Subject: Radio Astronomy Blocks for Gnuradio Companion
Memo: 20, Revision 1
From: Glen Langston
Date: 2018 April 17

Summary: New Gnuradio Companion Blocks for Radio Astronomy are presented. These simple-to-use blocks are the starting point for Science Aficionados creating their own Radio Astronomy Observatories.

The Science Aficionados can now build their own Radio Astronomy Telescope for charting the structure of the Milky Way, based on Gnuradio (See https://www.gnuradio.org) blocks. We present new blocks that record observations and are also resistant to short duration radio interference, often seen in city locations. Also these new blocks are free to download. The blocks calibrate astronomical observations, so the galaxy can be seen in just a few seconds of observations. The Aficionados will record their observations using ascii data writing block, so they can map the spiral structure of our Milky Way Galaxy. The spectral line data format is described in LightWork Memo 18 and the mapping software, for reducing the observations to images of the sky, is described in LightWork Memo 19.

Gnuradio Companion Considerations

The Gnuradio Companion (GRC) is provided as a standard part of the Gnuradio installation, so is relatively easy to obtain (see https://wiki.gnuradio.org/index.php/InstallingGR). GRC is fun-to-use visual programming tool. There are many guides online for using different aspects of GRC (see https://wiki.gnuradio.org/index.php/GNURadioCompanion for instance). Here the new radio astronomy blocks are presented (for background on Radio Astronomy see: https://www.cv.nrao.edu/course/astr534/Introradastro.html). Radio Astronomy is also introduced in a Great Course by Dr. Jay Lockman[1].

For a very short, and enjoyable, introduction to GRC take a look our Youtube “Nsf Listens” video (https://bit.ly/2HsFndr), staring Sophie, Evan and myself.

These new blocks were written to enable Aficionados to create their own receiver systems, and replace my first generation of python Gnuradio astronomy software (NsfWatch.py, see LightWork Memos 4 and 14). While the original software worked well, the features I used were not compatible with the latest revisions of Gnuradio, so the original version was falling out of date. Also there were some apparent differences between performance on Macintosh and Linux systems. The new GRC-compatible design allows Aficionados to follow the latest updates to Gnuradio. The new blocks work equally well on Linux and Mac operating systems.

GRC can organize data into simple streams of numbers into *vectors,* which are groups of numbers. For radio astronomy we often group data into vectors to perform a Fourier Transform, to look at the Universe as a series of tones at different frequencies. Because the astronomical are very efficiently studied as frequencies, will process the data as groups of vectors.

Before starting on the details of the new software, we first take a step back and remember where we are in the Milky Way.

---

[1]https://www.thegreatcourses.com/courses/radio-astronomy-observing-the-invisible-universe.html

