The high frequency calibrator conundrum: Pipeline and identification algorithm for potential calibrator sources

Cameron Trapp[1]

Advisors: Ylva Pihlström[1] and Lorant Sjouwerman[2]

[1] University of New Mexico, Department of Physics and Astronomy, [2] National Radio Astronomy Observatory (NRAO)

ABSTRACT

We present the development and implementation of the Calibrator Identification Algorithm, which can be used to automatically calibrate, analyze, and rank potential calibrator sources at a variety of radio frequencies. This pipeline examines candidate sources for a variety of properties related to intrinsic brightness and compactness. It was developed specifically for the Bulge Asymmetries and Dynamical Evolution (BAaDE) project in order to identify high frequency radio calibrators for future studies of the most central regions of our Galaxy, although it can be applied to other studies at arbitrary frequency bands as well. Additionally, we present the results of an initial high frequency (43 GHz) Very Large Array calibrator survey analyzed with this pipeline. The ranking algorithm categorized 30 new sources as good calibrators. Software and calibrator catalogs will be made available at the National Radio Astronomy Observatory website as well as the appendices of this thesis.

Subject headings: Galaxy: bulge—Galaxy: kinematics and dynamics—stars: calibrators—techniques: pipeline and algorithm

Contents

1. Introduction

1.1. Scientific background: the BAaDE Project

The BAaDE (Bulge Asymmetries and Dynamical Evolution) project aims to significantly improve models of the structure and dynamics of the Galactic bulge and inner Galaxy by mapping the positions and velocities of up to 34,000 silicon oxide (SiO) maser stars not observable in optical surveys (-6°<b<6°). Based on observations of infrared morphology (Blitz & Spergel 1991), maser stars (Habing et al. 2006), and dynamics of red giants (Rich et al. 2007; Kunder et al. 2012), the central region of the galaxy is now known to be dominated by a massive bar structure, (Fig. 1). Models fit to the red giant datasets imply that this bar formation may be the result of dynamical buckling of a preexisting massive disk, leading to a bulge (Kormendy & Kennicutt 2004). This model is supported by observations of cylindrical rotation of the bulge, which is associated with triaxial or boxy bulges (Howard et al. 2008). Additionally, recent evidence, such as observations of 2 red clump populations of stars inconsistent with current bar models (McWilliam et al. 2010), points towards the possibility of an additional X-shaped structure. Such structures are often seen in boxy bulges within other galaxies (Whitmore & Bell 1998; Bureau et al. 2006; Howard et al. 2008, 2009) and various simulations (Patsis et al. 2002; Athanassoula 2005). Even though much is known about the overall structure of the Galactic bulge, details about these dynamical features are still lacking. Therefore, the BAaDE project has begun conducting observations in order to detect approximately 20,000 red giant SiO maser sources with the Very Large Array (VLA) and Atacama Large Millimeter Array (ALMA). This large number of point sources should be sufficient to trace the complex structures and dynamics near the Galactic Center. Following these observations, sufficiently bright sources will be observed with the Very Long Baseline Array (VLBA) in order to perform follow up orbit and parallax (a method to measure distance) determination.

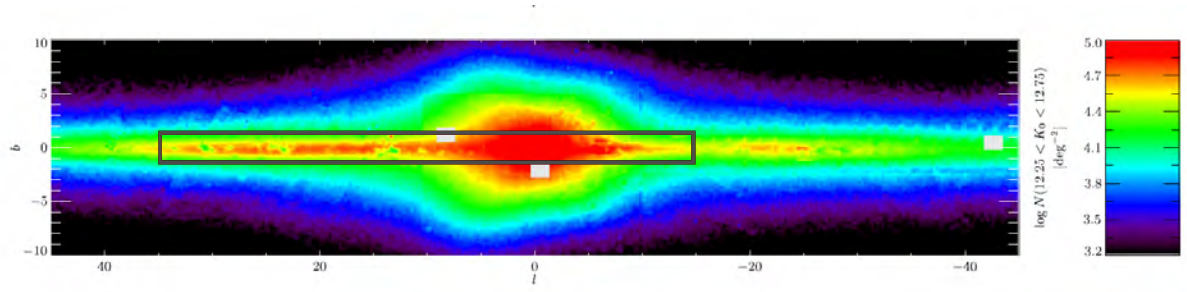


Fig. 1.— This figure gives a general idea of the structure of the Milky Way. Colors indicate the surface density of stars, smoothed with a Gaussian kernel of 0.1°. Image was taken from Wegg et al. 2015. A box was added from -15°<l <+35°and -1.5°<b <1.5°to show the general area surveyed in this study.

1.2. Radio interferometry

When studying the central parts of the Milky Way, observations at radio frequencies can probe regions unobservable at optical frequencies. Obscuration is caused by extinction due to dust around and within the bar of the galaxy, which scatters light in a frequency dependent fashion. The higher the frequency, the greater the extinction (roughly $\sigma_s \propto \nu$, where σ_s is the scattering cross section for a photon and ν is the frequency of the photon). Extinction due to dust is thus negligible at radio frequencies, making the VLA and VLBA ideal instruments to study structures in the galactic plane. Due to higher spatial resolution from large antenna separations, interferometry provides the best tool for measuring positions and motions of sources at the distances associated with the central regions of the Milky Way (kiloparsecs) (Petrov et al. 2012). Radio interferometry combines signals from multiple radio detectors (e.g., dishes, dipoles) in order to synthesize an aperture with diameter equal to the size of the longest telescope separation (baseline). When forming an interferometric image, spatial resolution is inversely proportional to the baseline length. Interferometry determines phase differences along a wavefront across various baselines in order to sample the spatial frequency domain of a source, known as the UV plane. Short baselines sample low spatial frequencies, while long baselines sample high frequencies. Although signals from interferometers are more sparsely sampled than a single dish of equitable size (limiting sensitivity), there are practical limitations to building single radio dishes with effective apertures equal to the A-configuration VLA or VLBA (36.4 and 10,328 km respectively), including cost, the ability to point the telescope, and the structural integrity of such large structures. In order to take advantage of the properties of radio interferometry, the BAaDE project operates within the radio frequency regime, utilizing masers as point mass tracers.

1.3. Masers

Astrophysical masers provide a convenient way to map the positions and velocities of stars at various low frequencies. Masers (Microwave Amplification by Stimulated Emission of Radiation) function on the same principles as lasers. They require a metastable energy state, in which an electron or molecule can exist for an extended period of time without spontaneously decaying to the lower energy state, and a pumping mechanism in order to raise the populations to this metastable state, creating a population inversion (Fig. 2). As an excited molecule decays to a lower energy state, a photon is emitted. This photon can stimulate the decay of other molecules in metastable states, which emit additional photons. If there is a large number of molecules pumped into the metastable state, a cascade effect will take place, producing an exponentially growing number of photons at the same frequency. Certain conditions must be met for an astrophysical maser to exist. In order to achieve population inversion, there must be a strong source of pumping energy from radiation or collisions. There must also be velocity coherence along the line of sight of the maser such that photons are not Doppler shifted relative to other molecules and are, therefore, at the correct frequency to stimulate further emission. Appropriate molecules must also be present

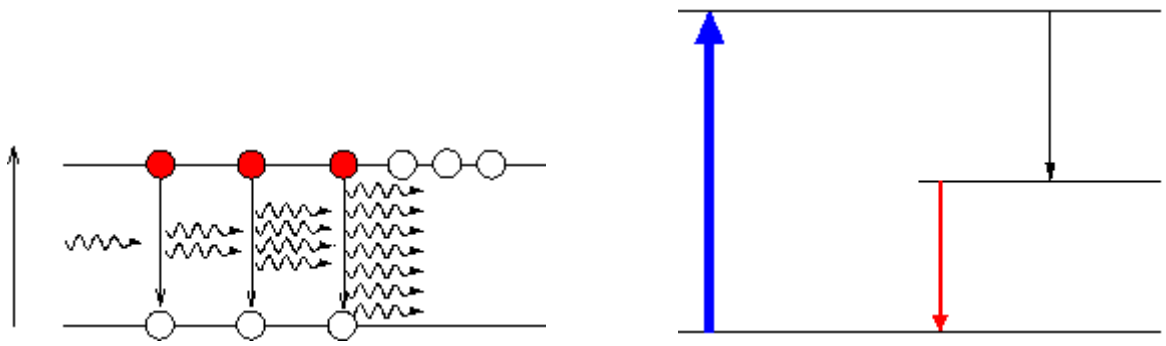


Fig. 2.— The diagram on the left shows the effect of stimulated emission. A photon disturbs an electron/molecule (red circle), which decays to a lower energy state and releases an additional photon. The diagram on the right shows the energy levels of an example maser system. The blue arrow is the pumping energy, which elevates an electron/molecule into an elevated energy level. The red arrow represents the masing frequency, in which the the electron/molecule decays from the metastable state.

in the area of interest, such as hydroxide, water, ammonia, and silicon oxide (Reid & Moran 1981).

The conditions present in the surroundings of red giant stars, specifically asymptotic giant branch (AGB) stars, provide suitable conditions for maser formation. When stars below 10 solar masses begin to die, they go through a series of predictable phases, the final of which is the AGB phase. In this phase, there is a low velocity outflow of matter, which forms a cold, circumstellar envelope (Habing et al. 2006). Combined with the pumping from the star, various molecules within this envelope are able to produce masers. Although AGB stars are short lived, a vast majority of stars with $M < 10M_{\odot}$ go through this phase, resulting in a large AGB population within the Galactic plane (van Loon et al. 2003). In addition, the high intensity and low frequency of the associated masers allow for consistent tracking even in dusty, heavily obscured regions of our Galaxy.

The BAaDE study uses SiO masers as tracers of these red giant stars. Compared with other maser species present, such as hydroxide, SiO masers tend to be found lower in the stellar envelope, below the dust condensation layer and closer to the star (Habing 1996). Additionally, SiO mases at 43 and 86 GHz, which are relatively high frequencies for observations with radio telescopes. These higher radio frequencies provide enhanced spatial resolution (resolution $\propto \nu^{-1}$), making SiO masers very precise tracers of stellar position. It is also relatively easy to predict the conditions that will lead to strong emission of SiO masers. For example, there is a strong correlation between H-alpha emission as well as infrared continuum emission with maser intensity (Heske 1989; Bujarrabal et al. 1987). More recently, Sjouwerman et al. (2009) developed a method to predict the presence of maser species through cross-identifications between the *IRAS* and *MSX* source catalogs. By identifying sources within color regions corresponding to high levels of oxygen and carbon (*MSX*), and a large oxygen to carbon ratio (*IRAS*), the existence of SiO masers can be predicted to a high

level of accuracy (approximately 89%). These selection criteria are sensible, as without oxygen, no SiO molecules would be able to form. Additionally, oxygen tends to bond with carbon over silicon, so there must be a significantly higher amount of oxygen than carbon for an appropriate concentration of SiO to be present (Sjouwerman et al. 2009). These relationships simplify the process of source selection, making it possible to choose a large number of candidate sources likely to harbor masers.

1.4. The need for high frequency calibrators

In order to generate high-quality images using interferometry, reliable calibrators are needed. Good calibrators are bright enough to be detected in short amounts of time on individual baselines, have consistent or slowly varying flux, and are compact in image space, (Fig. 3). Due to varying spectral properties and frequency dependent resolution, a good calibrator at a certain frequency band is not necessarily viable in another. While at low frequencies (<5 GHz) the sky is relatively well mapped and catalogs of calibrator sources for different arrays are readily available (e.g., via the NRAO web pages), at increasingly higher frequencies the telescope primary beam becomes smaller, making sky surveys time expensive. As a consequence, catalogs of high frequency calibrators are far from complete. For high precision astrometry with VLBI, the pointing position of the array is frequently moved between the target and calibrator sources in order to estimate phase changes, which can cause position errors when reconstructing the image. This technique can drastically reduce phase variations caused by atmospheric fluctuations, which can vary on timescales much less than a minute for baselines up to 8000 km (Petrov et al. 2012). Therefore, calibrator sources are needed within 1 - 2° of the targets. While there has been a handful of calibrator surveys at high frequencies (e.g the VERA KCAL, KVN QCAL), they do not sample declinations associated with the Galactic bulge region densely enough for the proposed survey.

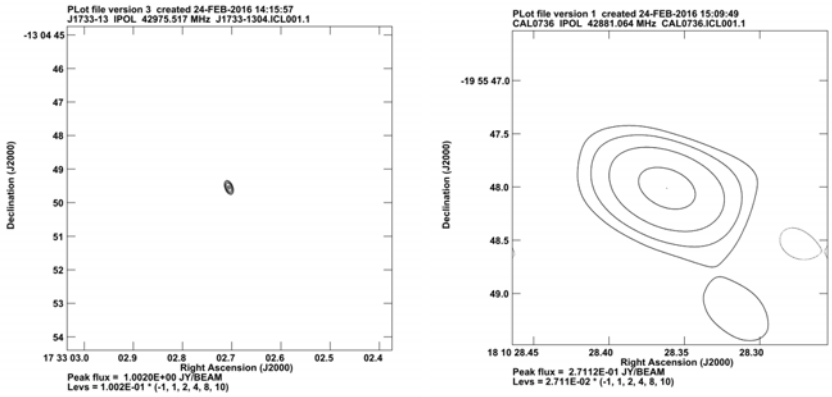


Fig. 3.— Left: A known good calibrator, J1733−1304 (Rank 2041.0). Right: A suitably bright, but diffuse source; not ideal for calibration (Rank 328.9).

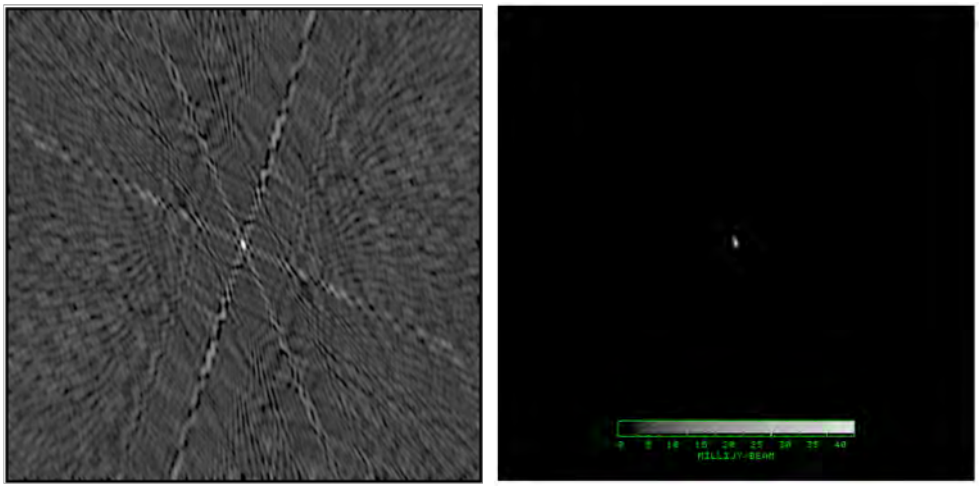


Fig. 4.— Dirty (left) and cleaned (right) images for known calibrator J1755−2232.

The ideal calibrator is a bright point source. In order to generate a point source, an object's observable extent must be less than the resolution of the instrument, i.e. the object is not resolved. An image is formed by convolving the raw signal from the observed source with the point spread function (PSF) inherent to the instrument. Therefore, the generated image for a point source will be composed of an isolated PSF. For a single telescope/antenna, the PSF is simply the Fourier transform of the aperture. For interferometry, the PSF is the Fourier transform of the interferometers sampling function, $S(u,v)$, which describes how the various baselines cover the spatial frequency domain. The raw signal can also be described in this frequency domain as the Fourier transform of the sky brightness distribution:

$$V(u,v) = \mathcal{F}\Big[I(l,m)\Big] = \int I(l,m) exp[-i2\pi(ul+vm)] dl\, dm, \tag{1}$$

where $I(l,m)$ is the sky brightness in image space and $V(u,v)$ is the visibility in frequency space. Therefore, the uncalibrated, or dirty, image results from the convolution of this PSF ($\mathcal{F}[S(u,v)]$) and the brightness distribution ($\mathcal{F}[V(u,v)]$) as such:

$$I_{dirty} = \mathcal{F}\Big[S(u,v)V(u,v)\Big]. \tag{2}$$

Interferometric techniques reconstruct an image from various, limited visibilities, making the explicit calculation of the PSF difficult (Woody 2001). By observing a point source, this PSF (referred to as the beam) can be directly measured:

$$\mathcal{F}\Big[S(u,v)V(u,v)_{point}\Big] = \mathcal{F}\Big[S(u,v)\Big] * \delta(l,m) = \mathcal{F}\Big[S(u,v)\Big], \tag{3}$$

where $\delta(l,m)$ is a point source in image space. Images of unknown structure can then be deconvoluted with this function through various image processing techniques, such as the CLEAN algorithm

(Högbom, J. 1974). This deconvolution removes artificial components of the image associated with the PSF, such as side lobes, corrects phase variations across various baselines, and allows for accurate analysis to be performed, (Fig. 4). Since VLBI has higher angular resolution then other interferometry techniques, more compact sources are necessary to generate a PSF. Therefore, having a sample of confirmed compact calibrators available is particularly important for future VLBI studies.

In this paper we report on our development and implementation of an automated pipeline and algorithm for the identification and ranking of potential calibrator sources at arbitrary frequencies. In addition, we present the results of a VLA A-configuration survey of over 1200 potential calibrator sources for future use in the BAaDE VLBA studies and its analysis with the aforementioned pipeline. The resulting catalog is presented in this paper and will be made available for use on the BAaDE and NRAO websites.

2. Observations

2.1. Very Large Array

In order to identify useful calibrator sources for maser detection at 43 GHz, an initial VLA calibrator survey concentrated to the inner Galaxy and innermost Galactic plane was conducted. This survey was performed in Q-band, a frequency band from 40.0-50.0 GHz. When observing with the VLA, incoming signals are down-converted to lower frequencies, resulting in an observing structure containing sub-bands, or intermediate frequencies (IFs). Our observations covered 2 GHz centered around 43 GHz and spanned 16 IFs in total, each with a spectral width of 128 MHz. The configuration of the VLA antennae are changed throughout the year in order to account for the spatial frequency needs of a variety of studies. During our observations, the VLA was in BnA and A configuration, giving a maximum baseline length of 36.4 km and a synthesized beam width of 0.043 arcsec in Q-band, (Fig. 5). In order to narrow our focus to calibrator candidates likely to be bright at these frequencies, sources from the NRAO/VLA Sky Survey (NVSS) with flux densities greater than 50 mJy at 1.4 GHz were selected for our observations at Q-band (Condon et al. 1998). These selection criteria resulted in more than 1200 sources over 150 square degrees within the innermost region of the plane and the inner Galaxy, i.e. (around -15°$<l<$+35°) and (-1.5°$<b<$1.5°).

2.2. Very Long Baseline Interferometry

To determine the viability of sources on VLBI-scale baselines and train our identification algorithm, observations of strong candidate VLBI calibrators will be conducted with the VLBA, as well as the European VLBI Network (EVN). Five hours have been observed with the VLBA and we have been approved for 12 hours on the EVN. Sources to be observed were selected to be

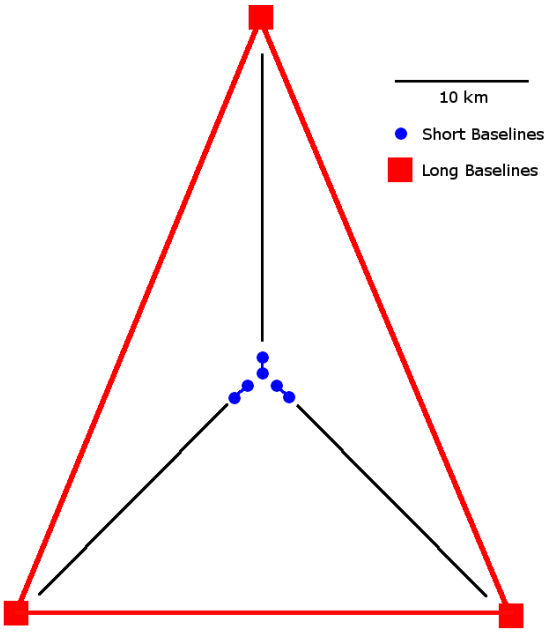


Fig. 5.— The VLA in A configuration. It consists of 27 antennas (each 25 m in diameter) arranged in a 'Y' shape. Red and blue lines show baselines used for analysis of source compactness, while the labeled circles and squares represent the constituent antennae. Long and short baselines are approximately 36.4 km and 0.68 km in this configuration, respectively.

high ranking candidates reachable with the respective arrays. Precise astrometry is not necessary at this stage, so simple detection experiments will be performed. Candidate calibrators will be observed for 5 minutes at Q-band in order to determine whether they are bright and compact enough to function as calibrator sources on a single baseline, polarization, and sub-band. These observations will be used to train the algorithm to reliably identify compact sources for VLBI. As not all data have been collected, a comprehensive analysis of this feedback will not be presented in this manuscript.

3. Data Reduction and Analysis

3.1. Calibration of VLA Data

Data from the VLA must be corrected for atmospheric errors and converted to a proper amplitude scale before analysis can be performed. Images were calibrated using 'doosro', an existing AIPS pipeline used for VLA projects. The primary flux calibrator source was J1733−1304, which is reported to have a flux of 9 Jy in Q-band by the VLA calibrator manual. Additional known calibra-

tors J1851+0035, J1820−2528, J1832−1035, J1744−3116, J1832−2039, J1733−3722, J1824+0119, J1755−2232, and J1717−3342 were used to verify calibration as well as the viability of the ranking algorithm. Antenna E08 was used as the reference antenna. Sources were saved as calibrated single source files and multi-source files for further analysis. Information from the task 'getjy', which reported on the derived flux for each source, was saved to a text file following calibration in order to quickly and efficiently identify detected sources.

3.2. Processing

3.2.1. AIPS Procedures

In order to evaluate whether a candidate source can be used as a calibrator, certain parameters were measured and calculated from the final images. When analyzing data in AIPS, several different tasks are available for extracting various properties of a source, such as flux in a certain region of an image, amplitude RMS differences across baselines, compactness of an image, and amplitude on a baseline. Custom AIPS procedures were created in order to automatically extract these parameters for the 1200 sources in our sample, (Appendix A.1). In this section, we briefly describe what these parameters are and how they were determined.

First, the signal to noise ratio (SNR) was calculated using the task 'imean'. An 80 by 80 pixel region of interest was placed at the center of the image and in the bottom left corner in order to estimate the source flux and the noise level, respectively. While this measure gives a good general metric for differentiating between strongly and weakly detected sources, it gives no explicit insight into differences across baselines, which would correspond to the compactness of the source.

Therefore, the task 'evauv' was used to assess the compactness of the candidate sources. A point source model was created using the integrated flux from the clean component image generated during calibration. This model was then used to assess the candidate source through two methods: subtraction and division. In the subtraction model, the point source model is subtracted from the calibrated UV data. In the division model, the calibrated UV data is divided by the point source model and unity is subtracted across the image. Since the model has the same flux, the closer the candidate source is to a point source, the closer the residual image will be to zero.

A second method to investigate compactness is to compare the amplitudes on long and short baselines. An ideal point source has flux uniformly distributed across the spatial frequency domain, (Fig. 6). Therefore, for a compact source, approximately the same flux is expected on both the long and short baselines. A resolved source, on the other hand, would have a relatively low amount of flux on the long baselines, (Fig. 6). The flux amplitudes across various baselines were extracted with the task 'uvprt' using the multi-source files as inputs. Since observations were made in the transitional BnA configuration, the maximum baseline lengths were not the same for each source. Therefore, in order to maintain consistency, the short baselines were set to approximately 0.68 km,

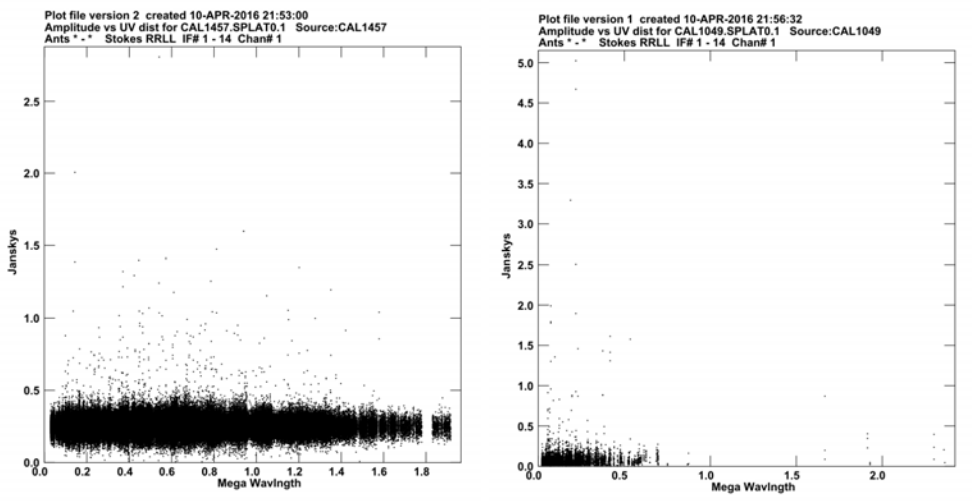


Fig. 6.— UV amplitude for a good, compact calibrator (left, rank=1518.3) and a bad, diffuse calibrator (right, rank=952.5). Long baselines correspond to large UV values, while short baselines correspond to small UV values. The good calibrator has relatively uniform amplitude distribution across all baselines, while the bad calibrator drops off sharply with UV distance.

while the long baselines were limited to approximately 11.1 km.

Finally, the task 'evasn' was used to evaluate the solutions tables generated during calibration. 'Evasn' reports the amplitude corrections for each baseline. By analyzing the RMS value of these corrections for each IF, we get a measure of how easy it was to detect and calibrate the source. Low RMS differences correspond with a strong detection on all baselines. Since an ideal point source will have identical flux on all baselines, a good calibrator source will have low RMS differences in this analysis.

All task outputs were printed to text files for further processing. Contour plots were generated for each source using the task 'cntr' with levels at 10, 20, 40, 80, and 100 percent of peak flux in order to show structure. A contour at -10 percent was included to show noise level in weakly detected sources. Images were saved as postscript files using the task 'lwpla' (Fig. 7; Appendix C). Procedures were saved as AIPS run files (Appendix A.1) and, along with the existing calibration pipeline, were called by a central python script (Appendix A.3).

3.2.2. Algorithm and Perl Scripts

Outputs from the AIPS procedures described in 3.2.1 were subsequently reduced using a Perl script (Appendix A.2). A source list was generated based on the results of 'getjy'. Sources with amplitude RMS values from 'getjy' above a certain threshold (0.01 Jy) were ignored, as it indicated

no source was detected. All others were printed to a file and flagged for further analysis. This thresholding improves overall efficiency of this pipeline by reducing the number of sources processed.

