Science_Flight_20151118

November 18, 2015

In this report, we’ll review the science flight of the ER-2 starting on 11/18/2015. This was a ~7-h flight that overflew post-frontal convective showers off the coast of the Olympic Peninsula. First let’s import all the needed modules and ingest and process the raw data.

```
In [1]: from __future__ import print_function
        import numpy as np
        import matplotlib.pyplot as plt
        import datetime as dt
        import os
        import glob
        import pyart
        import rawpyampr
        import pyampr
        from awot.graph.common import create_basemap
        from awot.graph.flight_level import FlightLevel
        from pyart_tools import list_nexrad_files, read_nexrad_aws
        %matplotlib inline
```

```
In [2]: import warnings
        warnings.filterwarnings(’ignore’)
        def delete_file(fname):
            try:
                os.remove(fname)
            except:
                pass
```

```
In [3]: datadir = ’./’
        files = glob.glob(datadir + ’*.dat’)
        print(files)
        fname = os.path.basename(files[0])[:-4]
```

```
[’./AMPR-20151118-174617.dat’, ’./AMPR-20151118-221444.dat’]
```

```
In [4]: payload = rawpyampr.ampr_payload.AMPR_Payload(files)
        l1file = fname + ’_L1.nc’
        l2file = fname + ’_L2.nc’
        delete_file(l1file)
        payload.writeLevel1B(l1file)
        L1B = rawpyampr.ampr_level1b.AMPR_QC(l1file)
        delete_file(l2file)
        L1B.writeLevel2B(l2file)
```

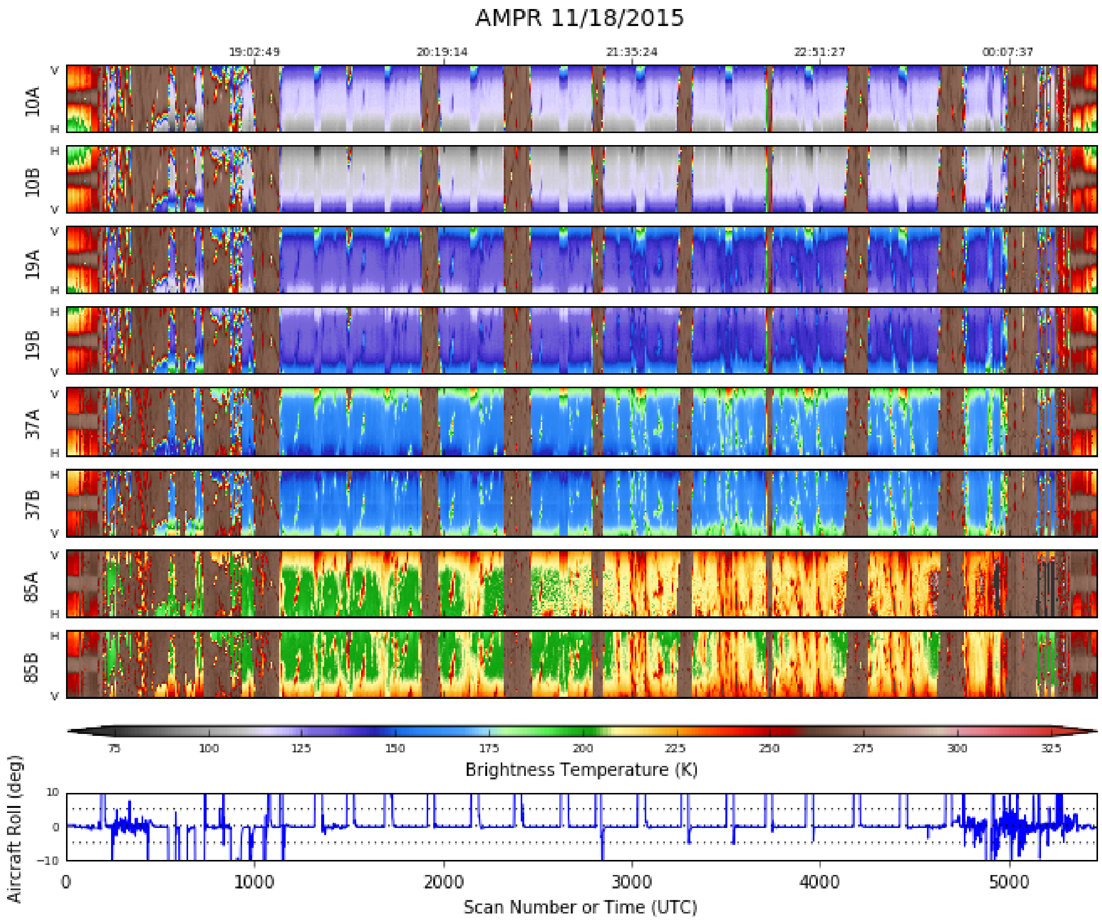
```
All of file: ./AMPR-20151118-174617.dat Read Successfully
End of data stream reached
All of file: ./AMPR-20151118-221444.dat Read Successfully
```

```
End of data stream reached
Interpreting Navigation Records as: IWG1
 No navigation file found
Navigating pixels using internal recording of nav data.
Number points to converge: 4
Writing to output file:  AMPR-20151118-174617_L1.nc
Found Navigation Data!
Writing to output file: AMPR-20151118-174617_L2.nc
File containing water fraction not on path
```

Now we are ready to read in and display the L2 geolocated brightness temperatures.

```
In [5]: data = pyampr.AmprTb(l2file)
        data.plot_ampr_channels()
```

