models generated by the digital backend (DBE) to GPS offset, plus the peculiar offset, should result in residual delays less than 100nsec (roughly).

[1] https://github.com/whi-llc/fmout
[2] https://github.com/USNO-VLBI/config/blob/main/etc/pec-offset-vgos.dat

V2D file setup

The first section of the V2D file looks like this:

```
vex = vr2502.vex.obs
mjdStart = 2025y071d00h00m00s
mjdStop = 2025y072d00h00m00s
singleScan = True
exhaustiveAutocorrs = True
machines = sc03, sk21, sk22, sk23, sk24, sk26, ...and so on
nCore = 54
nThread = 7

startSeries = 1000
antennas = Gs,Hb,K2,Oe,Ow,Wf,Ws,Wn,Sa,Hv,Nn,Yg,S6,T1

SETUP prepass {
# standard 2usec search window is nChan=160, tInt=1
# wide +/-16usec SBD search window is nChan=1024, tInt=1
# nChan = 128
  nChan = 160
# nChan = 320
# nChan = 1024
  tInt=1
# subintNS = 5000000
  subintNS = 10000000 # factor of two faster at Haystack
}
```

Machine definitions will be specific to each correlator.

V2D file setup

The first section of the V2D file looks like this:

```
vex = vr2502.vex.obs
mjdStart = 2025y071d00h00m00s
mjdStop = 2025y072d00h00m00s
singleScan = True
exhaustiveAutocorrs = True
machines = sc03, sk21, sk22, sk23, sk24, sk26, ...and so on
nCore = 54
nThread = 7

startSeries = 1000
antennas = Gs,Hb,K2,Oe,Ow,Wf,Ws,Wn,Sa,H

SETUP prepass {
# standard 2usec search window is nChan
# wide +/-16usec SBD search window is n
# nChan = 128
  nChan = 160
# nChan = 320
# nChan = 1024
  tInt=1
# subintNS = 5000000
  subintNS = 10000000 # factor of two f
}
```

nChan = 160 restricts the SBD search window of +/-2usec.

To expand the SBD search window, set:

`nChan = 320 (for +/-4usec)`
`nChan = 1024 (for +/-16usec)`

...and in the fourfit control file, change the sb_win as appropriate:

```
*sb_win -2. 2.
sb_win -16. 16.
```

Machine definitions will be specific to each correlator.

V2D file setup: RULE section

When prepassing, I like to select either a single scan:

```
scan = 220-1242a
```

...or a group of scans (for example calibrator scans):

```
scan = 220-1301, 220-1305, 220-1359b, 220-1403, 220-1500, 220-1504, 220-1600,
220-1605, 220-1700, 220-1705, 220-1801, 220-1805, 220-1904, 220-2004, 220-2105,
220-2201, 220-2305, 221-0001, 221-0101, 221-0201, 221-0300, 221-0401, 221-0405,
221-0500, 221-0600, 221-0701, 221-0705a, 221-0800, 221-0804a, 221-0859b, 221-
0904, 221-1000, 221-1004, 221-1101, 221-1105
```

...or all scans from trusted sources:

```
source = 0059+581, 0202+319, 0235+164, 0458-020, 0552+398, 0814+425, 1144+402,
1156+295, 1424-418, 1741-038, 1749+096, 1751+288, 1803+784
```

Pick sources with a wide spread of RA coordinates so you span the whole session, and declinations that are visible to northern & southern stations (hard to do...).

Also, you can grep the VEX schedule to find frequently observed sources for a particular station:

```
$ grep -A 6 "source = " vr2502.vex.obs | grep -B 5 "station = K2" | grep source |
sort | uniq -c
```

Fourfit control file setup

Fourfit control files have three broad sections:

- Global parameters
- Station-specific parameters that generally don't change (tonemasks, sampler delays)
- Station specific parameters that change in every session (pc_phases_{x,y}, pc_delay_y, pc_phase_offset_y)

The global parameters look like this:

```
dr_win -5.e-6  5.e-6
sb_win -2. 2.

pc_mode multitone
pc_period 1
ion_smooth true
mbd_anchor sbd

samplers 4 abcdefgh ijklmnop qrstuvwx yzABCDEF

*** NOTE NEW REFERENCE FREQUENCY, see BEC memo 2022 July 11
ref_freq 10680.4

weak_channel 0.1
pc_amp_hcode .001
ion_npts 75
ion_win  -100.0 100.0
```

Fourfit control file setup

Fourfit control files have three broad sections:

- Global parameters
- Station-specific parameters that generally don't change (tonemasks, sampler delays)
- Station specific parameters that change in every session (pc_phases_{x,y}, pc_delay_y, pc_phase_offset_y)

The global parameters look like this:

```
dr_win -5.e-6  5.e-6
sb_win -2. 2.

pc_mode multitone
pc_period 1
ion_smooth true
mbd_anchor sbd

samplers 4 abcdefgh ijklmnop qrstuvwx yzABCDEF

*** NOTE NEW REFERENCE FREQUENCY, see BEC memo 2022 July 11
ref_freq 10680.4

weak_channel 0.1
pc_amp_hcode .001
ion_npts 75
ion_win  -100.0 100.0
```

Use the new 10.68GHz reference frequency! This improves the estimation of the formal error of the fringe phase. It will change the pc_phase_offset calculated by fourphase.

Fourfit control file setup

Fourfit control files have three broad sections:

- Global parameters
- Station-specific parameters that generally don't change (tonemasks, sampler delays)
- Station specific parameters that change in every session (pc_phases_{x,y}, pc_delay_y, pc_phase_offset_y)

The global parameters look like this:

```
dr_win -5.e-6  5.e-6
sb_win -2. 2.

pc_mode multitone
pc_period 1
ion_smooth true
mbd_anchor sbd

samplers 4 abcdefgh ijklmnop qrstuvwx yzABCDEF

*** NOTE NEW REFERENCE FREQUENCY, see BEC memo 2022 July 11
ref_freq 10680.4

weak_channel 0.1
pc_amp_hcode .001
ion_npts 75
ion_win  -100.0 100.0
```

These are the parameters for the ionospheric (dTEC) search. A search range of +/-100 TECU is good enough for most scans/baselines.
75 search points is the max number allowed in HOPS3.

Fourfit control file setup

Remember, global parameters in the control file must be set before any "if" statements.

sampler_delay_{x,y} **must** be defined in the control file for each station, or else the phasecal delays are not applied. If in doubt, set to zero:

```
if station Z
  sampler_delay_x    0    0    0    0
  sampler_delay_y    0    0    0    0
```

The pc_tonemasks identify bad phasecal tones (due to RFI or other effects). Each pc_tonemask identifies channels and a binary mask (in decimal notation) for each channel. **ToDo**: write a script to modify the pc_tonemask using phasecal data.

```
if station G
  pc_tonemask cf    2 16
```

Finally, copy the pc_phases_{x,y} for each station from a recent session. These are your apriori pc_phases; they will be updated by the ffres2pcp script later in the process.

ToDo: Track sampler delays and pc_phases the same way we track peculiar offsets. Correlators should use be using the same values in their initial control files, to prevent biasing dUT1 and dTEC.

Now you are ready to correlate!

Prepass 1:

Choose one or two scans to correlate first, to check for errors in the setup, verify fringes, check that phasecal tones are present, tune clocks, etc.

Choose scans with trusted sources and good network coverage (calibrator scans are good). Short baselines are best!

Visually inspect the fringe plots to verify all aspects of station health:

- Is there any data loss? Are there any missing bands or polarizations?
- Check that each polarization is different, eg XX is different than XY?
- Are the phasecal tones present?
- Do all stations have a fringe detection?
- Are the SBD residuals less than 100nsec?

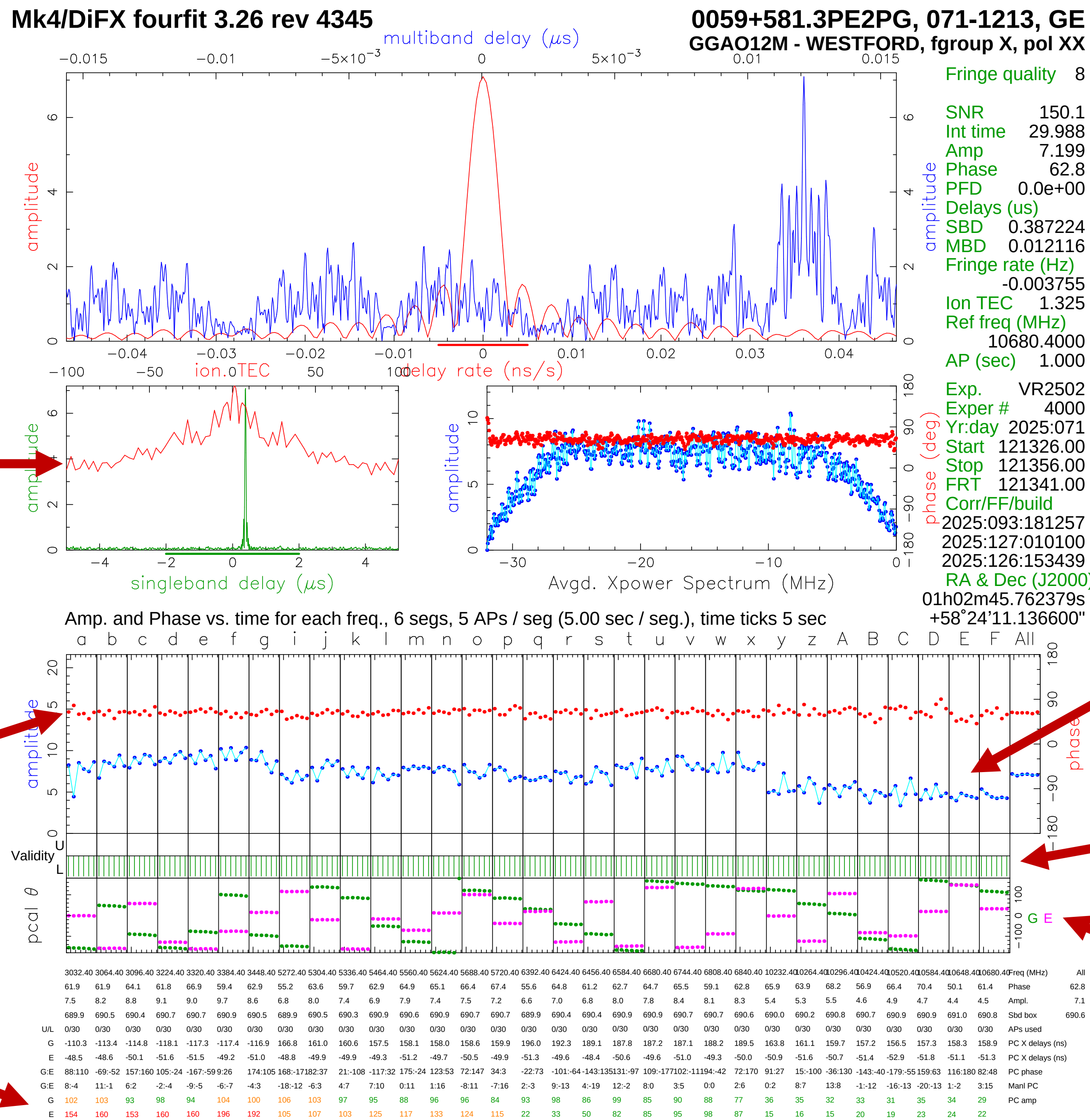
Prepass 1: fringe plot checks

A strong fringe detection!

Clear peaks in SBD and dTEC search

Residual phases already quite flat

Phasecal tone amplitudes are strong at both stations



Integration time matches expectation, no data loss

The SBD residual is a bit large; correct this with the VEX clock models

Good amplitude in all bands & channels

Green validity lines indicate no data loss

Phasecal phases in each channel are steady across scan (no jumps or drifts)

Prepass 1: fringe checks

What if you can't find a fringe?

- Try with another station that has confirmed fringe detections
- Try another scan; it might be a bad source
- Check autocorrelations; are phasecal tones visible in the Xpow spectrum?
- Do the tones have the right frequency?
- If yes, you have set up the FREQ definition properly, and the data files are being correlated properly.
- Check the station log for a near-in-time dbe_gps_offset value to guide clock search
- Widen the SBD search window; you can check +/-16usec (32 usec total) at a time.
- Search around very large offsets (+/-1sec) in case of DBE misalignment.

Autocorrelations can be very informative, especially for new stations. Check each channel individually from the command line with the "set" option to fourfit:

```
$ fourfit –pt –c cf_3862_initial –b EE –P XX 071-0000a/0025+197.3PLXVD set freqs a
```

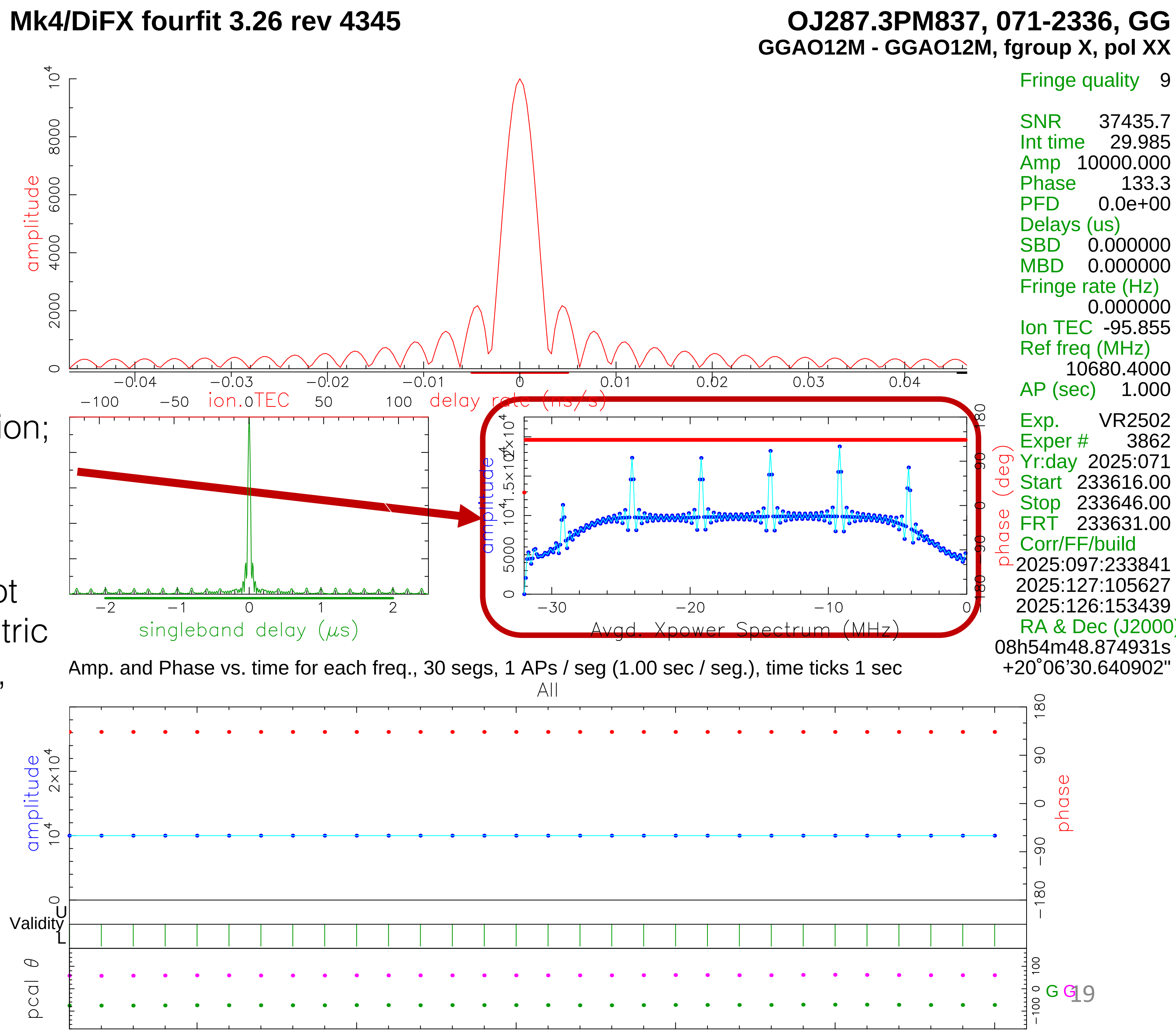
...and then b, c, d, etc.

Stations can make autocorrelation checks before they send the data to the correlator!

Autocorrelation example: good

This is a 5MHz station; the phasecal tones are clear and loud

The bandpass is not flat, but it is symmetric across the channel, flat in the middle.