By using the outputs from 'imean','evasn','evauv', and 'uvprt', the final parameters to be used in source evaluation were calculated and recorded for each source. The peak and off source flux were extracted from the 'imean' outputs, giving the signal to noise ratio. The RMS of the gain between baselines (calculated both for right and left polarizations) were pulled from the output of 'evasn' for each IF. The averages across IFs for each source were calculated and printed. The residuals from the subtraction and division models for each source were pulled from the output of 'evauv'. Amplitudes across the six baselines selected for analysis were pulled from the output of 'uvprt'. Averages were calculated separately for the three short and three long baselines, yielding a ratio between the average long and short baseline amplitudes (BLR_{obs}).

The resulting set of characteristic parameters for each source were used to estimate its 'goodness' as a calibrator. Ranks for each source were calculated based on the following model:

$$Rank = \left[\frac{A}{Residuals} + B_{\mathrm{SNR}} + \frac{C}{\mathrm{RMS}_{Ant}} + D_{Baselines}\right]\Theta_{\mathrm{RMS}} \tag{4}$$

$$B_{\mathrm{SNR}} = \begin{cases} B_1 \times \mathrm{SNR}_{obs} & \mathrm{SNR}_{obs} < 100 \\ B_2 \times \mathrm{SNR}_{obs} + 100B_1 & \mathrm{SNR}_{obs} \geq 100 \end{cases} \tag{5}$$

$$D_{Baselines} = \begin{cases} D_1 \times \mathrm{BLR}_{\mathrm{obs}} & \mathrm{SNR}_{obs} < 10 \\ D_2 \times \left[\frac{\mathrm{SNR}_{obs} \times \mathrm{BLR}_{\mathrm{obs}} - 1}{\mathrm{SNR}_{obs} - 1}\right] & \mathrm{SNR}_{obs} \geq 10 \end{cases} \tag{6}$$

$$\Theta_{\mathrm{RMS}} = \begin{cases} 0 & \mathrm{RMS}_{\mathrm{IF}} < 0.01 \\ 1 & \mathrm{RMS}_{\mathrm{IF}} \geq 0.01 \end{cases} \tag{7}$$

where a high *Rank* corresponds to a good calibrator. The values of A, B_1, B_2, C, D_1, and D_2 were initially varied in order to match calibrator ranking with visual inspection of contour plots (50, 2, 0 .1, 0.01, 0.01 and 1000 respectively). In the first term in equation 4, the parameter *Residuals* represents the residual amplitude from the division model, extracted from the task 'evauv'. The second term contains two separable expressions depending on the observed SNR value (Eq. 5), allowing for its effect to be adjusted at a certain threshold. We cannot separate signal from noise when measuring the source flux, so $\mathrm{SNR}_{obs} = \frac{Signal+Noise}{Noise}$. The third term depends on RMS_{Ant}, which is the gain RMS difference across baselines from 'evasn'. The fourth term (Eq. 6) depends on the observed SNR and amplitude ratio between long and short baselines ($\mathrm{BLR}_{\mathrm{obs}}$). In equation 7, $\mathrm{RMS}_{\mathrm{IF}}$ is the amplitude RMS difference across IFs from 'getjy', which is being used to threshold out non detections in equation 4. Sources were sorted based on rank and printed to a final reduced list containing source name, coordinates, and the other relevant values discussed.

The more complex relative baseline amplitude term (Eq. 6) accounts for the fact that pure noise images may display high observed baseline ratios. For images near pure noise, this ratio can

be measured to be greater than one. This is due to the fact that random noise profiles across baselines tend to be similar. Sources with lower SNRs will have a larger relative noise component, and therefore, the measured peak amplitudes will be naturally more similar between baselines, regardless of the underlying structure of the source. A linear application of the observed baseline ratio in the algorithm results in the consistent high ranking of non compact, poorly detected sources and non-detections. Equation 6 accounts for this bias at lower SNRs by calculating the isolated signal baseline ratio. This effectively damps low SNR observed baseline ratios, while leaving the ratio unaffected as SNR goes to infinity. This equation breaks down for extremely low SNRs due to the singularity at SNR = 1 and the fact that baseline ratios may be observed to be greater than 1 at high noise. Therefore, sources with SNR < 10 have their baseline term damped by a constant factor. The derivation is as follows, where BLR_{obs} is the observed baseline amplitude ratio, BLR is the actual baseline amplitude ratio of the source, S is the signal, N is the noise, and $\mathrm{SNR}_{obs} = \frac{S+N}{N}$ is the measured signal to noise:

$$\mathrm{BLR}_{obs} = \frac{\mathrm{BLR} \times S + N}{S + N} \tag{8}$$

$$\mathrm{BLR} = \frac{\mathrm{SNR}_{obs} \times \mathrm{BLR}_{obs} - 1}{\mathrm{SNR}_{obs} - 1} \tag{9}$$

3.2.3. Python Scripts

As the BAaDE project will perform many additional hours of these calibrator searches, we aimed to automate as much of the data reduction process as possible. Therefore, a central script was developed in python in order to run all parts of the pipeline for multiple data sets, (Appendix A.3). The script initially loads user defined settings from a text file, including information about the primary calibrator, the reference antenna of choice, the short and long baselines, the preferred AIPS user number, the naming conventions for the data to analyze, and the data location. A bash shell script is written to a text file including commands to launch AIPS, load the multi-source RUN files, calibrate the images with doosro, and run all saved procedures explained in section 3.2.1. The calibration and plotting functionalities of the pipeline can be toggled on and off within the parameter file in case data has already been calibrated or imaged. AIPS was only able to handle approximately 60 sources per user number, so the shell script simply restarts AIPS and increments the user number after three RUN files are processed. This text file is then run as a subprocess in python. Following this, the Perl script described in section 3.2.2 is called as a second subprocess in order to reduce the outputs generated and rank the candidate sources.

In order to run this pipeline from the python script, the user must first define two environment variables on their system. The variable AIPSOUT determines the output path for all data, including the unreduced outputs from AIPS, the reduced outputs from the Perl script, and the generated contour plots. The variable PROCPATH must be defined as the directory containing the AIPS

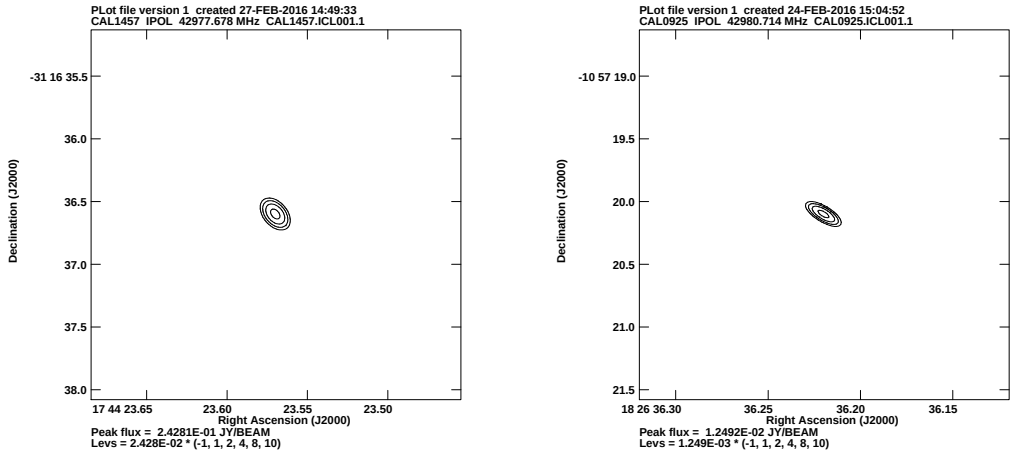


Fig. 7.— Contour plots of the two top ranking calibrator candidates. Rank 1518.3 (left) and 1450.1 (right). See appendix III for all contour plots.

procedure files described in section 3.2.1. Following this, the user must modify the parameters file (cal_identify_parameters.txt) included with the scripts in order to define the relevant parameters for loading the data, calibration, and analysis (Appendix A.4). Finally, the python script can be called from the command line with four input arguments corresponding to the AIPS user number to use, minimum and maximum run, and the final reduce filename (e.g. python cal_identify.py 101 1 3 calList.txt). Additionally, stand alone versions of the AIPS procedures and Perl scripts were written if running the entire pipeline at once is not desired.

4. Results

4.1. Calibrators

Out of the 1200 sources surveyed, 97 were flagged as detected (Appendix B). An initial threshold rank of 1100 was determined with the above parameters by manual inspection of the VLA data. With this thresholding, 30 sources were identified as good calibrators (Table 1, Appendix B). Therefore, 2.5% of all surveyed sources were identified as potentially useful calibrators at 43 GHz, and 30.9% of detected sources were useful calibrators. Contour plots for the two highest ranked

Source	RA	Dec	l	b	Flux (Jy)	Rank
CAL1457	17 h 44 m 23.57 s	-31° 16' 36.60"	-1.7°	-0.71°	2.2	1518.3
CAL0925	18 h 26 m 36.22 s	-10° 57' 20.10"	22°	1.3°	0.11	1450.1
CAL0824	18 h 18 m 2.96 s	-17° 05' 42.30"	14°	-0.51°	0.73	1439.7
CAL0588	17 h 55 m 26.31 s	-22° 32' 11.00"	7.6°	1.9°	0.36	1387.8
CAL0608	17 h 58 m 22.97 s	-23° 43' 11.40"	7.2°	0.92°	1.0	1354.8

Table 1: Top 5 ranked calibrator candidates. See Appendix B for complete ranked list of sources.

sources are presented in Figure 7. All other flagged source plots may be found in Appendix C. The distribution of good calibrators tended towards high galactic longitudes and latitudes of around 1°(Figure 8,9), with the largest number of good calibrators being found around $l = 27°$, (Figure 8). Mean separation between good calibrators varied between 1-2.5°, with an average of 1.7°.

4.2. Computational Efficiency

In order to accommodate a large amount of sources, special attention was given to efficiency of this pipeline. The computational cost is linearly dependent on the number of sources. If toggled on, the calibration step dominates the cost. This calibration step and the successive AIPS procedures used to analyze the data have a linear dependence on the number of sources (Equation 10,11). The data reduction component of the identification algorithm is linearly dependent only on the number of detected sources, making it much less computationally costly than the other components (Equation 12).

$$\text{Cost}_{cal} \propto n_{sources} \tag{10}$$

$$\text{Cost}_{aips} \propto n_{sources} \tag{11}$$

$$\text{Cost}_{perl} \propto n_{detected} \tag{12}$$

Therefore, the total computational cost of the ranking goes as:

$$\text{Cost}_{total} = (C_{cal} + C_{aips})n_{sources} + C_{perl}n_{detected} \tag{13}$$

The values for C_{cal},C_{aips}, and C_{perl} will vary depending on the system used, but are approximately 2 minutes, 1.5 s, and 15 ms respectively. If a large data set of calibrated sources already exists, the algorithm will be able to sort through them relatively quickly, increasing its potential applications.

5. Discussion

5.1. Spatial distribution of detections

In order to comprehensively map the central regions of the Milky Way, the BAaDE project requires a dense catalog of calibrator sources distributed throughout and around the primary window of interest (-15°<l <+35°and -1.5°<b <1.5°). Ideally, there would be a known calibrator within 1-2° of every SiO maser source being surveyed. The density of maser sources is relatively high, with up to 100 masers within 1° of each other in the densest regions being surveyed. Therefore, a single calibrator in the correct location may be able to be used to observe a large number of masers. This survey provides a suitable density of calibrators within certain regions, however, it is still lacking in others. In general, we found a large number of good calibrators in the region +13°<l <+29° and 0°<b <1°, while regions such +5°<l <+10° and -1°<b <0° lack new calibrator sources, (Fig. 8). This bias is due to the underlying distribution of the data sets analyzed in this study, (Fig. 9).With the relatively small number of sources observed, it is difficult to be certain about distribution statistics, but there appears to be no strong detection trend as a function of position. This is to be expected, as we are observing background sources uniformly distributed across the sky and should not be limited by extinction from the Galactic plane.

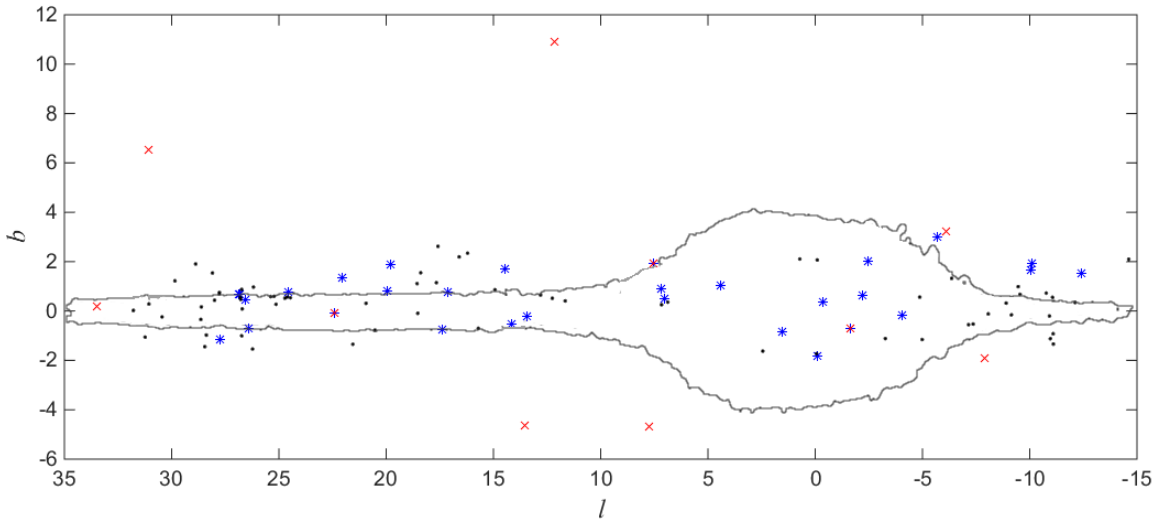


Fig. 8.— Plot of galactic coordinates (l,b) for the detected sources. Good calibrators above the selected threshold are plotted as blue *'s, known calibrators are as red x's, and bad detections as black dots. The 4.7 contour from Figure 1 is plotted to provide reference to galactic structure.

5.2. Discussion on finer details of the algorithm

The current ranking equation is dominated by the relative baseline amplitude term and terms dependent on signal to noise. The function controlling the linear weight of the SNR term (Eq. 5) contains two expressions to account for the fact that, as far as calibration is concerned, the brightness of a source does not matter significantly above a certain threshold. As long as an otherwise good calibrator source is detectable within a reasonable amount of time such that the effects of atmospheric fluctuations (timescales $<<$1 min) are minimized, it can be utilized for VLBI studies. Therefore, after a threshold SNR of 100, as measured by the peak over RMS noise in a clean component image, the additional linear effects of SNR are diminished by a factor of 20 by the selection of B_2.

The algorithm determining the ranking of candidate calibrators will continue to be refined as additional VLBI data is received since we are interested in finding a strong correlation between a good VLBI detection and a high rank from our VLA data. Terms dependent on the point source model residuals and the RMS differences between antennae did not return as consistent of results as the relative baseline amplitude term, and therefore serve as auxiliary terms in order to further differentiate sources based on compactness and allow for flexibility of the model for future iterations. When analyzing the residual terms, the subtraction model returned values relatively constant for all sources and did not vary consistently for known calibrators. The division model, however, returned values with a greater range and showed low residuals for all known calibrators. Therefore, the subtraction model was discarded and the division model was used in equation 4. While giving additional insight into compactness, the task 'evasn', which was used to determine the RMS differences between antennae, did not output data for all sources. It was therefore weighted softly in order to minimize effects, but allow for minor differences between similarly ranked sources. If these ranking terms do not seem to have any consistent impact on VLBI candidates, they may be thrown out in future iterations. The contour plots generated for each source were organized based on these ranking terms in order of descending rank, (Appendix C).

5.3. Comparison of ranking and contour plots

Contour plots show consistently compact sources at high ranks. Faintly detected sources still appear high in the list as long as they are compact (e.g. CAL1157, CAL0616). Since relatively dim sources can still be used as calibrators, this shows the robustness of the algorithm for identifying a variety of potentially useful calibrators. In contrast, sources may also appear lower on the ranking list despite appearing to resemble good calibrators based solely on visual inspection (e.g. CAL0944). While bright, these sources have relatively low baseline ratios (approximately 0.5). This implies that they are resolved, non compact sources, and would not work very well as calibrators. In this case, the algorithm is going beyond what traditional visual inspection is able to accomplish.

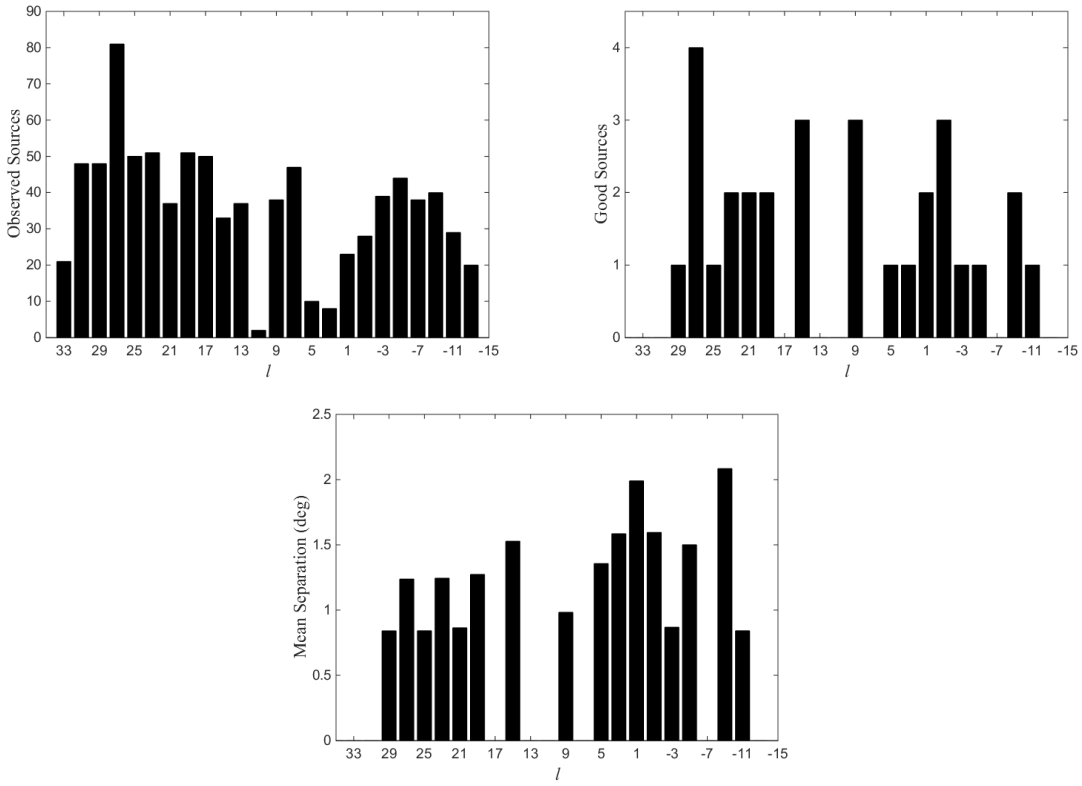


Fig. 9.— Histograms using 2° Galactic longitude bins for the total observed sources (left) and the good calibrators identified (right).The mean source separation of the good calibrators is shown on the bottom. Overall distribution of observed and good sources is approximately the same. This allows us to visualize how densely we have sampled certain longitudes

5.4. Algorithm Robustness

Sources below the rank of 100 (Appendix B) are very poor or non detections that made it past the initial IF RMS thresholding. While the computational cost of the pipeline would be marginally improved by further fine tuning of this threshold (Eq. 7), the inclusion of non-detections is more desirable than the exclusion of detections. In all cases, these weak/non-detections are ranked significantly lower than all detected sources, making them easy to identify. Their presence does not contaminate the results and provides additional support that the algorithm is behaving as expected.

The robustness of our algorithm is further verified by the ranking of the known calibrators surveyed. Across all runs, data was collected multiple times for these sources. Each separate image for a given known calibrator returned the same rank when analyzed. This consistency between runs displays the reliability of the algorithm, as the assigned rank is not specific to a given image, but rather the actual source being observed. The top five ranking sources are known calibrators,

however, five additional secondary calibrators are interspersed with the candidate sources. The primary calibrator, J1733−1304 (rank=2041.0), is ranked higher than all other sources, known calibrators or otherwise. The lower ranked, secondary calibrators, are of a similar value as the candidate sources identified as good calibrators, with 15 candidate sources ranked higher than the lowest ranking known calibrator (rank=1220.5). No known calibrators appear at a rank below the designated threshold and all appear as compact point sources when plotted as contour maps. This overall ranking distribution implies that the algorithm is behaving as expected, and that the sources categorized as good are comparable to previously known calibrators.

5.5. Computational efficiency

In order to allow for the processing of a large amount of sources at a single time, attention was paid to the computational efficiency of the algorithm. Reduced data from each individual AIPS output were saved into arrays. While it would be more computationally efficient to store all data in the same order and loop through it once at the end, the formatting and reliability of AIPS outputs are not consistent enough to allow for this (i.e. sources being skipped, certain files not correctly being outputted, no source names provided in some outputs). It therefore becomes necessary to implement a single nested loop in order to verify that the data from each AIPS output corresponds with the same source. The cost of this loop is offset by analyzing only sources flagged as detected in the initial analysis of 'getjy'. In the case of our study, this reduced the number of sources from $n = 1200$ to $n = 97$, corresponding to a reduction in computation time from 18s to 1.5s (Eq 10). While this improvement is significant if the Perl script is being run independently, the computational cost of the AIPS processing still dominates the overall computation time.

5.6. Future Work

The catalog generated by this thesis will be used directly by the BAaDE project in upcoming VLBA SiO maser surveys. Following additional feedback from VLBI calibrator surveys, the ranking threshold will be adjusted such that at least 95% of calibrator sources above a certain rank are good calibrators. This final calibrator list will be made available on the National Radio Astronomy Observatory (NRAO) website for future studies at Q-Band. The pipeline itself will continue to be used to identify additional calibrator sources from future searches associated with the BAaDE project. If few or no calibrators surveyed prove to be useful for VLBA observations, our future observations will be severely limited by the number of calibrators we do have. This algorithm will remain useful to identify calibrators for VLA studies and, with additional parameter refinement, may still prove useful to identify potential VLBI candidate calibrators. While initially designed for high frequencies, this pipeline is not confined to any specific spectral band. Studies in other frequency bands lacking a suitable number of calibrator sources, such as extreme low frequencies (e.g. 4 band at 74 MHz) and other high frequency bands (e.g. K and Ka band at 22 and 33 GHz

respectively), will be able to make use of this pipeline in order to streamline the identification process of calibrators. The terms used in the ranking algorithm are not explicitly frequency dependent, and their overall behavior should not change. Before implementation, however, specific values for the constants A, B_1, B_2, C, D_1, and D_2 should be verified at the frequency range of interest and adjusted accordingly.

6. Conclusions

The pipeline presented in this thesis reliably and robustly identifies and ranks observed calibrator candidates by analyzing the compactness and brightness of the images generated. The VLA calibrator survey, which was analyzed utilizing this algorithm, returned 30 new high frequency (Q Band) calibrators within the central most regions of the Galaxy. These sources will be used as calibrators in future BAaDE studies of 43 GHz SiO masers in order to probe the structure and kinematics of the inner Galaxy. Future VLA calibrator surveys and VLBA observations will be used to expand this catalog and refine the specifics of the algorithm. The pipeline and catalog will also be made available for other studies on the NRAO and BAaDE websites.

7. Acknowledgments

Primarily, I would like to thank Dr. Pihlström and Dr. Sjouwerman for making this project possible and providing support and guidance throughout. Additionally, I would like to thank Michael Stroh for providing assistance with more advanced programming techniques, as well as the entire BAaDE team. Special thanks to my grandfather, Dr. John Dettmer, for his comments on the text and Ryan Poliner for assistance with computational efficiency. Finally, I would like to thank the University of New Mexico Physics and Astronomy Undergraduate Committee for reviewing this thesis.

REFERENCES

Athanassoula, E. 2005, MNRAS, 358, 1477

Babusiaux, C. & Gilmore, G. 2005, Proceedings of the 'Resolved stellar Population' conference

Babusiaux, C. & Gilmore, G. 2005, MNAS, 358, 1309-1319

Blitz, L & Spergel, D.N. 1991, ApJ, 379,631-638

Bujarrabal, V., Planesas, P., & del Romero, A. 1987, A&A, 175, 164

Bureau, M., Aronica, G., Athanassoula, E., Dettmar, R.-J., Bosma, A., & Freeman, K. C. 2006, MNRAS, 370, 753

Condon, et al. 1998, AJ, 115, 1693

De Propris, R., Rich, R.M., et al. 2011, ApJL, 732, L36

Dwek, E., Arendt, R.G., et al. 1995, ApJ, 45, 716-730

Habing, H. J., 1996, AA, 7, 97

Habing, H.J., Sevenster, M.N. et al. 2006, AA, 458, 151-162

Heske, A., 1989, A&A, 208, 77

Högbom, J., 1974, Astrophys. J. Suppl. Ser. 15, 417-426

Howard, C.D., Rich, R.M., et al. 2008, ApJ, 688, 1060-1077

Howard, C.D., et al. 2009, ApJ, 702, L153

Kormendy, J. and Kennicut, R.C. 2004, Annual Review of Astronomy and Astrophysics, 42, 603-683

Dunder, A., Koch, A., et al. 2012, AJ, 143, 57

McWilliam, A., Fullbright, J., & Rich, R.M. 2010, Chemical Abundances in the Universe:Connecting First Stars to Planets, 265, 279-284.