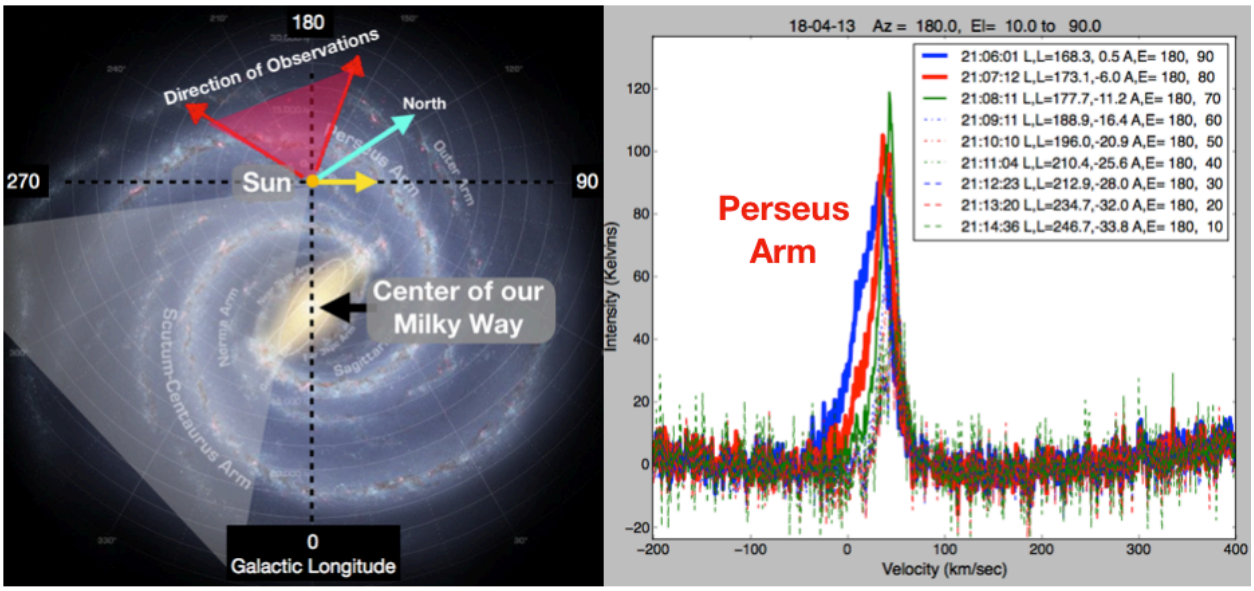

Figure 1: Overview of our place in the Milky Way Galaxy (Left) and 10 Minutes of Observations of the Perseus Arm. *The sketch at left shows our Sun (and us) far from the center of the Milky Way. The image was drawn as if we are way above our galaxy. Our galaxy is a disk and the coordinate of the center of our galaxy is at Galactic Longitude = 0. The galactic longitude, latitude coordinates are centered on us. The plot at right shows 10 beautiful minutes of data. With some research, you can figure out that you've discovered the Perseus Arm of our Galaxy. The plot shows calibrated intensity (Kelvins) versus the velocity of the hydrogen measured. The observations were taken with telescope Azimuth and different Elevations (A,E =). The GRC block calculates the Longitude and Latitude (L,L=) for the time of the observations (21:06 to 21:14 UTC).*

Overall Goal: Observations of the Milky Way

The goal of these new software updates is to make it easy for Science Aficionados to create their own telescope and add their own improvements to the receiver software. As noted in LightWork Memo 14, the entire Milky Way galaxy is shining with radio light, generated by hydrogen atoms. Since there is a vast amount of hydrogen in our galaxy, the structure of the galaxy is easily visible with a home-built radio telescope. The Aficionados can use the LightWork memo series to build their own telescopes.

To set the stage for your observations, look at **Figure 1**, which shows a cartoon of the Milky Way, with the Earth and Sun (and You!) marked with an Orange dot. A glance at the figure shows we're way out from the center of our galaxy. Since we're all in this together, when talking about the Sun, Earth and people, we'll say "we". In the left figure, the green arrow shows the direction the we're moving.

You need to know a little about coordinate angles to understand your observations. **Figure 1** has the Galactic coordinate system. The figure shows the Galactic plane, where the Latitude is zero. The different directions of the sky show different Galactic Longitudes. Zero degrees Galactic Longitude is the direction of the center of our galaxy and 180 degrees longitude is the opposite direction, out of our galaxy. The plot in **Figure 1** show observations out of our Galaxy, in the Longitude Range 168 to 246 degrees. **Figure 1** shows a range of Galactic Latitudes,

from Latitude = 0 and moves up, out of the Galaxy to Latitude = 34 degrees. The hydrogen signal weakens as the telescope points away from the galactic plane. We're all moving in a circle around the center of our galaxy, at 90 degrees Galactic Longitude. The Milky Way is a disk, and the most hydrogen emission is at Galactic Latitude = 0, the galactic plane. We're moving round the center of the galaxy, and our direction is marked with a yellow arrow. The Earth is tilted at an odd angle relative to our galaxy, and the North Pole is roughly where the blue arrow points. Actually, our north pole is pointed a little out of the plane of our galaxy, but for now, the blue arrow is close enough.

The nice graphic in **Figure 1** was drawn by Robert Hurt, (http://www.spitzer.caltech.edu/mission/profile/50-Robert-Hurt), sketching our position relative to the center of our galaxy. This plot shows what we think our galaxy looks like, if we were viewing the galaxy from far, far above.

The point of this memo is to motivate you to build your own observatory, and the plot on the right side of **Figure 1** shows how easy is to see our galaxy with a radio telescope. This clear signal is from the Perseus Arm of our galaxy, and took only 10 minutes of observations with a home-built telescope and this software. In order to make sense of the data, you need to know the direction the telescope was pointing. For all these observations the telescope was pointed due south, Azimuth = 180 degrees. I moved the telescope, by hand, in elevation (that is up and down) from 90 degrees, straight up, down near the horizon, 10 degrees elevation. Each observation was actually only about 30 seconds long, with 30 seconds more needed to point the telescope to a new elevation. (Our north pole direction is actually Galactic Longitude = 123 degrees, Galactic Latitude = 27 degrees.)

