GBT Docs

Release 0.1

A. Schmiedeke

Sep 08, 2023

CONTENTS

GBT documentation is segmented into 4 pillars.

Tutorials

Learning-oriented lessons that take you through a series of steps to complete a project.

Most useful when you want to get started with the GBT.

Go to Tutorials

How-To Guides

Practical step-by-step guides to help you achieve a specific goal.

Most useful when you're trying to get something done.

Go to How-To Guides

Explanation

Big-picture explanations of higher-level concepts.

Most useful for building understanding of a particular topic.

Go to Explanation Material

References

Nitty-gritty technical descriptions of how the GBT works.

Most useful when you need detailed information about different GBT components.

Go to Reference Guides

CHAPTER

ONE

TUTORIALS

Learning-oriented lessons that take you through a series of steps to complete a project. Most useful when you want to get started with the GBT.

CHAPTER

TWO

HOW-TO GUIDES

Practical step-by-step guides to help you achieve a specific goal. Most useful when you're trying to get something done.

2.1 Data Processing

Data Reduction Examples using GBTIDL

2.2 Quick Guides

HI, position-switched, single-pointing

2.2.1 GBTIDL Data Reduction Examples

Below are a series of common data reduction GBTIDL processes with explanations and screenshots mixed in. To get an in-depth explanation for any gbtidl-specifig procedure or function, you can type "usage" with the procedure/function name and the "/verbose" option:

```
usage, “getps”, /verbose
```

2.2.1.1 Basic On/Off

Here is an example of a basic data reduction process on an L-band On/Off procedure.

Data

ngc2415.fits

Load the data into gbtidl.

```
filein, "data/ngc2415.fits"
```

Display a summary of its contents.

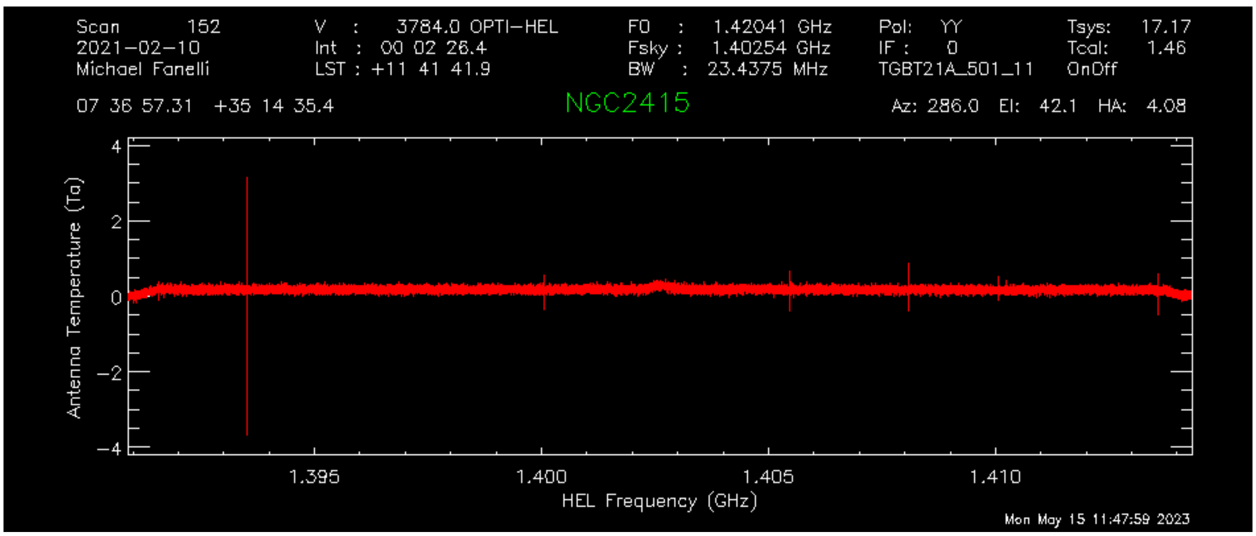
```
summary
```

```
GBTIDL -> summary
  Scan             Source      Vel    Proc Seq    RestF nIF nInt nFd     Az    El
-------------------------------------------------------------------------------
   152           NGC2415   3784.0   OnOff   1    1.420   5  151   1  286.0  42.1
   153           NGC2415   3784.0   OnOff   2    1.420   5  151   1  286.6  41.6
```

We can see that this data set has two scans on the galaxy NGC 2415, with one slightly offset in position. The catalog used for this observation says the galaxy has a velocity of 3784 km/s, the rest frequency of the observed line in the first frequency window is 1420 MHz/1.420 GHz, there are 5 frequency windows, 151 integrations, 1 beam/feed, and lastly the azimuth/elevation values of the GBT at the time of the scan.

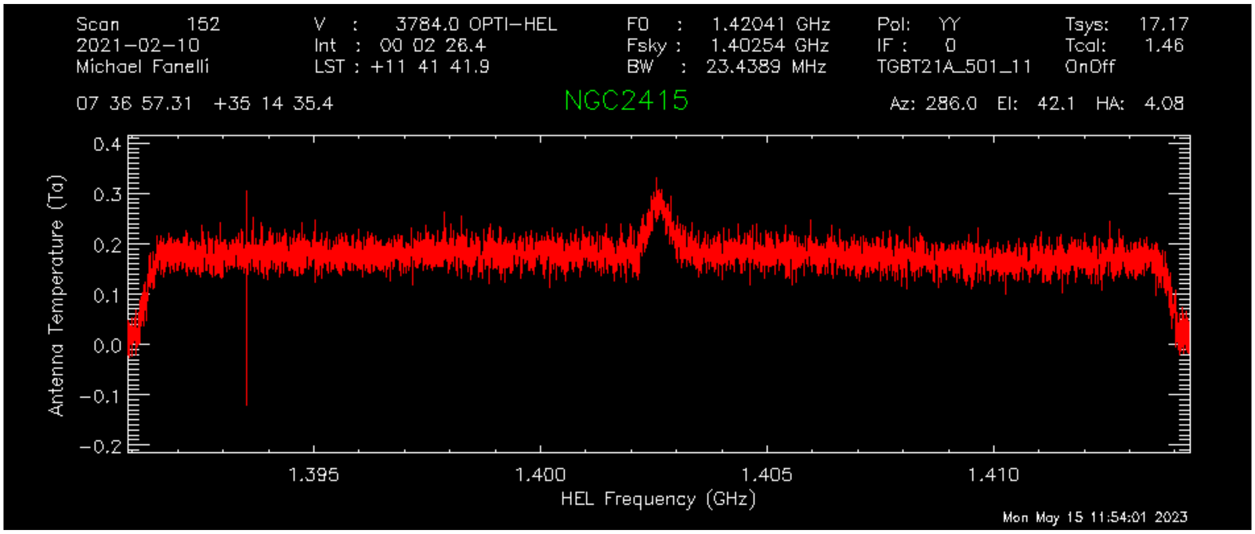
For a first look at the spectrum, we call the "getps" procedure, which calibrates the On vs. Off scans and averages up all 151 integrations. Without designating the polarization or spectral window, it will default to the first polarization (Linear YY for the L-band receiver) and first spectral window (centered on the redshifted HI line in this case).

```
getps, 152
```



In order to see anything useful, we may have to smooth the data. This is usually done with the procedures "gsmooth" or "boxcar", with the latter using a gaussian smoothing kernel and the latter using a flat kernel. The first input "N" is the size of the kernel in channels and the second option keeps only every N-th channel.

```
gsmooth, 5, /decimate
```

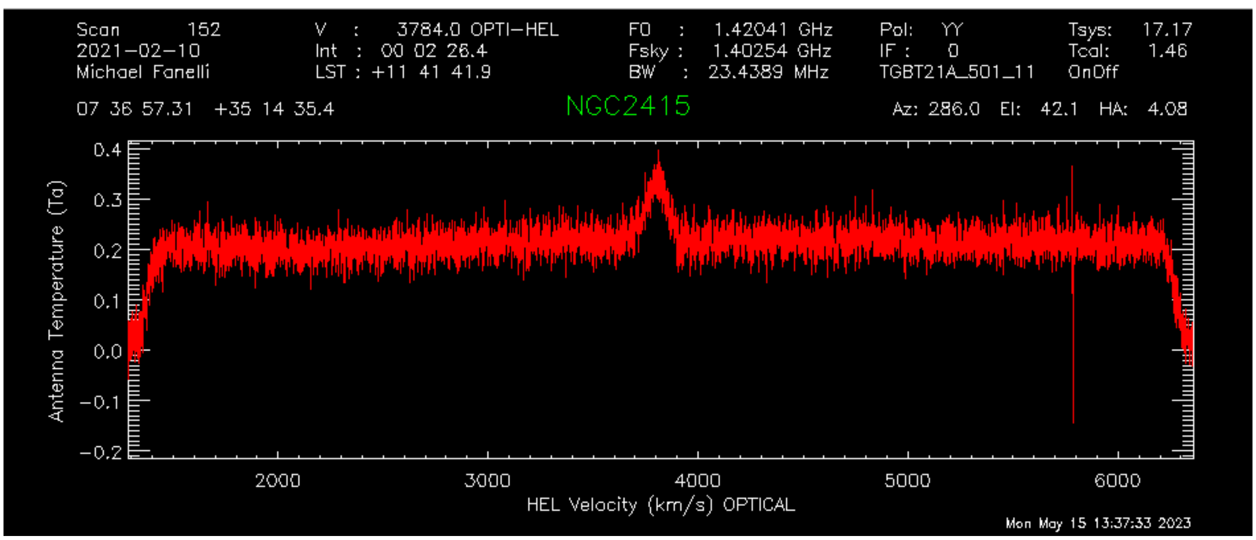


We can convert the data to the same velocity frame specified in the observing configuration with "setxunit". By default this will apply a doppler correction. This can also be done by selecting "km/s" in the GUI - the fourth dropdown menu from the left.

```
setxunit, "km/s"
```

We may learn that the TCAL value in the database is out of date, and the signals are too weak by a factor of 1.2. "scale" removes this systematic offset.

```
scale, 1.2
```



Now we may want to save these results for posterity or further analysis. You can also use "write_csv" to make a csv file instead of an ascii-columned text file, or "write_ps" to output a "publication-quality" postscript image of the spectrum.

write_ascii,"NGC2415_HI.ascii"

gbtidl also allows for multi-line input and loops in the same line. We can use this to quickly check the other spectral windows. The line breaks are made with the "&" character. First, let's freeze the plotter so it doesn't auto-update after each loop:

```
freeze
```

And change the x-axis back to frequency so that each spectral window is plotted correctly.

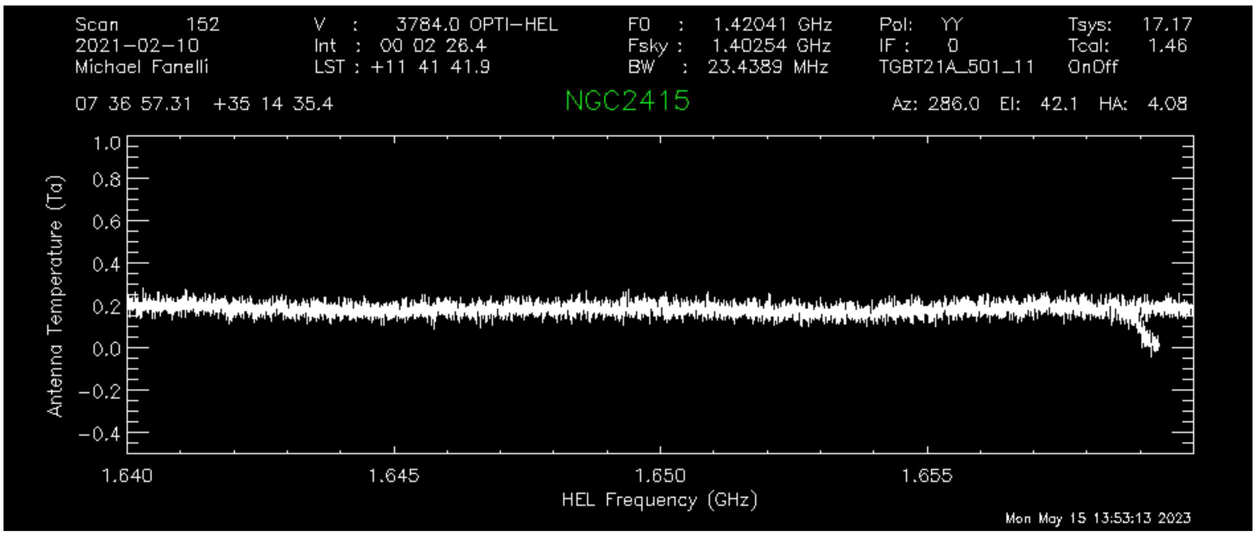
```
setxunit,"GHz"
```

Now finally type in the loop:

```
for k=1,4,1 do begin & getps,152,ifnum=k & gsmooth,5,/decimate & scale,1.2 & oshow & endfor
```

Note that the third and fourth spectral windows overlap significantly. We're not going to save these spectra. We can zoom into the OH line:

```
setxy,1.64,1.66,-0.5,1
```



And it doesn't look like there is anything there.

2.2.1.2 Basic Frequency-Switched (fsw)

Here is an example of a basic data reduction process on a nearby ammonia cloud.