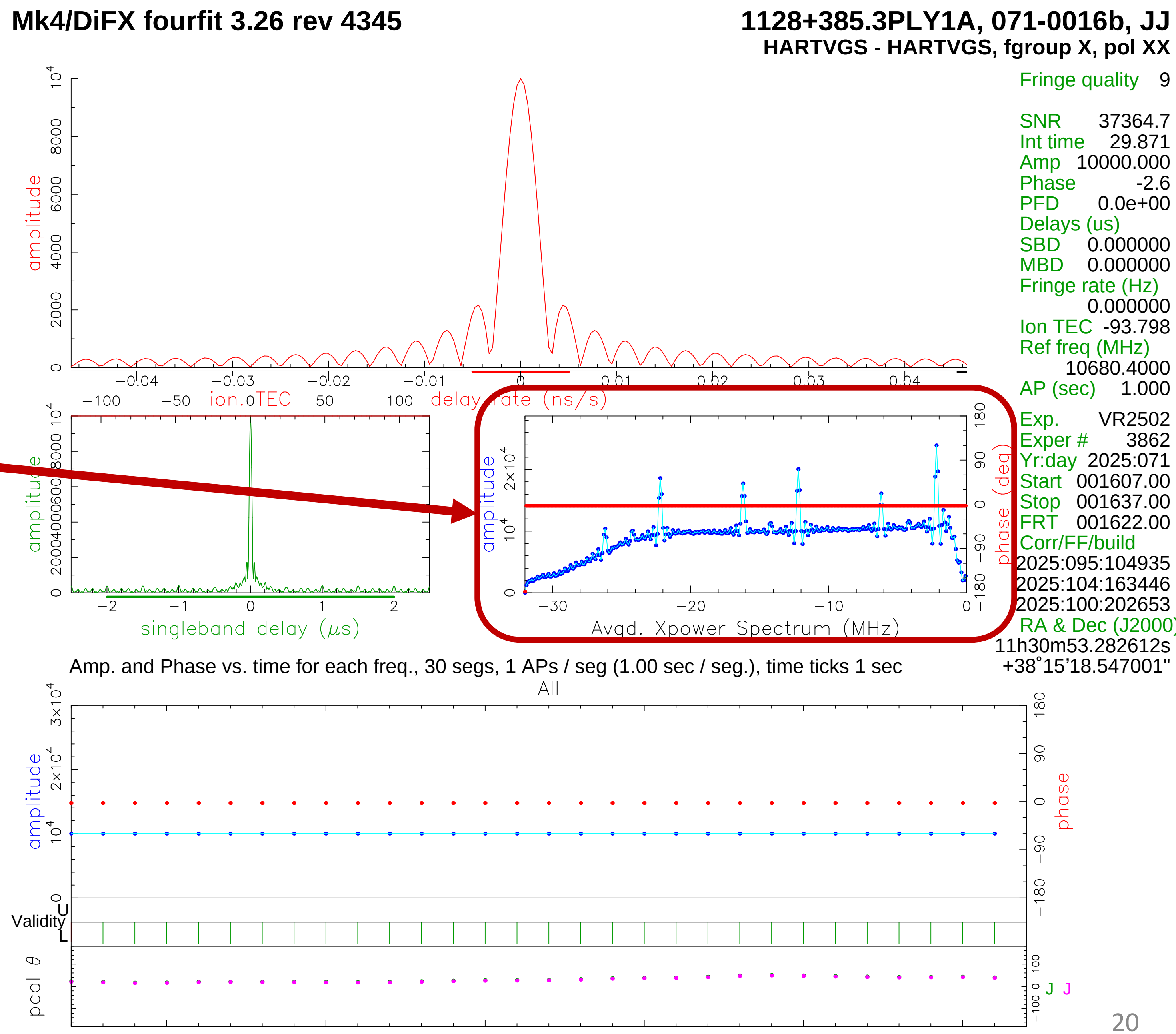


Autocorrelation example: not so good; spurious tones

This is a 10MHz station; note the spurious tones

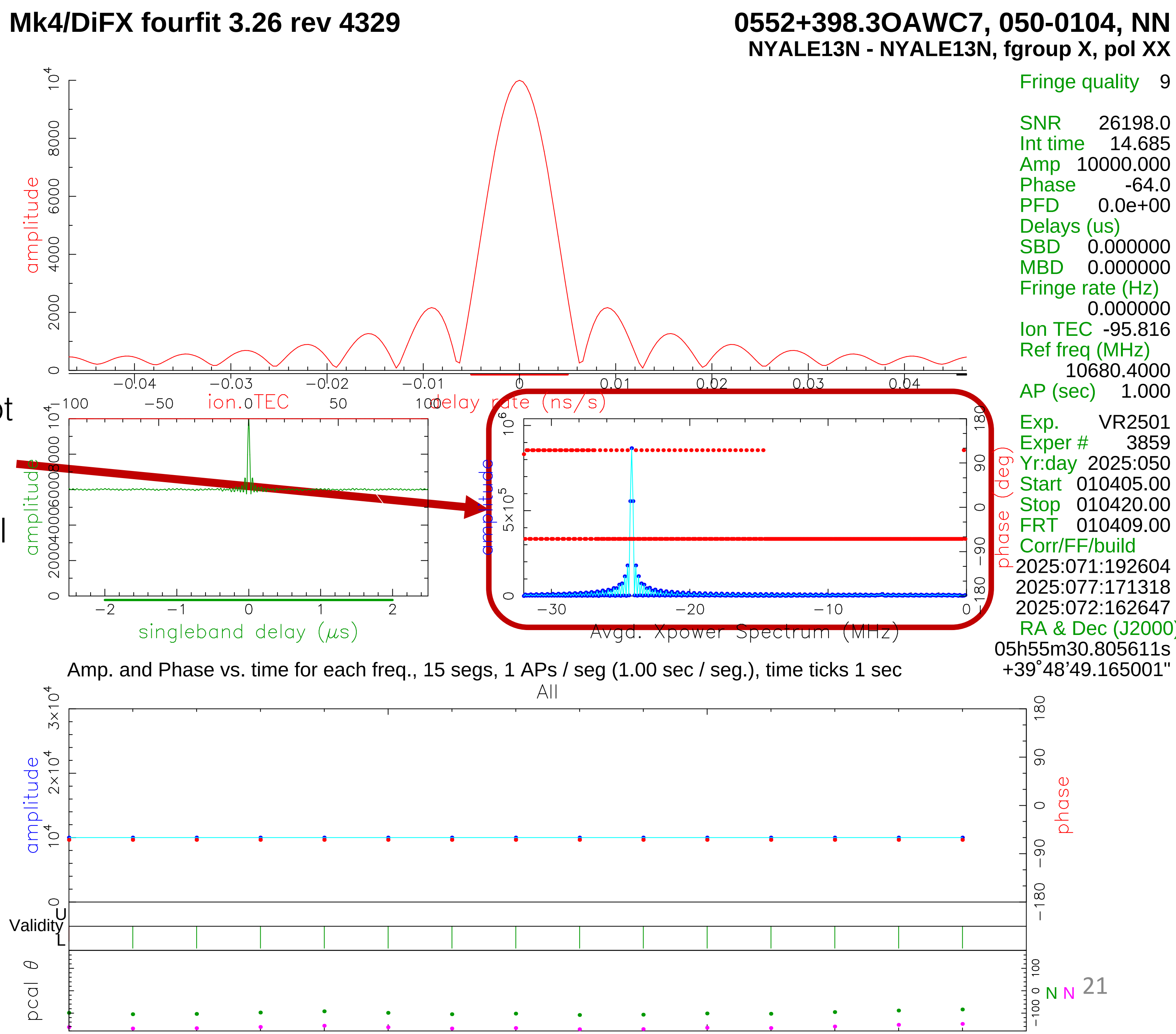
There are 10MHz tones in the right locations; for this channel, zero on the xaxis in the Xpow spectrum is 5272.4MHz, and there is a peak at each 10MHz step.

You can use a notch filter in the control file to confirm the peak frequency; this is a good check of the channel FREQ def in the VEX file.



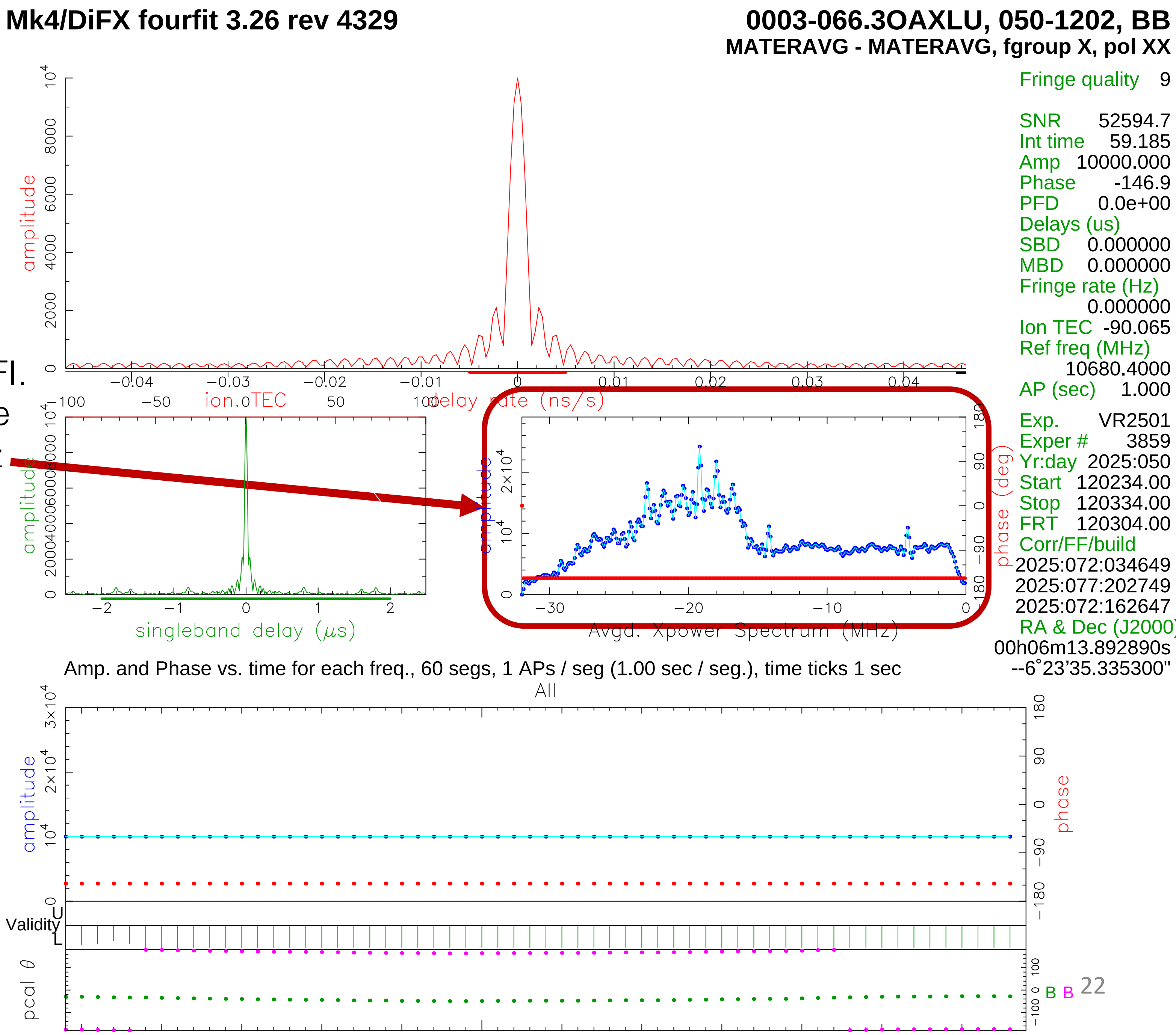
Autocorrelation example: not so good: a single loud tone

Only one peak, not at the right frequency, 100x louder than typical phasecal tones.



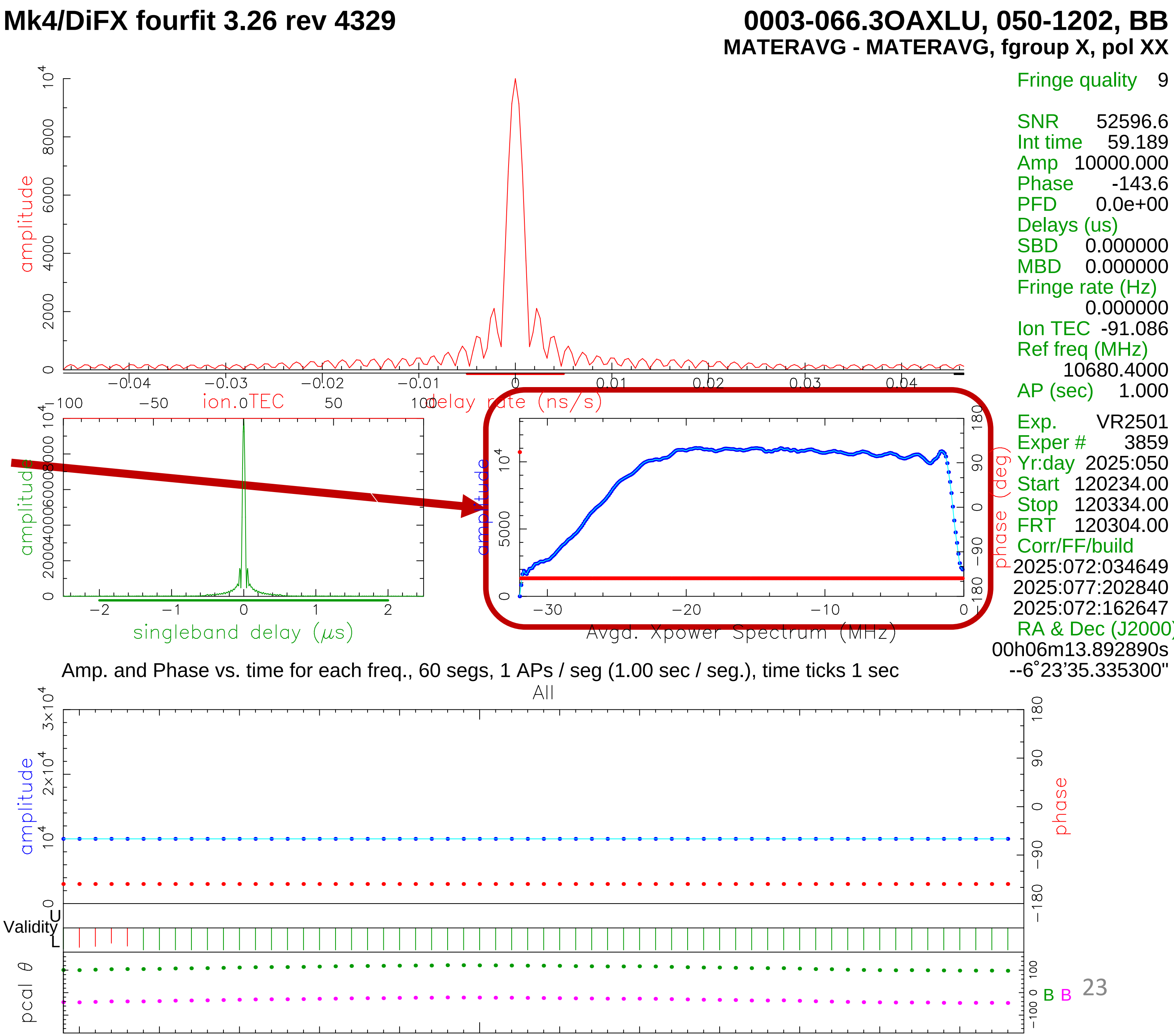
Autocorrelation example: not so good: malformed bandpass

Could be due to RFI. Two tones might be present at -4.4MHz and -14.4MHz.



Autocorrelation example: not so good: no tones visible

This is channel a – the pcal tones here should be loud, but for this station they are missing!



Sampler delays – see VGOS memo #50

Phasecal tones are necessary to align the phase across channels and across time.

But the phasecal tone spacing introduces an ambiguity:

- stations with 5MHz phasecal have a 200nsec ambiguity (1/5MHz=200nsec)
- stations with 10MHz phasecal have a 100nsec ambiguity (1/10MHz=100nsec)

We correct for this ambiguity using the "sampler_delay_{x,y}" values in the fourfit control file. Fourfit will calculate the optimal phasecal delay ("PC delay") from the phase of each tone, and then add the appropriate number of 200ns or 100ns ambiguities as specified by the sampler delays.

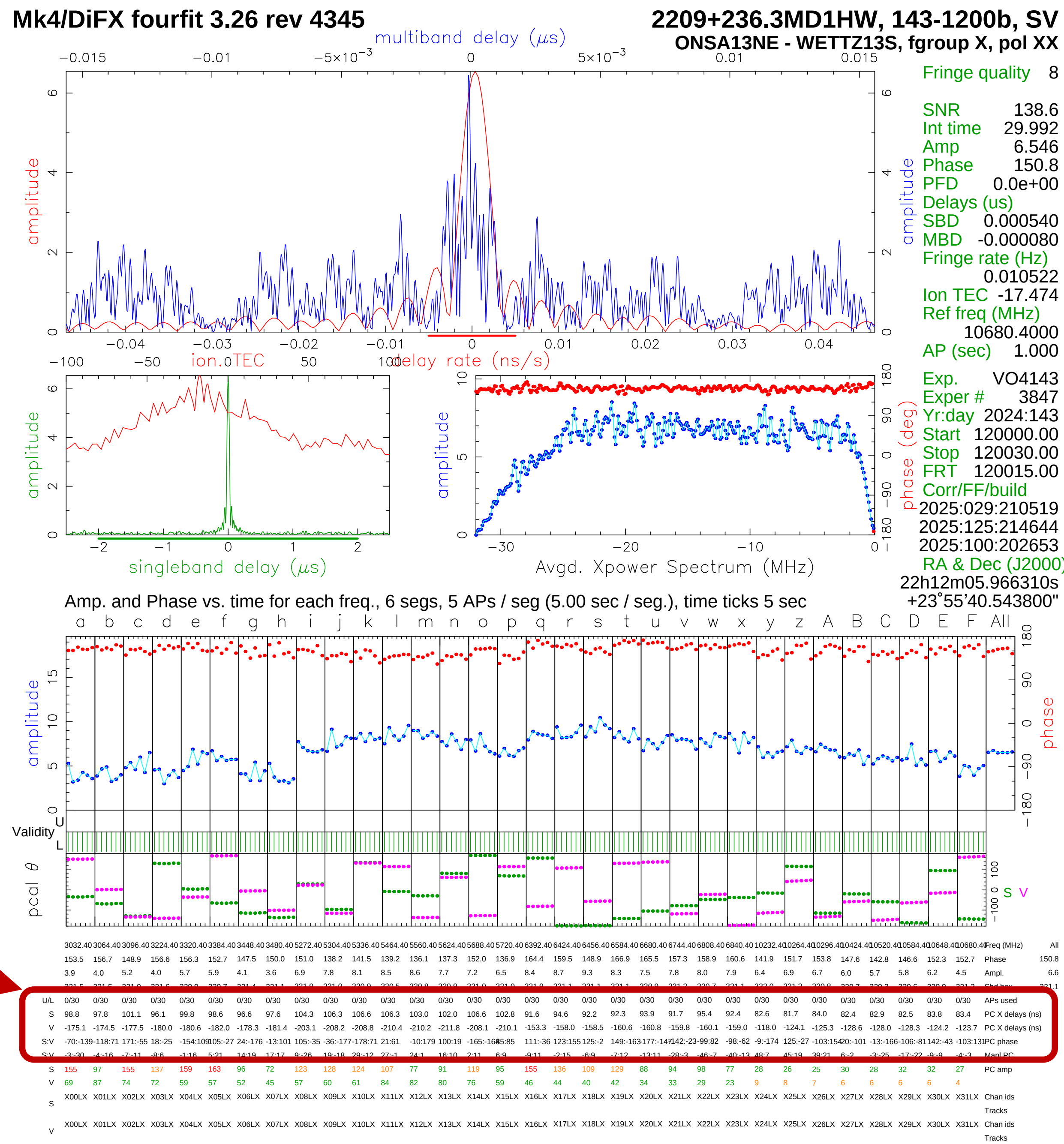
The sampler_delay values should be close to zero, and the residual SBD offset should be absorbed into the station clock model in the VEX file.

For most stations, all four bands should be within the same ambiguity. But NASA stations (GGAO, MGO, Kokee) will have a larger delay in bandA. For example Kokee (H):

```
if station H
  sampler_delay_x   -210  75  75  75
  sampler_delay_y   -210  75  75  75
```

Ideally, the sampler_delay values should be set to the average of the optimal PC delay for the eight channels in each band. Unfortunately the PC delays are not stored in the type200 fringe files...and on the fringe plot they are in very tiny font...

Sampler delays



If a fringe is detected, the PC delays on the fringe plot are the optimal delay values.

Average the eight channels in each band to get the sampler_delay values for the control file.

