NRAO AIPS++ Users Group	NAUG-SW-xxx Revision:1.6.5 2003-02-25 *VLA Evaluation* S. Myers, D. Shepherd

NAUG Evaluation of the AIPS++ Package for VLA Processing

VLA Evaluation

NRAO AIPS++ Users Group (NAUG)

Keywords: Requirements, Offline, Software, Science, Calibration

Author Signature: S. Myers,D. Shepherd	Date: 2003-02-25
Approved by: S. Myers	Signature:
Institute: NRAO	Date:
Released by:	Signature:
Institute:	Date:

Change Record

Revision	Date	Author	Section/ Page affected	Remarks
1.0	2002-10-25	Myers	All	template based on VLA
1.1	2002-10-28	Brogan/Shepherd	All	flesh out reqs
1.2	2002-11-01	Myers/Brogan	All	new level scheme
1.3	2002-11-04	Shepherd	All	modify explanations & reqs
1.4	2002-11-08	Shepherd	All	reorganize & modify reqs for VLA
1.5	2002-12-13	Shepherd/All	All	input changes from NAUG members
1.6	2002-12-18	Myers/Shepherd	All	grade/severity change, sec3.3
1.6.1	2003-01-08	Myers	All	new Tex macros
1.6.2	2003-01-21	Myers	All	appendix
1.6.3	2003-02-16	Shepherd	All	John's 1st version + Ed's sect
1.6.4	2003-02-17	Shepherd	All	John's 2st version + Ed's sect
1.6.5	2003-02-25	Myers	2	Steve's section 2 grades
				Grades added:
	2003-01-09	Shepherd		Version 2: Imaging & Doc
	2003-01-09	Taylor		Version 1: Pol Cal & editing
	2003-01-09	Brisken		Version 2: Astrometry
	2003-01-09	Bastian/Shepherd		Version 2: Solar obs
	2003-01-10	Chandler		Version 2: Apriori
	2003-01-10	Brisken		Version 2: Analysis
	2003-01-10	Chandler		Version 3: Ops
	2003-01-13	Myers		Version 1: Data Handling
	2003-01-20	Owen		Appendix: Wide-field considerations
	2003-01-24	Fomalont		Version 1: mosaicing,wide-field
	2003-02-20	Brogen/Brisken/Hibbard		Version 3: Data analysis/Visualization
	2003-01-25	Hibbard		Version 2: flagging
	2003-02-25	Myers		2. Data handling, import/export
				No comments yet:
	2003-02-16	Fomalont/Bhatnagar		1. Operations: Benchmarking
	2003-02-16	Claussen		2. Data examination
	2003-02-16	Butler		5. Data analysis: uv fitting
	2003-02-16	Claussen		7. Special Features: simulations
	2003-02-16	Butler		7. Special Features: solar system

`$Id: naug_vla_eval.tex,v1.6.5,2003/02/25 myers Exp myers $`

Contents

General Considerations

This document provides a template for listing the results of the evaluation of the AIPS++ software package for the purposes of reducing VLA data.

A. There are three categories under which each requirement will be graded:

Functionality — whether the tool works, accuracy
Useability — ease-of-use, speed
Documentation — clearness, accuracy

These three should be evaluated independently for each requirement. Note that the *Documentation* grade should encompass the entries in the User Reference Manual, the Getting Started and Getting Results documents (the VLA "Cookbook" chapter of Getting Results in particular), and the online help given in the Tool GUIs (particularly the page that the Help button takes you to).

B. The requirements are assigned *Levels* of Functionality at which they can be fulfilled:

Level 1 = basic operations
Level 2 = advanced operations

Level 1 items are part of the path for the basic VLA data processing that applies to the mainstream of VLA observations and includes single-field continuum and spectral line observations, with emission within the half-power field-of-view, including (linear) polarimetry, at bands L through Q. It is intended that Level 1 functionality must be adequately handled by the Package and work with high efficiency. We estimate that these items cover 90% of the VLA user base programs.

Level 2 items are part of the more advanced processing path, and include special and more challenging modes such as mosaicing, wide-field imaging, high dynamic range imaging and spectroscopy, astrometry, simulations, and observations in the 4-meter and P bands, pulsar and solar observing. This category also includes service operations such as holography and baseline determination as well as considerations for off-line pipeline processing.

Ideally, individual requirements should be either Level 1 or Level 2, for example with the main reqirement and some sub-requirements being Level 1, and some subrequirements Level 2. However, it will often be the case that the same requirements will have both Level 1 and Level 2 functionality, and this should be made clear in the comments.

C. We adopt a descriptive grading scheme and grade codes stating how well the Package fulfills a given requirement:

A = adequate as it stands
E = adequate, but some enhancement is desired
I = inadequate, needs further development to meet requirements
N = not available in AIPS++ at all
U = category not applicable to the requirement, or unable to grade for some reason

For items deemed inadequate (I), missing (N) or Enhancement Desired (E) the reasons for this should be listed in the *Comment* following the grade. Note that a subjective choice will have to be made between items at are adequate but could use further improvement (E), and those deemed (I) or (N). The Level of the requirement should guide this assessment keeping in mind that Level 1 items will impact the average user to a greater extent.

In the grades for *Functionality*, the Level should be attached to the grade, e.g. A1, A2, E1, E2, I1, I2, N1, N2.

For items deemed inadequate (I) or missing (N), the severity of defect should be noted: *low*, *med*, *high*. These should be appended to the grade, e.g. I1/high, N2/med.

D. For each requirement, the tool or function in the AIPS++ Package fulfilling that requirement is to be listed in the *Comment* following the grade. This will allow interested parties to review the evaluation process and identify which tools should be targeted for improvement or be accepted as complete.

E. Based on the evaluation and the Level of the requirements, a list of the I and E items that need work will be constructed. The goal will be to fulfill Level 1 functionality (grade A1) as soon as possible, and to determine a realistic schedule to complete Level 2 functionality (grade A2) along with adequate documentation and useability.

F. The evaluation should reflect the best possible assessment of the AIPS++ package as of Jan. 15, 2003.

1 Operations
Ed, Sanjay (benchmarking), Claire (installation & interface)

This section examines operational issues for both beginning and advanced users. It includes installation and start up of the Package, the user interface, and run-time performance issues.

1.1 Installation

1.1-R1 The Package shall be installable at the users home institution. Installation of the Package by an end user must be straightforward, preferably without special (e.g. root) user permission. It shall be portable to **supported platforms designated by the VLA Project**, including systems without network connections and laptops.

Levels: 1

Functionality: A1

Useability: E1

Documentation: E1

Comments: CJC: It is straightforward to install the release version from CD, although there are some inconsistencies between information in the release version and the web pages concerning the build number for release version 1.7. Also, the instructions on specifying the AIPS++ root directory in the installation script for version 1.7 are not clear. The instructions for installing the ISO image if it is downloaded from the web include the instructions to burn the ISO image to a CD using xcdroast, but xcdroast requires root privilege to be set up initially, at least on NRAO machines.

1.1-R2 It shall be straightforward to update an installation of the Package to the latest "stable" version without special user permission.

Levels: 2

Functionality: A2

Useability: A2

Documentation: A2

Comments: CJC: Updating a release version to the current intermediate stable version is straightforward using the function "aupdate."

1.2 Initialization

There should be an easy way for the user to figure out the best operational configuration for their machine (e.g. memory usage, or cache location). This could be done with an initialization GUI or a Glish interactive session which steps the user through the initialization procedure and writes an appropriate .aipsrc file. Applicable documentation should include Getting Started and documentation on specific customization options (e.g. use of an .aipsrc file).

1.2-R1 Initial startup of the Package, once installed, should be straightforward for the user to optimize for their specific machine to minimize run time.

Levels: 1,2

Functionality: A1,A2

Useability: E1,E2

Documentation: I1/high,I2/high

Comments: CJC: Level 1: The setupwizard successfully creates a .aipsrc file and enters initial setup values. Its useability could be enhanced by (1) having it run automatically the first time aips++ is invoked, and (2) having it accessible somewhere from the tool manager or other GUI, rather than having to include it from the command line. Documentation on how the basic setup parameters affect performance needs to be included in Chapter 1 of Getting Started. Level 2: In order to minimize run time information on the system memory and other paramaters should also be included in the setupwizard. Documentation on how **all** the setup parameters affect performance is needed, both in Chapter 1 of Getting Started, and in the setupwizard. JEH: my reading of this is that it should be possible to *optimize* performance; i.e. that aips++ would look at your processor and available memory, and use that information to optimize how fast it runs. So I would grade this as N1/high.

1.2-R2 It should be possible to reconfigure run-time options manually, e.g., by using a favourite editor.

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: CJC: A note to the effect that this is possible should be included in the documentation.

1.2-R3 It should be possible for the user to configure certain default options at startup, such as browser, level of help, and default viewer or pgplotter options.

Levels: 1

Functionality: N1/med

Useability: N1

Documentation: N1

Comments: JEH: I understand that these options can be saved individually, but some should be specifiable once and for all at initialization.

1.3 General User Interface Requirements

Applicable tools for the general interface include: `tool manager`, `scripter`, and GUI and Glish interfaces. Applicable documentation should include Getting Started.

1.3-R1 The user must be able to choose from a variety of interface styles, including:

1.3-R1.1 A Command Line Interface (CLI) must be provided, with access via both an interactive input and via script.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: Note from ALMA audit: "The glish interface is a suitable CLI environment."

1.3-R1.2 A Graphical User Interface (GUI) must be provided for interactive processing.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC: Grade mainly just reflects the fact that a GUI exists. JEH: I would grade the GUI as A1 functionality but E1/high on useability. As Bryan pointed out nicely in his evaluation of "Getting Started", the GUI is really an interface to execute aips++ objects, not an intuitive users interface to a logical reduction path. That latter is desperately needed. Documentation is

E1: while there is documentation describing almost all modules/tools/functions in the a general sense, there is very little documentation describing just what algorithms are being used (e.g., for averaging operations, for compression); adequate input parameter descriptions are often missing, especially "Allowed" values, which often only gives the type of entry that is accepted (integer, string, etc), but not the actual values which are recognized. A comprehensive index to the documentation (including all functions and all input parameters and covering URM, GR1-3, GS) would really help.

1.3-R2 The user shall be able to interact with the host operating system with command sequences invoked from the UI.

Levels: 1,2

Functionality: A1,E2

Useability: A1,E2

Documentation: A1,E2

Comments: CJC: Level 1: The misc.os tool is adequate in functionality and useability for basic commands. Level 2: The misc.os tool is very limited in its options, and makes it unusable for advanced interactions with the host operating system using the GUI. Shell and misc.shellcmd are adequate for the CLI.