Data

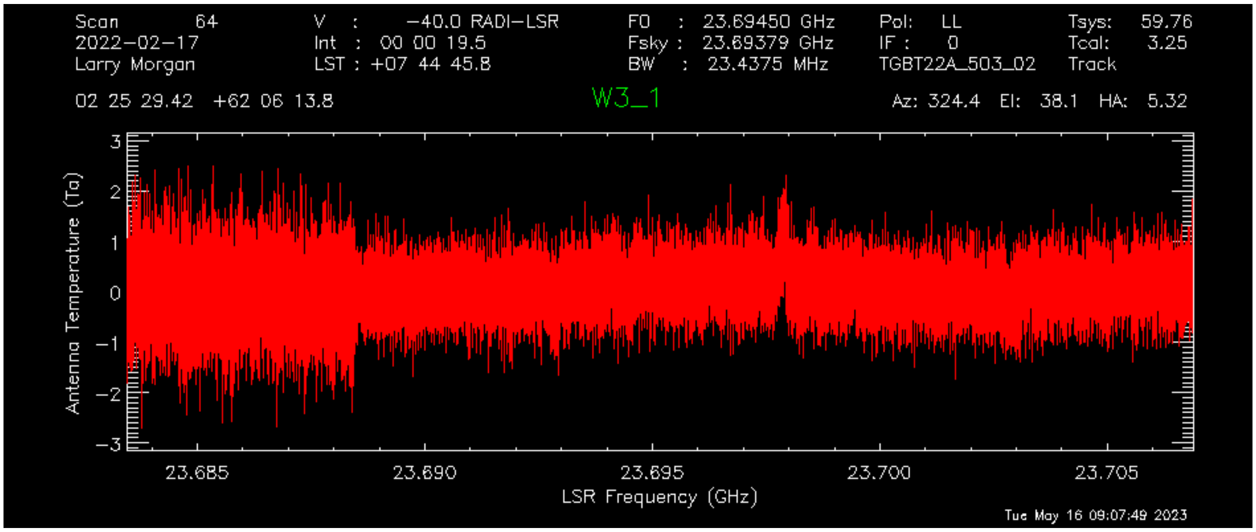
TGBT22A_503_02.raw.vegas

Load in the data. This is a directory, so either "dirin" or "filein" will work.

```
filein, "data/TGBT22A_503_02.raw.vegas"
summary
```

In here, there is one frequency-switched scan (#64), and two nodding scans (#62 and #63). For the KFPA, which is 7 beams arranged in a hexagon, the *Track()* command will use the central beam by default, so fdnum=0. The frequency-switched scan is calibrated with

```
getfs,64,fdnum=0
```



It looks like there is a small detection at around 23.698 GHz. We can smooth the spectrum to see a little more clearly.

```
gsmooth,5,/decimate
```

There is a very nice detection of ammonia! Let's average the two polarizations to reduce the noise even further. First, put the current spectrum in the primary accumulation buffer;

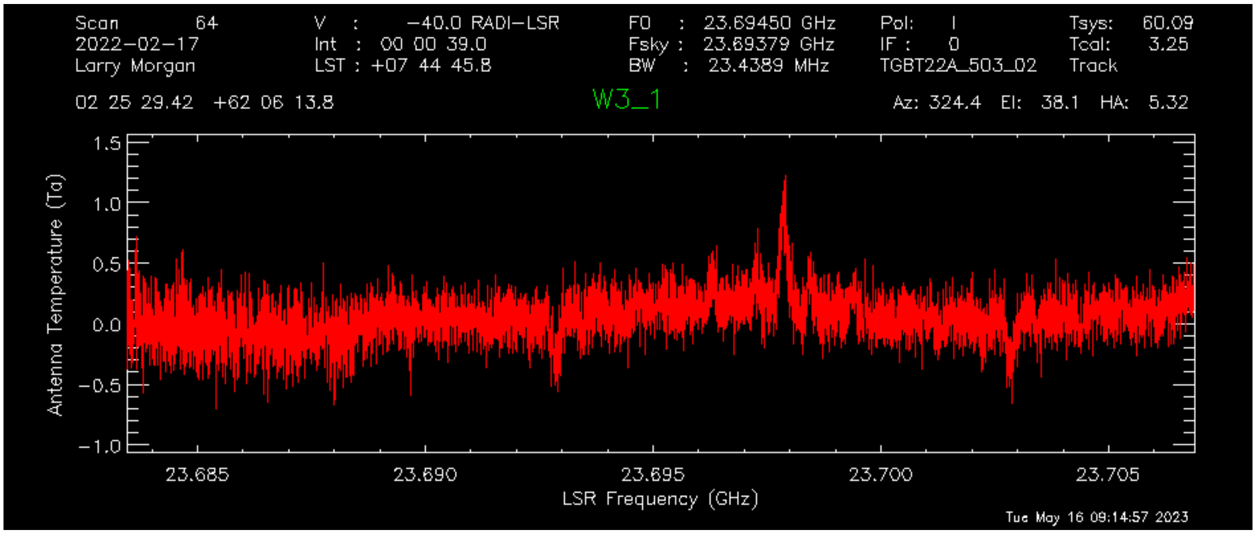
```
accum
```

And load the other polarization, smooth it to the same frequency resolution, and add it to the primary accumulation buffer. You can press the up arrow in GBTIDL for an input history.

```
getfs,64,fdnum=0, plnum=1
gsmooth,5,/decimate
accum
```

Now we average the two spectra in the accumulation buffer together, which will automatically drop the result in the primary data container.

```
ave
```



We can output this spectrum to an sdfits file with the "keep" procedure. We have to set a filename first.

```
fileout, “W3_1_NH3.fits”
keep
```

To compare with the nodding scans in the next example, we can save this to the next data container in GBTIDL.

```
copy,0,1
```

2.2.1.3 Basic Nodding

Here is an example of a basic data reduction process on a nearby ammonia cloud, from a nodding scan.

Data

TGBT22A_503_02.raw.vegas

Nodding scans are only done with multibeam receivers on the GBT. They are performed by tracking the source with one beam for a certain amount of time, then moving the whole telescope in azimuth/elevation so that a different beam tracks the source for the same amount of time. It effectively functions like a double On/Off scan. The same data file we used above has two nodding scans on the same ammonia cloud.

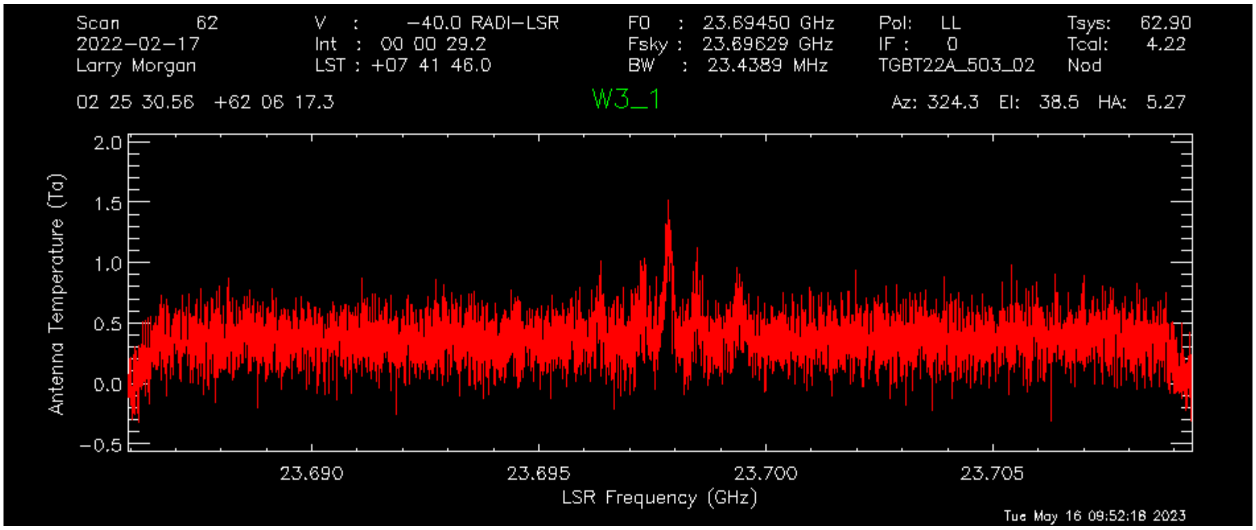
```
filein, "data/TGBT22A_503_02.raw.vegas"
```

Normally, nodding scans are calibrated with "getnod", which is currently broken for KFPA data. We can use "getsigref" instead, which functions almost identically. We just have to define the signal and reference scans for each beam. For this data, the nodding was between beams 3 and 7, which correspond to fdnum values of 2 and 6. First, "sclear" makes sure the accumulation buffer from the previous example is cleared.

```
sclear
```

Beam 3 was on source in scan 62 (the "signal" scan) and offset in scan 63 (the "reference" scan).

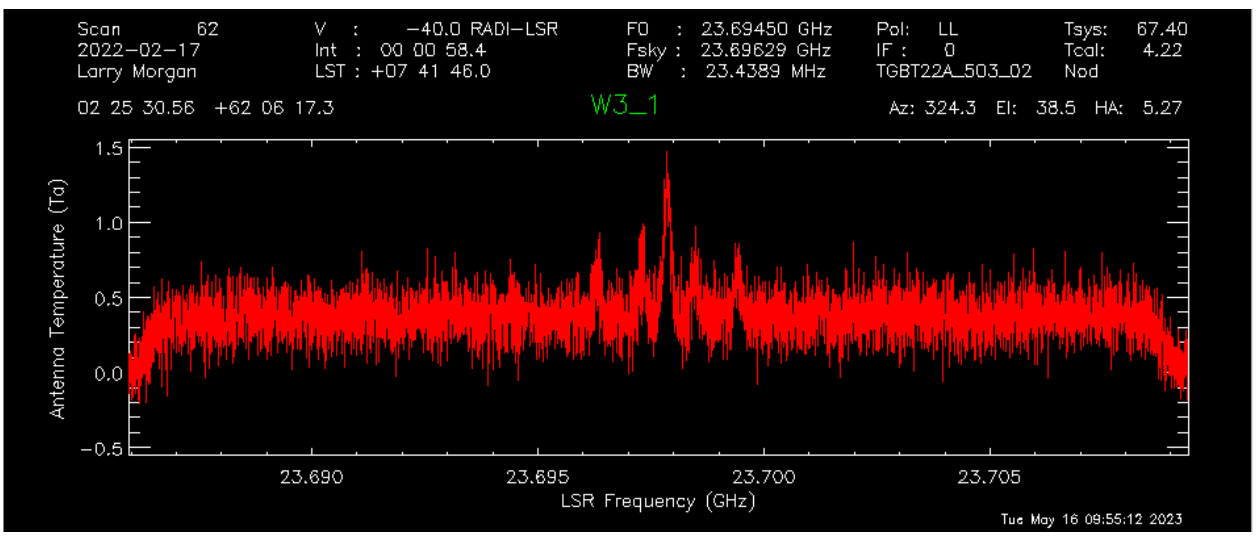
```
getsigref, 62, 63, fdnum=2
gsmooth, 5, /decimate
```




```
accum
```

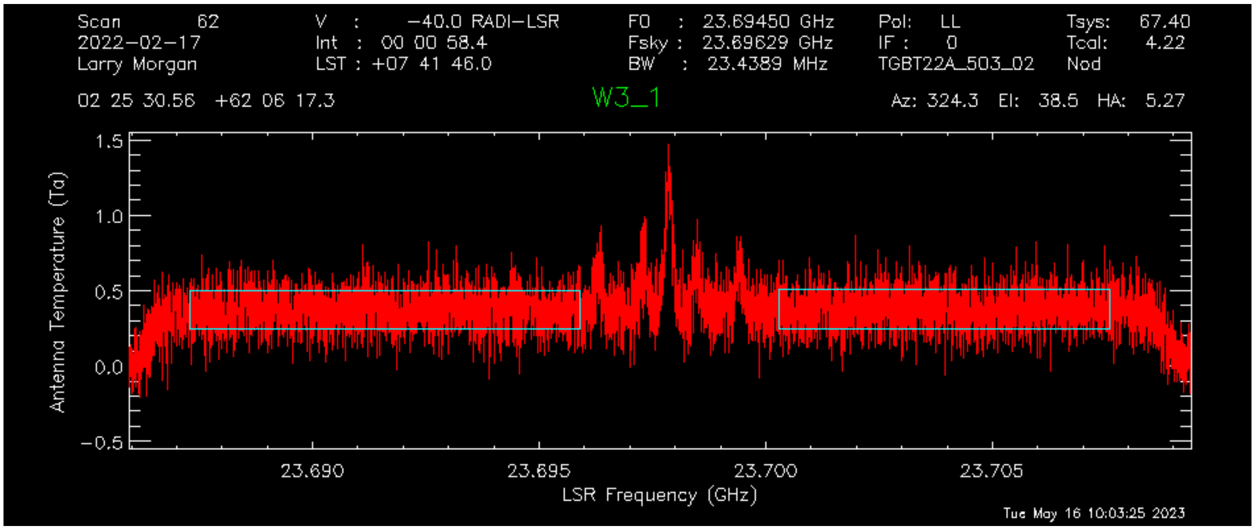
Beam 7 was also part of the nod, but was offset in the opposite way. So, scan 63 is now the signal scan, and 62 is the reference scan.

```
getsigref, 63, 62, fdnum=6
gsmooth, 5, /decimate
accum
ave
```



The continuum is slightly offset from 0, so we can use the baseline feature to subtract that out. “setregion” sets the areas the fitting procedure uses, and this can be done either on the GUI with the left/right mouse buttons or by designating a series of start/stop points in channel number from the command line input. The regions in this case should be everything except for the rolloff at the edges of the band and around the signal itself.

```
setregion
```

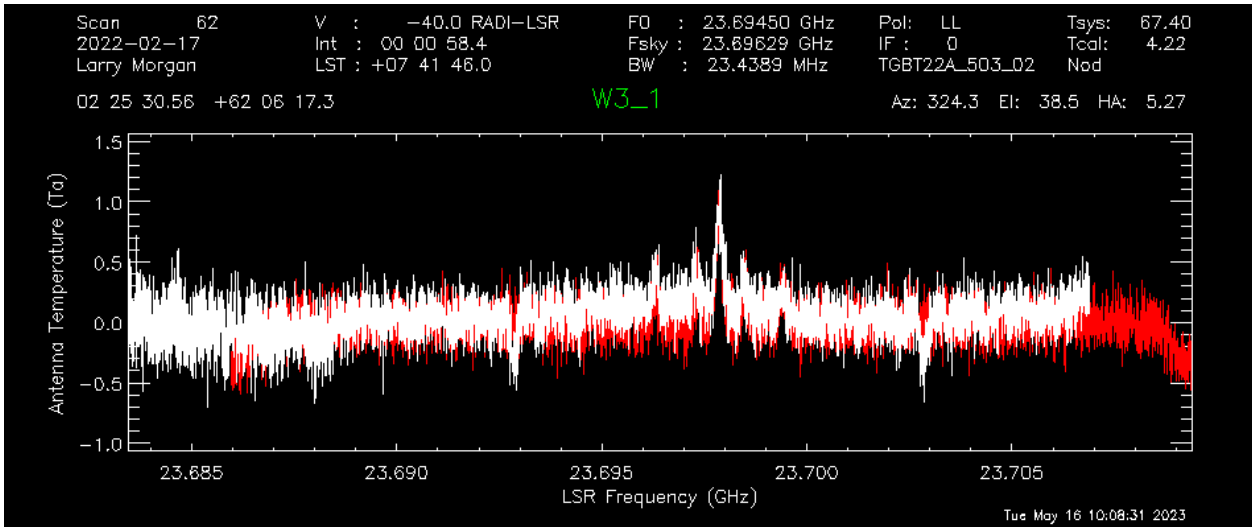


"baseline" by default uses a 0th order polynomial - a flat line - to fit. The continuum is already pretty flat, so this is all that's needed.

```
baseline
```

Now the spectrum's baseline should be centered about Ta = 0 Kelvin. Next, to compare with the frequency-switched data, we use "oshow" with the number of the data container we saved to.

```
oshow,1
```