Now there are 9 different observations on the right side of **Figure 1.** It happens the earth was rotated so that elevation of 90 degrees had the telescope pointed right along the plane of our galaxy, and the hydrogen intensity is strong. As I moved the telescope down in elevation, the telescope was pointed out of our galaxy and the hydrogen signal decreased, and also the peak velocity changed. We'll talk much more about velocity in a later memo. The point to take away from **Figure 1**, is that you can see the Perseus Arm of the galaxy in a just a few minutes and start to discovery the our motion around the center of our galaxy.

These observations were made at the frequency of hydrogen atoms, 1420.4 MHz. Other atoms and molecule each shine at different frequencies, according to the structure of the atoms and the rules of quantum mechanics. It is because all the atoms and molecules that make us follow the same rules as atoms and molecules in the rest of the universe that we can understand the universe. The fact that the universe has only one set of rules is really a miracle…

Setting Astronomical Observations: Ra_Integrate

This section describes the observer interface created through the GRC system, assisted with new GRC blocks. The Aficionados must tell the system which way the telescope is pointed (Azimuth and Elevation) plus the observing setup, that is the Frequency and Bandwidth (MHz). The interface is built entirely using GRC. The Aficionados also want to take credit for their observations, so record their names in the "Obs:" field and Telescope name in the "Tel:" field. We're hoping that groups of Science Aficionados will get together to run their observations, so will need to give each telescope a unique name.

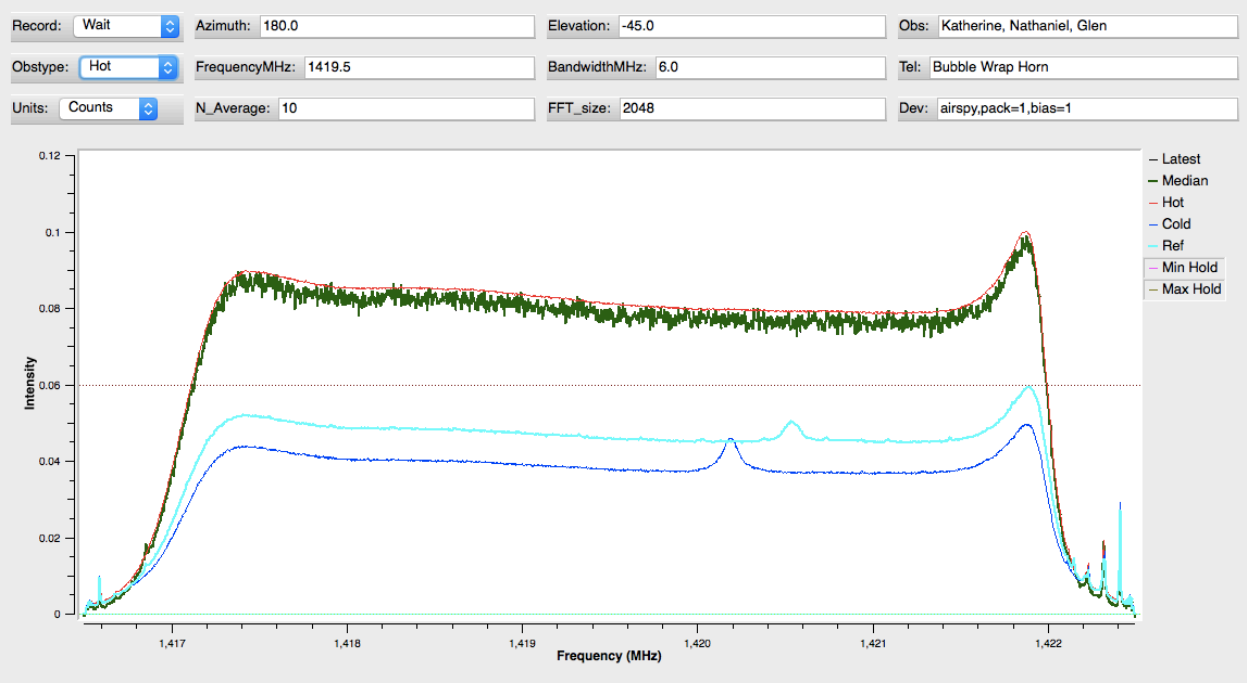

Figure 2: Plot of Raw Intensity (counts) versus Frequency in MHz for spectra selected using the RAI block. *The RAI block is used to confirm proper operation of the telescope. The observer inputs the direction the telescope is pointing and frequency setup information. The plots show intensity versus Frequency (MHz). The intensity axis has three possible units, 1) counts, 2) power in dB and 3) calibrated intensity in Kelvins. This plot shows the intensity in counts. The little peaks new 1420.4 MHz are due to galactic hydrogen, which is clearly visible in these observations. Each of these observations are only a few seconds long.*