1.3-R3 Multitasking for all interfaces shall be available where appropriate. It must be possible to run one or more long-running calculations in the background. While background tasks are running normal interactive activities must be possible.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: Note from ALMA audit: "Multiple tools and windows can be run concurrently, as well as multiple instances of aips++ (memory permitting)."

1.4 Graphical User Interface (GUI)

Note: The GUI is intended to be the primary interactive mode for users, especially neophyte users, and thus should be tailored for clarity and ease of use. Use of the Package GUIs should be a pleasurable, not a frustrating, experience!

1.4-R1 The GUI shall provide real-time feedback via standard compact displays:

1.4-R1.1 Window updating must be fast (less than 0.1s on same host).
Levels: 1
Functionality: U1
Useability: U1
Documentation: U1
Comments: CJC: I do not know how to quantify this. Note from ALMA audit: "Benchmarking at this time is not ideal. However, it does appear that updating of windows (when GUIs are constructed in glish) can be significantly slower than this. This should be quantified within the next year." JEH: Not sure what this means. rubberbanding and back/front are fast. Updating GUI entries is not. e.g., clicking on a module can take several seconds before the available tools appear.

1.4-R1.2 Windows shall not take up excessive screen space, with full GUI controls visible on a window one-third the size of a standard view surface (or approximately 800 × 800 pixels).
Levels: 1
Functionality: A1
Useability: E1
Documentation: N1/med
Comments: Note from ALMA audit: "Most GUIs are compact enough, though some can be large (e.g. the viewer Adjust Panel can be too long (section Axis label properties) for the 1024 × 780 screen of a typical laptop), and some tools generate many windows. Some of these are being changed. Questions like this should be addressed by an interface focus group (with ALMA participation)." CJC: Grade is based on ALMA audit comment. JEH: documentation grade should be U?

1.4-R1.3 Users shall have the choice of cascading windows or re-use of a single window for new operations.
Levels: 2
Functionality: N1/low
Useability: N1
Documentation: N1
Comments: Note from ALMA audit: "The manner of windowing is not under user control." CJC: Grade is based on ALMA audit comment.

1.4-R1.4 Moving and resizing of all windows must be available, robust, and easy.
Levels: 1
Functionality: E1
Useability: E1
Documentation: E1
Comments: CJC: Windows can be moved easily; manual resizing seems to turn off the automatic resizing, and can result in information falling off the bottom of subsequent windows. (Note from ALMA audit: "this has been submitted as a defect, and is being worked upon.") There is a statement in Getting Started to the effect that no aips++ window should be resized, but no explanation of why this is the case, or the possible results of manual resizing, is given beyond "you may get strange results in subsequent operations."

1.4-R1.5 There shall be a master control GUI for process control which keeps track of sub-windows and tools.
Levels: 1
Functionality: E1
Useability: E1
Documentation: E1
Comments: Note from ALMA audit: "There is a Tool Manager GUI which keeps track of tools, but not of windows per se (must use the OS window manager). Could use some improvement." JEH: I don't know what would be wanted above and beyond "Tools in Use". So I would grade it as A1. E2 could be included if someone specifying what they want beyond what is there.

1.4-R2 It must be easy to run GUIs remotely from the host machine (e.g. via X displays).

Levels: 1

Functionality: A1

Useability: A1

Documentation: N1/low

Comments: Note from ALMA audit: "This seems straightforward."

1.4-R3 There shall be a master GUI from which the user has access to, and can graphically launch, all available tools and functions.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: CJC: The Tool Manager is adequate. JEH: setupwizard is not available via the Tool Manager. Haven't found anything else missing.

1.4-R3.1 There shall be the facility within the master GUI to locate the desired tool or function based upon context (e.g. by scientific or data reduction purpose).
Levels: 1,2
Functionality: A1,A2
Useability: A1,I1/high
Documentation: A1,I2
Comments: CJC: Level 1: With the VLA Cookbook the useability and documentation for finding tools in the Tool Manager is adequate. Level 2: Finding tools and functions in the toolmanager for advanced operations not covered in the VLA Cookbook is not intuitive. The documentation needs significant improvement. JEH: A comprehensive index of GS, GRvol1-3, & URM is really needed. At the minimum, all functions and all inputs should have entries.

1.4-R4 The use of the GUI shall not entail an excessive learning curve. Average users, with experience with the current generation of packages (e.g. AIPS, GILDAS, IRAF, MIRIAD) shall be able to become proficient in GUI use in a timescale of approximately 12 hours dedicated use, and truly neophyte users (e.g. graduate students) should be able to reach proficiency with an investment not exceeding 40 hours of dedicated use.

Levels: 1,2

Functionality: A1,E2

Useability: E1,E2

Documentation: A1,E2

Comments: Note from ALMA audit: "The intent of this requirement is unclear as stated — as written it seems to refer to the mechanics of the GUI operation rather than ease of use following astronomical tasks which was probably the intent of the writers. From a mechanical standpoint, the use of the GUIs from the Tool Manager is straightforward, and the viewer operation is also easy to learn. In hindsight, the learning time requirements should refer to astronomical processing specifically, rather than GUI mechanics, and a new requirement should be written as such, though in practice these issues are covered under other requirments...". CJC: Level 1: Use of the GUIs themselves is in general straightforward, although the structuring of information in some tools could be enhanced (e.g., re-ordering setup functions in the imager tool, and highlighting entries that are actually required to be set before running a function). The documentation in the VLA Cookbook adequately describes the use of GUIs. Level 2: For VLA reduction the learning curve could be significantly improved by combining all the essential tools and functions in a single package. For advanced operations documentation similar to the VLA Cookbook is needed. Since the GUI for a tool or function is perhaps the easiest way for a user to work out what inputs are needed to run that process, it would also be useful for users if generic tools for all functions could be created without the need to supply the name of a measurement set. It should in all cases be possible to specify the measurement set name from within the tool itself.

1.4-R5 GUI-based tools shall be available for reduction of data taken in all standard VLA observing modes.

Levels: 1,2

Functionality: I1/high,N2/med

Useability: E1,N2

Documentation: A1,N2

Comments: CJC: Level 1: Currently GUI-based tools are available for the reduction of most basic VLA observing modes supported in some form by aips++. Not all modes are supported, however. For example, it is not currently possible to reduce high frequency VLA data. The documentation in the VLA Cookbook is adequate for those modes currently available in aips++. Level 2: The reduction of more advanced VLA observing modes is currently not possible in aips++. JEH: I believe GUIs are *available* for all supported reduction paths, but not all reduction paths work. There are many standard operations that users can do only via scripts and not GUIs, e.g., continuum subtraction. I think GUI interfaces for constructing LEL and TaQL expressions are needed. Overall, I don't know how to grade this.

1.4-R6 The look and feel of the GUI shall be uniform throughout the entire package.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: Note from ALMA audit: "This is currently the case, as there are few custom GUIs and those are acceptable." JEH: A notable exception is the msplot viewer option, which has a different adjust panel than the general viewer when viewing an ms.

1.4-R7 The look and feel of the GUI must be acceptable to both novice and more advanced users. The GUI mode might be customizable (perhaps through menu selection for "novice" or "advanced" mode) or a different simpler set of GUI tools might be available for beginners. The GUI features for beginner mode shall include:

1.4-R7.1 built-in help facility with access to novice-oriented help documents (e.g. sections in the cookbook).
Levels: 1,2
Functionality: A1,E2
Useability: A1,E2
Documentation: E1,I2/med
Comments: Note from ALMA audit: "There are buttons and wrench menu items available for help which drive a browser to the (hopefully) appropriate entry." CJC: Level 1: Some form of help is generally available through the help button. The help documentation itself, however, is not always very helpful. Pointers to the VLA Cookbook would be more useful to novice users. Level 2: It should be possible to specify the type of browser to which help documents will be sent on selecting "function help." The help itself for tools such as the imager should include examples. JEH: I read this as saying that you should be able to go directly from the GUI to the cookbook documentation, and this is not there. There is no option to access anything other than the URM from the GUI, so I don't think this can be considered A1. Maybe it should be a L2 requirement only.

1.4-R7.2 sensible defaulting of values for parameters, with guidance for user choices where needed.
Levels: 1
Functionality: A1
Useability: E1
Documentation: I1/med
Comments: CJC: At present it is not always clear which parameters are absolutely required in order to run a function or tool. Furthermore, on returning to a function after it has already been used once, values remain displayed in the sections where parameters are specified. It is not clear to a novice user that the function needs to be run again. Functions and parameters that are required in order to run a function should perhaps be displayed in a different colour. The VLA Cookbook includes some examples of user choices, but more documentation is needed

for functions not included in the Cookbook. JEH: I would grade this as A1 – there are useful defaults. E2 – would like required parameters more clearly spelled out. Useability is tricky – I find the many "set something" functions to be very confusing (e.g., in flagger, calibrator, etc). These really are not functions in the way the user is used to thinking of them – they are inputs for other functions. I would prefer to see all these functions eliminated and incorporated in the appropriate master function (flag, calibrate). But this is a big issue for general GUI operation, so I am not sure it belongs here.

1.4-R7.3 integrated functionality built around common data analysis tasks (e.g. single-field spectroscopic observations with fast switching, mosaicing in continuum mode, snapshot observations of a large number of targets in single-field continuum mode)
Levels: 2
Functionality: I2/med
Useability: I2
Documentation: I2
Comments: Note from ALMA audit: "There are few integrated tools at this time available from the GUI. Many of the early attempts at wizards or integrated tools have been abandoned or have not been updated for a long time. Some progress has been made in building interactivity in some tools (e.g. interactive=T in imager.clean)." CJC: Grading is based on ALMA audit comments.

1.4-R8 It is shall be easy for users to develop and include their own custom GUIs in the Package.

Levels: 1,2

Functionality: A1,E2

Useability: A1,E2

Documentation: A1,E2

Comments: Note from ALMA audit: "There is a substantial set of GUI building tools (the guiutils module)." CJC: Level 1: Access to the relevant documentation is available from the web. There is also "toolgui" for customizing tool GUIs. I have not tried to make a GUI myself, although it seems relatively straightforward. Level 2: Access to the guiutils module and toolgui is via the CLI mode only. It would be great if there were a GUI for making GUIs...

1.4-R9 Functionality of the CLI mode should be available in GUI mode where applicable.

Levels: 1,2

Functionality: E1,I2/med

Useability: E1,I2

Documentation: E1,I2

Comments: CJC: Not all CLI functionality is available in GUI mode at present.