2.2.1.4 Advanced On/Off

RFI excision and baselining

Data

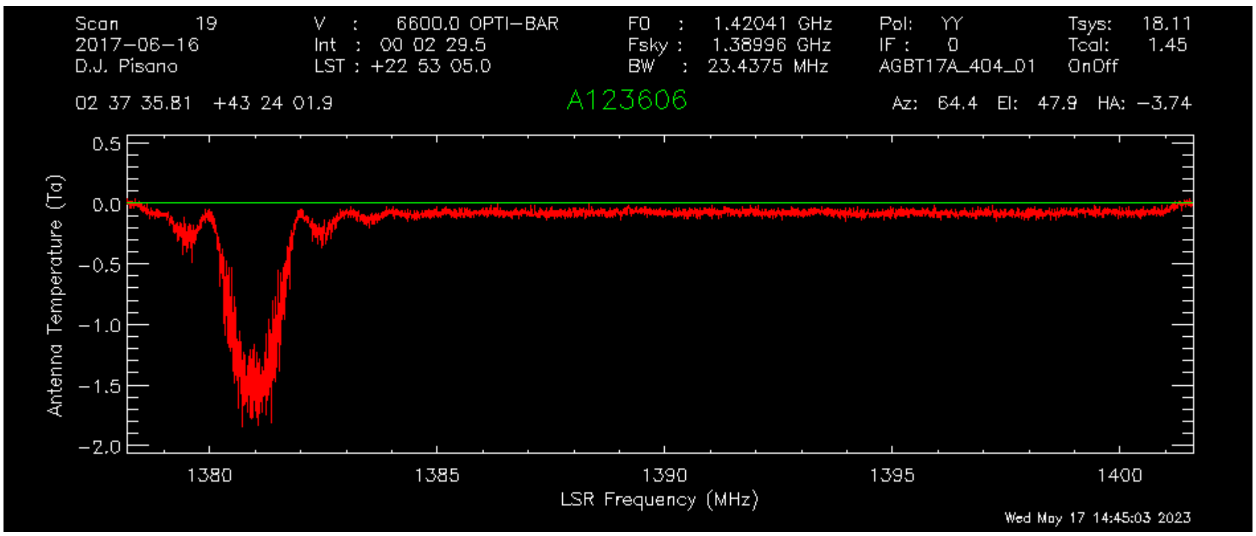
AGBT17A_404_01.raw.vegas

Load in the data:

```
filein, "data/AGBT17A_404_01.raw.vegas"
```

You can see there are two sets of position-switched L-band scans here. We will start with the latter two and see if we can find an HI detection:

```
getps,19
zline
```



"zline" will help with modelling the baseline later. We can see there is a huge GPS-L3 RFI signal flooding out the left side of the band. We can step through one integration at a time (there are 60 total plus one blanked integration) to see how bad/pervasive the GPS is.

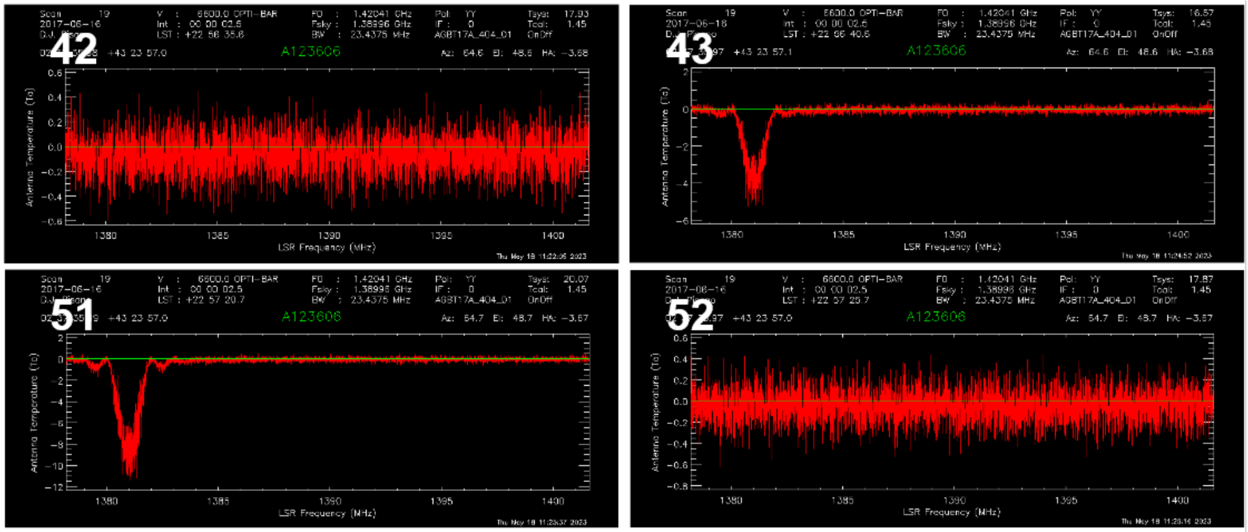
```
for i=0,61 do begin & getps, 19, intnum=i, plnum=0 & end
```

This will step through all 60 integrations as fast as your computer can calibrate and plot them. If you want to see it a little slower, you can add a wait statement:

```
for i=0,61 do begin & getps, 19, intnum=i, plnum=0 & wait, 0.3 & end
```

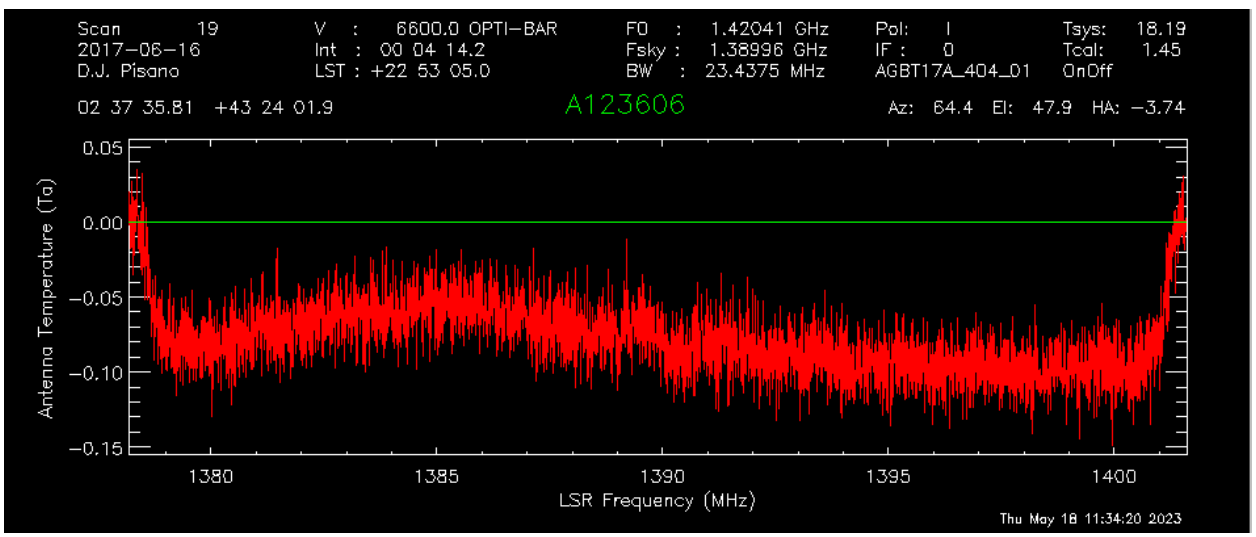
From this, we can see there is only a portion in the latter half of the OFF scan that is blocked by RFI. Stepping through the integrations manually, we can see the trouble starts in integration #43 and ends at integration #51.

```
getps,19, intnum=42
getps,19, intnum=43
getps,19, intnum=51
getps,19, intnum=52
```



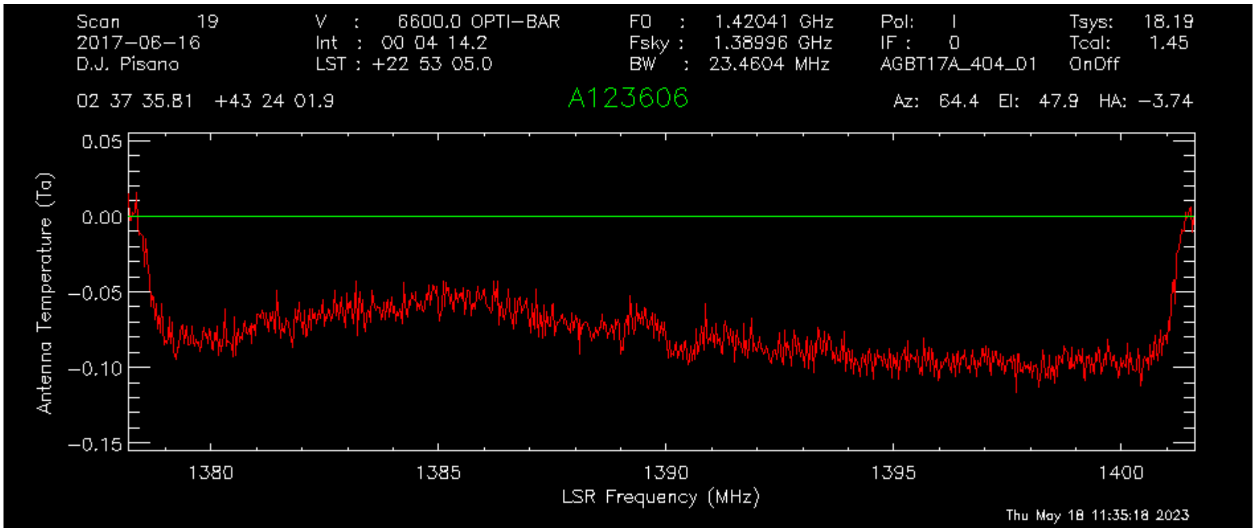
So let's accumulate all the clean integrations for both polarizations, and see if there's any HI detection. Keep in mind the IDL for loops are inclusive on both ends.

```
sclear
for i=0,42 do begin & getps, 19, intnum=i, plnum=0 & accum & end
for i=0,42 do begin & getps, 19, intnum=i, plnum=1 & accum & end
for i=52,60 do begin & getps, 19, intnum=i, plnum=0 & accum & end
for i=52,60 do begin & getps, 19, intnum=i, plnum=1 & accum & end
ave
```



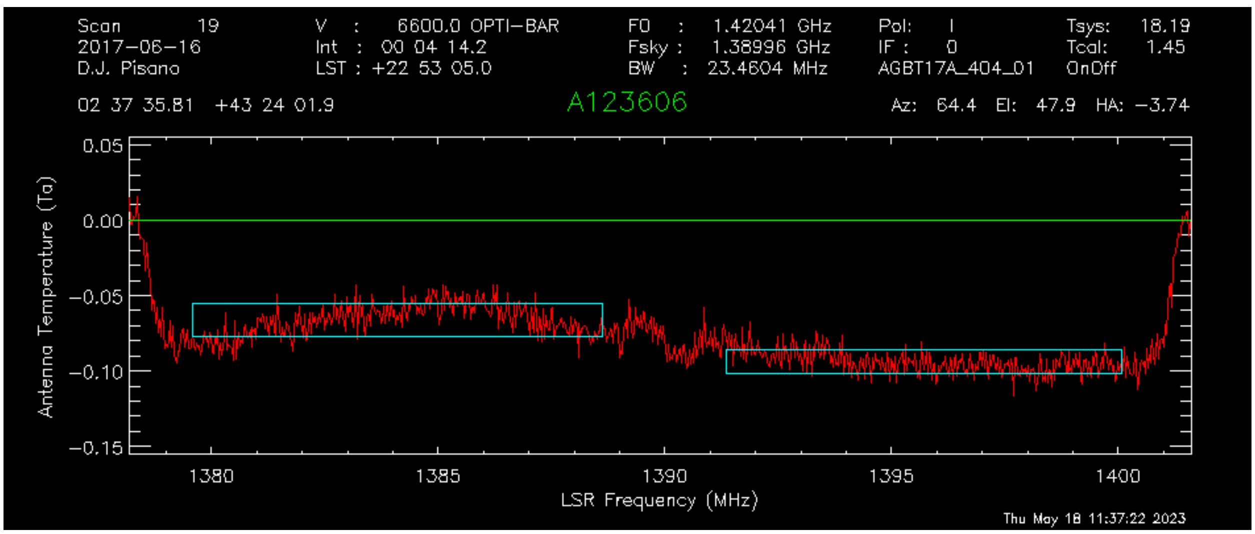
Smooth is:

```
boxcar, 5, /decimate
```



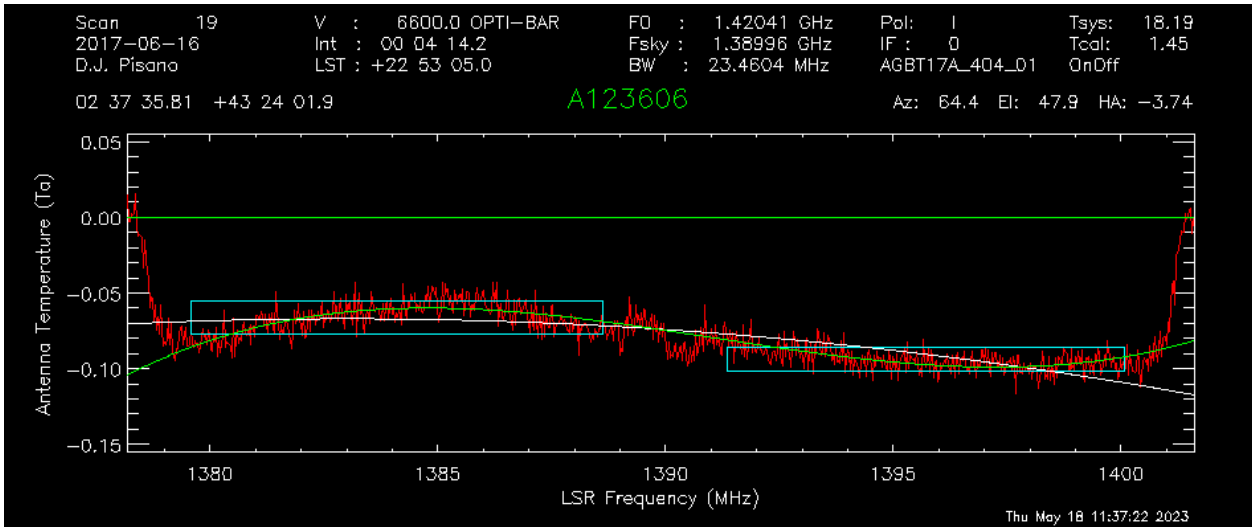
There may be a small detection at 1389.5 MHz. Let's try to fit a baseline - we may have to fit either a 2nd or 3rd order polynomial. First, we will set a checkpoint here by copying the current spectrum to the second data container so we can go back to this step. Then, setregion to everything but the bandpass edges and the possible signal in the middle:

```
copy, 0, 1
setregion
```



