

running the gnssrefl code

Kristine M. Larson

<https://kristinelarson.net>

<https://github.com/kristinemlarson>

kristinem.larson@gmail.com

Where did gnssrefl come from?

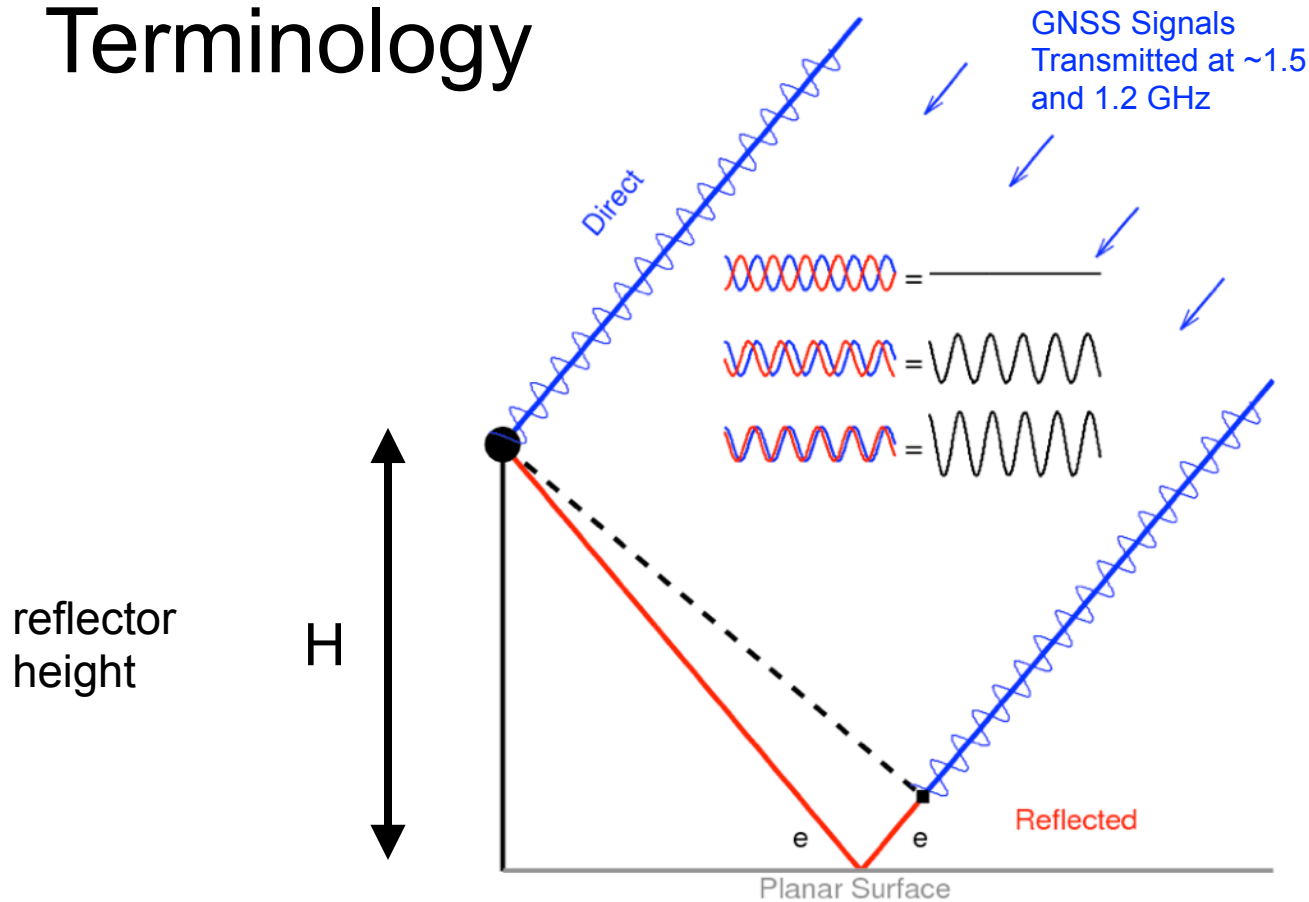
- PBO H₂O was a GPS research group based in Boulder that made environmental products (soil moisture, snow, and vegetation) from data collected by the Plate Boundary Observatory GPS network:
 - Fortran for everything computationally intensive
 - c-shell scripts to fetch files
 - lots and lots of plain text files
 - Matlab for environmental models
 - Web programmer (Evan Crouch) to make products available
- Ran with NSF support until 2017
 - With NASA support, we gave the PBO H₂O code to JPL
 - My fortran-based RINEX translation codes were/are public.
 - I took a very tiny part of the code - the periodograms - translated them to Matlab for the cryosphere community.
 - Carolyn Roesler and I took our reflection zone mapping knowledge and put it into Matlab.

So what is gnssrefl?

- Effort to take the knowledge gained from PBO H₂O (and tide gauge work done in parallel) and provide it in an accessible way to the community.
- Open source python hosted by GitHub (embedded Fortran)
- multi-GNSS by design
- no shell scripts, queries GNSS archives and orbits for you
- RINEX 2.11, RINEX 3+ (NMEA)
- Soil moisture, snow accumulation, water levels.
- readthedocs documentation with use cases
- Dockers and Jupyter Notebooks.

Felipe has gone through the
theoretical background for GNSS-IR

Terminology



- The interference pattern has a frequency directly related to H , the distance between the reflecting surface and the antenna
- Footprint of GNSS-IR method depends on H .
- Reflection surface also has an impact on the interference pattern, primarily in how strong the reflection is.

How to implement that ...

You can say to *extract the dominant frequency*, but what is the best way? What is the fastest way? How can you tell it is a “good” reflector height? Which GNSS frequencies should you use? How can I compute the reflection zones? Etc Etc

Basic Steps

- You need GNSS data.
 - your own data or you can pick one of our supported archives
 - Must be compliant (header info, SNR data, follows community naming rules)
- You need orbits
 - GPS (default) or multi GNSS
- Define the region where you think GNSS reflections will be useful.
- Isolate each rising and setting GNSS satellite arc in that region
- Find the frequency of each (useful) interference pattern - relate that to H.
- Make sure you trust those H values.
- Repeat, repeat, repeat

let's go thru these steps

Basic Steps

- You need GNSS data.
 - your own data or you can pick one of our supported archives
 - Must be compliant (header info, SNR data, follows community naming rules)
- You need orbits
 - GPS (default) or multi GNSS

>>>> This is the data translation stage <<<<<

Basic Steps

- Define the region where you think GNSS reflections will be useful.
- Isolate each rising and setting GNSS satellite arc in that region
- Find the frequency of each (useful) interference pattern - relate that to H.

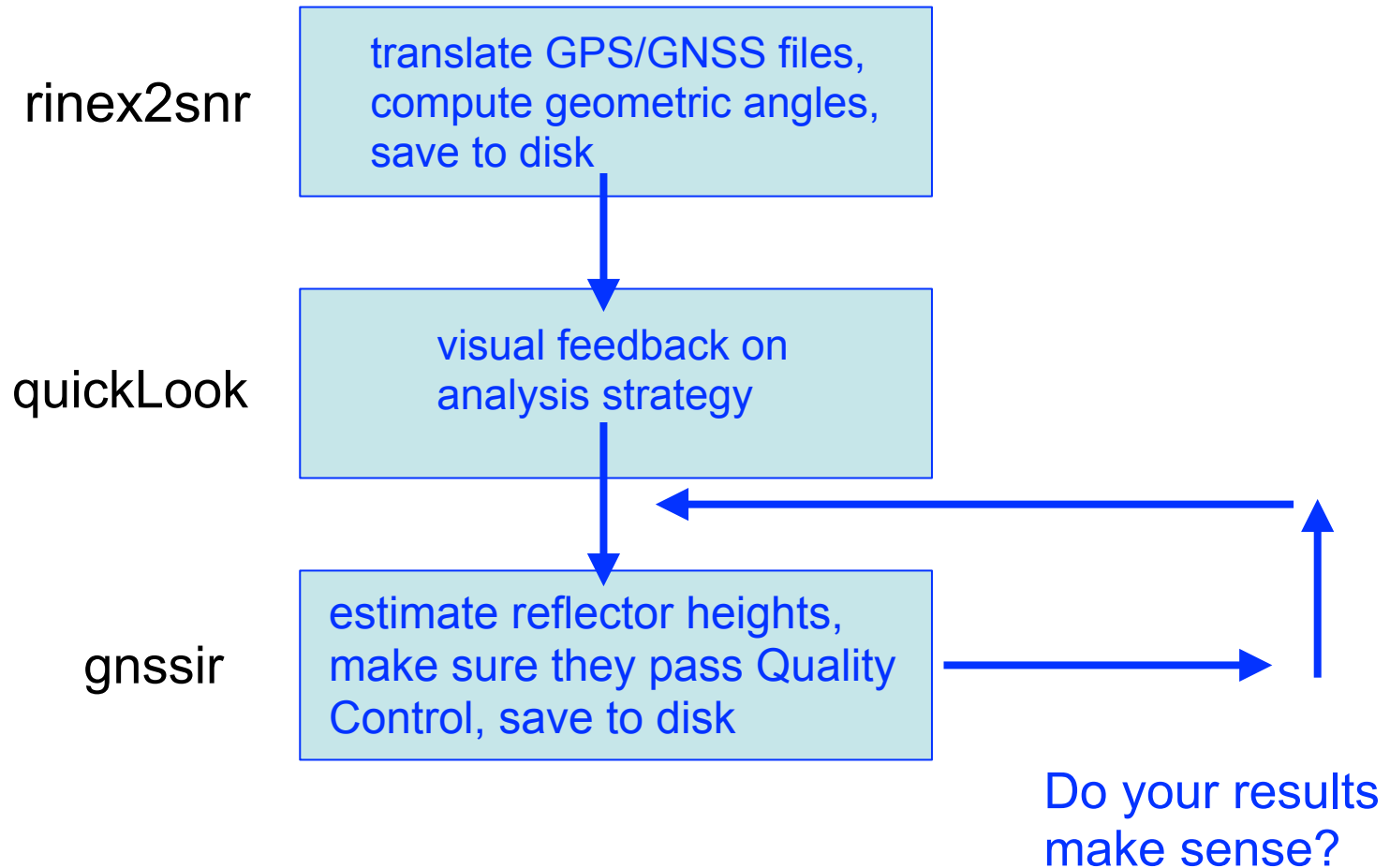
>>>> This is when you start to make decisions about the analysis choices (the azimuth and elevation angle masks), frequencies, and quality of the data outputs (the reflector heights) <<<<<

Basic Steps

- Make sure you trust those H values.
- Repeat, repeat, repeat

>>>> Analyze a lot of data - but make sure you have good quality control settings. Modify choices if needed. <<<<<

gnssrefl software building blocks



gnssrefl

rinex2snr

translate GPS/GNSS files,
compute geometric angles,
save to disk



gnssir

estimate reflector heights,
make sure they pass Quality
Control, save to disk



snowdepth

vwc

water
levels

The gnsrefl code uses environment variables

- Settings:
 - With github or pypi, these environment variables should be stored in your .bashrc
 - Using the docker, these are defined for you.
 - Using the Jupyter notebooks, these are defined for you.
- EXE - this is where support executables, CRX2RNX and gfzrnrx, are stored.
- REFL_CODE - main directory for everything related to gnsrefl.
- ORBITS - where the GNSS orbits will be stored. This can be the same as REFL_CODE.

GPS/GNSS data

- We primarily use the community standards, RINEX 2.11 and RINEX 3.
- NMEA is possible.
- Your file MUST have SNR data in it. Why?
- Your file MUST have coordinates in the header. Why?
- RINEX 2.11: lowercase filenames and naming conventions used by geodetic archives (e.g. station names are four characters).
 - Example, station p041 day of year 30 and year 2020 is called p0410300.20o
 - If the file is Hatanaka compressed, p0410300.20d
- RINEX 3: filenames are very complicated.
- Can you use your own data? Yes. But you still need to follow the rules.
- We suggest: practice with our highlighted use cases first. Then try new data.
- (I also have a small video on GNSS files and lingo if you have an interest)

step 1: rinex2snr

picks up and translates RINEX files, picks up orbits, computes geometric angles, and extracts SNR data

- Main input - GPS/GNSS RINEX files
 - your own RINEX files
 - or the code will download them from a long list of international geodetic archives
- The output is called a SNR file. Unfortunately these files have non-intuitive file endings.
- Defaults
 - GPS only (L1, L2, L5)
 - elevation angles < 30 degrees
- `rinex2snr -h` and the GitHub documentation for more information
- command line requires station, year, and day of year

rinex2snr p041 2020 132

would produce this file:

`$REFL_CODE/2020/snr/p041/p0411320.20.snr66`

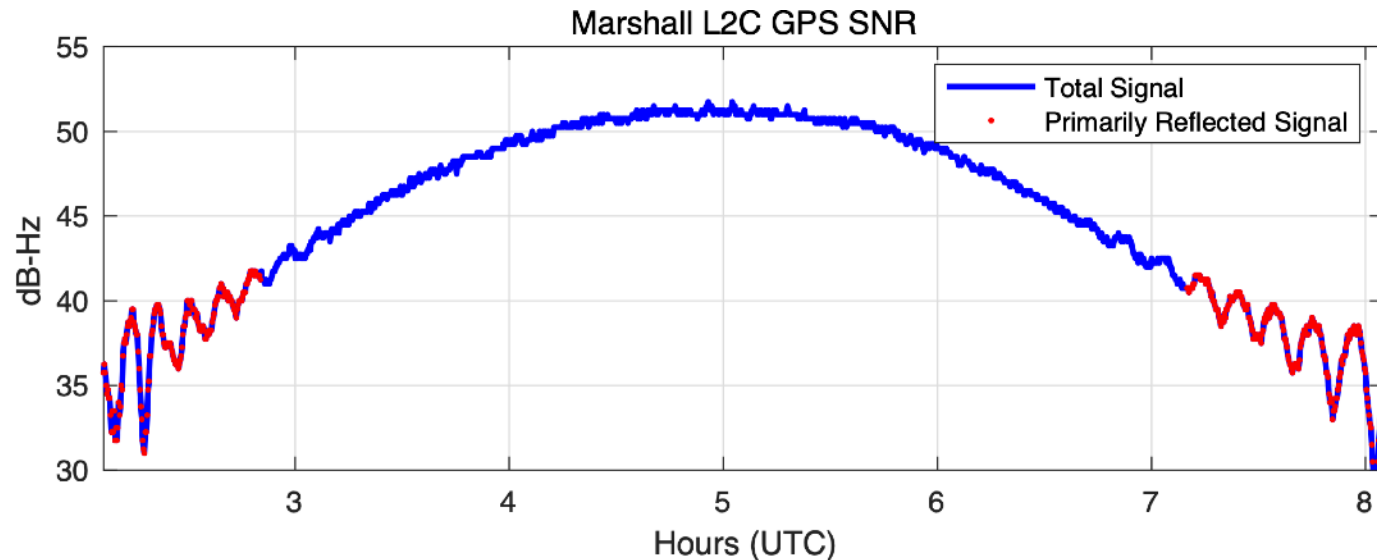
rinex2snr Options

https://gnsrefl.readthedocs.io/en/latest/api/gnsrefl.rinex2snr_cl.html

What is stored in a SNR file?