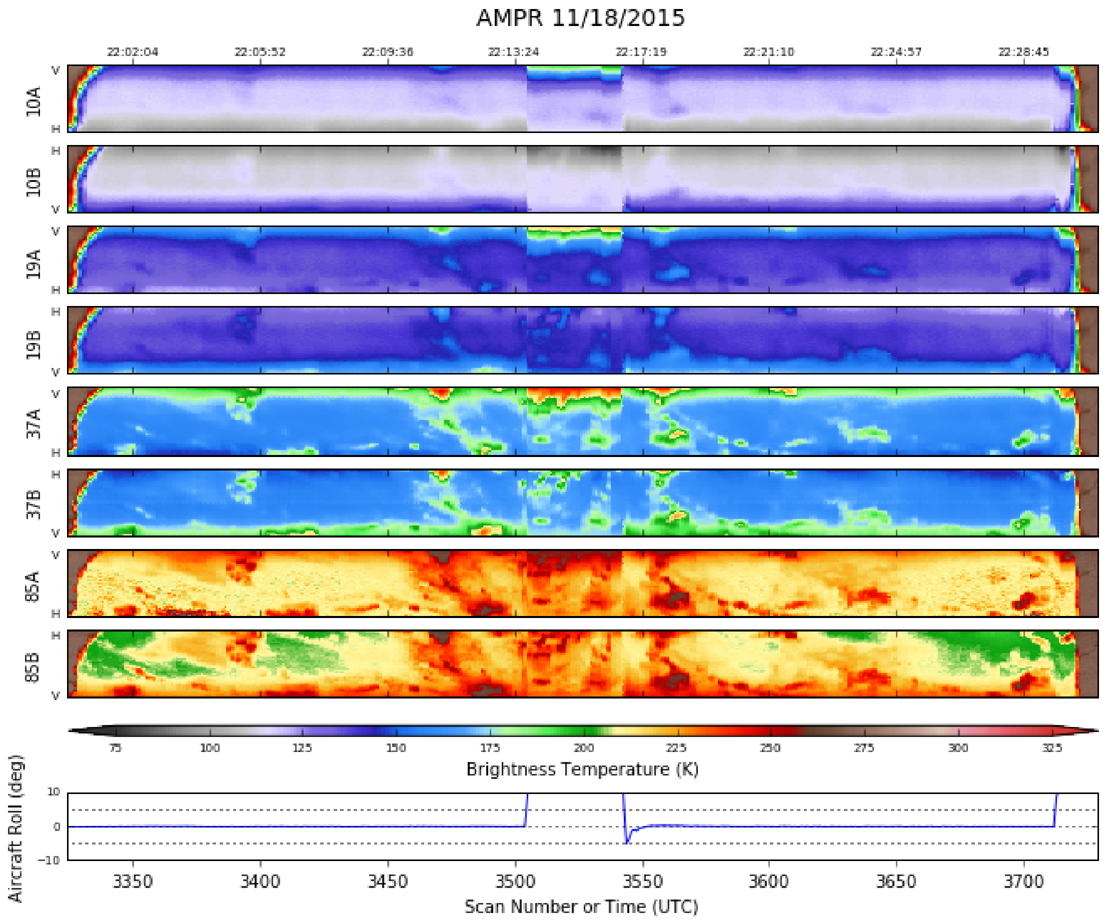
```
********************
read_ampr_tb_level2b(): Reading AMPR-20151118-174617_L2.nc
Assuming IPHEX data structure.
Change to proper project if incorrect, otherwise errors will occur.
Currently available field projects: IPHEX, MC3E, TC4, TCSP, JAX90, COARE,
CAMEX1, CAMEX2, CAMEX3, CAMEX4, TRMMLBA, KWAJEX, TEFLUNA, FIRE3ACE, CAPE
Default: project = 'IPHEX'
Found Navigation Data!
(5466,)
********************


********************
plot_ampr_channels():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
********************
```



Overall, this looks pretty good. There are some questionable TBs on the 85 GHz (A) channel starting about 3 h into the flight, which we'll explore later. First, let's focus on the science. The ER-2 performed several orbits over the showers, and some of the best runs occurred between 2100 and 2330 UTC. Let's look at one particular orbit.

```
In [6]: data.plot_ampr_channels(timerange=['22:00:00', '22:31:00'])
```

```
********************
plot_ampr_channels():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
********************
```



Let's see what the local radar sees around this time, and compare it to the geolocated brightness temperatures during the outbound leg. We'll start with AMPR.