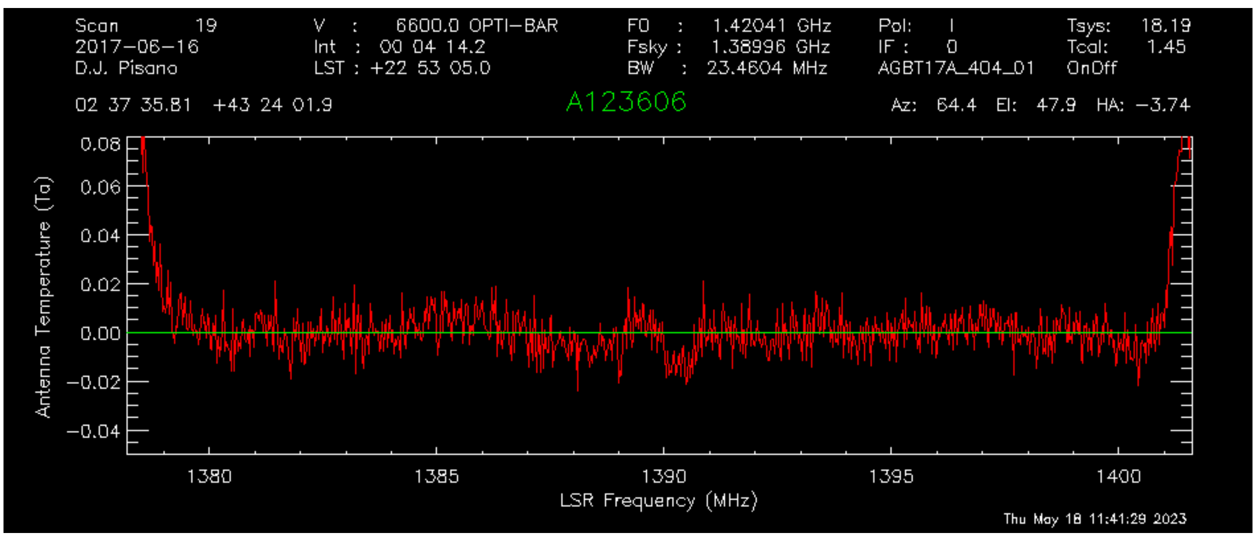
We can trial baseline fits with the "bshape" procedure.

```
bshape, nfit=2
bshape, nfit=3, color=!green
```



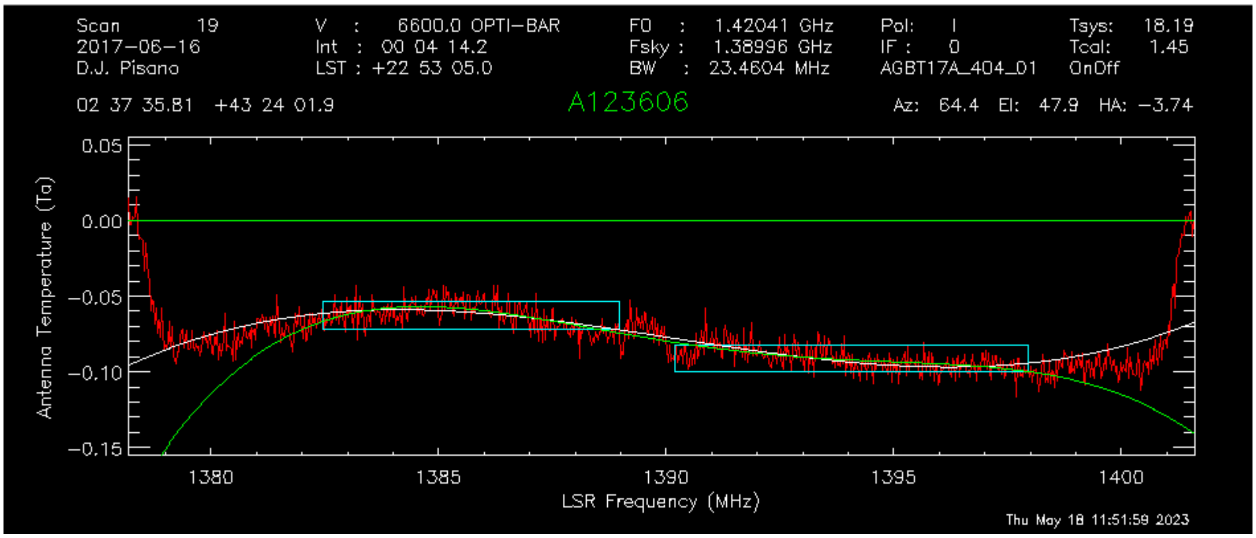
The 3rd order fit (green) looks much better than the 2nd order fit (white). Next, the "bsubtract" procedure applies the last fit computed and subtracts it from the data - in this case, our 3rd order fit.

```
bsubtract
sety, -0.05, 0.08
```



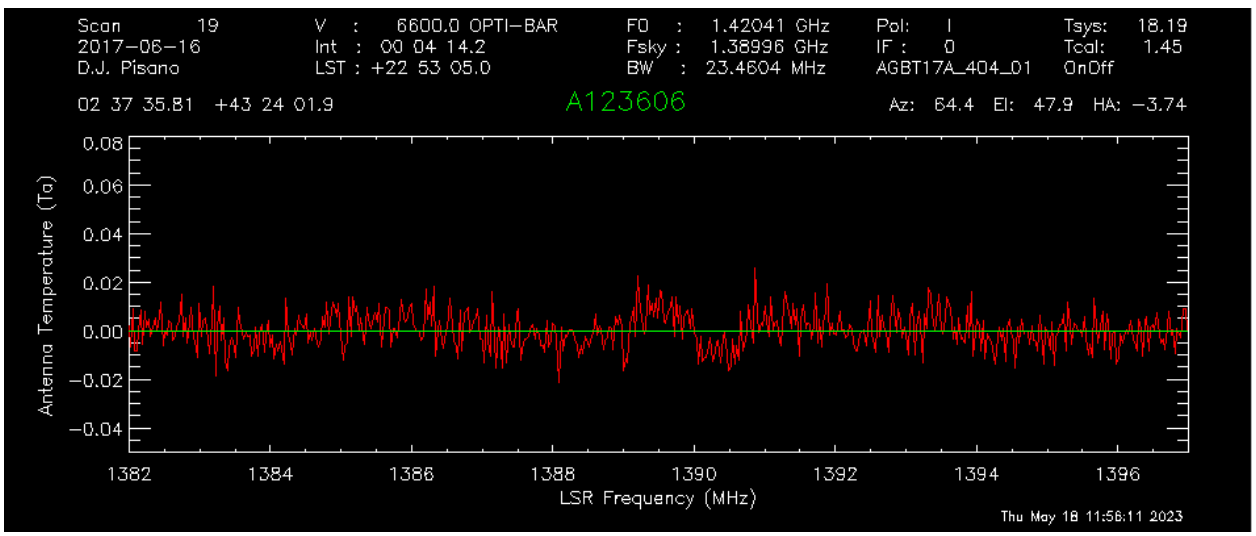
There may be a tiny detection, but the baseline fit is not the best, particularly noticeable in the 1384 - 1389 MHz range. We might go back and see if we can apply a more strict fit, setting the region to be closer in to our possible detection and avoiding more of the bandpass edge.

```
copy,1,0
setregion                          ; see image below for the range I chose
bshape, nfit=3
bshape, nfit=4, color=!green
```



The fourth order fit looks to follow that hump at 1385 MHz a little better, so we might pick that despite the large divergence towards the edges of the band.

```
bsubtract
setxy, 1382,1397,-0.05,0.08
```

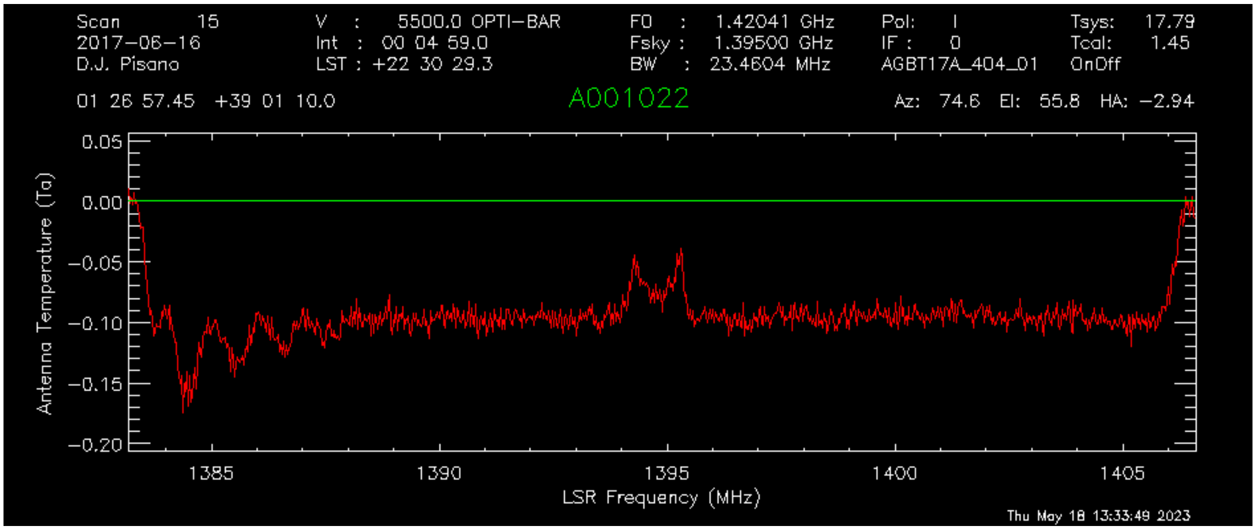


The possible signal looks slightly more significant, but maybe not quite enough to warrant a detection.

Double Gaussian feature

Now let's turn our attention to scan 15. First, accumulate both polarizations together.

```
sclear
getps,15
accum
getps,15, plnum=1
accum
ave
boxcar, 5, /decimate
```



It does look like there is some GPS-L3 interference on the left side again, we can ignore that since it is far away. Let's grab some info about the spectrum and switch to velocity units.

```
header
```

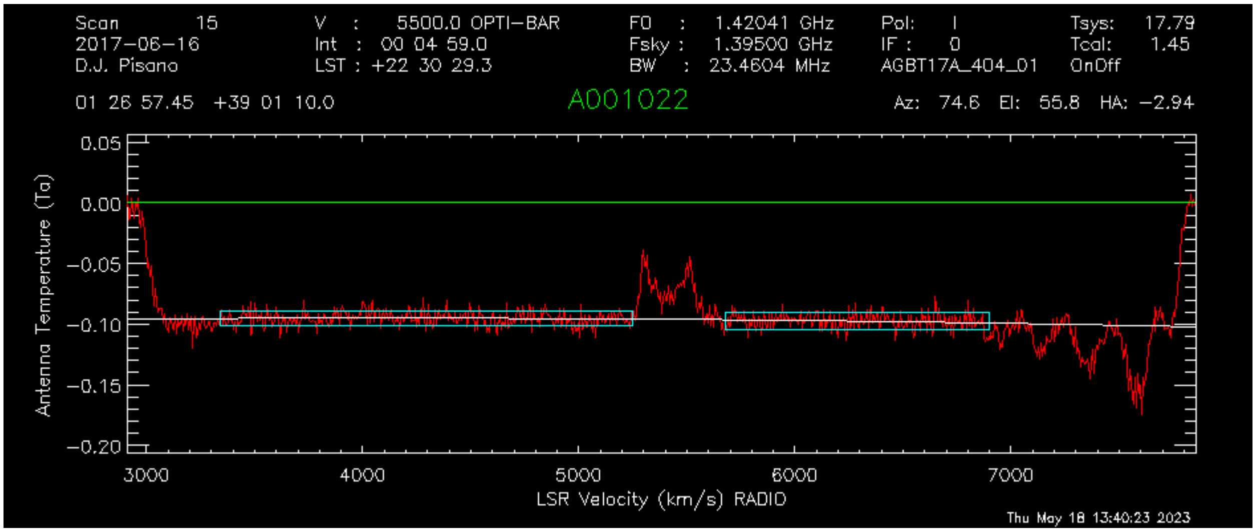
```
---------------------------------------------------------------------------------
Proj: AGBT17A_404_01   Src   : A001022                   Obs : D.J. Pisano

Scan:         15       RADec :  01 26 57.4  +39 01 10      Fsky:   1.394995 GHz
Int :          0       Eqnx  :  2000.0                     Frst:   1.420406 GHz
Pol :          I       V     :  5500.0       OPTI-BAR      BW  :  23.460    MHz
IF  :          0       AzEl  :   74.621      55.767        delF:  28.610    kHz
Feed:          1       Gal   :  130.437     -23.334        Exp :  299.0     s
Proc:      OnOff       UT    : +10 10 19.0   2017-06-16    Tcal:    1.45    K
Seqn:          1       LST/HA: +22 30 29.3   -2.94         Tsys:   17.79    K
---------------------------------------------------------------------------------
```

The sky frequency, ν_{sky}, is 1395 MHz and the smoothed frequency resolution, $d\nu$, is 28.61 kHz, which corresponds to a velocity resolution, dv, of 6.15 km/s.