This is for one satellite pass

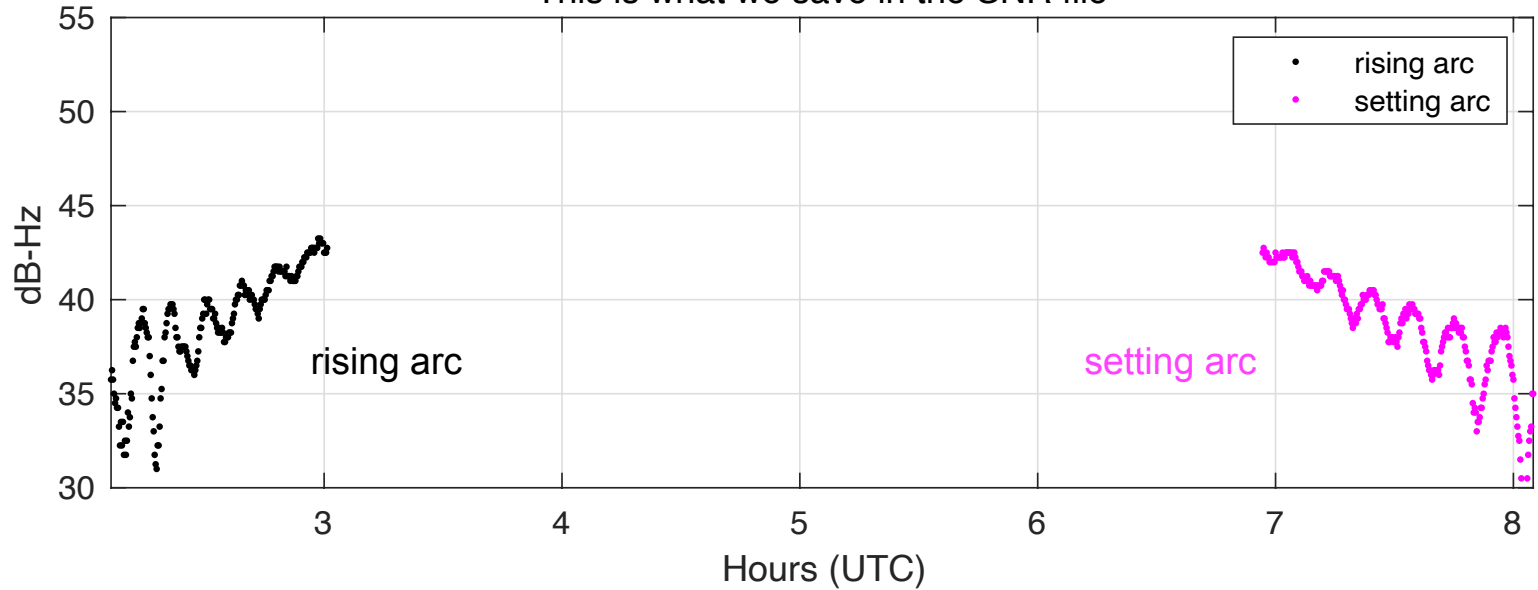
Satellite Number
Time
Elevation Angle
Azimuth Angle
SNR data



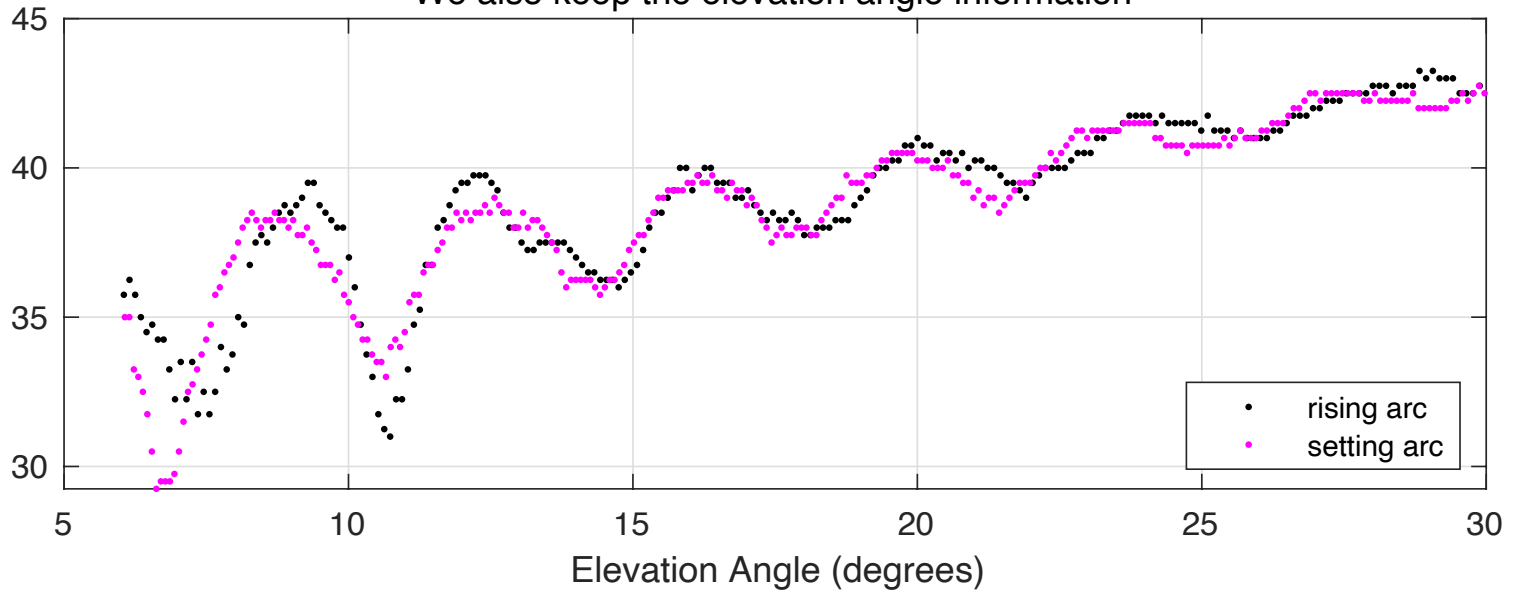
This is what SNR data look like in the time domain

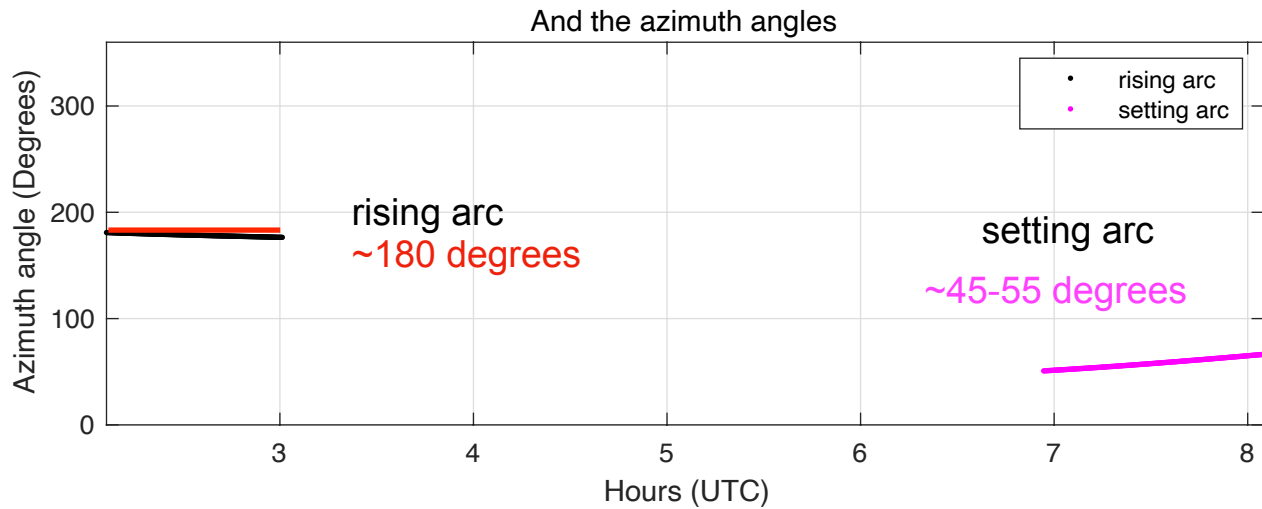
We really only want the red part - the low elevation angles -where we can see reflected signals.

This is what we save in the SNR file

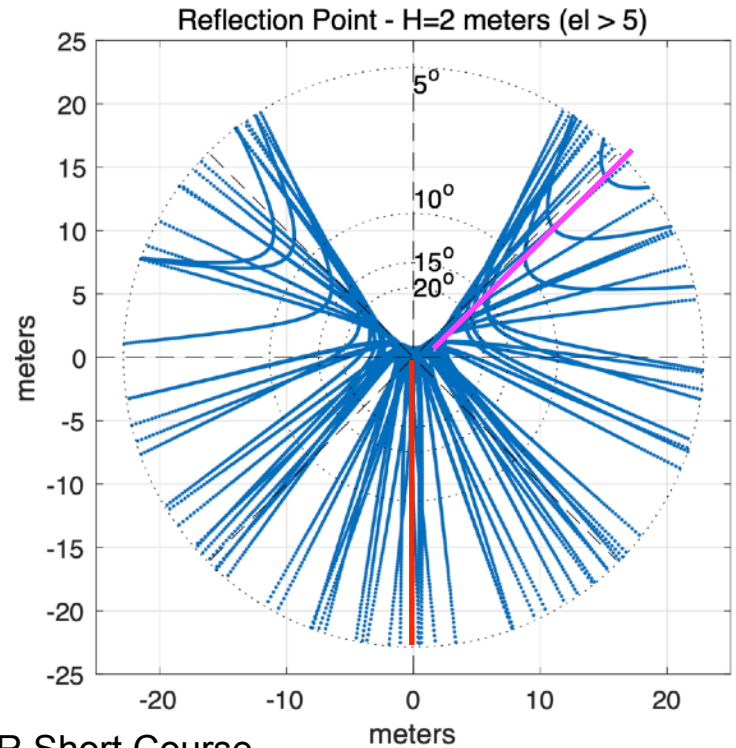


We also keep the elevation angle information





One satellite arc - you can have up to two per satellite per day.



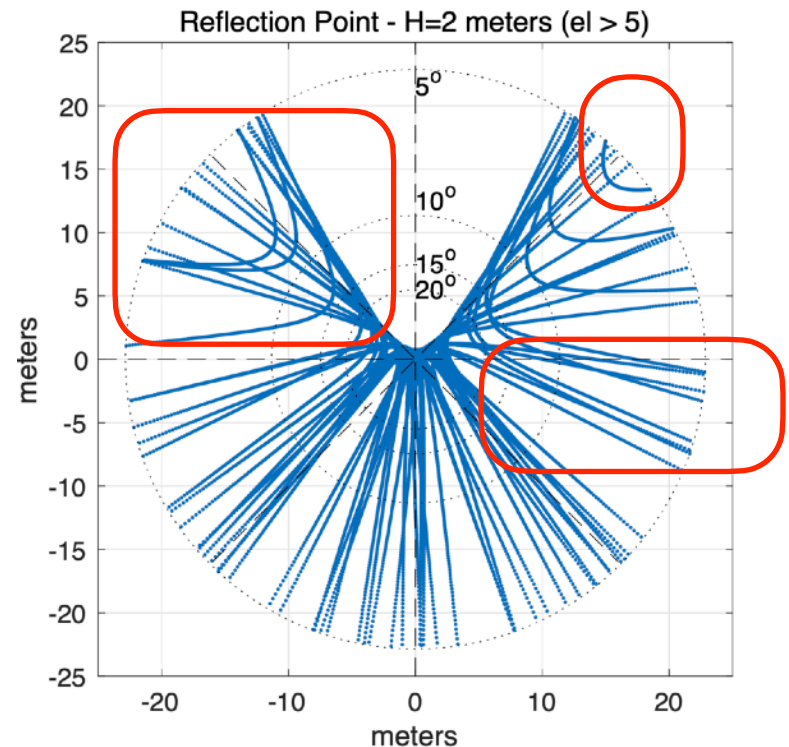
Where do files live?