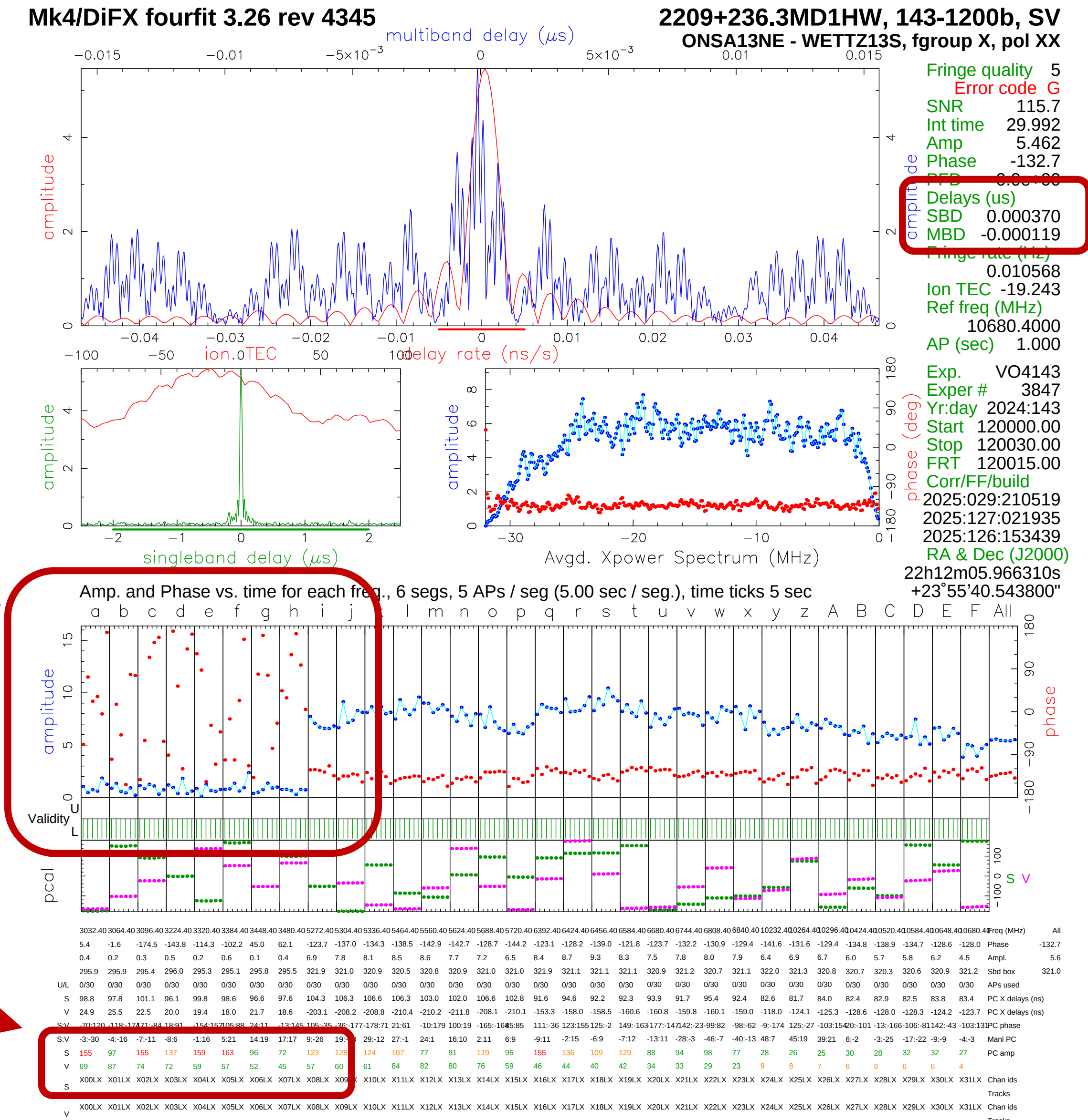
ToDo: modify Frederic's ps2samplers2.sh script to check for channel disagreement.

Sampler delays

Identifying when single band has an incorrect sampler delay is straightforward...

No detection across an entire band...if we fringe fit only this band, we find the SBD residual is different from the other bands by 100nsec.

...but phasecal tone amplitude is ok, so the signal chain is working



The three bands with detections agree on a small SBD residual

Sampler delays

If one band has no fringe detection, the PC delays on the fringe plot for that band will be incorrect; you can't use them to set sampler_delay_{x,y}. You will have to tune the sampler delays for that band by hand.

Fringe fit just the one band:

```
$ fourfit –pt –c cf –b SV –P XX 143-1200b/2209+236.3MD1HW set freqs a b c d e f g h
```

...and change the sampler_delay in the control file until the SBD residual matches the fringe for the other three bands. Explore in large steps, equal to half the ambiguity for that station (eg +/-50nsec).

Once you find the correct SBD residual, the PC delays on the fringe plot will be informative. Average them together to get the optimal sampler_delay value.

Setting the sampler_delay in the control file to be the optimal PC delays protects from small variations in delay in subsequent scans; it provides some room for error.

If two bands need to have their sampler delays adjusted, you need to pick which two bands are correct. Choose the bands with the smaller SBD residual, to avoid biasing dUT1.

Prepass 2: tune clock models to minimize SBD residuals

Once we have verified that:

- Fringes have been detected to all stations
- All bands/polarizations have detections (or, you understand why they don't)

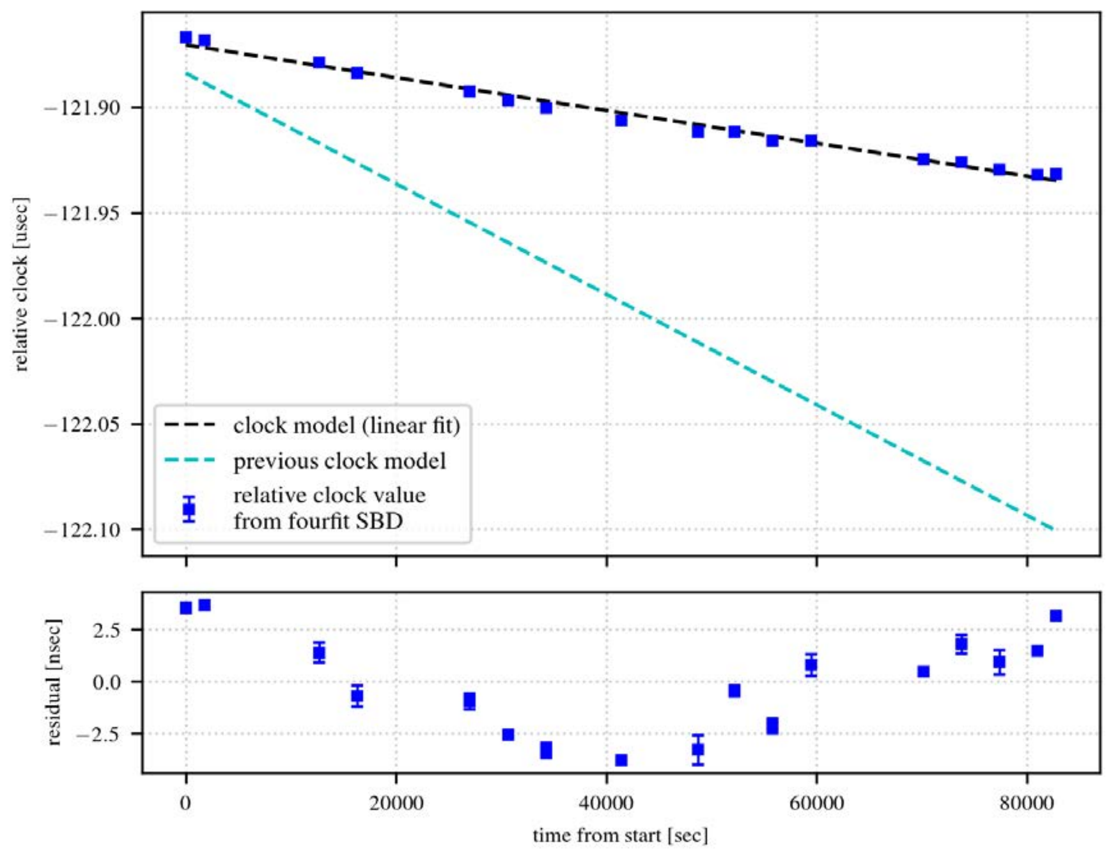
...we can proceed to fine-tuning the station clock models.

Now we can correlate a larger number of scans across the session; I like to use the calibrator scans, which are usually placed once per hour, and have large sub-nets that cover the entire network.

After fringe fitting, I have a script that collects the SBD residuals, reads the clock models from the VEX file, and fits the clock models to remove the SBD residual (relative to a reference station). You can also do this with aedit or other programs, or by hand.

Now that the VGOS network is global, you might have to do this twice, using a different reference station in both hemispheres.

The goal is to reduce the SBD residuals on all baselines to <15nsec.

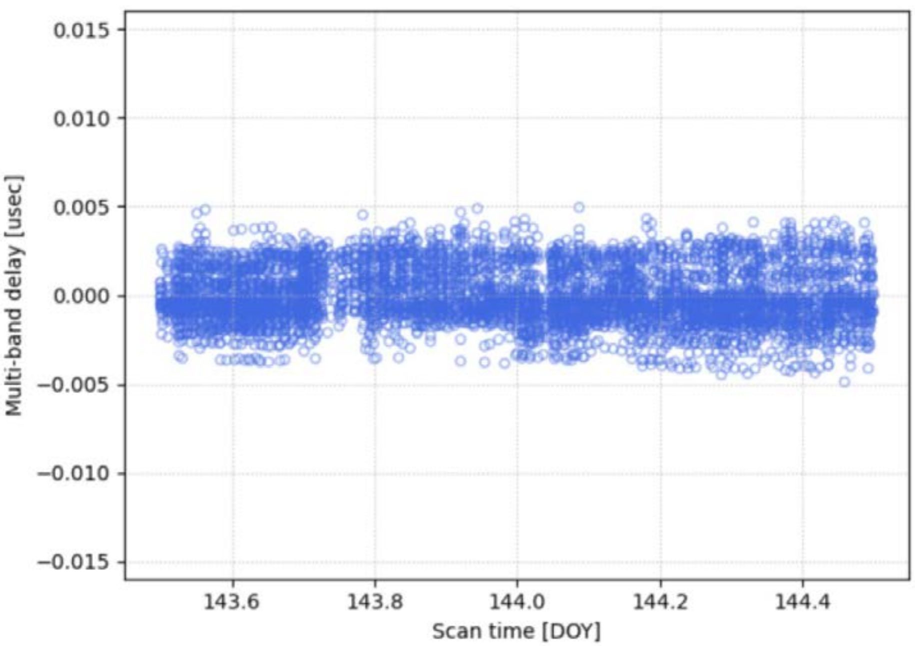


Prepass 3: tune clock models to minimize MBD residuals

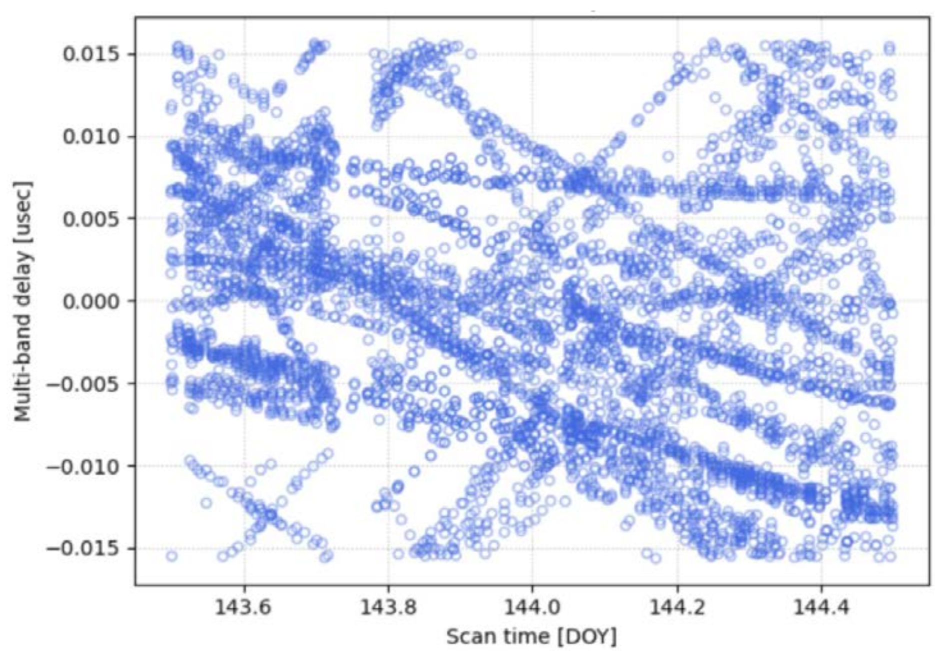
The observable we are interested in is the group delay, or multiband delay (MBD).

It's not absolutely necessary to tune the clock models to make the MBD residual as close to zero as possible, but it is desirable:

- MBD residuals are a good diagnostic for station performance and session noise
- MBD residuals are used later to estimate pc_delay_y, it helps to have them minimized
- The MBD search window is +/-15.6nsec; residuals will wrap at the boundaries
- **ToDo**: provide a script for clock tuning



Group delay residuals after tuning each clock model. Residual MBD offsets and rates are close to zero.



Group delay residuals without tuning the clock models. Some baselines have significant offsets and residual rate.

Production correlation

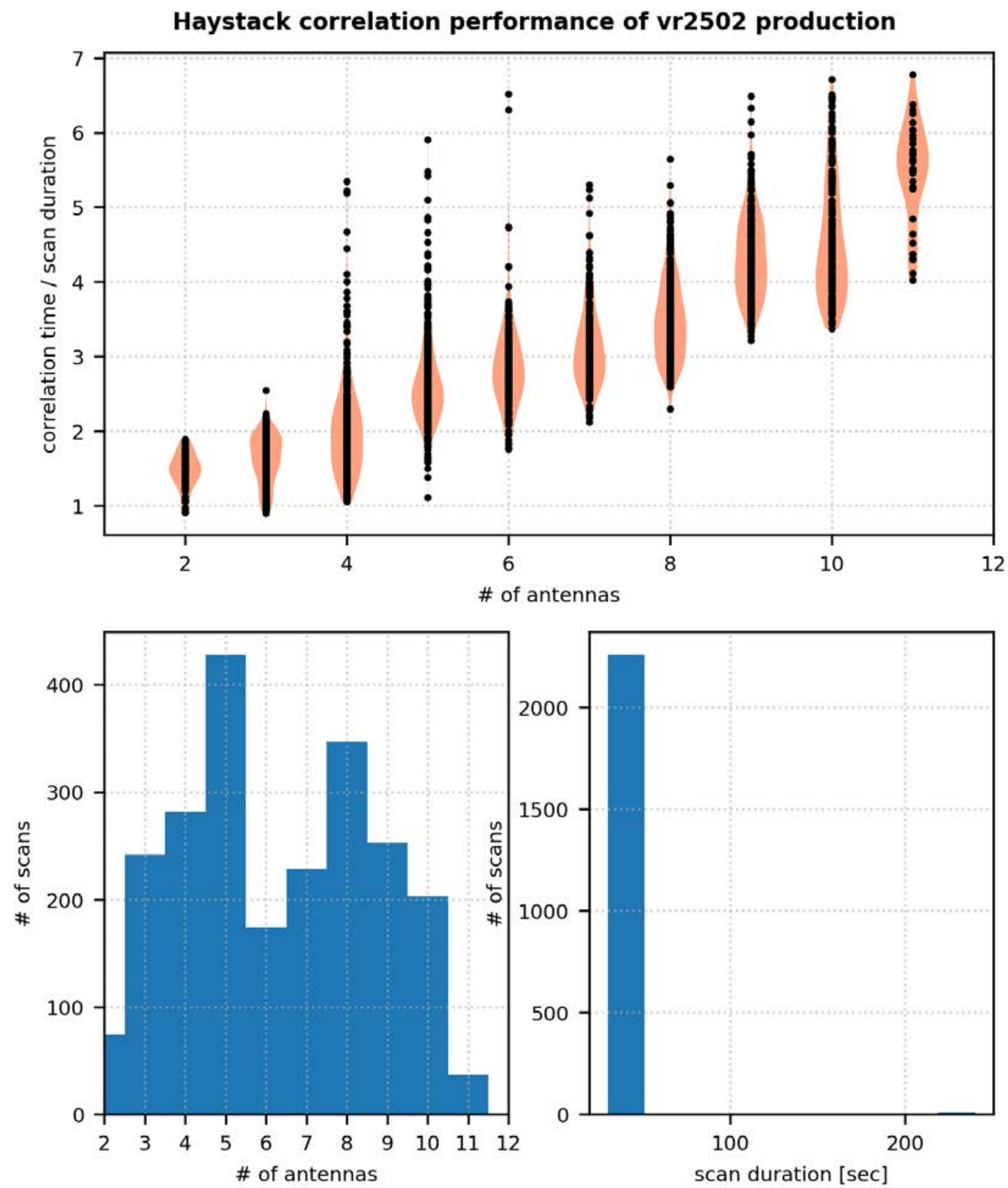
Once the clock models are tuned so the MBD residuals are minimized, we are ready to correlate the full session.

For a 14-station, 24hr session, correlation at Haystack takes about 59 hours (~2300 scans, scan duration is 30sec, average job time is 90sec.)

There is large variation in job time due to stations involved, I/O bottlenecks. Data recorded to flexbuff requires 4x as many processes on the Mk6 units.

Using DiFX 2.5.4, I find it's necessary to keep a 'difxwatch' process running to kill zombie jobs. Set the timeout long enough that it doesn't kill the calibrator scan jobs!

'grep' the difxlog files for SEVERE errors, check for a wallclock time (indicates successful completion).

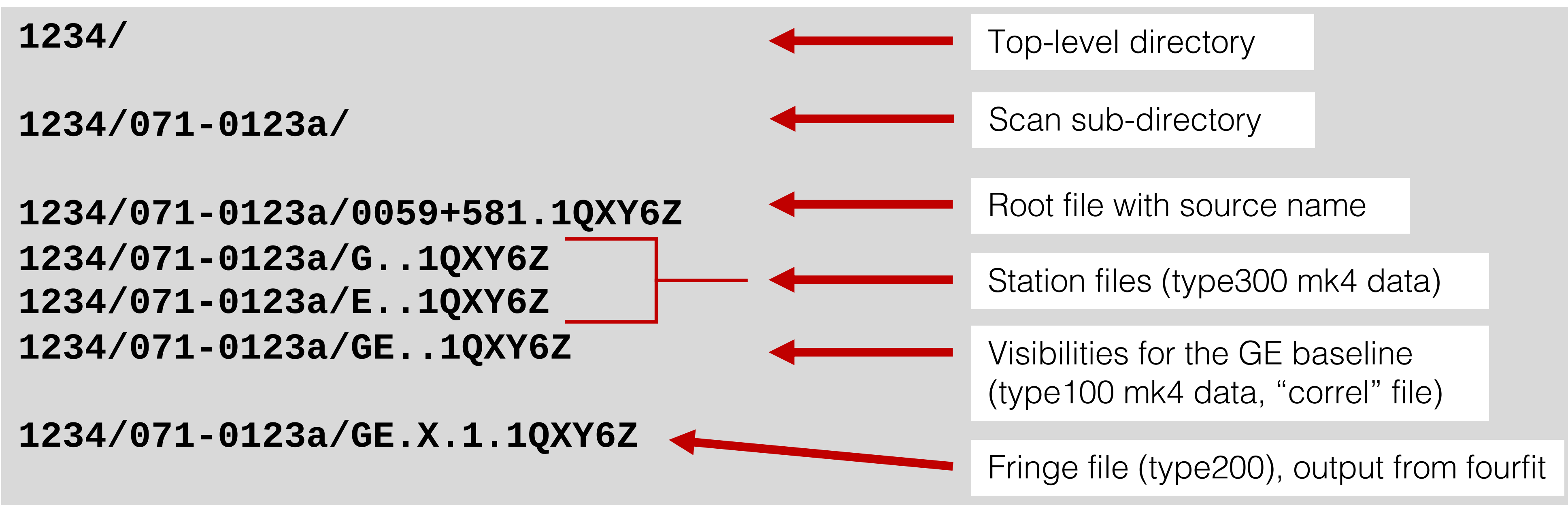


VGOS post-processing: difx2mark4 and file arrangement

Once correlation has finished, use the difx2mark4 utility to convert the DiFX files (“SWIN” format) into the ”mark4” format used by fourfit. It requires a “codes” file that translates the 2-letter station codes from DiFX to the 1-letter station codes used by fourfit.

