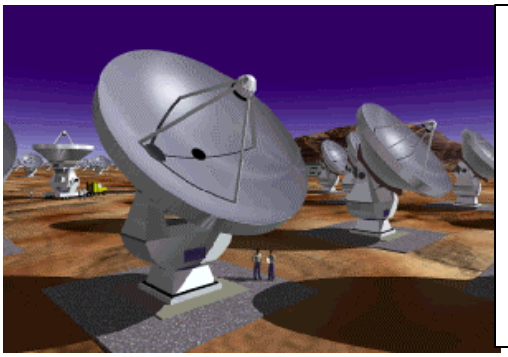


Atacama Large Millimeter Array

ALMA-SW-NNNN

Revision: 0.3

2002-03-11

Test Report

Robert Lucas

ALMA Memo #472: AIPS++ Reuse Analysis Test: Report on Phase I

Test Report

Athol Kemball, Kumar Golap, George Moellenbrock
NRAO
Robert Lucas, Dominique Broguière
IRAM

Keywords: Offline, Software, Science, Calibration, Imaging

Author Signature: Robert Lucas	Date: 2003-03-11
Approved by:	Signature:
Institute:	Date:
Released by:	Signature:
Institute:	Date:

Change Record

REVISION	DATE	AUTHOR	SECTIONS/PAGES AFFECTED
	REMARKS		
0.1	2002-09-01	A. Kemball	All
	Initial version presented to ASAC		
0.2	2003-03-03	R. Lucas	Most
	Changes after Data reduction week in Garching		
0.3	2003-03-11	R. Lucas	Most
	Precise FTE numbers, include final figures		

1 Introduction

This report covers the results of Phase I of the test of AIPS++ using IRAM interferometer data from Plateau de Bure. The report describes the origins and objectives of the test, the course of implementation, and the final results and conclusions. A second phase, in which the software implemented in Phase I will be used to reduce several other data sets, is now underway. A third phase, which will investigate performance for visibility datasets of a size expected from the full ALMA array, has started too.

2 Objectives

The test was proposed in the summer of 2001 by the ALMA Computing Division, in the framework of a re-use analysis of existing off-line data reduction software and subject to general requirements of the ALMA Science Advisory Committee (ASAC). The test was also conducted in order to provide a more concrete basis for ALMA discussions of off-line and pipeline reduction systems and the suitability of AIPS++ for future ALMA needs. The specific terms of the test were then agreed between the AIPS++ project and the ALMA Computing Division in a group including broad representation from the North American and European ALMA partners. Participants in the test have included the following personnel, in various roles: ESO-ALMA (Gianni Raffi and Joe Schwarz); NRAO-ALMA (Brian Glendenning); IRAM-ALMA (Robert Lucas, Dominique Broguière, Frederic Gueth and Jerome Pety) and AIPS++/NRAO Data Management (Tim Cornwell, Athol Kemball, Kumar Golap and George Moellenbrock).

The objectives of the test, as agreed at the outset, were as follows:

- *"How can AIPS++ be adapted to reduce data of an instrument for which it was not initially designed?*
- *How long is the learning curve for developers who have sufficient experience in the processing of millimeter data, but no experience at all the AIPS++ programming environment?*
- *Can we perform an end-to-end experiment on actual, real-life millimeter-wave spectroscopic data?"*

It was agreed to address these questions by performing an end-to-end reduction of IRAM Interferometer data from Plateau de Bure within AIPS++. The scope of the test was defined in terms of the following steps:

- *"Read the raw data in an agreed FITS format (this could be the data format foreseen for the ALMA Test Interferometer).*
- *Perform atmospheric calibration to convert data into T_a* scale*
- *Use atmospheric path length corrected data*

- *Reduce bandpass calibration obtained on a strong celestial point source*
- *Reduce phase and amplitude phase calibration using quasars*
- *Transfer the phase calibration from 3mm wavelength observations to 1mm wavelength observations.*
- *Image continuum and spectroscopic data*
- *Deconvolve single field images"*

  It was agreed that the goal of the test would be to produce a report for the ALMA Project *"containing short, motivated answers to the well-defined questions posed above"*. The duration of the test was set at nine months, ending in April, 2002.

3 Scheduled tasks

The following milestones were agreed for the test:

- *Phase I (end November 2001):*
  - *a. Agree on the export FITS format for visibility data.*
  - *b. Provide a copy of the CLIC/GILDAS code to AIPS++*
  - *c. Select two representative IRAM datasets covering the data reduction scope outlined above.*
    - *i. Provide these uncalibrated data in the agreed FITS format.*
  - *d. Reduce in CLIC/GILDAS*
    - *i. Reduce the test data in CLIC/GILDAS as described in the data reduction scope above.*
    - *ii. Export interim copies of the data after each calibration step in the agreed FITS format.*
    - *iii. Export the calibration solutions for each step.*
    - *iv. Export the final images in FITS format.*
    - *v. Provide a basic text log of the CLIC/GILDAS reduction sequence*
    - *vi. Provide a point of contact to answer queries regarding CLIC or the reduction sequence.*
  - *e. Reduce in AIPS++*
    - *i. Reduce the same test data in AIPS++ as described in the data reduction scope above.*
    - *ii. Provide documented AIPS++ scripts for end-to-end reduction of each test dataset.*
    - *iii. Write a memorandum summarizing the agreement in the final image product and interim calibration solutions.*
- *Phase II (end April 2002):*
  - *f. Reduction of other data*

i. *Select additional IRAM datasets meeting the data reduction scope defined for the test.*

ii. *Reduce these data in AIPS++ using the scrip ts provided from Phase I.*

iii. *Inter-compare with CLIC/GILDAS results*.

- *Optional phase III (extending beyond April 2002):*

g. Evaluate the scalability of AIPS++ from IRAM to ALMA data

i. *Measure the performance of AIPS++ on simulated ALMA datasets using 64 antennas.*

4 Test progression

This section describes the progress of the test and the course of implementation. This is not given in strict chronological progression, but is rather grouped by test area. Test activities are cross-referenced to the scheduled milestones given in Section 3 above, where applicable.

4.1 Collaboration and communication

Communication mechanisms were established in order to facilitate an effective collaboration between the participating groups at the different sites. These included reciprocal visits, teleconferences and e-mail contact. An important part of this test has been the building of a strong technical collaboration between the participating groups. NRAO and IRAM have hosted reciprocal visits during the course of the test, as follows:

a) Dominique Broguiere (IRAM) visited NRAO in Socorro from September 9th to 22nd, 2001;
b) Kumar Golap (NRAO/AIPS++) visited IRAM from November 27th to December 8th, 2001, and spent one day (December 10th) at Observatoire de Paris;
c) Athol Kemball (NRAO/AIPS++) and Kumar Golap (NRAO/AIPS++) visited IRAM from February 6th to March 5th, 2002, and Observatoire de Paris on March 6th;
d) Athol Kemball (NRAO/AIPS++) and Kumar Golap (NRAO/AIPS++) visited IRAM from July 23rd to July 31st, 2002;
e) Athol Kemball (NRAO/AIPS++) and George Moellenbrock (NRAO/AIPS++) visited IRAM from August 24th to September 5th, and August 25th to August 31st respectively.
f) Dominique Broguiere (IRAM) and Michel Caillat (Observatoire de Paris) were invited to, and attended, the internal 2002 AIPS++ Developers' Meeting held in northern New Mexico (USA) from May 27th to June 1st 2002.
g) Tim Cornwell (NRAO) and Kumar Golap (NRAO/AIPS++) visited IRAM from December 2nd to December 7th 2002.
h) Finally Tim Cornwell, Kumar Golap, George Moellenbrock (NRAO/AIPS++), Robert Lucas, Dominique Broguière (IRAM) visited ESO from February 27th to February 31st 2003, to finish phase I of the test.

4.2 Data interchange format

The first action at the start of the test was to agree a data interchange format to allow calibrated and un-calibrated visibility data from the IRAM Interferometer to be imported into AIPS++ [3(a)]. This data format needed to meet certain requirements: i) full transfer of all information from the IRAM Interferometer native output format required to permit subsequent calibration and imaging in AIPS++; ii) maximal compatibility with the existing CLIC/GILDAS data format to facilitate inter-comparison of the data reduction in both packages, and iii) to be of possible future use so that effort invested in a data filler in AIPS++ would not be without use beyond the test itself.

Existing data-interchange formats, including UVFITS and FITS-IDI, were unsuitable based on these criteria, although fillers exist within AIPS++ for these formats. It was mutually agreed to use the ALMA-TI format adopted for the ALMA test interferometer (Lucas & Glendenning 2001); it was further agreed that the AIPS++ project would provide a filler for this interchange format.

4.3 Test data for Phase I

As the test dataset for Phase I, the IRAM group selected an observation of the young quadruple system GG Tau, taken in 1997 by Stephane Guilloteau as principal investigator. The IRAM project code for the observations was G067, and these data were subsequently published as Guilloteau, Dutrey & Simon (1999). The selection of the Phase I test data met [3(b)] above.

These observations consist of simultaneous observations of the HCO^+ J=1-0 transition (which has a rest frequency of 89.188523 GHz), and the ^{13}CO 2-1 transition (at a rest frequency of 220.398686 GHz) towards GG Tau. Calibrators for bandpass, phase, amplitude and the absolute flux density scale were observed throughout the project. These included the sources 0528+134, 0415+379, MWC 349, CRL 618, 2230+114 and NRAO150. The data were correlated in a range of line and continuum frequency sub-bands, of varying channel number and spectral resolution, across the bands of interest containing the two transitions listed above. The project was observed on eight separate days in 1997, spanning a range of Plateau de Bure array configurations. The observation dates were 7 February 1997, 10 February 1997, 20 February 1997, 25 March 1997, 31 March 1997, 18 September 1997, 16 October 1997 and 18 October 1997.

These data were re-reduced in CLIC by the IRAM group and made available in ALMA-TI format at various stages of calibration as specified by [3(d)] above.

4.4 CLIC/GILDAS

IRAM provided a copy of the CLIC/GILDAS software to AIPS++, as agreed in [3(b)] and provided excellent assistance throughout the test in describing the algorithms used in the package and their specific implementation in the code base. In turn, the AIPS++ project became familiar with the technical architecture and implementation structure of

CLIC/GILDAS and explored the implementation of the IRAM algorithms at the source code level in the package. NRAO also installed a development version of CLIC/GILDAS at the AOC, with assistance from IRAM.

4.5 Reduction sequence within CLIC/GILDAS

This section includes a brief description of the reduction sequence within CLIC/GILDAS. A full summary of IRAM data reduction techniques can be found, for example, in the proceedings of the IRAM summer school (Dutrey et al. 2000), and references therein.

The IRAM Interferometer can observe simultaneously in two receiver bands, at 1 mm and 3 mm wavelength. Each day of observing and each receiver band for a given project is calibrated separately (although 3 mm phase calibration is used in standard practice to guide 1 mm phase calibration, as discussed below). The calibrated data from all epochs are then combined to produce separate calibrated visibility files in each receiver band suitable for continuum or line imaging within the GILDAS imaging package. The customary calibration steps for each observing epoch and receiver band are as follows:

4.5.1 Atmospheric calibration

Single-dish amplitude calibration of IRAM Interferometer data proceeds using measured hot- and cold-load temperatures, the known antenna forward efficiency and an atmospheric transmission model, ATM, available in CLIC (Cernicharo 1985). An iterative approach is taken to solve for the atmospheric opacity and temperature (Dutrey 2000), to permit proper scaling of the system temperatures for atmospheric absorption and spillover. This scaling is infrequently repeated in subsequent off-line data reduction, but this feature is available in CLIC as command *ATMOSPHERE*

4.5.2 Quality assessment of radiometric phase corrections

The IRAM Interferometer applies an on-line radiometric phase correction based on total-power measurements in a 500 MHz wide region of the standard 1 mm receiver band. Both radiometrically phase-corrected and un-corrected visibility data are preserved in the telescope output format. The on-line phase correction is not entirely reversible due to time-averaging, but it is possible to accept or reject the radiometric phase correction in post-processing. The quality metric used is the ratio of the vector-averaged amplitude of the phase-corrected and uncorrected data for each calibrator scan. This acceptance or rejection criterion is extrapolated to cover half the scan preceding and half the scan following the calibrator scan. This step is known as *MONITOR 0* in CLIC. Re-computation of the radiometric phase correction is possible in post-processing, but over a longer time interval than that used in the on-line system (*MONITOR Dt)*.

4.5.3 Bandpass calibration: CLIC/RF

This calibration step involves determining the bandpass correction for each receiver sideband using a calibrator source of sufficiently high correlated flux density. The sub-bands in each sideband are averaged over time, gridded onto a common frequency axis and decomposed into amplitude and phase components. The amplitude and phase response are then fit separately as antenna-based Chebyshev functions over frequency in a global least-squares decomposition. This method can be shown (Anterrieu 1992) to be a superior calibration estimator for the case of low signal-to-noise-ratio (SNR) data, as is common for millimeter interferometers. The improved statistical properties derive from the use of antenna-based polynomial functions, which decrease the number of degrees of freedom over that in regular, discretely-sampled self-calibration over finite solution intervals (as is common in centimeter wavelength interferometry). It is also possible to decrease the variance in low -SNR calibration solutions by fitting polynomials to the regular self-calibration solutions post-facto. This is the technique used for the BIMA millimeter array, and is available in AIPS++ as tool *gainpolyfitter*, written by the NCSA/BIMA group within the AIPS++ consortium.

The bandpass corrections per sideband are normalized to unit mean amplitude and zero mean phase on application. Antenna-based ratios of the complex electronic gains between the two sidebands for each receiver are maintained separately, and applied along with the normalized bandpass as a separable calibration correction. The option also exists in RF to pre-normalize the input visibility data going to the solver by dividing each baseline by a mean phasor over frequency of a wider sub-band in the sideband. This improves the coherent time pre-average of the visibility data before the antenna-based solution.