$dv = c * \frac{(\nu_{\text{sky}} - (\nu_{\text{sky}} - d\nu)}{(\nu_{\text{sky}} - d\nu)}$

There seems to be a slight downward curve in the baseline, so I will fit a 2nd order baseline.

```
velo
setregion
bshape, nfit=2
```

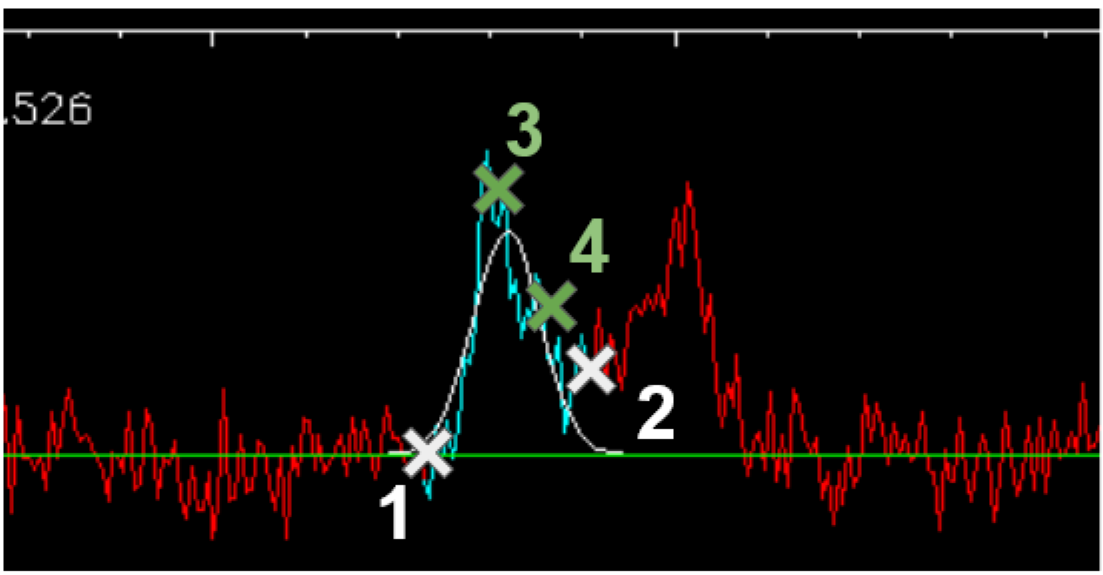



```
bsubtract
sety, -0.05, 0.08
```

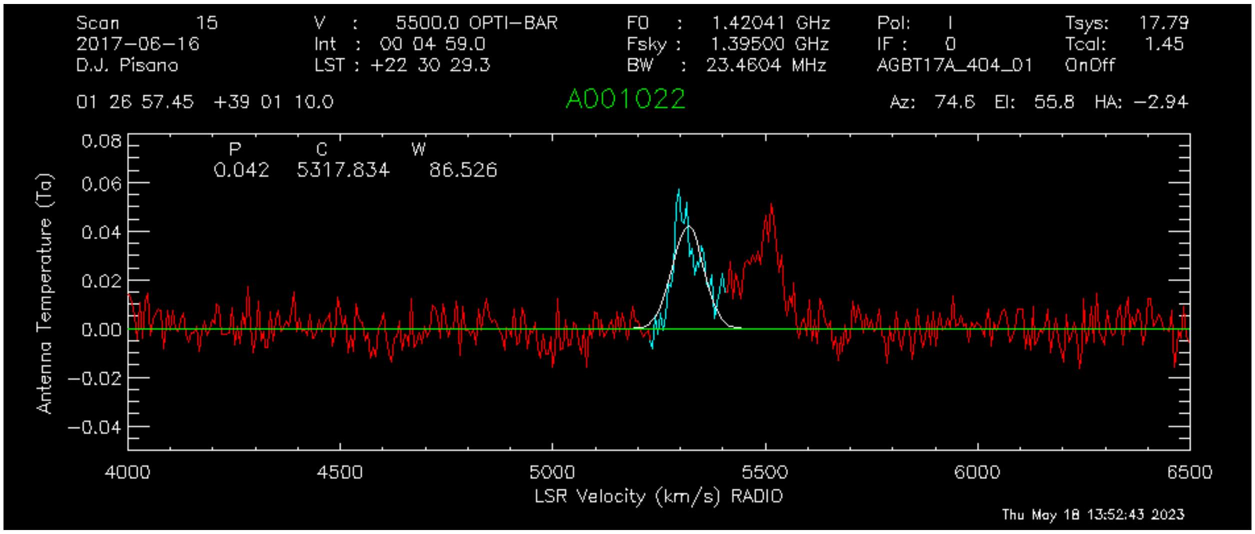
Now we will fit two gaussians to this detection. Since this is a rotating HI galaxy, the actual model should be a two-horn profile, but two gaussians should be enough to fit this. GBTIDL does not have a native two-horn profile fitting procedure.

```
setx,4000,6500
fitgauss,modelbuffer=2
```

The program will tell you what to do, but the process involves left clicking the boundaries of the signal, then giving it guesses to model with the middle mouse button. In the zoomed in image below, I left click at the white X marks on either side, then use the middle mouse button to click at the top of the signal, then the half-power point in the order shown:



And finally, a right click tells GBTIDL to model the Gaussian:

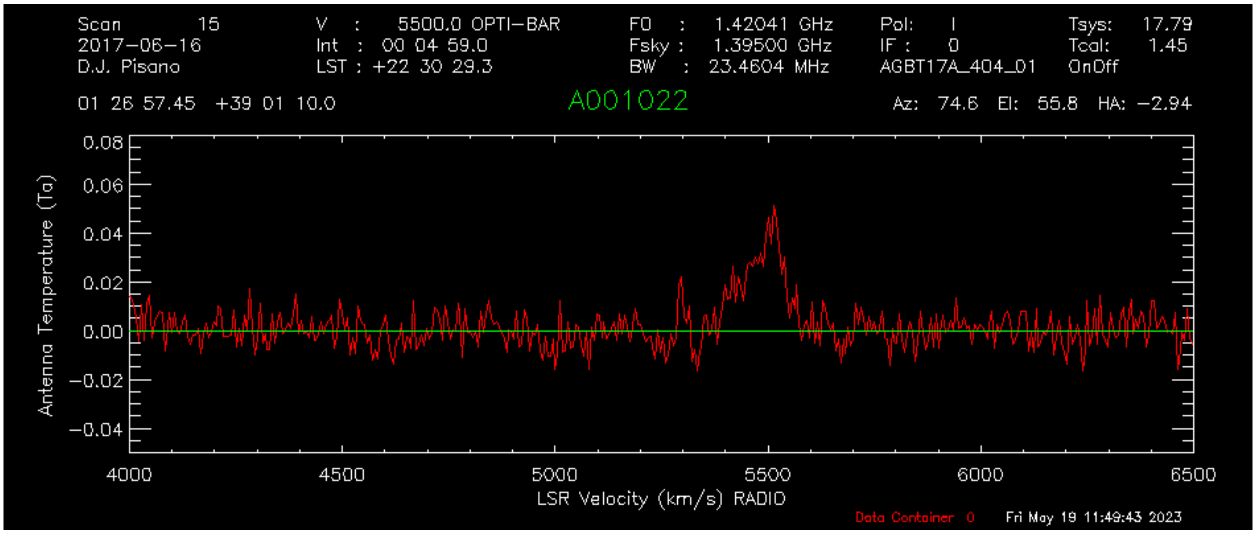



```
GBTIDL -> fitgauss,modelbuffer=2
***********************************************************
Instructions for fitgauss procedure:
Left Mouse Button: to mark regions to be fit.
Center Mouse Button: to mark initial guesses (center then width)
Right Mouse Button: to calculate and show fits
***********************************************************
Limits accepted for region        1
Guesses accepted for gaussian        1
***** Initial Guesses
          G#       Height        Center  (km/s)    FWHM  (km/s)
Init:      1      0.04912         5308.9331         55.83

***** Fitted Gaussians
        Height                       Center (km/s)                FWHM (km/s)
 1     0.04194 (  0.003969)     5317.8335 (      4.008)       86.53 (      9.610)
% Program caused arithmetic error: Floating underflow
```

We'll copy the original spectrum to data container 4, then subtract this gaussian out so we can model the other one.

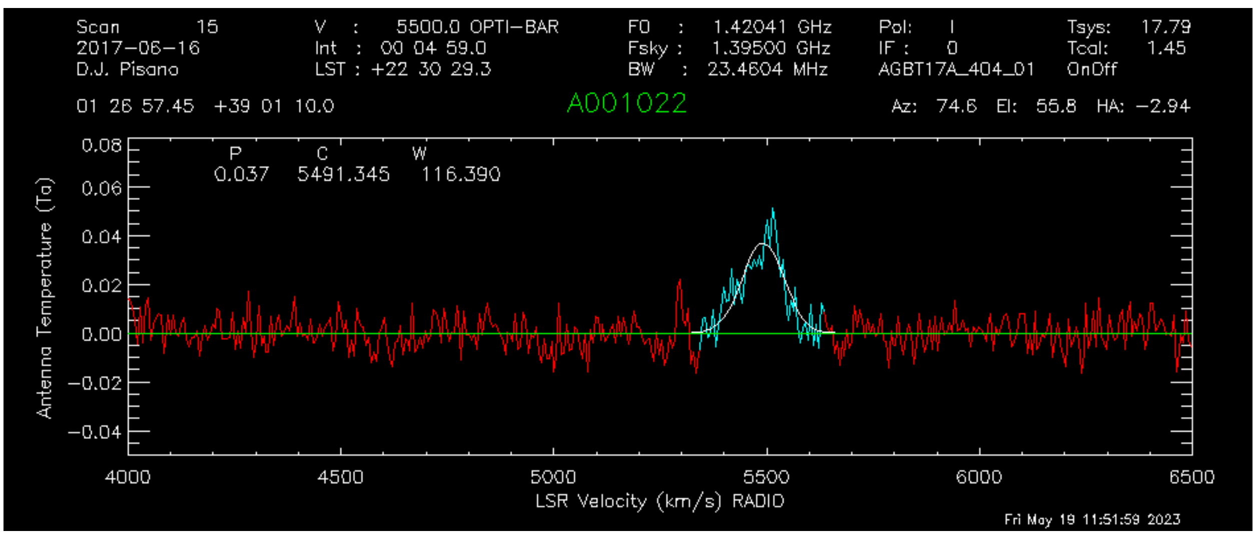
```
copy,0,4
subtract, 4, 2, 0
```

So now the primary data container has the results of DC4 - DC2.



Fit the other Gaussian:

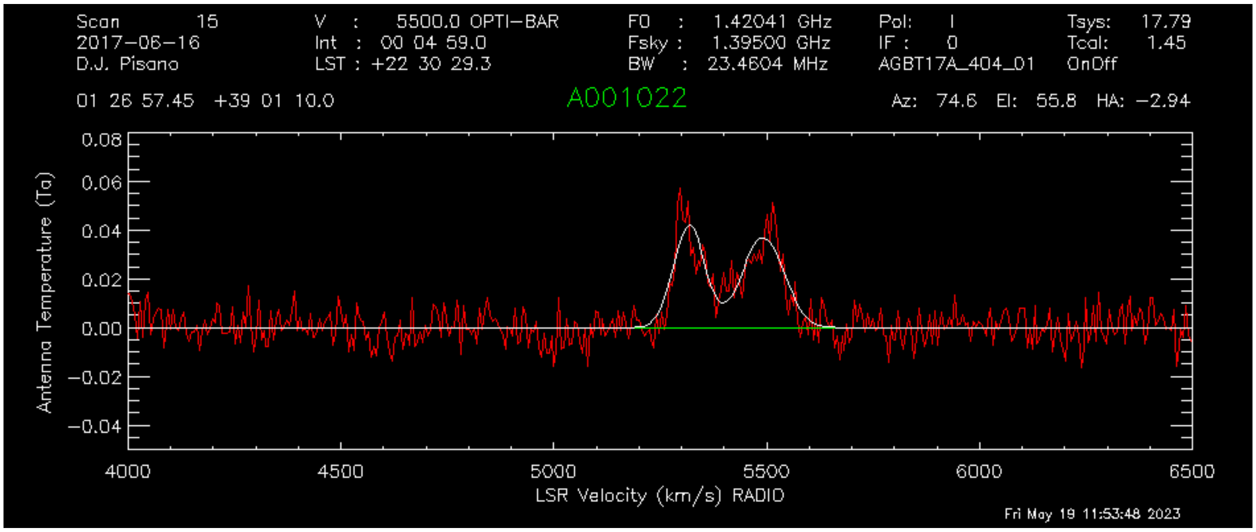
```
fitgauss, modelbuffer=3
```




```
***** Fitted Gaussians
      Height                    Center (km/s)              FWHM (km/s)
 1     0.03694 (   0.002464)    5491.3452 (      3.806)     116.4 (      8.978)
```

And then show the original spectrum with the two models overlaid:

```
copy, 4, 0
add, 2, 3, 5
oshow, 5
```


Scan 15
2017-06-16
D.J. Pisano
V : 5500.0 OPTI-BAR
Int : 00 04 59.0
LST : +22 30 29.3
F0 : 1.42041 GHz
Fsky : 1.39500 GHz
BW : 23.4604 MHz
Pol: I
IF : 0
AGBT17A_404_01
Tsys: 17.79
Tcal: 1.45
OnOff
01 26 57.45 +39 01 10.0
A001022
Az: 74.6 El: 55.8 HA: -2.94
Antenna Temperature (Ta)
0.08
0.06
0.04
0.02
0.00
-0.02
-0.04
4000
4500
5000
5500
6000
6500
LSR Velocity (km/s) RADIO
Fri May 19 11:53:48 2023

CHAPTER

THREE

EXPLANATION MATERIAL

Big-picture explanations of higher-level concepts. Most useful for building understanding of a particular topic.

CHAPTER

FOUR

REFERENCE GUIDES

Nitty-gritty technical descriptions of how the GBT works. Most useful when you need detailed information about different GBT components (receivers, observation procedures, data analysis tools).

4.1 Receivers

Argus

16-pixel receiver operating from 75-115 GHz.

Argus

4.2 Software

Astrid commands

Details on Astrid's scheduling block (SB) commands.