1.5 Command Line Interface (CLI)

Note: the CLI is the primary mode for automatic reduction for VLA, and it is anticipated that there will be a suite of "standard" scripts developed to help users in data reduction tasks. Thus it is important that the Package support all of its critical modes in the CLI.

1.5-R1 The interface must have the facility to read in command files for batch processing of a sequence of CLI commands.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: CJC: Documentation on how to read in glish scripts is adequate.

1.5-R2 The CLI shall have command-line recall and editing.

Levels: 1

Functionality: A1

Useability: A1

Documentation: I1/low

Comments: Note from ALMA audit: "There is command-line recall (with editing) in glish." CJC: There is no documentation on command-line recall and editing, although users familiar with unix/linux systems probably wouldn't need any.

1.5-R3 The CLI shall have name completion where appropriate.

Levels: 1

Functionality: N1/med

Useability: N1

Documentation: N1/low

Comments: Note from ALMA audit: "...there is no name completion, probably due to the general nature of glish syntax. Given the ability to introduce and overload variable names, it might be argued to be never appropriate, but should be investigated." CJC: Name completion is needed, for measurement set names, aips++ function calls, etc.

1.5-R4 All functionality of the GUI must also be available in CLI mode (although possibly with loss of simplicity in instances where the graphical selection is important).

Levels: 1,2

Functionality: A1,A2

Useability: A1,A2

Documentation: A1,E2

Comments: Note from ALMA audit: "Nearly all GUI tools are direct implementations of the CLI equivalents. There are a few GUIs (e.g. msplot, viewer) that have no CLI level equivalents (invocation from the CLI brings up the GUI), but these are necessary." CJC: Level 1: Finding equivalent GUI functionality in CLI mode for basic functions is relatively easy using the VLA Cookbook. Level 2: Documentation similar to the VLA Cookbook is also needed for advanced functions.

1.5-R5 The CLI shall be usable remotely over low-speed (14400 baud) modem lines or network connections, with ASCII terminal emulation.

Levels: 1

Functionality: A1

Useability: A1

Documentation: N1/low

Comments: Note from ALMA audit: "This works but is infrequently used." CJC: Grades are based on ALMA audit comment.

1.5-R6 A menu or other means of displaying and editing function input parameter values in CLI mode is desirable.

Levels: 1,2

Functionality: E1,N2/low

Useability: E1,N2

Documentation: E1,N2

Comments: CJC: Level 1: Defined here as the display of current input parameters and their values. Parameter names can be found using help('package.module.tool.function'). Parameter values can only be set in the command executing the function, or in scripts. A significant enhancement might be to include command line adverbs for setting (and displaying) individual parameter values prior to executing a function. Level 2: Defined here as the ability to edit parameter values. Since parameter values are currently not remembered from one execution of a function to the next in CLI mode, the editing of parameter values in aips++ amounts to being able to edit the command executing a function, or inputs to a script. This is straightforward using command line editing, which is covered by a separate requirement.

1.6 Interface Programming, Parameter Passing and Feedback

1.6-R1 The UI must have basic programming facilities such as:

1.6-R1.1 variable assignment and evaluation
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.2 array handling
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.3 conditional statements
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.4 control loops
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.5 string manipulation
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.6 user-defined functions and procedures with argument or parameter passing
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.7 process control, interrupts, error handling
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC: Note from ALMA audit: "A strong point of glish. Used by Tony Willis for ACSIS (and in fact by all the GUIs and tools)."

1.6-R1.8 standard mathematical operations and functions
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC

1.6-R1.9 efficient special vector and matrix operations
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC: Note from ALMA audit: "Glish handling of vectors and arrays is efficient programming-wise, though benchmarking will be necessary to evaluate the speed."

1.6-R1.10 user-defined data structures
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CJC: Note from ALMA audit: "Glish records are somewhat arbitrary, but are probably adequate for what users would want."

1.6-R2 Input parameter and syntax checking shall be done when a function is called or parsed with reporting of incorrect, suspicious or dangerous choices before execution where possible.

Levels: 1

Functionality: E1

Useability: I1/high

Documentation: I1/high

Comments: Note from ALMA audit: "The parameter checking is inconsistent across tools, and seems to depend on whether the programmer bothered to implement this or not. There does seem to be basic type checking (which can give confusing results for more complicated measures), but few user-friendly warnings for suspicious or dangerous choices. There is room for improvement here, though many cases have been submitted as bugs and are being fixed." CJC: Grade is based on ALMA audit comments. JEH: Illegal inputs are reported from the GUI with the red "X", but programs often allow you to run with bad inputs, which then choke on upon execution.

1.6-R3 Parameters shall be passable between applications in as transparent a manner as possible. However, global variables should not be the default, unless designated specifically by the user-programmer.

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: Note from ALMA audit: "All glish applications are function calls with arguments. By default, function variables have function scope only, but can be defined to have local, wider, or global

scope. What this requirement means for GUIs is unclear, though cut and paste of values seems to suffice." CJC: Grade is based on ALMA audit comments. JEH: For the GUI the "Command Arguments lastsave" line at the bottom serves this purpose. Documentation on this is poor. CJC: Note also that the "Command Arguments lastsave" will only work in this context as long as variable names are consistent from one application to the next (see next requirement).

1.6-R4 Application variables shall be named consistently and as clearly as possible indicating their intended use using astronomical terms where appropriate.

Levels: 1

Functionality: E1

Useability: E1

Documentation: E1

Comments: Note from ALMA audit: "For the most part, variables have sensible names. However, there are cases where naming between tools is not consistent, in particular the way in which channels, etc. are selected in tools such as ms, flagger, imager. Many of these have been noted and are being worked on." CJC: Grade is based on ALMA audit comments. JEH: I found some cases of inconsistent naming between the tool and the documentation. e.g., msplot flagfile, which is properly listed in the constructor help, but referred to in the "background on editing" section as the flag column table. Presumably there are not many instances of this.

1.6-R5 Parameter inputs to tools shall be stored for later recall:

1.6-R5.1 Tool inputs shall be saved on closure, and reinstated on the next instance of the tool.
Levels: 1,2
Functionality: A1,E2
Useability: A1,E2
Documentation: A1,E2
Comments: Note from ALMA audit: "The GUI tools accessible from *Tool Manager* menus remember previous inputs, though in script mode inputs are not remembered and parameters default to the original defaults." CJC: Grade is based on ALMA audit comments.

1.6-R5.2 It shall be possible for the user to save the state of the parameters for the entire package, as well as for individual tools, as a named set (e.g. SAVE/GET in AIPS), and to recall these when desired.
Levels: 1
Functionality: I1/med
Useability: I1
Documentation: I1
Comments: Note from ALMA audit: "The Tool Manager level GUI parameters are saved in a cache (see above), and some GUI tools (e.g. viewer) allow saving of internal parameters, but this is not consistently implemented." CJC: Grade is based on ALMA audit comments.

1.7 Error reporting and handling

1.7-R1 Error reporting and handling shall be user-understandable and non-destructive at all levels in the Package:

1.7-R1.1 Error reporting messages shall be written for end users, not programmers.
Levels: 1
Functionality: I1/high
Useability: I1
Documentation: I1

Comments: Note from ALMA audit: "Many error messages, especially those generated by glish failures such as glisk-tk widgets, are unduly cryptic and are alarming to users." CJC: Grade is based on ALMA audit comments.

1.7-R1.2 There must be provision for job control such as interrupt and abort.
Levels: 1
Functionality: E1
Useability: E1
Documentation: E1
Comments: Note from ALMA audit: "GUI tools have Abort buttons and cntl-c gives the option to abort from glish. However, this does not always work well, and more graceful job control is desired." CJC: Grade is based on ALMA audit comments.

1.7-R1.3 Error handling shall be non-destructive (data shall not be left corrupted, and recovery of recent changes be available). Common failure modes such as invalid application parameters, exceeding of resource limits (disk/memory), and algorithm failure modes (e.g. no convergence) should be handled gracefully.
Levels: 1
Functionality: I1/high
Useability: I1
Documentation: N1
Comments: Note from ALMA audit: "There are still failure modes (e.g. glish-tk problems) where aips++ crashes or hangs, sometimes crashing or hanging the windowing system of the OS in addition. Sometimes, files are left corrupted, though it is ususally possible to reproduce processing to the point of crash (though not always). Memory usage of AIPS++ can be high, and problems can occur when using a default setup on a machine with less memory — the should be transparent to the user. These kind of severe errors are high priority with the AIPS++ programmers, and improvement in this area should be steady over the next releases." CJC: Grade is based on ALMA audit comments.

1.7-R1.4 Code traceback of execution errors shall be available. This should be geared to the effective reporting of failure modes and bugs by the user.
Levels: 1
Functionality: A1
Useability: E1
Documentation: I1/med
Comments: Note from ALMA audit: "There seems to be traceback on errors, though they are nearly impossible for non-programmers to figure out." CJC: Grade is based on ALMA audit comments.

1.8 Session Logging

1.8-R1 There shall be session logging, including the following features:

1.8-R1.1 logging of commands and user inputs shall be provided from both the CLI and GUI
Levels: 1,2
Functionality: E1,I2/high
Useability: E1,I2
Documentation: E1,I2
Comments: Note from ALMA audit: "The scripter currently does not log all commands, only (most) GUI operations (bug AOCso03842), while the logger has the option to log the CLI commands but does not do the GUI commands. A full logging (in correct order), perhaps by adding the GUI commands to the logger, is needed." CJC: Grade is based on ALMA audit comments.

1.8-R1.2 logging of tool results such as success or error, files written, time of completion shall be provided

Levels: 1,2
Functionality: A1,A2
Useability: A1,A2
Documentation: A1,A2
Comments: Note from ALMA audit: "The logger seems to record the pertinent information."

1.8-R1.3 logging of tool output such as summaries of results shall be provided
Levels: 1,2
Functionality: A1,A2
Useability: A1,A2
Documentation: A1,A2
Comments: Note from ALMA audit: "The logger seems to record the pertinent information."

1.8-R1.4 the session log shall be readable by the user (i.e. in a text file, not in a binary format)
Levels: 1,2
Functionality: A1,E2
Useability: A1,E2
Documentation: A1,E2
Comments: Note from ALMA audit: "The scripter output is a glish script (ascii), though the logger data is stored in an AIPS++ table." CJC: Grade is based on ALMA audit comments.

1.8-R1.5 session logs shall be executable by the package UI, reproducing the entire session
Levels: 1,2
Functionality: E1,I2/high
Useability: E1,I2
Documentation: E1,I2
Comments: Note from ALMA audit: "The scripter output is an executable glish script, though the scripter output is currently incomplete (see above)." CJC: Grade is based on ALMA audit comments.