The CLIC bandpass solver by default masks *n* central frequency channels (due to the Gibbs effect) and the upper and lower edges of each sub-band used in the bandpass solution. The latter parameter in CLIC is expressed as the percentage of the band edge to mask, with a default of 5 %.

4.5.4 Phase calibration: CLIC/PHASE

Once the bandpass correction and sideband ratios have been determined, the phase correction over time is determined using a similar antenna-based polynomial least-square decomposition as used for the bandpass over frequency (Anterrieu 1992; Lucas 2000). The most common polynomial form used is a spline polynomial, with the spline knot positions determined automatically using a heuristic taking the time series sampling as input.

The phase correction polynomials are determined per receiver band, combining all constituent sub-bands and sidebands. The 1 mm receiver band is dual sideband and both are combined before the phase solution. At 3 mm wavelength, a single sideband is preferred for optimal sensitivity. For the Phase I test data from project G067, this is the lower sideband.

The 3mm calibration curve is obtained by a direct spline fit into the 3mm calibrator phases.

The 1mm phase calibration is obtained in two steps:

1. For each data point on the phase calibrators, the 3mm observed phase is subtracted from the 1mm observed phase, after scaling by the ratio of frequencies;

2. An incremental spline polynomial is fitted in the residuals of the 1mm calibrator phases.

The final 1mm calibration curve is the superposition of the 3mm calibration curve, scaled by the same ratio of frequencies, and of the incremental 1mm calibration curve.

Care is taken not to introduce spurious phase discontinuities in this process.

4.5.5 Flux density calibration: CLIC/FLUX

Once the bandpass response, sideband gain ratios and phase corrections over time have been determined and applied, it is possible to establish the flux density scale for the visibility data relative to known amplitude calibrators. This is an iterative process, requiring informed user interaction. In each iteration a set of calibrators with flux densities to be held fixed are specified; this allows antenna efficiencies and the flux densities of all other calibrators to be computed as input to the next iteration. This process is repeated until consistent flux densities are obtained and the computed antenna efficiencies are as constant over time as possible. Secondary flux density calibrators (MWC 349 and CRL 618), which have model flux densities, are also observed to provide a complementary check of the flux density scale (Dutrey 2000).

4.5.6 Amplitude calibration: CLIC/AMPLI

Once the flux density scale has been established, residual amplitude errors over time are fit as spline polynomials for the calibrated, sideband-averaged data, using the method of Anterrieu (1992). This is analogous to the case of phase calibration, described above.

4.5.7 Concatenation of calibrated data: CLIC/UVT

The calibrated data on the target source are concatenated and combined to a single calibrated output visibility file for each sub-band of interest, suitable for final imaging in the GILDAS imaging package. This concatenation step is possible once the final calibration for each observing epoch in the project has been completed.

4.5.8 Imaging

Imaging then proceeds using standard methods image formation and deconvolution techniques, as available in the GILDAS imaging package.

4.6 Reduction Within Aips++

This section includes a brief description of the reduction process in AIPS++. An important part of this re-use test was to provide a demonstration of algorithm transfer to AIPS++ from an existing millimeter reduction package such as CLIC/GILDAS, which contains advanced and mature millimeter-wave reduction algorithms. These algorithms, and the ALMA-TI data, were fully integrated into

the standard calibration and imaging framework within AIPS++. These capabilities were collected at the highest level as a thin tool, *iramcalibrater,* which provides access to the new features added to the standard tool, *calibrater,* in a format similar to that provided by CLIC.

4.6.1 Data import

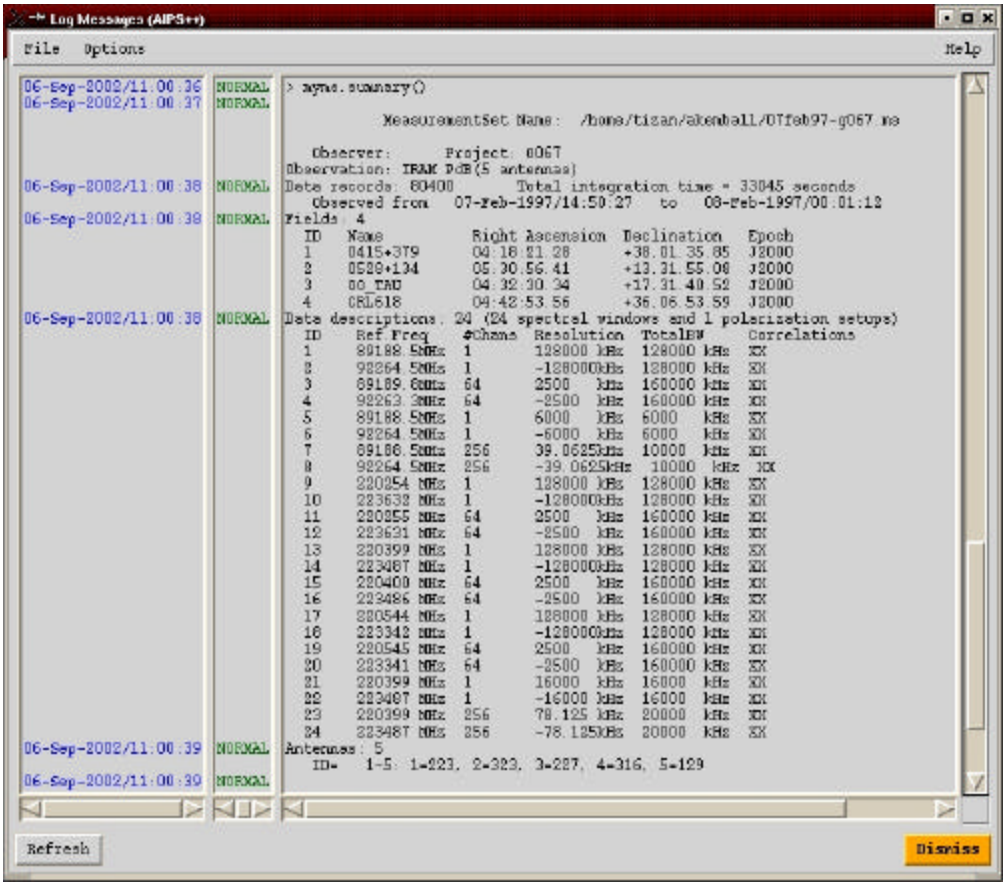
As noted above, AIPS++ agreed to provide a data filler for the ALMA-TI data interchange format as part of this test. This was implemented as tool *almati2ms* in AIPS++, using the existing AIPS++ FITS and MS access classes. Data for each observing epoch for each project were provided as FITS binary files in ALMA-TI format; there were typically of order ten ALMA-TI FITS files for each epoch. The *almati2ms* data filler reads and converts these files from the ALMA-TI format, and provides input data selection and the option of concatenation to an existing output MS as user options. As each epoch is calibrated separately, the ALMA-TI files for each epoch were concatenated to a single output MS. The filler also provides the option of output file compression (using 16-bit scaled words to represent 32-bit floating point visibility data).

4.6.2 Data format

The AIPS++ project has adopted a generic data format for visibility and single-dish data, developed after wide consultation within the AIPS++ consortium, and therefore strongly informed by the data formats in use over a broad range of instruments already in the consortium. The current revision is MS v2 (Kemball & Wieringa 2000; Wieringa & Cornwell 1996). The data format was developed to represent data from radio-telescopes in a generic format, in order to permit the development of instrument-independent data reduction capabilities. The overwhelming fraction of data recorded at radio telescopes constitutes common physical quantities or parameters, and these are represented in a common core framework within the MS. The option is provided however, to add telescope-specific data columns or sub-tables so as not to restrict the development of site-specific reduction capabilities when required. In practice, the telescope-specific data are most often auxiliary data taken by specific back-end devices or other monitoring equipment not common to other telescopes. However, these additional columns can also point to omissions in the core MS format which may need to be unified in the future. The MS data format permits use of a common calibration framework (as described below) and maximizes the re-use of instrument-independent core data reduction modules in AIPS++. However, only site-specific applications know about telescope-specific data, and only they can therefore make use of these data.

The ALMA-TI data were well-matched to the core MS data framework and were filled in this format. However, the on-line radiometric corrections were added as telescope-specific MS data columns; this option is fully supported within the MS definition. Both corrected and un-corrected data need to be carried forward from the telescope output format as the option is provided in subsequent data reduction to override the on-line phase correction subject to a quality metric, as noted above. An irreversible selection of phase-corrected or uncorrected data in the filler could have been supported without any telescope-specific columns in the MS format. The need for telescope-specific columns to

capture the on-line radiometric phase corrections (and allow them to be reversible) indicates a weakness in the core MS data format in this area. Radiometric phase correction schemes should be represented in a common format in the core MS framework, and this could easily be done. A process is underway at present to develop a common format for such data, which are not used by any telescopes in the AIPS++ consortium at present.



4.6.3 Quality assessment of the on-line radiometric phase corrections: iramcalibrater.phcor()

This step was implemented as a Glish script in AIPS++ (as part of *iramcalutil.g)*, applying the same quality metric used in CLIC/GILDAS to select or reject the radiometric phase-corrections applied by the on-line system. The primary reason this reduction step was implemented in Glish was to provide an example of application development in the scripting layer, as opposed to C++. Glish also provides an environment in which efficient algorithm development and exploration is possible. However, as a Glish script, this application runs significantly slower than if it were implemented in a compiled language, such as C++. It would be straight-forward to migrate this algorithm to C++ in the future.

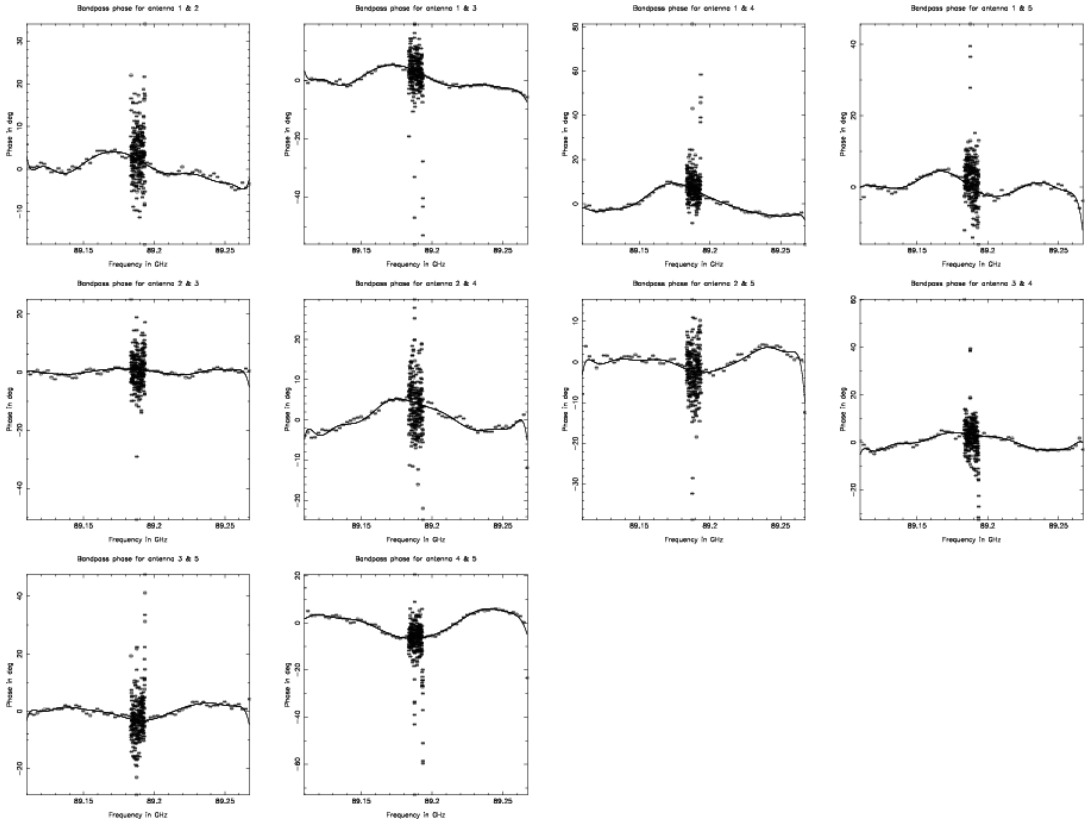
This step in the reduction modifies the data column in the MS depending on the selection or rejection of the on-line radiometric phase-corrected or un-corrected data. This step is also, reversible, and can be repeated as required.

4.6.4 Calibration and imaging framework

The AIPS++ package is built on a principle of instrument-independent reduction. This stems from a design objective to support (then) future instruments such as ALMA, and from the organization of the project as an international consortium. Instrument-independence is achieved by using a generic data format (Kemball & Wieringa 2000; Wieringa & Cornwell 1996), and a generic calibration and imaging formalism (Hamaker, Bregman & Sault 1996). The latter formalism, known as the Measurement Equation (ME) within AIPS++, has a more complete treatment of interferometer calibration errors, allows arbitrary polarization, and permits arbitrary parameterization of individual calibration components. Individual calibration components are represented in this framework as Jones matrices. A generic calibration table format (Kemball 2001) completes the elements required for instrument-independent development of common calibration capabilities in AIPS++. Support for arbitrary parameterization allows algorithms from other instruments to be easily migrated to AIPS++ if they are re-structured to meet a simple interface, and in turn allows them to be re-used elsewhere. Each specialized Jones matrix representation is used directly in the ME at the position of that particular calibration component. As such, they plug-in directly to the common instrument-independent framework, but also have considerable freedom in how they solve for, or compute their individual parameterized corrections.

4.6.5 Bandpass calibration: iramcalibrater.rf()

The bandpass solver was represented in AIPS++ as a specialized B Jones calibration matrix, with parameters for the Chebyshev coefficients, the valid polynomial domain and associated scale and reference factors. This capability was added to the general calibration tool, *calibrater*, as function *setsolvebandpoly()*. This new bandpass Jones matrix solver re-uses the CLIC/GILDAS FORTRAN fitting kernel *polyant*, as an illustration of code re-use and algorithm migration. The calibration solutions are written to, and applied from a specialized B Jones calibration table, within the generic calibration table format.