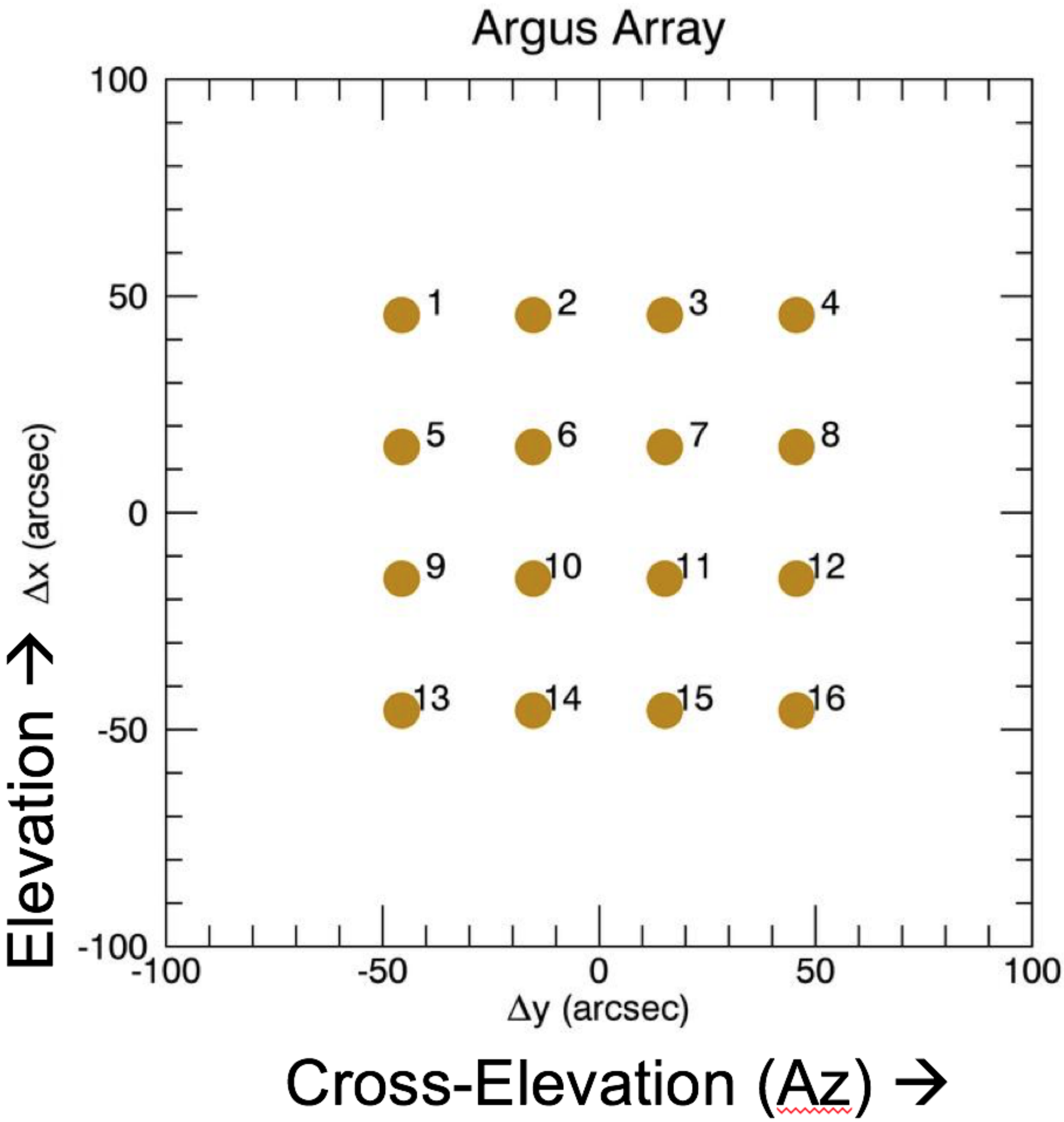
Astrid's Scheduling Block Commands

4.2.1 Argus

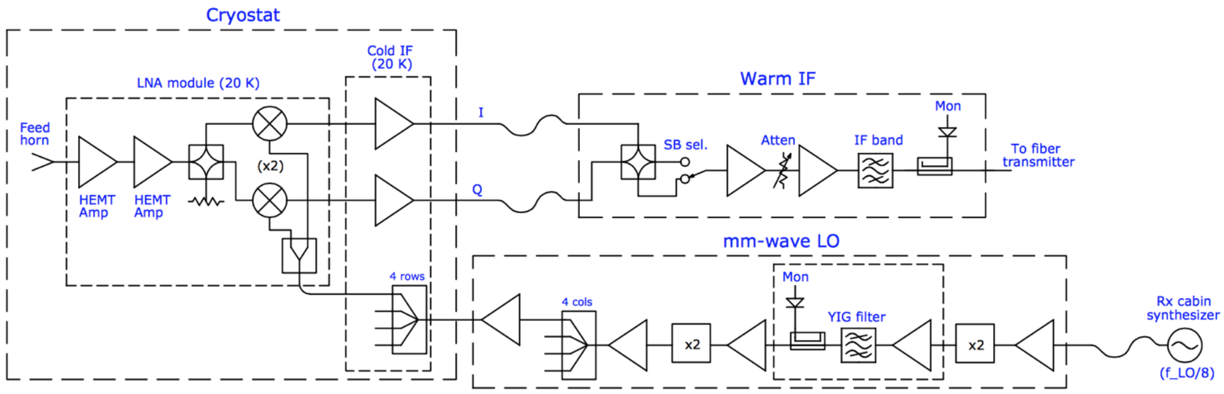
> The documentation below is a copy of the content in the Observer Guide. This is still a big mix-match of Tutorial, How-To Guide, and Reference Guide and will need to get sorted out.

4.2.1.1 Introduction

Argus is a 16 element focal-plane-array operating from 74 GHz o 116 GHz. The feeds are configured in a 4x4 array and are each separated by 30.4 arcsec on the sky.



Argus has an absorber vane that can be placed over the entire array for calibration. The 16 Argus beams can be connected to the 16 separate VEGAS channels (VEGAS has 8 bank, each of which have two channels [A1, A2, B1, B2, . . . , H1, H2]). Each of the 16 beams only measure 1 polarization (linear, X). Argus uses an IQ-mixer scheme to separate the USB and LSB, and the side-band isolation is aproximately 15-20 dB.



The instantaneous bandwidth for Argus is ~1.5 GHz, which is similar to the VEGAS spectrometer bandwidth. Users can observe multiple lines simultaneously using the VEGAS sub=banding modes (modes 20-29) for lines separated by less than the ~1.25 GHz effective bandwidth of an individual VEGAS bank. For spectral line mapping experiments, Argus is typically configured with each of the Argus beams connected to an individual VEGAS channel. BEams 9-16 use the regular GBT IF system and can be configured with multiple VEGAS banks, or the DCR for pointing, focus and OOF. Beams 1-8 have dedicated IF paths that are only connected to specific VEGAS banks.

For the chopper-vane calibration technique that Argus adopts, teh natural temperature scale measures is T_A* (GBT Memo 302). This temperature scale has the advantage of correcting for atmosphericattenuation while its derivation is nearly independent of opacity. Users need to take a vane calibration sequence whenever the configuration changes or whenever the IF system is balanced to calibrate the data.

Argus does not have noise diodes.

4.2.1.2 Configuration

Argus uses the standard config-tool software that automatically configures the system based on user input (e.g., frequency and bandwidth). Example Argus configuration files are given in /home/astro-util/projects/Argus/OBS. The configuration keywords specific to Argus are the following:

```
receiver     = "RcvrArray75_115"
beam         = "all"                # or list the beams separately, e.g., beams="10,11"
swmode       = "tp_nocal"           # or "sp_nocal"
polarization = "linear"
sideband     = "LSB"                # for best performance: LSB < 112 GHz, USB >= 112 GHz
```

4.2.1.3 Observing

The observing strategies for Argus are similar to those presented for the 4mm W-Band receiver. To maximize the telescope efficiency for targets smaller or similar in size to the beam (~8 arcsec), observations should be carried out during the night under stable thermal conditions. Depending on the science goal, successful daytime observations are possible for extended sources, where accurate beam shapes are not as crucial. Example Argus observing scripts are located at /home/astro-util/projects/Argus/OBS. The recommended observing procedures are the following

1. Startup Astrid and go online (with control of the telescope, when given permission by the operator)
2. Run the argus_startup script. This script checks the instrument status, turns ON the instrument if it is currently off and pre-configures Argus for the default 90 GHz parameters.
3. At the start of the observing session, run an AutoOOF to optimize teh surface, unless the exact beam shape is not important for your science goals. This procedure will correct the surface for the current thermal conditions

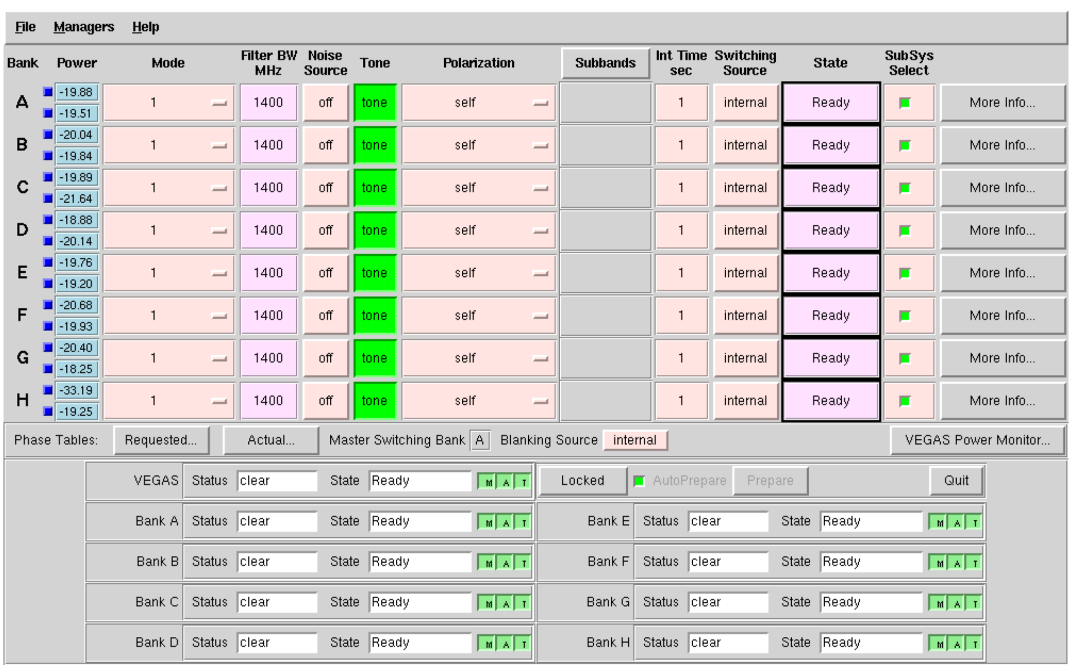
and derive the initial pointing and focus corrections. For AutoOOF it is recommended to use the brightest point source in the sky between 25-80 degrees elevation. If the Ka-Band receiver is available, run the AutoOOF at Ka-Band instead of Argus for more accurate surface corrections. When running AutoOOF with Argus, it is recommended to avoid using the calSeq=False keyword, so the data will be properly calibrated in the fitting for the surface model. The Astrid Observation Management Log will report the system temperatures on the sky from the initial vanecal scans. The same Astrid command "AutoOOF(source)" can be used for any of the receivers that use AutoOOF (i.e., Ka, Q, W-Band, Argus, Mustang-2) and the software will adopt the appropriate defaults for each band.

4. After the AutoOOF solutions are applied, run a point and focus with Argus to confirm the telescope collimation offsets.

5. For Argus, run AutoPeakFocus() with a bright pointing source (>1.5 Jy) within ~ 30 degree of the target region; brighter sources are better than closer sources, since the GBT gregorian pointing model is fairly accurate. Choose a frequency that is the approximate frequency of your science frequency since the YIG filter system can take time to adjust to large frequency shifts. For the best science results, AutoPeakFocu() should be run every 30-50 minutes depending on conditions (point more often during the day and after sunrise and sunset). Avoid pointing in the keyhole (el>80deg). Since the elevation pointing offsets can be larger than those observed typically in azimuth, use the elAzOrder=True keyword to observe the elevatin Peak scans first. An example pointing command showing the usage of the frequency, calSeq, and elAzOrder keyword is AutoPeak(source, frequency=90000., calSeq=False, elAzOrder=True).

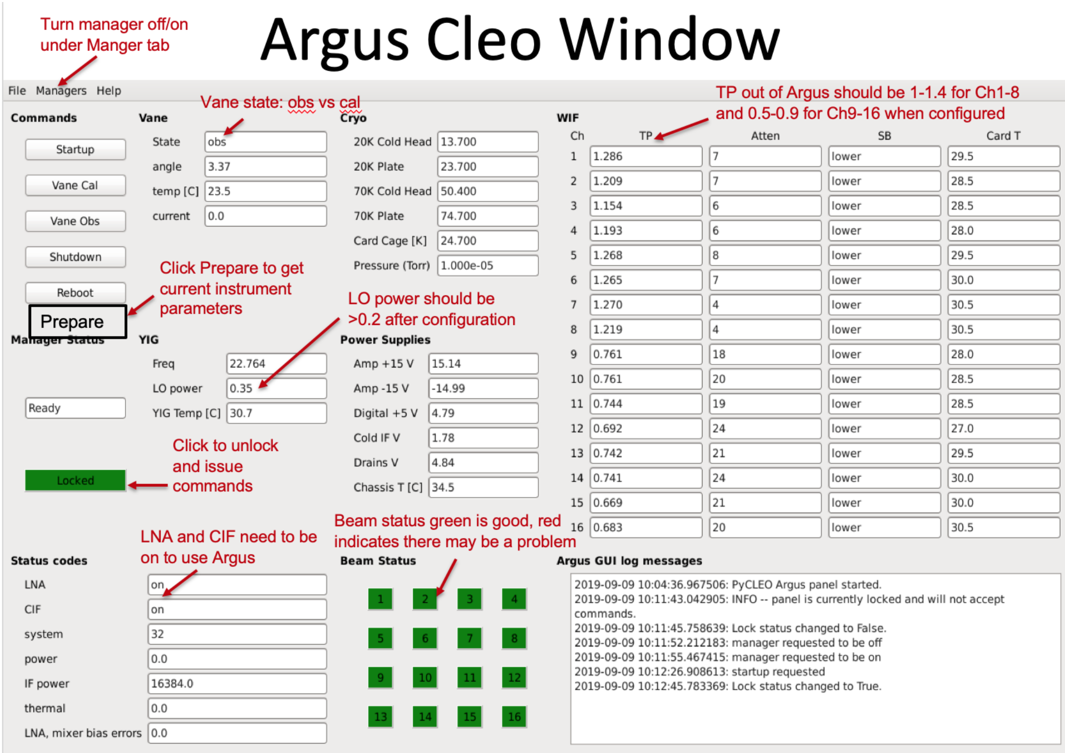
6. If pointing is problematic with Argus, e.g., observations during the day, or in periods of marginal weather, or in cases where the pointing source is too weak, observers can point and focus in X-band and use these telescope corrections for their Argus observations. Also, if there are no nearby bright sources to point with Argus and the telescope is at slow slew rate (at cold temperatures), it can be faster to switch receivers for pointing than to slew far away and point with Argus. Pointing and focus using Argus requires special attention, and users should not blindly accept the default solutions provided by the software system. Users can enter solutions manually as needed as discussed in Section~ref{sec:gfmsendcorrections}. If you are unsure of the Argus pointing results, point in X-band (which is adjacent to Argus in the turret).

7. After configuration and balancing VEGAS for science observations, check the power levels in the system. The VEGAS levels should be ~ -20+/i 3 dBm.



8. The target power levels in the IF rack (for beams 9-16) are 1.5 Volts. The YIG LO power level going into the

instrument should range from ~0.2–0.6 Volts (above 84 GHz). The power coming out of the warm electronics of Argus should read about ~1.0–1.4 for beams 1-8 and ~0.5-0.9 for beams 9-16 (under the TP column of the WIF section of the Argus pyCLEO window.



9. Users must run the argus_vanecal procedure to calibrate the data (located in /users/astro-util/projects/Argus/OBS/argus_vanecal) after each configuration and/or balance for observations that need to be calibrated. The vane calibration is stable over longer periods than is needed for pointing and focusing, so only one argus_vanecal procedure is required for each set of VEGAS observations between the pointing and focus observations. Under stable temperature conditions, the vane calibration is consistent over several hours while it is recommended to point and focus every 30–50 minutes for Argus.