1.9 Performance metrics & benchmarks

1.9-R1 The performance of the Package shall be quantifiable and commensurate with the data processing requirements of VLA output and the scientific needs of users at a given time. The timing and reproducibility of results for a fiducial set of reduction tasks on specified test data will be benchmarked (e.g. "AIPSmarks") and compared against other packages.

Levels:

Functionality:

Useability:

Documentation:

1.9-R2 For tasks that are similar to existing AIPS tasks, the Package must run in the same time as AIPS or less.

Levels:

Functionality:

Useability:

Documentation:

1.9-R3 For tasks that go beyond existing AIPS capabilities, the NRAO AIPS++ Users Group and the AIPS++ Project team will develop a list of benchmark specifications that will be reviewed and maintained a minimum of once a quarter.

Levels:

Functionality:

Useability:

Documentation:

2 Data Handling
Steve, Bryan (data objects, import/export), Mark (examination)

2.1 VLA Data Import and Export

Applicable tools include: `ms.fitstoms, vlafiller, vlafillerfromdisk, and ms.ms.tofits.`

2.1-R1 A variety of data formats must be supported by the Package:

2.1-R1.1 The **VLA standard archival data format** must be supported for input without loss of functionality or information.
Levels: 1
Functionality: I/med
Useability: A
Documentation: A
Comments: (STM) The tools `vlafiller` tool and supporting functions fulfill this requirement. The basic fill operations are available, though some useful auxiliary information is missing such as weather.

2.1-R1.2 Standard FITS format (uv and image) shall be supported for both input and output without loss of functionality or information.
Levels: 1
Functionality: E
Useability: A
Documentation: A
Comments: (STM) The tools `image.imagefromfits` function fulfills this requirement. Currently there are problems handling beamsize information from AIPS generated images (e.g. restored clean maps, and convolved maps) where the info is not in the FITS file proper. This should be resolved so the two are compatible. Also, image reading from a FITS tape directly does not seem to be supported.

2.1-R2 Disk-based data storage must be supported.

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) Filling and image import work from disk. Other access (e.g. via MS, ascii) also work from disk.

2.1-R2.1 File sizes shall not have artificial (e.g. OS-imposed 2GB) limits that are visible to the user.
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) I believe that this is the case, though there may be some large upper limit on file size. This should be checked.

2.1-R3 Offline data storage (e.g. DAT, DDS, DLT) must be supported.

Levels: 1

Functionality: A

Useability: A

Documentation: E

Comments: (STM) AIPS++ currently supports tape devices, and it is probably adequate to buffer to disk and use system I/O to devices when this does not work. It appears that any UNIX tape device (e.g. `/dev/nrst0` under Linux) will work, as in the `vlafiller.tapeinput` function description, but there should be an explicit document that explains this.

2.1-R4 The Package must be able to handle, efficiently and gracefully, datasets larger than main memory of the host system.

Levels: 1

Functionality: A

Useability: I/high

Documentation: A

Comments: (STM) The tiling in the table system purports to handle this, and seems to work in practice. Benckmarking is being carried out to determine the efficiency of this, and we do seem to incur fairly large performance penalties on operations on large datasets (hence Usability problems).

2.1-R5 The ability to drop flagged data on inport and export shall be included.

Levels: 1

Functionality: N/low

Useability: N

Documentation: A

Comments: (STM) It does not appear there is a way to throw away flagged data, the flags are carried along.

2.1-R6 The Package internal data format, which may be independent of other supported formats, must not be "bloated" and the required storage should not exceed by more than 1.5× the raw data format.

Levels: 1

Functionality: I/med

Useability: I

Documentation: I

Comments: (STM) This is dataset dependent — it appears that for continuum data the ms to fits filesize ratio is significantly worse: 1.6 and greater for the raw data versus 1.1 or so for line data. However, when the `MODEL_DATA`, `CORRECTED_DATA`, and `IMAGING_WEIGHT` scratch columns are added by `imagr` this jumps to 2.4 for continuum or 1.8 for line data with scratch column compression turned on in `imagr`, and 2.6 or more without compression in both cases. Looking ahead, since the EVLA will operate mostly in line mode (even continuum data will have multiple channels), we can expect to get a bloat factor of about 1.8 with compression. This is only slightly above our limit of 1.5 and thus may be acceptable as is. Compression of other columns or tables would probably easily bring this in line. The capability is built into low-level routines but has not yet made its way up to the user visible level (e.g. in `vlafiller` or `fitstoms`. One issue that must be addressed is whether this level of compression will compromise the subsequent data processing (like it can in AIPS with weights).

2.1-R7 There shall be a facility for dropping data from baselines involving shadowed antennas, with the ability to specify the allowable shadowing (including negative values).

Levels: 1

Functionality: I/high

Useability: I

Documentation: I

Comments: (STM) I suspect that the `vlafiller` tool does drop shadowed data, but there is not *set* function for specifying this, and there is no documentation describing whether this happens.

2.1-R8 Averaging of data over time, IFs, and spectral channels shall be possible, for:

2.1-R8.1 data filled from VLA, FITS or MS format (upon import);
Levels: 1
Functionality: N/med
Useability: N
Documentation: N
Comments: (STM) Averaging is not available on import in `vlafiller`, `fitstoms` or in `ms`.

2.1-R8.2 data written to MS or FITS (upon export).
Levels: 1
Functionality: N/med
Useability: N
Documentation: N
Comments: (STM) Averaging is not available in `ms`.

2.1-R9 Calibration and ancillary monitoring data (e.g. weather information, WVR data, GPS data, pointing) that are filled with the data must be preserved, if requested by the user.

Levels: 1

Functionality: N/low

Useability: N

Documentation: N

Comments: (STM) This is not currently available in `vlafiller` (e.g. for weather data included in the on-line system. Although there are tables in the MS for ancillary data, there is no way to fill them, e.g. from VLA monitor files.

2.2 Import of Images and Other Data Products

2.2-R1 Standard multi-dimensional images must be supported, including:

2.2-R1.1 Spectra and image slices (1D);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.2-R1.2 Planar images (2D);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.2-R1.3 Spectral and time cubes (3D);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.2-R1.4 Higher-dimensional images (4D+).
Levels: 1
Functionality: A
Useability: A

Documentation: A

Comments: (STM) AIPS++ images, e.g. as supported in the `images` module, can have arbitrary dimensionality or shape. I/O to and from FITS, ASCII, and the AIPS++ image formats is provided. I have tested this on a few AIPS-generated image cubes.

2.2-R2 The package shall support all standard projections, including:

2.2-R2.1 sine (SIN);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.2 tangent or gnomonic (TAN);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.3 cylindrical plate carrée (CAR);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.4 Mercator (MER);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.5 stereographic (STG);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.6 Hammer-Aitoff (AIT);
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.2-R2.7 Molleweide (MOL);
Levels: 1
Functionality: A
Useability: A
Documentation: E

Comments: (STM) The support for coordinate systems in the `images` module is extensive and compliant with the WCS standards. For example, see the documentation for the `coordsys` tool for details. You can get the list of supported systems using `coordsys.projection('all');`, the current list is: `AZP TAN SIN STG ARC ZPN ZEA AIR CYP CAR MER CEA COP COD COE COO BON PCO SFL PAR AIT MOL CSC QSC TSC`. One minor bug found was that the links to the WCS documentation at ATNF were broken (defect AOCso04348 submitted), it would be good to have a link that was guaranteed to be robust (doc enhancement).

2.3 Data Objects

2.3-R1 The Package must support data taken in any VLA observing mode (see A).

Levels: 1,2

Functionality: U1/U2

Useability: U

Documentation: U

Comments: (STM) The modes have only been roughly defined and test data is not available (as far as I know) for all modes.

2.3-R2 The Package shall be able to handle the integrated data objects corresponding to the observational programs carried out by VLA. These objects may be implemented in any manner appropriate, though relations between the components of the object must be maintained through some mechanism. These include:

2.3-R2.1 Program header information, e.g. is there information about the observation or history of the data stored in a location that can be easily accessed?
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) There are a number of keywords and values stored various places in the MS, and the current filling seems to be adequate. We should make a list of such information as an appendix here.

2.3-R2.2 Field (e.g. Source ID) information;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) There is a `FIELD` table in the ms (with `FIELD_ID` keyword), and a `SOURCE` table (with `SOURCE_ID`). This is filled correctly on data that I have tested on.

2.3-R2.3 source name;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) There is a `SOURCE` table which associates a `NAME` with a `SOURCE_ID`. This is filled correctly on data that I have tested on.

2.3-R2.4 u, v, w coordinates;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This is in the `UVW` column in the `MAIN` table.

2.3-R2.5 time;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This is in the `TIME` column in the `MAIN` table. It appears that extra time precision can be included using the optional `TIME_EXTRA_PREC` column, though Im not sure if this is used anywhere yet.

2.3-R2.6 baseline;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This is designated by the `ANTENNA1` and `ANTENNA2` columns in the `MAIN` table.

2.3-R2.7 Data from interferometer organized by:

2.3-R2.7.1 position;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This is designated by the `FIELD_ID` column in the `MAIN` table.

2.3-R2.7.2 polarization products;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are in the `DATA` column in the `MAIN` table. There is a matrix of products for each channel or lag. There is a `POLARIZATION` table in the ms which describes what the correlation products are in the data.

2.3-R2.7.3 spectral channels;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are in the `DATA` column in the `MAIN` table. There is a matrix of products for each channel or lag. The channel information is stored in the `SPECTRAL_WINDOW` table in the ms, each visibility in the ms has N channels.

2.3-R2.7.4 frequency bands;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are in the `DATA_DESC_ID` column in the `MAIN` table, which reference spectral windows via `SPECTRAL_WINDOW_ID`, which in turn are specified in the `SPECTRAL_WINDOW` table, e.g. as frequency groups.

2.3-R2.7.5 IFs;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are treated the same as frequency bands (see above).

2.3-R2.7.6 subarray;
Levels: 1
Functionality: U
Useability: U
Documentation: U
Comments: (STM) These are in the `ARRAY_ID` column in the `MAIN` table. These used to be specified in the `ARRAY` table, but the MS2.0 definition claims this is moved to the `OBSERVATION` and an optional `SCAN_SUMMARY` table which I think has yet to be defined. It is not clear to me that multi-array data is fully supported, this needs to be tested.

2.3-R2.8 Coherence function (visibility) data from interferometer, including:

2.3-R2.8.1 cross-correlations;

Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are in the `DATA` column in the `MAIN` table.