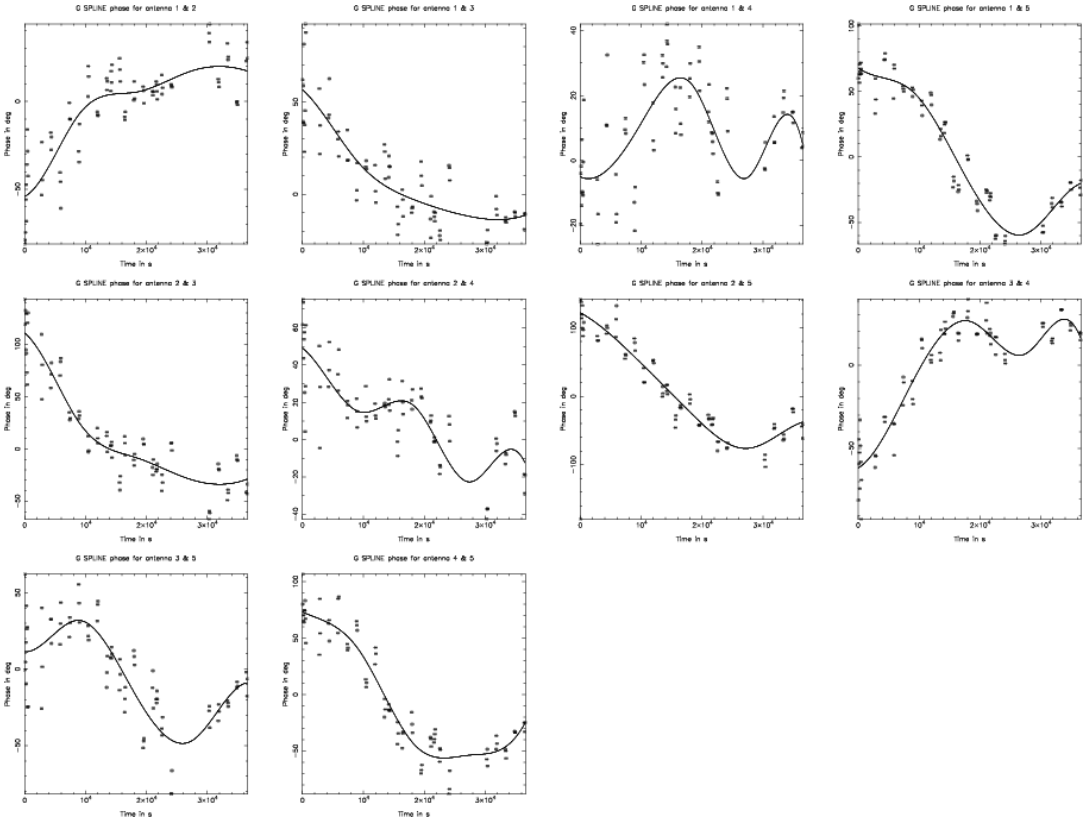


3mm bandpass fit for 1997-feb-20.

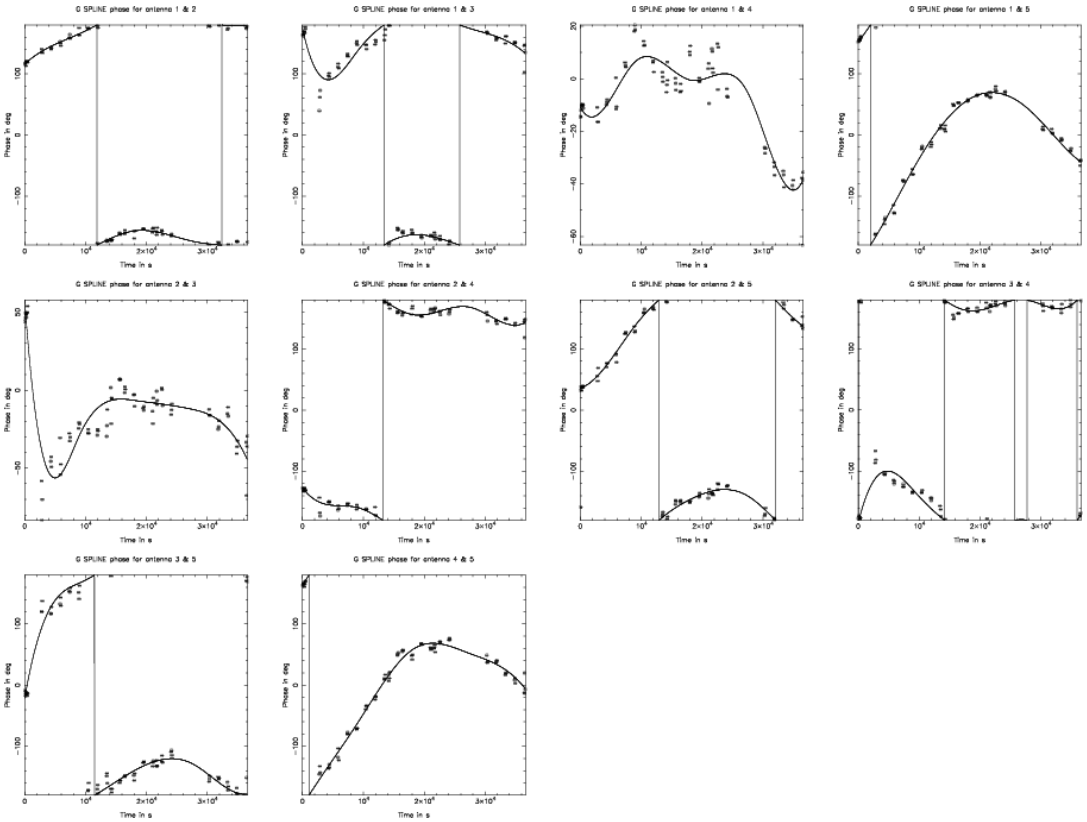
4.6.6 Phase calibration: iramcalibrater.phase()

The representation of phase calibration over time in the form of spline polynomials, was made available both as a specialized G Jones electronic gain matrix and as a specialized T Jones atmospheric calibration Jones matrix. The spline polynomials were treated as a specialization of general polynomials, containing extra parameters to record the position of the spline knots over time, but falling within the generic calibration table format. This capability was added to general calibration tool as *calibrater.setsolvegainspline()*. Analogous to the case of RF, the CLIC FORTRAN solver fitting kernel *splinant* was re-used by the solver for this Jones matrix specialization.

The option was provided in AIPS++ to pre-apply the 3 mm phase corrections to the 1 mm data before solving for the residual phase calibration in the higher band. The 3 mm phase corrections in this case are scaled by the ratio of frequencies between the two bands and serve to reduce the variance and discontinuities in the 1 mm phase data before solution, as described in the standard CLIC reduction sequence above.



3mm phase fit for 1997-feb-20.



1mm phase fit for 1997-feb-20

4.6.7 Flux density calibration: iramcalibrater.flux()

The flux-density scale computation performed by CLIC task FLUX was implemented in AIPS++ as a Glish tool *iramfluxcal* within the file *iramcalutil.g*. Glish was chosen for this application due to its interactive nature, and the environment it offers for ease of algorithm development. This tool performs the same calculations as its counterpart CLIC/FLUX and also produces a plot of normalized antenna efficiency over time at each step in the iterative process of establishing the flux density scale. This process can be repeated until a satisfactory solution is obtained.

4.6.8 Amplitude calibration: iramcalibrater.amp()

Once the flux density scale has been established, residual amplitude errors are fitted as spline polynomials over time, as in CLIC, using the algorithm of Anterrieu (1992). These are represented as part of the G Jones phase polynomial matrices and calibration components solved for in the PHAS step above. These components store amplitude and phase corrections as separate polynomials within the same table, and offer the option of amplitude-only, phase-only or amplitude and phase solution. When applied, these matrices are computed as a single complex polynomial. These polynomials are scaled directly to by FLUX to establish the overall flux density scale.

4.6.9 Concatenation of calibrated data: iramcalibrater.uvt()

The concatenation of the calibrated visibility data from each observing epoch were provided using the data access and concatenation capabilities provided by the existing AIPS++ MS tools, *ms* and *msconcat.* This step writes separate calibrated MeasurementSets for each sub-band of interest for the target source, suitable for subsequent imaging.

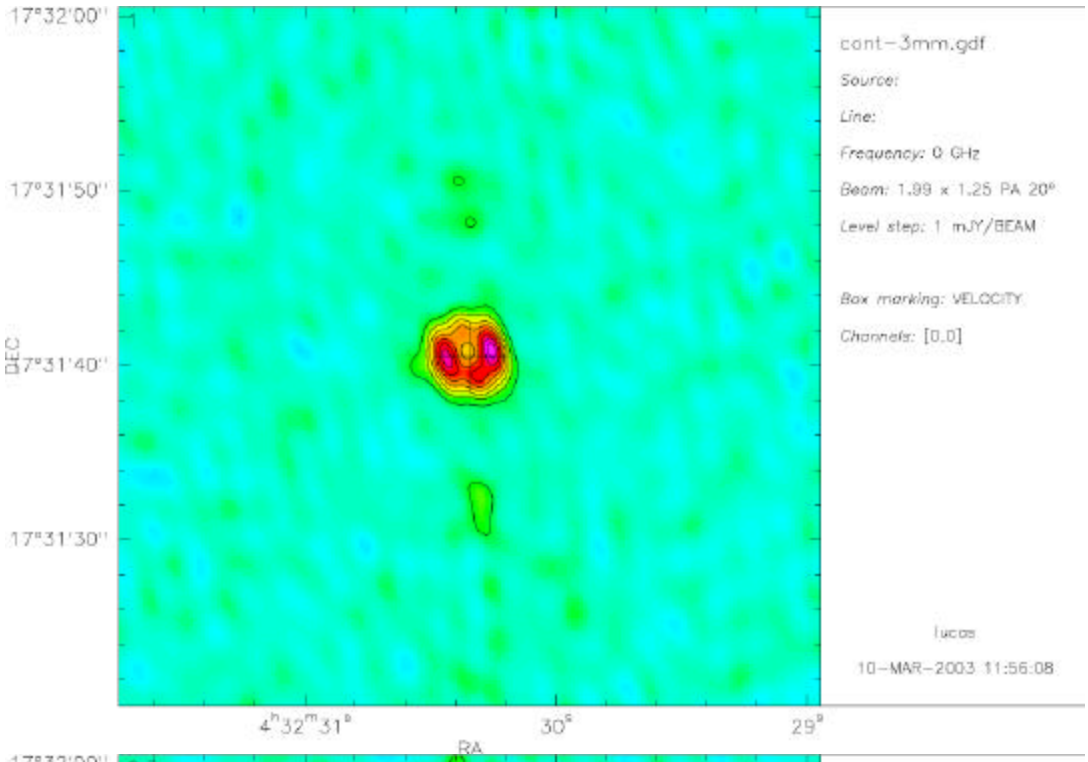
4.6.10 Imaging

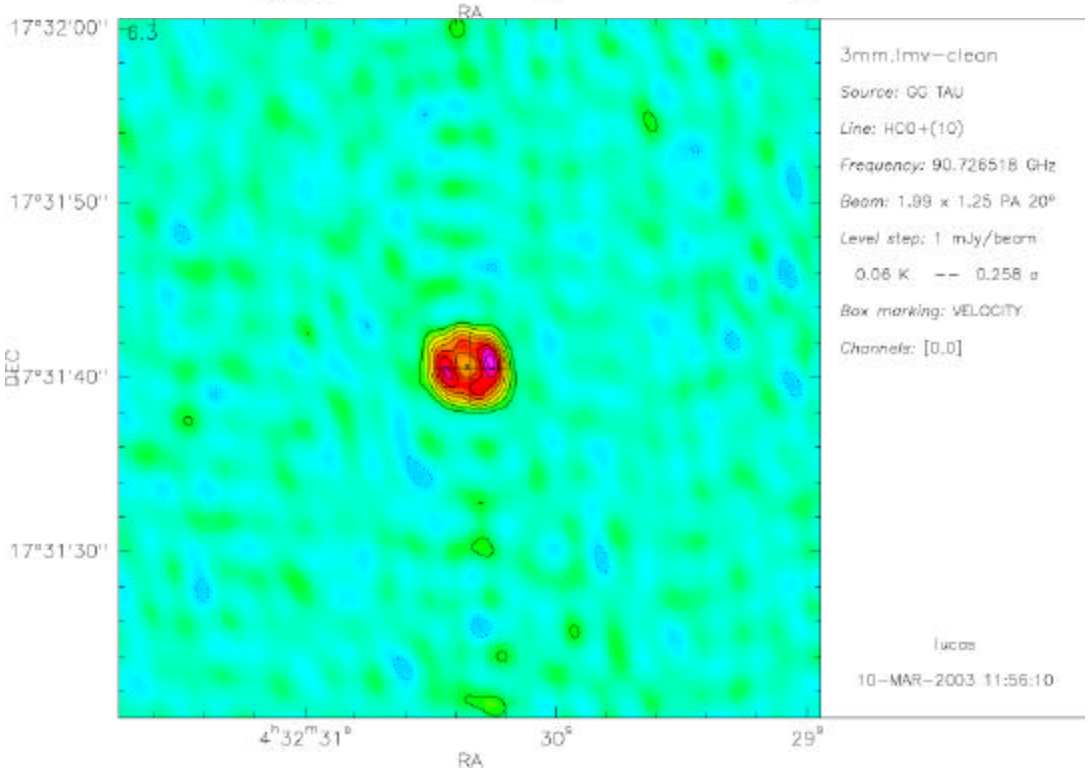
The data were imaged directly using the general AIPS++ imaging and deconvolution tool, *imager*. No major changes in the AIPS++ imaging software proved necessary for the test, although some small defects were corrected in channel selection, made visible by the IRAM data. The calibration corrections and images produced were inter-compared between CLIC and AIPS++ by inspection.