File structure for station abcd in the year YYYY (last two characters YY), doy DDD:

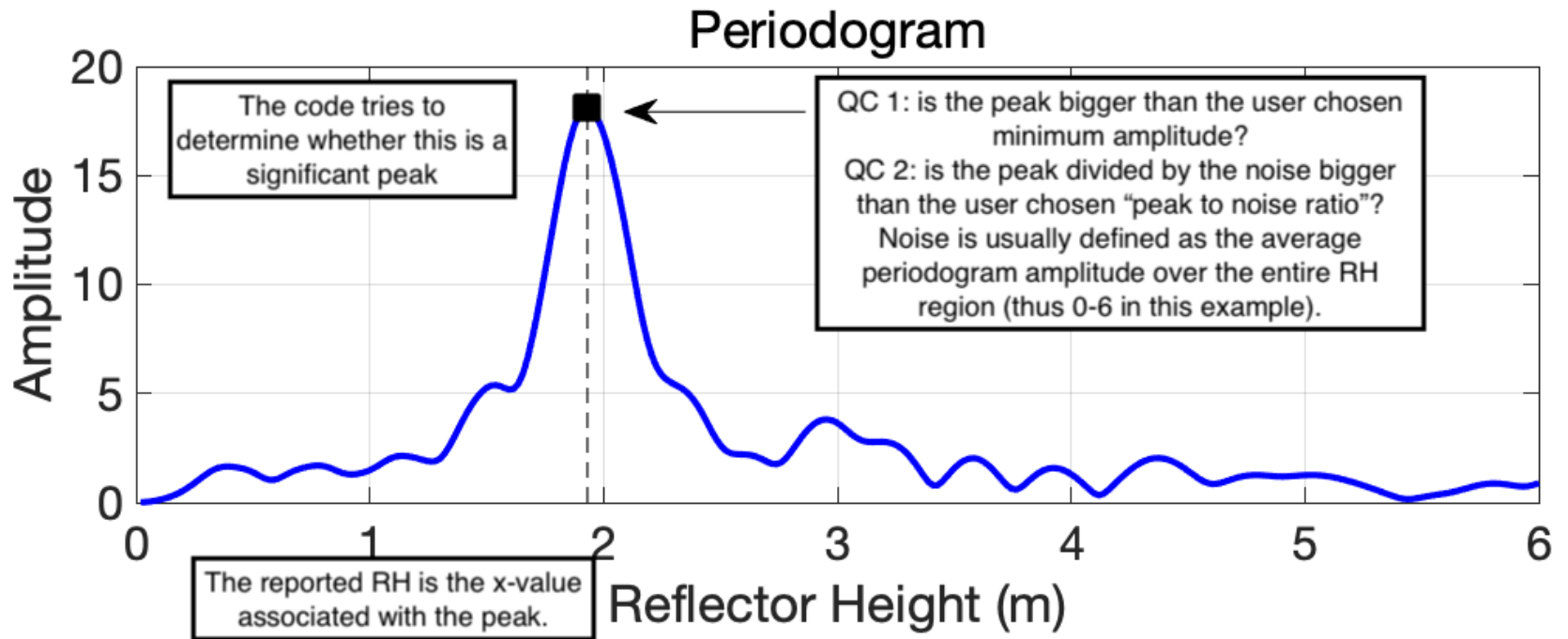
- REFL_CODE/input/abcd.json - instructions for gnssir analysis, refraction files
- REFL_CODE/YYYY/snr/abcd/abcdDDD0.YY.snr66 - SNR files
- REFL_CODE/YYYY/results/abcd/DDD.txt Lomb Scargle analysis
- REFL_CODE/YYYY/phase/abcd/DDD.txt phase analysis
- REFL_CODE/Files/ - various output files and plots will be placed here
- ORBITS/YYYY/nav/autoDDD0.YYn - GPS broadcast orbit file
- ORBITS/YYYY/sp3/ - sp3 files of orbits - these use names from the archives.

Step 2. quickLook

- For each geographic quadrant (northwest, southeast, etc), it checks all satellites to see if it can find
 - a rising or setting arc
 - computes a periodogram to find the dominant frequency (reflector height is on the x-axis)
 - defaults are
 - 5-25 degrees elevation angle
 - reflector heights < 8 meters
 - L1 GPS frequency
 - Quality Control metrics also displayed



Quality Control

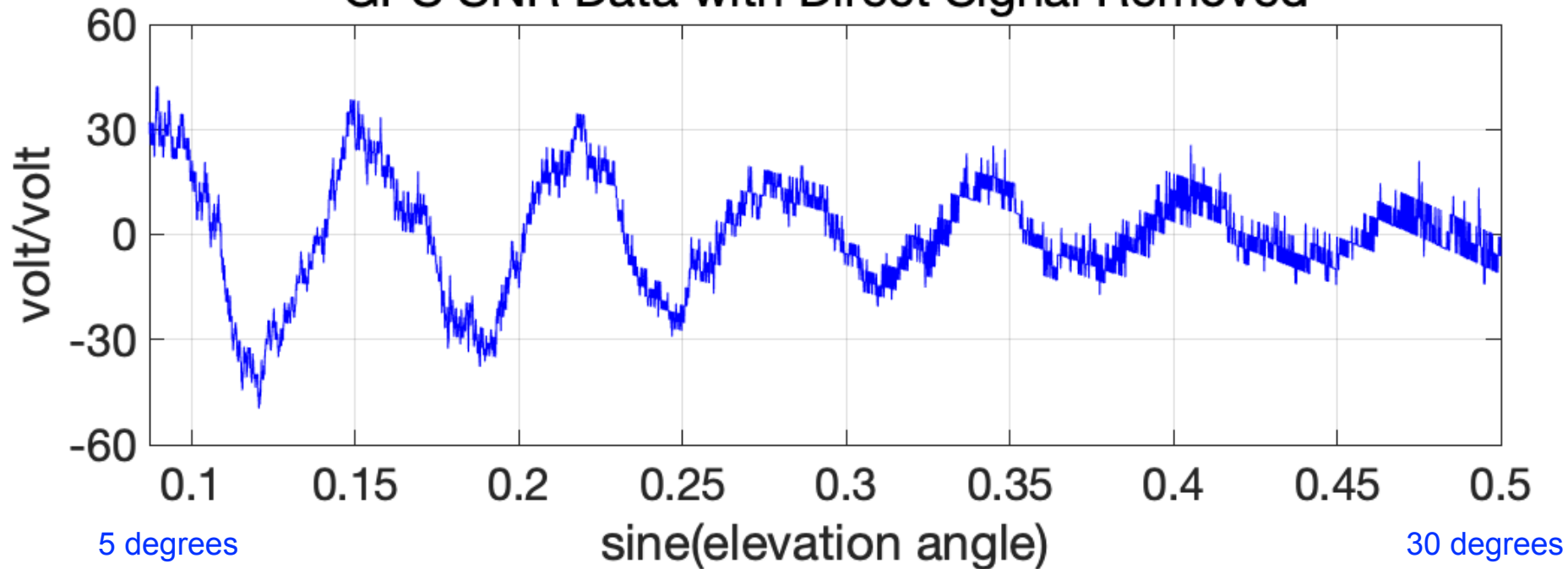


You can set the RH zone to start at a different value - but changing the end value requires thought.

Why isn't there one special number for quality control?

- Different surfaces and elevation angle masks lead to different peak amplitudes.
- Different receivers (and frequencies) have different quality SNR data (as we will see for L2P vs L2C).
- Peak to noise is a simple metric - it works pretty well, but misses complexity from things like double peaks.
- The “noise” metric should be computed in a consistent way appropriate for your surface.
- QC was easier for PBO H2O because all the stations were the same. With tides, it is a bit trickier. We are still working on making it easier for users.
- quickLook provides feedback to let you choose good QC values for your dataset.

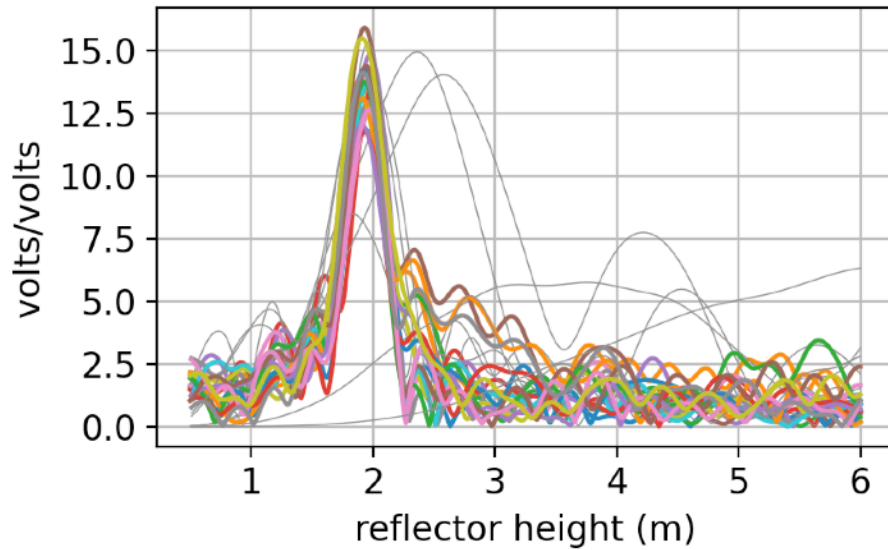
GPS SNR Data with Direct Signal Removed



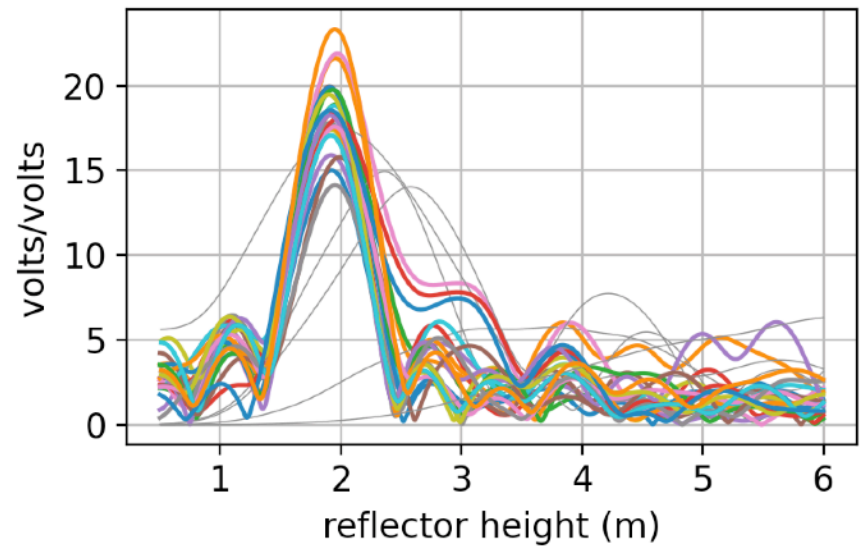
A spectral peak computed for this region will be smaller than the spectral peak using this region

Example periodograms for SNR data

Using elevation angles 5-25 degrees

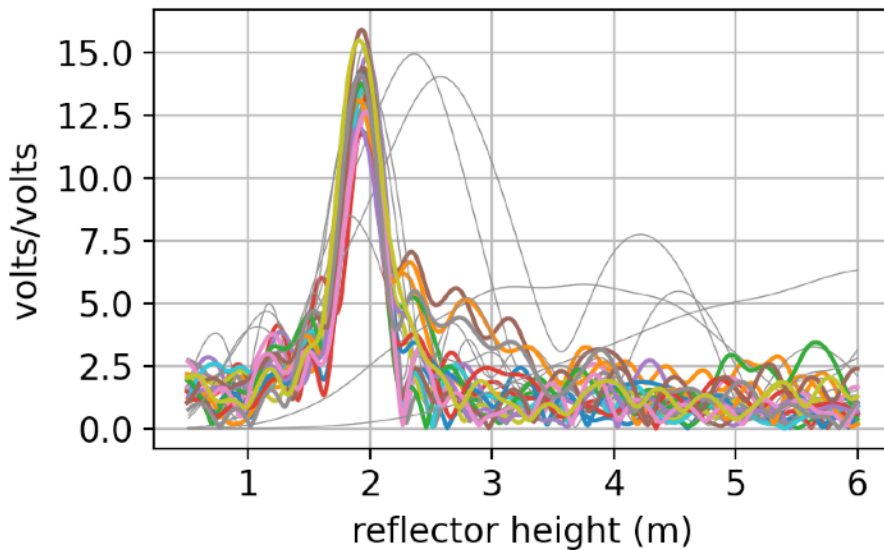


Using elevation angles 5-15 degrees



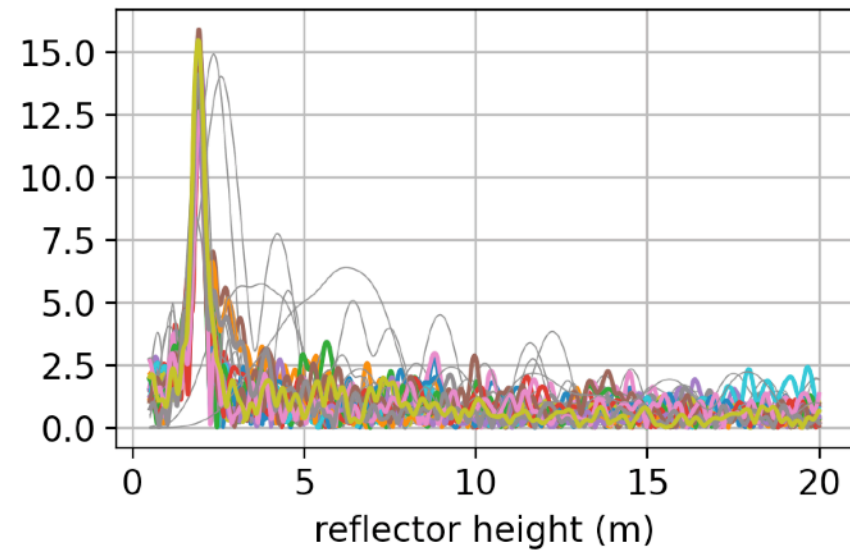
Peak to Noise Ratio

RH 0.5 to 6 meters (original default)