Don’t use the ‘-d’ option in difx2mark4! I use a bash script to difx2mark4 each scan individually.

fourfit requires the mk4 data to be arranged in a directory with a four-digit name. Data for each scan is in a sub-directory with the scan name. These sub-directories have a “root file” with the source name, station files with the phasecal data, and correlation files that have the visibilities calculated by DiFX.



VGOS post-processing: setup for fringe fitting, fetch IONEX predictions

Put the initial control file into the top-level directory. This control file has the updated sampler delays from the prepasses (if they had to be updated) and the apriori pc_phases_{x,y} from a recent session.

```
1234/                                  ← Top-level directory

1234/cf_initial
```

Run the get_ionex script to fetch IONEX predictions for the dTEC value for each scan/baseline:

```
$ get_ionex_dtec_bounds.py -a jpl /path/to/codes/files
           /path/to/calc/files /path/for/ionex/files output_filename
```

For example:

```
$ get_ionex_dtec_bounds.py -a jpl /data/vr2502/production/codes
       /data/vr2502/production /home/tmp/ionex vr2502_ionex_jpl.json
```

The json file has the IONEX prediction for the line-of-sight TEC for each station in each scan. The difference between the station TEC is the dTEC prediction.

ToDo: provide a script to plot the dTEC prediction; it's useful to know which scans will have large dTEC outliers.

VGOS post-processing: ffres2pcp

Now in our top-level directory we have the scan sub-directories, the initial control file, and the file with the IONEX predictions.

```
1234/

1234/cf_initial
1234/vr2502_ionex_jpl.json
1234/071-0123a/
...and so on.
```

Now we are ready to run ffres2pcp, which will fine-tune the pc_phase_{x,y}.

ffres2pcp will fringe-fit every baseline from the remote stations to a reference station. It will generate a ‘scratch’ sub-directory named with a timestamp, make symbolic links to the files in the top-level directory, and put fringe files for all four pol-products there (XX,XY,YX,YY).

```
$ ffres2pcp.py cf_initial H AEMNSTV . –n 100 –p –s 15 –d 1 –q 5 –a 30

Creates:
/1234/scratch/YYYYDDMM-HHMMSS/1234/
/1234/scratch/YYYYDDMM-HHMMSS/1234/071-0123a/
/1234/scratch/YYYYDDMM-HHMMSS/1234/071-0123a/HE.X.1.1QXY6Z
/1234/scratch/YYYYDDMM-HHMMSS/1234/071-0123a/HE.X.2.1QXY6Z
...and so on.
```

VGOS post-processing: ffres2pcp

I like to rename the scratch folder to something meaningful that identifies the iteration of ffres2pcp:

```
$ mv scratch/YYYYDDMM-HHMMSS scratch/ffres1
```

ffres2pcp will generate a log file in the top-level directory, a report file in the scratch directory, and a new control file with the updated pc_phase_{x,y} in the scratch directory.

```
/1234/ffres2pcp-Apr-08-0313PM-2025.log
/1234/scratch/ffres1/1234/ffres2pcp-report-Apr-08-0313PM-2025.json
/1234/scratch/ffres1/1234/cf_pcphases
```

Generate diagnostic plots by running summarize_report.py on the report file:

```
$ summarize_report.py ffres2pcp-report-Apr-08-0313PM-2025.json
```

There are two kinds of diagnostic plots:
 - plots that compare the old (apriori) pc_phases with the new (estimated) pc_phases
 - plots that show the minimum SNR and the maximum dTEC deviation (ddTEC) of the scans used to estimate the new pc_phases

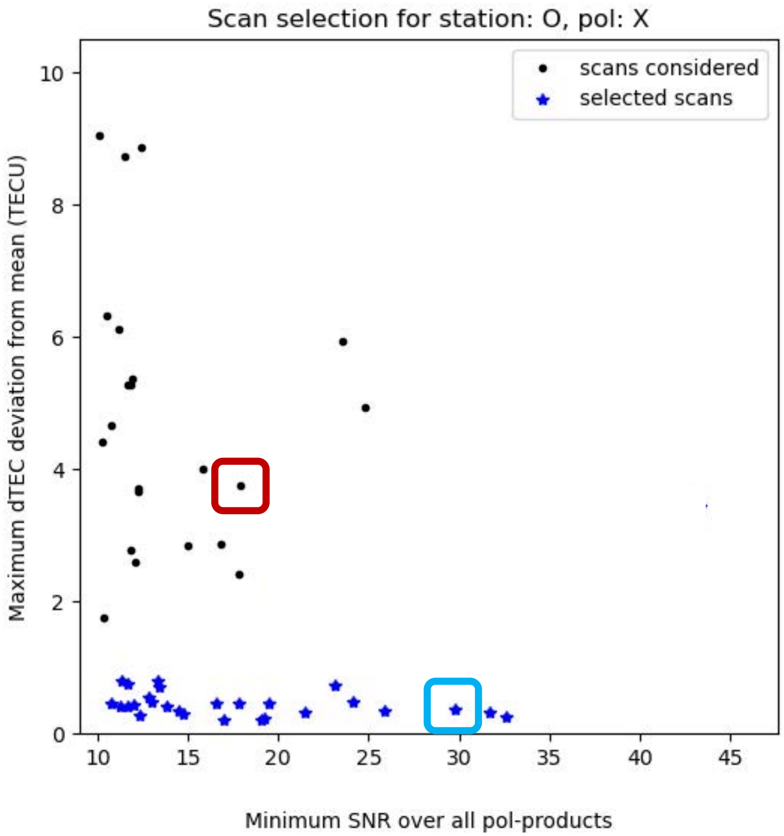
VGOS post-processing: ffres2pcp

It's important to require ffres2pcp to use good scans to estimate the pc_phases

- Low-snr scans will just inject noise into the updated phases
- Scans with large disagreement in the dTEC between the pol-products (XX,XY,YX,YY) will introduce a dTEC bias into the pc_phases. Since dTEC and MBD are highly correlated, this will bias the group delay.

Some example scans:

	071-0123a		071-1008b	
	dTEC	SNR	dTEC	SNR
XX	-28.6	29.8	-19.8	17.9
XY	-28.2	30.4	-19.6	24.0
YX	-28.8	29.9	-19.7	24.7
YY	-28.7	37.8	-14.7	21.9
mean dTEC	-28.6		-18.5	
max ddTEC	0.4		3.8	
min_snr		29.8		17.9
	accepted		rejected	



VGOS post-processing: ffres2pcp

It's important to require ffres2pcp to use good scans to estimate the pc_phases

- Low-snr scans will just inject noise into the updated phases
- Scans with large disagreement in the dTEC between the pol-products (XX,XY,YX,YY) will introduce a dTEC bias into the pc_phases. Since dTEC and MBD are highly correlated, this will bias the group delay.

These parameters are set on the command line for ffres2pcp:

```
$ ffres2pcp.py cf_initial H AEMNSTV . –n 100 –p –s 15 –d 1 –q 5 –a 30
```

Minimum allowed SNR among the pol-products for each scan/baseline (`-s 15`)

Maximum allowed difference in dTEC among the pol-products (`-d 1`)

It's important to keep the '-s' parameter high; 15 or 20 is preferred.
It's important that '-d' be 1.

But for some stations, it's hard to find high-SNR scans in all the pol-products.
And the dTEC disagreement can be large.

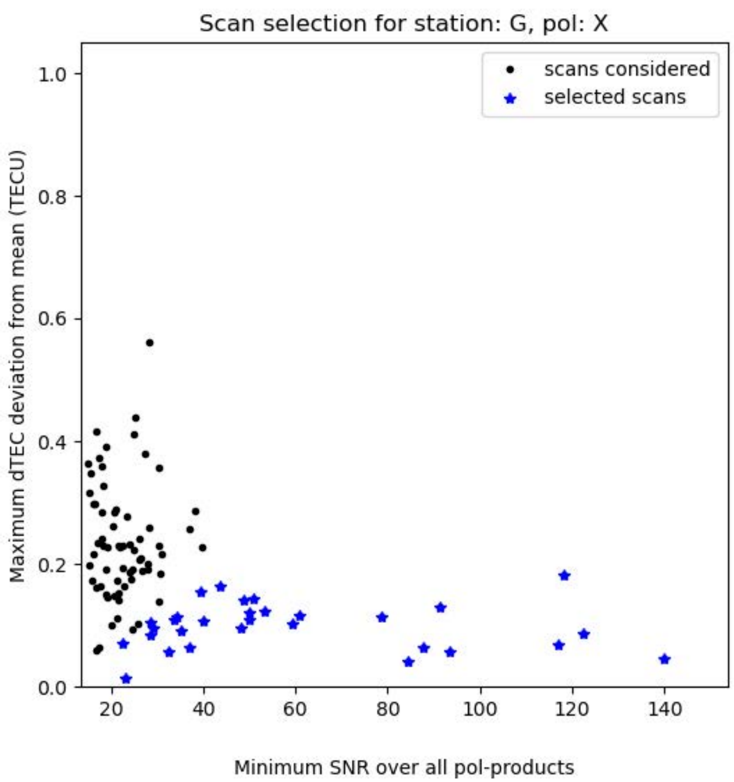
VGOS post-processing: ffres2pcp

The scan_selection plots from ffres2pcp show how many scans were used to calculate the new pcphases.

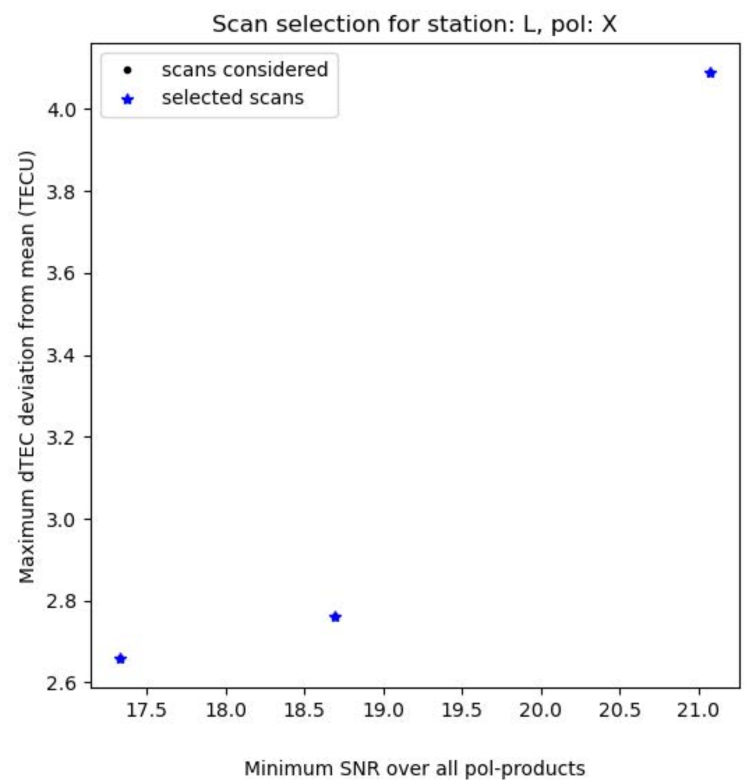
The range of the plots is set by the command-line parameters to ffres2pcp; only scans with ddTEC less than the '-d' limit and min_snr greater than the '-s' limit are plotted.

If there aren't enough scans meeting these thresholds, the plots can look very sparse!

Good selection of scans

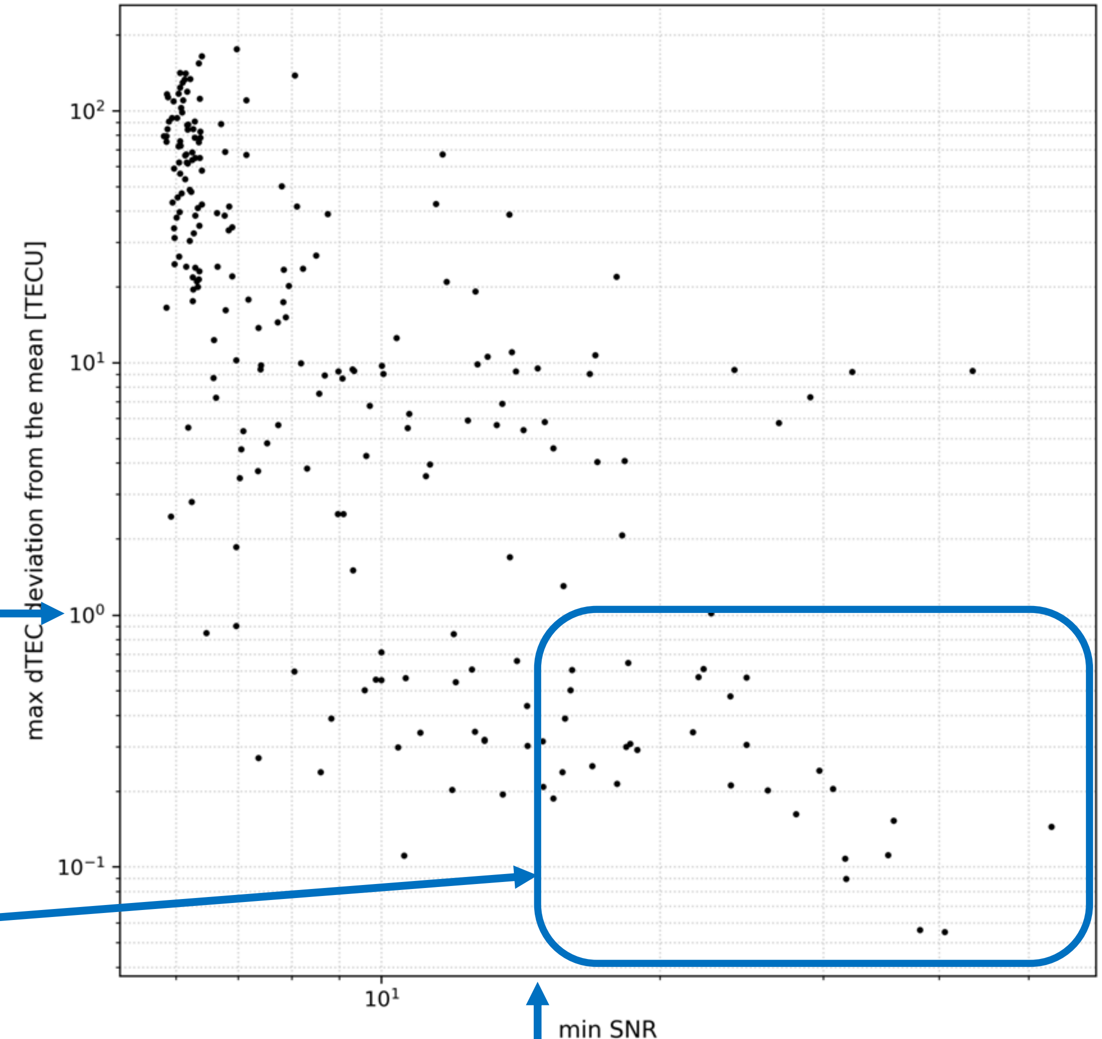


Not enough scans



VGOS post-processing: ffres2pcp

Use the plot_ddTEC_minsnr.py script (new to HOPS 3.26) to plot the entire range of the ddTEC – min_snr parameter space:



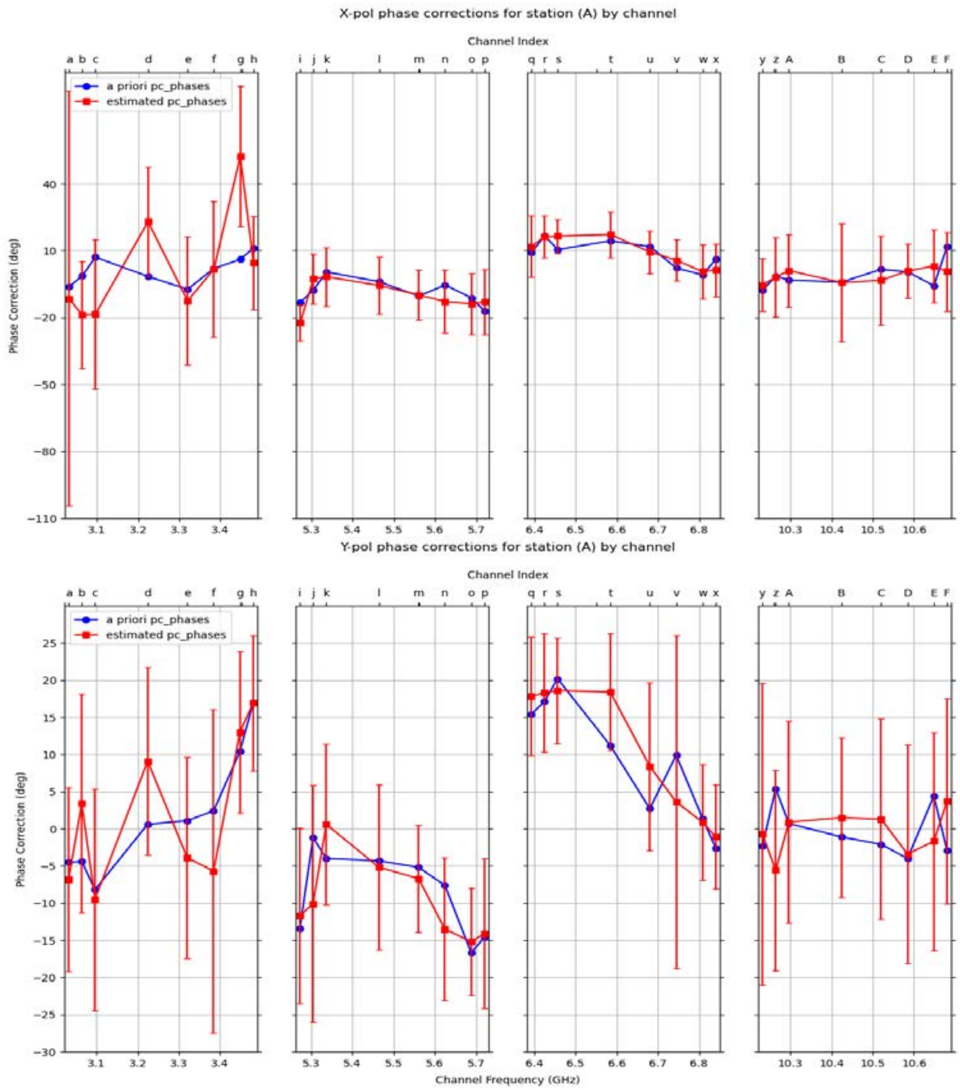
max ddTEC = 1

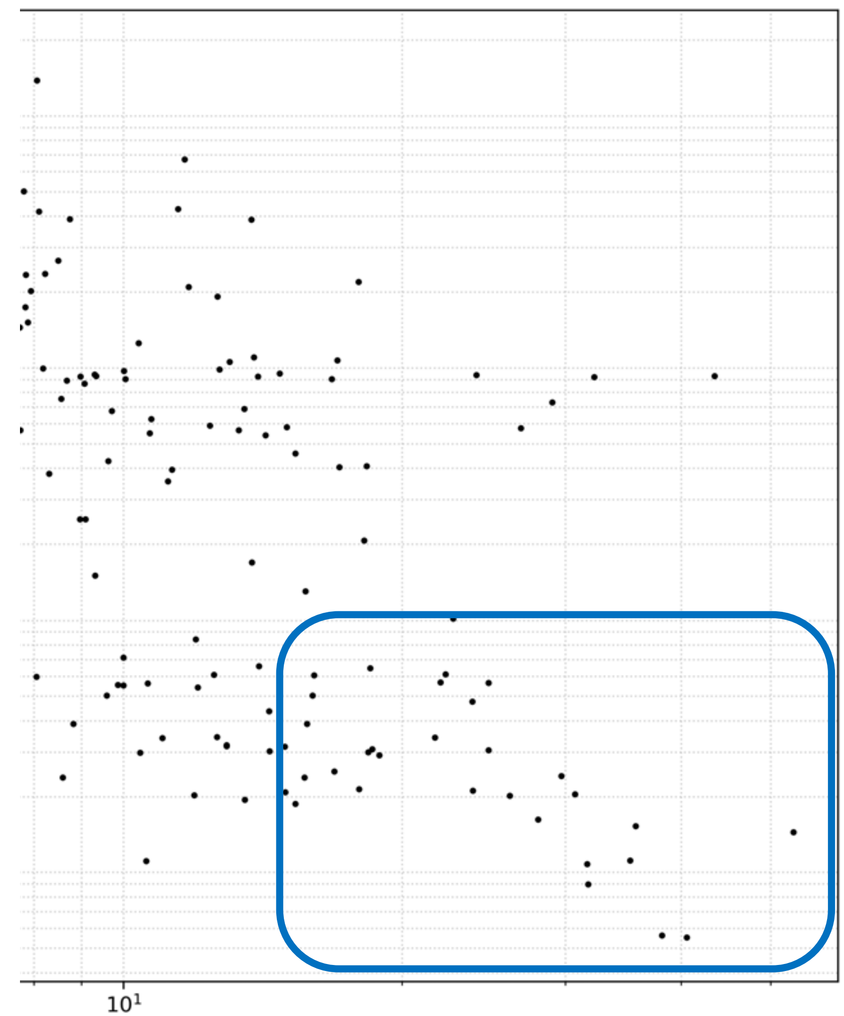
Only these scans would be used to calculate the new pc_phases:
ddTEC<1, min_snr>15

min_snr = 15

VGOS post-processing: ffres2pcp

In this example, the number of scans is good (26 scans were accepted, after additional cuts), and the pc_phase comparison plots are sensible. Some bad channels.





Error bars of +/-20deg on these figures are not unusual. **ToDo**: study the distribution of the parameter estimation; is it gaussian or is it bimodal?

VGOS post-processing: ffres2pcp

What if, for a particular station, there are no scans meeting the ddTEC<1, min_snr>15 criteria?

First, check the full ddTEC-min_snr parameter space with plot_ddTEC_min_snr.py

Are there no scans with high SNR in all pol-products? Maybe choose a different reference station. Kokee is a good choice because it has visibility almost everywhere (but not to HartVGS!). MGO, GGAO, the Onsalas are other good options. The GGAO-Westford baseline has almost no SNR in the cross-hands; the same is true for NyAlesund-Onsala.

Are there few or no scans with low ddTEC? Maybe there is a fixed offset between the polarizations of this station.

If there are just one or two scans, you can iterate: run ffres2pcp with '-d 1', run it again with the updated pc_phases. The number of scans with ddTEC<1 usually improves; the updated pc_phases have corrected the dTEC bias between the polarizations.

Another trick: to make the scan_selection plots show a larger parameter space, first run ffres2pcp with an expanded '-d' and '-s' thresholds:

```
$ ffres2pcp.py cf_initial H AEMNSTV . –n 100 –p –s 10 –d 10 –q 5 –a 30
```

The plots generated from the report will show scans with ddTEC<10, min_snr>10.
Once you have inspected the scans available, rerun ffres2pcp *in the scratch directory* with the '-w' option and '-d 1'. This will re-use the fringe results (remember to use the same control file).

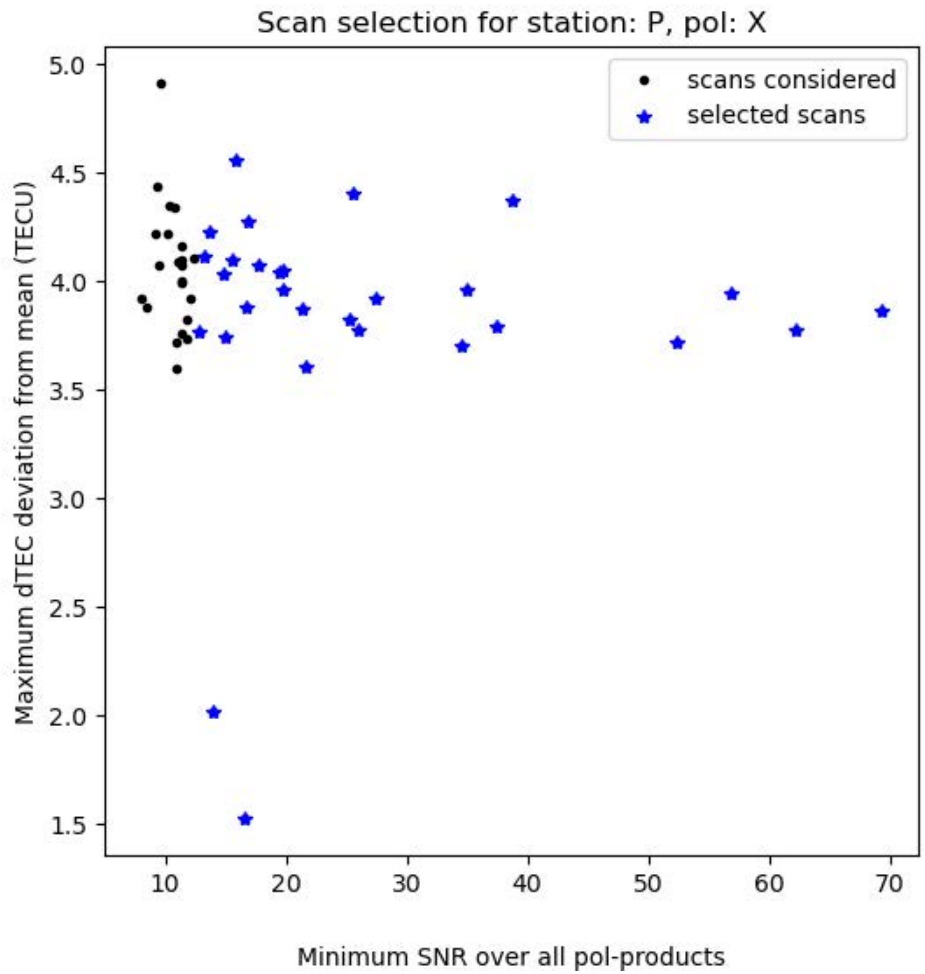
```
$ cd scratch/ffres1/1234/
$ ffres2pcp.py cf_initial H AEMNSTV . –n 100 –p –s 15 –d 1 –q 5 –a 30 -w
```

VGOS post-processing: ffres2pcp

In very bad cases, there may be a large dTEC disagreement between the polarizations.

For example, if scans with good min_snr (>20) have ddTEC>1, this means fourfit is finding a constant dTEC offset between the X-pol and Y-pol results. This can happen if the initial pc_phases have a dTEC bias.

Example: Katherine from VO3278



This is bad:

- The apriori pc_phases are applying a dTEC bias between the polarizations
- Updating the pc_phases will only reinforce this bias
- Downstream products like the y-x delay and the Ixy results will have this bias baked-in

You must fix this problem before moving forward!

VGOS post-processing: ffres2pcp

The solution (following Arthur's VGOS memo #56 for co-located antennas) is to force one polarization to agree with the dTEC of the other polarization. The steps are:

1) Pick the polarization whose dTEC result you trust. If the station just performed some maintenance on one polarization, maybe that one is the problem. Otherwise, flip a coin...here I have chosen to trust the YY dTEC result and change the Katherine pc_phases_x.

1) Pick a scan with high SNR on a baseline between the problem station (here, P, Katherine) and the reference station (here, Kokee, H).

2) In the control file, comment out the pc_phases for the problem station for the problem polarization (here, pc_phase_x). Keep the (just updated!) pc_phases for the reference station.

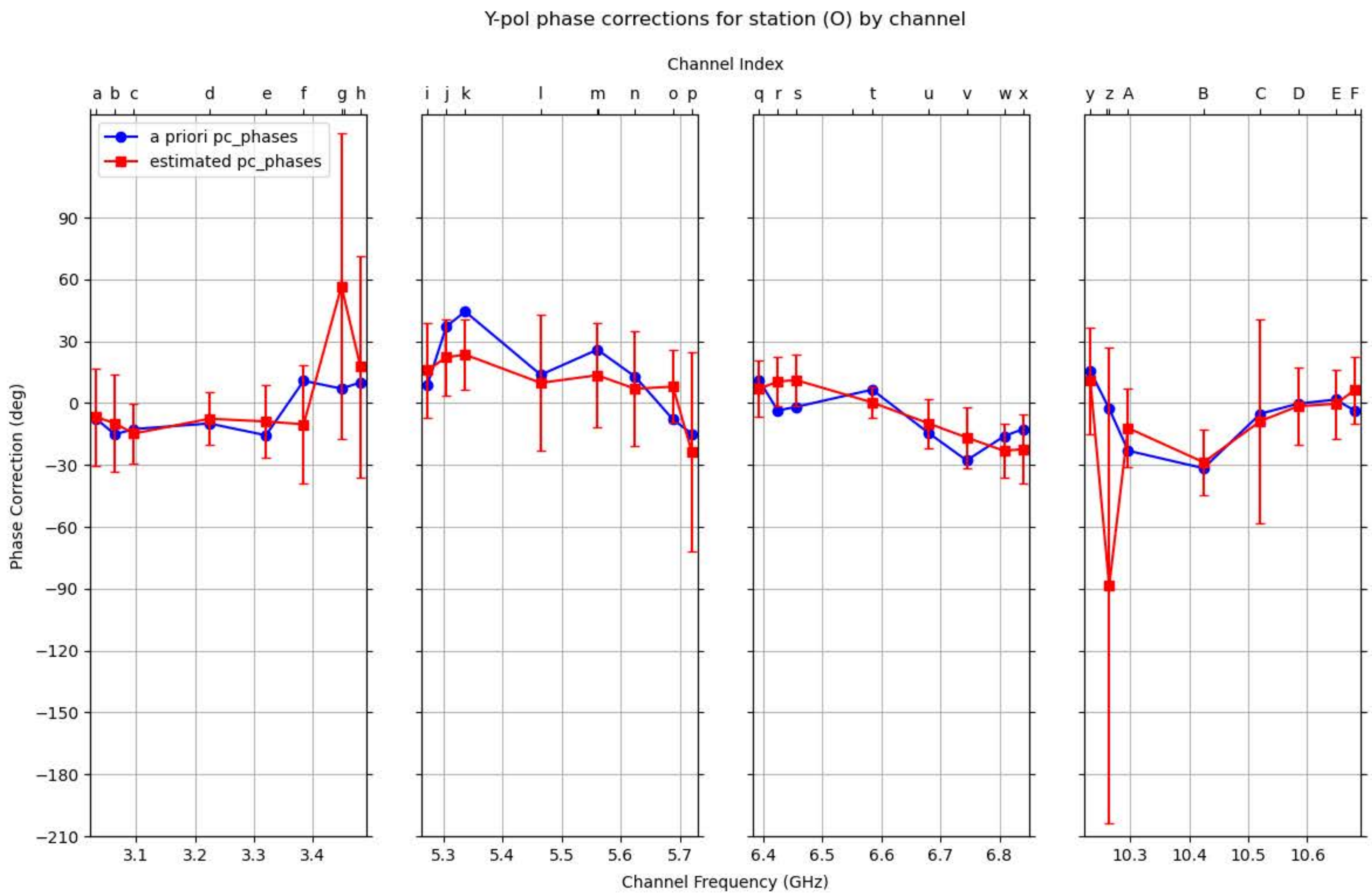
3) Run fourfit on the problem polarization, disable the ionospheric search ("ion_npts 1"), and force the problem station ionosphere to be the value of the trusted polarization (here, -27.8 TECU):

```
$ fourfit –m 1 -pt -c cf_3835_pcphases1 -b HP –P XX 279-1613a/0458-020.2YL7WX set ion_npts 1 if station H ionosphere 0.0 if station P ionosphere -27.8
```

This will output the residual phases; these are the new pc_phases_x for station P. Paste these into the control file. Katherine is the remote station so you will have to flip the sign. If you fourfit again, the XX polarization should have the same dTEC as the YY polarization (-27.8) and the residual phases should be the same in all channels & close to zero.

VGOS post-processing: ffres2pcp

Finally, check the pc_phase comparison plots. Keep an eye for bad channels or large corrections. Are there large (>20deg) corrections for one band? Do any channels have large error bars? Does that relate to a hardware modification at the station? Do the pc_phases vary smoothly within each band, or is one channel an outlier?



Here, Hobart Y-pol looks ok; probably channels g and z are bad; h, p, and C aren't great.

VGOS post-processing: fourphase

Once you have updated the pc_phases with ffres2pcp, it is time to run fourphase to calculate the delay and offset between the X and Y polarizations at each telescope.

fourphase has the same requirements as ffres2pcp: the dTEC difference among the polarization products must be less than 1, the minimum SNR among the polarization products must be greater than 15. Otherwise, the y-x delay you measure will be biased.

In principle this should be a smoother operation, since you have already fixed any problems while running ffres2pcp.

Command line syntax is:

```
$ fourphase.py cf_3862_pcphases H AEGJNOPSTVWX . -n 100 -s 15 -a 30
-d 1 -p
```

The reference station (here, H) is not meaningful; fourphase will fringe-fit every baseline.

This is a long job – for VR2502 (14 stations) it was 148k fourfit jobs!

If you need to stop the process, you can restart by navigating to the scratch folder and rerunning the same command line with the '-w' flag. Any previously generated fringe results will be preserved.

When finished, I like to rename the scratch folder to something I can remember:

```
$ mv scratch/YYYYDDMM-HHMMSS scratch/four1
```

VGOS post-processing: fourphase

fourphase.py uses the residual MBD and the fringe phase to calculate the delay and relative phase between the X and Y polarization feeds.

Each scan/baseline that passes cuts provides two measurements of each quantity for each station. For example, for the HE baseline:

$$H_{delay} = MBD_{yy} - MBD_{xy}$$
$$H_{delay} = MBD_{yx} - MBD_{xx}$$

$$E_{delay} = MBD_{xx} - MBD_{xy}$$
$$E_{delay} = MBD_{yx} - MBD_{yy}$$

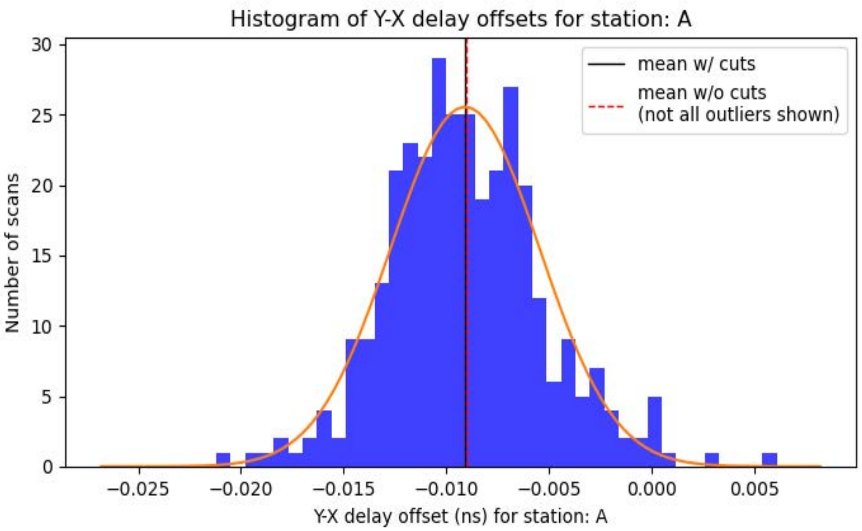
$$H_{phase} = Phase_{yy} - Phase_{xy}$$
$$H_{phase} = Phase_{yx} - Phase_{xx}$$

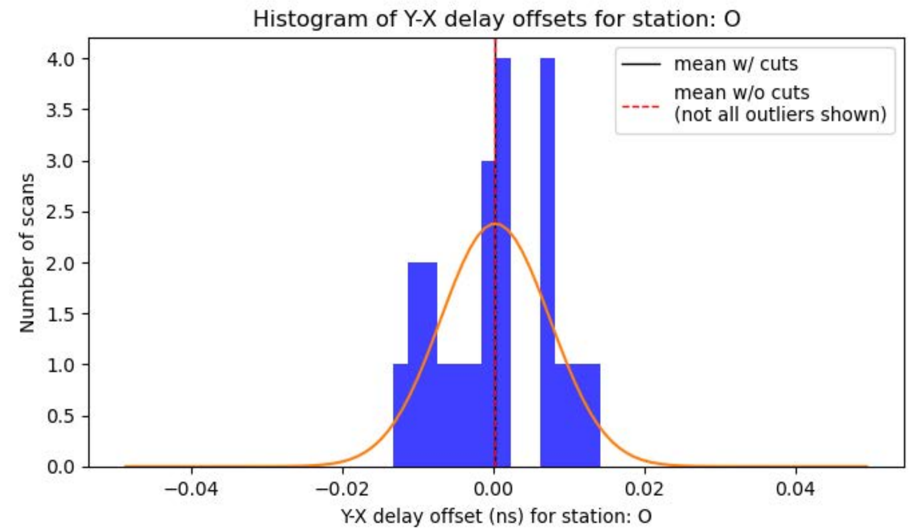
$$E_{phase} = Phase_{xx} - Phase_{xy}$$
$$E_{phase} = Phase_{yx} - Phase_{yy}$$

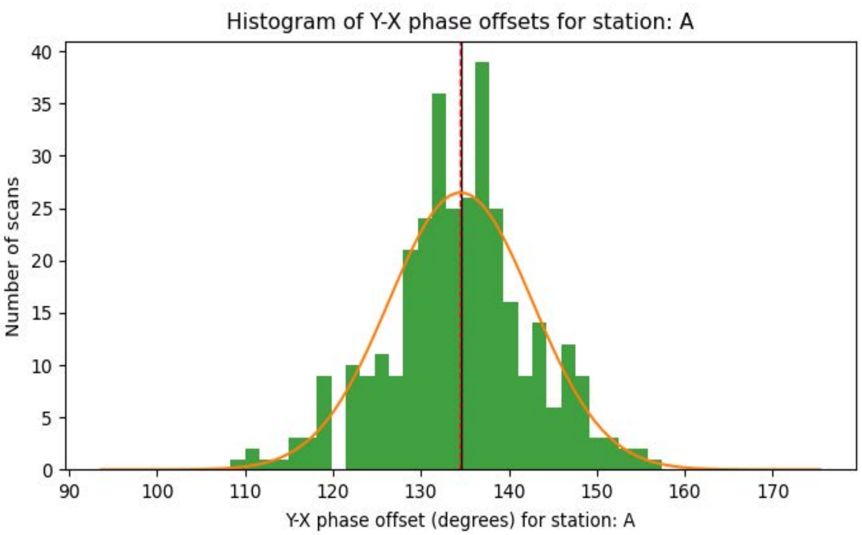
...and for each station they are averaged together. The results are in agreement with independent tests (eg known hardware delays and estimates from PolConvert of the complex cross-polarization bandpass).