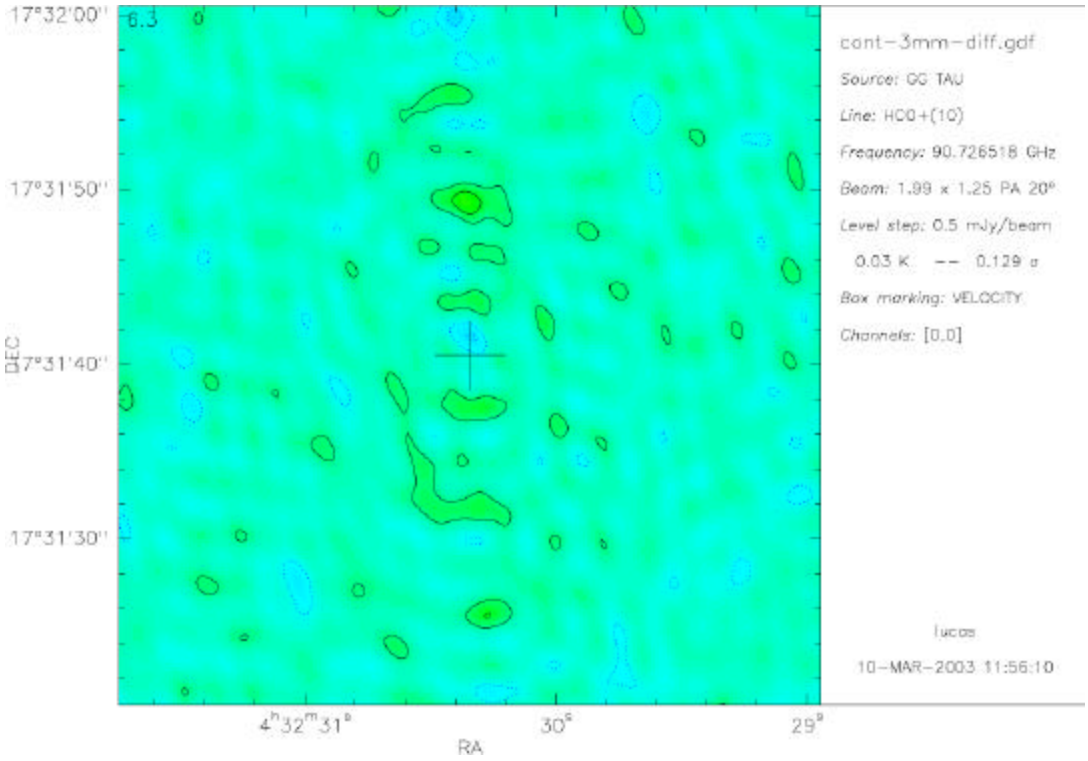
4.7 Comparison of results

We have made difference images and fidelity images (ratio of difference image to reference image) inside Gildas (as a script was available to do this).

4.7.1 3-mm continuum images



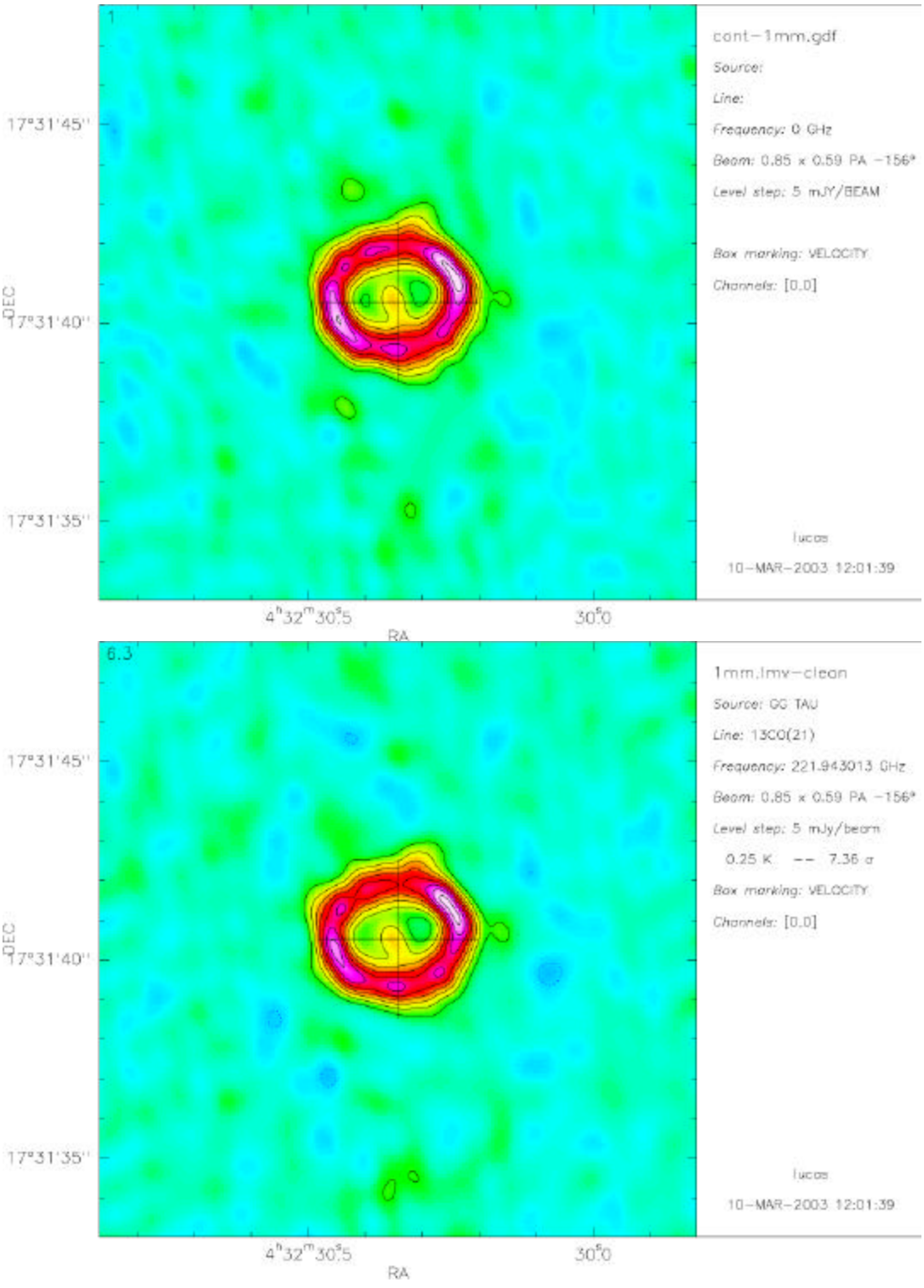


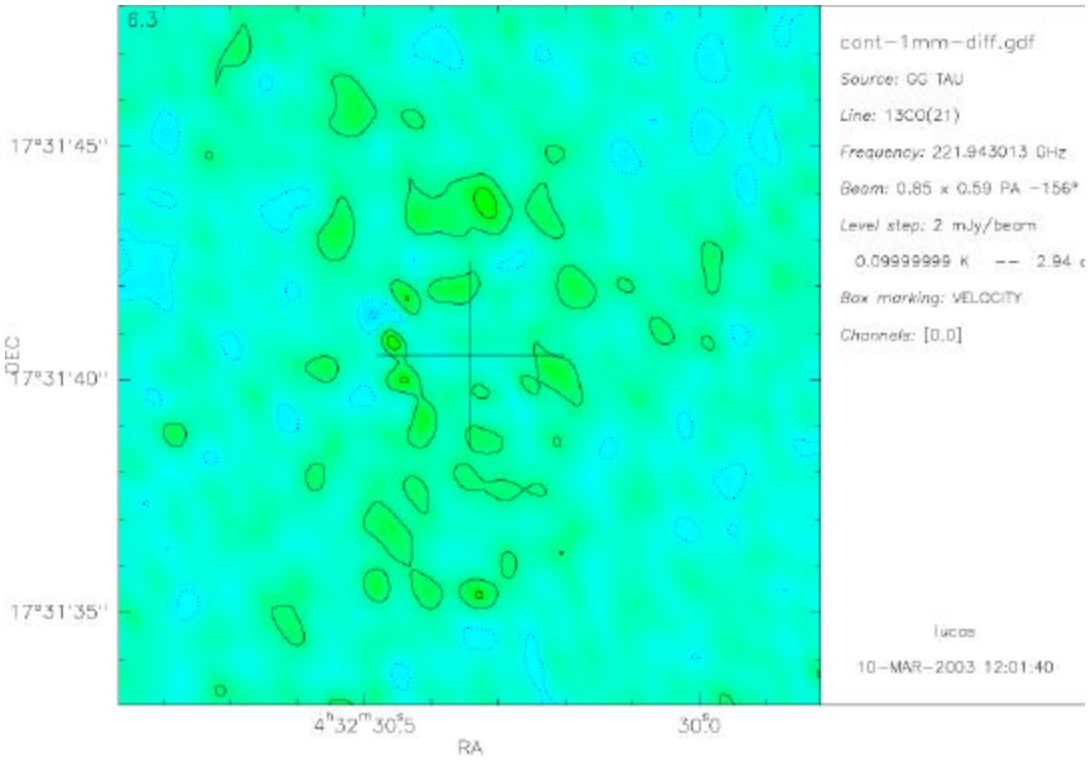


The above figures depict the Aips++ image, the Gildas image, and the difference image. In the Aips++ image the integrated flux is 38 mJy and the noise is 0.27 mJy/beam. In the Gildas image the integrated flux is 36 mJy and the noise is 0.35 mJy/beam.

In the difference image the integrated flux is 1.8 mJy and the noise is 0.20 mJy/beam.