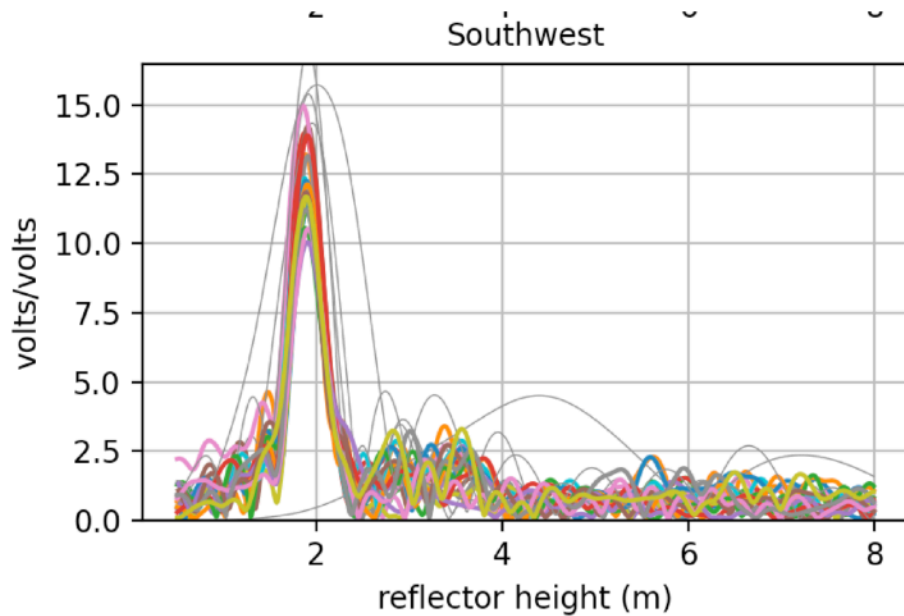
Background noise will be larger here,
so peak2noise of ~6

RH 0.5 to 20 meters



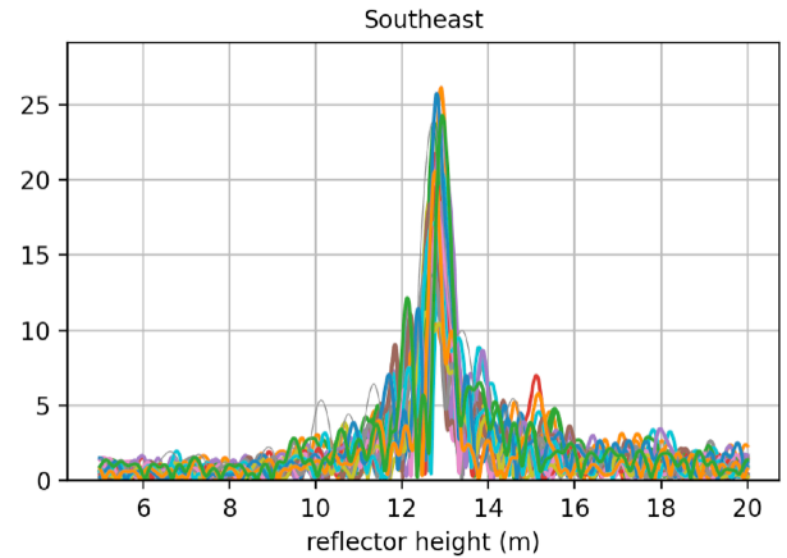
Background noise will be
smaller, so peak2noise of ~15

Peak amplitudes depend on the surface



here the x-axis tells you the reflector height value is ~1.9 meters.
The different colors are different satellites

PBO H2O site - bare soil



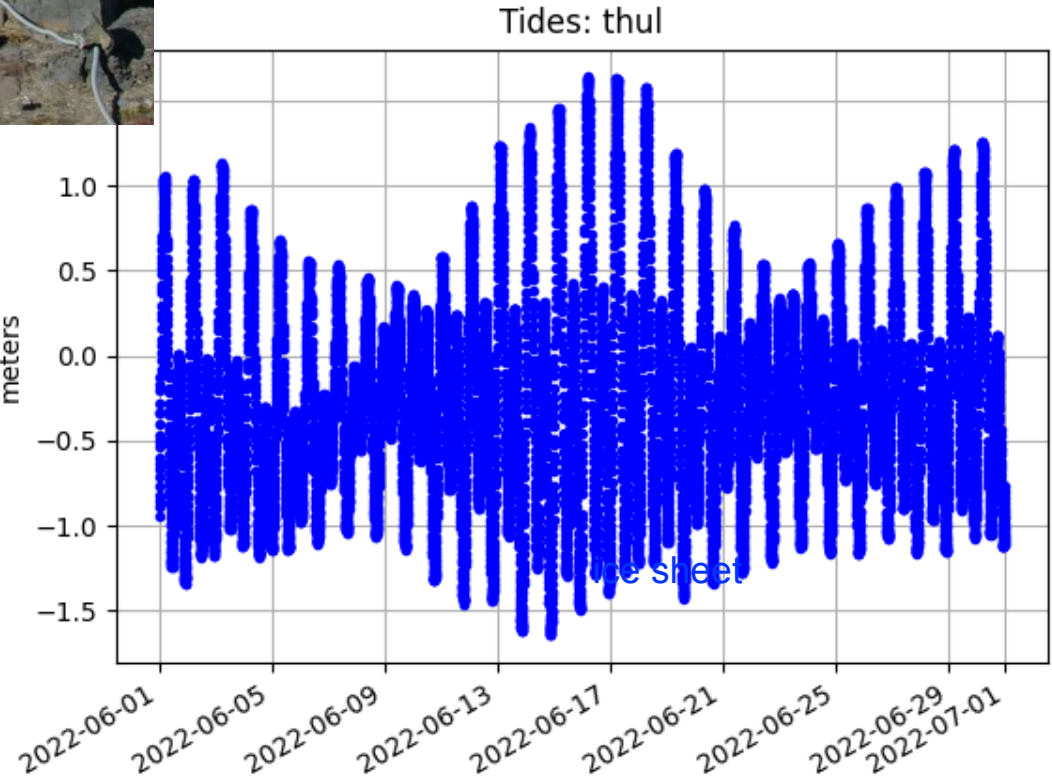
in this case the antenna is 13 meters
above the Greenland ice sheet

ice sheet

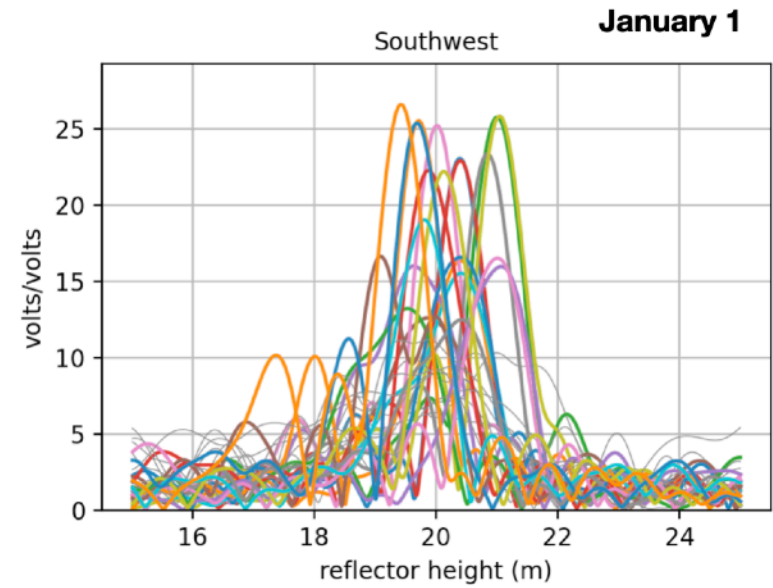
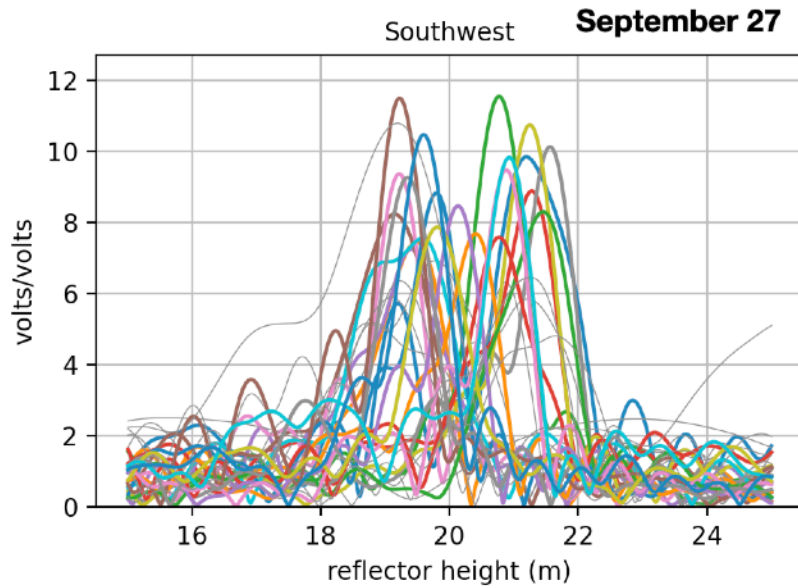




Surface Examples for another site in Greenland



How is September in northern Greenland different than January?



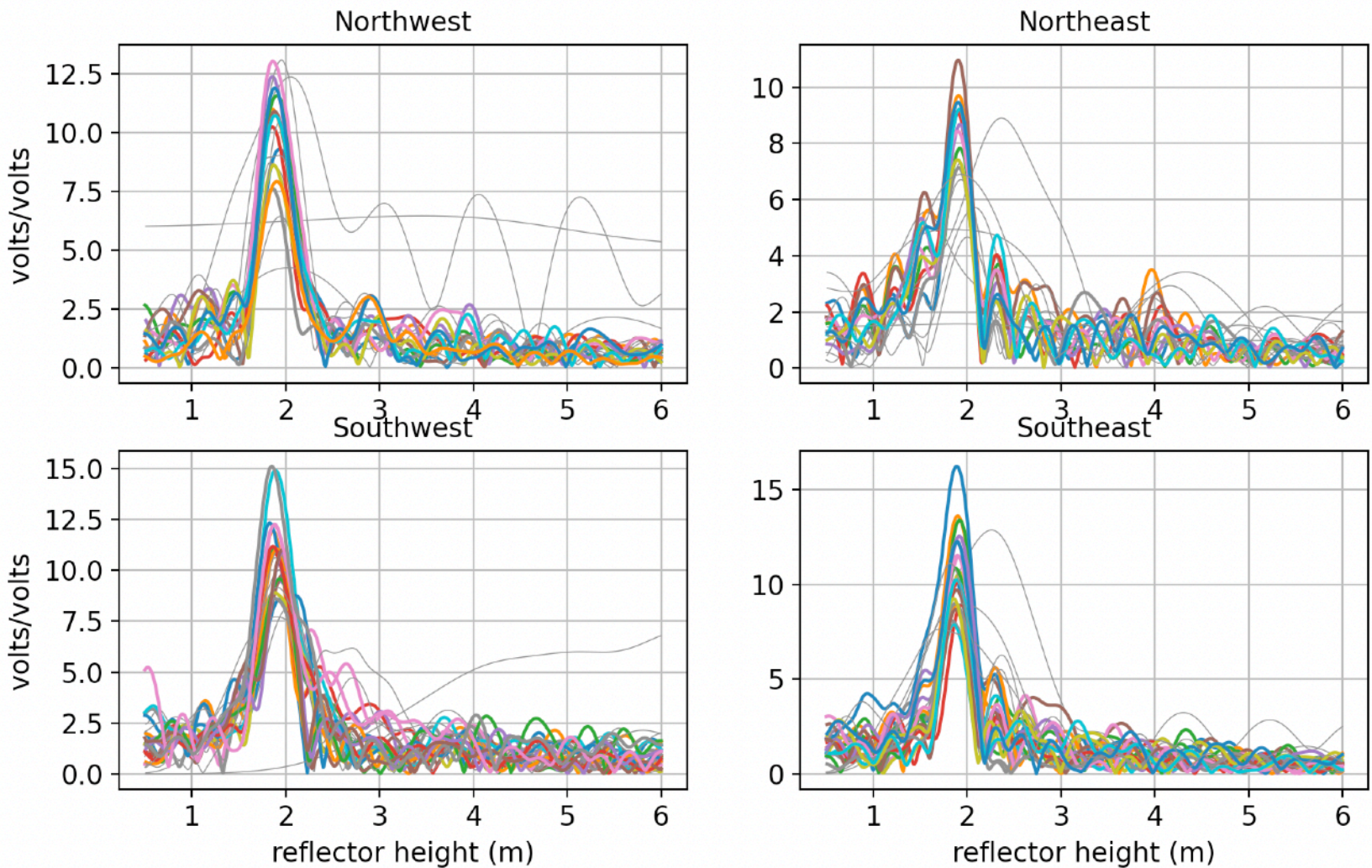
These are for a site that measures water reflections in Thule Greenland. Why are the peaks at different x-axis values?