McWilliam, A. & Zocalli, M. 2010, ApJ, 724, 1491-1502

Messineo, M., Petr Gotzens, M., et al. 2006, Journal of Physics Conference Series, 54, 238-242

Murphy, T., Sadler E.M., et al. 2010, MNRAS, 402, 2403-2423

Murphy, T., Cohen, M., et al. 2010, MNAS, 405, 1560-1572

Ness, M., Freeman, K., Athanassoula, E. 2012, Galactic Archaeology:Near-Field Cosmology and the Formation of the Milky Way, 458, 195

Ness, M., Freeman, K., et al. 2012, ApJ, 756,22

Patsis, PA., Skokos, Ch., & Athanassoula, E. 2002, MNRAS, 337, 578

Petrov, L et al. 2012, AJ, 144, 150

Petrov, L., Honma, M., & Shibata, S.M. 2012, AJ, 143, 35

Reid, M. & Moran, J., 1981, Annual review of astronomy and astrophysics, 19, 231-276

Rich, R.M., Origlia, L., & Valenti, E. 2007, ApJL, 665, L119-L122

Sjouwerman, L., Capen, S., & Claussen, M., 2009, ApJ, 705, 2, 1554-1565

van Loon, J., et al., 2003, MNRAS, 338, 4, 857-879

Vasquez, S. et al. 2013, AA, 555, A91

Wegg, C. & Gerhard, O. 2013, The Messenger, 2013, 154, 54-56

Whitmore, B.C. & Bell, M. 1988, ApJ, 324, 741

Woody, D., 2001 ALMA MEMO 389

A. Computer Code

A.1. AIPS Procedures

CAL.001 - Load Data

```
procedure allcalco
string*16 oldstrng
inext'su';keyword'num row';getthead;n=keyvalue(1)
for m=1:n;pixxy m 2;tabget
if((substr(keystrng,9,9)='=')!(substr(keystrng,1,2)='3C'))then
if(subst(keystrng,9,9)='=')then;keystrng=subst
(keystrng,10,14);tabput;end
keystrng='q';pixxy m 4;tabput
else;keystrng='d';pixxy m 4;tabput;end;
pixxy m 2;tabget
if((m=1) & (substr(keystrng,1,10)='J1733-1304'))then
keystrng='k';pixxy 1 4;tabput;oldstrng=keystrng;end
if((m=2) & (substr(keystrng,1,10)=oldstrng))then
keystrng='dummy';inext'su';pixxy 1,2,1;tabput;end
end;clrtemp
finish

procedure calload
default bdf2aips
dowait 1;outdisk dsk;order -1;asdm(1)'/media/cameron/ymp2tb06/ylva/15A426/
asdm(2)='15A426_'!!char(i);outname'run-'!!char(i);typ asdm(2) outname
clint=3/60
bdf2aips;indisk outdisk;inname outname;allcalco;task'indxr';gx;clrtemp
finish

for i=startrun:endrun;calload;end
```

CAL2.001 - Calibrate

```
procedure calproc
task'doosro';default
workdisk=dsk;catnum=jj
vlantcor=1;nopaus = 1
phaint=0.2;ampint=0.3
baseband=2;refant=14
domodel=-1;ampcal'j1733-1304';domodel=-1
bndcal'j1733-1304';phacal'*''
doimages=1;allimg=1;imgtype'cont'
imsize=256;niter=1000
doosro;
finish

jj=1;calproc;end
jj=2;calproc;end
jj=3;calproc;end

procedure naming
string*2 minrun,midrun,maxrun;
getn 1;keyword 'IMNAME';gethead;minrun = substr(keystrng,5,6);
getn 2;keyword 'IMNAME';gethead;midrun = substr(keystrng,5,6);
getn 3;keyword 'IMNAME';gethead;maxrun = substr(keystrng,5,6);
finish

naming

inp prtmsg;prtask 'getjy';prtime 1;
outprint 'AIPSOUT:GETJY_'!!minrun!!'-'!!maxrun!!'.OUT'
prtmsg
```

SNRC.001 - Find Signal to Noise

```
procedure imnrun
doprint 1;dowait 1;getn(x+y);
STRING*8 src
keyword'IMNAME';gethead;src=substr(keystrng,1,7);

BLC 10,10;TRC 90,90;outtext 'AIPSOUT:imean'!!src!!'_noise.txt';
go imean;wait

BLC 0;TRC 0;outtext 'AIPSOUT:imean'!!src!!'_source.txt';
go imean;wait
finish

procedure imnproc
task 'imean'
y=0;
if (n>1) then;y=scannum(1);end
if (n>2) then;y=y+scannum(2);end
y=y*3+3+5*n+scannum(n)*2;

for x=1:scannum(n);getn(y+x);imnrun;typ y+x;end
finish

for n=1:runs;imnproc;end
```

MDLC.001 - Calculate Point Source Model residuals

```
procedure evuvrun
keyword 'ccflux';dowait=1;
getn(y+idx);gethead;smodel(1)=keyvalue(1);
getn(x+idx);get2n(y+idx);outname=inname;go;

finish

procedure evuvproc
task 'evauv'
y=0;x=0
if (n>1) then;y=scannum(1);x=scannum(1);end
if (n>2) then;y=y+scannum(2);x=x+scannum(2);end
x=x*3+3+5*n;
y=y*3+3+5*n+scannum(n)*2;
for idx=1:scannum(n);evuvrun;wait;end
finish

for n=1:runs;evuvproc;end
```

SNEVL.001 - Evaluate SN Tables

```
procedure evsnrun
getn(y+idx);keyword 'IMNAME';gethead;
getn(x);sources keystrng;go evasn;wait;
finish

procedure evsnproc
task 'evasn'
BIF 1;EIF 14;INEXT 'SN';INVERS 2;
y=0;x=0
if (n>1) then;y=scannum(1);x=scannum(1);end
if (n>2) then;y=y+scannum(2);x=x+scannum(2);end
x=x*3+3+5*(n-1)+4;
y=y*3+3+5*n;
typ n
for idx=1:scannum(n);evsnrun;typ idx;end
finish

for n=1:runs;evsnproc;end
```

BLCMP.001 - Compare Baseline Amplitudes

```
procedure BLcomp
task 'uvprt'
indisk dsk;string*2 prtrun
scalar runn
if (n=1) then;runn=7;prtrun=minrun;end
if (n=2) then;runn=7+(n-1)*5+3*scannum(n-1);prtrun=midrun;end
if (n=3) then;runn=7+(n-1)*5+3*scannum(n-1)+3*scannum(n-2);prtrun=maxrun;end
getn runn;sources '';docrt -1;
antennas lonA1 lonA2 lonA3;baseline lonB1 lonB2 lonB3;
outprint 'AIPSOUT:BLcomp_long_'!!prtrun!!'.TXT';
go uvprt;wait;
antennas shA1 shA2 shA3;baseline shB1 shB2 shB3;
outprint 'AIPSOUT:BLcomp_short_'!!prtrun!!'.TXT';
go;wait;
finish

for n=1:runs;BLcomp;end
```

CNTRS.001 - Generate Contour Plots

```
procedure contourrun
task 'CNTR'
getn(y+idx);blc 0;trc 0;
PLEV 10;LEVS -1 1 2 4 8 10;
DOTV -1;go cntr;wait;
STRING*8 src
keyword'IMNAME';gethead;src=substr(keystrng,1,7);
task 'lwpla'
plver 0;invers 0;docolor 0;
if (n=1) then;OUTFILE 'AIPSOUT:'!!minrun!!'_'!!src!!'_contour.ps';end
if (n=2) then;OUTFILE 'AIPSOUT:'!!midrun!!'_'!!src!!'_contour.ps';end
if (n=3) then;OUTFILE 'AIPSOUT:'!!maxrun!!'_'!!src!!'_contour.ps';end
go;wait
finish

procedure cntrproc
y=0;
if (n>1) then;y=scannum(1);x=scannum(1);end
if (n>2) then;y=y+scannum(2);x=x+scannum(2);end
y=y*3+3+5*n+scannum(n)*2;
for idx=1:scannum(n);contourrun;end
finish

for n=1:runs;cntrproc;end
```