4.7.2 1-mm continuum images

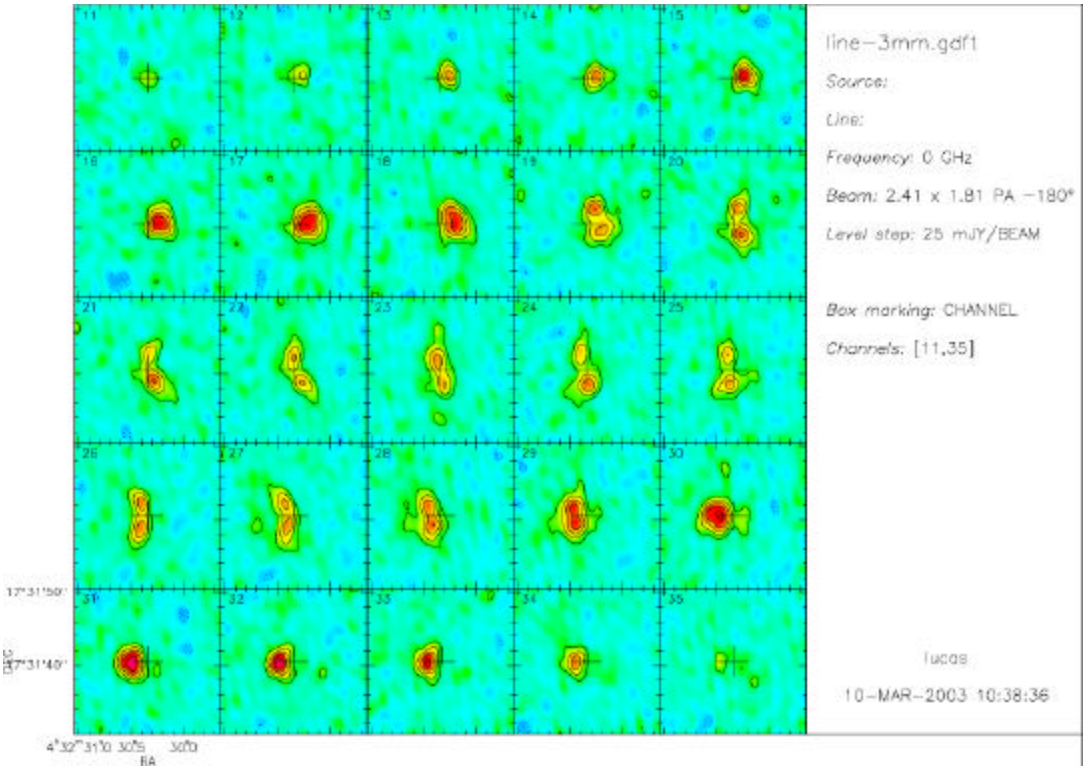


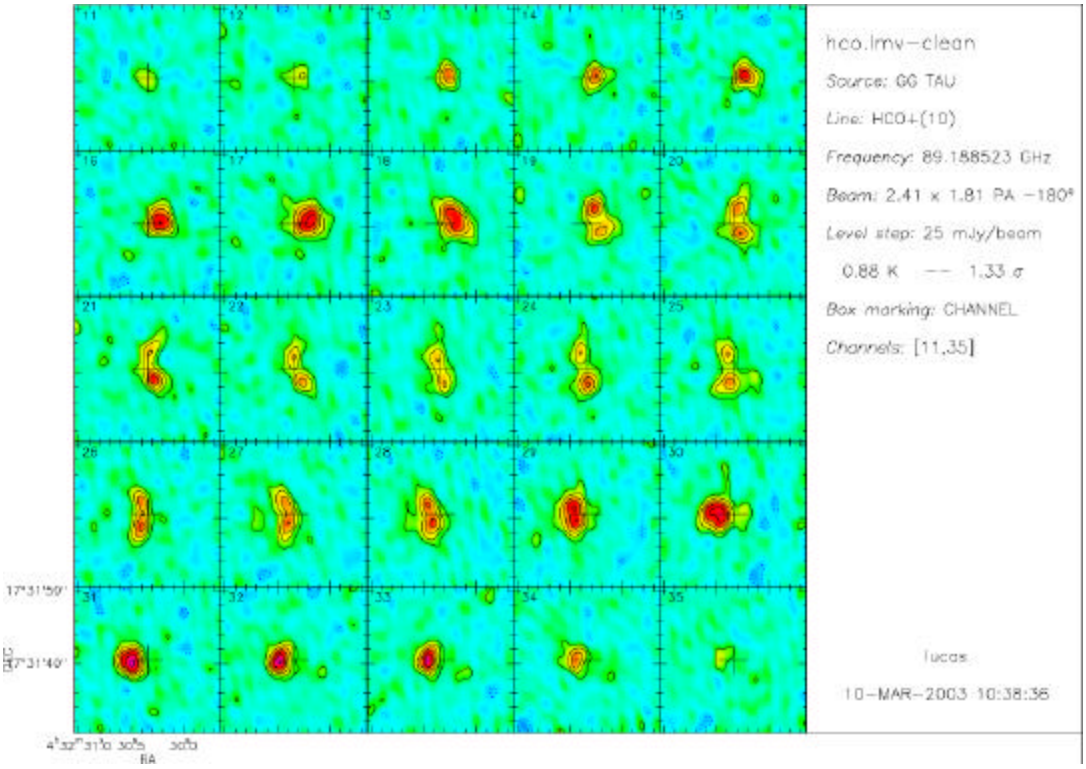


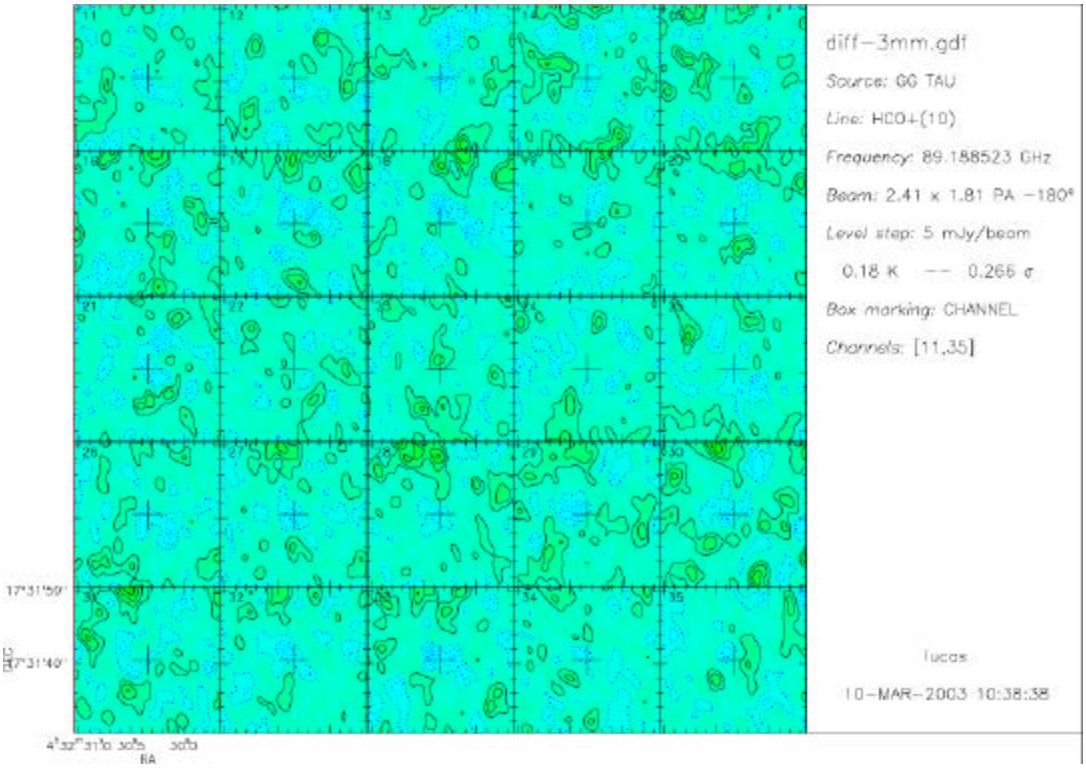
The above figure depict the Gildas image, the Aips++ image, and the difference image. . In the Aips++ image the integrated flux is 529 mJy and the noise is 1.3 mJy/beam.In the Gildas image the integrated flux is 488 mJy and the noise is 1.5 mJy/beam

In the difference image the integrated flux is 30 mJy and the noise is 1.2 mJy/beam.

4.7.3 3-mm line images

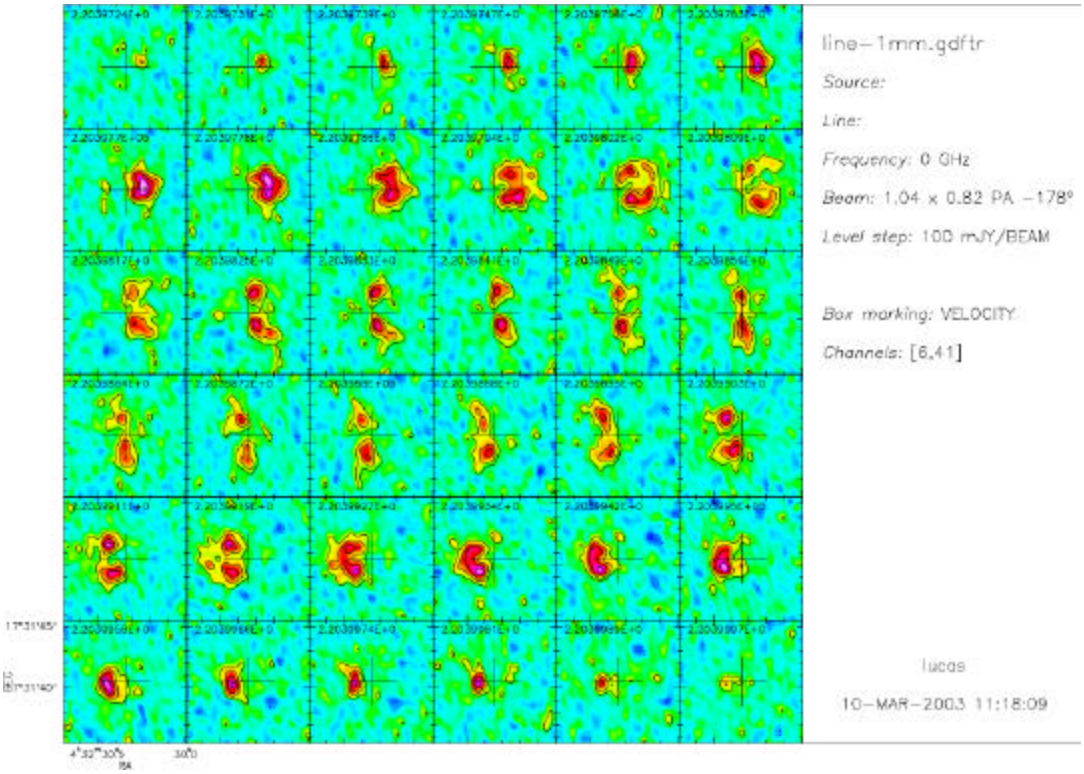


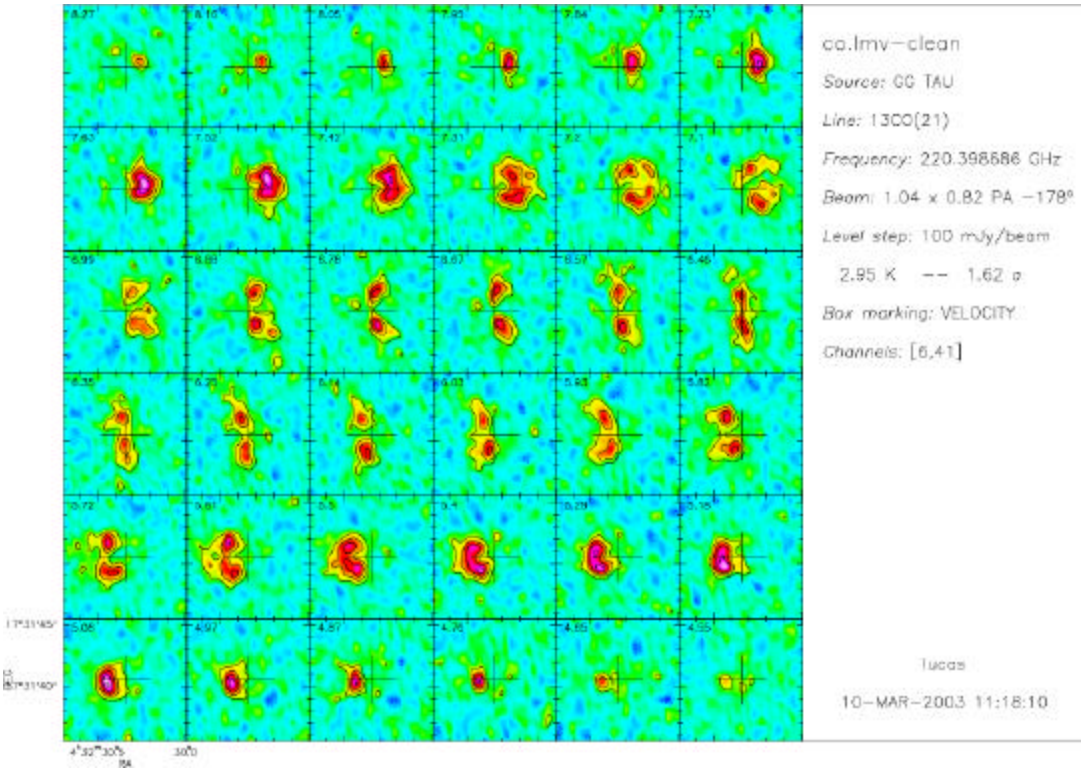


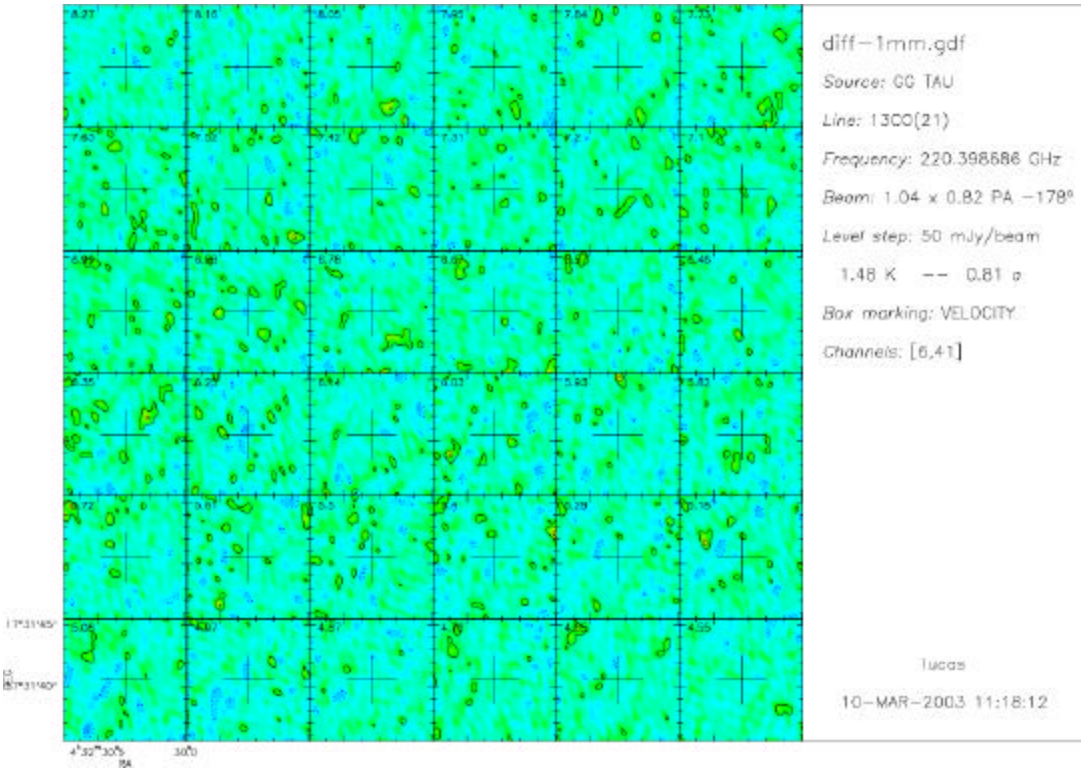


The above figures show the Aips++ image, the Gildas image, and the difference image of HCO+ in GG Tau. The noise in emission-free channels is 9.0, 9.5 and 5.2 mJy/beam in each of the three images.

4.7.4 1-mm line images







The above figures show the Aips++ image, the Gildas image and the difference image of 13CO(2-1) in GG Tau. The noise in emission-free channels is 36, 32 and 24 mJy/beam in each of the three images.

4.7.5 Visibilities

Figures 1-4 show point-for-point uv data comparisons for the 10 baselines of the 1997 February 10 g067 dataset. Each plot shows, in the real/imaginary plane, the unaveraged calibrated visibilities for GG_TAU as produced from both packages. For the high-resolution line data, time-averaged spectral plots are also shown. CLIC is shown as red circles, AIPS++ as green stars, and the complex difference as blue dots. Clearly, the point-for-point correspondence is very good. There are some differences in detail, including effects due to differences in smoothing the data before bandpass calibration (CLIC does, AIPS++ does not), and to small differences in the flagging of data before determining the calibration (one unflagged outlier can drag the solutions obtained in AIPS++ and introduce noticable differences in these plots). Aside from such effects, there also remains a significant phase offset (typically < 5 degrees) between the AIPS++ and CLIC results. This is thought to be due to differences in the normalization of the phase solutions introduced during the processing, e.g., CLIC (AIPS++) uses a baseline-based (antenna-based) calibration of the time-dependent phase before obtaining the bandpass calibration. The size of the cluster of blue dots in each plot is due primarily to this phase difference. Nonetheless, the image comparison shows that these detailed differences are largely unimportant.