run gnsstest for a simple site

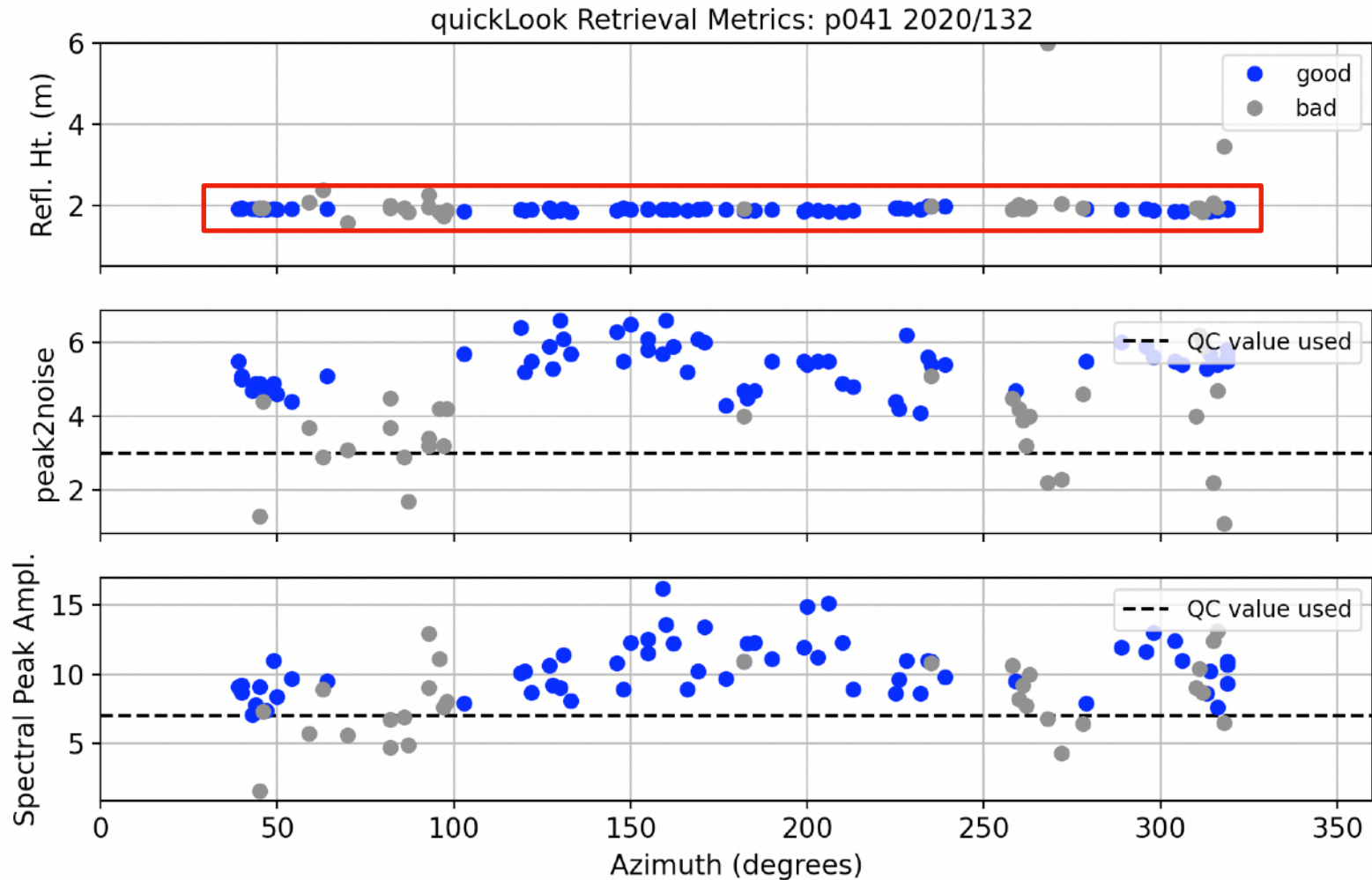


station
p041

GNSS-IR results: P041 Freq:GPS L1 Year/DOY:2020,132

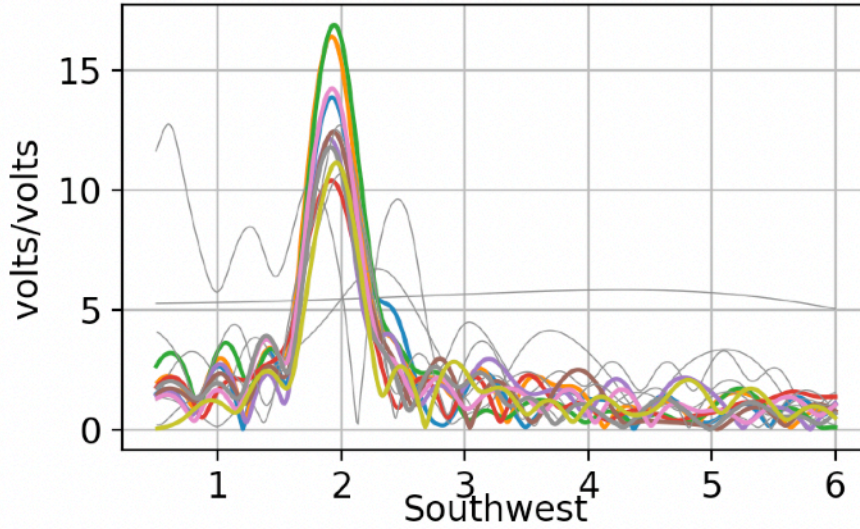


check quality control

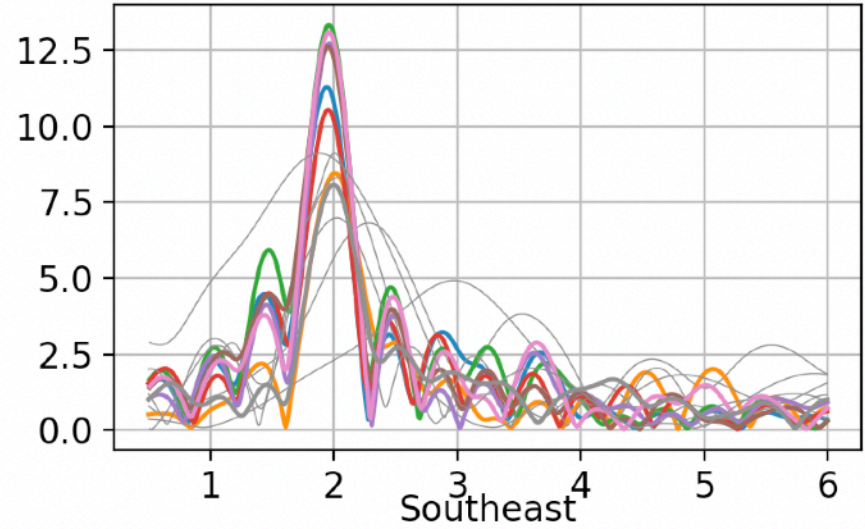


GNSS-IR results: P041 Freq:GPS L2C Year/DOY:2020,132

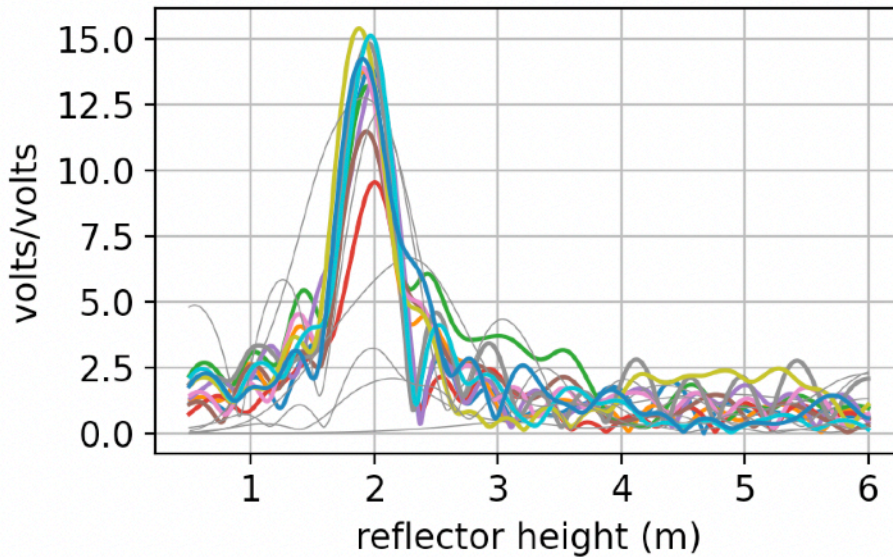
Northwest



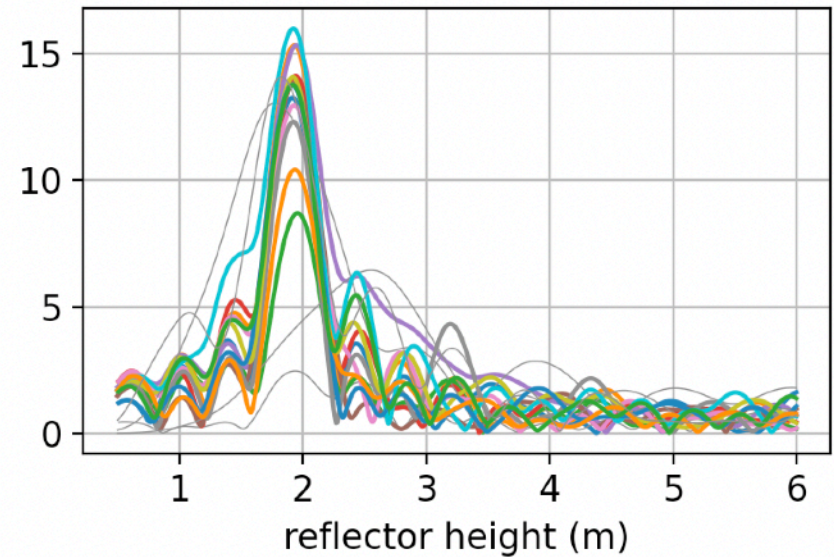
Northeast



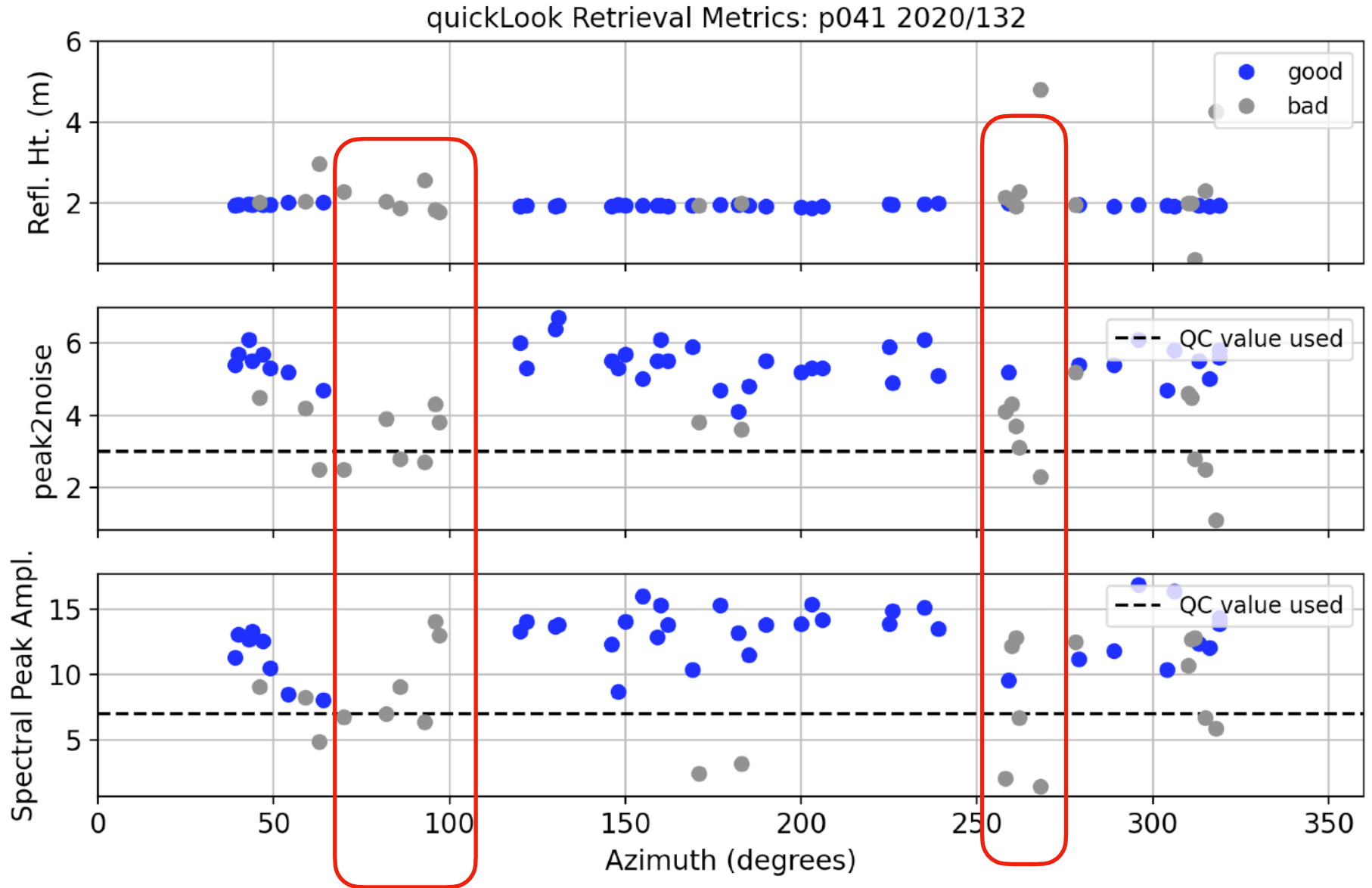
Southwest



Southeast

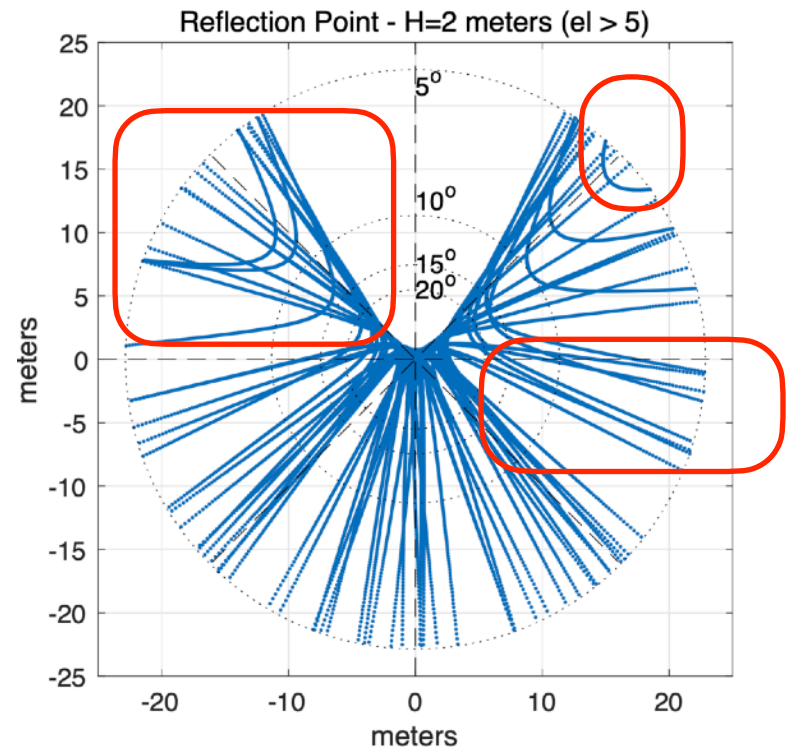


what causes these?