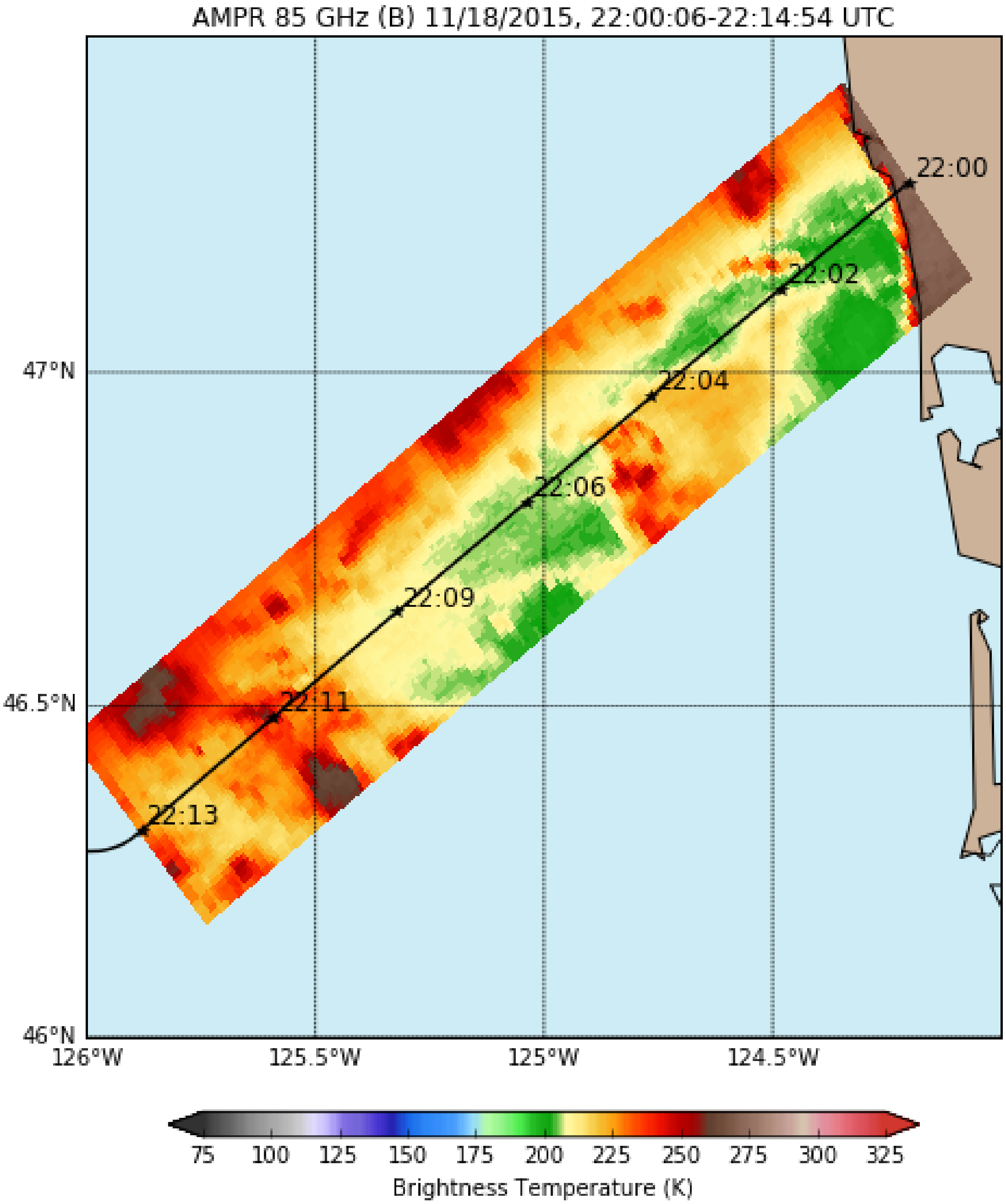
```
In [8]: # Import the ER-2 nav into AWOT (https://github.com/nguy/AWOT)
        # This simplifies plotting the track with time stamps
        def read_simple_er2_into_awot(AmprTB):
            citdic = {}
            varlist = ['latitude', 'longitude', 'altitude', 'time']
            for var in varlist:
                citdic[var] = {}
            citdic['latitude']['data'] = AmprTB.Aircraft_Nav['GPS Latitude']
            citdic['longitude']['data'] = AmprTB.Aircraft_Nav['GPS Longitude']
            citdic['altitude']['data'] = AmprTB.Aircraft_Nav['GPS Altitude']
            ampr_datetime = []
            for et in AmprTB.Epoch_Time:
                ampr_datetime.append(dt.datetime(1970, 1, 1) + dt.timedelta(seconds=np.float(et)))
            citdic['time']['data'] = ampr_datetime
            for var in varlist:
                citdic[var]['data'] = np.ma.masked_array(citdic[var]['data'], mask=False)
            citdic['flight_number'] = 'None'
            citdic['project'] = 'OLYMPEX'
```

```
    citdic['platform'] = 'NASA ER-2'
    citdic['Uwind'] = None
    citdic['Vwind'] = None
    return citdic

flight = read_simple_er2_into_awot(data)
```

```
In [17]: lonrange = [-126, -124]
         latrange = [46, 47.5]
         stuff = data.plot_ampr_track(
             '85b', timerange=['22:00:00', '22:15:00'], latrange=latrange,
             maneuver=False, return_flag=True, lonrange=lonrange,
             meridians=0.5, parallels=0.5, resolution='i')
         f2 = FlightLevel(flight, basemap=stuff[2])
         start = '2015-11-18 22:00:00'
         end = '2015-11-18 22:15:00'
         f2.plot_trackmap(min_altitude=50., lw=2.5, start_time=start,
                          end_time=end, ax=stuff[1])
         f2.time_stamps(start_time=start, end_time=end,