2.3-R2.8.2 auto-correlations;

Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are in the `DATA` column in the `MAIN` table, where `ANTENNA1` equals `ANTENNA2`.

2.3-R2.8.3 uncorrected and/or online WVR corrected data (if available);

Levels: 2
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These are supported in the MS in the `iramcalibrater` support for Plateau de Bure data. Extra columns are inserted in the `MAIN` table, and the `DATA` column can be made to contain one or the other.

2.3-R2.8.4 phased-array data (if chosen by user);

Levels: 2
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Can be treated as another (sub)array or telescope. Explicitly mentioned in the MS memos (used e.g. for WSRT and VLA as VLBI element). But I have not tested this. There is an optional `PHASED_ARRAY_ID` column in the `ANTENNA` table to indicate phased array data.

2.3-R2.9 Weights or data uncertainties;

Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) There are `WEIGHT` and `SIGMA` columns in the MS. Later processing can add columns such as `IMAGING_WEIGHT`.

2.3-R2.10 Flagging data or masks;

Levels: 1
Functionality: E
Useability: E
Documentation: A
Comments: (STM) There are `FLAG`, `FLAG_CATEGORY` and `FLAG_ROW` columns in the data. There is also a `FLAG_CMD` table presumably as a record of flagging. This could use some improvement, such as supporting indentifiable flags (enhancement).

2.3-R2.11 Derived calibration data, including:

2.3-R2.11.1 gain solutions;

Levels: 1
Functionality: A
Useability: A
Documentation: A

Comments: (STM) This is in a calibration table (Jones matrix types **G** or **T**).

2.3-R2.11.2 bandpasses;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This is in a calibration table (Jones matrix type **B**).

2.3-R2.11.3 source flux densities;
Levels: 1
Functionality: E
Useability: A
Documentation: A
Comments: (STM) There is a `SOURCE` table though it currently does not contain columns for flux density or spectral info (as in the FITS SU table). At this time, source flux densities are included as part of the model that fills the `MODEL_DATA` column of the main ms table (computed in the `imager.setjy` function). Some gain might be made by having a `SOURCE_FLUX` column in the `SOURCE` table (e.g. for `setjy` type operations in `calibrater` (enhancement).

2.3-R2.11.4 antenna feed polarization parameters;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) The leakages are in a calibration table (Jones matrix type **D**). This covers all necessary parameters, when combined with information on the mount in the `ANTENNA` table, and on the feed polarization (e.g. `POLARIZATION_TYPE` and `POL_RESPONSE`) in the `FEED` table.

2.3-R2.12 A priori calibration data, including:

2.3-R2.12.1 bandpasses;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This would be in a **B** calibration table. The user currently has to keep track of the tables available and what they are for. Note that there are filled gain tables (e.g. from online), but there are no barriers to doing this.

2.3-R2.12.2 source flux densities;
Levels: 1
Functionality: E
Useability: A
Documentation: A
Comments: (STM) These are currently carried out through using an a-priori model to fill the `MODEL_DATA` column of the `MAIN` table (computed in the `imager.setjy` and `imager.ft` functions). It would be nice to have a `SOURCE_FLUX` column in the `SOURCE` table that could be filled using the online fluxes for example (enhancement).

2.3-R2.12.3 antenna feed polarization parameters;
Levels: 1
Functionality: E
Useability: E
Documentation: A
Comments: (STM) The leakages are in a calibration table (Jones matrix type **D**), which would have to be copied from another data set or filled using a table. Mount information is stored the `ANTENNA` table, and the feed polarization in the `POLARIZATION_TYPE` and `POL_RESPONSE` columns in the `FEED` table. Since these are common things to apply a priori

or transfer between observations, it would be useful to have an intermediary table method to store these (enhancement).

2.3-R2.12.4 antenna gain parameters;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This can currently only be applied using a glish script to the `DATA` column itself, but a mechanism for storing this in the proper calibration table (**G** or **T**) is being worked on.

2.3-R2.12.5 atmospheric optical depth;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) This can currently only be applied using a glish script to the `DATA` column itself, but the mechanism for storing this in the proper calibration table (**T**) is being worked on.

2.3-R2.13 Images and/or models produced from data;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These currently would be separate files, and knowledge of the relationship between images and data would be maintained by the user.

2.3-R2.14 Processing history;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) There is a `HISTORY` table in the MS, so the mechanism is there. However, it currently does not get filled in with the AIPS++ processing history info. It seems to contain only the AIPS history if you used `fitstoms` to read in an AIPS FITS file.

2.3-R2.15 Observed position as a function of time, for moving sources;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These can be stored in the `FIELD` table (e.g. as a polynomial) or the `POINTING` table.

2.3-R2.16 System temperature information, per antenna/IF/poln as a function of time;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These can be stored in the `SYSCAL` table.

2.3-R2.17 Weather information, as a function of time.
Levels: 2
Functionality: A
Useability: A
Documentation: A
Comments: (STM) These can be stored in the `WEATHER` table.

Comments: (STM) These requirements pertain to the existence of the needed items, not the method of filling them. See the MeasurementSet definition V2.0 document for details on the ms. We find the ms format to be general enough to cover all proposed instances, though it may end up bloated if we ask for too much. Note that description of the calibration table formats is given in Calibration table definition version 2.0.

2.3-R3 There must be a selection mechanism integrated within tools to choose between the various available data subsets such as:

2.3-R3.1 source names, field names, and/or user specified data qualifiers (with wildcarding);
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) These are currently specified by `fieldid` parameters, or `FIELD_ID` keywords in queries. Some tools allow specification by source name (e.g. '0137+331') — this should be possible in all cases. Wildcarding would work on (probably complicated) queries using TaQL, but should be built into the more general source name specification. Although functional, the current system is difficult to use due to the differing specification methods among tools, which should be uniform across the package. Perhaps all access should be through common methods in the `ms` module.

2.3-R3.2 time;
Levels: 1
Functionality: I/high
Useability: I/high
Documentation: A
Comments: (STM) These are currently specified differently in various places, mostly through TaQL queries, occasionally in ranges (e.g. `ms.select`). There is currently not enough control (like timeranges for calibration solutions), and even when it works the current system is difficult to use due to the differing specification methods among tools, which should be uniform across the package. Perhaps all access should be through common methods in the `ms` module.

2.3-R3.3 baselines;
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) Specification by interferometer number (e.g. in `msplot`, by antennas in a set of baselines (`flagger.setbaselines`), or by TaQL, are available. Again, this selection needs to be uniform across the package.

2.3-R3.4 antennas;
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) Specification by interferometer number (e.g. in `msplot`, by antennas in a set of baselines (`flagger.setbaselines`), or by TaQL, are available. Again, this selection needs to be uniform across the package.

2.3-R3.5 u, v, w coordinates;
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) Currently available only through TaQL (e.g. `flagger.query`,

`calibrater.setdata`, `ms.selecttaql`) or using the items allowed in `ms.select` operations. Also some functions have the more useful uv-range setting (e.g. `calibrater.setdata`, `imager.uvrange`), but again this is not uniform across the package.

2.3-R3.6 polarization products or channels;
Levels: 1
Functionality: I/med
Useability: I/med
Documentation: A
Comments: (STM) Differs among tools, as the others, but is specified by `POLARIZATION` ids (e.g. LL,RL,LR,LL) in most tools (e.g. `msplot`, which has the most straightforward selection) or ids (e.g. flagger.setpol). Also via TaQL and `ms.select`. Does not entirely work for dealing with a single polarization (e.g. RR) as opposed to Stokes I.

2.3-R3.7 bands (frequency bands, IFs);
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) The IFs are usually specified by the `spwid` numbers in most tools, or using TaQL strings on `SPECTRAL_WINDOW_ID`. I would guess frequency bands would be specified as a frequency group with its own spwid or `SPECTRAL_WINDOW_ID`, or by a name as in `iramcalibrater`. Needs to be made uniform.

2.3-R3.8 spectral channels;
Levels: 1
Functionality: I/med
Useability: I/high
Documentation: A
Comments: (STM) The channels are not always set in the same way, usually through `nchan`, `start`, and `step` parameters in the selecting function, but also sometimes as a vector of channel numbers (e.g. in the `flagger.setchan` function). You can also select by velocity instead of channel in many cases. Just having start, stop, and step is insufficient, and sets of ranges or lists of channels needs to be available uniformly across the package.

2.3-R3.9 subarrays;
Levels: 1
Functionality: A
Useability: I/low
Documentation: A
Comments: (STM) This would be specified with a parameter (e.g. `array_id` in `ms.select` or `arrayid` in `flagger.setids`) or `ARRAY_ID` keyword in a query. Needs to be uniform.

2.3-R3.10 mosaic or scanning pointing centers;
Levels: 2
Functionality: E
Useability: I/med
Documentation: A
Comments: (STM) These are specified by `fieldid` parameters, or `FIELD_ID` keywords in queries. Should be made uniform, and might need some more flexibility to handle OTF mosaics easily (enhancement).

2.3-R3.11 WVR-corrected or uncorrected baselines (if available).
Levels: 2
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Currently only in `iramcalibrater.phcor`, but this works if the proper

columns are filled in the dataset.

Comments: (STM) The key to these requirements is that the functions be uniform between tools, hence the problems with usability. These could be fixed at one fell swoop by use of a common `ms` selection across tools.

2.3-R4 For polarization products, transformation must be provided to the desired Stokes output parameter(s).

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) This is available in the context of imaging.

2.3-R5 All standard time systems shall be supported, including:

2.3-R5.1 Universal Time (UT), also UT1;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.2 Coordinated Universal Time (UTC);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.3 International Atomic Time (IAT);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.4 Local Sidereal Time (LST);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.5 Greenwich Mean Sidereal Time (GMST);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.6 Julian Date (JD), also Modified Julian Date (MJD);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R5.7 Dynamical Times (TDT, TDB, ET (Ephemeris Time)).
Levels: 1
Functionality: A
Useability: A
Documentation: A

Comments: (STM) The tools (e.g. the `measures.epoch` function) support the following epochs (as listed using `dm.listcodes(dm.epoch())`: `LAST LMST GMST1 GAST UT1 UT2 UTC TAI TDT TCG TDB TCB IAT GMST TT ET UT`. `MJD` and `JD` are supported as time formats.