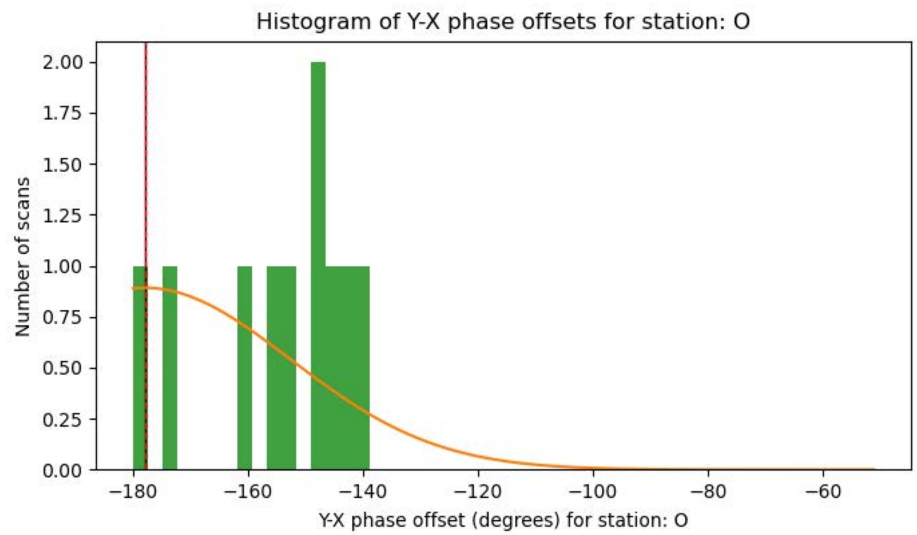
VGOS post-processing: fourphase

Example fourphase plots:









Santa Maria: good results, lots of scans

Hobart: a bit marginal in this session

ToDo: fix the phase wrapping in the plot so it displays the entries that have wrapped around to +180deg

VGOS post-processing: Pseudo-Stokes-I batch job

With the control file output from fourphase we are ready to run fourfit to calculate the pseudo-Stokes-I fringe result.

It's convenient to create another scratch directory to hold the fringe files, so they don't get confused with other products. I use a utility in the "hopstestb" python library. With the HOPS installation in your $PATH:

```
$ python
>> import hopstestb as ht
>> exp_dir = '/path/to/mk4files/1234/'
>> work_dir = '/path/to/mk4files/1234/scratch/Ixy1/1234'
>> ht.mirror_directory_with_symlinks(exp_dir, work_dir,
exclude_list=['prepass', 'scratch', 'diagnostics'])
```

This creates an "Ixy1" folder in your scratch directory. Copy the control file output from fourphase there, and the JSON file with the IONEX predictions, and run batch_fourfit:

```
$ cd scratch/Ixy1/1234/
$ batch_fourfit.py cf_3862_pstokes HAEGJNOPSTVWX I . -n 100 -p -t
vr2502_ionex_tec.json.jpl
```

ToDo: provide a 'make scratch folder' option for batch_fourfit.

VGOS data quality checks

Once the Ixy job is finished, we can make data quality checks and determine any bad channels that should be ignored.

I make the following checks, using scripts that are available in HOPS:

- Channel phase residuals for each station
- Channel phase residuals, as a timeseries on each baseline
- Phasecal amplitudes for each station
- Phasecal amplitudes for each station over time
- Channel amplitudes for each station

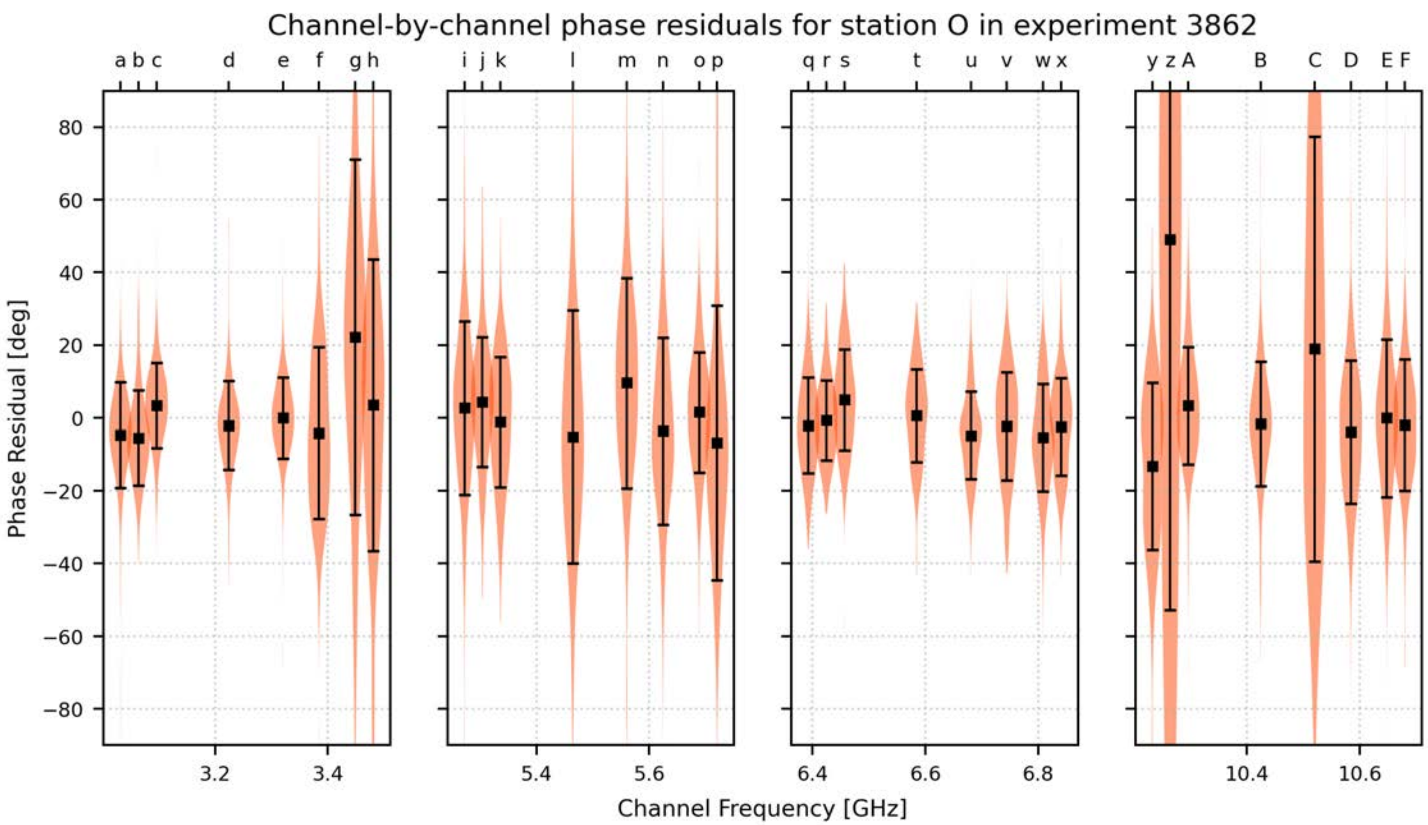
I also tabulate G codes for each channel from each station and from each source, and tabulate H codes from each station.

VGOS data quality checks: Channel phase residuals

In the folder with the Ixy fringe results, run the following for each station:

```
$ channel_phase_resid.py cf_3862_pstokes O HP I . -n 10 -p -s 20 -q 5
```

This generates a plot for Hobart (O) using data from scans/baselines with Kokee (H) and Yarragadee (P). The channel phase residuals for each reference station (in this case Hobart) will be a combination of the noise in the reference station and the remote stations, so choose remote stations that have good data quality.



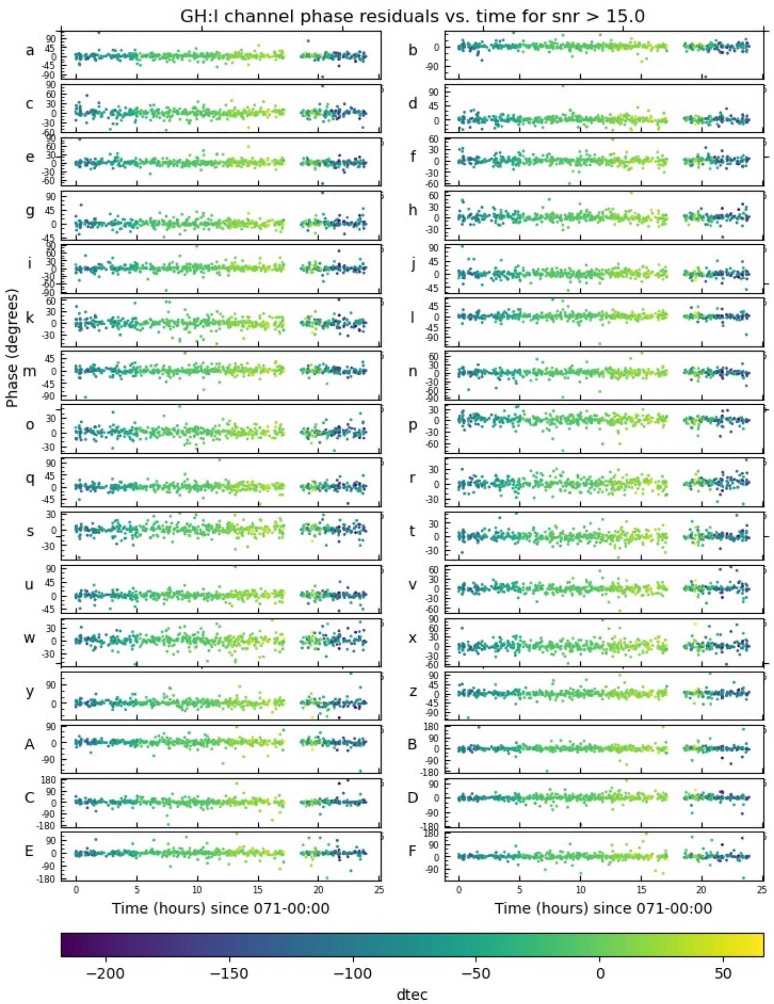
Channels g,h,z,C are outliers (large phase offsets, large scatter).

VGOS data quality checks: Channel residuals over time for each baseline

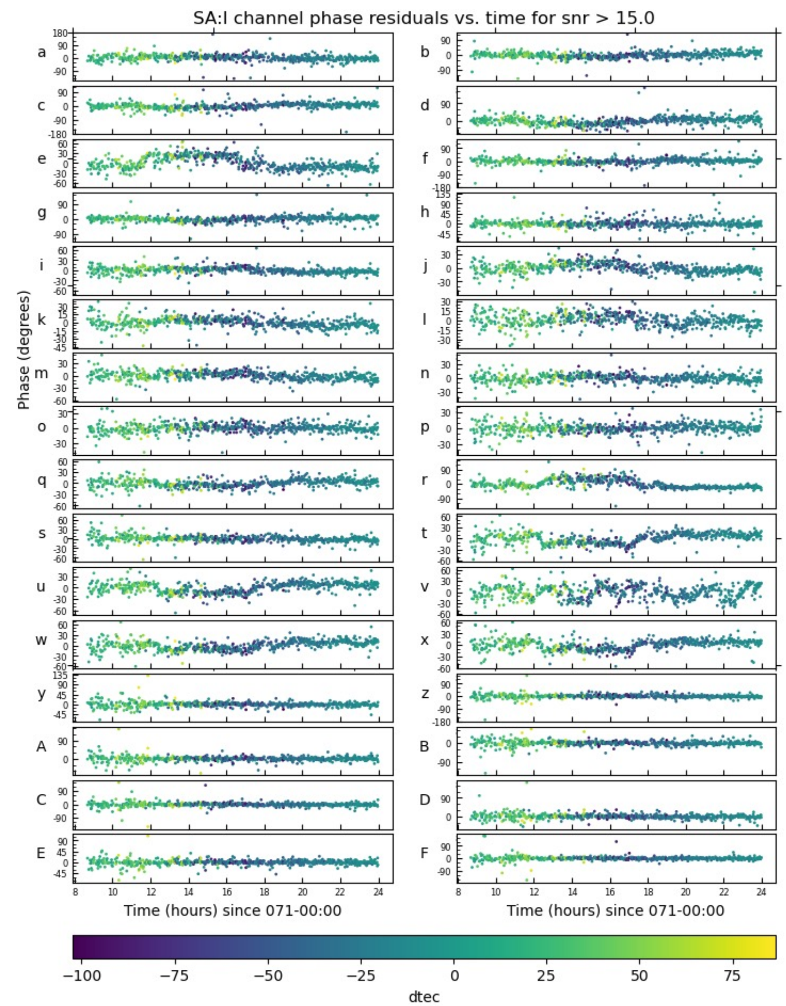
In the folder with the Ixy fringe results, run the following:

```
$ phase_resid.py cf_3862_pstokes HAEGJNOPSTVWX I . -z dtec -n 100 -s 15 -q 5
```

This generates plots for all baselines.



GGAO-Kokee, looks very good



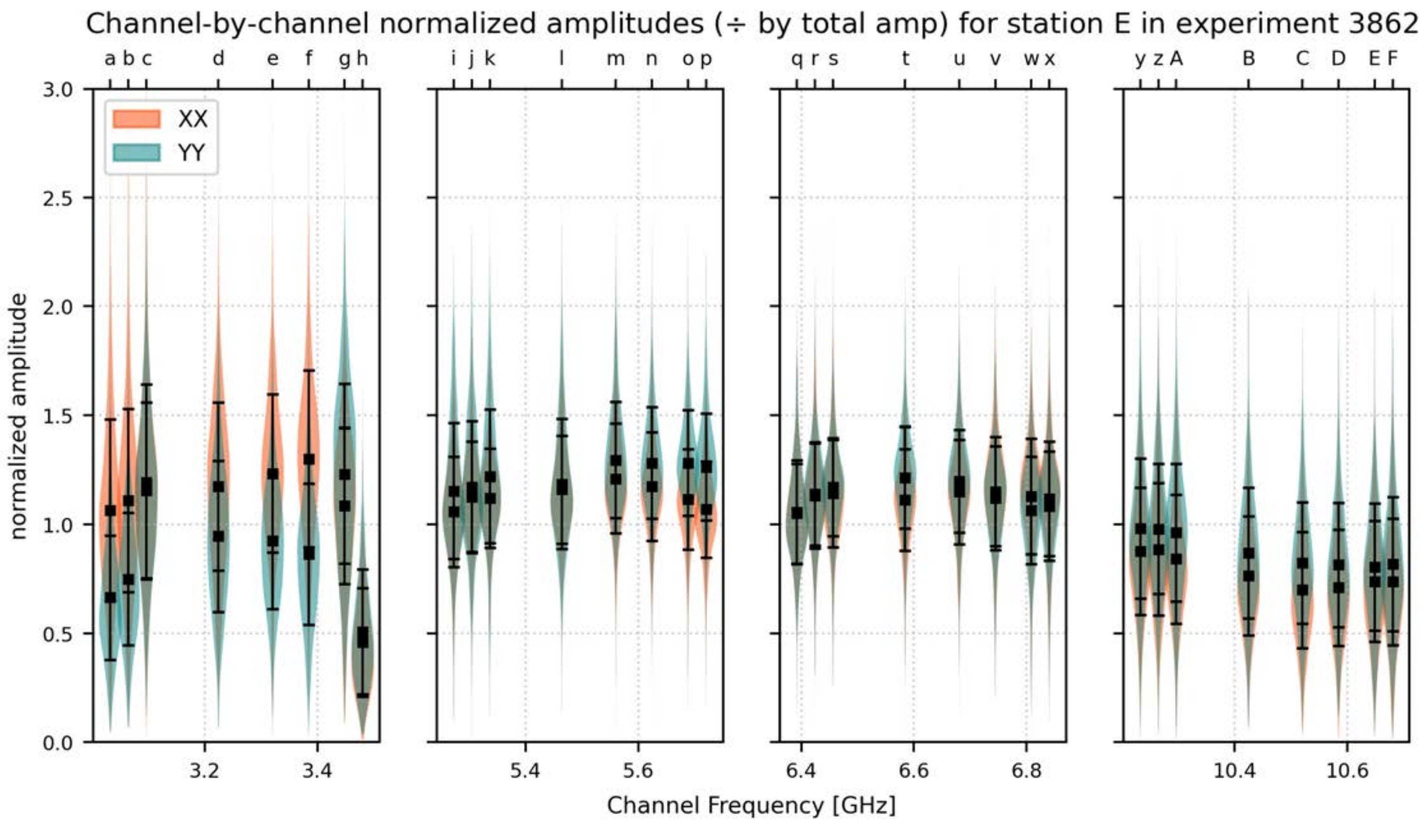
Onsala-East – Santa Maria, lots of time-varying structure

VGOS data quality checks: Channel amplitudes

In the folder with the fourphase fringe results, run the following for each station:

```
channel_amplitude_compare.py cf_3862_pcphases E GSTW . -s 15 -q 5
```

Use the control file that was the input to fourphase.py. This generates a plot for Westford (E) using data from scans/baselines with GGAO, Onsala-East, Onsala-West, and Wettzell-North. It plots the fringe amplitude in each channel, so you can spot weak channels or weak bands.