A.2. Data Reduction: Perl Script

```
#!/usr/bin/perl -w
use strict;

=explain
-------------------------------------------------------------------------
  usage: master_comp_interface.pl output CalSource minrun maxrun inpath knwn_name unwn_name
-------------------------------------------------------------------------
By Cameron Trapp(ctrapped@gmail.com) originated UNM/NRAO
-------------------------------------------------------------------------
Reduces data from AIPS outputs generated from getJy, evasn, evauv,
uvprt, and Imean in order to extract parameters for Ranking. Interfaces
with cal_identifier.py. Generates a variety of outputs, including reduced
files for each task, a final reduced source list, and optional outputs
for Latex and Matlab data display.
=cut

#########################################################################

use Math::Trig;
use Math::Trig ':pi';
use Math::BigFloat;


my$filename = $ARGV[0]; #output filename fed from master script
my$calsource = $ARGV[1]; #source to use as calibrator
my$minrun= $ARGV[2];
my$maxrun= $ARGV[3];
my$irpath = $ARGV[4];
my$knwn_name = $ARGV[5];
my$unwn_name = $ARGV[6];

my$knwn1 = substr($knwn_name,0,1); #First letters of the naming conventions
my$unwn1 = substr($unwn_name,0,1);

my$threshold = .01; #Threshold for RMS across IF channels
my$SNR_TH = 10;
my$ampcal=9; #Change if you forget to set flux of primary calibrator in aips

#my$irpath = "\/home\/cameron\/AIPS_outputs\/Todos";
my$irout = "$irpath\/Reduced"; #Make sure to create a Reduced folder in your path
my$getjyOut  = "$irout\/getjy_reduced.txt";
my$getjySource;

$filename ||= "CalList_equal\.txt"; #default filename
my$outfile = "$irout\/$filename"; #output file path

#Optional features
my$cntrtab = "$irout\/latex_cntr.txt"; #generates latex file to plot ranked contours
my$tbletab = "$irout\/latex_tble.txt"; #generates latex file to create ranked table
my$ltabG = "$irout\/matlab_larray_good.txt"; #Matlab vectors for plotting positions in gal. coord.
my$btabG = "$irout\/matlab_barray_good.txt";
my$ltabB = "$irout\/matlab_larray_bad.txt";
my$btabB = "$irout\/matlab_barray_bad.txt";
my$ltabKNWN = "$irout\/matlab_larray_known.txt";
my$btabKNWN = "$irout\/matlab_barray_known.txt";

##Ranking Parameters##
my$rank_th = 995; #Good calibrator threshold
my$A = 50; #Model Residuals
my$B1 = 2; #SNR below 100
my$B2 = 1/10; #SNR above 100
my$C = 0.01; #RMS across antennae
my$D1 = 0.01; #Baseline Ratio for low SNR
my$D2 = 1000; #Baseline Ratio for good SNR
my$Dbl;my$Bsnr;


unlink ("$getjyOut") if (-e $getjyOut);
open OUTT, ">$getjyOut" or die "Cannot open file $getjyOut";
```

```
my$SourceOut = "$irout\/source_list.txt";
open OUTT2, ">$SourceOut" or die "Cannot open file $SourceOut";
print OUTT2 "User ID   Run    Source\n";
print OUTT2
"-------------------------------------------------------------------------------------------------------------
\n";


#############################################################################
##REDUCE GETJY DATA
#############################################################################
#############################################################################
my$tick = 1;my@source;my$source; #Initialize variables
my$sourcenum = 1;my$ii;
my$flag = 0;
my$sourcename;
my@IFval;my@errval;
my@IF;my@errors;
my$IFsum=0;my$errsum=0;
my$IFavg;my$erravg;
my$IFdiff;my$errdiff;
my$stdv;my$newpage;
my$newflag = 0;my$newcount = 0;
my$test;my$stdv_norm;my$IFnum;my$userid;
my$tmp;my$run;my$runcheck;
my@getjy_sources;my@getjy_run;my$mattick=0;
my@getjy_rms;
for(my$idx = $minrun;$idx < $maxrun;$idx=$idx+3) { #Loop through all files generated
        $tmp = $idx+2;
        $run = $idx;
        $getjySource = "$irpath\/GETJY_$idx-$tmp.OUT"; #Find correct file
        open SOURCET, "$getjySource" or die "Cannot open file $getjySource";
        my$linetracker = 1;

        while (<SOURCET>) { #Loop through file
                $test = 0;$test = 1 unless /\S/; #Make sure not on blank line
                $newpage = (split ' ')[1]; #Check to see if you hit a new page
                if ($test == 1) {
                        $newpage = "blank";
                }
                if ($linetracker == 2) {
                        $userid = (split ' ')[8]; #pull out userid
                        print OUTT2 "$userid        $run       $calsource\n"; #getjy ignores primary
calibrator
                        printf OUTT "%-10s       %-10s       %-10s        %-10s\n",$calsource,0,0,$run;
                        $getjy_sources[$mattick] = $calsource;
                        $getjy_run[$mattick] = $run;$getjy_rms[$mattick] = 0; #Save for future use
                        $mattick++;
                }

                $runcheck = (split ' ')[8]; #check if run has ended to add new primary cal to list
                $runcheck ||= 'BLANK';
                if ($runcheck eq 'ended') {
                        $run = $run +1;
                        print OUTT2 "$userid        $run       $calsource\n"; #getjy ignores primary
calibrator
                        printf OUTT "%-10s       %-10s       %-10s        %-10s\n",$calsource,0,0,$run;
                        $getjy_sources[$mattick] = $calsource;
                        $getjy_run[$mattick] = $run;$getjy_rms[$mattick] = 0;
                        $mattick++;
                }
                $linetracker=$linetracker+1; #increment line
                $IFsum = 0; #intitialize sums
                $errsum = 0;
                $stdv = 0;
                $source = (split ' ')[5]; #grab name of source
                $source ||= 'BLANK';
                $IFnum = (split ' ')[9];
```

```
                @source = split('',$source);


                if (($source[0] eq $knwn1) || (($source[0] eq $unwn1) && ($source[1] eq 'A'))) {
                        print OUTT2 "$userid        $run       $source\n"; #Print reduced file
                }


                my$letter1 = substr($source,0,3);
                if ((($letter1 eq $unwn_name) || ($source[0] eq $knwn1)) && ($IFnum == 1)) {
                        $flag = 1; #Flag for first IF of source
                }


                if ($newpage eq "AIPS") {
                        $newflag = 1; #flag for skipping to next page
                }
                if ($newflag==1) {
                        $newcount = $newcount+1; #count for new pages
                }
                if ($newcount ==4) { #only 3 runs/pages per user
                        $newflag = 0;
                        $newcount = 0;
                }

                if (($flag == 1) && ($newflag == 0))  { #Read off various values if on right line
                        if ($tick == 1) {
                                $IFval[$tick-1] = (split ' ')[10]; #Value for IF
                                $errval[$tick-1] = (split ' ')[13]; #error
                                $errval[$tick-1] ||=100;
                                $sourcename = $source;
                        }
                        if (($tick < 15) && ($tick>1)) { #Different format for successive lines
                                $IFval[$tick-1] = (split ' ')[6];
                                $errval[$tick-1] = (split ' ')[9];
                                $errval[$tick-1] ||=100;
                        }
                        $tick++; #Next IF
                        if ($tick == 15) { #All IFs measured, calculate stats
                                for ($ii = 0;$ii<14;$ii++) {
                                        $IFsum = $IFsum + $IFval[$ii];
                                        $errsum = $errsum + $errval[$ii];
                                }
                                $IFavg = $IFsum/14;$erravg = $errsum/14;
                                for (my$jj = 0;$jj<14;$jj++) {
                                        $IFdiff = $IFavg-$IFval[$jj];$errdiff = $erravg-$errval[$jj];
                                        $stdv = $stdv + 1/14*$IFdiff**2;
                                        $stdv_norm = $stdv/$IFavg;
                                }

                                if ($stdv < $threshold) { #Print out stats for detected sources only
                                        printf OUTT "%-10s       %-10s       %-10s        %-10s\n",
$sourcename,$stdv,$stdv_norm,$run;
                                        $getjy_sources[$mattick] = $sourcename;
                                        $getjy_run[$mattick] = $run;$getjy_rms[$mattick] = $stdv;
                                        $mattick++;
                                }
                                $tick = 1;
                                $flag = 0;
                        }
                }


        }
        close SOURCET;
}

my$getjy_tick = $mattick;
close OUTT;close OUTT2;
```

```
print "Getjy reduced\n";


###############################################################################
##REDUCE EVAUV DATA
###############################################################################
###############################################################################

my$evauvSource;
my$evauvOut  = "$irout\/EVAUV_reduced\.txt";

unlink ("$evauvOut") if (-e $evauvOut);

open OUTT, ">$evauvOut" or die "Cannot open file $evauvOut";

my$check1;my$check2;my$flag1;my$flag2;my$skipflag;
my@RPsub;my@RPsub_err;my@IPsub;my@IPsub_err;my@Ampsub;my@Ampsub_err;my@badamp_sub;
my@RPdiv;my@RPdiv_err;my@IPdiv;my@IPdiv_err;my@Ampdiv;my@Ampdiv_err;my@badamp_div;
my@names;my$prnt1;my$prnt2;my$prnt3;
my$intab;

$mattick=0;my@evauv_sources;my@evauv_divmod;my@evauv_submod;

print OUTT "Source   Subtraction Model Amp    Division Model Amp \n";
print OUTT
"-------------------------------------------------------------------------------------------------------------------
\n";
my$tst;
$tick=-1;
my$jj;
my$screwupflag=0; #For erroneous lines printed by AIPS
for (my$ii=$minrun;$ii<$maxrun;$ii=$ii+3) { #Loop through all user files generated
        $tmp = $ii+2;
        $intab = "$irpath\/EVAUV_$ii-$tmp\.TXT";
        open SOURCET, "$intab" or die "Cannot open file $intab";
        $flag1=0;$flag2=0;
        while (<SOURCET>) { #Loop through evauv file
                $check1 = (split' ')[4];
                $check1 ||= 'BLANK';
                if ($check1 eq "EVAUV") { #Check what line you are on
                        $check2 = (split' ')[5];
                        if ($check2 eq "Task") {
                                $tick = $tick + 1;
                        }

                        if ($check2 eq "IGNORING") { #Gives name of the source
                                $names[$tick] = (split' ')[8];
                        }

                        if ($check2 eq "subtract") { #Subraction model info
                                $skipflag = 0;
                                if ($flag1==0) {
                                        $RPsub[$tick]=(split' ')[6];$RPsub_err[$tick]=(split' ')[8];
                                        $IPsub[$tick]=(split' ')[9];$IPsub_err[$tick]=(split' ')[11];
                                        $Ampsub[$tick]=(split' ')[12];$Ampsub_err[$tick]=(split' ')[14];
                                        $flag1 = 1;
                                        $skipflag = 1;
                                }
                                if (($skipflag == 0) && ($flag1==1)) { #Secondary sub. model info
                                        $badamp_sub[$tick]=(split' ')[8];
                                        $flag1 = 0;
                                }
                        }

                        $screwupflag=0;

                        if ((substr($check2,0,8) eq "divide-1") && ($check2 ne "divide-1")) {#Some lines
don't separate correctly
                                $check2 = "divide-1";
```

```
                        $screwupflag = 1;
                }
                if ($check2 eq "divide-1") {


                        $skipflag = 0;
                        if ($flag2==0) { #Division model info
                                $RPdiv[$tick]=(split' ')[6];$RPdiv_err[$tick]=(split' ')[8];
                                $IPdiv[$tick]=(split' ')[9];$IPdiv_err[$tick]=(split' ')[11];
                                $Ampdiv[$tick]=(split' ')[12];$Ampdiv_err[$tick]=(split' ')[14];
                                if ($screwupflag) { #Different positions when lines are messed up
                                        $RPdiv[$tick]=substr((split' ')[5],9,13);$RPdiv_err
[$tick]=(split' ')[7];

                                        $IPdiv[$tick]=(split' ')[8];$IPdiv_err[$tick]=(split' ')
[10];
                                        $Ampdiv[$tick]=(split' ')[11];$Ampdiv_err[$tick]=(split'
')[13];
                                }
                                $flag2 = 1;
                                $skipflag = 1;
                                $Ampdiv[$tick] ||= 100; #If residuals are above 100 they screw
everything up

                        }
                        if (($skipflag == 0) && ($flag2==1)) { #secondary div. model info
                                $badamp_div[$tick]=(split' ')[8];
                                $flag2 = 0;
                        }
                }
        }
}


        close SOURCET;
}

        for ($jj=0;$jj<=$tick;$jj++) { #Save reduced model stats
                $prnt1 = $names[$jj];$prnt2 = $Ampsub[$jj];$prnt3=$Ampdiv[$jj];
                print OUTT "$prnt1     $prnt2    $prnt3\n";
                $evauv_sources[$mattick] = $prnt1;$evauv_divmod[$mattick] = $prnt3;
                $evauv_submod[$mattick]=$prnt2;
                $mattick++;
        }

close OUTT;
print "EVAUV data reduced\n";

my$evauv_tick=$mattick;



################################################################################
##REDUCE EVASN DATA
################################################################################
################################################################################
my$EVASNSource;
my$EVASNOut  = "$irout\/EVASN_reduced\.txt";

unlink ("$EVASNOut") if (-e $EVASNOut);

open OUTT, ">$EVASNOut" or die "Cannot open file $EVASNOut";

$tick=-2;my$linecount=0;
my$check3;my@Rgains;my@Lgains;my@Rrms;my@Lrms;
my@userid;my@sourcenames;
```

```
my$prnttest;


my$Rgainsum;my$Rrmssum;my$Lgainsum;my$Lrmssum;my$RgainRMS;my$LgainRMS;
my$sourcetick2;my$brkflag;
my$Rgainavg;my$Rrmsavg;my$Lgainavg;my$Lrmsavg;

$mattick=0;my@evasn_sources;my@evasn_Rrms;my@evasn_Lrms;
my@evasn_bad;my@evasn_ex;


print OUTT "Source                  Avg R Gain              RMS of Gain             R
RMS                   AVG L Gain              RMS of Gain             L RMS                   Bad
IF                  Excess IF \n";
print OUTT
"-------------------------------------------------------------------------------------------------------------------
\n";


my$badIF;my$exIF;my$sourcetick;
my$sourcetab = "$irout\/source_list.txt";
open SOURCET2, "$sourcetab" or die "Cannot ccopen source file: $sourcetab";
my@nummat;my$useridx=1;
while (<SOURCET2>) {
        $tick++;
        if ($tick>0) {
                $userid[$tick-1] = (split' ')[0];
                $sourcenames[$tick-1] = (split' ')[2];
                $linecount++;
                if ($tick==1) {
                        $nummat[0] = $userid[0];
                }
                if ($tick>1) {
                        if ($userid[$tick-1] != $userid[$tick-2]) {
                                $nummat[$useridx] = $userid[$tick-1];
                                $useridx++;
                        }
                }
        }
}


close SOURCET2;


my$num;
$useridx = 0;
for (my$ii=$minrun;$ii<$maxrun;$ii=$ii+3) { #Loop through all user files generated
        $tmp = $ii+2;
        $intab = "$irpath\/EVASN_$ii-$tmp\.TXT";


        $num = $nummat[$useridx];

        open SOURCET, "$intab" or die "Cannot open file $intab";


        $flag1=0;
        $tick=-3;
        $sourcetick = 0;
        $badIF=0;
        $exIF=0;
        while (<SOURCET>) { #Loop through evasn files
           $check1 = (split' ')[5];$check2=(split' ')[4];
           $check1 ||= "BLANK";$check2||="BLANK";
           if ($check2 eq 'EVASN') { #Check if you hit a new source
                if ($check1 eq "Task") {
                        $flag1 =1;
                        $sourcetick++;
```

```
                }
                if ($flag1>0) { #Record info for source
                        $tick++;
                        if  (($tick>0) && ($tick<15)) {
                                if ($check1 eq "*") { #different lines have different formats
                                        $Rgains[$tick-1] = (split' ')[7];
                                        $Rrms[$tick-1] = (split' ')[8];
                                        $Lgains[$tick-1] = (split' ')[9];
                                        $Lrms[$tick-1] = (split' ')[10];
                                }
                                if ($check1 ne "*") { #these have bad IFs
                                        $Rgains[$tick-1] = 0;
                                        $Rrms[$tick-1] = 0;
                                        $Lgains[$tick-1] = 0;
                                        $Lrms[$tick-1] = 0;
                                        $badIF++;
                                }

                        }

                        if (($tick > 16) && ($tick <31)) {
                                if ($check1 eq "*") { #these have excesses
                                        $exIF++;
                                }

                        }


                        if($tick > 30) {
                                $flag1 = 0;
                                $tick = -3;
                                $sourcetick2 = 0;
                                $brkflag=0;

                                for(my$idx=0;(($idx<$linecount) && ($brkflag !=1));$idx++) { #Find the
correct source name (not rec. in aips file)
                                        $prnttest = $userid[$idx];
                                        if ($prnttest == $num) {
                                                $sourcetick2++;
                                                if ($sourcetick2==$sourcetick) {
                                                        $source = $sourcenames[$idx];
                                                        $brkflag=1;
                                                }
                                        }
                                }

                                $Rgainsum=0;$Rrmssum=0;$Lgainsum=0;$Lrmssum=0; #Calculate the sums and
means for sources
                                for(my$idx2=0;$idx2<14;$idx2++) {
                                        $Rgainsum = $Rgainsum+$Rgains[$idx2];
                                        $Rrmssum = $Rrmssum+$Rrms[$idx2];
                                        $Lgainsum = $Lgainsum+$Lgains[$idx2];
                                        $Lrmssum = $Lrmssum+$Lrms[$idx2];
                                }
                                $Rgainavg = $Rgainsum/14;$Rrmsavg=$Rrmssum/14;$Lgainavg=$Lgainsum/14;
$Lrmsavg=$Lrmssum/14;
                                $RgainRMS=0;$LgainRMS=0;
                                for(my$idx3=0;$idx3<14;$idx3++) {
                                        $RgainRMS = ($Rgains[$idx3] - $Rgainavg)**2+$RgainRMS;
                                        $LgainRMS = ($Lgains[$idx3] - $Lgainavg)**2+$LgainRMS;
                                }

                                $RgainRMS = ($RgainRMS/14)**(1/2);$LgainRMS=($LgainRMS/14)**(1/2);

                                printf OUTT "$source    %-20s    %-20s    %-20s    %-20s    %-20s
%-20s    %-20s    %-20s\n",$Rgainavg,$RgainRMS,$Rrmsavg,$Lgainavg,$LgainRMS,$Lrmsavg,$badIF,$exIF;
                                $badIF=0;$exIF=0;
                                $evasn_sources[$mattick]=$source;
```

```
                                $evasn_Rrms[$mattick]=$Rrmsavg;$evasn_Lrms[$mattick]=$Lrmsavg;
                                $evasn_bad[$mattick]=$badIF;$evasn_ex[$mattick]=$exIF;
                                $mattick++;
                        }
                }


           }
        }


        close SOURCET;
$useridx++;
}

my$evasn_tick=$mattick;
close OUTT;
print "EVASN data reduced\n";


################################################################################
#REDUCE UVPRT DATA (BLCOMP)
################################################################################
################################################################################


$tick=0;
my $shortSource;my$longSource;my$srcheck;my$srcflag=0;my$newrunflag=0;
my@uvprtnames;my $ampcheck;my$firstflag;
my$ants;my$ampRR;my$ampLL;
my$ampRRavg;my$ampLLavg;
my@ampRRshort;my@ampLLshort;
my@ampRRlong; my@ampLLlong;
my$srctick=0;my$srctick2=0;
my$prt1;my$prt2;
my@BLshortnames;
my@BLlongnames;

my@uvprtShort_sources;my@uvprtLong_sources;
my@uvprt_RRshort;my@uvprt_RRlong;
my@uvprt_LLshort;my@uvprt_LLlong;
my$mattickS=0;my$mattickL=0;

my$uvprtOut = "$irout\/BLCOMP_short_reduced.TXT"; #Reduced Files
my$uvprtOut2 = "$irout\/BLCOMP_long_reduced.TXT";
unlink ("$uvprtOut") if (-e $uvprtOut);
open OUTT, ">$uvprtOut" or die "Cannot open file $uvprtOut";
open OUTT2, ">$uvprtOut2" or die "Cannot open file $uvprtOut2";
print OUTT "Source      RR                    LL\n";
print OUTT "-----------------------------------------------------\n";
print OUTT2 "Source      RR                    LL\n";
print OUTT2 "-----------------------------------------------------\n";
for(my$idx = $minrun;$idx < $maxrun;$idx++) { #Loop through all sources
        if ($idx != 15) { #!!!Issue with run 15 in our dataset, delete for future use!!!
        $shortSource = "$irpath\/BLCOMP\_SHORT\_$idx\.TXT";
        $longSource = "$irpath\/BLCOMP\_LONG\_$idx\.TXT";
        open SOURCET, "$shortSource" or die "Cannot open file $shortSource";
        open SOURCET2, "$longSource" or die "Cannot open file $longSource";

        $firstflag = 1;
        $ampRRavg=0;$ampLLavg=0;$tick=0;
        while (<SOURCET>) { #Loop through short baseline information
                $srcheck = (split' ')[0];
                $srcheck ||= "BLANK";
                $ampcheck = substr($srcheck,1,1);


                if ($srcheck eq "Source") { #new source
                        if ($firstflag ==0) {
```

```
                                #PRINT OUT STATS, CLEAR VARIABLES
                                $ampRRshort[$srctick] = $ampRRavg/$tick;
                                $ampLLshort[$srctick] = $ampLLavg/$tick;
                                $BLshortnames[$srctick] = $sourcename;
                                $prt1 = $ampRRshort[$srctick];$prt2 = $ampLLshort[$srctick];

                                printf OUTT "%-8s    %-16s    %-16s\n",$sourcename,$prt1,$prt2;


                                $sourcename = (split' ')[1]; #Get next source name
                                $srctick++;$tick=0;
                                $ampRRavg=0;$ampLLavg=0;

                        } else {
                        $firstflag = 0; #Get sourcename if first source
                        $sourcename = (split' ')[1];
                        }
                }



                if ($ampcheck eq '/') { #Find which line you're on
                        $tick = $tick+1;
                        $ants = (split' ')[1];
                        if (length($ants)==2) { #Different lines have diff. formats depending on length
of numbers
                                $ampRR = (split' ')[6];
                                if (length($ampRR) > 5) {
                                        $ampRR = substr($ampRR,0,5);
                                        $ampLL = (split' ')[7];

                                } else {
                                        $ampLL = (split' ')[8];
                                }
                                if (length($ampLL) > 5) {
                                        $ampLL = substr($ampLL,0,5);
                                }
                        } else {
                                $ampRR = (split' ')[5];
                                if (length($ampRR) > 5) {
                                        $ampRR = substr($ampRR,0,5);
                                        $ampLL = (split' ')[6];

                                } else {
                                        $ampLL = (split' ')[7];
                                }
                                if (length($ampLL) > 5) {
                                        $ampLL = substr($ampLL,0,5);
                                }
                        }
                        $ampRRavg = $ampRRavg + $ampRR; #Calculate running sum
                        $ampLLavg = $ampLLavg + $ampLL;
                }
        }
        $ampRRshort[$srctick] = $ampRRavg/$tick; #Calculate averages
        $ampLLshort[$srctick] = $ampLLavg/$tick;
        $prt1 =$ampRRshort[$srctick];$prt2 = $ampLLshort[$srctick];
        $BLshortnames[$srctick] = $sourcename;
        print OUTT "$sourcename    $prt1     $prt2\n";


        $firstflag = 1;
        $ampRRavg=0;$ampLLavg=0;$tick=0;

        while (<SOURCET2>) { #Loop through long baselines
                $srcheck = (split' ')[0];
                $srcheck ||= "BLANK";
                $ampcheck = substr($srcheck,1,1);
```

```
                if ($srcheck eq "Source") {
                        if ($firstflag ==0) { #new source

                            #PRINT OUT STATS, CLEAR VARIABLES
                                $ampRRlong[$srctick2] = $ampRRavg/$tick;
                                $ampLLlong[$srctick2] = $ampLLavg/$tick;
                                $BLlongnames[$srctick2] = $sourcename;
                                $prt1 = $ampRRlong[$srctick2];$prt2 = $ampLLlong[$srctick2];
                                printf OUTT2 "%-8s    %-16s    %-16s\n",$sourcename,$prt1,$prt2;
                                $sourcename = (split' ')[1]; #new sourcename
                                $srctick2++;$tick=0;
                                $ampRRavg=0;$ampLLavg=0;


                        } else {
                        $firstflag = 0; #only record name if first source
                        $sourcename = (split' ')[1];
                        }
                }



                if ($ampcheck eq '/') {
                        $tick = $tick+1;
                        $ants = (split' ')[1];
                        if (length($ants)==2) { #Different lines have diff. formats depending on length
of numbers
                                $ampRR = (split' ')[6];
                                if (length($ampRR) > 5) {
                                        $ampRR = substr($ampRR,0,5);
                                        $ampLL = (split' ')[7];

                                } else {
                                        $ampLL = (split' ')[8];
                                }
                                if (length($ampLL) > 5) {
                                        $ampLL = substr($ampLL,0,5);
                                }
                        } else {
                                $ampRR = (split' ')[5];
                                if (length($ampRR) > 5) {
                                        $ampRR = substr($ampRR,0,5);
                                        $ampLL = (split' ')[6];

                                } else {
                                        $ampLL = (split' ')[7];
                                }
                                if (length($ampLL) > 5) {
                                        $ampLL = substr($ampLL,0,5);
                                }
                        }
                        $ampRRavg = $ampRRavg + $ampRR; #Calc. running sum
                        $ampLLavg = $ampLLavg + $ampLL;
                }
        }



        $ampRRlong[$srctick2] = $ampRRavg/$tick; #Calculate average
        $ampLLlong[$srctick2] = $ampLLavg/$tick;
        $prt1 = $ampRRlong[$srctick2];$prt2 = $ampLLlong[$srctick2];



        $BLlongnames[$srctick2] = $sourcename;
        print OUTT2 "$sourcename    $prt1     $prt2\n";
```

```
        close SOURCET;
        close SOURCET2;
}
}

close OUTT;
close OUTT2;
print "UVPRT Data reduced.\n";

$evasn_tick = $mattick;

################################################################################
################################################################################
##COMBINE AND RANK
################################################################################


my$tst1;my$tst2;my$tst3;my$tst4;

#Rotation Matrix for Equatorial to Galactic
my$R11=-0.054876;my$R12=-0.873437;my$R13=-0.483835;
my$R21=0.494109;my$R22=-0.444830;my$R23=0.746982;
my$R31=-0.867666;my$R32=-0.198076;my$R33 = 0.455984;
my$RArad;my$decrad;my$x1;my$x2;my$y1;my$y2;my$z1;my$z2;
my$l;my$b;my@lprt;my@bprt;

my@nameprt;my@RAprt1;my@RAprt2;my@RAprt3;my@Decprt1;my@Decprt2;
my@Decprt3;my@snrprt;my@rmsprt;my@peakprt;my@stdevprt;my@subresprt;
my@divresprt;my@rmsRprt;my@rmsLprt;my@exIFsprt;my$fullname;

my@Rank;
my$intab2;
my$intab3;
my$outtab  = "$outfile";

unlink ("$outtab") if (-e $outtab); #Final out
unlink ("$cntrtab") if (-e $cntrtab); #latex file for contours
unlink ("$tbletab") if (-e $tbletab); #latex file for table
unlink ("$ltabG") if (-e $ltabG); #matlab vector for good sources
unlink ("$btabG") if (-e $btabG);
unlink ("$ltabB") if (-e $ltabB); #matlab vector for bad sources
unlink ("$btabB") if (-e $btabB);
unlink ("$ltabKNWN") if (-e $ltabKNWN); #matlab vector for known sources
unlink ("$btabKNWN") if (-e $btabKNWN);

open OUTT, ">$outtab" or die "Issue with output, Cannot open file $outtab";
open CNTROUTT, ">$cntrtab" or die "Issue with output, Cannot open file $cntrtab"; #For plotting contours
in latex
open TBLEOUTT, ">$tbletab" or die "Issue with output, Cannot open file $tbletab"; #For generating tables
in latex

open LOUTT, ">$ltabG" or die "Issue with output, Cannot open file $ltabG"; #for Gal coordinate vectors in
matlab
open BOUTT, ">$btabG" or die "Issue with output, Cannot open file $btabG";
open LOUTT2, ">$ltabB" or die "Issue with output, Cannot open file $ltabB";
open BOUTT2, ">$btabB" or die "Issue with output, Cannot open file $btabB";
open LKNWN, ">$ltabKNWN" or die "Issue with output, Cannot open file $ltabKNWN";
open BKNWN, ">$btabKNWN" or die "Issue with output, Cannot open file $btabKNWN";

my@runs;
my$name;my$name2;
my@line;my@RAs;my@decs;my@rms;my@rmshist;my@peaks;my@flux;my$stdev;
my$rmsval;my$rmshistval;my$fluxval;my$peakval;my$snr;my$snrhist;
$linecount=1;
my$rmsnoise;my$fluxdensity;my$peak;my$rmsnoisehist;
my$errflag;my$errflag2;my$subres;my$divres;
my$BLratio;my$shortampRR;my$shortampLL;my$longampRR;my$longampLL;my@BLratioprt;
my$l_sig;my$b_sig;my$peakval_sig;
```

```
#Final list format
print OUTT "Source    Rank     [RA              Dec          ]     l(deg)      b(deg)     Peak/RMS
Noise      RMSNoise(jy)  Peak(jy)      Long BL/Short BL     Sub Res       Div Res       R-
RMS                  L-RMS               IF Std. Dev\n";
print OUTT
"-------------------------------------------------------------------------------------------------------------------
\n";
print TBLEOUTT "\\begin{table}[h]\n\\begin{center}\n\\begin{tabular}{||c c c c c c c ||}\n";
print TBLEOUTT "\\hline\nSource & RA & Dec & l & b & Flux (Jy) & Rank \\\\ [0.5ex]\n\\hline\n";
print LOUTT "lG=[";
print BOUTT "bG=[";
print LOUTT2 "lB=[";
print BOUTT2 "bB=[";
print LKNWN "lK=[";
print BKNWN "bK=[";

my$ticker=0;
#THRESHOLDED GETJY SOURCES MATCHED WITH OTHER STATS
for (my$ii=0;$ii<$getjy_tick;$ii++){
  $fullname = $getjy_sources[$ii]; #pull out source name for matching
  $stdev = $getjy_rms[$ii];
  $run = $getjy_run[$ii]; #pull out run num for keeping track of contour plots
  $name = substr($fullname,0,7); #Names truncated for files generated by AIPS

  $intab2 = "$irpath\/IMEAN$name\_NOISE\.TXT";
  $intab3 = "$irpath\/IMEAN$name\_SOURCE\.TXT";


  #print "Working on $name...\n";
  open SOURCET2, "$intab2" or die "Issue with IMEAN noise, Cannont open file $intab2";
  open SOURCET3, "$intab3" or die "Issue with IMEAN source, Cannot open file $intab3";

  $errflag = 0;
  $errflag2 = 0;
  for (my$jj=0;$jj<$evauv_tick;$jj++) { #EVAUV evaultion
        $name2 = $evauv_sources[$jj];
        $name2 = substr($name2,0,7);
        if ($name eq $name2) {
                $subres = $evauv_submod[$jj];
                $divres = $evauv_divmod[$jj];
        }
   }


my$rmsR;my$rmsL;my$badIFs;my$exIFs;
 for ($jj=0;$jj<$evasn_tick;$jj++) { #EVASN evaulation
        $name2 = $evasn_sources[$jj];
        $name2 = substr($name2,0,7);
        if ($name eq $name2) {
                $rmsR = $evasn_Rrms[$jj];
                $rmsL = $evasn_Lrms[$jj];
                $badIFs = $evasn_bad[$jj];
                $exIFs = $evasn_ex[$jj];
        }
   }


  while (<SOURCET2>) { #IMEAN EVALUATION
        if ($linecount == 7) {
          @line = (split ' ')[2];
          $rmsnoise = join(' ',@line); #Pull out the noise

          if ($rmsnoise eq 'Rms=') { #Split depending on format
                @line = (split ' ')[3];
                $rmsnoise = join(' ',@line);
          }
          if ($rmsnoise eq 'JY/BEAM') {
                @line = (split ' ')[1];
```

```
                $rmsnoise = join(' ',@line);
          }
        }

        if ($linecount == 5) {
          @line = (split ' ')[2];
          $rmsnoisehist = join(' ',@line);

          if ($rmsnoisehist eq 'Rms=') {
                @line = (split ' ')[3];
                $rmsnoisehist = join(' ',@line);
          }
          if ($rmsnoisehist eq '****') {
                @line = (split ' ')[1];
                $rmsnoisehist = join(' ',@line);
          }
        }
        $linecount = $linecount+1;

  }
  close SOURCET2;
  @rms = split 'E',$rmsnoise;
  @rmshist = split 'E',$rmsnoisehist;
  $rmsval = $rms[0]*10**($rms[1]);
  $rmshistval = $rmshist[0]*10**($rmshist[1]);
  $linecount = 1;


  while (<SOURCET3>) { #MORE IMEAN
        if ($linecount == 8) { #pull out the flux
          @line = (split ' ')[3];
          $fluxdensity = join(' ',@line);
        }
        if ($linecount == 11) { #pull out the peaks
          @line = (split ' ')[1];
          $peak = join(' ',@line);
        }
        if ($linecount == 12) { #pull out the coordinates of the source
          @RAs = (split ' ')[2,3,4];
          @decs = (split ' ')[6,7,8];
        }
        $linecount = $linecount+1;
  }
  close SOURCET3;

  #Calculate galactic coordinates
  $RArad = ($RAs[0]+$RAs[1]/60+$RAs[2]/3600)*2*pi/24;
  $decrad = ($decs[0] + $decs[1]/60 + $decs[2]/3600)*2*pi/360;

  $x1 = cos($decrad)*cos($RArad); #Convert to cartesian
  $y1 = cos($decrad)*sin($RArad);
  $z1 = sin($decrad);

  $x2 = $R11*$x1 + $R12*$y1 + $R13*$z1; #Rotate to Galactic
  $y2 = $R21*$x1 + $R22*$y1 + $R23*$z1;
  $z2 = $R31*$x1 + $R32*$y1 + $R33*$z1;

  $l = atan($y2/$x2);
  $b = atan($z2/($x2**2+$y2**2)**(1/2)); #Convert back to Spherical

  if ($x2 < 0) { #Verify correct Quadrant
        $l = $l+pi;
  }
$b=$b*360/2/pi; #Convert to degrees
$l=$l*360/2/pi;



  $linecount = 1;
```

```
  @flux = split 'E',$fluxdensity; #pull out fluxes
  @peaks = split 'E',$peak;
  $fluxval = $flux[0]*10**($flux[1])*$ampcal;
  $peakval = $peaks[0]*10**($peaks[1]);
  $snr = $peakval/$rmsval; #calculate signal to nose
  $snrhist = $peakval/$rmshistval;

  $rmsR ||= 100*$C;$rmsL ||= 100*$C;

  #UVPRNT Baseline comp info
  for (my$kk=0;($kk <= $srctick);$kk++) {
        $name2 = $BLshortnames[$kk];
        $name2 = substr($name2,0,7);
        if ($name2 eq $name) {
                $shortampRR = $ampRRshort[$kk];
                $shortampLL = $ampLLshort[$kk];

        }
  }
  $shortampRR ||= 100;
  $shortampLL ||= 100;

  for (my$ll=0;($ll <= $srctick2);$ll++) {
        $name2 = $BLlongnames[$ll];
        $name2 = substr($name2,0,7);
        if ($name2 eq $name) {
                $longampRR = $ampRRlong[$ll];
                $longampLL = $ampLLlong[$ll];

        }
  }
 $longampRR ||= 1;
 $longampLL ||= 1;

 $BLratio = ($longampRR + $longampLL)/($shortampRR+$shortampLL);

################################################################################
##  Ranking Algorithm##
################################################################################
  if ($snr < 100) {
        $Bsnr = $B1*$snr
  }
  if ($snr >= 100) {
        $Bsnr = $B2*$snr+100*$B1
  }

  if ($snr < 10) {
        $Dbl = $D1*$BLratio
  }
  if ($snr >= 10) {
        $Dbl = $D2 *(($snr*$BLratio-1)/($snr-1));
  }

  $Rank[$ticker] = $A/$divres + $Bsnr + $C/($rmsR+$rmsL) + $Dbl;
################################
##Record all relevant info

  $nameprt[$ticker]=$fullname;
  $RAprt1[$ticker]=$RAs[0];$RAprt2[$ticker]=$RAs[1];$RAprt3[$ticker]=$RAs[2];
  $Decprt1[$ticker]=$decs[0];$Decprt2[$ticker]=$decs[1];$Decprt3[$ticker]=$decs[2];
  $lprt[$ticker]=$l;$bprt[$ticker]=$b;
  $snrprt[$ticker]=$snr;$rmsprt[$ticker]=$rmsval;$peakprt[$ticker]=$peakval;$stdevprt[$ticker]=$stdev;
$subresprt[$ticker]=$subres;
  $divresprt[$ticker]=$divres;$rmsRprt[$ticker]=$rmsR;$rmsLprt[$ticker]=$rmsL;$exIFsprt[$ticker]=$exIFs;
  $BLratioprt[$ticker]=$BLratio;$runs[$ticker]=$run;
  $ticker=$ticker+1;
}

print "Ranking...\n"; #Orders the ranks from high to low
```

```
my@Rank_copy = @Rank;
my$RankMax=0;my$idxidxer=0;my$termflag=0;my@idxmat;
my$RA1;my$RA2;my$RA3;my$decs1;my$decs2;my$decs3;my$rmsR;my$exIFs;my$Rank;my$rmsL;
while ($termflag==0) {
        $RankMax=0;
        for (my$looper=0;$looper<$ticker;$looper++) { #Find the highest rank
                if ($Rank[$looper] > $RankMax) {
                        $RankMax = $Rank[$looper];
                        $idxmat[$idxidxer]=$looper; #order index for this source
                        $tmp = $looper;
                }
        }
        $Rank[$tmp] = 0; #Ignore this index in future slopes
        $idxidxer++;
        if ($RankMax==0) {
                $termflag=1; #Stop after all have been ranked
        }
}

my$print_looper=0;
my$KCtick=0;my@KClist;my$goflag;
my$CCtick=0;my@CClist;

for (my$looper2=0;$looper2<$ticker;$looper2++) { #Loop through all sources in order and print out
        $ii = $idxmat[$looper2];
        $Rank = substr($Rank_copy[$ii],0,6);
        $name=$nameprt[$ii];$RA1=$RAprt1[$ii];$RA2=$RAprt2[$ii];$RA3=$RAprt3[$ii];
        $decs1=$Decprt1[$ii];$decs2=$Decprt2[$ii];$decs3=$Decprt3[$ii];
        $l=$lprt[$ii];$b=$bprt[$ii];
        $snr=$snrprt[$ii];$rmsval=$rmsprt[$ii];$peakval=$peakprt[$ii];$stdev=$stdevprt[$ii];
        $subres=$subresprt[$ii];$divres=$divresprt[$ii];$rmsR=$rmsRprt[$ii];$rmsL=$rmsLprt[$ii];$exIFs=
$exIFsprt[$ii];
        $BLratio = $BLratioprt[$ii];$run=$runs[$ii];
        $exIFs ||='?';

        $goflag=1;
        if (substr($name,0,1) eq $knwn1) { #Only list known calibrators once

                $termflag=0;
                for ($ii=0;(($ii<$KCtick) && ($termflag!=1));$ii++) {
                        if ($name eq $KClist[$ii]) {
                                $goflag = 0;

                                $termflag=1;
                        }
                }
                if ($goflag==1) {
                        $KClist[$KCtick] = $name;
                        $KCtick++;

                }

        }

        if (substr($name,0,1) eq $unwn1) { #Make sure there are no duplicate sources

                $termflag=0;
                for ($ii=0;(($ii<$CCtick) && ($termflag!=1));$ii++) {
                        if ($name eq $CClist[$ii]) {
                                $goflag = 0;
                                print("Warning:Repeated Calibrator Candidate name. Run $run, source $name
\n");
                                $termflag=1;
                        }
                }
                if ($goflag==1) {
                        $CClist[$CCtick] = $name;
                        $CCtick++;
```

```
                }

        }
        $peakval = $peakval*$ampcal;
        if ($goflag==1) {
                $l_sig = new Math::BigFloat $l;
                $b_sig = new Math::BigFloat $b;
                $peakval_sig = new Math::BigFloat $peakval;
                #Print final ranked table (.txt)
                printf OUTT "$name   $Rank   [$RA1 $RA2 %05d  $decs1 $decs2 %-10s]  %-10s   %-10s
%-16s    %-10s    %-10s    %-17s    %-10s    %-10s    %-20s    %-20s    %-20s\n",$RA3,$decs3,$l,$b,$snr,
$rmsval,$peakval,$BLratio,$subres,$divres,$rmsR,$rmsL,$stdev;
                #Print Contour plots for Latex
                printf CNTROUTT "\\includegraphics[scale=0.3]{..\/contours\/$run\_$name\_CONTOUR\.pdf}\n";
                #Print table Latext
                printf TBLEOUTT "\\hline\n $name & $RA1 h $RA2 m %.2f s & $decs1\\textdegree \$\\,\$
$decs2' %.2f'' & $l_sig\\textdegree & $b_sig\\textdegree & $peakval_sig & $Rank \\\\ \n",$RA3,$decs3;

                #Print galactic coords for Matlab analysis
                if (substr($name,0,1) eq $unwn1) {
                        if ($Rank >= $rank_th) {
                                printf LOUTT "$l,";
                                printf BOUTT "$b,";
                        }
                        else {
                                printf LOUTT2 "$l,";
                                printf BOUTT2 "$b,";
                        }
                }
                if (substr($name,0,1) eq $knwn1) {
                        printf LKNWN "$l,";
                        printf BKNWN "$b,";
                }


                if ((($print_looper+1) % 35)==0) && ($print_looper!=0)) {
                        print TBLEOUTT "\\hline\n\\end{tabular}\n\\end{center}\n\\end{table}\n";
                        print TBLEOUTT "\\begin{table}[h]\n\\begin{center}\n\\begin{tabular}{||c c c c c
c c ||}\n";
                        print TBLEOUTT "\\hline\nSource & RA & Dec & l & b & Flux (Jy) & Rank \\\\ [0.5ex]
\n\\hline\n";
                }
                $print_looper++;
        }
}

print TBLEOUTT "\\hline\n\\end{tabular}\n\\end{center}\n\\end{table}\n";
print LOUTT "];";print LOUTT2 "];";print LKNWN "];";
print BOUTT "];";print BOUTT2 "];";print BKNWN "];";

close OUTT;
close TBLEOUTT;close CNTROUTT;
close LOUTT;close LOUTT2;close LKNWN;
close BOUTT;close BOUTT2;close BKNWN;

print "Sources Ranked!\n"
```