                        labelspacing=30, ax=stuff[1])
         stuff[2].fillcontinents(color='#CCB299', lake_color='#CEECF5',
                                 ax=stuff[1], zorder=0)
         stuff[2].drawmapboundary(fill_color='#CEECF5', ax=stuff[1])
```

```
********************
plot_ampr_track():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
Filtering out significant aircraft maneuvers
********************
```

```
Out[17]: <matplotlib.patches.Rectangle at 0x128d88668>
```



Now let's plot the radar with the flight track overlay

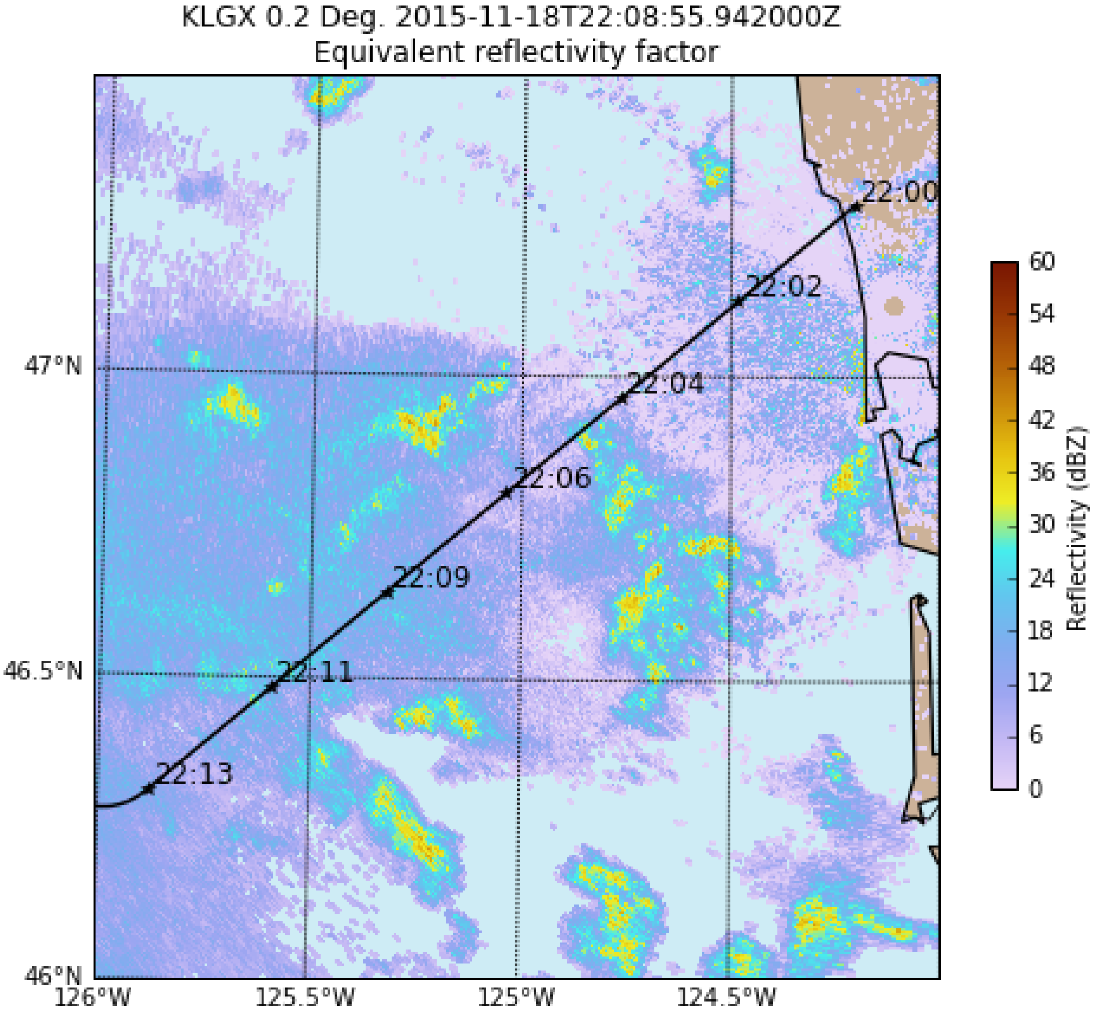
```
In [13]: # print(list_nexrad_files(year='2015', month='11', day='18', station='KLGX'))
         radar = read_nexrad_aws('KLGX20151118_220836_V06.gz')
```

```
https://noaa-nexrad-level2.s3.amazonaws.com/2015/11/18/KLGX/KLGX20151118_220836_V06.gz
```

```
In [23]: fig = plt.figure(figsize=(8, 8))
         display = pyart.graph.RadarMapDisplay(radar)
```

```
display.plot_ppi_map(
    'REF', vmin=0, vmax=60, cmap='pyart_LangRainbow12', colorbar_flag=False,
    resolution='i', min_lon=lonrange[0], max_lon=lonrange[1],
    min_lat=latrange[0], max_lat=latrange[1], projection='lcc',
    lon_lines=np.arange(-130, -110, 0.5), lat_lines=np.arange(30, 50, 0.5))
f2 = FlightLevel(flight, basemap=display.basemap)
f2.plot_trackmap(min_altitude=50., lw=2.5, start_time=start,
                 end_time=end)
f2.time_stamps(start_time=start, end_time=end,
               labelspacing=30)
display.basemap.fillcontinents(color='#CCB299', lake_color='#CEECF5',
                               ax=plt.gca(), zorder=0)
display.basemap.drawmapboundary(fill_color='#CEECF5', ax=plt.gca())
plt.colorbar(display.plots[0], label='Reflectivity (dBZ)', shrink=0.5)
```

```
Out[23]: <matplotlib.colorbar.Colorbar at 0x130888a90>
```