The Aficionados need to keep track of a few important points. One is the type of software defined radio (SDR) Device they are using. In **Figure 2,** we are using an AIRSPY with packed data bits and have turned on the voltage to power the first two amplifiers (Dev: airspy,pack=1,bias=1).

The observing setup changes are recorded in a GRC configuration file named Watch.con.

The new data processing blocks are listed below:

- **RA_Integrate (RAI)** : Allows Aficionados to identify spectra for calibration of later observations. This block controls the averaging of the input data stream and selecting different spectra for comparison. Normally this block is only used to confirm the radio telescope is properly operating.
- **RA_Ascii_Sink (RAS)**: The Ascii_Sink block replaces all the data recording functions of the first generation software, NsfWatch.py. The block reads and writes the notes file (Watch.not), which contains the observing setup needed to later calibrate the observations.

The Aficionados, starting out with their own, home-built, radio telescopes will first want to confirm their telescope is working properly. The Ra_Integrate (RAI) block is used to select observations for calibration and record these calibration data for later use. **Figure 2** shows a plot of some test observations. The plots are all created exclusively using the GRC QT Vector

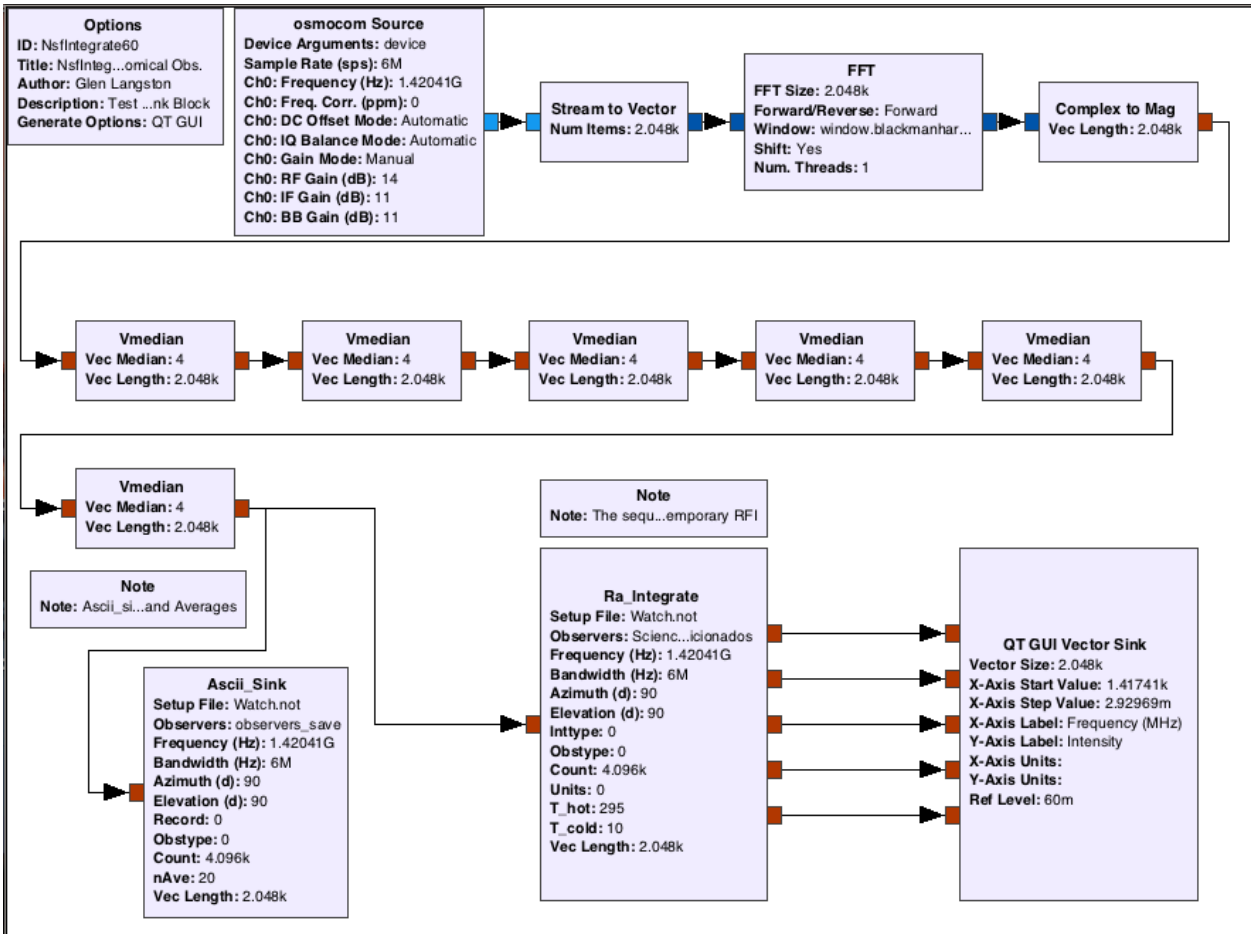

Figure 3: GRC layout of the NsfIntegrate design for Radio Telescope Observations. This is a screen capture of the NsfIntegate60.grc design for use with the AIRSPY SDR. The data flow is simple, from the OSMOSDR source, on the top left, through a block to create a complex vector and a Fourier Transform. The data rate is reduced via a sequence of 6 Vector Median blocks, each that take 4 input vectors and produce a single output vector. These 6 blocks reduce the data rate from one new spectrum every 0.0003 seconds to one spectrum every 1.4 seconds. This reduces the CPU load for plotting and averaging so that all data may be captured with a modest multi-core computer. The filtered data are fed to the data writing block, Ascii_Sink and also to the RAI block and plotter to monitor the observations.

Graphical User Interface (GUI), so the control software should stay consistent with GRC updates.

The Aficionado can collect a huge amount of data with this program, and needs to identify when the telescope is being setup (Record=Waiting), observing normally (Record=Average) or recording special calibration files (Record=Save). After the observations are saved, they will be reloaded each time the observer interface restarts. That way, calibration measurements can be restored, each session.