Remember ...

- For each geographic quadrant (northwest, southeast, etc), it checks all satellites to see if it can find
 - a rising or setting arc
 - computes a periodogram to find the dominant frequency (reflector height is on the x-axis)
 - defaults are
 - 5-25 degrees elevation angle
 - reflector heights < 8 meters
 - L1 GPS frequency
 - Quality Control metrics also displayed



But now I want to look at Glonass data. We originally used defaults, so only GPS data are in the SNR file. To get Glonass, Galileo, Beidou, you need to rerun rinex2snr

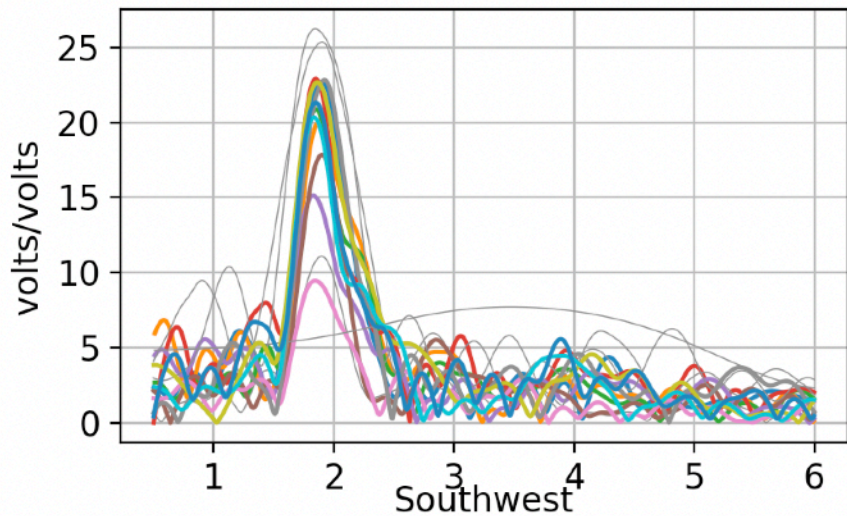
```
rinex2snr p041 2020 132 -orb gnss -overwrite True
```

after mid-2021

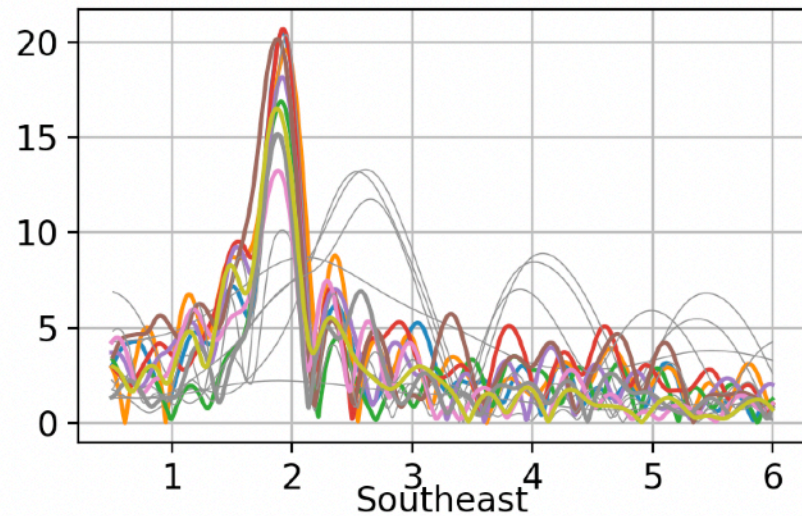
```
rinex2snr p041 2020 132 -orb rapid -overwrite True
```

GNSS-IR results: P041 Freq:Glonass L1 Year/DOY:2020,132

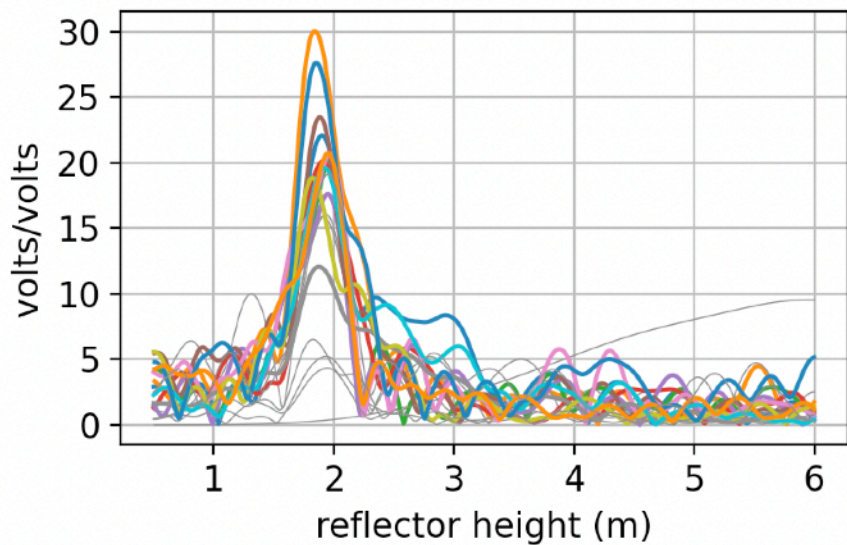
Northwest



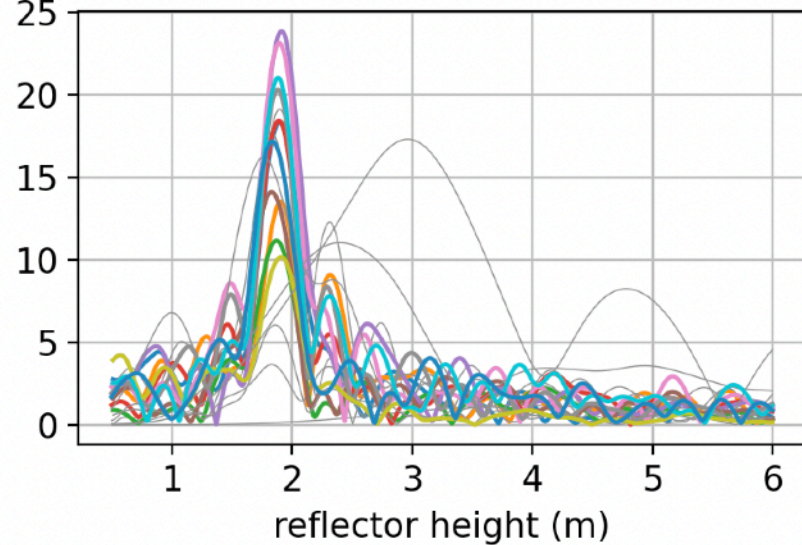
Northeast



Southwest



Southeast

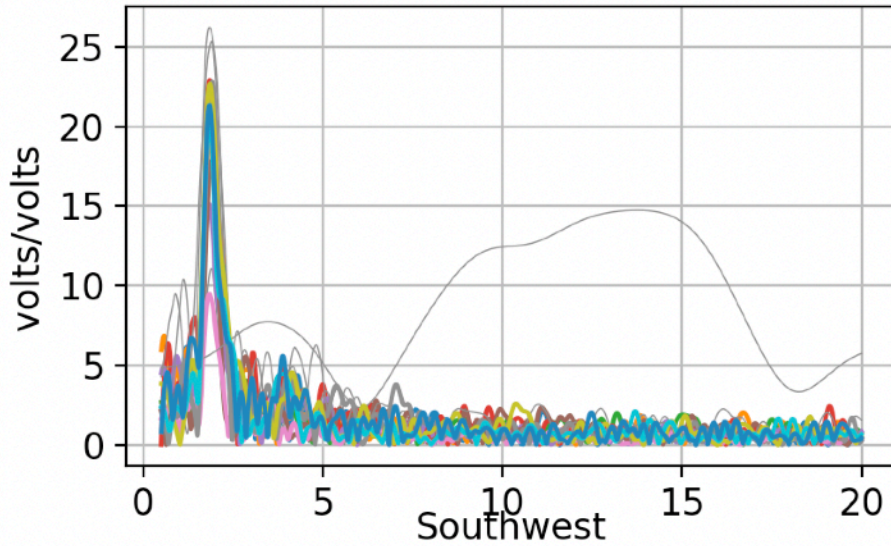


What about higher frequencies?

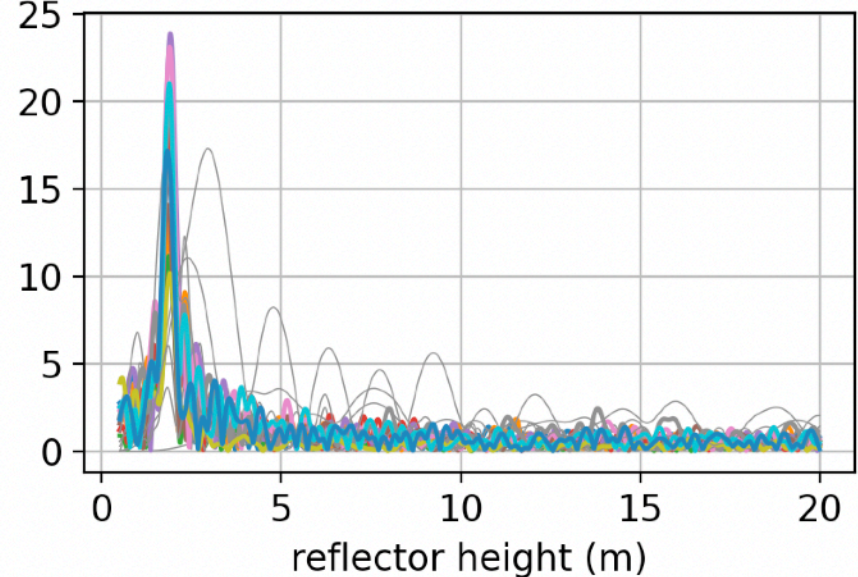
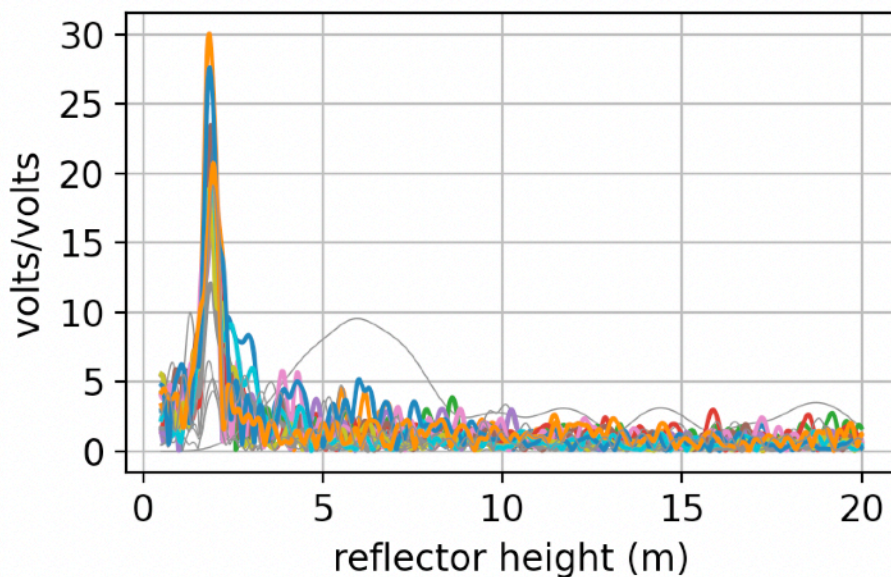
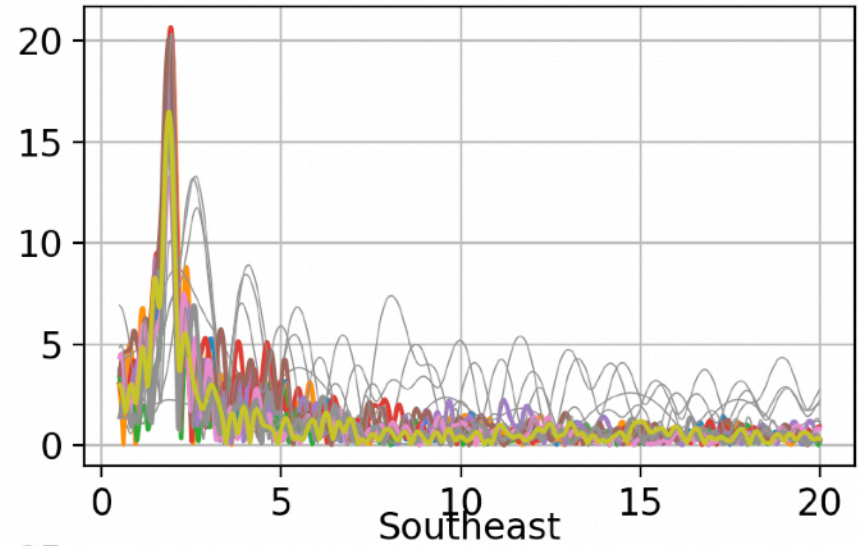
quickLook p041 2020 132 -fr 101 -h2 20