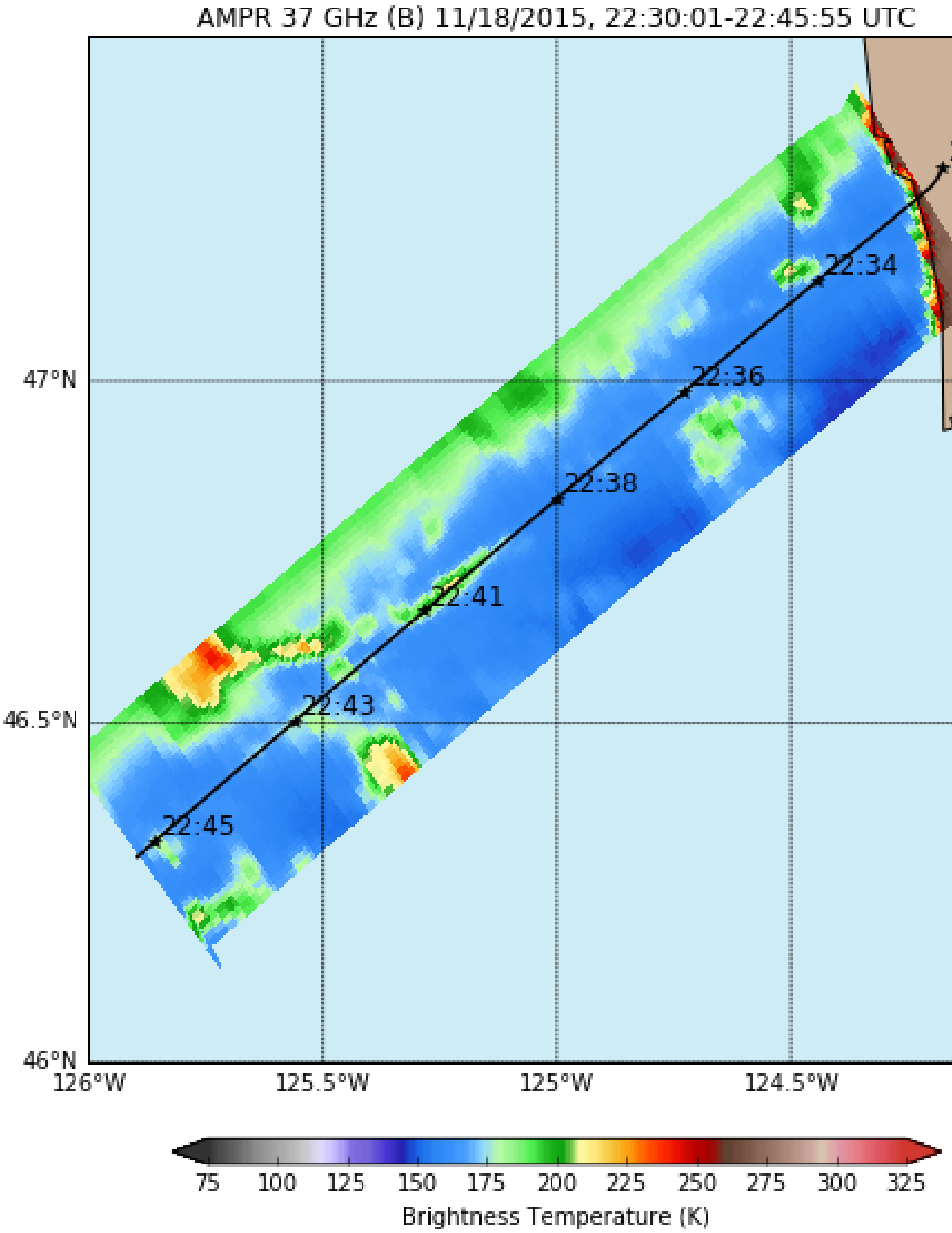


Overall, good correspondence between the highest 85-GHz brightness temperatures (likely corresponding to microwave emissions from liquid water) and the locations of the heaviest radar-inferred precipitation. Let's check out one other orbit, this time at 37 GHz.

```
In [29]: stuff = data.plot_ampr_track(
             '37b', timerange=['22:30:00', '22:46:00'], latrange=latrange,
             maneuver=False, return_flag=True, lonrange=lonrange,
             meridians=0.5, parallels=0.5, resolution='i')
         f2 = FlightLevel(flight, basemap=stuff[2])
         start = '2015-11-18 22:32:00'
         end = '2015-11-18 22:46:00'
         f2.plot_trackmap(min_altitude=50., lw=2.5, start_time=start,
                          end_time=end, ax=stuff[1])
         f2.time_stamps(start_time=start, end_time=end,
                        labelspacing=30, ax=stuff[1])
         stuff[2].fillcontinents(color='#CCB299', lake_color='#CEECF5',
                                 ax=stuff[1], zorder=0)
         stuff[2].drawmapboundary(fill_color='#CEECF5', ax=stuff[1])
```

```
********************
plot_ampr_track():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
Filtering out significant aircraft maneuvers
********************
```

```
Out[29]: <matplotlib.patches.Rectangle at 0x136b74748>
```

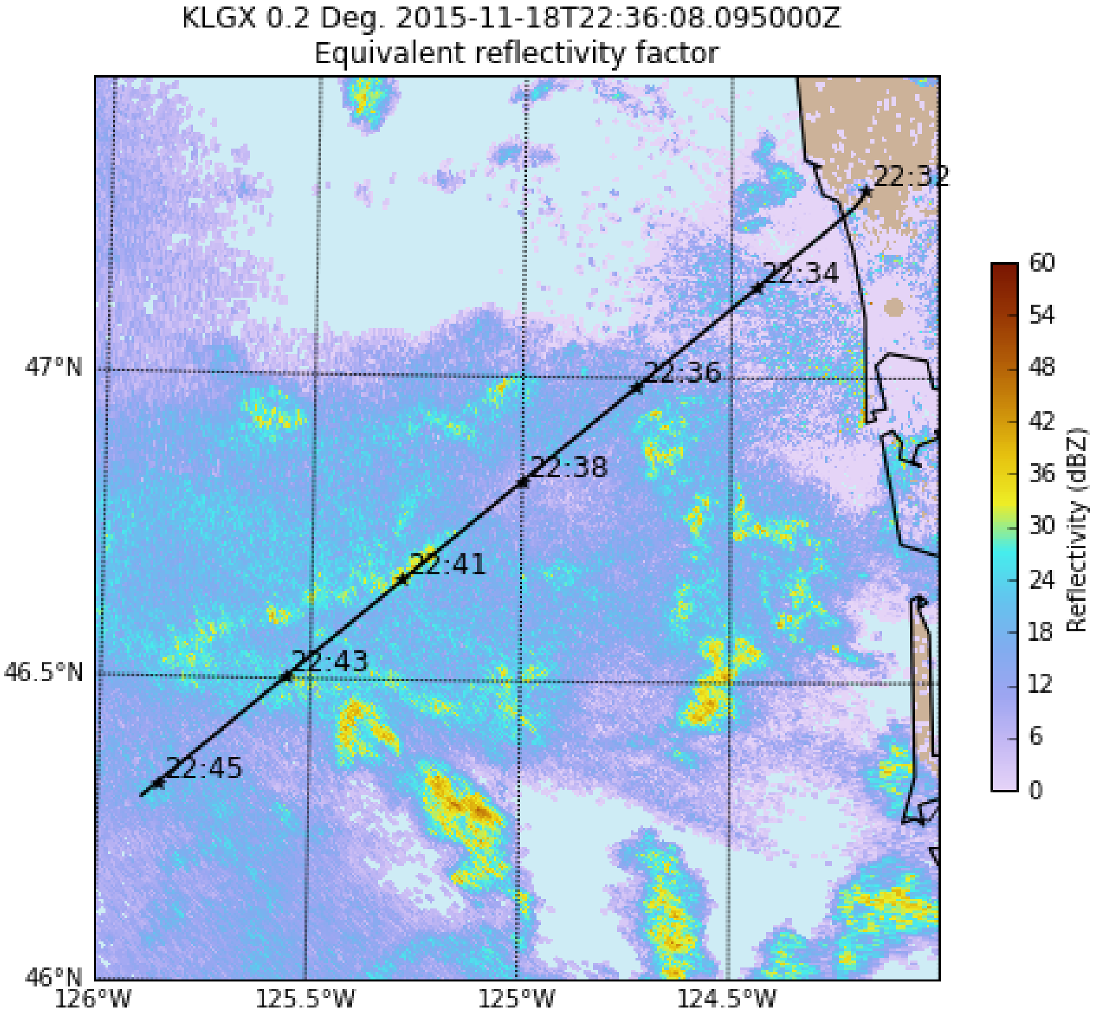


```
In [31]: # print(list_nexrad_files(year='2015', month='11', day='18', station='KLGX'))
         radar = read_nexrad_aws('KLGX20151118_223548_V06.gz')
```

```
https://noaa-nexrad-level2.s3.amazonaws.com/2015/11/18/KLGX/KLGX20151118_223548_V06.gz
```

```
In [33]: fig = plt.figure(figsize=(8, 8))
         display = pyart.graph.RadarMapDisplay(radar)
         display.plot_ppi_map(
             'REF', vmin=0, vmax=60, cmap='pyart_LangRainbow12', colorbar_flag=False,
             resolution='i', min_lon=lonrange[0], max_lon=lonrange[1],
             min_lat=latrange[0], max_lat=latrange[1], projection='lcc',
             lon_lines=np.arange(-130, -110, 0.5), lat_lines=np.arange(30, 50, 0.5))
         f2 = FlightLevel(flight, basemap=display.basemap)
         f2.plot_trackmap(min_altitude=50., lw=2.5, start_time=start,
                          end_time=end)
         f2.time_stamps(start_time=start, end_time=end,
                        labelspacing=30)
         display.basemap.fillcontinents(color='#CCB299', lake_color='#CEECF5',
                                        ax=plt.gca(), zorder=0)
         display.basemap.drawmapboundary(fill_color='#CEECF5', ax=plt.gca())
         plt.colorbar(display.plots[0], label='Reflectivity (dBZ)', shrink=0.5)
```

```
Out[33]: <matplotlib.colorbar.Colorbar at 0x13692acf8>
```