The first version of the data recording software combined averaging with plotting. That only worked well with slow data rates. The new design does averaging first, before sending data tot the plotter. That allows much more data to be completely captured. The first six median blocks

reduce the data rate by a factor of 4096 and strongly reject short term interference. The **RAS** block does the final averaging step, and takes as input the number of spectra to average, N_Average. The default value is N_Average=10, which corresponds to an average time of 14 seconds, for fftsize of 2048 samples at a 6 MHz sample rate (BandwidthMHz). Using N_Average=86 corresponds to a little over two minutes of averaging time.

The Earth is always rotating and rotates 1 degree every 4 minutes. Since small telescopes have pretty large beams, greater than 15 degrees wide, fairly long integration times can be used. After the initial setup, recording data every 4 minutes is pretty good, by setting N_Average=192.

Each time an Aficionado observes at a new location, they must update the Notes file (Watch.not) with their new Longitude and Latitude location on the Earth. There are functions to set this up automatically. More on setup in a later memo. The computer clock must also be set accurately at least to within a minute of actual time.

Saving the Observations: **Ra_Ascii_Sink (RAS)**

The Aficionado observations are written in simple, ascii, format files. Each of these files includes a summary of the observing setup. Figure 1 shows a plot of the average of a set of ascii files. The file setup information is also written to the notes file, Watch.not, before the observations start. The file is then updated with the time and date of the observations along with the coordinates describing the direction the telescope is pointed.

The post processing software knows how to interpret the observations and average related data. The calibration process is described LightWork Memo 4. The calibration is done by looking at the ground with your telescope (the hot load), and recording a negative elevation. Elevation = -90 indicates your telescope is pointed straight down. A cold load observation is also needed, and the software looks through all your observations for data recorded far from the galactic plane and at high elevation.

Putting the GRC files all together

The GRC system is a great tool for playing with frequencies and time plots. With almost any computer you can create a GRC design that creates interesting tones.

The system we're presenting here is designed to be integrated into your own version of GRC, and the steps to include these files are listed at Github. **Figure 3** shows the GRC design used to record astronomical observations. There are many GRC variables in the full design, and this figure excludes variables to simplify the display. The observing setup is completely recorded via GRC "Variable Config" blocks and the Note file.