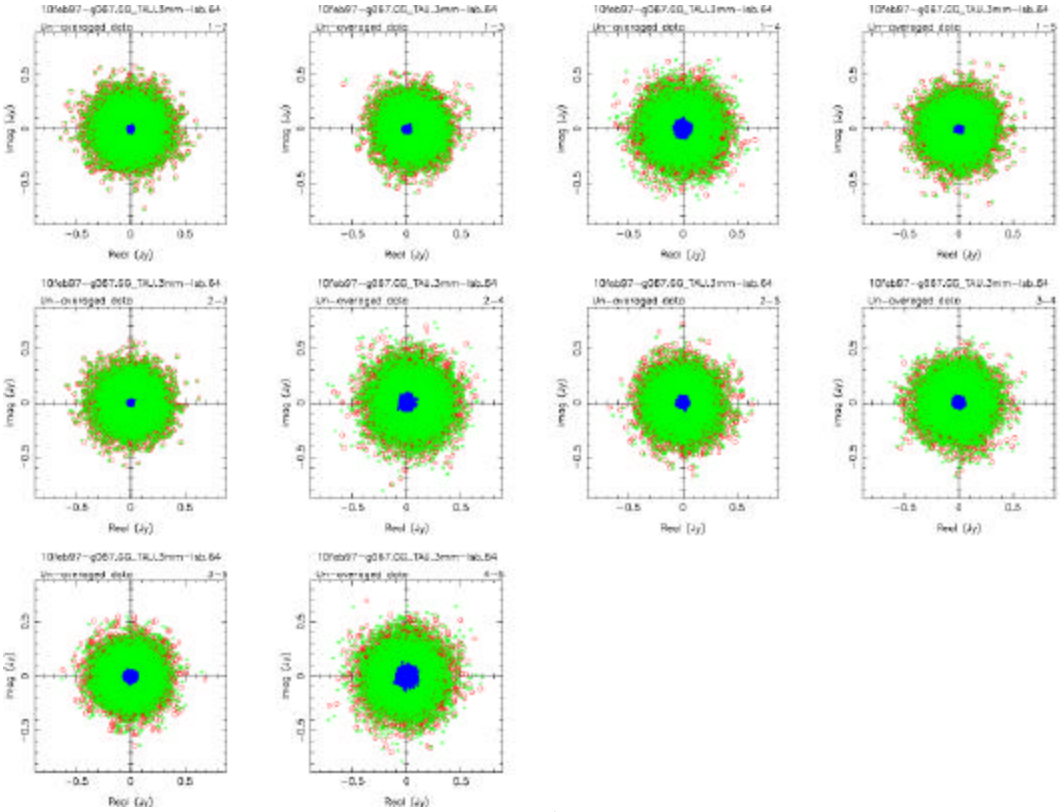


Figure 1: Low-spectral-resolution 3mm (58 of 64 channels) calibrated(unaveraged) visibility comparison for GG_TAU data obtained from bothpackages, with visibilities plotted as real vs. imaginary, for all 10baselines. CLIC data is shownas red circles, AIPS++ as green stars, and the complex difference as blue dots. See text for explanation.

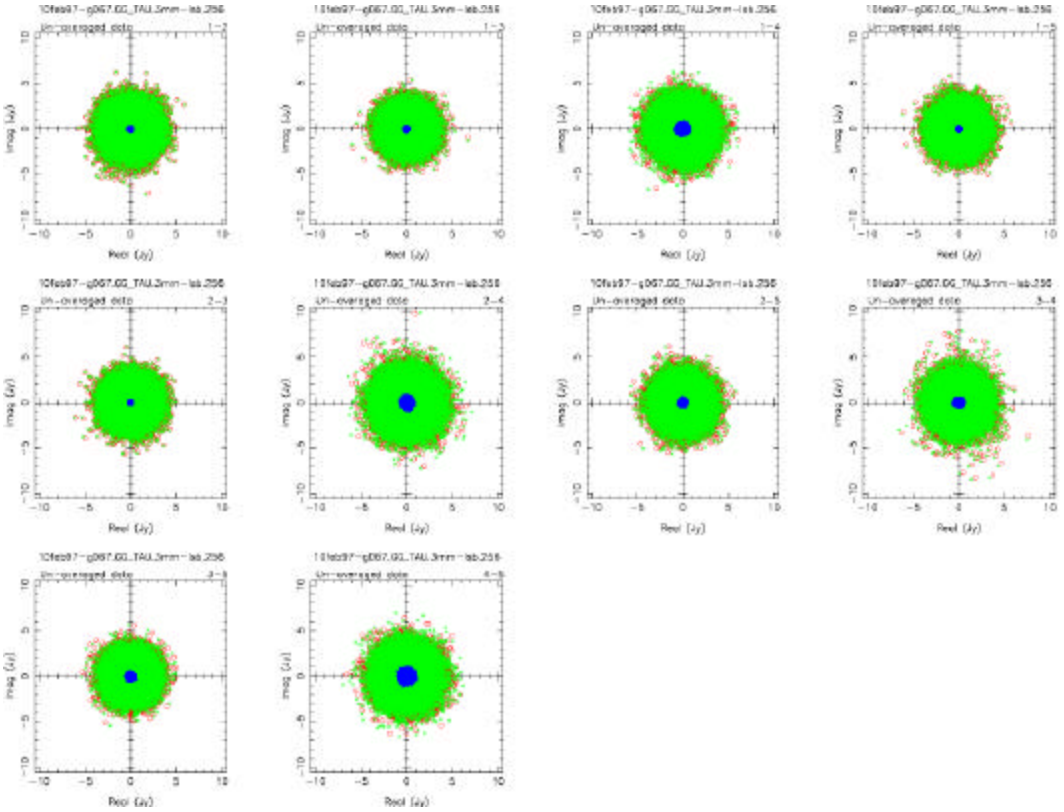


Figure 2a: Same as Figure 1, for high-spectral-resolution (195 of 256 channels) 3mm data.

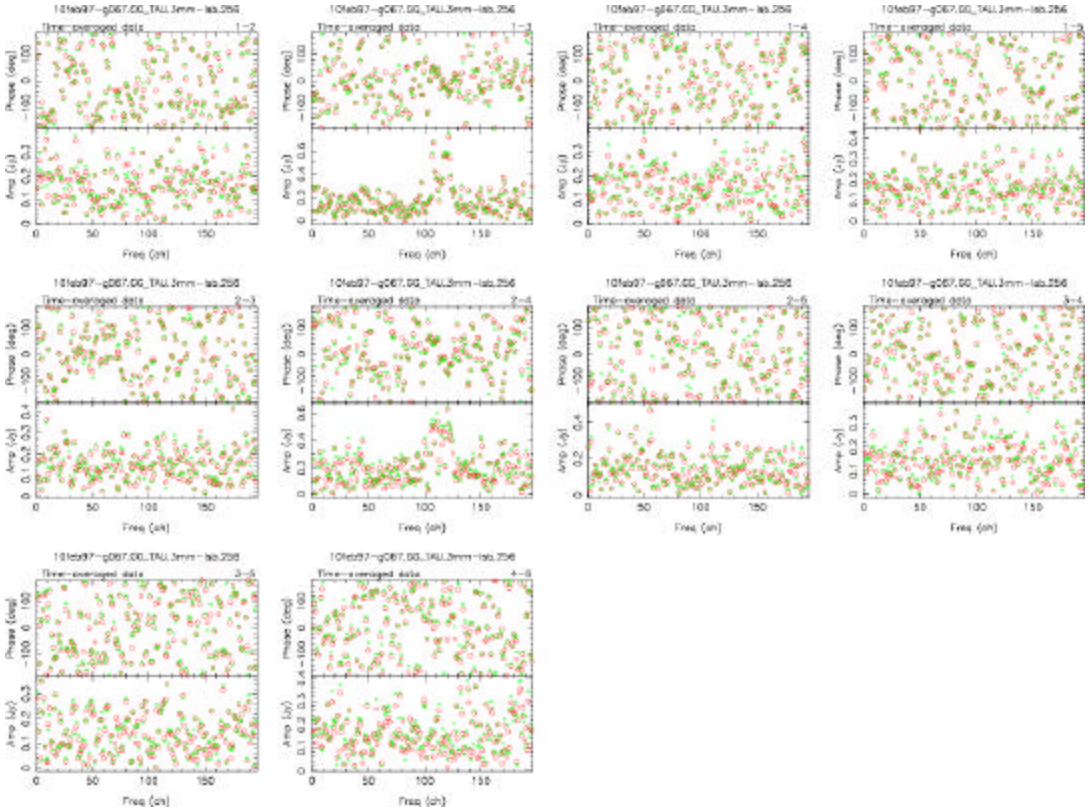


Figure 2b: Data from Figure 2, averaged in time to show spectral agreement.

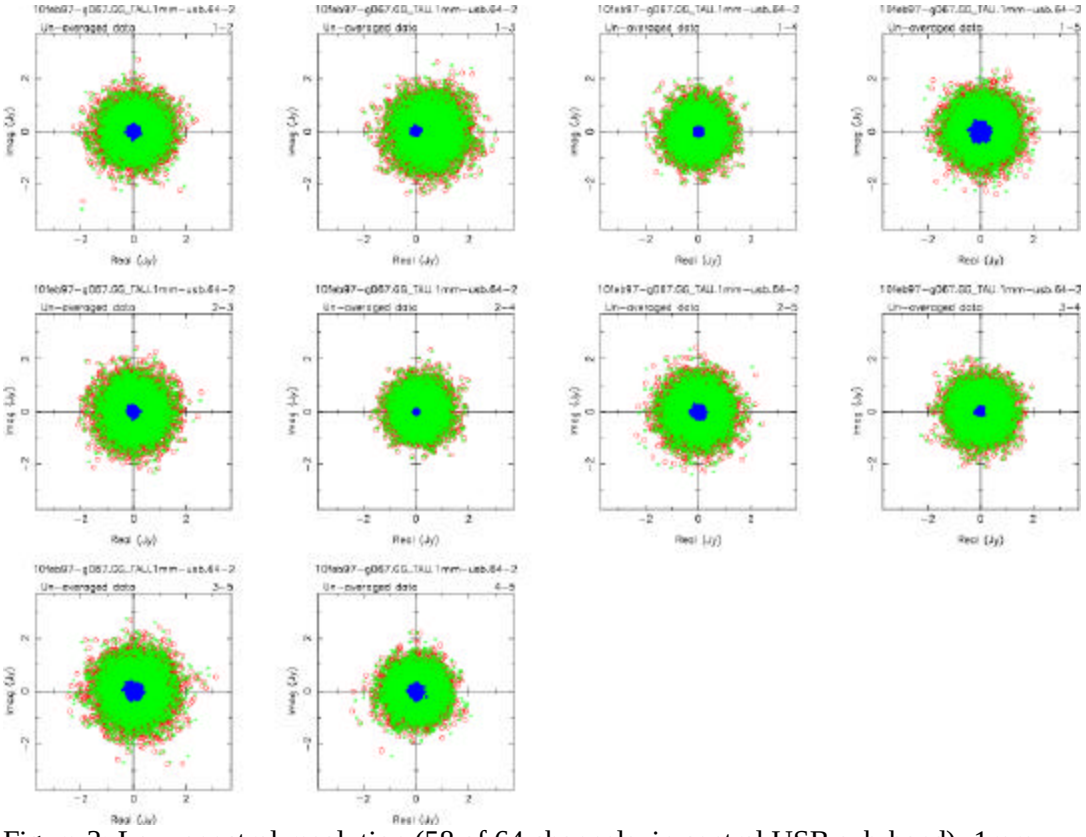


Figure 3: Low-spectral-resolution (58 of 64 channels, in central USB sub-band) 1mm calibrated (unaveraged) visibility comparison for GG_TAU data obtained from both packages, with visibilities plotted as real vs. imaginary. CLIC data is shown as red circles, AIPS++ as green stars, and the complex difference as blue dots.

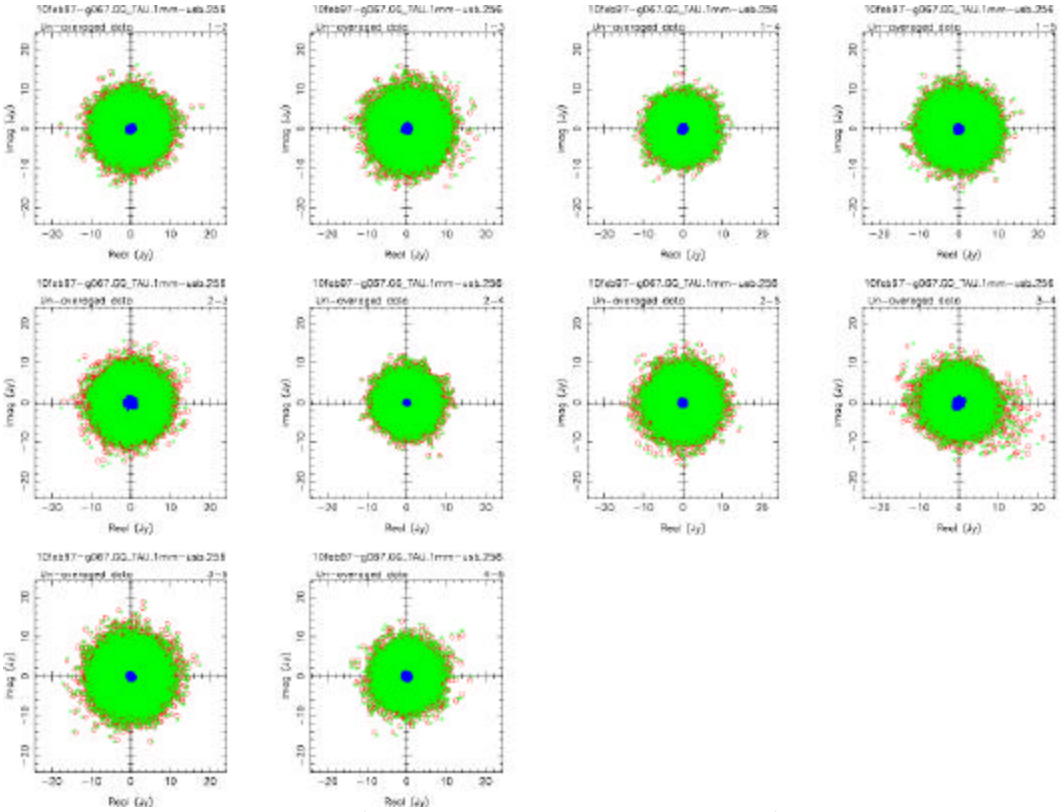


Figure 4a: Same as Figure 3, for high-spectral-resolution (230 of 256 channels, USB) 1mm data.