2.3-R6 All standard coordinate systems shall be supported, including:

2.3-R6.1 equatorial (RA, DEC);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R6.2 terrestrial (AZ, EL);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R6.3 galactic (GLON, GLAT);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R6.4 ecliptic (ELON, ELAT);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R6.5 helioecliptic (HLON, HLAT);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R6.6 supergalactic (SLON, SLAT);
Levels: 1
Functionality: A
Useability: A
Documentation: A

Comments: (STM) Allowed directions (listed using `dm.listcodes(dm.direction())`) are: `J2000 JMEAN JTRUE APP B1950 BMEAN BTRUE GALACTIC HADEC AZEL AZELSW AZELNE AZELGEO AZELSWGEO AZELNEGEO JNAT ECLIPTIC MECLIPTIC TECLIPTIC SUPERGAL ITRF TOPO` and MERCURY VENUS MARS JUPITER SATURN URANUS NEPTUNE PLUTO SUN MOON COMET.

2.3-R7 All standard coordinate reference frames and equinoxes shall be supported, including:

2.3-R7.1 'Virtual Place': geocentric coordinates of standard equinox, including:

2.3-R7.1.1 J2000 (and other FK4 equinoxes);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R7.1.2 B1950 (and other FK5 equinoxes);
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R7.1.3 X2050;
Levels: 1
Functionality: U
Useability: U
Documentation: U
Comments: (STM) Note that this is not defined anywhere I could find online, Google only returned the EVLA SSR email! Nor was it in the versions of the Astronomical Almanac Explanatory that I could find. Unless someone can point to a reference for this, I will treat it as an FK5 equinox and drop this reqirement. This seems more like an EVLA requirement anyway.

2.3-R7.2 'Local Place': topocentric coordinates of standard equinox;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R7.3 'Apparent Place': geocentric coordinates of date;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R7.4 'Topocentric Place': topocentric coordinates of date;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R7.5 International Celestial Reference System (ICRS).
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) ICRS can be derived from ITRF as far as I can tell.

Comments: (STM) Allowed directions (listed using `dm.listcodes(dm.direction())`) are: `J2000 JMEAN JTRUE APP B1950 BMEAN BTRUE GALACTIC HADEC AZEL AZELSW AZELNE AZELGEO AZELSWGEO AZELNEGEO JNAT ECLIPTIC MECLIPTIC TECLIPTIC SUPERGAL ITRF TOPO` and MERCURY VENUS MARS JUPITER SATURN URANUS NEPTUNE PLUTO SUN MOON COMET.

2.3-R8 All standard velocity definitions shall be supported, including:

2.3-R8.1 radio;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R8.2 optical;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R8.3 redshift.
Levels: 1
Functionality:
Useability:
Documentation:

Comments: (STM) The doppler frames `RADIO Z RATIO BETA GAMMA OPTICAL TRUE RELATIVISTIC` are supported (e.g. in the `measures.doppler` function, listed using `dm.listcodes(dm.doppler())`).

2.3-R9 All standard velocity frames shall be supported, including:

2.3-R9.1 topocentric;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.2 kinematic LSR;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.3 dynamic LSR;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.4 geocentric;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.5 barycentric;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.6 heliocentric;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.7 galactocentric;
Levels: 1
Functionality: A
Useability: A
Documentation: A

2.3-R9.8 local group;
Levels: 1
Functionality: N/low
Useability: N
Documentation: N
Comments: (STM) This does not appear to be available.

2.3-R9.9 with respect to a user-specified system of rest.
Levels: 1
Functionality: I/low
Useability: I
Documentation: I
Comments: (STM) You can add an offset to an existing radial velocity frame, but I do not see a way to define a new one in `measures`.

Comments: (STM) The velocity frames `LSRK LSRD BARY GEO TOPO GALACTO` are supported in the (`measures.radialvelocity`) function (obtained using `dm.listcodes(dm.radialvelocity())`). This seems to cover all but possibly Local Group and user defined frames.

2.3-R10 Coordinates and locations (e.g. of the antennas, or the array center) defined with respect to the standard frames shall be supported, including:

2.3-R10.1 topocentric;
Levels: 1
Functionality: A
Useability: A
Documentation: E

2.3-R10.2 geocentric.
Levels: 1
Functionality: A
Useability: A
Documentation: E

Comments: (STM) Frames for positions include the the WGS84 and ITRF formats (obtained via `dm.listcodes(dm.position())`). Wim Brouw says that these are covered. More explanation in the documentation is necessary I think (enhancement).

2.3-R11 Any existing (e.g. online) flagging mask or table must be filled along with the data and available for use in AIPS++.

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) In `vlafiller` the online flags are filled.

2.3-R11.1 The flagging mask or table must be maintained and associated with the data it refers to during any subsequent operations (such as splitting of data sets).
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Currently, flags are applied in columns of the ms, and thus transferred subsequently.

2.3-R11.2 The flagging mask or table shall be transferable to other equivalent data sets (e.g. flags derived for a continuum dataset should be transferable to a line dataset derived from the same observations).
Levels: 1
Functionality: I/med
Useability: I/med
Documentation: A
Comments: (STM) Though there are flag tables generated by some tools (e.g. `msplot`), this is not supported universally (e.g. in `flagger`). It is not yet possible to transfer flags easily.

2.3-R12 Merging (e.g. concatenation) and splitting of datasets shall supported:

2.3-R12.1 Extraction of specified subsets of data (e.g. by source, time, band, feed) shall be supported.
Levels: 1
Functionality: E
Useability: E

Documentation: A
Comments: (STM) It appears that you can do this with the `ms.command` function which makes a 'new' ms that is referenced to an old ms (so you cannot delete the old one). You might be able to use some `table` commands to then copy this (e.g. `table.copy`). It should be made much simpler.

2.3-R12.2 Merging of data subsets (e.g. combination of different epochs, configurations, subarrays, frequency offsets, mosaic pointings) shall be supported.
Levels: 1
Functionality: E
Useability: E
Documentation: A
Comments: (STM) Concatenation passing through uvfits is possible using `ms.concatenate`. It should be made more widely applicable.

2.3-R12.3 The merging and splitting process, including selection of data to be processed (e.g. merging of frequency channels which may be differently labeled) shall be straightforward and not overly complex.
Levels: 1
Functionality: I/high
Useability: I/high
Documentation: A
Comments: (STM) Concatenation of ms is still difficult (`ms.concatenate` must go through FITS first). This needs substantial improvement (and is on the development list), and would be a high-impact improvement. Note that there is some better merging and splitting functionality in the `iramcalibrater` tool, though it can still be finicky.

2.3-R12.4 The use of merged or split data in the Package shall be robust and transparent to the user. Subsequent operations shall work the same, whether the data is in its original form or built from merged data subsets, where possible.
Levels: 1
Functionality:
Useability:
Documentation:
Comments: (STM) This appears to be true, as long as it is a valid MS.

2.3-R12.5 The tools shall be able to operate on multiple data objects, including multiple subsets of multiple objects, without explicit concatenation (ie. allow operations on unions of objects) seamlessly.
Levels: 1
Functionality: N/low
Useability: N/low
Documentation: I
Comments: (STM) This would be a high-impact capability in AIPS++, but is low priority for VLA (but should be included for EVLA).

2.3-R12.6 The appropriate calibration and ancillary monitoring data for the merged or split data (e.g. keeping only the data relevant to sources split out) shall be preserved.
Levels: 1
Functionality: I/low
Useability: I
Documentation: I
Comments: (STM) It appears that the current splitting does not deal with the subtables completely (it may copy whole rather than selecting relevant parts).

2.3-R13 Users shall have access to, and the ability to change, all aspects of the data including the header.
Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) Full access to all aspects of the ms is available, and relatively straightforward, in AIPS++ at glish level.

2.3-R14 The Package must support locking data files so that there is no possibility of one process corrupting a file that is also being written to by another process in the Package. The default model should be: "one writer, multiple readers."

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) Table locking is implemented in AIPS++.

2.3-R15 Distinctions between "single-source", "multi-source", single-dish, and interferometer datasets shall be avoided with context built into the dataset or header.

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: (STM) This is the case for measurement sets.

2.3-R16 When sorting or indexing is needed for performance enhancement, it shall be carried out in a manner transparent to the user.

Levels: 1

Functionality: E

Useability: A

Documentation: A

Comments: (STM) This seems mostly to be the case (e.g. in `imager`), with tiling in the table system allowing reasonably efficient access along all axes of the data. However, tiling seems to be set during MS writing and then fixed. This is OK for VLA data now, but on-the-fly sorting will probably be needed for large datasets and EVLA (enhancement).

2.3-R17 Compression of the data, with a selectable level of loss, shall be possible at:

2.3-R17.1 the filling or import stage;
Levels: 2
Functionality: N/low
Useability: N/low
Documentation: N
Comments: (STM) The `vlafiller` and `fitstoms` functions do not appear to have compression built in.

2.3-R17.2 arbitrary stages of the processing path.
Levels: 2
Functionality: I/low
Useability: I/low
Documentation: I
Comments: (STM) Some compression, such as scratch column compression in the `imager` constructor, is available this release. However, it does not appear that user controlled compression

is fully available after filling. There should be a way to generally compress columns (e.g. in `ms`) later on. Note that there is compression (using the **bitpix** parameter) available on FITS export using `image.tofits`.

2.4 Data Examination

Applicable tools include: `ms, msplot` (e.g. summary information, antenna, & uv plots), and the `catalog` GUI.

Note: This section covers the non-graphical tools for looking at the state of the data (graphical methods are in Visualization), e.g. summaries of what physically is there in the data set (LISTR type functions). I have put the GAIN and MATRIX style LISTR functions here also.

2.4-R1 The Package should provide easy access to, and printable output for, summaries of the data set, including:

2.4-R1.1 header information (e.g. AIPS IMHEAD);
Levels: 1
Functionality: E
Useability: A
Documentation: A
Comments: (STM) Available using `ms.summary` for all important information, though it is not clear whether the rows in table corresponds to number of visibilities. For images, `image.summary` does the same, and gives everything that AIPS IMHEAD does except perhaps the max and min values (enhancement).

2.4-R1.2 source and field summaries (e.g. AIPS LISTR option SCAN);
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Available using `ms.summary` with **verbose=T**.

2.4-R1.3 scan summaries (e.g. AIPS LISTR option SCAN);
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Available using `ms.summary` with **verbose=T**.

2.4-R1.4 antenna locations (e.g. AIPS PRTAN);
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Available using `ms.summary` with **verbose=T**.

2.4-R2 The User should be able to determine the current state of the data set at any time in a straight forward manner, with printable output, including:

2.4-R2.1 listings of calibration tables present (e.g. AIPS IMHEAD);
Levels: 1
Functionality: A
Useability: A
Documentation: A

Comments: (STM) These are separate files on disk, and the user must keep track of the naming themselves. However, this is acceptable. If calibration subtables attached to the MS is implemented as are current plans, then these will be viewable in the `catalog` browser and will show up in `ms.summary`.