GNSS-IR results: P041 Freq:Glonass L1 Year/DOY:2020,132

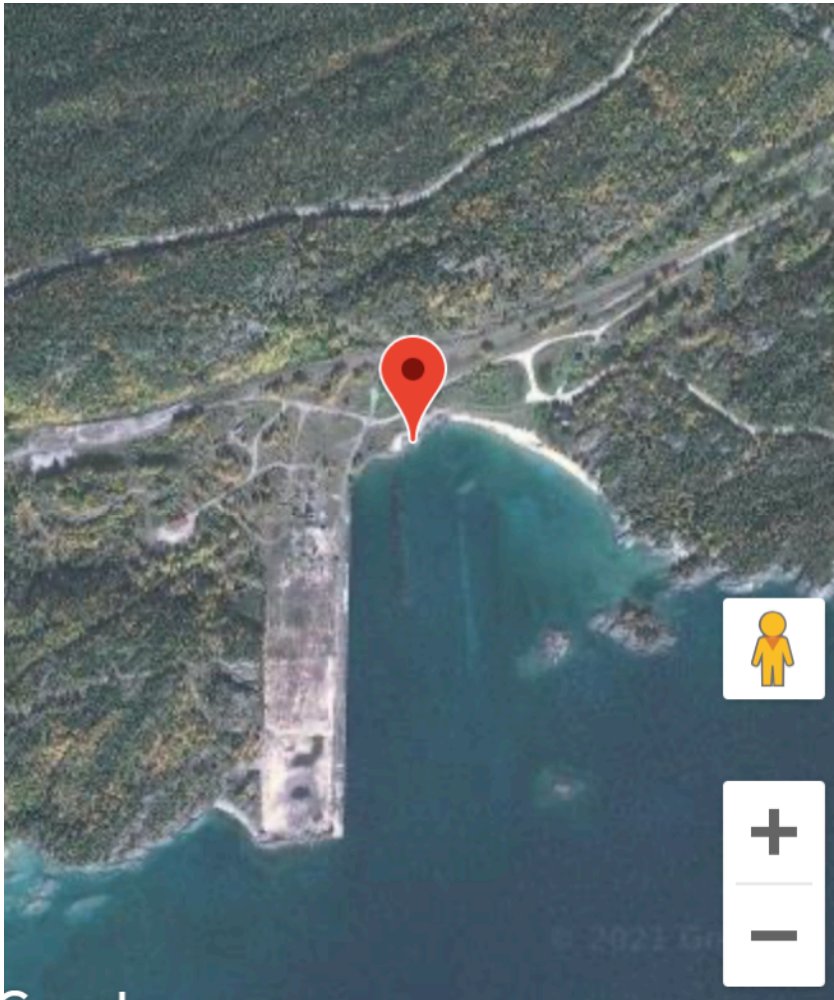
Northwest



Northeast



More typical situation for water reflections



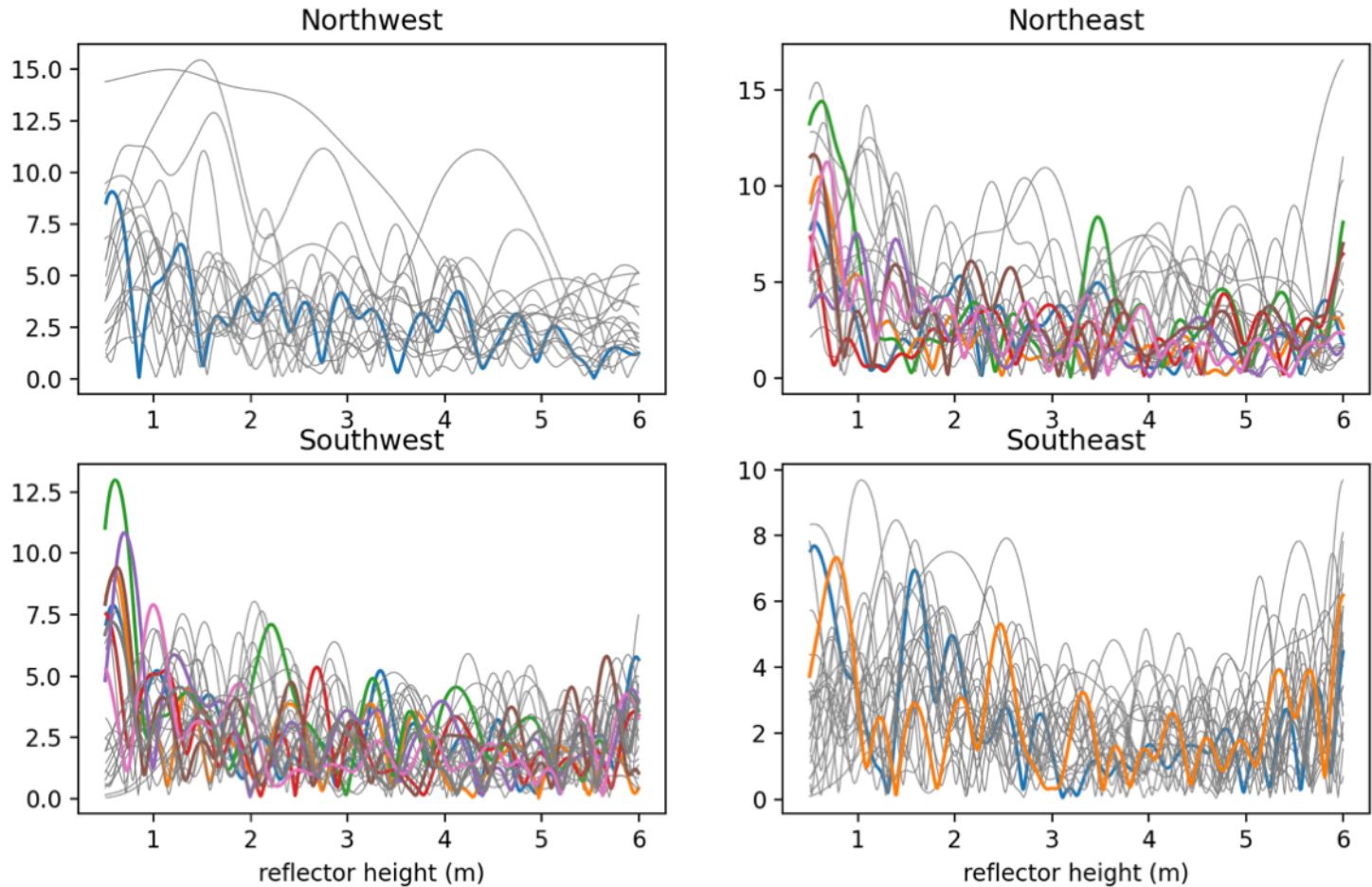
station mchn

<https://gnss-reflections.org/rzones>

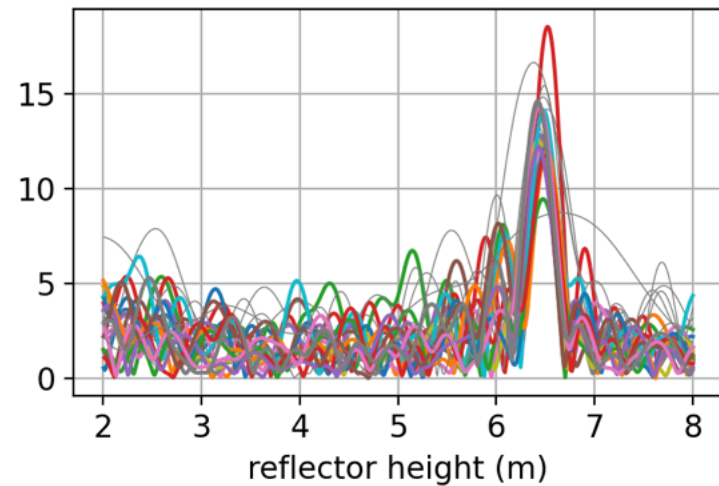
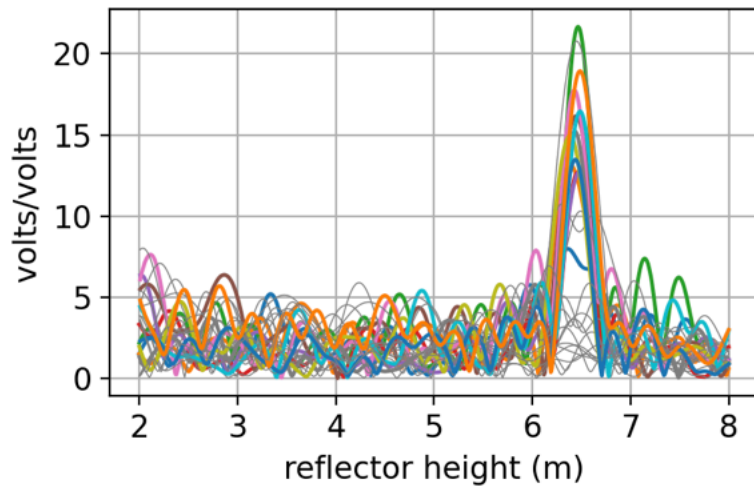
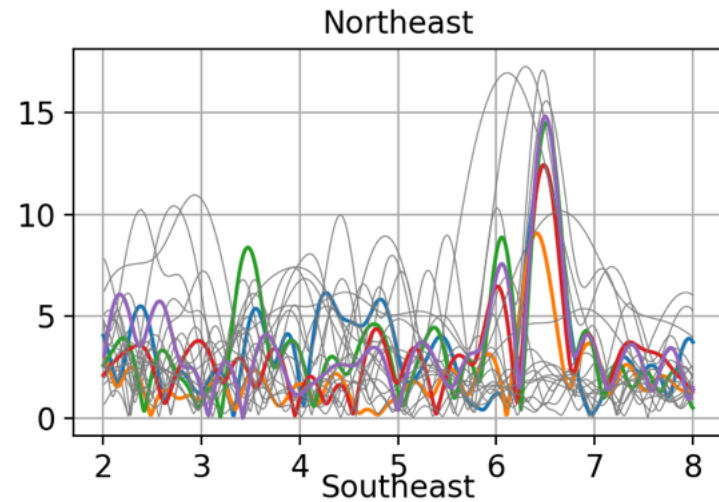
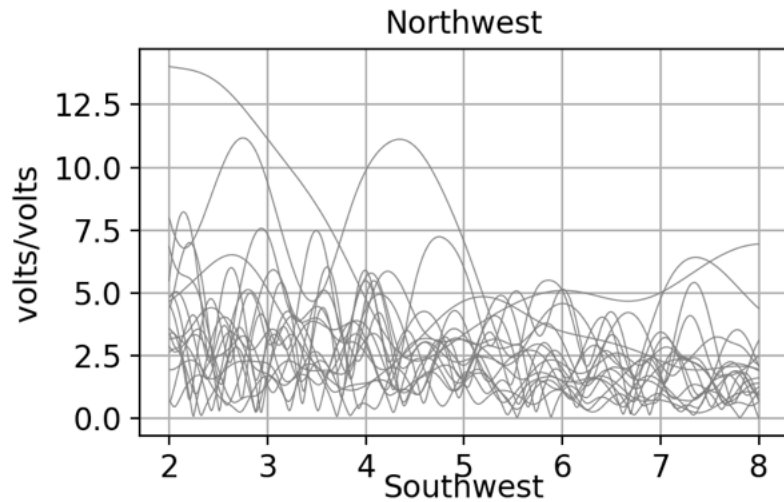
If you used the original defaults, 0.5-6 meters