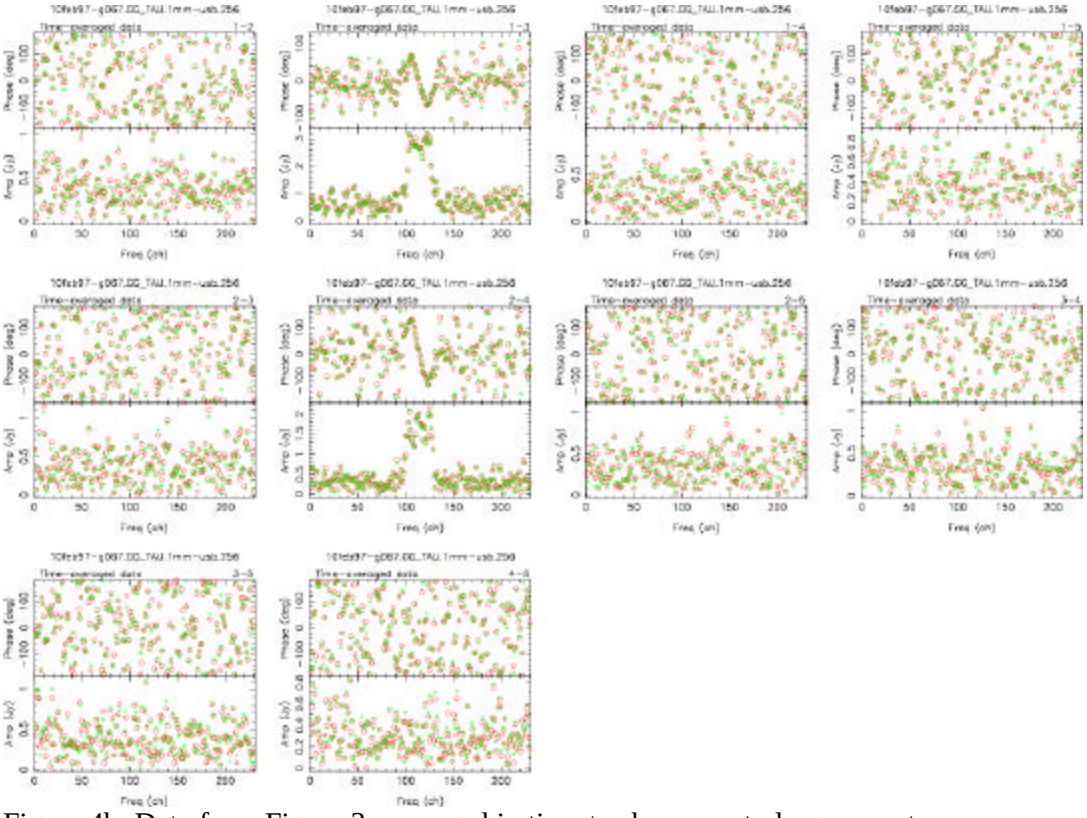


Figure 4b: Data from Figure 3, averaged in time to show spectral agreement.

5 Conclusions

We feel that this test has been an overall success and a valuable activity and important investment for the ALMA project. It has succeeded in building a strong technical and scientific collaboration between the AIPS++ project and the scientific software group at IRAM. As part of this collaboration it has allowed information exchange on a broad range of issues concerning millimeter-wave reduction techniques de veloped at IRAM for the Plateau de Bure interferometer, and their specific implementation in the CLIC/GILDAS package. IRAM algorithms have been successfully migrated to AIPS++ and millimeter-wave spectroscopic data from Plateau de Bure reduced end-to-end. It has also allowed information to be exchanged on the structure and functions of the AIPS++ package, the nature of development processes within the project and the means by which new or existing algorithms can be migrated to the package.

The test has taken longer than originally envisaged in the initial schedule. The AIPS++ resource expenditure on the test is estimated at approximately 32 FTE-weeks over the period from September 2001 to September 2002. Adding subsequent work needed in the third quarter of 2002 and early 2003 to terminate phase I (12 FTE-weeks), we reach approximately 0.8-1.2 FTE-year overall. For comparison, total AIPS++ development efforts are, yearly, approximately 12.5 FTE. The schedule delays do not have obvious technical origins but arose rather from conflicting resource demands on AIPS++ in other areas during this period.

The agreed test objectives posed some specific questions. These are addressed in the points below:

5.1 "How can AIPS++ be adapted to reduce data of an instrument for which it was not initially designed ?"

AIPS++ was designed with a philosophy of instrument-independence, which is captured in the underlying data format and generic calibration and imaging formalism. The test has demonstrated that it is possible to migra te advanced millimeter-wave algorithms from CLIC/GILDAS to AIPS++ and fully integrate them into this generic framework. The IRAM algorithms have been implemented as specialized Jones matrices in the Measurement Equation, as is done for any other calibratio n solver added to the system. In addition, these solvers have re-used some software components directly from CLIC by design. The AIPS++ data format was found suitable for the ALMA-TI data used in this test. However, radiometric phase-corrected data were supported as telescope-specific columns, as noted above.

5.2 "How long is the learning curve for developers who have sufficient experience in the processing of millimeter data, but no experience at all of the AIPS++ programming environment ?"

During the course of the test, AIPS++ developers have shared information with developers at IRAM regarding C++ development in AIPS++, and have provided access to code added for IRAM reduction. In addition, two developers from ADACE attended the 2002 internal AIPS++ developers week in May, which is the annual technical forum

within the project. However, this particular test does not add much new information to the particular question posed here. It confirms that Glish development is reasonably easy with a very short lead time for new developers, but that C++ development using the full AIPS++ library takes more time, and benefits from close contact with the communication channels available to developers within the AIPS++ project. Experience within AIPS++, unrelated to this test, shows variable start-up times for new developers in AIPS++, largely depending on their familiarity with C++ and radio astronomy data reduction. This period can be as short as 6 weeks and as long as several months.

5.3 "Can we perform an end-to-end experiment on actual, real-life millimeter-wave spectroscopic data ?"

The test has shown that this is true, and that millimeter-wave spectroscopic data can be reduced end-to-end in AIPS++.

6 Future directions

We all feel that this strong collaboration should continue and build upon the successes to date; in the short-term the Phase II activities will continue. Activities not planned as part of the test, but which we agree would be worth exploring as part of the continuing ALMA collaboration, include the development of a custom GUI for IRAM reduction, built on the existing tool *iramcalibrater*, evaluation of the current algorithms with IRAM mosaic data, and use of the IRAM algorithms on high-frequency VLA data. All are small incremental costs beyond the current investment. At this point the reduction of IRAM mosaic data has been included in Phase II.

An additional phase, Phase III, which concerns performance of millimeter-wave reduction algorithms on very large datasets, such as those expected from the full ALMA array has started in February 2002. The current anticipated objectives for Phase III are :

- ***Explore millimeter reduction of ALMA-sized datasets in AIPS++***
- ***Establish a set of standard performance benchmarks for ALMA***
- ***Provide initial information to allow resource estimates for future AIPS++ ALMA development in the area of large datasets***

The anticipated process for Phase III is currently envisaged to be as follows:

- Performance work will proceed in a stepwise approach, working from smaller to larger problem sizes
- Define and agree the associated benchmark codes and sequence, based on capabilities implemented in Phase I & II
- For each agreed benchmark:
    - Simulate data and add to AIPS++ Global Data System
    - Add associated benchmark code and script to AIPS++

- Measure and profile AIPS++ performance
- Optimize AIPS++ or estimate scope of work for optimization

- Technical considerations:
  - Memory model:
    - Estimate maximum memory based on extrapolated desktop system in 2007
    - Measure performance at a range of memory sizes
  - I/O model:
    - Clearly separate single-channel I/O problems (those less than a reasonable fraction of current SCSI rates) from parallel I/O problems (multiples of single-channel rates)
    - Parallel I/O for ALMA is a longer-term problem and will require specific resource allocations

7 References

Anterrieu, E., 1992, *PhD thesis,* Univ. Toulouse.

Cernicharo, J., 1985, *Atmospheric model: ATM,* IRAM Internal Report.

Dutrey et. al. (ed.), 2000, *IRAM Milllimeter Interferometry Summer School II Proceedings,* IRAM internal publication.

Guilloteau, S., Dutrey, A. & Simon, M., 1999, *GG Tauri: the ring world*, A&A, **348,** 570.

Kemball, A. & Wieringa, M., 2000, *MS v2,* AIPS++ Note 229 (http:://aips2.nrao.edu)

Kemball, A., 2001, *Calibration table format v2,* AIPS++ Note 240 (http://aips2.nrao.edu).

Lucas, R., 2000, *Bandpass and phase calibration, IRAM Millimeter Interferometry Summer School II Proceedings,* IRAM internal publication.

Lucas, R. & Glendenning, B., 2001, *ALMA Test Interferometer Raw Data Format,* ALMA Computing Memo. 15.

Wieringa, M. & Cornwell, T., 1996, *MS v1,* AIPS++ Note 199 (http://aips2.nrao.edu)

8 Appendix A: New Aips++ files

This appendix contains a list of the files which were added to AIPS++ during this test, the underlying classes and tools in each file, and a brief description of their purpose.

ALMA-TI data filler		
Filename	**Class or tool**	**Purpose**
almati2ms.g	almati2ms	Tool to convert ALMA-TI data into an AIPS++ Measurement Set (MS)
almati2ms.help		Help file for the almati2ms tool
DOalmati2ms[.h,.cc]	almati2ms	C++/Glish binding for the almati2ms tool
AlmaTI2MS[.h,.cc]	AlmaTI2MS	Class to provide all filler functions for import of ALMA-TI data into AIPS++
almati2ms.cc		Glish/C++ binding for the almati2ms tool
almati2msFactory[.h,.cc]	almati2msFactory	Glish/C++ binding for the almati2ms tool

User inferface to the CLIC reduction functions		
Filename	**Class or tool**	**Purpose**
iramcalibrater.g	iramcalibrater	Tool to provide similar IRAM data reduction functions to those available in CLIC/GILDAS
iramcalutil.g	iramphcor iramfluxcal	Glish tools used by iramcalibrater.g to implement PHCOR and FLUX
iramcalibrater_meta.g		File defining the toolmanager GUI interface for the iramcalibrater tool
iramcalibrater.help		Help file for the iramcalibrater tool
alma.help		Definition of a new ALMA package in the AIPS++ documentation system

RF bandpass solver

Filename	Class or tool	Purpose
BJonesPoly[.h,.cc]	BJonesPoly	A specialized BJones matrix implementation to solve for a Chebyshev polynomial bandpass
BJonesDesc[.h,.cc], BJonesMCol[.h,.cc], BJonesTable[.h,.cc], BJonesMBuf[.h,.cc] parametricsolver.f	BJonesPolyDesc, BJonesPolyMCol, BJonesPolyTable, BJonesPolyMBuf polyant, splinant and others	Calibration table handling for the parametrized polynomial bandpass solutions FORTRAN solvers re-used directly from CLIC/GILDAS in AIPS++

PHAS and AMP spline polynomial solvers		
Filename	**Class or tool**	**Purpose**
[G|T]JonesPoly[.h,.cc]	[G|T]JonesSpline	A specialized [G|T]Jones matrix implementation to solve for a spline polynomial amplitude and phase corrections over time
[G|T]JonesDesc[.h,.cc], [G|T]JonesMCol[.h,.cc], [G|T]JonesTable[.h,.cc], [G|T]JonesMBuf[.h,.cc]	[G|T]JonesSplineDesc, [G|T]JonesSplineMCol, [G|T]JonesSplineTable, [G|T]JonesSplineMBuf	Calibration table handling for the parameterized spline polynomial solutions

Atmosphere model		
Filename	**Class or tool**	**Purpose**
Atmosphere[.h,.cc]	Atmosphere	Implementation of the Cernicharo (1985) ATM atmospheric transmission model
DOatmosphere[.h,.cc], atmosphere.g	atmosphere	Glish/C++ binding for the atmosphere tool

9 Appendix B: Reduction Scripts

This section contains example scripts used to fill and reduce 3 mm and 1mm data from IRAM project G067.