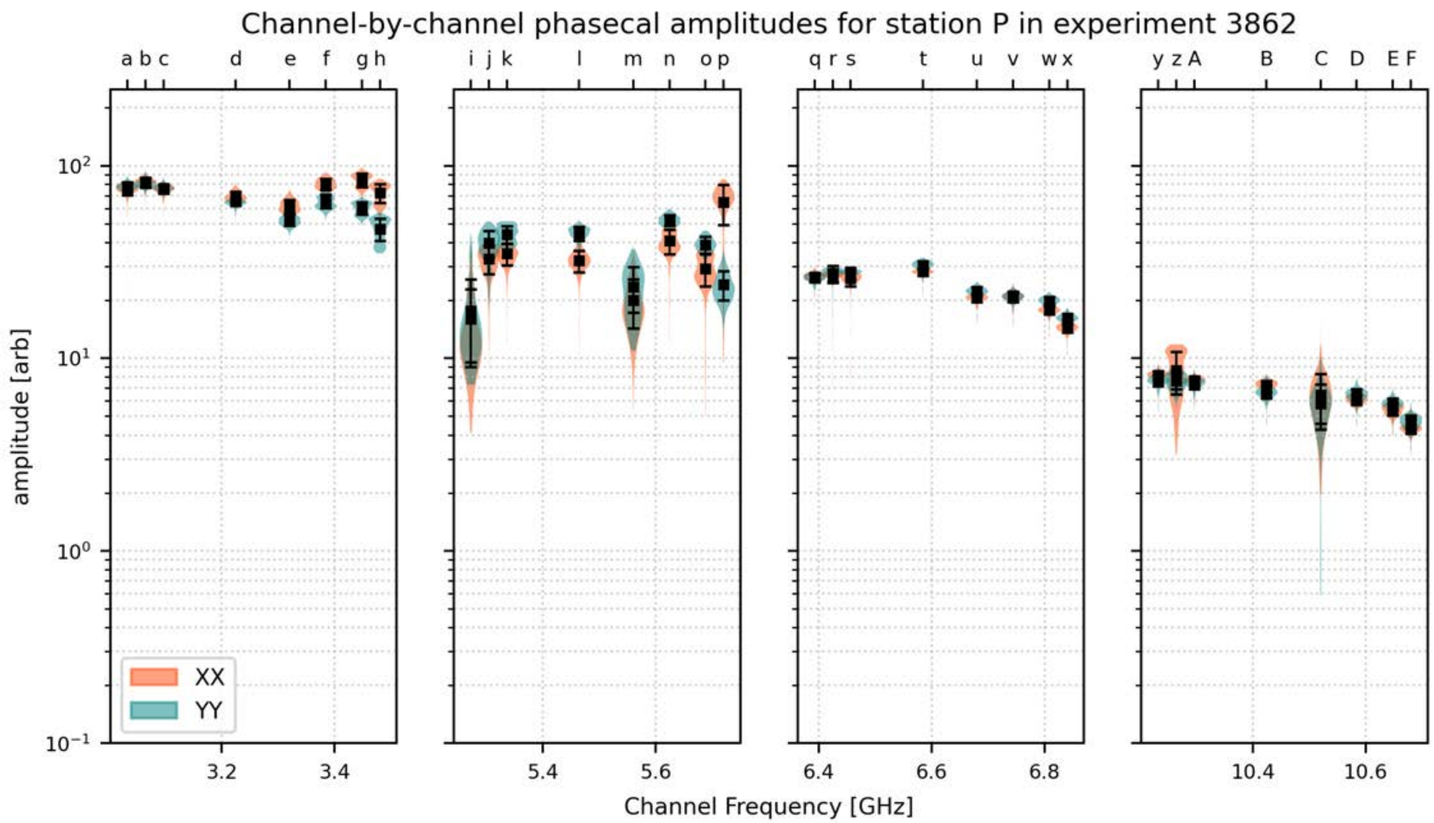
Westford has low amplitude in channel h. The bands are all about equal. Y-pol in bandA is a little weak.

VGOS data quality checks: Phasecal amplitudes

In the folder with the fourphase fringe results, run the following for each station:

```
$ channel_pcal_amp.py cf_3862_pcphases P HO . -s 10 -q 5
```

This generates a plot for Yarragadee (P) using scans with Kokee (H) and Hobart (O). For this result, the remote stations don't matter – choose stations that provide a large number of scans for the reference station.



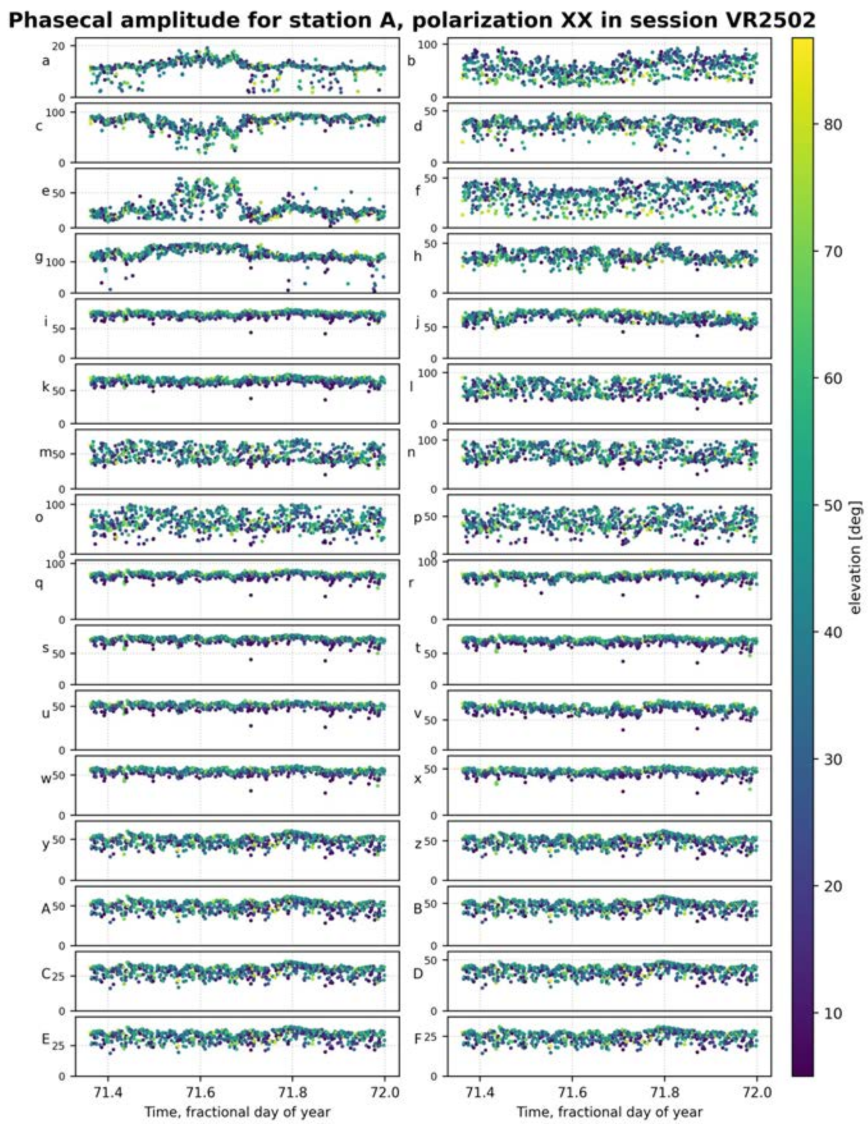
Phasecal amplitudes decline with frequency (as expected). Channels in bandB (i,m,p) have variable amplitude. Also channels z and C.

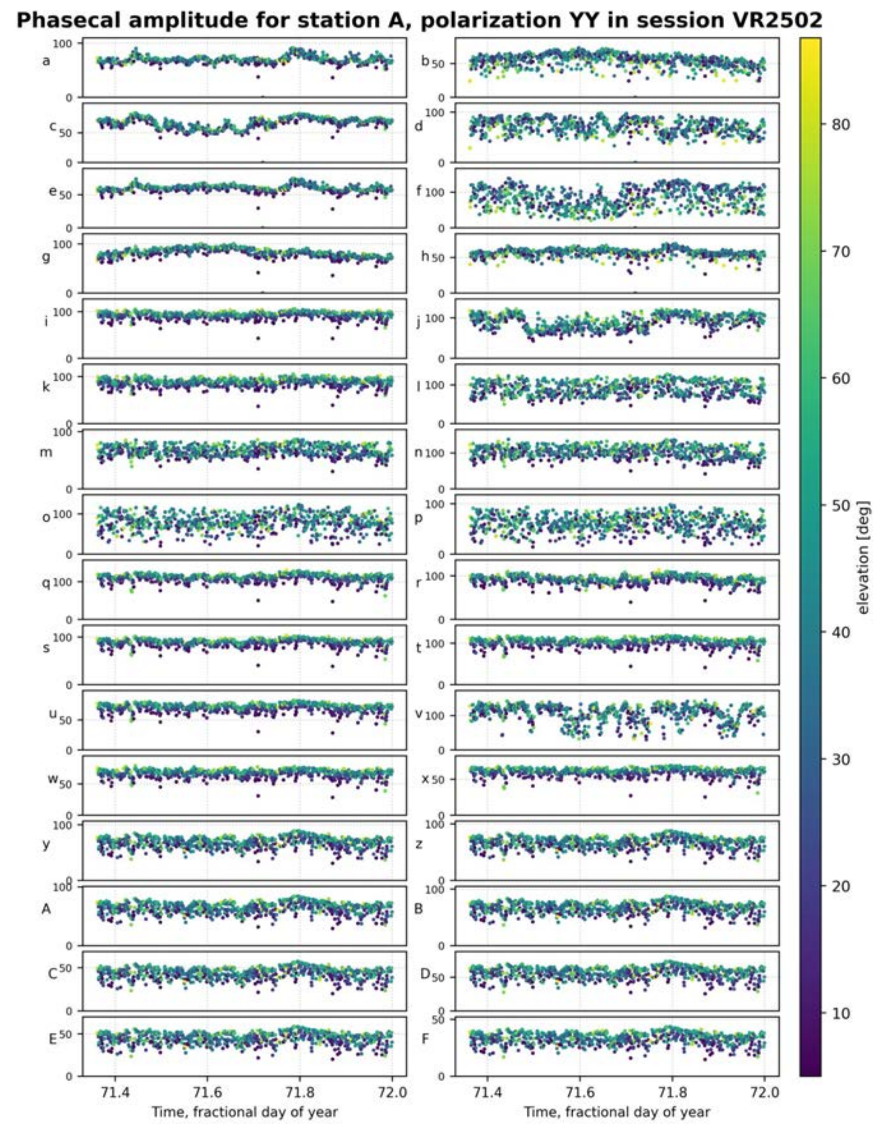
VGOS data quality checks: Phasecal amplitudes over time

In the folder with the fourphase fringe results, run the following for each station:

```
$ pcal_amp_over_time.py cf_3862_pcphases A GNT XX ./ -e VR2502 -p elev
```

This generates a plot for Santa Maria (A) X-pol using data from scans with GGAO, NyAelsund and Onsala-West. Datapoint color will be drawn according to Santa Maria's antenna elevation.



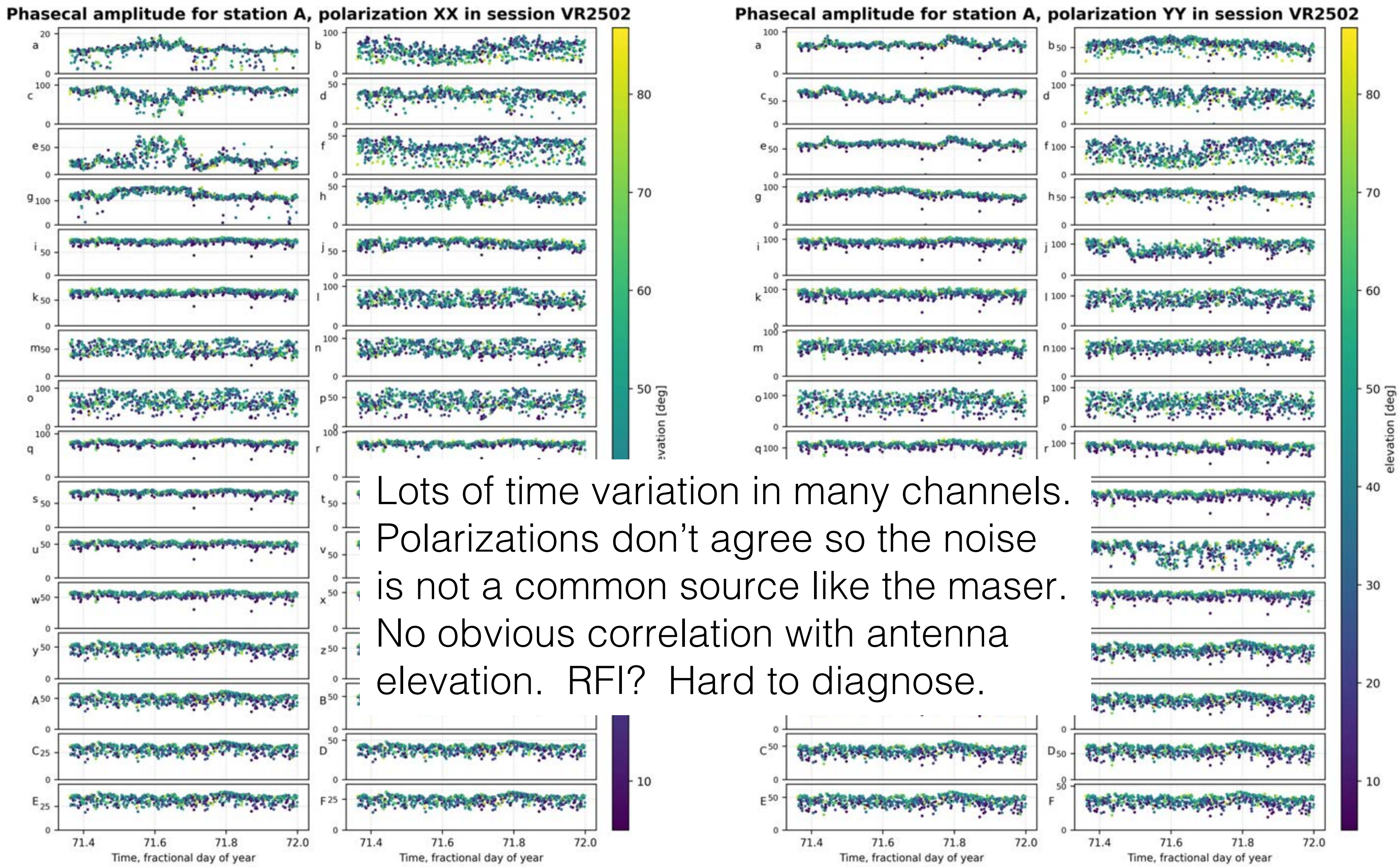


VGOS data quality checks: Phasecal amplitudes over time

In the folder with the fourphase fringe results, run the following for each station:

```
$ pcal_amp_over_time.py cf_3862_pcphases A GNT XX ./ -e VR2502 -p elev
```

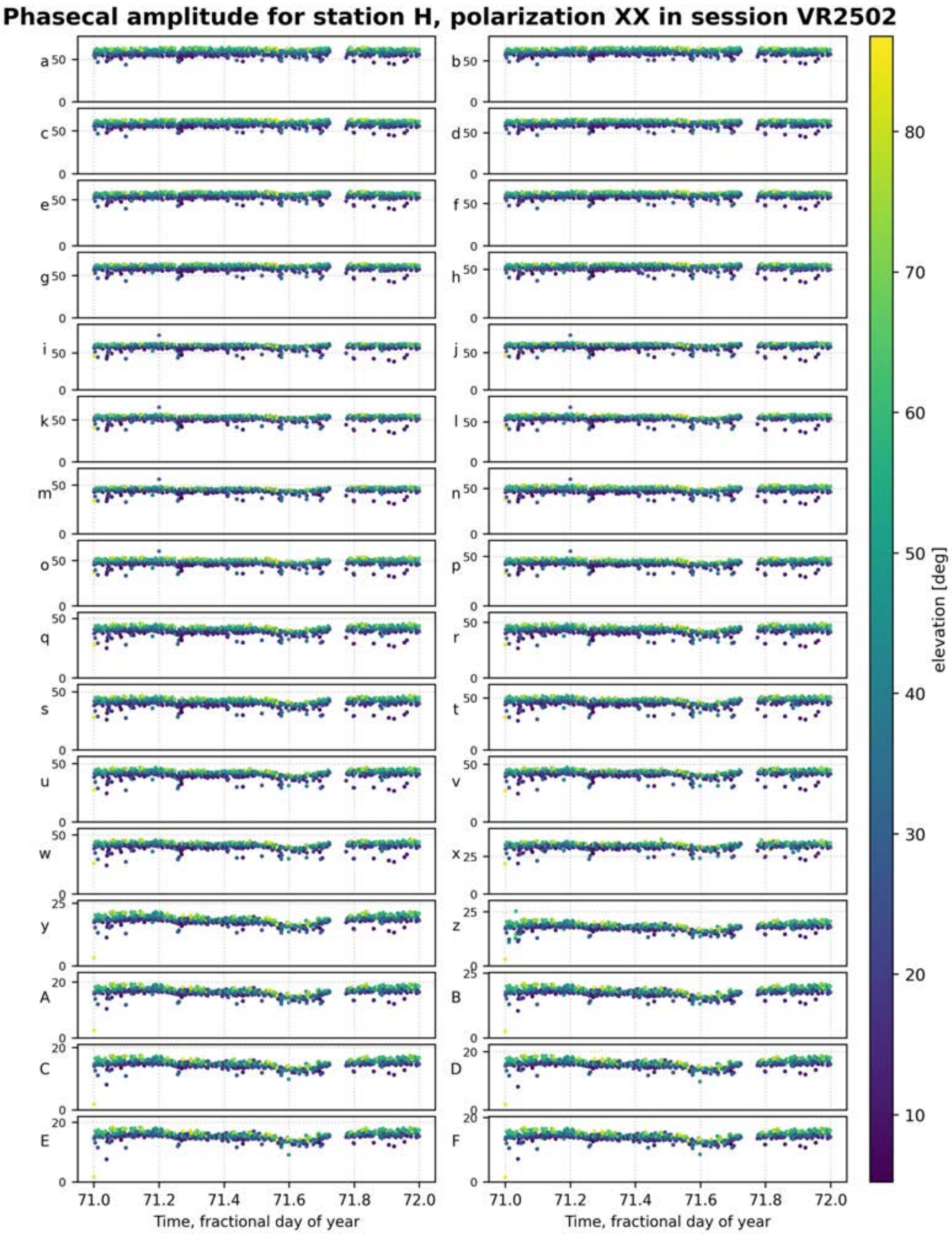
This generates a plot for Santa Maria (A) X-pol using data from scans with GGAO, NyAelsund and Onsala-West. Datapoint color will be drawn according to Santa Maria's antenna elevation.