10. It is best to observe similar frequencies together in time since it can take a few minutes for the YIG system to adjust to large frequency jumps. If you need to switch by a large amount in frequency (e.g., >4 GHz), configure and wait a couple of minutes before observing. If the YIG LO power is low after a large frequency shift (e.g., <0.2), re-configure again.

11. Only beams 9-16 that go through the GBT IF Rack can be configured with the DCR. All 16 beams can be configured with VEGAS using the 8 dedicated optical fibers for Argus beams 1-8.

12. Beam-8 has very little sideband rejection and will show higher noise when using the LSB at high frequency (e.g., when the oxygen atmospheric line is in the USB).

13. The "Auto" procedures will run vanecal observations by default. For pointings/focus scans that do not need to be calibrated, observers can use the calSeq=False keyword, e.g., AutoPeak(source, frequency=90000., calSeq=False). The use of the calSeq=False keyword will save a minute or two of time. However, it is recommended to run the vanecal observations while pointing between science blocks of observations in order to track the performance of the system from the calibrated peak scans. If your frequency is not specified, the default frequency for the Argus Auto procedures is 86000 MHz.

14. Beam 10 is the default signal beam and beam 11 is the default reference beam for pointing, focus, and OOF observations.

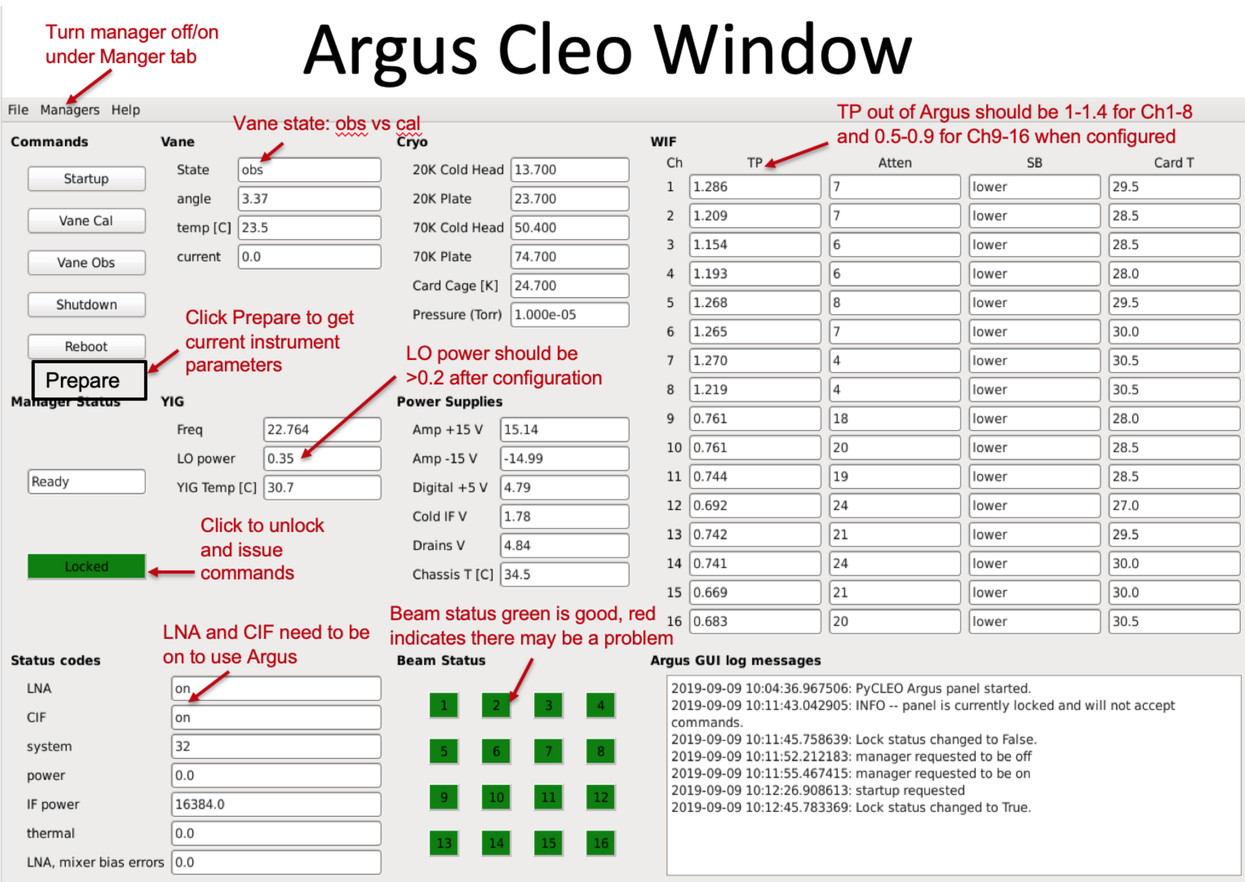
15. The instrument performance using VEGAS can be checked by running the vanecal.pro procedure within

GBTIDL. Example Argus data reduction scripts are located at /users/astro-util/projects/Argus/PRO. The vanecal.pro routine uses the getatmos.pro procedure which derives the opacities and atmospheric conditions from the Green Bank Weather database.

16. For absolute calibration carryout PEAK scans after applying good surface, pointing, and focus corrections for a source of known flux density (e.g., ALMA source catalog (https://almascience.eso.org/sc/). The ALMA calibrator catalog can also be used to check the strength of your pointing/focus source. The calibration methods and performance of the telescope are presented in GBT Memo #302.

4.2.1.4 Monitoring and Diagnostics

The Argus pyCleo page can be used to monitor the status of the instrument.



This tool can be started from the CleoLauncher under the Receivers tab labeled "RcvrArray75_115". The pyCleo can be used for running basic instrument commands, such as startup or the vane control. Before issueing a command, you must unlock the CLEO window by clicking the green "locked" button to unlocked (red). The instrument parameters showin by pyCleo for Argus are updated after a configuration, at the start of each scan during observing, and every 30 minutes when not observing. Updated instrument values can be obtained by issuing a "prepare" command, which is done under the top managers tab (prepare) or the prepare button under the reboot button.

The Beam Status buttons are color coded, where green means the signal associated with the beam is good and red indicates a potential issue with the beam. If a beam is read, the data may still be usable depending on the system temperature associated with the beam.

The Vane state is "obs" when Argus feeds are looking at the sky (with an angle of ~ 3.4) and "cal" when the vane is covering Argus during the Vane calibration scan (angle of ~ 1.6).

The LNA and CIF (low noise amplifiers and cold IF electronics) need to be in the on state to carry out observations. After configuration, the YIG LO power (listed under the YIG section) should be ~ 0.2-0.6 V. The total power levels of the WIF (warm IF electronics) should read ~ 1.0-1.4 for beams 1-8 and ~0.5-0.9 for beams 9-16 after configuration and while observing.

4.2.1.5 IF Routing

The mapping between the VEGAS channels, Converter Modules, IF channels, and the VEGAS beams is shown below

Mapping Argus Beams to VEGAS and IF Channels

VEGAS Bank	VEGAS (J)	Argus Beam	Converter Module CM	IFrack Optical Driver OD	Dedicated Fibers
A1	1	9	1	1	-
A2	2	11	5	3	-
B1	3	10	2	2	-
B2	4	12	6	4	-
C1	5	1	3	-	1
C2	6	3	7	-	3
D1	7	2	4	-	2
D2	8	4	8	-	4
E1	9	13	9	5	-
E2	10	15	13	7	-
F1	11	14	10	6	-
F2	12	16	14	8	-
G1	13	5	11	-	5
G2	14	7	15	-	7
H1	15	6	12	-	6
H2	16	8	16	-	8

Observers should verify the VEGAS power levels ~-20+/-3 dBm via the VEGAS Cleo window.

As an example shown by Figure fig:argusvegas, the VEGAS channel H1 is -33 dBm which is too low to yield useful data. The H1 VEGAS bank corresponds to VEGAS channel J15, converter module CM12, dedicated fiber "6", and Argus beam 6. In this example, the data associated with Beam-6 from Argus would be bad and would show non-physical system temperatures.

4.2.1.6 Troubleshooting Guide

The Argus CLEO window can be used to troubleshoot Argus issues, which can be launched from cleo under the Receivers tab by selecting "RcvrArray75-115"

- To control the instrument unlock the system by selecting the green button "Locked" on left to unlock the window (it turns red when unlocked).
- To get the current status of the instrument click the "Prepare" button which is the last Command listed in the upper left.
- When done, lock the system to avoid accidently issuing a command by clicking the red unlocked button to green (locked).
- Confirm that the CIF and LNA are both on. If off, run the Argus startup script.
- Make sure the vane is in the desired position (e.g., obs for looking at the sky). If the vane is "stalled"or in an unknown state, click the Vane Obs button to move the vane to the obs position. If the vane is not already in the obs position, a configuration will also command the vane to the obs position. If the vane fails to move have the operator contact a support scientist.
- Confirm there is LO power from YIG after configuration.
- The status of the instrument is checked before each scan, and the scan will be aborted if there is not enough YIG LO power, or for other major issues. If the YIG power is too low, or the WIF power levels are low, and/or if one or more of the beam Status colors are red, reconfigure. If one or more the beams are bad, observations with the remaining beams can continue, but one must have sufficient YIG LO power to carry out Argus observations.

- Sometimes the GBT M&C system will report old Argus errors when everything is working. You can ignore and continue observing, or try to clear the lingering error messages with the following procedure.
  - "Prepare" to retrieve updated instrument paramaters which may clear the error.
  - If the error persists, turn the manager off and back on from the Managers pull-down menu at the top of the CLEO window. Click off, wait a couple of seconds, and then click on.
  - This usually clears the error messages. Sometimes the error message(s) need to be cleared manually from the messageMux system by the software group.
- If Argus communication errors occur (e.g., Netburner time out error), then the recent commands given to Argus may not have been done and you may need to re-configure and re-issue your observing script. Within the Argus CLEO window, click the "Prepare" button to collect the current state of the instrument. If the LNA/CIF are off under the Status Codes, run the Startup script and then reconfigure. Turn the manager off and back on again to clear the Netburner errors.
- If Argus is in a "fault" state after configuration and you are unable to collect data after multiple attempts then:
  - Turn manager off and back on again (under the Managers tab at the top of the Argus CLEO window) and reconfigure.
  - If cycling the manager does not work, have the operator restart turtle and/or "grail" and reconfigure.
  - If neither of the above work, then have the operator contact a support scientist.

4.2.1.7 Data Reduction

Argus is calibrated on the T_A* antenna temperature scale. Observations need to run a set of vanecal observations for each set of science data. For absolute flux calibration, a source of known flux density should be observed. The ALMA Calibrator Source Catalog has an extensive record of the flux density histories for many of the bright 3mm point sources (https://almascience.eso.org/sc/). By using ALMA flux density values as a function of time, ~10% absolute calibration uncertainties can be obtained for Argus data. The methods and equations for calibrating Argus data are presented in GBT Memo #302.

The standard GBTIDL scripts (getps, getnod, getfs) do not work with Argus data, since these assume a noise diode for calibration. Example Argus scripts for the reduction of spectral line data can be found at /home/astro-util/projects/Argus/PRO. Users can use the vanecal.pro procedure within GBTIDL to derive the Tcal value and the system temperatures for each of the Argus beams. Frequency switched data can be reduced using argus_fsw.pro, and position switch data can be reduced using argus_onoff.pro.

4.2.1.8 Documentation

- **Argus Web Page**: http://www.gb.nrao.edu/argus/
- **GBT Calibration Memo**: https://library.nrao.edu/public/memos/gbt/GBT_302.pdf
- **Argus configuration and observing scripts which are used in Astrid**: /home/astro-util/projects/Argus/OBS
  - **argus_startup**: Script that turns Argus on if it is not already on. The script configures the instrument with the default settings. This is run at the start of an Argus observing session.
  - **argus_vanecal**: Script that runs the vanecal observations. It observes with the vane over the array as well as blank sky scan with a default 6 arcmin offset from the commanded position to avoid a bright calibrator object. If observing the Moon or a very bright extended continuum source, one can use the argus_vanecal_bigoffset2 or argus_vanecal_bigoffset to observe blank sky.
  - **autooof**: Script that runs the AutoOOF observations. The sources listed are the brightest W-band sources in the sky.

- **autopeak_focus**: Script that runs pointing and focus observations. The pointing observations are run first, and then the script issues a break to allow the user to enter the solutions manually into the system before the focus scan.
- **autopeak_calibrate**: Calibration script to run on a known calibrator to compute the aperture and main-beam efficiency of the telescope after good pointing and focus corrections have been applied.
- **argus_config_example**: Example total power configuration (tp_nocal) for Argus.
- **mapRA**: Example frequency switching (sp_nocal) observing script for a RA/Dec map.

- *GBTIDL reduction scripts*: /home/astro-util/projects/Argus/PRO
  - **getatmos.pro**: Script that returns the atmospheric opacity and effective atmospheric noise and temperature for a specific time and frequency from the Green Bank Observatory weather database. This needs to be run on a Green Bank computer since using special code that only runs locally.
  - **vanecal.pro**: Script that computes the system temperature for each of the Argus beams from the vanecal observations. The script uses getatmos.pro to compute the Tcal value (see GBT Memo#302).
  - **argus_fsw.pro**: Script that calibrates a frequency switched observation.
  - **argus_onoff.pro**: Script that calibrates a position switched observation.
- **ALMA Source Catalog**: https://almascience.eso.org/sc/

4.2.2 Scheduling Block Commands

DecLatMap()

Balance()

A utility scan. The Balance() command is used to balance the electronic signal throughout the GBT IF system so that each devise is operating in its linear response regime. Balance() will work for any device with attenuators and for a particular backend. Individual devices can be balanced such as Prime Focus receivers, the IF system, the DCR, and VEGAS. The Gregorian receivers lack attenuators and do not need to be balanced. If the argument to Balance() is blank (recommended usage),, then all devices for the current state of the IF system will be balanced using the last executed configuration to decide what hardware will be balanced.

Advanced Syntax

Use only if you really know what you're doing.

```
Balance('DeviceName', {'DeviceKeyword': Value})
```

Examples

```
# example showing the expected use of Balance()

# load configuration
execfile('/home/astro-util/projects/GBTog/configs/tp_config.py')
Configure(tp_config)

# Slew to target so that you may balance "on source"
Slew('3C286')

# Balance IF and devices for specified configuration
Balance()
```

BalanceOnOff(*location*, *offset=None*, *beamName='1'*)

A utility scan. When there is a large difference in power received by the GBT between two positions on the sky, it is advantageous to balance the IF system power levels to be at the mid-point of the two power levels. Typically this is needed when the "source position" is a strong continuum source. This scan type has been created to handle this scenario; one chouls consider using it when the system temperature on and off source differ by a factor of two or more.