9.1 Filling data:

```
include 'almati2ms.g';

# Fill data from CDROM-4
almatifiller(msfile='07feb97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='07-feb-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='10feb97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='10-feb-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='25mar97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='25-mar-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='18sep97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='18-sep-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='16oct97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='16-oct-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

# Fill data from CDROM-5
almatifiller(msfile='20feb97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='20-feb-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='31mar97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='31-mar-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

almatifiller(msfile='18oct97-g067.ms',
             fitsdir='/home/broguier/aips_test2/GGTauComplete',
             pattern='18-oct-1997*.fits',
             append=F, compress=F, obsmode="CORR", chanzero="TIME_AVG");

##############################################################################
```

9.2 3-mm and-1mm data calibration:

```
#-----------------------------------------------------------------------
# 3mm & 1mm calibration script for IRAM project G067
#-----------------------------------------------------------------------

# Include relevant tools
include 'logger.g';
include 'iramcalibrater.g';
include 'os.g';

# Each observing epoch for project G067
#
epochs:=['07feb97-g067', '10feb97-g067', '20feb97-g067',
         '25mar97-g067', '31mar97-g067','18sep97-g067',
         '16oct97-g067', '18oct97-g067'];

# Bandpass calibrator used in every epoch
#
bpcal:= '0528+134';

# Phase and amplitude calibrators for each epoch (w/ flux densities)
#   (only using 0528, 0415, 2230, NRAO150)
#
```

```
#                          0528+134    0415+379    CRL618    MWC349    2230+114   NRAO150
#-------------------------------------------------------------------------------------------

apcal['07feb97-g067']:= ['0528+134', '0415+379'];
flux3['07feb97-g067']:= ['2.63Jy',   '5.81Jy',   '1.55Jy'];
flux1['07feb97-g067']:= ['2.05Jy',   '4.63Jy',   '2.20Jy'];

apcal['10feb97-g067']:= ['0528+134', '0415+379'];
flux3['10feb97-g067']:= ['2.92Jy',   '6.25Jy',   '-1Jy',   '0.96Jy'];
flux1['10feb97-g067']:= ['1.80Jy',   '4.48Jy',   '-1Jy',   '1.66Jy'];

apcal['20feb97-g067']:= ['0528+134', '0415+379'];
flux3['20feb97-g067']:= ['2.45Jy',   '5.43Jy',   '1.60Jy', '0.96Jy'];
flux1['20feb97-g067']:= ['1.50Jy',   '3.56Jy',   '2.50Jy', '1.66Jy'];

apcal['25mar97-g067']:= ['0528+134', '0415+379'];
flux3['25mar97-g067']:= ['2.30Jy',   '3.80Jy'];
flux1['25mar97-g067']:= ['1.54Jy',   '2.23Jy'];

apcal['31mar97-g067']:= ['0528+134', '0415+379'];
flux3['31mar97-g067']:= ['2.17Jy',   '3.68Jy',   '1.74Jy', '0.96Jy'];
flux1['31mar97-g067']:= ['1.54Jy',   '2.23Jy',   '3.64Jy', '1.66Jy'];

apcal['18sep97-g067']:= ['0528+134', '0415+379']; #
'2230+114','NRAO150'];
flux3['18sep97-g067']:= ['1.98Jy',   '2.80Jy',   '1.95Jy'];
flux1['18sep97-g067']:= ['1.25Jy',   '2.07Jy',   '2.99Jy'];

apcal['16oct97-g067']:= ['0528+134', '0415+379'];
flux3['16oct97-g067']:= ['1.92Jy',   '2.83Jy',   '1.66Jy', '0.96Jy'];
flux1['16oct97-g067']:= ['1.09Jy',   '2.13Jy',   '2.13Jy', '1.66Jy'];

apcal['18oct97-g067']:= ['0528+134', '0415+379']; #                '2230+114'];
flux3['18oct97-g067']:= ['1.98Jy',   '2.87Jy',   '-1Jy',   '0.96Jy'];
flux1['18oct97-g067']:= ['1.17Jy',   '2.03Jy',   '-1Jy',   '1.66Jy'];



# Loop over each observing epoch
#
for (k in 1:length(epochs)) {

    thisepoch:=epochs[k];
    filename:= spaste(thisepoch,'.ms');
    note(spaste('3mm/1mm calibration:', filename));

    # Create an iramcalibrater tool for this epoch (initcal=T)
    mycal:= iramcalibrater(filename,T);

    # Initialization: delete existing calibration tables for this epoch
    dos.remove(pathname=spaste(filename,'.3mm-LSB.visnorm'), mustexist=F);
    dos.remove(pathname=spaste(filename,'.3mm-LSB.bcal'), mustexist=F);
    dos.remove(pathname=spaste(filename,'.3mm-LSB.gcal'), mustexist=F);
    dos.remove(pathname=spaste(filename,'.1mm.visnorm'), mustexist=F);
    dos.remove(pathname=spaste(filename,'.1mm.bcal'), mustexist=F);
    dos.remove(pathname=spaste(filename,'.1mm.gcal'), mustexist=F);

    # Phase corrected data selectin (CLIC/PHCOR)
    mycal.phcor(F);

    # Bandpass calibration (CLIC/RF)
    mycal.rf(bpcal, freqgrp='3mm-LSB', bpnorm=F, visnorm=T, gibb=2, drop=0);

    # Phase calibration (CLIC/PHAS)
    mycal.phase(apcal[thisepoch], freqgrp='3mm-LSB');

    # Establish the absolute flux density scale (CLIC/FLUX)
    fluxes:=mycal.flux(fieldnames=apcal[thisepoch], fixed=flux3[thisepoch],
                       freqgrp='3mm-LSB', gibb=0, drop=0, plot=F);

    # Amplitude calibration (CLIC/AMP)
    mycal.amp(fieldnames=apcal[thisepoch], freqgrp='3mm-LSB');

    # Bandpass calibration (CLIC/RF)
    mycal.rf(bpcal, freqgrp='1mm',     bpnorm=F, visnorm=T, gibb=2, drop=0);
```

```
    # Phase calibration (CLIC/PHAS)
    mycal.phase(apcal[thisepoch], freqgrp='1mm');

    # Establish the absolute flux density scale (CLIC/FLUX)
    fluxes:=mycal.flux(fieldnames=apcal[thisepoch], fixed=flux1[thisepoch],
                       freqgrp='1mm', gibb=0, drop=0, plot=F);

    # Amplitude calibration (CLIC/AMP)
    mycal.amp(fieldnames=apcal[thisepoch], freqgrp='1mm');
    # Close the iramcalibrater tool for this epoch
    mycal.done();
};

#--------------------------------------------------------------------------
```

9.3 3-mm calibrated data concatenation:

```
#--------------------------------------------------------------------------
# 3mm calibrated data concatenation for IRAM project G067
#--------------------------------------------------------------------------

# Include relevant tools
include 'logger.g';
include 'iramcalibrater.g';
include 'flagger.g';
include 'os.g';

# Datasets for each observing epoch for IRAM project G067
#
filenames:=['07feb97-g067.ms', '10feb97-g067.ms', '20feb97-g067.ms',
        '25mar97-g067.ms', '31mar97-g067.ms','18sep97-g067.ms',
        '16oct97-g067.ms', '18oct97-g067.ms'];

# Field name to be imaged
#
field:= 'GG_TAU';

# Calibrated output 64-, and 256-channel data on GG_TAU
#
ggtau64ms:= 'ggtau_3mm_64.ms';
ggtau256ms:= 'ggtau_3mm_256.ms';
dos.remove(pathname=ggtau64ms, mustexist=F);
dos.remove(pathname=ggtau256ms, mustexist=F);
fileopt:= 'new';

# Loop over each observing epoch
#
for (k in 1:length(filenames)) {

    filename:= filenames[k];
    note(spaste("3mm calibrated data concatenation: ", filename));

    # Create an iramcalibrater tool for this observing epoch
    mycal:= iramcalibrater(filename);

    # Extract, and concatenate, calibrated 64- and 256-channel
    # data for the field to be imaged (GG Tauri)
    mycal.uvt(fieldname=field, spwid=3, filename=ggtau64ms, option=fileopt);
    mycal.uvt(fieldname=field, spwid=7, filename=ggtau256ms, option=fileopt);
    fileopt:= 'append';

    # Close the iramcalibrater tool for this epoch
    mycal.done();
};

# Flag the two central Gibbs channels in each dataset prior to imaging
# 64-channel
myflag:= flagger(ggtau64ms);
myflag.setchan([32,33]);
myflag.query('TIME > 0');
myflag.done();

#--------------------------------------------------------------------------
```

9.4 3-mm continuum imaging:

```
include 'imager.g';
myimager:=imager(filename="ggtau_3mm_64.ms");
myimager.setdata(mode="channel" , nchan=58, start=3, spwid=[1], fieldid=1);
myimager.setimage(nx=256, ny=256, cellx='0.22arcsec', celly='0.22arcsec',
         stokes="I" , doshift=F ,mode="mfs");

myimager.weight(type='briggs', rmode='norm', robust=0.45);

myimager.make('cont3mm');

myimager.setbeam(bmin='1.25arcsec', bmaj='1.99arcsec', bpa='20deg')

myimager.clean(algorithm="hogbom" , niter=300, gain=0.1,
           threshold=[value=0.0, unit="mJy" ],
           displayprogress=F, model="cont3mm" , fixed=F,
           complist='', mask='', image="cont3mm.restored" ,
           residual="cont3mm.residual" , interactive=F, npercycle=100);

myimager.done();
```

9.5 3-mm line imaging:

```
include 'imager.g';
myimager:=imager(filename="ggtau_3mm_256.ms");

myimager.setdata(mode="channel" , nchan=48, start=105, spwid=1, fieldid=1);

myimager.setimage(nx=256, ny=256, cellx='0.22arcsec', celly='0.22arcsec',
         stokes="I" , doshift=F ,mode="channel" ,
         nchan=48, start=105, step=1, spwid=1,
         fieldid=1, facets=1);

myimager.make(image="line3mm" );

myimager.setbeam(bmin='1.81arcsec', bmaj='2.41arcsec', bpa='-180deg');

myimager.clean(algorithm="hogbom" , niter=400, gain=0.1,
           threshold=[value=1.0, unit="mJy" ],
           displayprogress=F, model="line3mm_flag" , fixed=F,
           complist='', mask='', image="line3mm.restored" ,
           residual="line3mm.residual" , interactive=F, npercycle=100);

myimager.done();
```

9.6 1-mm calibrated data concatenation:

```
#--------------------------------------------------------------------------
# 1mm calibrated data concatenation for IRAM project G067
#--------------------------------------------------------------------------

# Include relevant tools
include 'logger.g';
include 'iramcalibrater.g';
include 'flagger.g';
include 'os.g';

# Datasets for each observing epoch for IRAM project G067
#
filenames:=['10feb97-g067.ms', '20feb97-g067.ms',
        '25mar97-g067.ms', '31mar97-g067.ms','18sep97-g067.ms',
        '16oct97-g067.ms', '18oct97-g067.ms','07feb97-g067.ms' ];

# Field name to be imaged
#
```

```
field:= 'GG_TAU';

# Calibrated output 64-, and 256-channel data on GG_TAU
#
ggtau64ms:= 'ggtau_1mm_64.ms';
ggtau256ams:= 'ggtau_1mm_256_a.ms';
ggtau256bms:= 'ggtau_1mm_256_b.ms';
dos.remove(pathname=ggtau64ms, mustexist=F);
dos.remove(pathname=ggtau256ams, mustexist=F);
dos.remove(pathname=ggtau256bms, mustexist=F);
fileopt:= 'new';

# Loop over each observing epoch
#
for (k in 1:length(filenames)) {

    filename:= filenames[k];
    note(spaste("1mm calibrated data concatenation: ", filename));

     # Create an iramcalibrater tool for this observing epoch
    mycal:= iramcalibrater(filename);

    # Extract, and concatenate, calibrated 64- and 256-channel
     # data for the field to be imaged (GG Tauri)
     mycal.uvt(fieldname=field, spwid=11, filename=ggtau64ms, option=fileopt);
     mycal.uvt(fieldname=field, spwid=23, filename=ggtau256ams, option=fileopt);
    if(filename=='07feb97-g067.ms')
       mycal.uvt(fieldname=field, spwid=23, filename=ggtau256bms, option=fileopt);
    else
       mycal.uvt(fieldname=field, spwid=24, filename=ggtau256bms, option=fileopt);
    fileopt:= 'append';
    mycal.uvt(fieldname=field, spwid=12, filename=ggtau64ms, option=fileopt);
     mycal.uvt(fieldname=field, spwid=15, filename=ggtau64ms, option=fileopt);
     mycal.uvt(fieldname=field, spwid=16, filename=ggtau64ms, option=fileopt);
     mycal.uvt(fieldname=field, spwid=19, filename=ggtau64ms, option=fileopt);
     mycal.uvt(fieldname=field, spwid=20, filename=ggtau64ms, option=fileopt);

     # Close the iramcalibrater tool for this epoch
    mycal.done();
};

# Flag the two central Gibbs channels in each dataset prior to imaging
# 64-channel
myflag:= flagger(ggtau64ms);
myflag.setchan([32,33]);
myflag.query('TIME > 0');
myflag.done();

# 256-channel [a] and [b]
#myflag:= flagger(ggtau256ams);
#myflag.setchan([128,129]);
#myflag.query('TIME > 0');
#myflag.done();

#myflag:= flagger(ggtau256bms);
#myflag.setchan([128,129]);
#myflag.query('TIME > 0');
#myflag.done();
#------------------------------------------------------------------------
```

9.7 1-mm continuum imaging:

```
include 'imager.g';
myimager:=imager(filename="ggtau_1mm_64.ms");

myimager.setdata(mode="channel" , nchan=58, start=3, spwid=[1:9], fieldid=1);

myimager.setimage(nx=256, ny=256, cellx='0.08arcsec', celly='0.08arcsec',
         stokes="I" , doshift=F ,mode="mfs" ,
         nchan=58, start=4, step=5, spwid=[1:9]);

myimager.weight(type='briggs', rmode='norm', robust=0.45);
```

```
myimager.make('cont1mm');

myimager.setbeam(bmin='0.59arcsec', bmaj='0.85arcsec', bpa='-156deg');

myimager.clean(algorithm="hogbom" , niter=1000, gain=0.1,
           threshold=[value=0.0, unit="mJy" ],
           displayprogress=F, model="cont1mm" , fixed=F,
           complist='', mask='', image="cont1mm.restored" ,
           residual="cont1mm.residual" , interactive=F, npercycle=100);
myimager.done();
```

9.8 1-mm line imaging:

```
include 'imager.g';

myimager:=imager(filename="ggtau_1mm_256_b.ms");

myimager.setdata(mode="channel" , nchan=48, start=105, spwid=1, fieldid=1);

myimager.setimage(nx=256, ny=256, cellx='0.08arcsec', celly='0.08arcsec',
          stokes="I" , doshift=F ,mode="channel" ,
          nchan=48, start=105, step=1, spwid=1,
          fieldid=1, facets=1);

#myimager.weight('natural');

myimager.make('line1mm');

myimager.setbeam(bmin='0.82arcsec', bmaj='1.04arcsec', bpa='-178deg')

myimager.clean(algorithm="hogbom" , niter=1000, gain=0.1,
           threshold=[value=0.5, unit="mJy" ],
           displayprogress=F, model="line1mm" , fixed=F,
           complist='', mask='', image="line1mm.restored" ,
           residual="line1mm.residual" , interactive=F, npercycle=100);

myimager.done();

include 'image.g'
myim:=image('line1mm.restored')
cs:=myim.coordsys()
cs.setrestfrequency(value=230.53797e9);
myim.setcoordsys(cs);
myim.done();
```