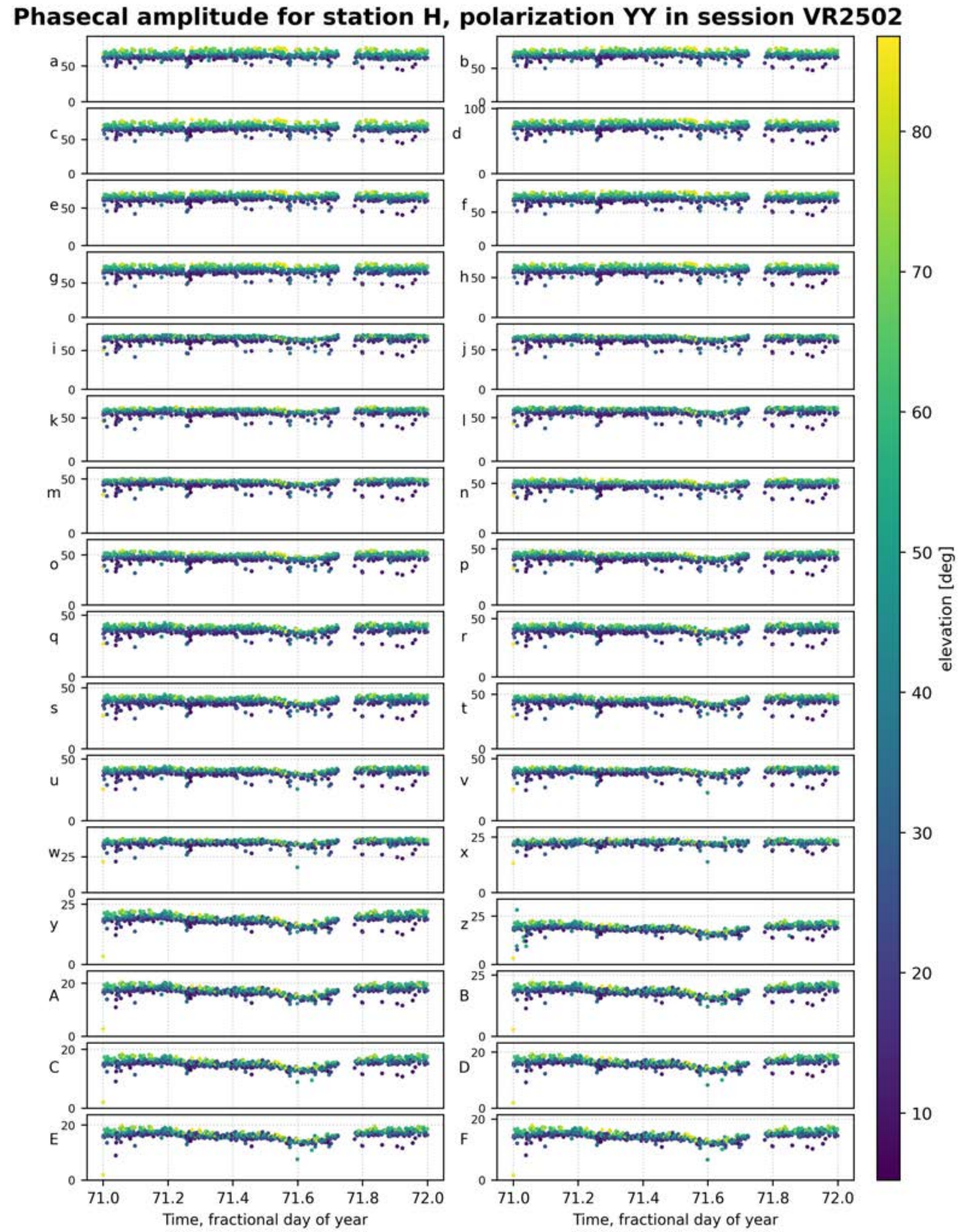


Lots of time variation in many channels. Polarizations don't agree so the noise is not a common source like the maser. No obvious correlation with antenna elevation. RFI? Hard to diagnose.

VGOS data quality checks: Phasecal amplitudes over time

Example of a “good” station, Kokee.

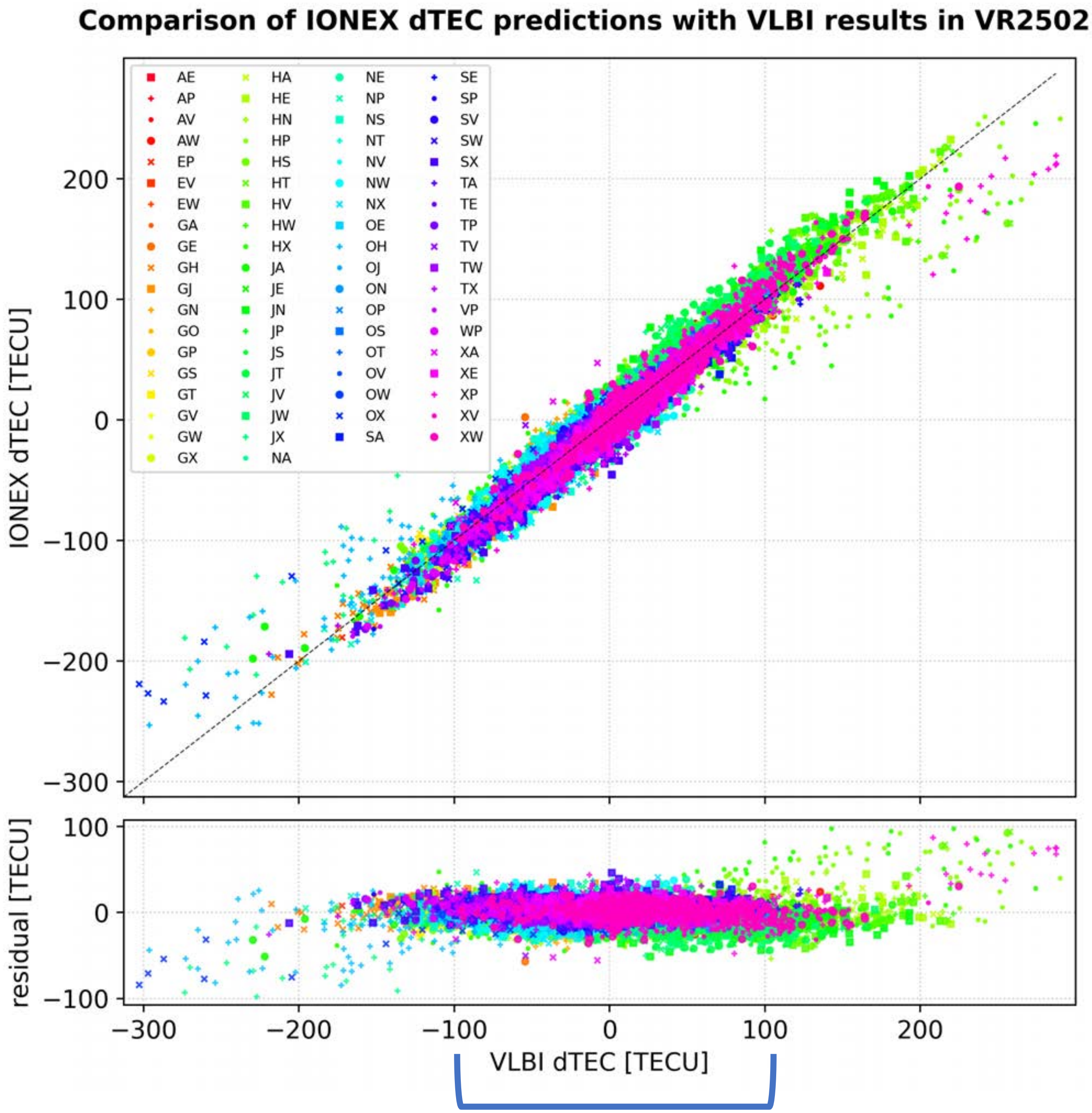




VGOS data quality checks : Compare dTEC from IONEX

The Sun is still very active, and we routinely observe dTEC on many baselines well outside of the standard fourfit ionospheric search window. This period of solar max will continue for another ~2 years.

It’s difficult to expand the search window due to the limit on the number of search points (ion_npts 75). If the search is too coarse you many miss the dTEC peak entirely.



Nominal search range +/-100

ToDo: make this plotting script available

Proxy cable cal

For stations without a self-contained cable delay measurement system, we need to provide an alternative delay correction to account for the delays induced by telescope motion and cable flexture.

Run this script to fit a delay to the phasecal tones in each band and polarization:

```
$ pcc_generate.py -o pcc_dat/ -e -f -v 3 HGIKOPVW ./
```

In principle, a cable delay should be common to all bands and polarizations; the select_bandpols.py script will choose the most similar bands-polarizations using a cross-correlation. Run this for each station:

```
$ select_bandpols.py -s G -d . -o pltG -m 0.9
```

Then, generate PCMT files with the band-pol selections:

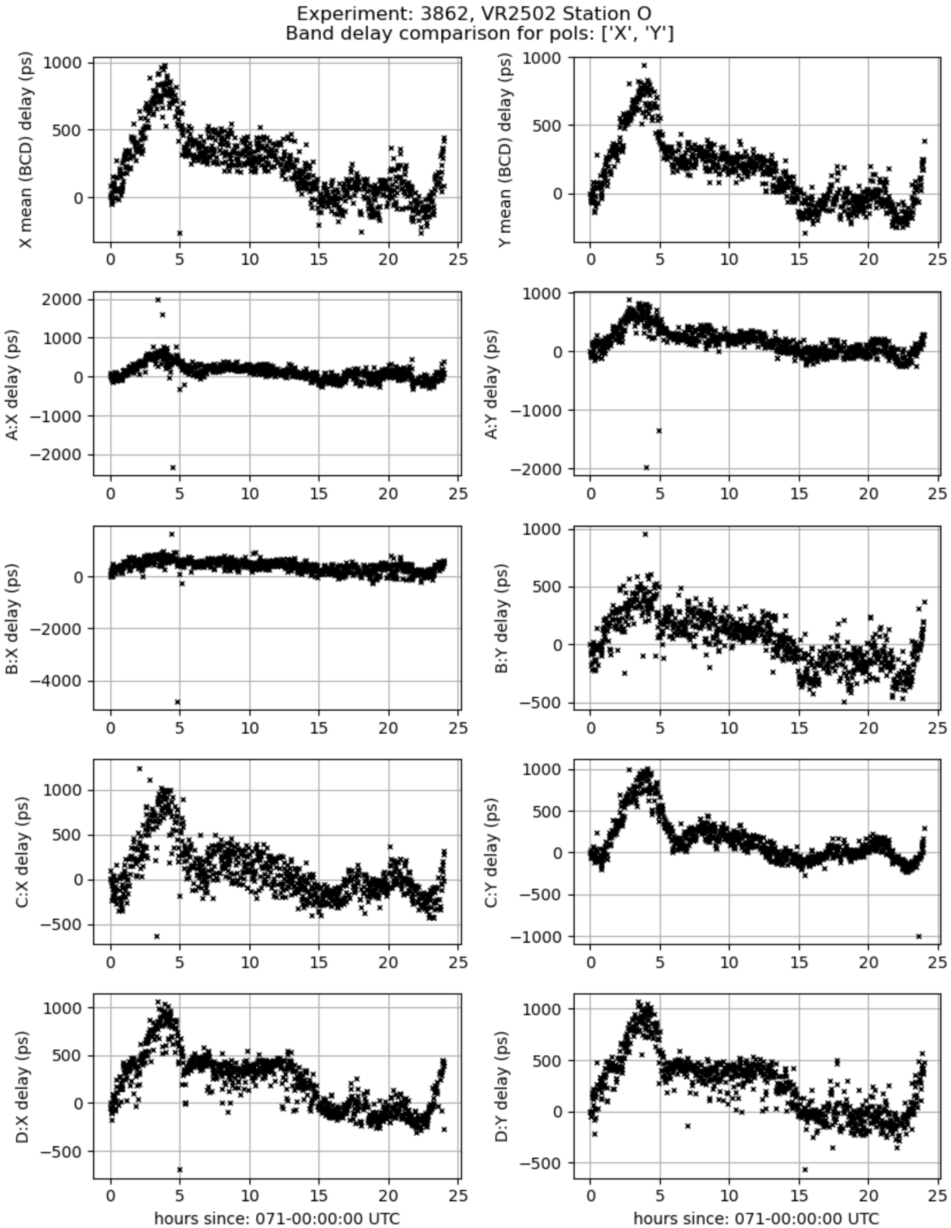
```
$ pcc_select.py -e vr2502 -d pcc_dat/ -o pcmt/ -s G:BX,BY,CX,CY,DX,DY
```

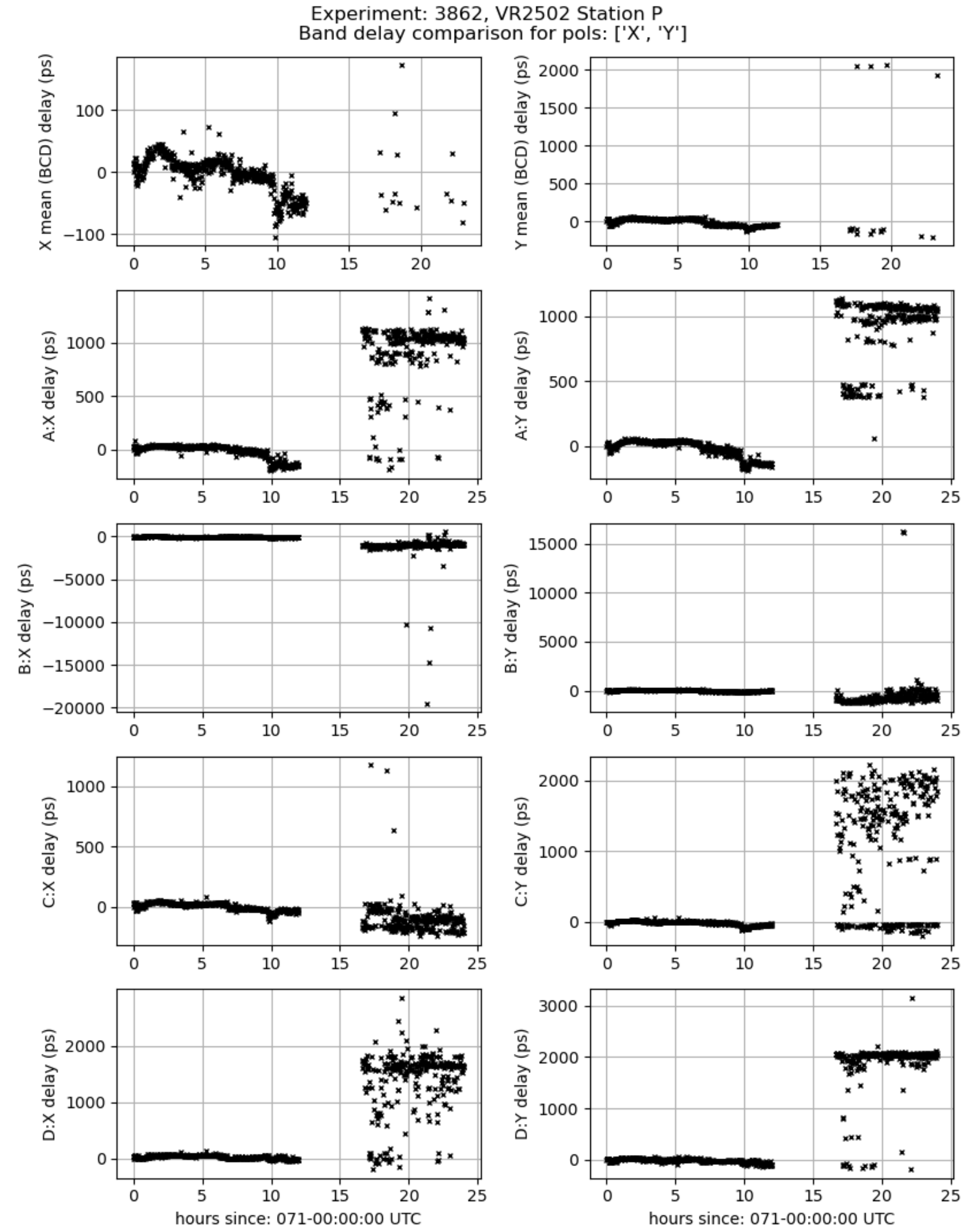
Don’t use bandA from the NASA stations (Kokee, MGO, GGAO, also Westford) – those use a different cable type for bandA and the delay will not agree.

The following stations have cable cal systems and don’t need proxy cable cal:
Santa Maria
Westford
NyAlesund
Both Onsalas

Proxy cable cal

The proxy cable cal results are often confusing.





Data quality notes for the correlator report

As I'm working on a session I keep notes in a spreadsheet, and mention problems in the correlator report.

The correlator report is important for:

- Stations, to diagnose and fix problems
- Analysts, to anticipate which stations should have additional clock breaks, whether to deweight the data, if proxy cc corrections are useful...
- Correlators, to be aware of similar problems in other sessions!

Station	Station	Scans	Data loss?	fmout clock model	Samplers	pc_phases	y-x delay	Phase resid	Pcal amps	Channel amps	G codes	H codes	Proxy cc
H	Kokee												
A	Santa Maria	Late start 0840				bandA		e,v		a	a		
E	Westford	Azimuth drive problems 0408-1210		Large rate		channel h		h	bandA	h	h		
G	GGAO			Large rate									
J	HartVGS	Missing data files after 2335				channel d		d	variable		bandA		
N	NyAelsund					channel c,h		a,b,c	very strange structure				
O	Hobart					channel z	few scans	g,h,z,C	very weak	g,h,l	g,h,l,z	75%	large corrections
P	Yarragadee	DBBC problems 1159-1637				channel p,C		p,C			C		clock break; bimodal
S	Onsala-E	Missing data files 0057-0123				channel r		r	big dropout, all channels		h,r		
T	Onsala-W	Missing data files 0057-0123				bandA X-pol			big dropout				
V	Wettzell-S	Recorder problem 0907-1723											
W	Wettzell-N										E,F		
X	Seshan		Missing threads; 13% loss			few scans, channels e,f,g,h,B		w,y,z,A		w		25%	
Z	Tianma				No phasecal								

Closeout

VGOS is an ongoing R&D project, and our methods and tools are changing as we collect more data!

Please email the IVS-corr mailing list with your questions & problems.