A.3. Master file: Python Script

```
#-------------------------------------------------------------------------
#  usage: cal_identifier.py Aips_usernum minrun maxrun output
#-------------------------------------------------------------------------
#By Cameron Trapp(ctrapped@gmail.com) originated UNM/NRAO
#-------------------------------------------------------------------------
#Ranking pipeline and algorithm to automatically load, calibrate, analyze,
#reduce, and rank potential calibrator sources. Loads and calibrates sources
#with the doosro pipeline. Analyzes sources primarily based on flux and
#compactness utilizing the AIPS tasks getjy, imean, evasn evauv, and uvprt.
#Reduces data using the perl script master_comp_interface.pl in order to
#rank and order candiate calibrator sources.
#
#In order to use this pipeline you must define two environment variables
#AIPSOUT: where you want the outputs to go
#and
#PROCPATH: where the AIPS procedures for this pipeline are stored
#
#Additionally, you must edit the cal_identifier_parameters.txt file
#with the relevant information for your data.
#########################################################################


import time
import email
import getopt
import getpass
import glob
import numpy
import os
import re
import shutil
import subprocess
import sys
import time
import math
import numpy as np



paramf = open("cal_identifier_parameters.txt","r")
line = paramf.readline()
tmp= line.split()
refant= int(tmp[1])

lncnt = 0
while line: #Read off parameters
    line = paramf.readline()
    tmp= line.split()
    if (lncnt == 0):
        shortBL = [[int(tmp[1]),int(tmp[4]),int(tmp[7])],[int(tmp[2]),int(tmp[5]),int(tmp[8])]]
    if (lncnt == 1):
        longBL =  [[int(tmp[1]),int(tmp[4]),int(tmp[7])],[int(tmp[2]),int(tmp[5]),int(tmp[8])]]
    if (lncnt == 2):
        calsource=tmp[1]
    if (lncnt == 3):
        calamp = int(tmp[1])
    if (lncnt == 4):
        known_names = tmp[1]
    if (lncnt == 5):
        unknown_names = tmp[1]
    if (lncnt == 6):
        docal = int(tmp[1])
    if (lncnt == 7):
        docntr = int(tmp[1])
    if (lncnt == 8):
        disk = int(tmp[1])
    if (lncnt == 9):
        indisk = tmp[1]
    if (lncnt == 10):
```

```
        inname = tmp[1]
    lncnt=lncnt+1

paramf.close()


usrstart = int(sys.argv[1])
minrun = int(sys.argv[2])
maxrun = int(sys.argv[3])
filename = sys.argv[4]
usrs = math.ceil((maxrun-minrun+1)/3) #Only 3 runs per usernum seems to work
count=0

idsk_sz = len(indisk); #sizes for initialization in AIPS
innm_sz = len(inname);

irpath = os.environ['AIPSOUT'] #output path
testpath = os.environ['PROCPATH'] #aips procedures path

if (irpath == 'None'):
    print "Please define output environment variable AIPSOUT\n"
    sys.exit()
if (testpath == 'None'):
    print "Please define AIPS procedures environment variable PROCPATH\n"
    sys.exit()

print "Determining AIPS user #'s...\n"
usrlist = []
while (count < usrs): #Define the list of users
    usrlist.append(usrstart+count)
    count=count+1


#Define AIPS FileName
aipsFileName = "cal_identify.csh" #cshell

#Create file that we want to run
f = open(aipsFileName,'w')

#Write to file
count=0
print "Writing AIPS script...\n"
f.write("#!/bin/tcsh\n")
f.write("aips tv=local:7<<EOF\n\n") #Start AIPS

for usrnum in usrlist:
    startrun = minrun+(3*count)
    endrun = startrun+2

    if(endrun>maxrun): #Accounts for groups of runs not divisible by 3
        endrun=maxrun

    runs = endrun-startrun+1
    runarray = []
    runarray.append(startrun)
    if (runs>1):
         runarray.append(startrun+1)
    if (runs>2):
         runarray.append(startrun+2)

    f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
    f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
    f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
    f.write("\n%s\n" % int(usrnum)) #Enter usernumber
    f.write("version 'PROCPATH'\n") #Load procedure directory
    f.write("x=%s;y=%s\n" % (idsk_sz,innm_sz))
    f.write("procedure gx\n")
    f.write("    go;wait\n")
    f.write("finish\n\n")
```

```
    f.write("procedure initialize\n")
    f.write("    scalar n,m,x,y,idx,i,jj,dsk,RANT \n")
    f.write("    scalar startrun,endrun,runs,runn \n")
    f.write("    scalar shA1,shA2,shA3,shB1,shB2,shB3 \n")
    f.write("    scalar lonA1,lonA2,lonA3,lonB1,lonB2,lonB3\n")
    f.write("    string*x pth; \n")
    f.write("    string*y nms; \n")
    f.write("    string*10 calname; \n")
    f.write("finish\n\n initialize\n\n")
    f.write("runs=%s\n" % int(runs))
    f.write("dsk=%s\n" % int(disk))
    f.write("RANT=%s\n" % int(refant))
    f.write("CALNAME='%s'\n" % calsource) #INITIALIZE THIS IN AIPS AND CHANGE CAL SETTINGS
    if(docal):
        f.write("indsk=%s;\n" % indisk)
        f.write("innm='%s';\n" % inname)
        f.write("flux=%s;" % int(calamp))
        f.write("startrun=%s;" % int(startrun))
        f.write("endrun=%s;\n" % int(endrun))
        f.write("run CAL\n") #Loads data, calibrates with doosro
        f.write("compress\n")
        f.write("msgkill -1\n")
        f.write("run doosro\n\n")
        f.write("run CAL2\n\n")
    if(docal==0):
        f.write("task 'getjy';\n") #Get fluxes if calibration is not run
        f.write("inclass 'SPLAT0';inseq 1;calsour '%s';\n" % calsource)
        f.write("SNVER 2\n")
        f.write("INNAME 'RUN-%s';go getjy;wait\n" % int(startrun))
        f.write("INNAME 'RUN-%s';go getjy;wait\n" % int(startrun+1))
        f.write("INNAME 'RUN-%s';go getjy;wait\n" % int(endrun))
        f.write ("inp prtmsg;prtask 'getjy';prtime .1;docrt 0\n")
        f.write ("outprint 'AIPSOUT:GETJY_%s-%s.OUT';\n" % (int(startrun),int(endrun)))
        f.write ("prtmsg\n")


    f.write("procedure naming\n") #Naming Conventions
    f.write("    string*2 minrun,midrun,maxrun;\n")
    f.write("    getn 1;keyword 'IMNAME';gethead;minrun = substr(keystrng,5,6);\n")
    f.write("    if runs>1 then;getn 2;keyword 'IMNAME';gethead;midrun = substr(keystrng,5,6);\n")
    f.write("    if runs>2 then;getn 3;keyword 'IMNAME';gethead;maxrun = substr(keystrng,5,6);\n")
    f.write("finish\n\n")
    f.write("naming;wait\n\n")
    f.write("procedure counter\n") #Count sources
    f.write("    scalar n,x,y,idx;\n")
    f.write("    ARRAY scannum(3)\n")
    f.write("    getn x;inext'su';keyword'num row';getthead;nx=keyvalue(1);\n")
    f.write("    scannum(x)=nx;\n")
    f.write("finish\n")
    f.write("for x=1:runs;counter;end\nwait;\n\n")
    f.write("run SNRC;wait\n") #Runs imean for SNR purposes
    f.write("run MDLC;wait\n") #Runs evauv for model calculations

    f.write("inp prtmsg;prtask 'EVAUV';prtime .5;docrt 0\n")
    f.write("outprint 'AIPSOUT:EVAUV_%s-%s.TXT';\n" % (int(startrun),int(endrun)))
    f.write("docrt -1;prtmsg;wait;\n")




    f.write("run SNEVL;wait\n") #Runs evasn
    f.write("inp prtmsg;PRTASK 'EVASN';PRTIME .5;PRNUMBER 0;docrt 0\n")
    f.write("outprint 'AIPSOUT:EVASN_%s-%s.TXT';\n" % (int(startrun),int(endrun)))
    f.write("docrt -1;prtmsg;\n\n")

    #Define the baselines to use
    f.write("shA1=%s;shA2=%s;shA3=%s\n" % (shortBL[0][0],shortBL[0][1],shortBL[0][2]))
    f.write("shB1=%s;shB2=%s;shB3=%s\n" % (shortBL[1][0],shortBL[1][1],shortBL[1][2]))
    f.write("lonA1=%s;lonA2=%s;lonA3=%s\n" % (longBL[0][0],longBL[0][1],longBL[0][2]))
```

```
    f.write("run BLCMP;wait\n")
    if (docntr==1):
        f.write("run CNTRS;wait\n") #Plots contour maps
    f.write("run SNRC;wait\n") #Redundancy for consistency
    f.write("run SNEVL;wait\n")
    if (maxrun>endrun):
        f.write("restart\n\n") #Go to next user account
        f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
        f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
        f.write("echo -ne \"\\n\" | aips\n") #Presses Enter in AIPS
    count = count + 1
f.write("kleenex\n") #Close out
f.write("EOF\n") #End of file
#Close file
f.close()


#Change permissions so that we can run it
os.chmod(aipsFileName,0o755)

print "Running AIPS pipeline, please standby...\n"
start_time = time.time()
#Run the file
p = subprocess.Popen(aipsFileName, stdout=subprocess.PIPE, stderr=subprocess.PIPE, shell=True)
stdout, stderr = p.communicate()
aips_time = time.time() - start_time

print ("AIPS: Elapsed time %s\n" % (aips_time))



#####Run Compilation Script####
time.sleep(5)
print "Reducing Data...\n"
start_time = time.time()
plname = './master_comp_interface_04182016.pl' #Perl File name
pipe = subprocess.Popen(["perl",plname,str(filename),str(calsource),str(minrun),str(maxrun),str
(irpath),str(known_names),str(unknown_names)], stdin=subprocess.PIPE)


perl_time = time.time() - start_time
print ("PERL: Elapsed time %s\n" % (perl_time))
```

A.4. Parameters File

```
refant=    14
shortBaselines=   9 11 , 14 26 , 20 27
longBaselines=    8 12 , 3 12 , 3 8
CalibratorSource=    J1733-1304
CalibratorAmplitude=    9
KnownCalibratorNamingConvention=    J
CandidateNamingConvention=    CAL
docal=    1
docntr=    1
disknum=    1
indisk=    /media/cameron/ymp2tb06/ylva/15A426/
inname=    15A426_
```

B. Data Tables

Source	RA	Dec	l	b	Flux (Jy)	Rank
J1733-1304	17 h 33 m 2.71 s	-13° 04' 49.55"	12°	11°	9.0	2041.0
J1851+0035	18 h 51 m 46.72 s	00° 35' 32.41"	33°	0.19°	2.6	1982.4
J1820-2528	18 h 20 m 57.85 s	-25° 28' 12.58"	7.7°	-4.7°	1.5	1622.2
J1832-1035	18 h 32 m 20.84 s	-10° 35' 11.20"	22°	-0.088°	1.8	1609.1
J1733-3722	17 h 33 m 15.19 s	-37° 22' 32.40"	-7.9°	-1.9°	2.8	1535.0
CAL1457	17 h 44 m 23.57 s	-31° 16' 36.60"	-1.7°	-0.71°	2.2	1518.3
CAL0925	18 h 26 m 36.22 s	-10° 57' 20.10"	22°	1.3°	0.11	1450.1
CAL0824	18 h 18 m 2.96 s	-17° 05' 42.30"	14°	-0.51°	0.73	1439.7
J1744-3116	17 h 44 m 23.58 s	-31° 16' 35.99"	-1.7°	-0.71°	2.3	1425.1
J1824+0119	18 h 24 m 48.14 s	01° 19' 34.17"	31°	6.5°	0.52	1409.1
CAL0588	17 h 55 m 26.31 s	-22° 32' 11.00"	7.6°	1.9°	0.36	1387.8
J1832-2039	18 h 32 m 11.05 s	-20° 39' 48.20"	14°	-4.6°	1.4	1382.4
CAL0608	17 h 58 m 22.97 s	-23° 43' 11.40"	7.2°	0.92°	1.0	1354.8
CAL0799	18 h 15 m 30.46 s	-18° 36' 14.00"	13°	-0.21°	0.15	1315.7
CAL1087	18 h 37 m 51.73 s	-06° 53' 38.30"	27°	0.68°	0.10	1306.9
CAL1096	18 h 38 m 9.08 s	-06° 32' 4.90"	27°	0.45°	0.17	1302.4
CAL0979	18 h 32 m 20.82 s	-10° 35' 10.20"	22°	-0.088°	1.8	1301.3
CAL0551	17 h 51 m 51.27 s	-25° 23' 59.40"	4.4°	1.0°	0.11	1293.3
J1717-3342	17 h 17 m 36.03 s	-33° 42' 8.76"	-6.1°	3.2°	2.5	1265.1
CAL0498	17 h 12 m 38.73 s	-37° 36' 45.60"	-10°	1.7°	0.60	1264.1
CAL1157	18 h 41 m 57.55 s	-07° 52' 5.60"	26°	-0.69°	0.029	1262.2
CAL1374	17 h 31 m 46.89 s	-30° 03' 9.10"	-2.5°	2.0°	0.20	1248.9
CAL0482	17 h 11 m 20.52 s	-37° 44' 15.40"	-10°	1.9°	0.76	1247.2
CAL1451	17 h 43 m 17.88 s	-30° 58' 19.20"	-0.34°	0.38°	0.26	1220.9
J1755-2232	17 h 55 m 26.28 s	-22° 32' 10.59"	7.6°	1.9°	0.37	1220.5
CAL0740	18 h 10 m 39.92 s	-16° 26' 51.20"	14°	1.7°	0.23	1198.2
CAL0852	18 h 20 m 22.99 s	-11° 11' 16.50"	20°	1.9°	0.19	1191.5
CAL1202	18 h 46 m 6.44 s	-06° 51' 28.10"	28°	-1.2°	0.17	1178.3
CAL0559	17 h 52 m 30.97 s	-30° 01' 7.80"	-0.12°	-1.8°	0.31	1172.9
CAL0593	17 h 19 m 34.39 s	-33° 54' 54.40"	-5.7°	3.0°	0.12	1169.9
CAL1091	18 h 37 m 58.04 s	-06° 53' 30.90"	27°	0.66°	0.16	1164.0
CAL1421	17 h 37 m 43.80 s	-31° 31' 16.50"	-2.2°	0.64°	0.15	1158.6
CAL0844	18 h 19 m 15.88 s	-14° 19' 2.00"	17°	0.75°	0.15	1156.5
CAL0992	18 h 33 m 19.53 s	-08° 55' 27.30"	25°	0.78°	0.14	1155.1
CAL0616	17 h 59 m 33.60 s	-23° 23' 24.30"	7.0°	0.52°	0.14	1147.4

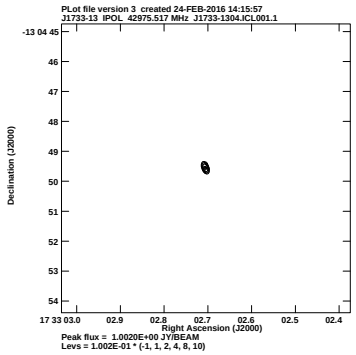
Source	RA	Dec	l	b	Flux (Jy)	Rank
CAL0906	18 h 25 m 11.71 s	-15° 51' 34.20"	17°	-0.73°	0.19	1127.7
CAL0560	17 h 52 m 33.16 s	-29° 56' 45.50"	1.5°	-0.85°	0.24	1123.6
CAL1406	17 h 36 m 15.18 s	-33° 31' 58.80"	-4.1°	-0.16°	0.39	1106.3
CAL1151	18 h 41 m 27.27 s	-03° 48' 45.40"	30°	1.2°	0.17	1101.2
CAL0459	17 h 05 m 57.38 s	-39° 36' 59.70"	-12°	1.5°	0.25	1099.9
CAL0743	18 h 11 m 5.92 s	-18° 28' 26.10"	13°	0.65°	0.11	1093.8
CAL0540	17 h 17 m 38.62 s	-39° 48' 51.30"	-11°	-0.20°	0.42	1084.8
CAL1219	18 h 47 m 45.48 s	-03° 40' 49.70"	30°	-0.24°	0.083	1070.3
CAL1144	18 h 40 m 51.89 s	-05° 48' 44.50"	28°	0.44°	0.097	1066.6
CAL1440	17 h 42 m 3.78 s	-33° 41' 33.90"	-3.3°	-1.1°	0.30	1064.7
CAL1045	18 h 35 m 19.64 s	-11° 15' 59.40"	22°	-1.3°	0.11	1044.2
CAL1288	17 h 24 m 17.33 s	-34° 25' 48.71"	-6.4°	1.3°	0.12	1039.0
CAL0754	18 h 11 m 43.13 s	-14° 16' 21.90"	16°	2.3°	0.11	1025.4
CAL1078	18 h 37 m 14.98 s	-03° 16' 4.50"	29°	1.9°	0.088	1020.7
CAL0487	17 h 11 m 48.37 s	-39° 09' 59.89"	-12°	0.35°	0.23	1015.7
CAL0897	18 h 24 m 36.39 s	-12° 51' 6.80"	20°	0.79°	0.23	1015.5
CAL0875	18 h 21 m 48.36 s	-16° 23' 6.70"	16°	-0.70°	0.18	1007.0
CAL0506	17 h 14 m 5.92 s	-38° 08' 5.00"	-11°	0.55°	0.27	999.90
CAL0670	17 h 20 m 8.52 s	-39° 15' 37.85"	-11°	-0.92°	0.26	994.11
CAL0447	16 h 56 m 46.15 s	-40° 14' 37.00"	-15°	2.1°	0.40	993.19
CAL1128	18 h 39 m 32.36 s	-05° 44' 20.90"	28°	0.70°	0.17	988.62
CAL1214	18 h 47 m 0.34 s	-02° 27' 51.80"	31°	0.29°	0.16	988.27
CAL1263	17 h 22 m 34.77 s	-37° 17' 44.40"	-9.1°	-0.16°	0.23	986.80
CAL1418	17 h 37 m 24.37 s	-28° 04' 31.80"	-0.095°	2.1°	0.31	967.89
CAL0832	18 h 18 m 50.50 s	-13° 48' 53.54"	18°	1.6°	0.12	965.60
CAL1225	18 h 49 m 15.55 s	-02° 59' 27.00"	32°	0.025°	0.033	963.17
CAL1363	17 h 31 m 16.20 s	-33° 14' 44.45"	-4.9°	0.56°	0.027	958.54
CAL1049	18 h 35 m 37.07 s	-06° 26' 42.21"	26°	0.97°	0.16	952.47
CAL0770	18 h 13 m 26.10 s	-13° 36' 53.00"	18°	2.6°	0.065	943.33
CAL1001	18 h 33 m 58.63 s	-08° 42' 51.79"	24°	0.54°	0.071	943.18
CAL0766	18 h 13 m 2.29 s	-14° 32' 41.20"	17°	2.2°	0.061	933.74
CAL1254	17 h 21 m 52.45 s	-39° 00' 47.79"	-11°	-1.3°	0.25	932.70
CAL1040	18 h 35 m 11.33 s	-07° 25' 8.56"	25°	0.59°	0.20	930.08
CAL0857	18 h 20 m 46.11 s	-13° 45' 25.80"	19°	1.1°	0.50	921.82
CAL0727	18 h 09 m 39.02 s	-19° 21' 16.36"	12°	0.41°	0.017	895.98
CAL1423	17 h 37 m 52.85 s	-34° 13' 4.69"	-5.0°	-1.2°	0.084	862.62

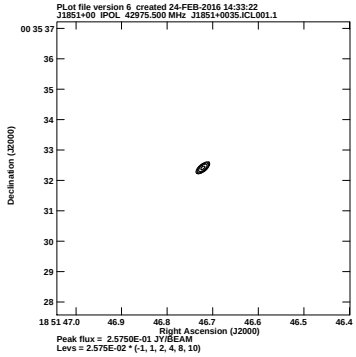
Source	RA	Dec	l	b	Flux (Jy)	Rank
CAL0944	18 h 28 m 10.78 s	-11° 29' 6.80"	21°	0.31°	0.38	819.78
CAL1168	18 h 42 m 57.93 s	-04° 14' 7.60"	29°	0.17°	0.78	559.83
CAL0736	18 h 10 m 28.36 s	-19° 55' 48.01"	12°	0.52°	2.4	328.91
CAL1306	17 h 25 m 25.35 s	-36° 12' 27.69"	-8.1°	-0.12°	0.55	314.59
CAL1334	17 h 29 m 1.10 s	-36° 33' 52.52"	-7.4°	-0.53°	0.20	68.455
CAL0642	18 h 00 m 49.65 s	-23° 21' 51.32"	7.2°	0.25°	0.071	68.406
CAL1178	18 h 43 m 41.25 s	-06° 01' 32.10"	27°	-1.0°	0.078	68.003
CAL0599	17 h 57 m 38.09 s	-28° 20' 7.30"	2.4°	-1.6°	0.14	63.174
CAL1076	18 h 37 m 6.89 s	-04° 24' 17.86"	28°	1.5°	0.11	63.150
CAL0549	17 h 18 m 1.20 s	-37° 26' 33.83"	-9.5°	0.68°	0.18	47.220
CAL1236	17 h 21 m 21.79 s	-39° 15' 16.20"	-11°	-1.1°	0.16	42.646
CAL1046	18 h 35 m 31.39 s	-07° 33' 22.69"	25°	0.58°	0.070	29.969
CAL0744	17 h 21 m 18.51 s	-37° 46' 8.45"	-8.9°	0.32°	0.014	19.887
CAL1058	18 h 36 m 10.55 s	-07° 11' 23.57"	25°	0.27°	0.0072	17.442
CAL1007	18 h 34 m 8.33 s	-08° 51' 34.10"	25°	0.57°	0.0086	14.137
CAL0833	18 h 18 m 52.80 s	-14° 58' 57.22"	18°	1.1°	0.0085	14.123
CAL0973	18 h 31 m 19.16 s	-12° 36' 51.97"	21°	-0.77°	0.011	13.785
CAL0535	17 h 17 m 0.38 s	-37° 41' 22.18"	-9.5°	0.99°	0.12	13.781
CAL1232	18 h 52 m 6.77 s	-02° 00' 27.57"	31°	-1.1°	0.0050	13.560
CAL1223	18 h 48 m 24.67 s	-05° 21' 25.15"	28°	-1.4°	0.0074	13.409
CAL1211	18 h 46 m 36.15 s	-05° 30' 23.27"	28°	-0.97°	0.0090	13.212
CAL1102	18 h 38 m 15.19 s	-06° 47' 51.18"	27°	0.55°	0.011	11.778
CAL1024	18 h 34 m 31.05 s	-08° 56' 15.85"	25°	0.52°	0.0063	11.722
CAL0510	17 h 14 m 16.37 s	-38° 28' 35.38"	-11°	0.72°	0.014	11.350
CAL1108	18 h 38 m 29.81 s	-06° 44' 35.71"	27°	0.47°	0.0067	11.309
CAL0903	18 h 25 m 2.95 s	-13° 09' 45.87"	19°	-0.093°	0.0054	11.219
CAL1131	18 h 39 m 44.90 s	-06° 30' 15.05"	27°	0.086°	0.0056	10.905
CAL1426	17 h 39 m 12.94 s	-28° 46' 58.61"	0.72°	2.1°	0.0073	10.474
CAL1191	18 h 44 m 49.21 s	-04° 01' 27.27"	29°	-0.34°	0.0052	10.431
CAL1119	18 h 39 m 20.25 s	-05° 45' 11.00"	28°	0.75°	0.0067	10.221
CAL1186	18 h 44 m 40.28 s	-07° 19' 59.56"	26°	-1.5°	0.0046	10.198
CAL0556	17 h 52 m 17.17 s	-30° 07' 11.95"	-0.061°	-1.7°	0.0044	9.4150
CAL0617	17 h 59 m 48.31 s	-23° 09' 44.33"	6.9°	0.36°	0.0086	9.1963
CAL0790	18 h 14 m 34.28 s	-16° 26' 32.14"	15°	0.86°	0.0094	8.9492
CAL1350	17 h 29 m 42.95 s	-36° 43' 50.93"	-7.1°	-0.55°	0.0080	8.0817
CAL1073	18 h 36 m 58.74 s	-06° 53' 13.76"	27°	0.87°	0.0035	7.7673

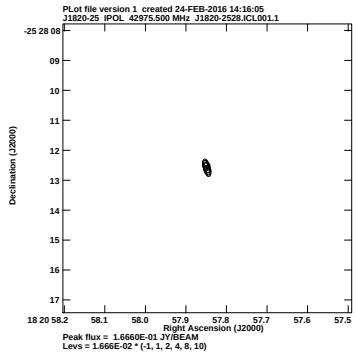
C. Ordered Contour Plots

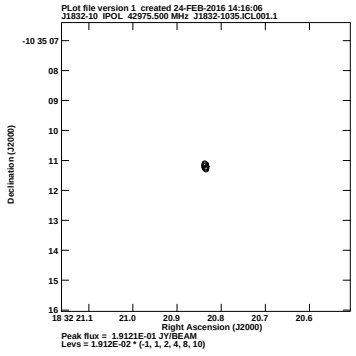
Contour plots for known and unknown Calibrators. Note, fluxes are normalized to 1 Jy in these plots.

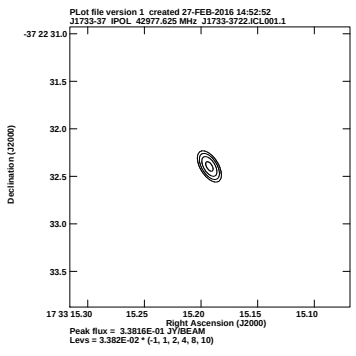
C.1. Known Calibrators

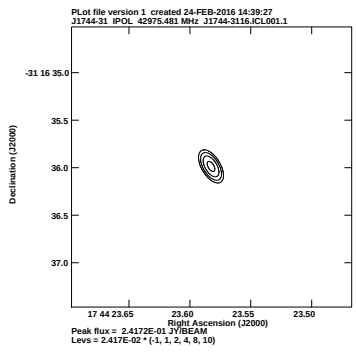


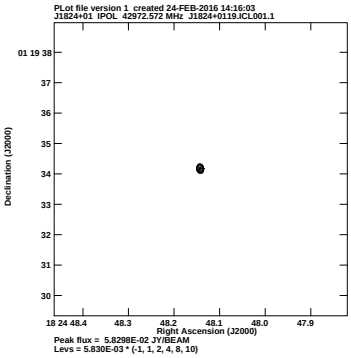


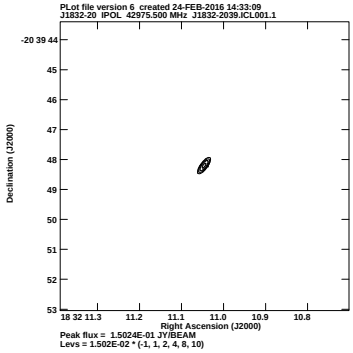


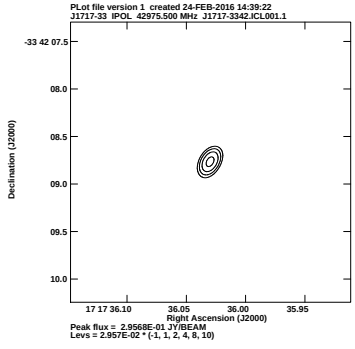


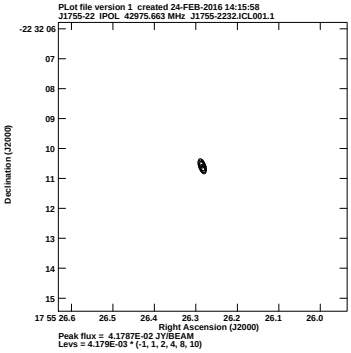


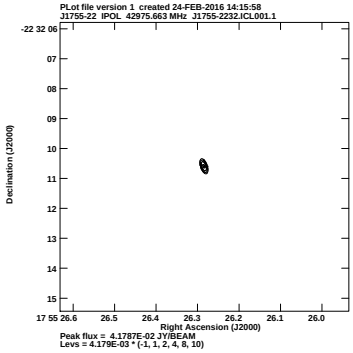



PLot file version 1 created 24-FEB-2016 14:16:03
J1824+01 IPOL 42972.572 MHz J1824+0119.ICL001.1
Declination (J2000)
01 19 38
37
36
35
34
33
32
31
30
18 24 48.4
48.3
48.2
48.1
48.0
47.9
Right Ascension (J2000)
Peak flux = 5.8298E-02 JY/BEAM
Levs = 5.830E-03 * (-1, 1, 2, 4, 8, 10)

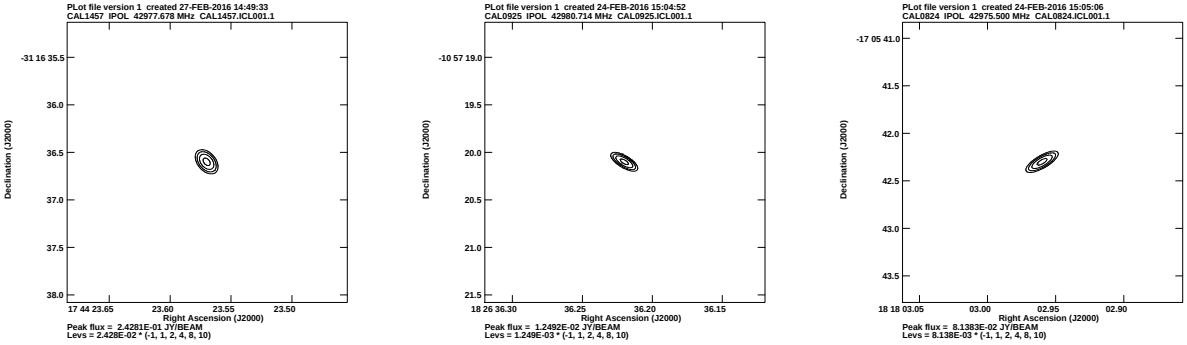

PLot file version 6 created 24-FEB-2016 14:33:09
J1832-20 IPOL 42975.500 MHz J1832-2039.ICL001.1
Declination (J2000)
-20 39 44
45
46
47
48
49
50
51
52
53
18 32 11.3
11.2
11.1
11.0
10.9
10.8
Right Ascension (J2000)
Peak flux = 1.5024E-01 JY/BEAM
Levs = 1.502E-02 * (-1, 1, 2, 4, 8, 10)