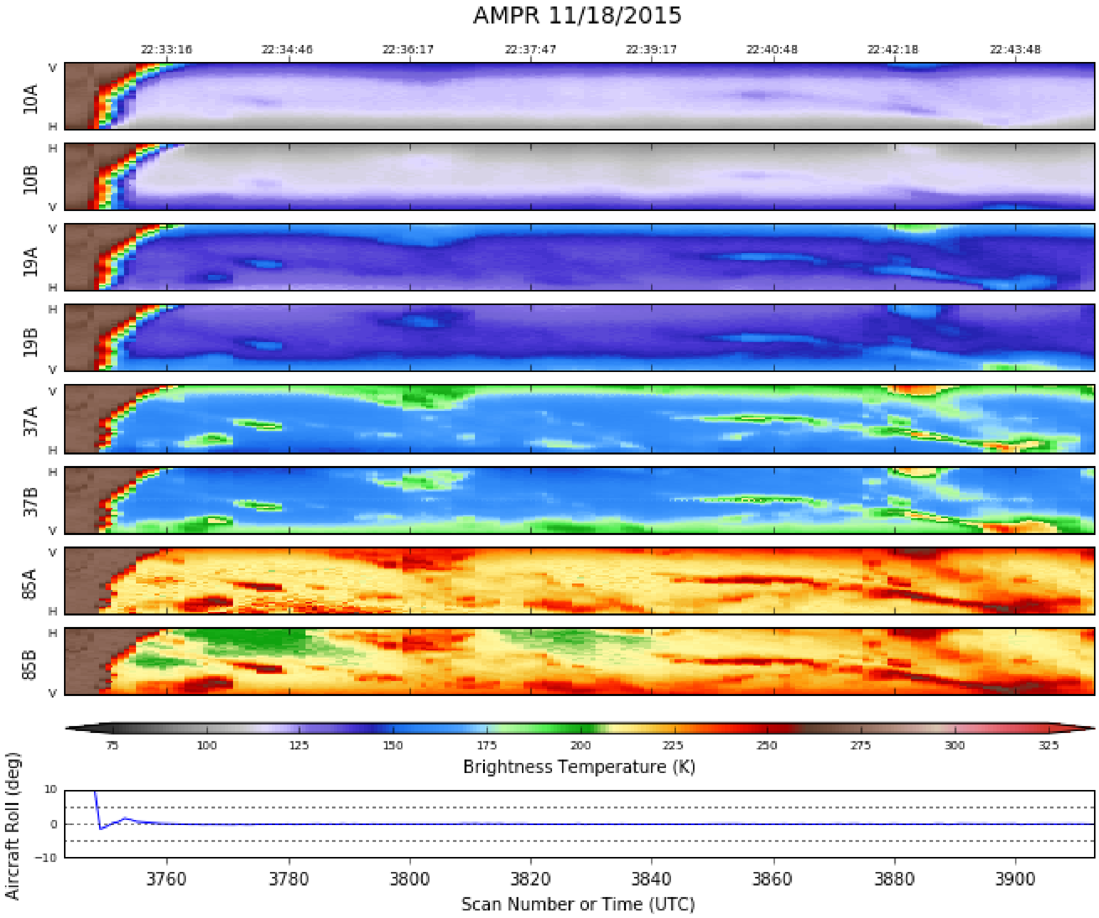


The along-track overflight of a short line at 2241 UTC should be really interesting - a good candidate for combined airborne radar/radiometer retrievals. It has signatures at all AMPR

frequencies, and the multi-frequency radars should have had a good view as well. See the zoomed-in strip chart below.

```
In [34]: data.plot_ampr_channels(timerange=[’22:32:00’, ’22:45:00’])
```

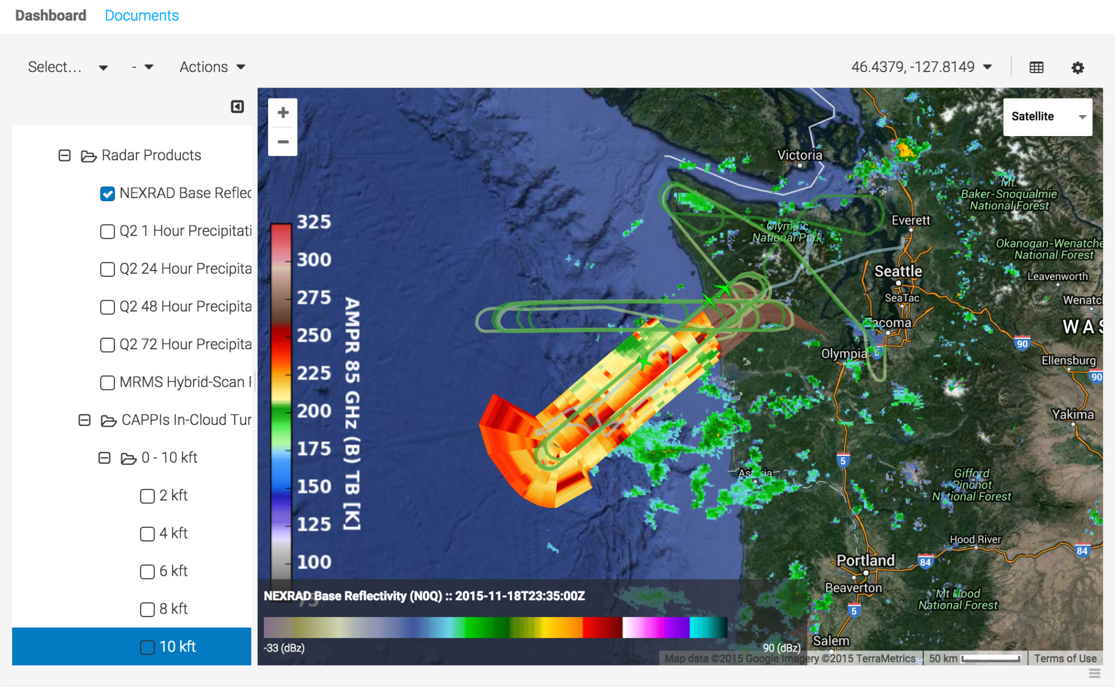
```
********************
plot_ampr_channels():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
********************
```