The observing setup is completely restored each time GRC is restarted. Also note that the data acquisition design can be directly run as a python program, without the GRC interface, further reducing the CPU load.

Conclusion

This memo has described a new GRC design for astronomical observations that allow sensitive observations of the Milky Way. It takes a while to learn how to use GRC and to build your own radio telescope, but taking these steps allows you to see the Universe from your own back yard.

The telescope and software are sensitive enough to see our Galaxy. With 10 minutes of observations with your sensitive radio telescope and capable control system, you can start to discover our galaxy. Give these commands a try with your own data!

Thanks to my family and friends their support for this project.

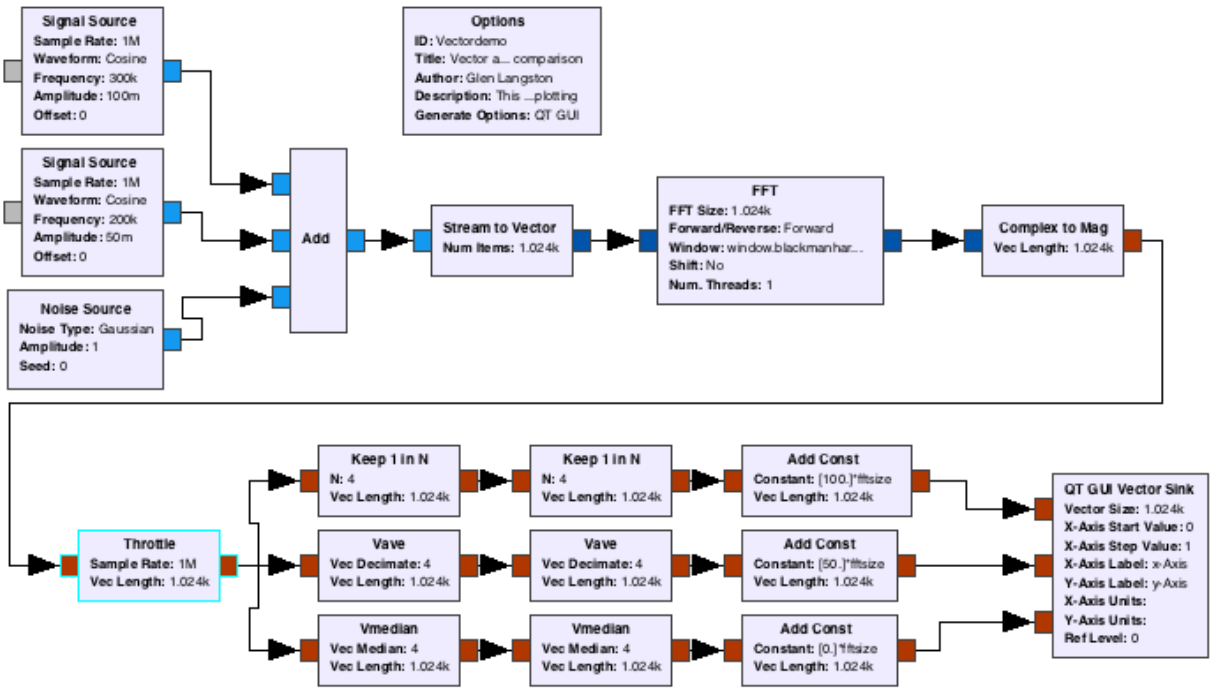

Figure A.1: GRC design Vectordemo.grc. *This figure shows a screen capture of the GRC design used to test both the vector average and median blocks. This design was used to produce Figure 1. The test sources bottom curve shows the Vector Median output after the spectra pass through two median blocks, each with decimation factor of 4. The middle plot shows the average of spectra passing through two vector average blocks each with a decimation factor of 4. The top plot shows a single input spectrum before passing through the vector processing blocks. counts. The points to note are first) both the median and average blocks significantly reduces the noise in the spectra. Note second) that the median does not show much more noise than the average block. The median block is much more resistant to short term interference, with only a small noise penalty.*

Appendix A Median Filter Test Setup

A key feature of the new GRC blocks is that they can be effectively tested through simulated spectra built within GRC. The vector median and average blocks were tested with a GRC design used to create a time sequence at a rate of 1 MHz. This test sequence consisted of three signals, 1) 0.3 MHz complex sine wave with amplitude of 0.15 units. The second test signal 2) was a 0.2 MHz complex sine wave with amplitude of 0.075 units. The third component of the test signal 3) was a gaussian distributed noise source with amplitude of 1.0. The noise source nominated the time sequence and the sin waves were barely visible.