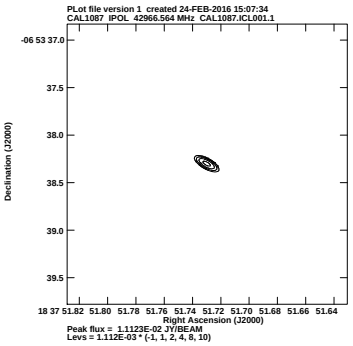

PLot file version 1 created 24-FEB-2016 14:39:22
J1717-33 IPOL 42975.500 MHz J1717-3342.ICL001.1
Declination (J2000)
-33 42 07.5
08.0
08.5
09.0
09.5
10.0
17 17 36.10
36.05
36.00
35.95
Right Ascension (J2000)
Peak flux = 2.9568E-01 JY/BEAM
Levs = 2.957E-02 * (-1, 1, 2, 4, 8, 10)

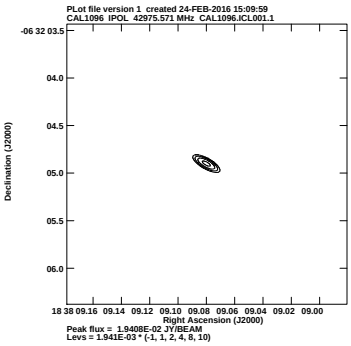

PLot file version 1 created 24-FEB-2016 14:15:58
J1755-22 IPOL 42975.663 MHz J1755-2232.ICL001.1
Declination (J2000)
-22 32 06
07
08
09
10
11
12
13
14
15
17 55 26.6
26.5
26.4
26.3
26.2
26.1
26.0
Right Ascension (J2000)
Peak flux = 4.1787E-02 JY/BEAM
Levs = 4.179E-03 * (-1, 1, 2, 4, 8, 10)

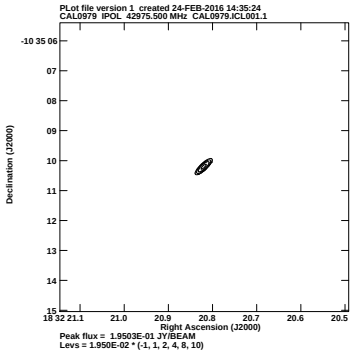

PLot file version 1 created 24-FEB-2016 14:15:58
J1755-22 IPOL 42975.663 MHz J1755-2232.ICL001.1
Declination (J2000)
-22 32 06
07
08
09
10
11
12
13
14
15
17 55 26.6
26.5
26.4
26.3
26.2
26.1
26.0
Right Ascension (J2000)
Peak flux = 4.1787E-02 JY/BEAM
Levs = 4.179E-03 * (-1, 1, 2, 4, 8, 10)

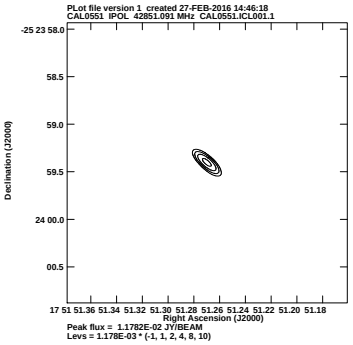
C.2. New Calibrators

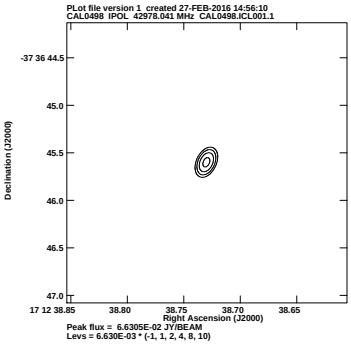


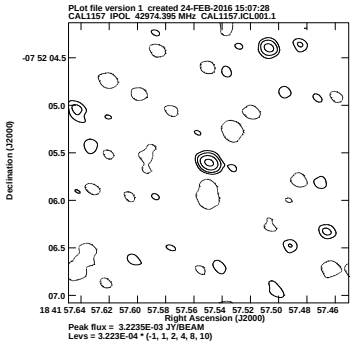

PLot file version 1 created 24-FEB-2016 15:07:34
CAL1087 IPOL 42966.564 MHz CAL1087.ICL001.1
Declination (J2000)
-06 53 37.0
37.5
38.0
38.5
39.0
39.5
18 37 51.82 51.80 51.78 51.76 51.74 51.72 51.70 51.68 51.66 51.64
Right Ascension (J2000)
Peak flux = 1.1123E-02 JY/BEAM
Levs = 1.112E-03 * (-1, 1, 2, 4, 8, 10)

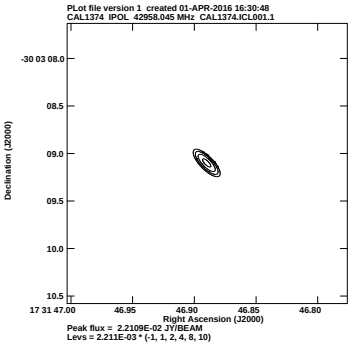

PLot file version 1 created 24-FEB-2016 15:09:59
CAL1096 IPOL 42975.571 MHz CAL1096.ICL001.1
Declination (J2000)
-06 32 03.5
04.0
04.5
05.0
05.5
06.0
18 38 09.16 09.14 09.12 09.10 09.08 09.06 09.04 09.02 09.00
Right Ascension (J2000)
Peak flux = 1.9408E-02 JY/BEAM
Levs = 1.941E-03 * (-1, 1, 2, 4, 8, 10)

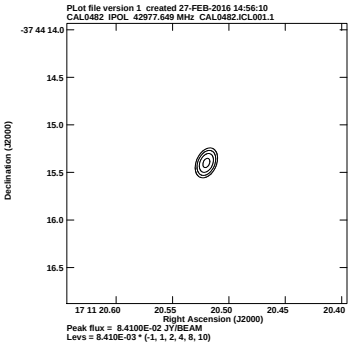

PLot file version 1 created 24-FEB-2016 14:35:24
CAL0979 IPOL 42975.500 MHz CAL0979.ICL001.1
Declination (J2000)
-10 35 06
07
08
09
10
11
12
13
14
15
18 32 21.1 21.0 20.9 20.8 20.7 20.6 20.5
Right Ascension (J2000)
Peak flux = 1.9503E-01 JY/BEAM
Levs = 1.950E-02 * (-1, 1, 2, 4, 8, 10)

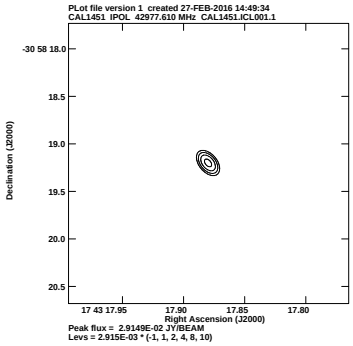

PLot file version 1 created 27-FEB-2016 14:46:18
CAL0551 IPOL 42851.091 MHz CAL0551.ICL001.1
Declination (J2000)
-25 23 58.0
58.5
59.0
59.5
24 00.0
00.5
17 51 51.36 51.34 51.32 51.30 51.28 51.26 51.24 51.22 51.20 51.18
Right Ascension (J2000)
Peak flux = 1.1782E-02 JY/BEAM
Levs = 1.178E-03 * (-1, 1, 2, 4, 8, 10)

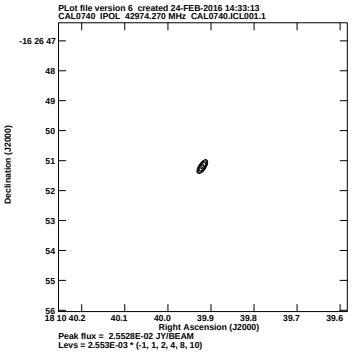

PLot file version 1 created 27-FEB-2016 14:56:10
CAL0498 IPOL 42978.041 MHz CAL0498.ICL001.1
Declination (J2000)
-37 36 44.5
45.0
45.5
46.0
46.5
47.0
17 12 38.85 38.80 38.75 38.70 38.65
Right Ascension (J2000)
Peak flux = 6.6305E-02 JY/BEAM
Levs = 6.630E-03 * (-1, 1, 2, 4, 8, 10)

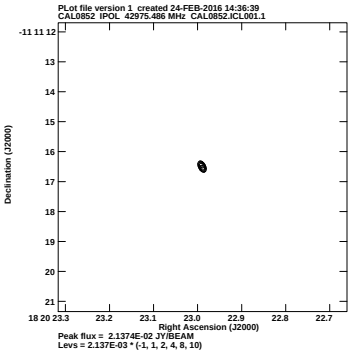

PLot file version 1 created 24-FEB-2016 15:07:28
CAL1157 IPOL 42974.395 MHz CAL1157.ICL001.1
Declination (J2000)
-07 52 04.5
05.0
05.5
06.0
06.5
07.0
18 41 57.64 57.62 57.60 57.58 57.56 57.54 57.52 57.50 57.48 57.46
Right Ascension (J2000)
Peak flux = 3.2235E-03 JY/BEAM
Levs = 3.223E-04 * (-1, 1, 2, 4, 8, 10)

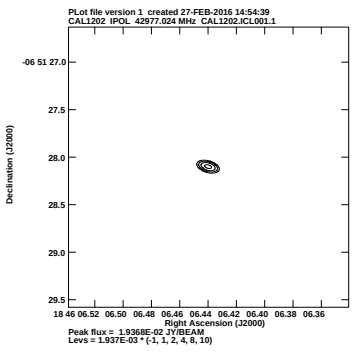

PLot file version 1 created 01-APR-2016 16:30:48
CAL1374 IPOL 42958.045 MHz CAL1374.ICL001.1
Declination (J2000)
-30 03 08.0
08.5
09.0
09.5
10.0
10.5
17 31 47.00 46.95 46.90 46.85 46.80
Right Ascension (J2000)
Peak flux = 2.2109E-02 JY/BEAM
Levs = 2.211E-03 * (-1, 1, 2, 4, 8, 10)

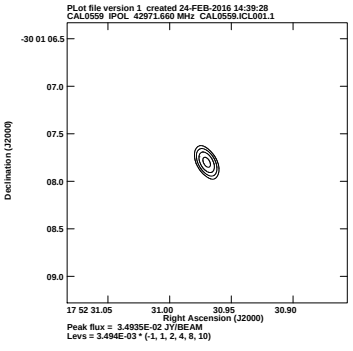

PLot file version 1 created 27-FEB-2016 14:56:10
CAL0482 IPOL 42977.649 MHz CAL0482.ICL001.1
Declination (J2000)
-37 44 14.0
14.5
15.0
15.5
16.0
16.5
17 11 20.60 20.55 20.50 20.45 20.40
Right Ascension (J2000)
Peak flux = 8.4100E-02 JY/BEAM
Levs = 8.410E-03 * (-1, 1, 2, 4, 8, 10)

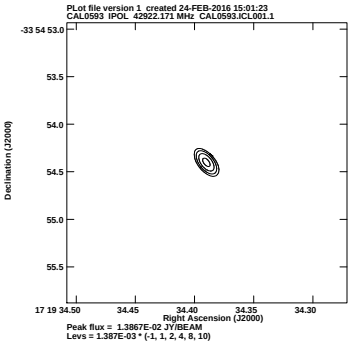

PLot file version 1 created 27-FEB-2016 14:49:34
CAL1451 IPOL 42977.610 MHz CAL1451.ICL001.1
Declination (J2000)
-30 58 18.0
18.5
19.0
19.5
20.0
20.5
17 43 17.95 17.90 17.85 17.80
Right Ascension (J2000)
Peak flux = 2.9149E-02 JY/BEAM
Levs = 2.915E-03 * (-1, 1, 2, 4, 8, 10)

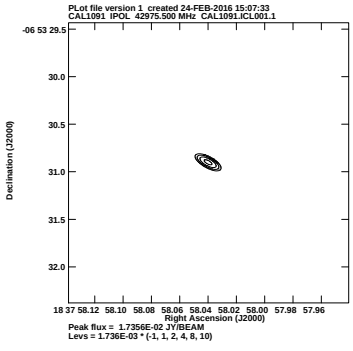

PLot file version 6 created 24-FEB-2016 14:33:13
CAL0740 IPOL 42974.270 MHz CAL0740.ICL001.1
Declination (J2000)
-16 26 47
48
49
50
51
52
53
54
55
56
18 10 40.2
40.1
40.0
39.9
39.8
39.7
39.6
Right Ascension (J2000)
Peak flux = 2.5528E-02 JY/BEAM
Levs = 2.553E-03 * (-1, 1, 2, 4, 8, 10)

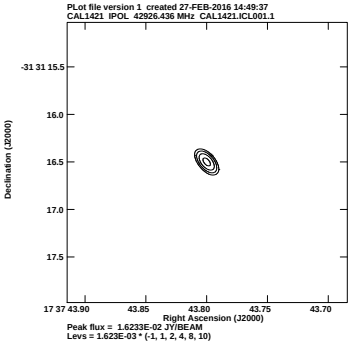

PLot file version 1 created 24-FEB-2016 14:36:39
CAL0852 IPOL 42975.486 MHz CAL0852.ICL001.1
Declination (J2000)
-11 11 12
13
14
15
16
17
18
19
20
21
18 20 23.3
23.2
23.1
23.0
22.9
22.8
22.7
Right Ascension (J2000)
Peak flux = 2.1374E-02 JY/BEAM
Levs = 2.137E-03 * (-1, 1, 2, 4, 8, 10)

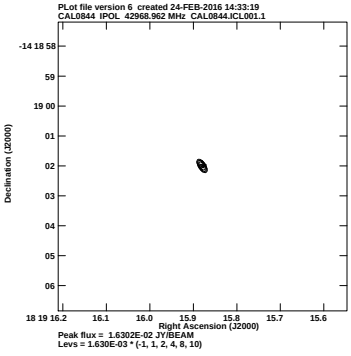

PLot file version 1 created 27-FEB-2016 14:54:39
CAL1202 IPOL 42977.024 MHz CAL1202.ICL001.1
Declination (J2000)
-06 51 27.0
27.5
28.0
28.5
29.0
29.5
18 46 06.52
06.50
06.48
06.46
06.44
06.42
06.40
06.38
06.36
Right Ascension (J2000)
Peak flux = 1.9368E-02 JY/BEAM
Levs = 1.937E-03 * (-1, 1, 2, 4, 8, 10)

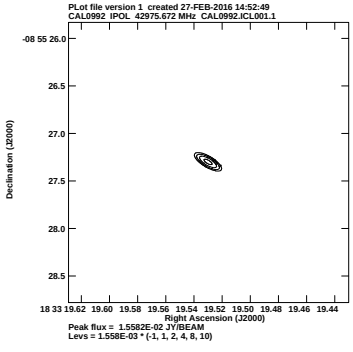

PLot file version 1 created 24-FEB-2016 14:39:28
CAL0559 IPOL 42971.660 MHz CAL0559.ICL001.1
Declination (J2000)
-30 01 06.5
07.0
07.5
08.0
08.5
09.0
17 52 31.05
31.00
30.95
30.90
Right Ascension (J2000)
Peak flux = 3.4935E-02 JY/BEAM
Levs = 3.494E-03 * (-1, 1, 2, 4, 8, 10)

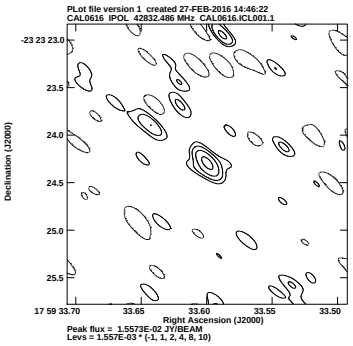

PLot file version 1 created 24-FEB-2016 15:01:23
CAL0593 IPOL 42922.171 MHz CAL0593.ICL001.1
Declination (J2000)
-33 54 53.0
53.5
54.0
54.5
55.0
55.5
17 19 34.50
34.45
34.40
34.35
34.30
Right Ascension (J2000)
Peak flux = 1.3867E-02 JY/BEAM
Levs = 1.387E-03 * (-1, 1, 2, 4, 8, 10)

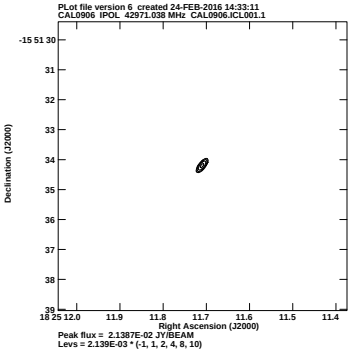

PLot file version 1 created 24-FEB-2016 15:07:33
CAL1091 IPOL 42975.500 MHz CAL1091.ICL001.1
Declination (J2000)
-06 53 29.5
30.0
30.5
31.0
31.5
32.0
18 37 58.12
58.10
58.08
58.06
58.04
58.02
58.00
57.98
57.96
Right Ascension (J2000)
Peak flux = 1.7356E-02 JY/BEAM
Levs = 1.736E-03 * (-1, 1, 2, 4, 8, 10)

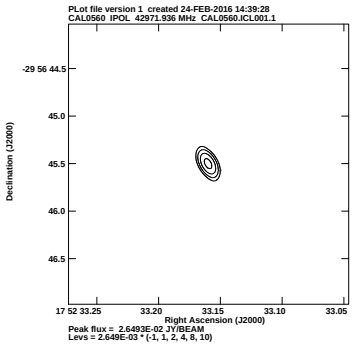

PLot file version 1 created 27-FEB-2016 14:49:37
CAL1421 IPOL 42926.436 MHz CAL1421.ICL001.1
Declination (J2000)
-31 31 15.5
16.0
16.5
17.0
17.5
17 37 43.90
43.85
43.80
43.75
43.70
Right Ascension (J2000)
Peak flux = 1.6233E-02 JY/BEAM
Levs = 1.623E-03 * (-1, 1, 2, 4, 8, 10)

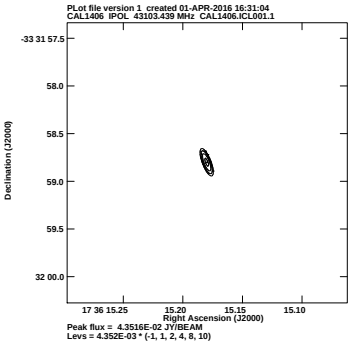

PLot file version 6 created 24-FEB-2016 14:33:19
CAL0844 IPOL 42968.962 MHz CAL0844.ICL001.1
Declination (J2000)
-14 18 58
59
19 00
01
02
03
04
05
06
18 19 16.2
16.1
16.0
15.9
15.8
15.7
15.6
Right Ascension (J2000)
Peak flux = 1.6302E-02 JY/BEAM
Levs = 1.630E-03 * (-1, 1, 2, 4, 8, 10)

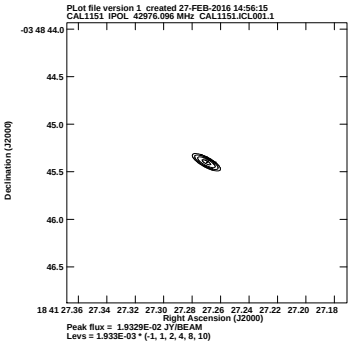

PLot file version 1 created 27-FEB-2016 14:52:49
CAL0992 IPOL 42975.672 MHz CAL0992.ICL001.1
Declination (J2000)
-08 55 26.0
26.5
27.0
27.5
28.0
28.5
18 33 19.62
19.60
19.58
19.56
19.54
19.52
19.50
19.48
19.46
19.44
Right Ascension (J2000)
Peak flux = 1.5582E-02 JY/BEAM
Levs = 1.558E-03 * (-1, 1, 2, 4, 8, 10)

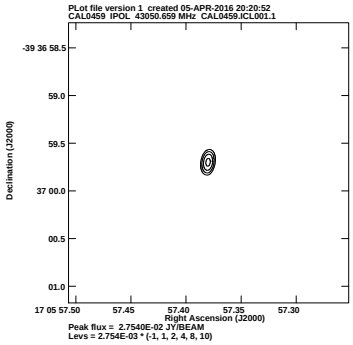

PLot file version 1 created 27-FEB-2016 14:46:22
CAL0616 IPOL 42832.486 MHz CAL0616.ICL001.1
Declination (J2000)
-23 23 23.0
23.5
24.0
24.5
25.0
25.5
17 59 33.70
33.65
33.60
33.55
33.50
Right Ascension (J2000)
Peak flux = 1.5573E-02 JY/BEAM
Levs = 1.557E-03 * (-1, 1, 2, 4, 8, 10)

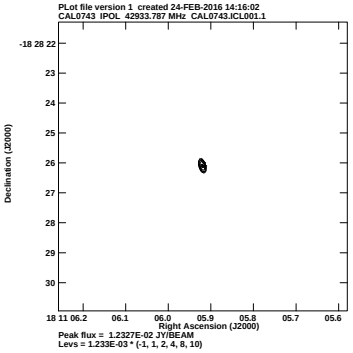

PLot file version 6 created 24-FEB-2016 14:33:11
CAL0906 IPOL 42971.038 MHz CAL0906.ICL001.1
Declination (J2000)
-15 51 30
31
32
33
34
35
36
37
38
39
18 25 12.0
11.9
11.8
11.7
11.6
11.5
11.4
Right Ascension (J2000)
Peak flux = 2.1387E-02 JY/BEAM
Levs = 2.139E-03 * (-1, 1, 2, 4, 8, 10)

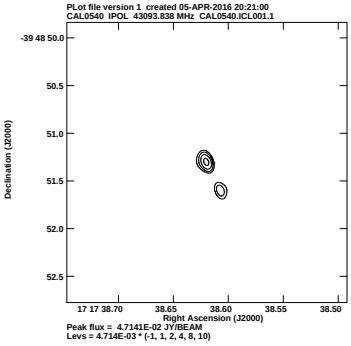

PLot file version 1 created 24-FEB-2016 14:39:28
CAL0560 IPOL 42971.936 MHz CAL0560.ICL001.1
Declination (J2000)
-29 56 44.5
45.0
45.5
46.0
46.5
17 52 33.25
33.20
33.15
33.10
33.05
Right Ascension (J2000)
Peak flux = 2.6493E-02 JY/BEAM
Levs = 2.649E-03 * (-1, 1, 2, 4, 8, 10)

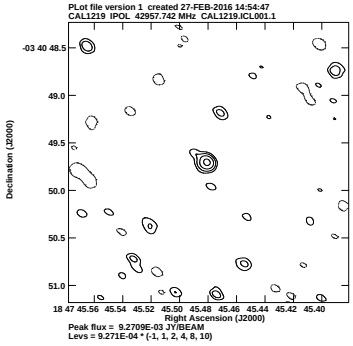

PLot file version 1 created 01-APR-2016 16:31:04
CAL1406 IPOL 43103.439 MHz CAL1406.ICL001.1
Declination (J2000)
-33 31 57.5
58.0
58.5
59.0
59.5
32 00.0
17 36 15.25
15.20
15.15
15.10
Right Ascension (J2000)
Peak flux = 4.3516E-02 JY/BEAM
Levs = 4.352E-03 * (-1, 1, 2, 4, 8, 10)

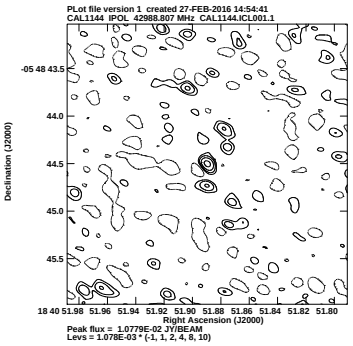

PLot file version 1 created 27-FEB-2016 14:56:15
CAL1151 IPOL 42976.096 MHz CAL1151.ICL001.1
Declination (J2000)
-03 48 44.0
44.5
45.0
45.5
46.0
46.5
18 41 27.36
27.34
27.32
27.30
27.28
27.26
27.24
27.22
27.20
27.18
Right Ascension (J2000)
Peak flux = 1.9329E-02 JY/BEAM
Levs = 1.933E-03 * (-1, 1, 2, 4, 8, 10)

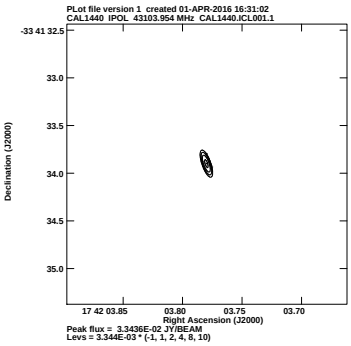

PLot file version 1 created 05-APR-2016 20:20:52
CAL0459 IPOL 43050.659 MHz CAL0459.ICL001.1
Declination (J2000)
-39 36 58.5
59.0
59.5
37 00.0
00.5
01.0
17 05 57.50
57.45
57.40
57.35
57.30
Right Ascension (J2000)
Peak flux = 2.7540E-02 JY/BEAM
Levs = 2.754E-03 * (-1, 1, 2, 4, 8, 10)

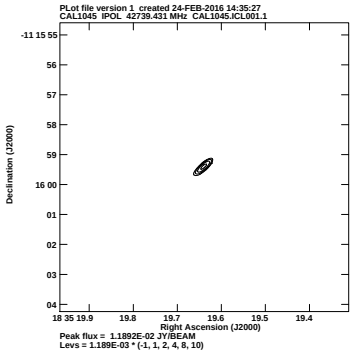

PLot file version 1 created 24-FEB-2016 14:16:02
CAL0743 IPOL 42933.787 MHz CAL0743.ICL001.1
Declination (J2000)
-18 28 22
23
24
25
26
27
28
29
30
18 11 06.2
06.1
06.0
05.9
05.8
05.7
05.6
Right Ascension (J2000)
Peak flux = 1.2327E-02 JY/BEAM
Levs = 1.233E-03 * (-1, 1, 2, 4, 8, 10)

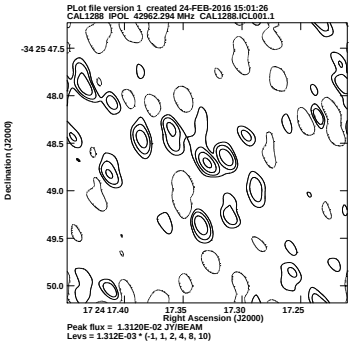

PLot file version 1 created 05-APR-2016 20:21:00
CAL0540 IPOL 43093.838 MHz CAL0540.ICL001.1
Declination (J2000)
-39 48 50.0
50.5
51.0
51.5
52.0
52.5
17 17 38.70
38.65
38.60
38.55
38.50
Right Ascension (J2000)
Peak flux = 4.7141E-02 JY/BEAM
Levs = 4.714E-03 * (-1, 1, 2, 4, 8, 10)