Some great news was that the Inmarsat finally worked and we were able to post realtime AMPR imagery to the NASA MTS. See an example image below.

```
In [46]: from IPython.display import Image
         Image(’/Users/tjlang/Documents/AMPR/OLYMPEX/20151118/ampr_mts.png’)
```

```
Out[46]:
```

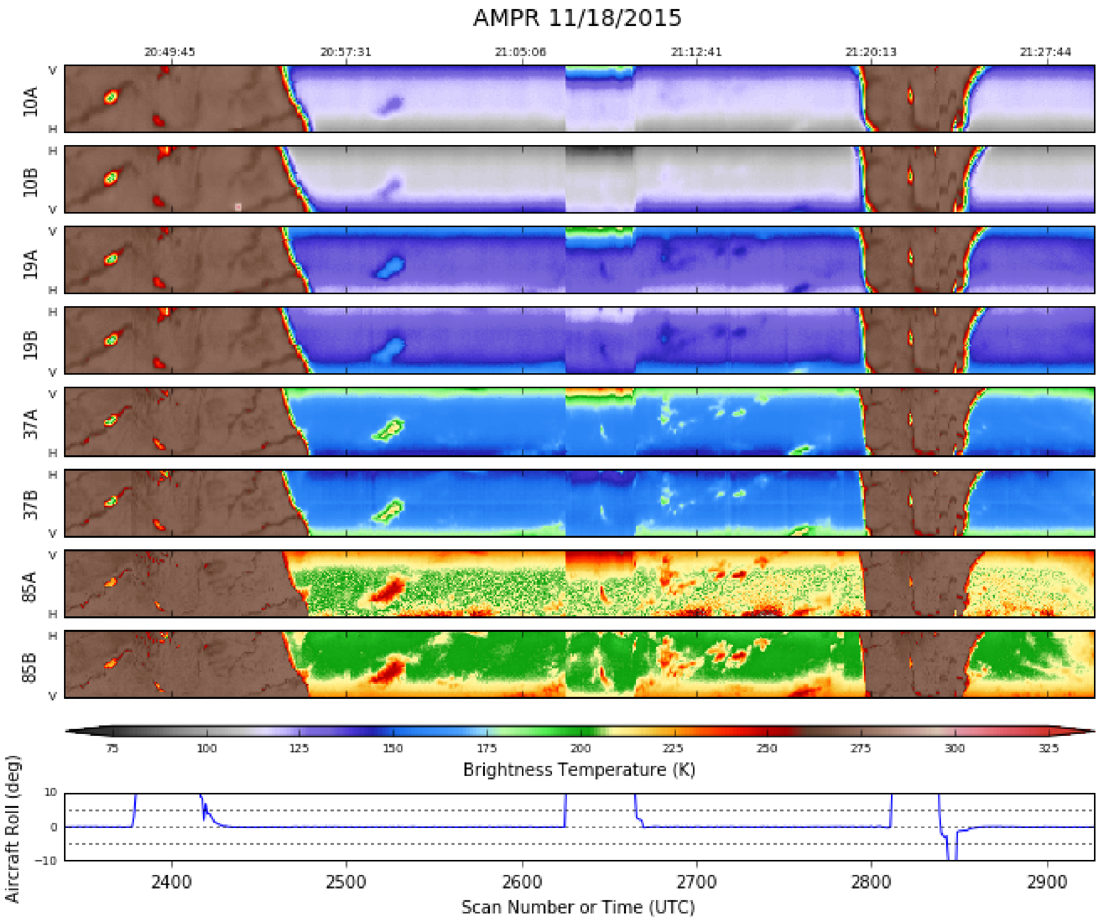


0.0.1 QC Issues

OK, now let's address what happened at 85 GHz (A) about 3 hours in. Looking at the strip charts, the issue manifests itself as an apparent rise in noise floor. For example, compare 85A to 85B in the strip chart below.

```
In [36]: data.plot_ampr_channels(timerange=['20:45:00', '21:30:00'])
```

```
********************
plot_ampr_channels():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
********************
```



Prior to this time, over the ocean 85A had very similar TBs to 85B (cf. first strip chart in this notebook). Now, it looks like the ocean is getting swallowed by by noise at 85A. This is different behavior than the problem we saw in IPHEx, where occasionally 85A was affected by short-lived severe noise that drowned out almost all signal. Here, we still see precip emission signal at 85A (e.g., 85A during the 22:41 overpass is usable), and the land signal is just fine, just not the ocean.