quickLook mchn 2019 205

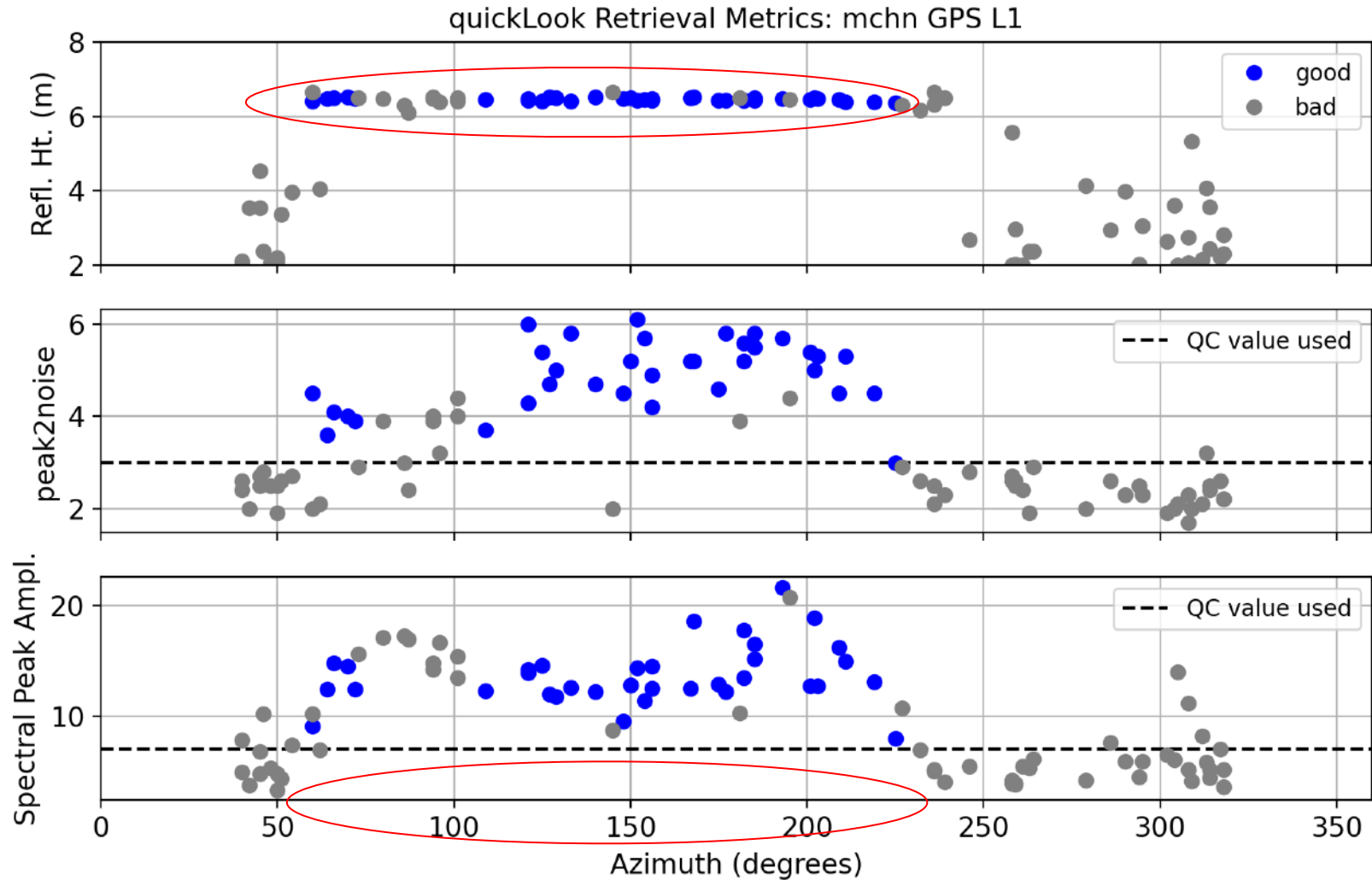
GNSS-IR results: MCHN Freq:1 2019/205



GNSS-IR results: MCHN Freq:GPS L1 Year/DOY:2019,205

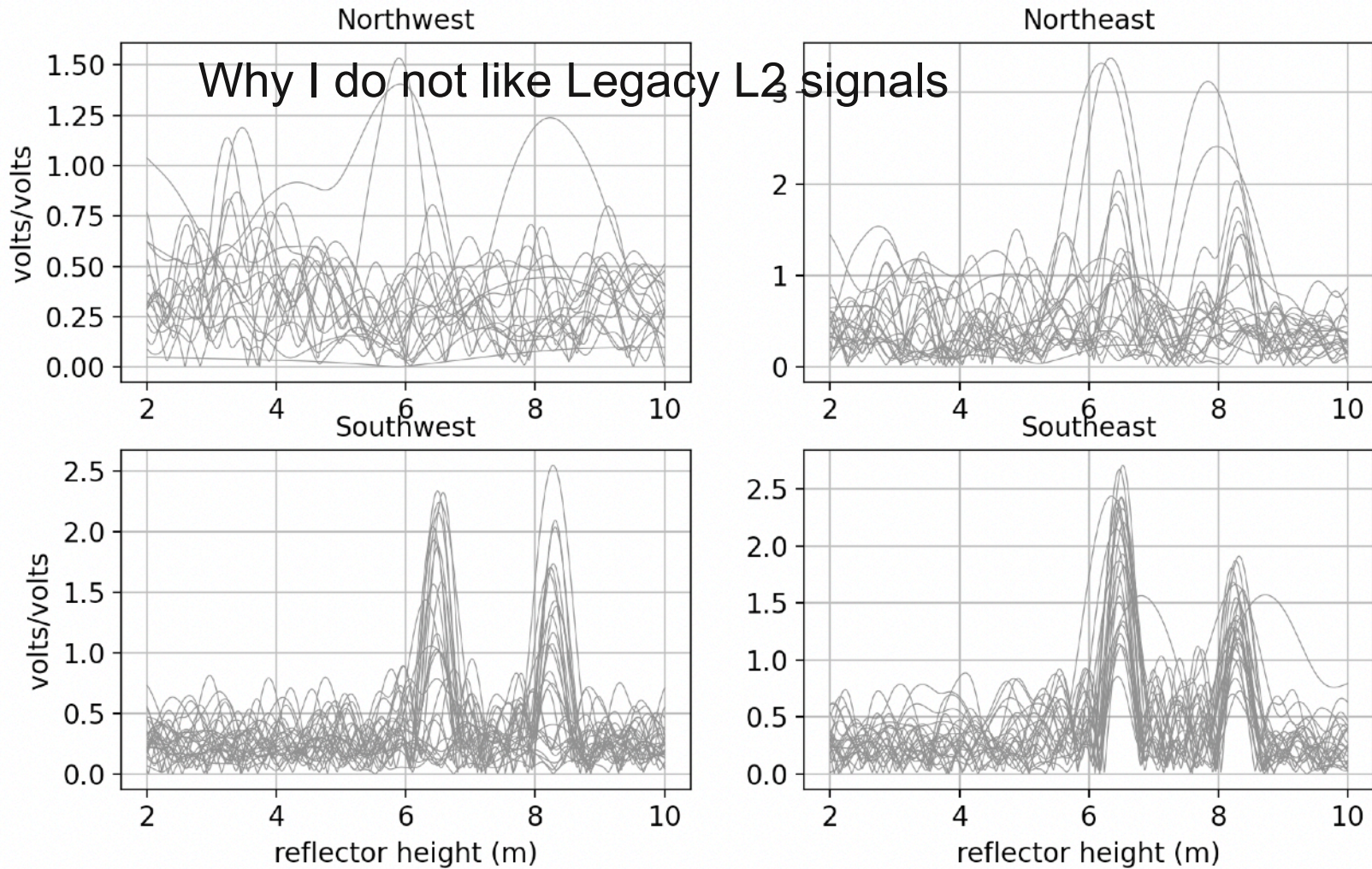


Azimuth mask



quickLook mchn 2019 205 -fr 2 -h1 2 -h2 10

GNSS-IR results: MCHN Freq:GPS L2 Year/DOY:2019,205



Once you have an analysis strategy, you need to save it and run *gnssir*

You need to save your strategy in a json file using `make_json_input`

For the easy case you can probably get away with the defaults. Originally I required you to input the latitude, longitude, and ellipsoidal height of a site:

```
make_json_input p041 39.9495 254.8057 1728.856
```

I later imported a database with ~20,000 GPS stations in it. If your site lives in that database, you can input zeroes instead of Lat,Long,Ht:

```
make_json_input p041 0 0 0
```

The latitude, longitude, and ellipsoidal height DO NOT need to be precise. They are only used to correct the refraction error.

The screen output tells you where this file goes - let's go look at it

```
{
  "station": "p041",
  "lat": 39.9495,
  "lon": 254.8057,
  "ht": 1728.856,
  "minH": 0.5,
  "maxH": 6.0,
  "e1": 5,
  "e2": 25,
  "NReg": [
    0.5,
    6.0
  ],
  "PkNoise": 2.7,
  "polyV": 4,
  "pele": [
    5,
    30
  ],
  "ediff": 2,
  "desiredP": 0.005,
  "azval": [
    0,
    90,
    90,
    180,
    180,
    270,
    270,
    360
  ],
  "freqs": [
    1,
    20,
    5
  ],
  "reqAmp": [
    6.0,
    6.0,
    6.0
  ],
  "refraction": true,
  "overwriteResults": true,
  "seekRinex": false,
  "wantCompression": false,
  "plt_screen": false,
  "onesat": null,
  "screenstats": false,
  "pltname": "p041_lsp.png",
  "delTmax": 75
}
```

Recovers all valid RH between 0.5 and 6 meters.
Elevation angles - 5 to 25

QC Noise region is 0.5 to 6 meters

QC Peak to noise is 2.7
This should be a bit higher for
soil moisture and snow.

pele: Direct signal model uses 5-30 degr elev. angles

edit discussed on next slide
RH precision is 0.005 meters

This says use all geographic quadrants (0-90,
90-180 and so on. You can edit these. Just be careful
not to forget that the last point in the array
doesn't have a comma

All GPS frequencies (1,20,5)

Amplitudes of the periodograms must be at least 6.
This is arbitrary! Choose the value you feel is best
for your dataset.

delTmax says arcs can't be longer than 75 minutes

What is ediff?

- It is a quality control metric that attempts to keep tracks about the same “length” in degrees of elevation angle.
- If we had perfect models, we would not need this.
- The default ediff value is 2.
- If you tell the code you want to use elevation angles from 5-25 degrees, it says an arc should at least be from 7-23 degrees
- If you say you want data from 5-15 degrees, it requires 7-13 degrees, and so on.
- You can turn it off by setting ediff to be a large number, like 5 degrees.
- Short arcs are generally creating those gray tracks on the periodogram plots.

Quality Control

- You can set the required periodogram amplitude to zero and rely entirely on peak2noise.
- peak2noise values vary by surface, because as we showed earlier, different surfaces have different amplitudes. And they absolutely depend on the noise region.
- When all the antennas were 2 meters above the ground, we used simple metrics like peak2noise to tell us whether to keep a track. It is a little more complicated now that we also measure tides.
- As Simon will share on Thursday, you can get in trouble for tides when you keep very long arcs, but for snow and soil moisture, these same arcs are fine, so it is complicated.

What if you only want to use L1 GPS data?

```
make_json_input p041 39.9495 254.8057 1728.856 -l1 True
```

Multi-GNSS?

```
make_json_input p041 39.9495 254.8057 1728.856 -allfreq True
```

```
make_json_input p041 39.9495 254.8057 1728.856 -frlist 1 20 5 101 102
```

What if you prefer a RH region of 2 to 10 meters?

```
make_json_input p041 39.9495 254.8057 1728.856 -h1 2 -h2 10
```

Your own azimuth regions

```
make_json_input p041 39.9495 254.8057 1728.856 -h1 2 -h2 10 -azlist 0 90 90 180
```

Planning to compare strategies

```
make_json_input p041 0 0 0 -h1 0 -h2 10 -e1 5 -e2 15 -extension E5_15
```

```
make_json_input p041 0 0 0 -h1 0 -h2 10 -e1 5 -e2 25 -extension E5_25
```


now run the gnssir code

```
(envnew) gnssrefl $ gnssir p041 2020 132
```

```
refraction file exists
```

```
Results are written to: /Users/kristine/Documents/Research/2020/results/p041//132.txt
```

```
reading from this snr file /Users/kristine/Documents/Research/2020/snr/p041/p0411320.20.snr66
```

By design, very little comes back to the screen - though you can change that in the json if you like. The result file looks something like this:

```
% gnssrefl, https://github.com/kristinmlarson
% Phase Center corrections have NOT been applied
% year, doy, RH, sat,UTCtime, Azim, Amp,  emin0,  emax0, NumbOf, freq, rise, EdotF, PkNoise  DelT      MJD  refr-appl
% (1) (2) (3) (4) (5) (6) (7) (8) (9) (10) (11) (12) (13) (14) (15) (16) (17)
%                               m          hrs      deg      v/v      deg      deg  values      hrs          min          1 is yes
2020 132  1.922   1 19.327  39.81   9.14   5.08  24.94  232   1 -1 -0.76368  5.52  57.75 58980.805289 1
2020 132  1.880   1 16.250 120.70 10.22   5.06  24.99  261   1  1  0.84247  5.19  65.00 58980.677083 1
2020 132  1.950   1 10.715 225.52   8.71   5.06  24.93  238   1 -1 -0.75834  4.47  59.25 58980.446435 1
2020 132  1.945   1  7.025 319.33   9.23   5.04  24.95  209   1  1  0.67509  5.81  52.00 58980.292708 1
2020 132  1.875   2 19.560 199.68 11.87   5.05  24.97  184   1  1  0.58630  5.45  45.75 58980.815012 1
2020 132  1.900   3 13.329 190.91 11.09   5.08  24.91  199   1 -1 -0.63093  5.46  49.50 58980.555382 1
2020 132  1.862   3  8.454 306.92 10.99   5.08  24.99  217   1  1  0.68909  5.31  54.00 58980.352257 1
2020 132  1.912   4 15.450 160.29 13.56   5.10  24.93  187   1 -1 -0.59764  6.54  46.50 58980.643738 1
2020 132  1.917   5  2.763  54.68   9.75   5.01  24.98  221   1 -1 -0.69707  4.44  55.00 58980.115104 1
2020 132  1.937   5 21.802 171.40 13.38   5.02  25.00  200   1  1  0.62900  6.03  49.75 58980.908414 1
2020 132  1.892   6 18.658 182.31 10.81   5.06  24.99  187   1  1  0.59269  4.65  46.50 58980.777431 1
2020 132  1.962   7 18.485 148.35   9.30   5.01  24.99  188   1 -1 -0.59823  5.21  46.75 58980.770220 1
2020 132  1.962   7 12.225 259.41   9.48   5.06  24.92  261   1  1  0.84626  4.72  65.00 58980.509363 1
2020 132  1.932   8 17.160  44.52   7.82   5.01  24.96  210   1 -1 -0.66810  4.91  52.25 58980.715012 1
2020 132  1.900   8 12.942 150.41 12.25   5.08  24.94  215   1  1  0.68181  6.52  53.50 58980.539225 1
2020 132  1.895   9 16.510 177.95   9.63   5.09  24.97  192   1 -1 -0.60948  4.28  47.75 58980.687928 1
2020 132  1.920   9 11.223 296.69 11.53   5.07  24.95  236   1  1  0.74895  5.86  58.75 58980.467616 1
2020 132  1.922  10  9.221 131.71 11.39   5.06  24.95  207   1 -1 -0.66678  6.05  51.50 58980.384190 1
2020 132  1.980  10  2.788 239.66   9.93   5.09  24.94  229   1  1  0.74302  5.45  57.00 58980.116146 1
```


What if you think you should have better results from gnssir?

- try `-screenstats T`
- It will try to tell you why your track was rejected, was the amplitude too small? Was the arc too long? Was the arc too short?

Other useful things to know

- We do not currently allow tracks to cross midnight. We know how to keep them, but it is not implemented yet.
- We also have plans to change how we find rising and setting arcs - but it should not impact users (except you will hopefully notice a few more arcs).

Can you change the strategy in *gnssir* on the command line?

- Sort of, but that defeats the purpose of having a strategy
- If you want to compare strategies, use the `-extension` flag. That way your results will be put into different directories.

Multiple days and years

```
gnssir p041 2020 1 -doy_end 366
```

```
gnssir p041 2015 1 -doy_end 366 -year_end 2017
```

What do you do next?

It depends on your environmental target.

For things that *do not* have significant sub-daily behavior, we use **daily_avg**. This will be discussed further in the context of snow (Day 2) and lakes (Day 3).

For tides, we use the **subdaily** module. This will be discussed further on day 3.

reflection zone utilities

<https://gnss-reflections.org/rzones>

https://gnssrefl.readthedocs.io/en/latest/api/gnssrefl.refl_zones_cl.html

other utilities you might not realize
are there

https://gnsrefl.readthedocs.io/en/latest/pages/quick_recall.html

web app if we have time