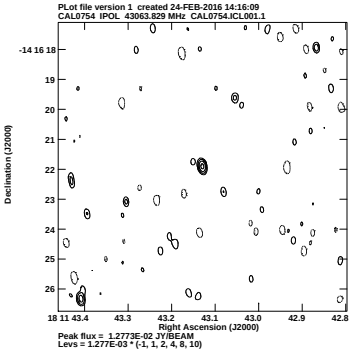

PLot file version 1 created 27-FEB-2016 14:54:47
CAL1219 IPOL 42957.742 MHz CAL1219.ICL001.1
Declination (J2000)
-03 40 48.5
49.0
49.5
50.0
50.5
51.0
18 47 45.56
45.54
45.52
45.50
45.48
45.46
45.44
45.42
45.40
Right Ascension (J2000)
Peak flux = 9.2709E-03 JY/BEAM
Levs = 9.271E-04 * (-1, 1, 2, 4, 8, 10)

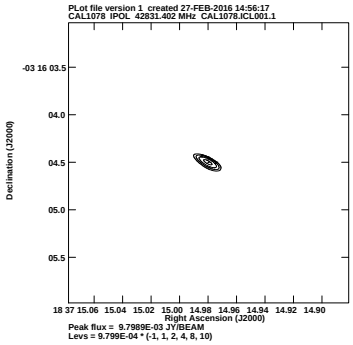

PLot file version 1 created 27-FEB-2016 14:54:41
CAL1144 IPOL 42988.807 MHz CAL1144.ICL001.1
Declination (J2000)
-05 48 43.5
44.0
44.5
45.0
45.5
18 40 51.98 51.96 51.94 51.92 51.90 51.88 51.86 51.84 51.82 51.80
Right Ascension (J2000)
Peak flux = 1.0779E-02 JY/BEAM
Levs = 1.078E-03 * (-1, 1, 2, 4, 8, 10)

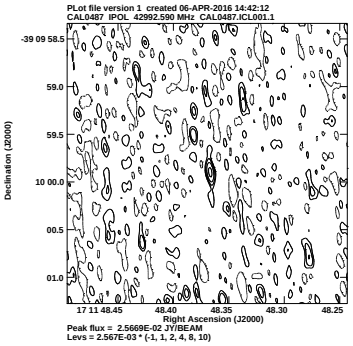

PLot file version 1 created 01-APR-2016 16:31:02
CAL1440 IPOL 43103.954 MHz CAL1440.ICL001.1
Declination (J2000)
-33 41 32.5
33.0
33.5
34.0
34.5
35.0
17 42 03.85 03.80 03.75 03.70
Right Ascension (J2000)
Peak flux = 3.3436E-02 JY/BEAM
Levs = 3.344E-03 * (-1, 1, 2, 4, 8, 10)

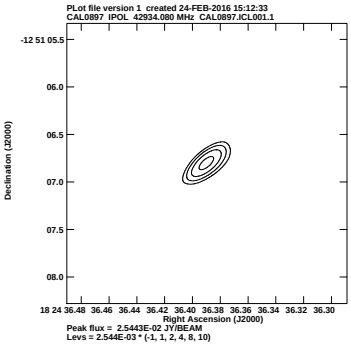

PLot file version 1 created 24-FEB-2016 14:35:27
CAL1045 IPOL 42739.431 MHz CAL1045.ICL001.1
Declination (J2000)
-11 15 55
56
57
58
59
16 00
01
02
03
04
18 35 19.9 19.8 19.7 19.6 19.5 19.4
Right Ascension (J2000)
Peak flux = 1.1892E-02 JY/BEAM
Levs = 1.189E-03 * (-1, 1, 2, 4, 8, 10)

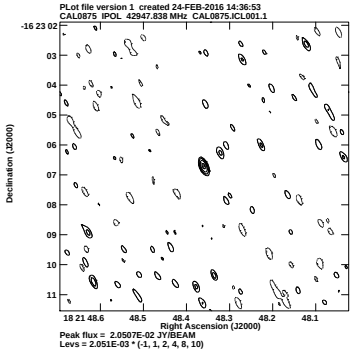

PLot file version 1 created 24-FEB-2016 15:01:26
CAL1288 IPOL 42962.294 MHz CAL1288.ICL001.1
Declination (J2000)
-34 25 47.5
48.0
48.5
49.0
49.5
50.0
17 24 17.40 17.35 17.30 17.25
Right Ascension (J2000)
Peak flux = 1.3120E-02 JY/BEAM
Levs = 1.312E-03 * (-1, 1, 2, 4, 8, 10)

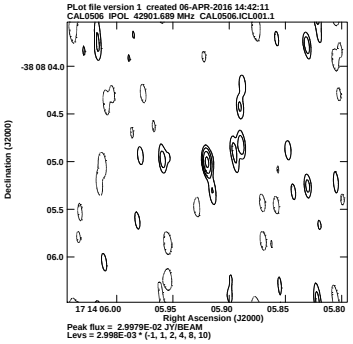

PLot file version 1 created 24-FEB-2016 14:16:09
CAL0754 IPOL 43063.829 MHz CAL0754.ICL001.1
Declination (J2000)
-14 16 18
19
20
21
22
23
24
25
26
18 11 43.4 43.3 43.2 43.1 43.0 42.9 42.8
Right Ascension (J2000)
Peak flux = 1.2773E-02 JY/BEAM
Levs = 1.277E-03 * (-1, 1, 2, 4, 8, 10)

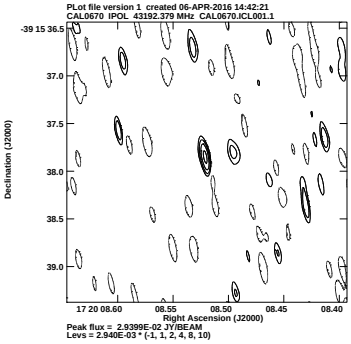

PLot file version 1 created 27-FEB-2016 14:56:17
CAL1078 IPOL 42831.402 MHz CAL1078.ICL001.1
Declination (J2000)
-03 16 03.5
04.0
04.5
05.0
05.5
18 37 15.06 15.04 15.02 15.00 14.98 14.96 14.94 14.92 14.90
Right Ascension (J2000)
Peak flux = 9.7989E-03 JY/BEAM
Levs = 9.799E-04 * (-1, 1, 2, 4, 8, 10)

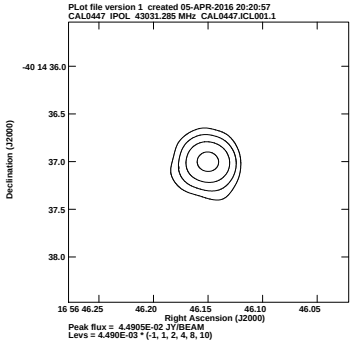

PLot file version 1 created 06-APR-2016 14:42:12
CAL0487 IPOL 42992.590 MHz CAL0487.ICL001.1
Declination (J2000)
-39 09 58.5
59.0
59.5
10 00.0
00.5
01.0
17 11 48.45 48.40 48.35 48.30 48.25
Right Ascension (J2000)
Peak flux = 2.5669E-02 JY/BEAM
Levs = 2.567E-03 * (-1, 1, 2, 4, 8, 10)

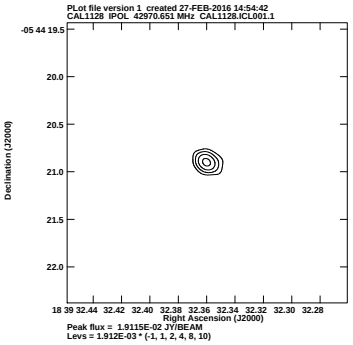

PLot file version 1 created 24-FEB-2016 15:12:33
CAL0897 IPOL 42934.080 MHz CAL0897.ICL001.1
Declination (J2000)
-12 51 05.5
06.0
06.5
07.0
07.5
08.0
18 24 36.48 36.46 36.44 36.42 36.40 36.38 36.36 36.34 36.32 36.30
Right Ascension (J2000)
Peak flux = 2.5443E-02 JY/BEAM
Levs = 2.544E-03 * (-1, 1, 2, 4, 8, 10)

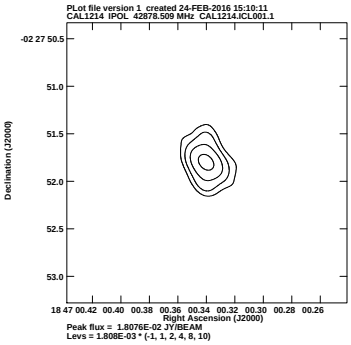

PLot file version 1 created 24-FEB-2016 14:36:53
CAL0875 IPOL 42947.838 MHz CAL0875.ICL001.1
Declination (J2000)
-16 23 02
03
04
05
06
07
08
09
10
11
18 21 48.6 48.5 48.4 48.3 48.2 48.1
Right Ascension (J2000)
Peak flux = 2.0507E-02 JY/BEAM
Levs = 2.051E-03 * (-1, 1, 2, 4, 8, 10)

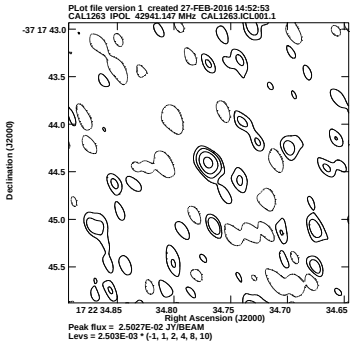

PLot file version 1 created 06-APR-2016 14:42:11
CAL0506 IPOL 42901.689 MHz CAL0506.ICL001.1
Declination (J2000)
-38 08 04.0
04.5
05.0
05.5
06.0
17 14 06.00
05.95
05.90
05.85
05.80
Right Ascension (J2000)
Peak flux = 2.9979E-02 JY/BEAM
Levs = 2.998E-03 * (-1, 1, 2, 4, 8, 10)

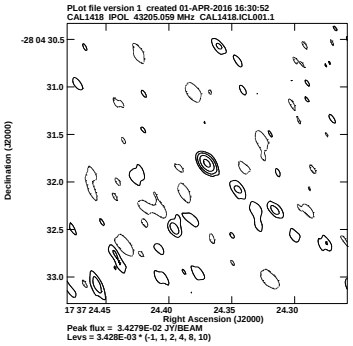

PLot file version 1 created 06-APR-2016 14:42:21
CAL0670 IPOL 43192.379 MHz CAL0670.ICL001.1
Declination (J2000)
-39 15 36.5
37.0
37.5
38.0
38.5
39.0
17 20 08.60
08.55
08.50
08.45
08.40
Right Ascension (J2000)
Peak flux = 2.9399E-02 JY/BEAM
Levs = 2.940E-03 * (-1, 1, 2, 4, 8, 10)

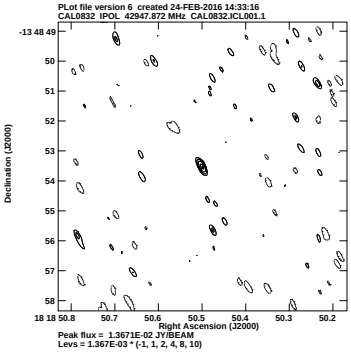

PLot file version 1 created 05-APR-2016 20:20:57
CAL0447 IPOL 43031.285 MHz CAL0447.ICL001.1
Declination (J2000)
-40 14 36.0
36.5
37.0
37.5
38.0
16 56 46.25
46.20
46.15
46.10
46.05
Right Ascension (J2000)
Peak flux = 4.4905E-02 JY/BEAM
Levs = 4.490E-03 * (-1, 1, 2, 4, 8, 10)

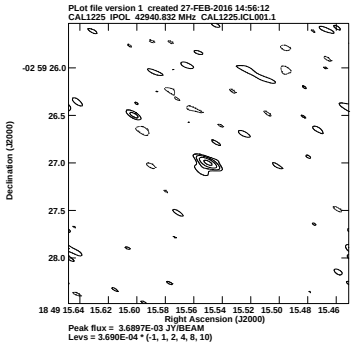

PLot file version 1 created 27-FEB-2016 14:54:42
CAL1128 IPOL 42970.651 MHz CAL1128.ICL001.1
Declination (J2000)
-05 44 19.5
20.0
20.5
21.0
21.5
22.0
18 39 32.44
32.42
32.40
32.38
32.36
32.34
32.32
32.30
32.28
Right Ascension (J2000)
Peak flux = 1.9115E-02 JY/BEAM
Levs = 1.912E-03 * (-1, 1, 2, 4, 8, 10)

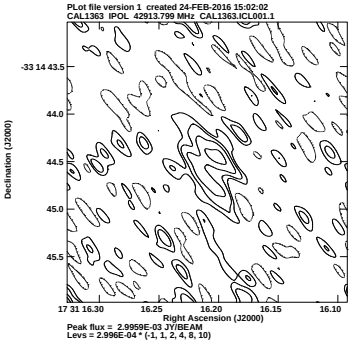

PLot file version 1 created 24-FEB-2016 15:10:11
CAL1214 IPOL 42878.509 MHz CAL1214.ICL001.1
Declination (J2000)
-02 27 50.5
51.0
51.5
52.0
52.5
53.0
18 47 00.42
00.40
00.38
00.36
00.34
00.32
00.30
00.28
00.26
Right Ascension (J2000)
Peak flux = 1.8076E-02 JY/BEAM
Levs = 1.808E-03 * (-1, 1, 2, 4, 8, 10)

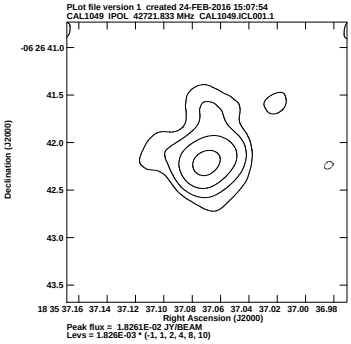

PLot file version 1 created 27-FEB-2016 14:52:53
CAL1263 IPOL 42941.147 MHz CAL1263.ICL001.1
Declination (J2000)
-37 17 43.0
43.5
44.0
44.5
45.0
45.5
17 22 34.85
34.80
34.75
34.70
34.65
Right Ascension (J2000)
Peak flux = 2.5027E-02 JY/BEAM
Levs = 2.503E-03 * (-1, 1, 2, 4, 8, 10)

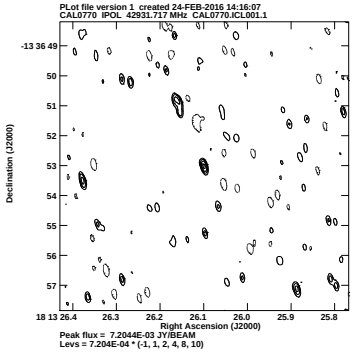

PLot file version 1 created 01-APR-2016 16:30:52
CAL1418 IPOL 43205.059 MHz CAL1418.ICL001.1
Declination (J2000)
-28 04 30.5
31.0
31.5
32.0
32.5
33.0
17 37 24.45
24.40
24.35
24.30
Right Ascension (J2000)
Peak flux = 3.4279E-02 JY/BEAM
Levs = 3.428E-03 * (-1, 1, 2, 4, 8, 10)

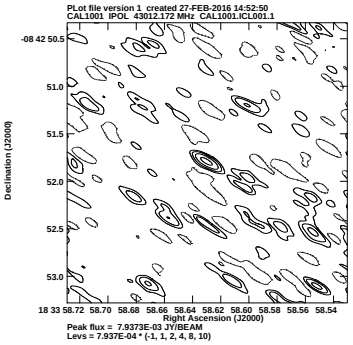

PLot file version 6 created 24-FEB-2016 14:33:16
CAL0832 IPOL 42947.872 MHz CAL0832.ICL001.1
Declination (J2000)
-13 48 49
50
51
52
53
54
55
56
57
58
18 18 50.8
50.7
50.6
50.5
50.4
50.3
50.2
Right Ascension (J2000)
Peak flux = 1.3671E-02 JY/BEAM
Levs = 1.367E-03 * (-1, 1, 2, 4, 8, 10)

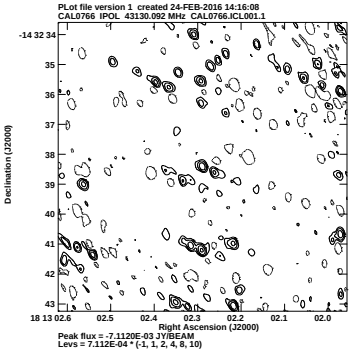

PLot file version 1 created 27-FEB-2016 14:56:12
CAL1225 IPOL 42940.832 MHz CAL1225.ICL001.1
Declination (J2000)
-02 59 26.0
26.5
27.0
27.5
28.0
18 49 15.64
15.62
15.60
15.58
15.56
15.54
15.52
15.50
15.48
15.46
Right Ascension (J2000)
Peak flux = 3.6897E-03 JY/BEAM
Levs = 3.690E-04 * (-1, 1, 2, 4, 8, 10)

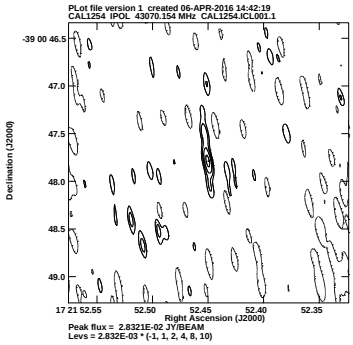

PLot file version 1 created 24-FEB-2016 15:02:02
CAL1363 IPOL 42913.799 MHz CAL1363.ICL001.1
Declination (J2000)
-33 14 43.5
44.0
44.5
45.0
45.5
17 31 16.30
16.25
16.20
16.15
16.10
Right Ascension (J2000)
Peak flux = 2.9959E-03 JY/BEAM
Levs = 2.996E-04 * (-1, 1, 2, 4, 8, 10)

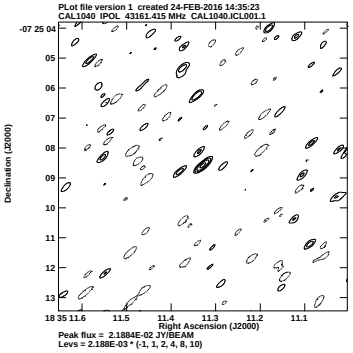

PLot file version 1 created 24-FEB-2016 15:07:54
CAL1049 IPOL 42721.833 MHz CAL1049.ICL001.1
Declination (J2000)
-06 26 41.0
41.5
42.0
42.5
43.0
43.5
18 35 37.16
37.14
37.12
37.10
37.08
37.06
37.04
37.02
37.00
36.98
Right Ascension (J2000)
Peak flux = 1.8261E-02 JY/BEAM
Levs = 1.826E-03 * (-1, 1, 2, 4, 8, 10)

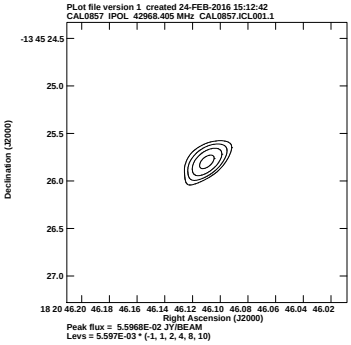

PLot file version 1 created 24-FEB-2016 14:16:07
CAL0770 IPOL 42931.717 MHz CAL0770.ICL001.1
Declination (J2000)
-13 36 49
50
51
52
53
54
55
56
57
18 13 26.4
26.3
26.2
26.1
26.0
25.9
25.8
Right Ascension (J2000)
Peak flux = 7.2044E-03 JY/BEAM
Levs = 7.204E-04 * (-1, 1, 2, 4, 8, 10)

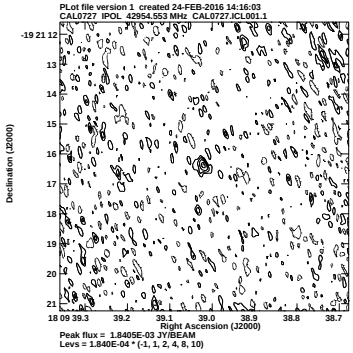

PLot file version 1 created 27-FEB-2016 14:52:50
CAL1001 IPOL 43012.172 MHz CAL1001.ICL001.1
Declination (J2000)
-08 42 50.5
51.0
51.5
52.0
52.5
53.0
18 33 58.72
58.70
58.68
58.66
58.64
58.62
58.60
58.58
58.56
58.54
Right Ascension (J2000)
Peak flux = 7.9373E-03 JY/BEAM
Levs = 7.937E-04 * (-1, 1, 2, 4, 8, 10)

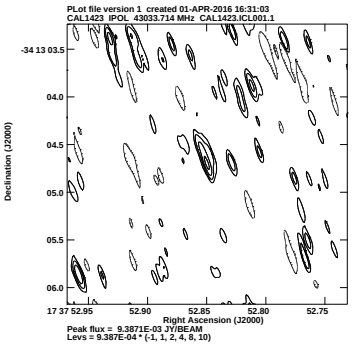

PLot file version 1 created 24-FEB-2016 14:16:08
CAL0766 IPOL 43130.092 MHz CAL0766.ICL001.1
Declination (J2000)
-14 32 34
35
36
37
38
39
40
41
42
43
18 13 02.6
02.5
02.4
02.3
02.2
02.1
02.0
Right Ascension (J2000)
Peak flux = -7.1120E-03 JY/BEAM
Levs = 7.112E-04 * (-1, 1, 2, 4, 8, 10)

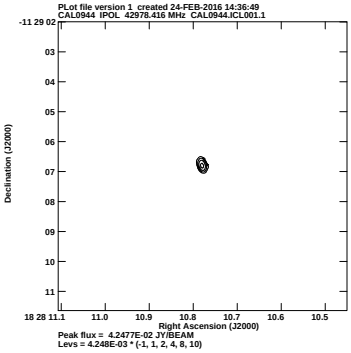

PLot file version 1 created 06-APR-2016 14:42:19
CAL1254 IPOL 43070.154 MHz CAL1254.ICL001.1
Declination (J2000)
-39 00 46.5
47.0
47.5
48.0
48.5
49.0
17 21 52.55
52.50
52.45
52.40
52.35
Right Ascension (J2000)
Peak flux = 2.8321E-02 JY/BEAM
Levs = 2.832E-03 * (-1, 1, 2, 4, 8, 10)

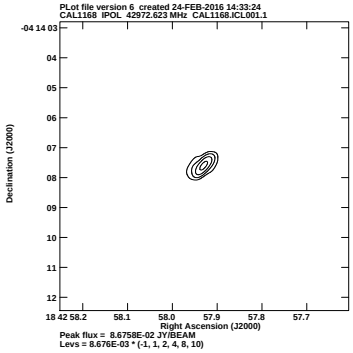

PLot file version 1 created 24-FEB-2016 14:35:23
CAL1040 IPOL 43161.415 MHz CAL1040.ICL001.1
Declination (J2000)
-07 25 04
05
06
07
08
09
10
11
12
13
18 35 11.6
11.5
11.4
11.3
11.2
11.1
Right Ascension (J2000)
Peak flux = 2.1884E-02 JY/BEAM
Levs = 2.188E-03 * (-1, 1, 2, 4, 8, 10)

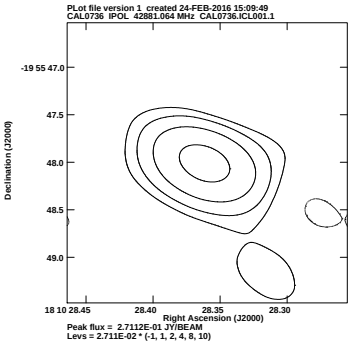

PLot file version 1 created 24-FEB-2016 15:12:42
CAL0857 IPOL 42968.405 MHz CAL0857.ICL001.1
Declination (J2000)
-13 45 24.5
25.0
25.5
26.0
26.5
27.0
18 20 46.20
46.18
46.16
46.14
46.12
46.10
46.08
46.06
46.04
46.02
Right Ascension (J2000)
Peak flux = 5.5968E-02 JY/BEAM
Levs = 5.597E-03 * (-1, 1, 2, 4, 8, 10)

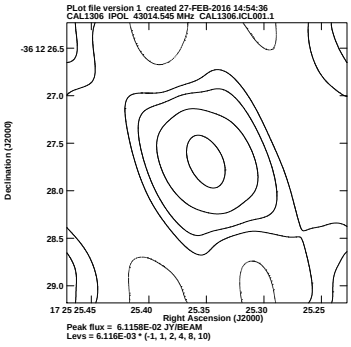

PLot file version 1 created 24-FEB-2016 14:16:03
CAL0727 IPOL 42954.553 MHz CAL0727.ICL001.1
Declination (J2000)
-19 21 12
13
14
15
16
17
18
19
20
21
18 09 39.3
39.2
39.1
39.0
38.9
38.8
38.7
Right Ascension (J2000)
Peak flux = 1.8405E-03 JY/BEAM
Levs = 1.840E-04 * (-1, 1, 2, 4, 8, 10)

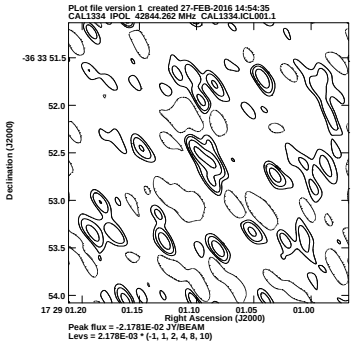

PLot file version 1 created 01-APR-2016 16:31:03
CAL1423 IPOL 43033.714 MHz CAL1423.ICL001.1
Declination (J2000)
-34 13 03.5
04.0
04.5
05.0
05.5
06.0
17 37 52.95
52.90
52.85
52.80
52.75
Right Ascension (J2000)
Peak flux = 9.3871E-03 JY/BEAM
Levs = 9.387E-04 * (-1, 1, 2, 4, 8, 10)

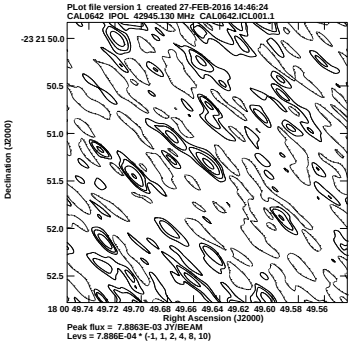

PLot file version 1 created 24-FEB-2016 14:36:49
CAL0944 IPOL 42978.416 MHz CAL0944.ICL001.1
Declination (J2000)
-11 29 02
03
04
05
06
07
08
09
10
11
18 28 11.1
11.0
10.9
10.8
10.7
10.6
10.5
Right Ascension (J2000)
Peak flux = 4.2477E-02 JY/BEAM
Levs = 4.248E-03 * (-1, 1, 2, 4, 8, 10)