The average blocks are used to reduce the noise, while preserving the wave signals. The design is shown in **Figure A.1**.

Testing New Blocks: **VA** and **VM**

The first new GRC blocks I created are an advance on existing GRC capabilities for observing average spectra in the presence of interference. These blocks are very simple vector processing blocks:

- **Vector Average (VA)**: Average a number of input vectors and output a single vector
- **Vector Median (VM)**: Median average a number of input vectors and output a single vector

Both of these blocks take a single input vector stream and output filtered vectors at a slower rate, after either averaging or median filtering the input vectors. These are both "decimation" blocks. The decimation factor reduces the computer load after the block. A series of these blocks further reduces the computer load.

The **VM** block is implemented by waiting for GRC to gather **N** vectors (usually 4) and then finding the min and max of each of **N** vectors for each of the many channels (usually 1024 or 2048 channels), then subtracting the min and max from the average of all **N** values. The VM block is exactly the mathematical median for a decimation factor of 3 or 4. Decimation factors of 1 or 2 are not allowed. For decimation factors greater than 4, the middle values are still averaged after removing the high and low values. For large N, the median calculation approaches the same result as the VA average block.

The VA block allows averaging with decimation factors between 1 and 15. The GRC scheduler does not allow capturing more than 15 vectors before calling the block. The full median filter test setup is described in Appendix A.

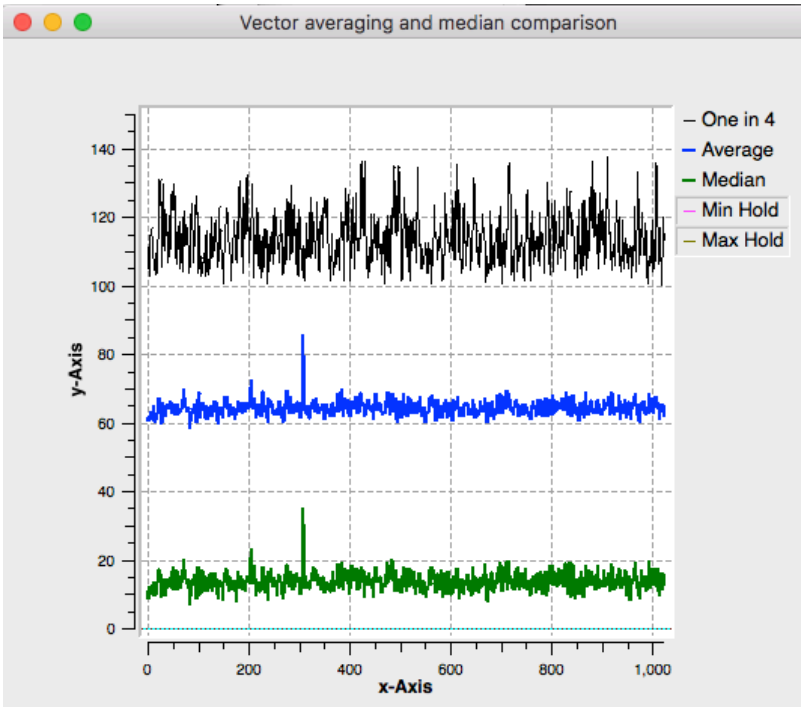

Figure A.2: Comparison of Vector Average and Vector Median blocks relative to an input model spectrum. *The bottom (green) curve shows the Vector Median output after the test spectra pass through two median blocks, each with decimation factor of 4. The middle plot (blue) shows the average of spectra passing through two vector average blocks each with a decimation factor of 4. The top (black) plot shows a single input spectrum before passing through the vector processing blocks. counts. The curves are offset to allow comparison.*

Figure A.2 shows a comparison of inputs and outputs of the VA and VM blocks. The points to note in **Figure A.2** are **1)** both the median and average blocks significantly reduce the noise in

the spectra, compared with a single input spectrum. Note **2)** that the median shows about the same noise level as the average block. The median block is much more resistant to short-term interference, with only a small noise penalty.

Appendix B: NsfIntegrate.grc Installation Guide

This section is underdevelopment, as I prepare to place all the components in Github.