The BalanceOnOff() slews to the source position and then balances the IF system. It then determines the power levels that are observed in the IF Rack. Then the telescope is slewed to the off position and the power levels are determined again. The change in the power levels is then used to determine attenuator settings that put the balance near the mid-point of the observed power range. Note that the balance is determined only to within +/-0.5 dB owing to the integer settings of the IF Rack attenuators.

Parameters

- **location** – A Catalog source name or Location object. It specifies the source of which the telescope should slew. The default is the current location in "J2000" coordinate mode.
- **offset** (*Offset object*) – It moves the beam to an optional offset position that is specified relative to the location specified in the location parameter value.
- **beamName** (*str*) – It specifies the receiver beam to use for the scan. beamName can be 'C', '1', '2', '3', '4', or any valid combination for the receiver you are using such as 'MR12'.

Examples

```
# example showing the expected use of Balance()

# load configuration
execfile('/home/astro-util/projects/GBTog/configs/tp_config.py')
Configure(tp_config)

# Balance IF and devices for specified configuration
BalanceOnOff('3C48', Offset('J2000', 1.0, 0.0))
```

DecLatMap(*location*, *hLength*, *vLength*, *hDelta*, *scanDuration*, *beamName='1'*, *unidirectional=False*, *start=1*, *stop=None*)

A Declination/Latitude map, or DecLatMap, does a raster scan centered on a specific location on the sky. Scanning is done in the declination, latitude, or elevation coordinate depending on the desired coordinate mode. This procedure does not allow the user to periodically move to a reference location on the sky, please see DecLatMapWithReference for such a map. The starting point of the map is at (-hLength/2, -vLength/2).

Focus(*location*, *start=None*, *focusLength=None*, *scanDuration=None*, *beamName=None*, *refBeam=None*)

A Utility Scan. Focus scan type moves the subreflector or prime focus receiver (depending on the receiver in use) through the axis aligned with the beam. Its primary use is to determine focus positions for use in subsequent scans.

Parameters

- **location** – Catalog source name or Location object. It specifies the source upon which to do the scan.
- **start** (*float*) – Specifies the starting position of the subreflector (in mm) for the focus scan.
- **focusLength** (*float*) – Specifies the ending position of the subreflector relative to the starting location (in mm).

- **scanDuration** (*float*) – Specifies the length of the scan in seconds. The default value is the recommended value for the receiver.
- **beamName** (*str*) – Specifies the receiver beam to use for measuring the focus. Make sure that you configure with the same beam with which you focus.
- **refBeam** (*str*) – Specifies the reference receiver beam to use for the receivers with more than one beam.

Examples

```
# Focus using default settings and calibrator 0137+3309
Focus('0137+3309')

# Focus from -200 to 200mm at 400mm/min with beam 1
Focus('0137+3309', -200.0, 400.0, 60.0, '1')
```

Nod(*location*, *beamName1*, *beamName2*, *scanDuration*)

The Nod procedure does two scans on the same sky location with different beams.

> **Caution:** Nod should only be used with multi-beam receivers.

Parameters

location – A Catalog source name or Location object. It specifies the source upon which to do the Nod.

beamName1: str

It specifies the receiver beam to use for the first scan. beamName1 can be '1' or '2'or any valid beam for the receiver you are using.

beamName2: str

It specifies the receiver beam to use for the second scan. beamName2 can be 'C', '1', '2' or any valid beam for the receiver you are using.

scanDuration: float

It specifies the length of each scan in seconds.

Examples

```
# Nod between beams 3 and 7 with a 60s scan duration
Nod('1011-2610', '3', '7', 60.)
```

OffOn(*location*, *referenceOffset*, *scanDuration=None*, *beamName='1'*)

The OffOn scan type is the same as the OnOff scan except that the first scan is offset from the source location.

Parameters

- **location** – A Catalog source name or Location object. It specifies the source upon which to do the "On" scan.
- **referenceOffset** (*Offset object*) – Is specifies the location of the "Off" scan relative to the location specified by the first parameter.
- **scanDuration** (*float*) – It specifies the length of each scan in seconds.

- **beamName** (*str*) – It specifies the receiver beam to use for the scan.

Examples

```
# OffOn scan with reference offsets of 1 degree in RA and 1 deg in Dec with a 60 s␣
↪scan duration using beam 1.
OffOn('0137+3309', Offset('J2000', 1.0, 1.0, cosv=False), 60, '1')
```

OnOff(*location*, *referenceOffset*, *scanDuration=None*, *beamName='1'*)

The OnOff scan type performs two scans. The first scan is on source, and the second scan is at an offset from the source location used in the first scan.

Parameters

- **location** – A Catalog source name or Location object. It specifies the source upon which to do the "On" scan.
- **referenceOffset** (*Offset object*) – Is specifies the location of the "Off" scan relative to the location specified by the first parameter.
- **scanDuration** (*float*) – It specifies the length of each scan in seconds.
- **beamName** (*str*) – It specifies the receiver beam to use for the scan.

Examples

```
# OnOff scan with reference offsets of 1 degree in RA and 1 deg in Dec with a 60 s␣
↪scan duration using beam 1.
OnOff('0137+3309', Offset('J2000', 1.0, 1.0, cosv=False), 60, '1')
```

OnOffSameHA(*location*, *scanDuration*, *beamName='1'*)

The OnOffSameHA scan type performs two scans. The first scan is on the source and the second scan follows the same HA track used in the first scan.

Parameters

- **location** – A Catalog source name or Location object. It specifies the source upon which to do the "On" scan.
- **scanDuration** (*float*) – It specifies the length of each scan in seconds.
- **beamName** (*str*) – It specifies the receiver beam to use for both scan.

Examples

```
# OnOffSameHA scan with 60s scan duration using beam 1.
OnOffSameHA('0137+3309', 60, '1')
```

Peak(*location*, *hLength=None*, *vLength=None*, *scanDuration=None*, *beamName=None*, *refBeam=None*, *elAzOrder=False*)

A utility scan. Peak scan type sweeps through the specified sky location in the four cardinal directions. Its primary use is to determine pointing corrections for use in subsequent scans.

Parameters

- **location** (*str*) – Catalog source name or Location object. It specifies the source upon which to do the scan.
- **hLength** (*Offset object*) – Specifies the horitzontal distance used for the Peak. hLength values may be negative. The default value is the recommended value for the receiver.
- **vLength** (*Offset object*) – Specifies the vertical distance used for the Peak. vLength calues may be negative. The default value is the recommended value for the receiver.
- **scanDuration** (*float*) – Specifies the length of each scan in seconds. The default value is the recommended value for the receiver.
- **beamName** (*str*) – Specifies the receiver beam to use for the scan. Make sure that you configure with the same beam with which you Peak.
- **refBeam** (*str*) – Specifies the reference receiver beam to use for the receivers with more than one beam.
- **elAzOrder** (*bool*) – If True, the elevation peak scans will be done first before the azimuth peak scans. This can be helpful for high-frequency observations (>40 GHz) since the elevation pointing offsets are typically larger than the azimuth offsets.

Caution: hLength, vLength and scanDuration should be overridden as a unit since together they determine the rate.

Examples

```
# Peak using default settings and calibrator 0137+3309
Peak('0137+3309')

# Peak using encoder coordinates with scans of 90' length in 30s
Peak('0137+3309', Offset('Encoder', '00:90:00', 0), Offset('Encoder', 0, '00:90:00
↪'), 30, '1')
```

Slew(*location=None*, *offset=None*, *beamName='1'*, *submotion=None*)

A utility scan. Slew moves the telescope beam to point to a specified locaiton on the sky without collecting any data.

Parameters

- **location** – Catalog source name or Location object. It specifies the source to which the telescope should slew. The default is the current location in "J2000" coordinate mode.
- **offset** (*Offset object*) – Moves the beam to an optional offset position that is specified relative to the location specified in the location parameter value.
- **beamName** (*str*) – Specifies the receiver beam to use for the scan. beamName can be 'C' (center), '1', '2', '3', '4' or any valid combination for the receiver you are using such as 'MR12' (i.e. track halfway between beams 1 and 2).

Note: Once Slew() is complete, the location will continue to be tracked at a sidereal rate until a new command is issued.

Examples

```
# Antenna slews to 3C48 with beam 1
Slew('3C48', beamName='1')

# Slew to 3C48 plus offset
Slew('3C48', ADD OFFSET)

# Slew to offset from current location
Slew(ADD OFFSET)
```

`SubBeamNod`(*location*, *scanDuration*, *beamName*, *nodLength*, *nodUnit='seconds'*)

For multi-beam receivers SubBeamNod causes the subreflector to tilt about its axis between two feeds at the given periodicity. The primary mirror is centered on the midpoint between the two beams. The beam selections are extracted from the scan's beamName, i.e. 'MR12'. The "first" beam ('1') performs the first integration. The periodicity is specified in seconds per nod (half-cycle). A subBeamNod is limited to a minimum of 4.483 s for a half cycle.

Parameters

`location` – A Catalog source name or Location object. It specifies the source upon which to do the Nod.

scanDuration: float

It specifies the length of each scan in seconds.

beamName: str

It specifies the receiver beam pair to use for nodding. beamName can be 'MR12'.

nodLengt: depends on unit of nodUnit (int for 'integrations', float or int for 'seconds')

It specifies the half-cycle time which is the time spent in one position plus move time to the second position.

nodUnit: str

Either 'integrations' or 'seconds'.

Examples

```
# nodLength units in second
SubBeamNod('3C48', scanDuration=60.0, beamName='MR12', nodLength=4.4826624)

# nodLength units are 'tint' as set in the configuration
SubBeamNod('3C48', scanDuration=60.0, beamName='MR12', nodLength=3, nodUnit=
↪'integrations')
```

Hint: The scan will end at the end of the scanDuration (once the current integration is complete) regardless of the phase of the nod cycle. When the subreflector is moving the entire integration during which this occurs is flagged. It takes about 0.5 seconds for the subreflector to move between beams plus additional time to settle on source (total time is about ~1.5 second).

For example, if we had previously configured for Rcvr26 40 and an integration time of 1.5 sec- onds (tint=1.5 in the configuration), example 2 in script 7.49 would blank roughly one out of every three integrations in a half-cycle (nodLength=3) while the subreflector was moving between beams. If nodLength=5, then only one in five integrations would be blanked. A resonable compromise in terms of performance and to minimize the amount of data blanked is to use subBeamNod with an integration time of 0.2s and a nodLength=30 (6 sec nodding between beams). It is important to use a small tint value to avoid blanking too much data (e.g., 0.5 sec or less).

The subBeamNod mode is useful to produce good baselines for the measurement of broad extra-galactic lines. For Ka-band, Q-band, and Argus, the performance of subBeamNod is signficantly better than Nod observations. For W-band and the KFPA, the beams are farther apart and the subBeamNod technique does not work as well, and users are recommended to use Nod observations.

The antenna uses the average position of the two beams for tracking the target, and SDFITS reports the positions of the beams relative to the tracking position. Although the SDFITS header postion will not match the target position, SubBeamNod successfully nods between the two beams during the scan. Control of the subreflector may be done with any scan type using the submotion class. This should only be done by expert observers. Those observers interested in using this class should contact their GBT "Friend".

`Tip`(*location*, *endOffset*, *beamName='1'*, *scanDuration=None*, *startTime=None*, *stopTime=None*)

A utility scan. Tip scan moves the beam on the sky from one elevation to another elevation while taking data and maintaining a constant azimuth. It is recommended to tip from 6 deg to 45 deg as the atmosphere will not change significantly above 45 deg.

Parameters

- **location** – Catalog source name or Location object. It specifies the source upon which to do the scan.
- **endOffset** (*Offset object*) – Specifies the beam's final position for the scan, relative to the location specified in the first parameter. The offset also must be in AzEl or encoder coordinates.
- **beamName** (*str*) – Specifies the receiver beam to use for the Tip. beamName can be 'C' (center), '1', '2', '3', '4' or any valid combination for the receiver you are using such as 'MR12' (i.e. track halfway between beams 1 and 2).
- **scanDuration** (*float*) – Specifies the length of the scan in seconds.
- **startTime** (*time str 'hh:mm:ss'*) – Allows the observer to specify a start time for the Tip.
- **stopTime** (*time str 'hh:mm:ss'*) – Allows the observer to specify a stop time for the Tip.

Note: The scan time may be specified by either a scanDuration, a stopTime, a startTime plus stopTime or a startTime plus scanDuration.

Examples

```
# Tip the GBT from 6 deg in elevation to 45 deg over a period of 5 min using beam 1
Tip(Location('AzEl', 1.6, 6.0), Offset('AzEl', 0.0, 39.0), 300.0, '1')
```

`Track`(*location*, *endOffset*, *scanDuration=None*, *beamName='1'*, *submotion=None*, *startTime=None*, *stopTime=None*, *fixedOffset=None*)

The Track scan type follows a sky location while taking data.

Parameters

- **location** – A Catalog source name or Location object. It specifies the source which is to be tracked.
- **endOffset** (*Offset object*) – Supplying an endOffset object with a value other than None will track the telescope across the sky at constant velocity. The scan will start at the specified location and end at (location+endOffset) after scanDuration seconds. If you wish to only

track a single location rather than slew the telescope between two points, use None for this parameter.

- **beamName** (*str*) – It specifies the receiver beam to use for the scan. beamName can be ‘C’, ‘1’, ‘2’, ‘3’, ‘4’, or any valid combination for the receiver you are using such as ‘MR12’.
- **startTime** (*Time object*) – This specifies when the scan begins in the Universal Time (UT). If startTime is in the past then the scan starts as soon as possible with a message sent to the scan log. If (startTime + scanDuration) is in the past, then the scan is skipped with a message to the observation log. The value may be.
- **stopTime** (*Time object*) – This specifies when the scan completes. If stopTime is in the past then the scan is skipped with a message to the observation log.
- **fixedOffset** (*Offset object*) – Track follows the sky location plus this fixed Offset. The fixedOffset may be in a different coordinate mode than the location. If an endOffset is also specified, Track starts at (location+fixedOffset), and ends at (location+fixedOffset+endOffset). The fixedOffset and endOffset must be both of the same coordinate mode, buyt may be of a different mode than the location. The fixedOffset parameter may be omitted.

Note: Scan timing must be specified by either a scanDuration, a stopTime, a startTime plus stopTime, or a startTime plus scanDuration.

Examples

```
# Track 3C48 for 60 seconds using the center beam
Track('3C48', None, 60.0)

# Track a position offset by 1 degree in elevation
Track('3C48', None, 60.0, fixedOffset=Offset('AzEl', 0.0, 1.0))

# Scan across the source from -1 to +1 degrees in azimuth
Track('3C48', Offset('AzEl', 2.0, 0.0), 60.0, fixedOffset=Offset('AzEl', -1.0, 0.0))
```

PYTHON MODULE INDEX

a

INDEX