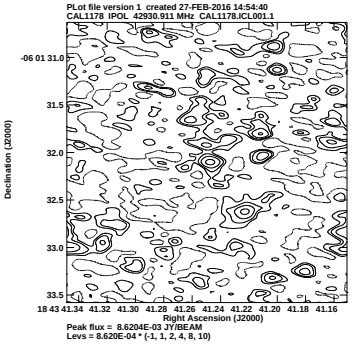

PLot file version 6 created 24-FEB-2016 14:33:24
CAL1168 IPOL 42972.623 MHz CAL1168.ICL001.1
Declination (J2000)
-04 14 03
04
05
06
07
08
09
10
11
12
18 42 58.2
58.1
58.0
57.9
57.8
57.7
Right Ascension (J2000)
Peak flux = 8.6758E-02 JY/BEAM
Levs = 8.676E-03 * (-1, 1, 2, 4, 8, 10)

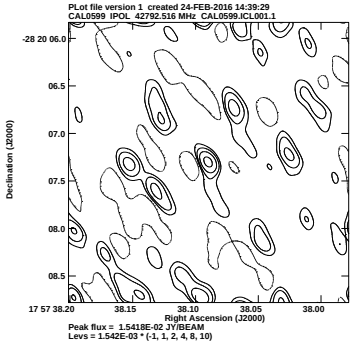

PLot file version 1 created 24-FEB-2016 15:09:49
CAL0736 IPOL 42881.064 MHz CAL0736.ICL001.1
Declination (J2000)
-19 55 47.0
47.5
48.0
48.5
49.0
18 10 28.45
28.40
28.35
28.30
Right Ascension (J2000)
Peak flux = 2.7112E-01 JY/BEAM
Levs = 2.711E-02 * (-1, 1, 2, 4, 8, 10)

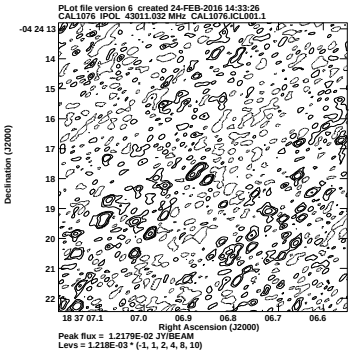

PLot file version 1 created 27-FEB-2016 14:54:36
CAL1306 IPOL 43014.545 MHz CAL1306.ICL001.1
Declination (J2000)
-36 12 26.5
27.0
27.5
28.0
28.5
29.0
17 25 25.45
25.40
25.35
25.30
25.25
Right Ascension (J2000)
Peak flux = 6.1158E-02 JY/BEAM
Levs = 6.116E-03 * (-1, 1, 2, 4, 8, 10)

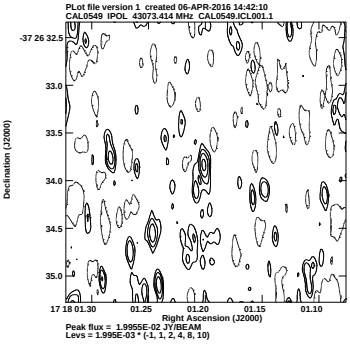

PLot file version 1 created 27-FEB-2016 14:54:35
CAL1334 IPOL 42844.262 MHz CAL1334.ICL001.1
Declination (J2000)
-36 33 51.5
52.0
52.5
53.0
53.5
54.0
17 29 01.20
01.15
01.10
01.05
01.00
Right Ascension (J2000)
Peak flux = -2.1781E-02 JY/BEAM
Levs = 2.178E-03 * (-1, 1, 2, 4, 8, 10)

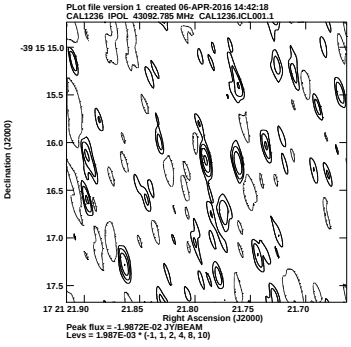

PLot file version 1 created 27-FEB-2016 14:46:24
CAL0642 IPOL 42945.130 MHz CAL0642.ICL001.1
Declination (J2000)
-23 21 50.0
50.5
51.0
51.5
52.0
52.5
18 00 49.74
49.72
49.70
49.68
49.66
49.64
49.62
49.60
49.58
49.56
Right Ascension (J2000)
Peak flux = 7.8863E-03 JY/BEAM
Levs = 7.886E-04 * (-1, 1, 2, 4, 8, 10)

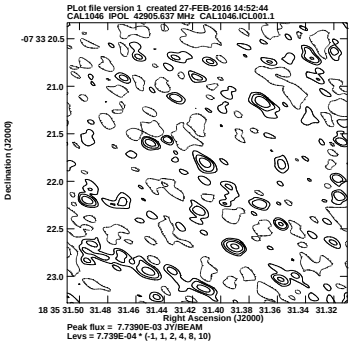

PLot file version 1 created 27-FEB-2016 14:54:40
CAL1178 IPOL 42930.911 MHz CAL1178.ICL001.1
Declination (J2000)
-06 01 31.0
31.5
32.0
32.5
33.0
33.5
18 43 41.34
41.32
41.30
41.28
41.26
41.24
41.22
41.20
41.18
41.16
Right Ascension (J2000)
Peak flux = 8.6204E-03 JY/BEAM
Levs = 8.620E-04 * (-1, 1, 2, 4, 8, 10)

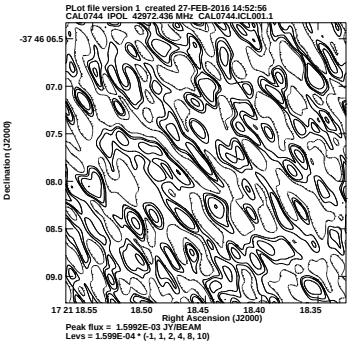

PLot file version 1 created 24-FEB-2016 14:39:29
CAL0599 IPOL 42792.516 MHz CAL0599.ICL001.1
Declination (J2000)
-28 20 06.0
06.5
07.0
07.5
08.0
08.5
17 57 38.20
38.15
38.10
38.05
38.00
Right Ascension (J2000)
Peak flux = 1.5418E-02 JY/BEAM
Levs = 1.542E-03 * (-1, 1, 2, 4, 8, 10)

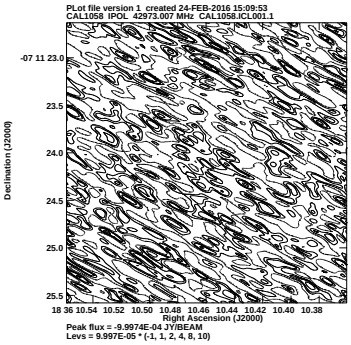

PLot file version 6 created 24-FEB-2016 14:33:26
CAL1076 IPOL 43011.032 MHz CAL1076.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 1.2179E-02 JY/BEAM
Levs = 1.218E-03 * (-1, 1, 2, 4, 8, 10)

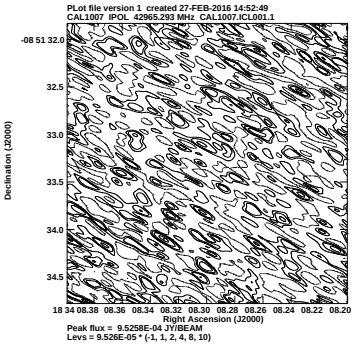

PLot file version 1 created 06-APR-2016 14:42:10
CAL0549 IPOL 43073.414 MHz CAL0549.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 1.9955E-02 JY/BEAM
Levs = 1.995E-03 * (-1, 1, 2, 4, 8, 10)

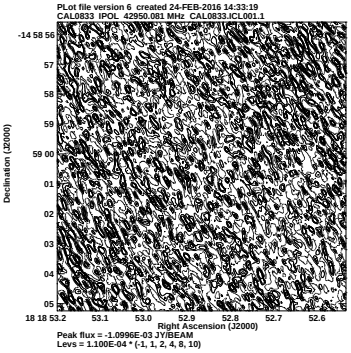

PLot file version 1 created 06-APR-2016 14:42:18
CAL1236 IPOL 43092.785 MHz CAL1236.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = -1.9872E-02 JY/BEAM
Levs = 1.987E-03 * (-1, 1, 2, 4, 8, 10)

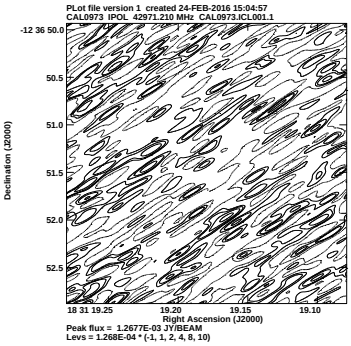

PLot file version 1 created 27-FEB-2016 14:52:44
CAL1046 IPOL 42905.637 MHz CAL1046.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 7.7398E-03 JY/BEAM
Levs = 7.739E-04 * (-1, 1, 2, 4, 8, 10)

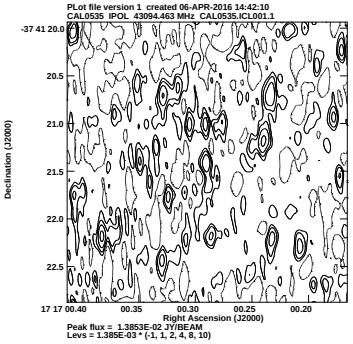

PLot file version 1 created 27-FEB-2016 14:52:56
CAL0744 IPOL 42972.436 MHz CAL0744.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 1.5992E-03 JY/BEAM
Levs = 1.599E-04 * (-1, 1, 2, 4, 8, 10)

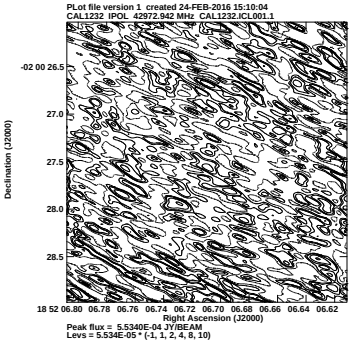

PLot file version 1 created 24-FEB-2016 15:09:53
CAL1058 IPOL 42973.007 MHz CAL1058.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = -9.9974E-04 JY/BEAM
Levs = 9.997E-05 * (-1, 1, 2, 4, 8, 10)

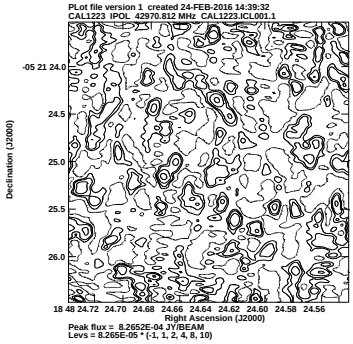

PLot file version 1 created 27-FEB-2016 14:52:49
CAL1007 IPOL 42965.293 MHz CAL1007.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 9.5258E-04 JY/BEAM
Levs = 9.526E-05 * (-1, 1, 2, 4, 8, 10)

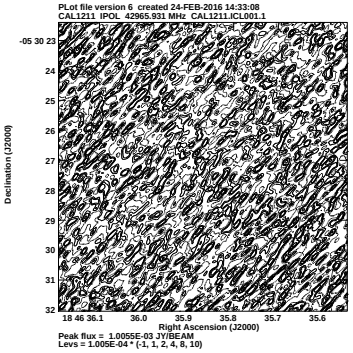

PLot file version 6 created 24-FEB-2016 14:33:19
CAL0833 IPOL 42950.081 MHz CAL0833.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = -1.0996E-03 JY/BEAM
Levs = 1.100E-04 * (-1, 1, 2, 4, 8, 10)

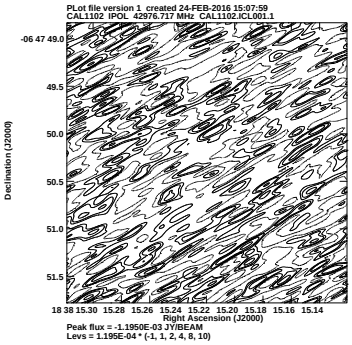

PLot file version 1 created 24-FEB-2016 15:04:57
CAL0973 IPOL 42971.210 MHz CAL0973.ICL001.1
Declination (J2000)
Right Ascension (J2000)
Peak flux = 1.2677E-03 JY/BEAM
Levs = 1.268E-04 * (-1, 1, 2, 4, 8, 10)

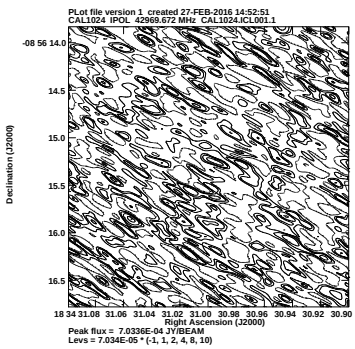

PLot file version 1 created 06-APR-2016 14:42:10
CAL0535 IPOL 43094.463 MHz CAL0535.ICL001.1
Declination (J2000)
-37 41 20.0
20.5
21.0
21.5
22.0
22.5
17 17 00.40
00.35
00.30
00.25
00.20
Right Ascension (J2000)
Peak flux = 1.3853E-02 JY/BEAM
Levs = 1.385E-03 * (-1, 1, 2, 4, 8, 10)

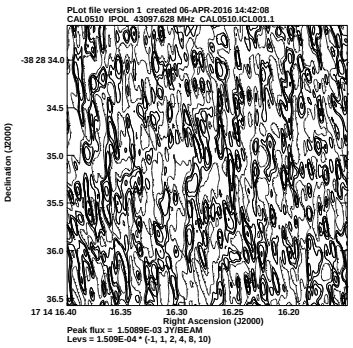

PLot file version 1 created 24-FEB-2016 15:10:04
CAL1232 IPOL 42972.942 MHz CAL1232.ICL001.1
Declination (J2000)
-02 00 26.5
27.0
27.5
28.0
28.5
18 52 06.80
06.78
06.76
06.74
06.72
06.70
06.68
06.66
06.64
06.62
Right Ascension (J2000)
Peak flux = 5.5340E-04 JY/BEAM
Levs = 5.534E-05 * (-1, 1, 2, 4, 8, 10)

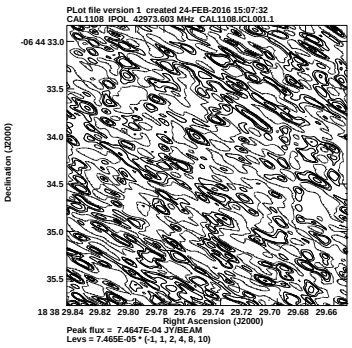

PLot file version 1 created 24-FEB-2016 14:39:32
CAL1223 IPOL 42970.812 MHz CAL1223.ICL001.1
Declination (J2000)
-05 21 24.0
24.5
25.0
25.5
26.0
18 48 24.72
24.70
24.68
24.66
24.64
24.62
24.60
24.58
24.56
Right Ascension (J2000)
Peak flux = 8.2652E-04 JY/BEAM
Levs = 8.265E-05 * (-1, 1, 2, 4, 8, 10)

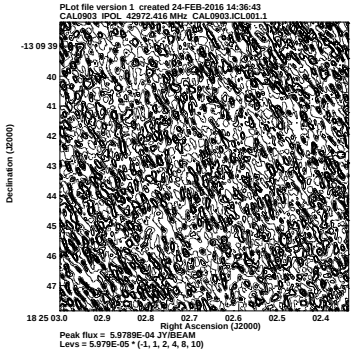

PLot file version 6 created 24-FEB-2016 14:33:08
CAL1211 IPOL 42965.931 MHz CAL1211.ICL001.1
Declination (J2000)
-05 30 23
24
25
26
27
28
29
30
31
32
18 46 36.1
36.0
35.9
35.8
35.7
35.6
Right Ascension (J2000)
Peak flux = 1.0055E-03 JY/BEAM
Levs = 1.005E-04 * (-1, 1, 2, 4, 8, 10)

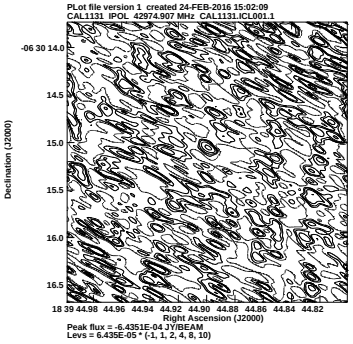

PLot file version 1 created 24-FEB-2016 15:07:59
CAL1102 IPOL 42976.717 MHz CAL1102.ICL001.1
Declination (J2000)
-06 47 49.0
49.5
50.0
50.5
51.0
51.5
18 38 15.30
15.28
15.26
15.24
15.22
15.20
15.18
15.16
15.14
Right Ascension (J2000)
Peak flux = -1.1950E-03 JY/BEAM
Levs = 1.195E-04 * (-1, 1, 2, 4, 8, 10)

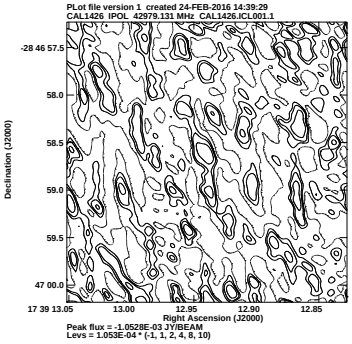

PLot file version 1 created 27-FEB-2016 14:52:51
CAL1024 IPOL 42969.672 MHz CAL1024.ICL001.1
Declination (J2000)
-08 56 14.0
14.5
15.0
15.5
16.0
16.5
18 34 31.08
31.06
31.04
31.02
31.00
30.98
30.96
30.94
30.92
30.90
Right Ascension (J2000)
Peak flux = 7.0336E-04 JY/BEAM
Levs = 7.034E-05 * (-1, 1, 2, 4, 8, 10)

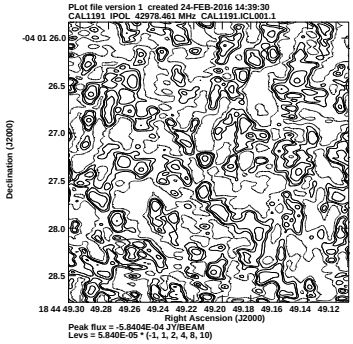

PLot file version 1 created 06-APR-2016 14:42:08
CAL0510 IPOL 43097.629 MHz CAL0510.ICL001.1
Declination (J2000)
-38 28 34.0
34.5
35.0
35.5
36.0
36.5
17 14 16.40
16.35
16.30
16.25
16.20
Right Ascension (J2000)
Peak flux = 1.5089E-03 JY/BEAM
Levs = 1.509E-04 * (-1, 1, 2, 4, 8, 10)

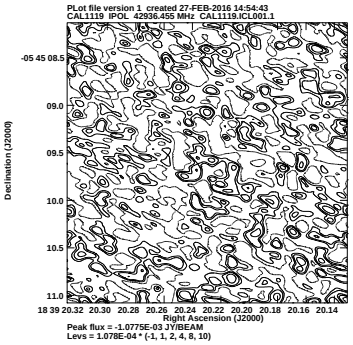

PLot file version 1 created 24-FEB-2016 15:07:32
CAL1108 IPOL 42973.603 MHz CAL1108.ICL001.1
Declination (J2000)
-06 44 33.0
33.5
34.0
34.5
35.0
35.5
18 38 29.84
29.82
29.80
29.78
29.76
29.74
29.72
29.70
29.68
29.66
Right Ascension (J2000)
Peak flux = 7.4647E-04 JY/BEAM
Levs = 7.465E-05 * (-1, 1, 2, 4, 8, 10)

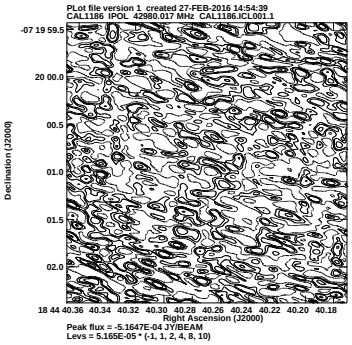

PLot file version 1 created 24-FEB-2016 14:36:43
CAL0903 IPOL 42972.416 MHz CAL0903.ICL001.1
Declination (J2000)
-13 09 39
40
41
42
43
44
45
46
47
18 25 03.0
02.9
02.8
02.7
02.6
02.5
02.4
Right Ascension (J2000)
Peak flux = 5.9789E-04 JY/BEAM
Levs = 5.979E-05 * (-1, 1, 2, 4, 8, 10)

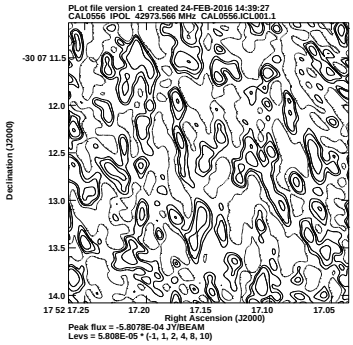

PLot file version 1 created 24-FEB-2016 15:02:09
CAL1131 IPOL 42974.907 MHz CAL1131.ICL001.1
Declination (J2000)
-06 30 14.0
14.5
15.0
15.5
16.0
16.5
18 39 44.98 44.96 44.94 44.92 44.90 44.88 44.86 44.84 44.82
Right Ascension (J2000)
Peak flux = -6.4351E-04 JY/BEAM
Levs = 6.435E-05 * (-1, 1, 2, 4, 8, 10)

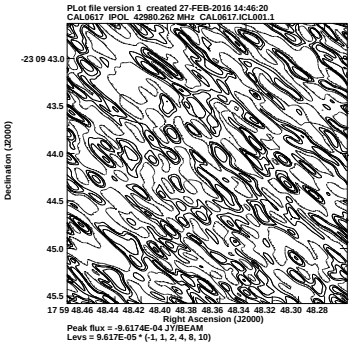

PLot file version 1 created 24-FEB-2016 14:39:29
CAL1426 IPOL 42979.131 MHz CAL1426.ICL001.1
Declination (J2000)
-28 46 57.5
58.0
58.5
59.0
59.5
47 00.0
17 39 13.05 13.00 12.95 12.90 12.85
Right Ascension (J2000)
Peak flux = -1.0528E-03 JY/BEAM
Levs = 1.053E-04 * (-1, 1, 2, 4, 8, 10)

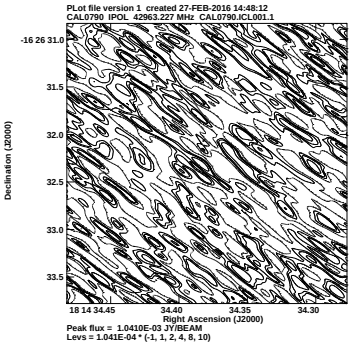

PLot file version 1 created 24-FEB-2016 14:39:30
CAL1191 IPOL 42978.461 MHz CAL1191.ICL001.1
Declination (J2000)
-04 01 26.0
26.5
27.0
27.5
28.0
28.5
18 44 49.30 49.28 49.26 49.24 49.22 49.20 49.18 49.16 49.14 49.12
Right Ascension (J2000)
Peak flux = -5.8404E-04 JY/BEAM
Levs = 5.840E-05 * (-1, 1, 2, 4, 8, 10)

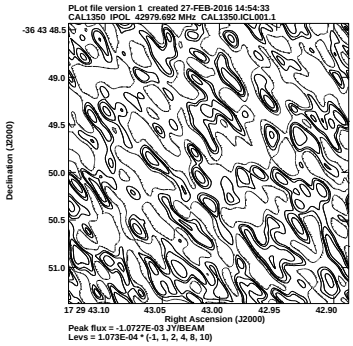

PLot file version 1 created 27-FEB-2016 14:54:43
CAL1119 IPOL 42936.455 MHz CAL1119.ICL001.1
Declination (J2000)
-05 45 08.5
09.0
09.5
10.0
10.5
11.0
18 39 20.32 20.30 20.28 20.26 20.24 20.22 20.20 20.18 20.16 20.14
Right Ascension (J2000)
Peak flux = -1.0775E-03 JY/BEAM
Levs = 1.078E-04 * (-1, 1, 2, 4, 8, 10)

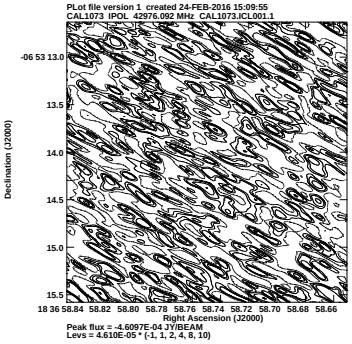

PLot file version 1 created 27-FEB-2016 14:54:39
CAL1186 IPOL 42980.017 MHz CAL1186.ICL001.1
Declination (J2000)
-07 19 59.5
20 00.0
00.5
01.0
01.5
02.0
18 44 40.36 40.34 40.32 40.30 40.28 40.26 40.24 40.22 40.20 40.18
Right Ascension (J2000)
Peak flux = -5.1647E-04 JY/BEAM
Levs = 5.165E-05 * (-1, 1, 2, 4, 8, 10)


PLot file version 1 created 24-FEB-2016 14:39:27
CAL0556 IPOL 42973.566 MHz CAL0556.ICL001.1
Declination (J2000)
-30 07 11.5
12.0
12.5
13.0
13.5
14.0
17 52 17.25 17.20 17.15 17.10 17.05
Right Ascension (J2000)
Peak flux = -5.8078E-04 JY/BEAM
Levs = 5.808E-05 * (-1, 1, 2, 4, 8, 10)


PLot file version 1 created 27-FEB-2016 14:46:20
CAL0617 IPOL 42980.262 MHz CAL0617.ICL001.1
Declination (J2000)
-23 09 43.0
43.5
44.0
44.5
45.0
45.5
17 59 48.46 48.44 48.42 48.40 48.38 48.36 48.34 48.32 48.30 48.28
Right Ascension (J2000)
Peak flux = -9.6174E-04 JY/BEAM
Levs = 9.617E-05 * (-1, 1, 2, 4, 8, 10)


PLot file version 1 created 27-FEB-2016 14:48:12
CAL0790 IPOL 42963.227 MHz CAL0790.ICL001.1
Declination (J2000)
-16 26 31.0
31.5
32.0
32.5
33.0
33.5
18 14 34.45 34.40 34.35 34.30
Right Ascension (J2000)
Peak flux = 1.0410E-03 JY/BEAM
Levs = 1.041E-04 * (-1, 1, 2, 4, 8, 10)


PLot file version 1 created 27-FEB-2016 14:54:33
CAL1350 IPOL 42979.692 MHz CAL1350.ICL001.1
Declination (J2000)
-36 43 48.5
49.0
49.5
50.0
50.5
51.0
17 29 43.10 43.05 43.00 42.95 42.90
Right Ascension (J2000)
Peak flux = -1.0727E-03 JY/BEAM
Levs = 1.073E-04 * (-1, 1, 2, 4, 8, 10)


PLot file version 1 created 24-FEB-2016 15:09:55
CAL1073 IPOL 42976.092 MHz CAL1073.ICL001.1
-06 53 13.0
13.5
14.0
14.5
15.0
15.5
Declination (J2000)
18 36 58.84 58.82 58.80 58.78 58.76 58.74 58.72 58.70 58.68 58.66
Right Ascension (J2000)
Peak flux = -4.6097E-04 JY/BEAM
Levs = 4.610E-05 * (-1, 1, 2, 4, 8, 10)