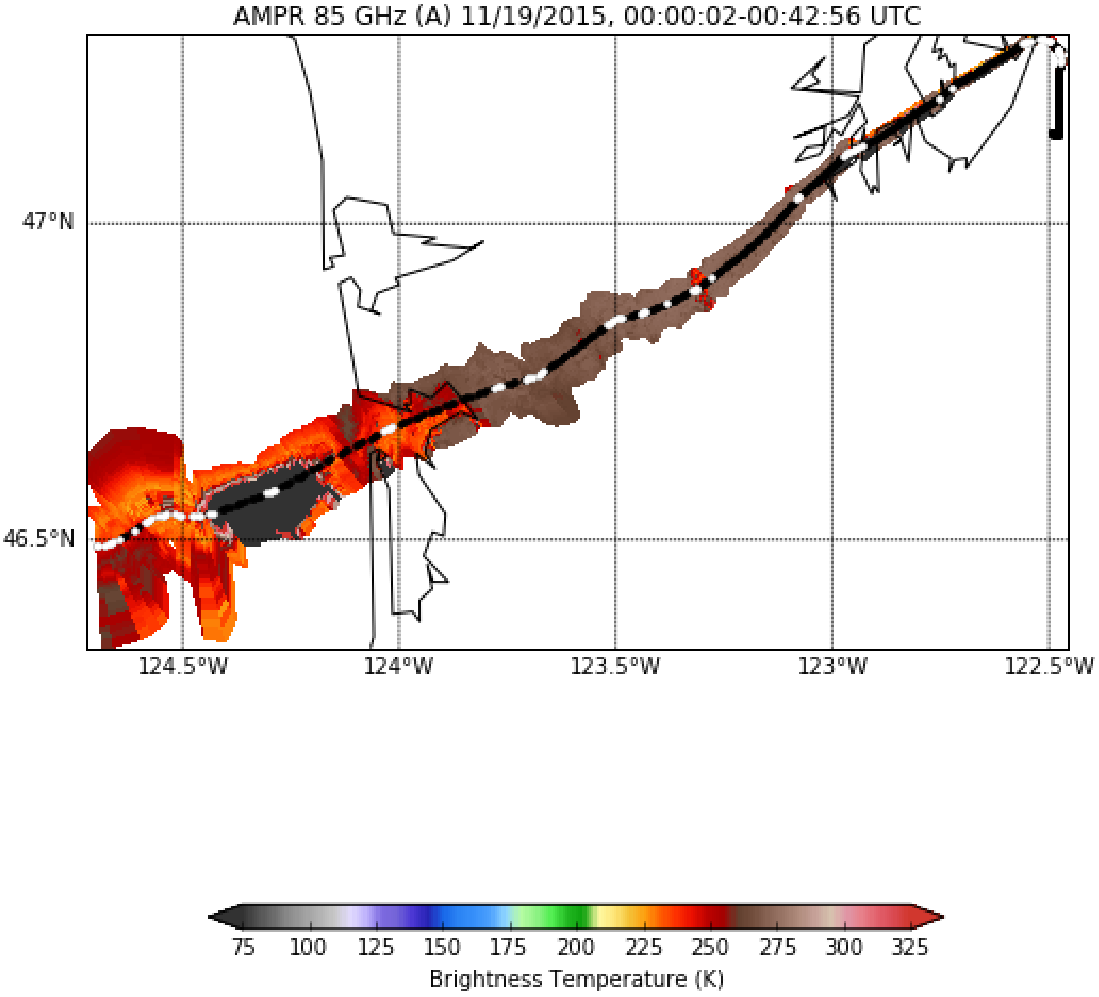
We are still reviewing the raw data counts from this flight. See below for some initial thoughts by our engineering consultant, Karthik Srinivasan: "It looks like the receiver gain hasn't changed much for both 85GHz channels, but there is a warming trend on the 85V counts, which translates directly to what appeared to be a higher noise floor. There was a small step rise in gain (close to sample 2600) on 85V, but that should have just resulted in a constant offset, but not a long term warming trend. Also it does appear that the 85V counts were a lot more noisy while this warming trend was on-going - again, the load counts (and hence gain) did not see this noise, which means it is a real 'signal' of some sort."

One possibility is that we heard one of the radars was reset during the flight. Radar resets likely caused noise issues during IPHEx too. We'll look into whether we can tune our L2 retrieval at 85A to recover accurate TBs over cold targets like the ocean in post-processing.

Finally, the 85A TBs went very high when viewing the ocean, very late in the flight. However, this occurred during approach, as the aircraft was descending in altitude. This started to affect the load temperatures, so one shouldn't read too much into these changes. See the figures below.

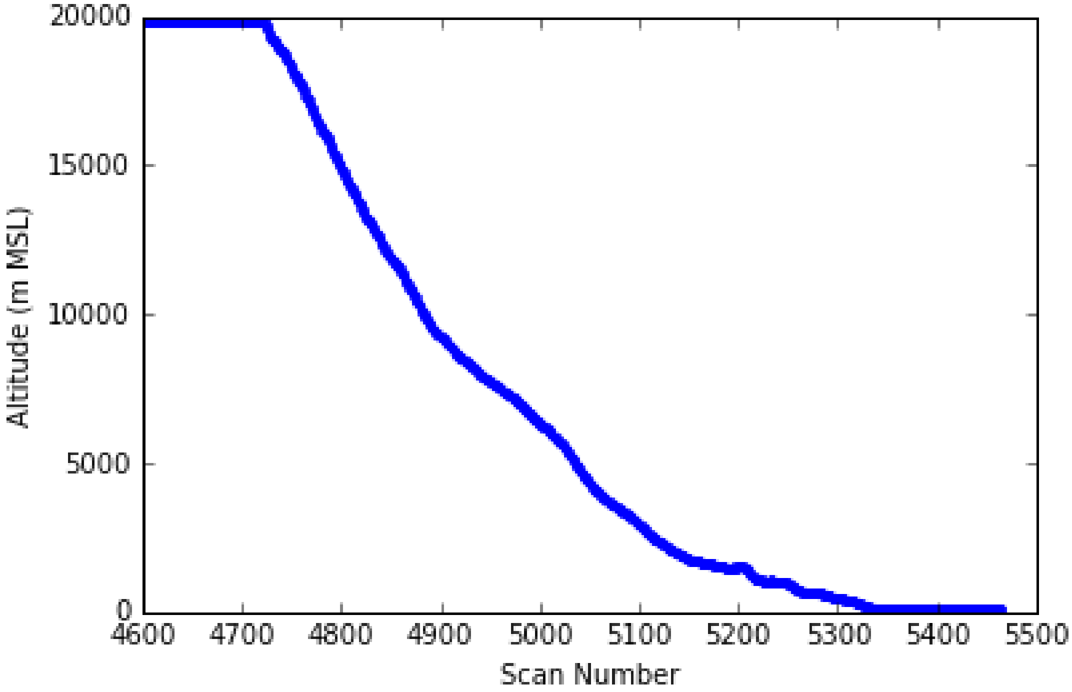
```
In [40]: data.plot_ampr_track('85a', timerange=['00:00:00', '01:00:00'],
                              show_track=True, meridians=0.5, parallels=0.5,
                              resolution='i')
```

```
********************
plot_ampr_track():
Available scans = 1 to 5466
Available times = 17:46:40 - 00:43:00
Warning: Your aspect ratio choice could lead to poor plotting results.
Best results are obtained when latrange ˜ lonrange.
********************
```


AMPR 85 GHz (A) 11/19/2015, 00:00:02-00:42:56 UTC
47°N
46.5°N
124.5°W
124°W
123.5°W
123°W
122.5°W
75
100
125
150
175
200
225
250
275
300
325
Brightness Temperature (K)

There were a lot of maneuvers during this time too (white dots above). Below see the descent of the aircraft during this period.

```
In [45]: plt.plot(data.Scan, data.Aircraft_Nav['GPS Altitude'], 'b.')
         plt.xlim(4600, 5500)
         plt.ylabel('Altitude (m MSL)')
         plt.xlabel('Scan Number')

Out[45]: <matplotlib.text.Text at 0x1238f7ef0>
```



Next possible flight day is Saturday. AMPR is green and ready to fly. We'll keep an eye on the 85 GHz (A) issue, but for now it should be good for viewing land and most precip regardless of whether the issue resurfaces.

```
In [ ]:
```