2.4-R2.2 summary of columns present in a specified table of the MS;
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: (STM) Viewable in the `catalog` browser, and through the `table.summary` function, which lists the columns.

2.4-R2.3 convenient listing of the contents of a specified column for a table of the MS (e.g. CORRECTED_DATA, MODEL_DATA, WEIGHT, & IMAGING_WEIGHT columns of the MAIN table, equivalent to AIPS PRTAB);
Levels: 1
Functionality: A
Useability: I/med
Documentation: A
Comments: (STM) Functionality (more or less) available through `ms` and `table` tools (e.g. `ms.getdata`) but it would be extremely difficult for users to format this for easy printing (it returns glish records). Can also browse in `catalog` but that is excruciating for large datasets. Convenient `ms` global functions should be constructed to carry these out (a LISTR emulator).

2.4-R2.4 convenient listing of the visibility data as a function of time, with optional averaging (AIPS LISTR option LIST);
Levels: 1
Functionality: I/med
Useability: I/med
Documentation: I
Comments: (STM) Functionality might be available through `ms` and `table` tools but it would be extremely difficult to do this, particularly the averaging (I don't see how to do this), and then formatting the output would be inconvenient. Convenient `ms` global functions should be constructed to carry these out (a LISTR emulator).

2.4-R2.5 convenient listing of the calibration data for a specified calibration table as a function of time (AIPS LISTR option GAIN);
Levels: 1
Functionality: I/med
Useability: I/med
Documentation: I
Comments: (STM) Functionality (more or less) available through the `table` tool but it would be extremely difficult for users to format this for easy printing (it returns glish records). Can also browse in `catalog` but that is excruciating for large datasets. Convenient `ms` global functions should be constructed to carry these out (a LISTR emulator).

2.4-R2.6 matrix-style listing of the data with time, and scan, averaging (AIPS LISTR option MATRIX).
Levels: 1
Functionality: I/med
Useability: I/med
Documentation: I
Comments: (STM) I would have to say this might be possible at glish level for a black-belt user, but is essentially not available. Convenient `ms` global functions should be constructed to carry these out (a LISTR emulator).

3 Calibration and Editing
Greg (polarization), John (spectral line), Claire (apriori calibration)

3.1 Flagging, Autoflagging, and Interactive Editing

Applicable tools for editing include: `flagger`, `autoflag`, and `msplot`. Editing functionality should include capabilities like: identification of bad data based on a plot of averaged data, clean component subtraction, and flagging based on plotcal solutions.

Comments: JEH: And `viewer`. Nowhere in the documentation (excluding VLA cookbook) could I find something that summarized these as the primary methods of editing. It doesn't help that the four tools are spread over 3 different packages (synthesis, general, and display). I think that a short section on Editing Data belongs in GR2, "Generic Processing". All this need do is mention the four main tools for examining and editing data (msplot, viewer, flagger, and autoflag) and provide links to the relevant sections. Maybe there are ways to edit data in e.g, pgplotter tools or via TaQL directly; if so these should also be mentioned. There was a good newsletter article on msplot; this should be linked to also.

Comments: JEH: (cont'd) The editing capabilities of the viewer are particularly hard to find on the web if you don't know where they are. This it the shortest path I could find from the URM:

Comments: JEH: (cont'd) Display: viewer-> Viewer-Module; follow link to "Viewer display panel tool" after GUI functionality -> GUI; midway through this page follow link for the "Adjust GUI" -> Options Record; near end of this page you final see the flagging options (Note: no mention of viewers editing functionality are mentioned on the previous pages, or even on the viewer "Fundamentals"page).

Comments: JEH: (cont'd) The Msplot documentation is better in this regard. Editing data is mentioned prevalently throughout, and there is a link to various editing modes directly in the ms - module TOC. Here we also find a section on "Editing via the viewer", which describes *a* viewer editing procedure, but this is different from the native viewer editing capabilities, and so should probably be called something else. The description here is poorly arranged; I'll file an enhancement on that.

Comments: JEH: (cont'd) So, I would add a requirement that all the various flagging paths be described clearly in some obvious place, with proper documentation for each, and would grade this as N1/high.

Comments: JEH: (cont'd) In the following, I could not get autoflag to run, so could not test it. setselect kept saying "Arguments are invalid: please check", and the other ones said "missing selected correlations, ignoring this chunk Unable to process this chunk with any active method". So that has not been evaluated.

3.1-R1 Editing and flagging shall not modify raw data – ever.

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: JEH

3.1-R2 Editing and flagging of data shall be easily reversible within the Package (i.e. not requiring re-reading of the data from the archive).

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: GBT: This is provided in msplot.

Comments: JEH: If what is meant is flagging points, this is true also for viewer, flagger, and autoflag: you can select a previously flagged region or range of parameters, and unflag those points. If what is meant is to undo the nth flagging operation, this is not possible (although viewer allows you to incrementally remove the last unsaved edit). We should discuss if this is the functionality we want, in which case we would add a level2,N2 to the grade.

Comments: JEH: Documentation: I couldn't find an adequate description in the documentation of what is in the $< msdata > .flags. < number >$ files that are created each time msplot is run with Edit=T. I assume they are the "flag column tables" discussed in the msplot "Background on editing" page; it would help to use the same nomenclature everywhere. In this section it seems to imply that these tables are made only if the user elects NOT to apply the flags, but they appear to be made whether flags are applied or not. When you exit msplot text appears in the logger which tells you what this table is and what you might do with it; that message belongs somewhere in the documentation. So I would give documentation an E

3.1-R3 Logging of editing steps will be clearly marked in a history table or data object (possibly distinct from a more readable history).

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: GBT: This is provided in msplot, but how to use the logging is unclear.

Comments: JEH: I believe this means a specific table attached or referenced to the dataset; I do not believe the logger fulfills this functionality. There is no real history table, as far as I can tell. There are flagfiles created each time msplot is started up, but it would be quite hard to tell after the fact what they each are. And they do not contain the current flags - if msplot was run but the flags were not saved, then the flagfile contains the flagging commands that were NOT applied. If the flags were applied, then it contains the flagged columns that were in effect prior to msplot being run (at least, that is how I understand it). There are no such files for flagger, autoflag, or the viewer. So I would grade this as level 2, N2/med.

3.1-R4 Individual edit undo should be possible.

Levels: 1

Functionality: I1

Useability: I1

Documentation: E1

Comments: GBT: In some cases it is possible to undo edits in msplot, but not all.

Comments: JEH: Does this mean individual editing commands? Like the 5th box that was drawn in msplot? If so, then this is not implemented; it is only possible to (1) unflag data that was by the accumulation of previous flagging commands, or (2) revert to different flagfiles that were in effect at some run of msplot (but this only works for msplot flags, not viewer or flagger or autoflag flags). These requirements are written as if flagging is like flag files in AIPS, and that is not how it works here. I quite extensively tested whether previously flagged points could be unflagged in different runs of the various tools, and that points flagged in one tool could be removed in another, and this all appears to work correctly. But this is reversing the flagging of individual data points, not undoing individual edits.

3.1-R5 Data editing and flagging shall be possible based upon array, environmental, astronomical, and calibration monitoring data, including:

Comments: JEH: this is probably possible via complicated TaQL operations, but it is not obvious how one would do this, and it is not well documented. I would guess flagging on Tsys and RFI monitoring would be level2/low priorities; can't speak for the other options.

3.1-R5.1 pointing data
Levels:
Functionality:
Useability:
Documentation:

3.1-R5.2 array tracking information (encoders)
Levels:
Functionality:
Useability:
Documentation:

3.1-R5.3 weather data (wind speed, temperature, relative humidity, pressure)
Levels:
Functionality:
Useability:
Documentation:

3.1-R5.4 T_{sys} data
Levels:
Functionality:
Useability:
Documentation:
Comments: JEH: N2/low?

3.1-R5.5 WVR data (if available)
Levels:
Functionality:
Useability:
Documentation:

3.1-R5.6 RFI monitoring
Levels: 2
Functionality: I2
Useability: I2
Documentation: I2
Comments: GBT
Comments: JEH: why is this I and not N?

3.1-R5.7 site-test interferometer (STI) and/or tipping radiometer
Levels: 2
Functionality: I2
Useability: I2
Documentation: I2
Comments: GBT

3.1-R6 Interactive data editing shall be largely graphical and intuitive. Specialized editing display tools should include:

3.1-R6.1 specification of data by selection on observational parameters, and/or plotting versus these parameters, including:

3.1-R6.1.1 antenna
Levels: 1
Functionality: A1
Useability: A1

Documentation: A1
Comments: GBT

3.1-R6.1.2 baseline
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: GBT

3.1-R6.1.3 time range
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: GBT

3.1-R6.1.4 uv range
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: GBT

3.1-R6.1.5 pointing center
Levels:
Functionality:
Useability:
Documentation:
Comments: GBT: Delete?
Comments: JEH: As STM points out in ALMA audit, you can sort of do this by specifying scan number or FIELD_ID. Not sure if we really want to be able to specify an arbitrary pointing center to flag on.

3.1-R6.1.6 azimuth, elevation
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N?

3.1-R6.1.7 hour angle range
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N?

3.1-R6.1.8 parallactic angle
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N?

3.1-R6.1.9 slice through data cube
Levels:

Functionality:
Useability:
Documentation:
Comments: GBT: Don't understand this.
Comments: JEH: Neither do I.
Comments: STM: delete this requirement

3.1-R6.2 displays of spectra and spectral cubes.
Levels:
Functionality:
Useability:
Documentation:
Comments: JEH: I don't understand this. Is this an image-based flagging requirement, like tvblank? The comments in the ALMA audit seem to think this refers to flagging with spectral frequency/channel along one axis. This is available in both viewer and msplot, as is viewing other plots and viewing along frequency axis. I would recast it is a requirement to flag on displays plotted against the frequency domain, and grade it as A1 in all catagories.
Comments: STM: delete this requirement

3.1-R6.3 display of closure quantities.
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT

3.1-R6.4 display of and selection on monitor data quantities
Comments: JEH: why is this I and not N? (e.g. Tatm, Tamb).
Levels: 2
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N?

3.1-R6.5 the above with baseline, time, band and/or channel averaging.
Levels: 2
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N? True averaging is not available. In msplot you can plot a range of channels, but this is not the same as averaging. In this mode, flags cannot extend beyond the channels that are displayed.

3.1-R6.6 for interferometer data, amplitude (phase) vs. time on each baseline (Difmap vplot), vs. time-baseline (AIPS TVFLG) with interactive zoom, selection, and clipping.
Levels: 1
Functionality: E1
Useability: E1
Documentation: E1
Comments: GBT
Comments: JEH: Why are these not A1? I would say A1, with a E2/high requesting amplitude (phase) for baselines plotted in U-V plane. Need an unzoom button for msplot. Documentation is not as clear as it could be - viewer documentation needs editing capabilities mentioned further up and listed in TOC; msplot description of tvflg-like capabilities needs to be straightened out..

3.1-R6.7 editing of data points based on statistical quantities, including:

Comments: JEH: autoflag is supposed to do some of this, but I couldn't get it to run.

3.1-R6.7.1 a data point versus a running mean over a timescale.
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N1/high?

3.1-R6.7.2 a data point versus a median over a timescale.
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N1/high?
Comments: STM: delete this requirement - would anyone need it?

3.1-R6.7.3 an rms scatter in a time range.
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: why is this I and not N1/high?

3.1-R6.7.4 difference versus a model.
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: This is A1 for both msplot and viewer (plotting vs. residuals/ratio). An example in the documentation would be nice.

3.1-R6.8 auto-scaling and user-specified scaling of axes
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: GBT

3.1-R6.9 auto-scaled and user-specified colormap or greyscale
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: GBT

3.1-R6.10 inclusion and marking of flagged data in plots and in auto-scaling
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT
Comments: JEH: A1 for viewer, N1/med? for msplot.

3.1-R7 Non-interactive and automated editing tools shall be available in the Package, including:

3.1-R7.1 direct editing of data based on user-specified ranges for quantities available for plotting or editing in interactive mode
Levels:
Functionality:
Useability:
Documentation:
Comments: JEH: A1 for all I believe

3.1-R7.2 automated editing with tunable criteria for automated selection of parameter ranges
Levels:
Functionality:
Useability:
Documentation:
Comments: JEH: this is autoflag, but I couldn't get it to do a damn thing, so E1/high?

3.1-R8 Flagging shall be possible based upon standard or user-defined models in either functional or tabular form. Arbitrary user-specified scaling of data shall be simple.

Levels: 1

Functionality: A1

Useability: E1

Documentation: I1

Comments: GBT: I'm sure glish provides some way to do this, but I don't know the syntax of it, or where it might be described.

Comments: JEH: This looks like two disjoint requirements. The first looks the same as 3.1-R6.7.4, and the second looks like 3.1-R6.8.

3.1-R9 Editing shall be incorporated into most visualization tools where data or data-derived quantities are plotted, such as from calibration solutions, amplitude vs. uv-distance plots, or any number of other plots. A "see-it, flag-it" capability shall be the standard within the tools.

Levels: 1

Functionality: E1

Useability: E1

Documentation: E1

Comments: GBT: This is generally the philosophy in msplot, but it is not always obeyed (e.g., one is not allowed to edit when multiple/plots/page are displayed, and sometimes edits apply unintentially to data that is not displayed.)

Comments: JEH: You can sort of edit vs. calibration solutions once MODEL column has been filled (via msplot or viewer, data=resid or ratio), but true editing vs calibration solutions (e.g., in calibrator.plotcal) is not there. Viewer could be graded A1 here, but I agree with GBT's statements on msplot.

3.1-R10 It shall be possible to expand the scope of the edits to apply to other spectral channels, IFs, or polarizations.

Levels: 1,2

Functionality: E1,N2/med

Useability: E1,N2

Documentation: E1,N2

Comments: GBT: This is generally true in msplot for the raster mode. For the XY plot mode it is not generally true. One would like to flag a specific antenna (e.g. see IBLED in AIPS), and this is not possible unless that antenna is specified beforehand.

Comments: JEH: I added the level 2. Not applicable for flagger or autoflag. viewer allows this (but loads the entire dataset - no option to display a subset as far as I can tell). msplot will not allow you to flag outside the setdata selection. I think the requirement suggests that we want to display a subset of data, but extend flagging commands outside this range, e.g., display a subset of channels, but have flagging commands apply to all channels, even those not included in dataselect; or display FIELD_ID=2, but extend flags to all fields; or show plots with only every nth point plotted (skip in msplot is supposed to do this, but screws up flagging if you use it), but extend flags to every point. Maybe we need a dataselect option and a datadisplay option?

3.2 Calibration & Self-Calibration of Interferometer Data

Applicable tools for calibration include: `calibrator` and `imager` (setjy, the masking and interactive clean interface, and the imager–calibrator interface for self-calibration). Applicable documentation should include Getting Results, volumn 2, chapter 1: Synthesis calibration.

Note: Antenna-based determination of calibration quantities such as gains, polarization leakages, bandpasses, will be the primary form of calibration where appropriate. However, in addition to antenna-based calibration, baseline dependent corrections will be important in some cases. For example, coherence loss due to atmospheric phase fluctuation depends on baseline length (this aspect will be more important at higher frequencies). Also, in general, the bandpasses are baseline dependent and contain non-closing terms.

Comments: JEH: I generally find the calibration documentation very sparse. There is a good description of the ME and Jones matricies solutions in GRvol2, as well as some nice examples. But the tool descriptions in the reference manual are really very poor (e.g. description for calibrator.solve: "Solve for the selected calibration components"). But what I find missing from any of the documentation is, how are the components solved for? How are the solutions interpolated? Are gains and phases solved for and interpolated independently, or as vector quantities? Are they applied to nearest data, or averaged? Also, nowhere in the description do I see the calibration formalism broken down to describe how amplitudes and phases factor in, and how baseline-dependent vs. antennae-dependent terms might be solved for (the ME description includes terms for both multiplicative and additive baseline-dependent terms, but no discussion on how they might be solved for, and the next paragraph ignores baseline-based terms). Perhaps most importantly, a lot of us grew-up understanding how good our calibration solutions were based on closure errors. Are there no closure errors anymore? What the heck is reported during the calibration solution? Fit per unit weight and sum of weights. What do these mean? For these reasons, most of the documentation would be graded E by me.

3.2-R1 The Package must be able to handle reliably all **designated VLA standard calibration modes**, possibly including but not exclusive to: fast-switching calibration transfer, planetary observations, and WVR data if available.

Levels: 1

Functionality: E1

Useability: E1

Documentation: E1

Comments: GBT: Ordinary calibration transfer (G calibration) works, but is still cumbersome in places (especially in the specification of antennas to use).

3.2-R2 Calibration and correction of data shall be easily reversible within the Package (i.e. not requiring re-reading of the data from the archive).

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: GBT: Its easy to delete tables, but also to hang up aips++ in the process if one isn't carefull.

3.2-R3 Data calibration operations shall take into account the scan structure and switching scheme of the data. The user shall be able to request calibration solution intervals that correspond to and reference from scan boundaries, for example.

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: GBT: It was difficult to find documentation indicating that an integration time of '0' meant to use scan averages. I figured it out by trial and error.

3.2-R4 Calibration shall involve flexible averaging of data and calibration quantities with user-controllable interpolation, filtering, weighting, and application scope.

Levels: 1

Functionality: I1

Useability: I1

Documentation: I1

Comments: GBT: If there are controls to for the filtering, editing and interpolation of calibration quantities I haven't seen them. Its probably possible to manipulate these tables using basic tools in aips++ like the table browser, but some more sophisticated tools are needed for astronomers.

Comments: JEH: agreed. As mentioned at top of this section, I can't find out how data are interpolated between calibrator and source. From GM, I guess it takes the nearest point, and we definitely want something more sophisticated than that. As STM points out in ALMA evaluation, at a minimum we would like the functionality provided by CLCAL.

3.2-R5 Data calibration and correction shall be possible based upon standard or user-defined models in either functional or tabular form. Arbitrary user-specified scaling of data shall be simple.

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: GBT

Comments: JEH: I suggest changing this to include only standard models, with user defined models covered in the next requirement. See my comments for that

3.2-R6 Incorporation of standard models (e.g. planetary disks, models for HII region structure, known source spectra) shall be easy for calibration operations. User-supplied models in a standard format (e.g. ASCII table) for these quantities shall be supported.

Levels: 2

Functionality: E

Useability: E

Documentation: E

Comments: GBT: It is possible to supply a model, but it has to be in aips++ table format. The ability to accept an ASCII table is needed.

Comments: JEH: I suspect there is a not-too difficult way to create a model image which could then be ft'd and calibrated on, although I couldn't find anything obvious (perhaps image.make and regionmanger?). We would like a more straight forward way to take an externally specified model and turn it into an image or to a UV model. There is a synthesis.componentmodels package that sounds like it provides a lot of this (including reading in from an ascii file), but its not entirely clear (it looks like all components have to have the same flux?). So this needs to be documented and linked to from GRvol2, URM and the cookbook.

3.2-R7 Access to time history of calibration information (e.g. source catalogs containing flux density histories) shall be built into calibration engines. Output of calibration procedures shall be exportable into similar structures.

Levels: 1,2

Functionality: A1,I2

Useability: A1,I2

Documentation: A1,I2

Comments: GBT: As this refers to 'setjy' type level 1 functions then yes its there. If this refers to fetching information from the flux history database (level 2 type function) then no, thats not there.

3.2-R8 The Package shall support the establishment and verification of the relative calibration of the various component epochs, configurations or other subsets for merged datasets.

Levels: 1

Functionality: I1

Useability: I1

Documentation: I1

Comments: GBT: Data combination is not readily possible at the moment.

Comments: JEH: I agree with the ALMA audit - what is meant by this, apart from the ability to concatenate datasets (which is covered elsewhere)? Do we want to take a B-array dataset and concatenate with a C-array, and figure out a better scaling between them? Tim has an article in the Dec2002 Aips++ newsletter that talks about combining single dish with VLA data, so some of this must exist.

3.2-R9 Antenna-based determination of calibration quantities shall be available, and are the default choice for calibration in tools where appropriate, for quantities including:

3.2-R9.1 antenna gains
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: GBT: As above

3.2-R9.2 polarization leakages
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1
Comments: GBT

3.2-R9.3 polarization angle calibration
Levels: 1
Functionality: I1

Useability: I1
Documentation: I1
Comments: GBT: This basic capability has just been put into daily, but I haven't tested it since I only have the latest stable on my laptop.

3.2-R9.4 antenna-dependent bandpasses
Levels: 1
Functionality: E/high
Useability: A
Documentation: E/high
Comments: JEH: Should have control on how to interpolate between BP solutions.

3.2-R10 Baseline dependent corrections shall be supported for quantities including:

3.2-R10.1 closure errors
Levels: 1
Functionality: I1
Useability: I1
Documentation: I1
Comments: GBT: No tools are present to plot, flag or calibrate closure phases or amplitudes.

3.2-R10.2 baseline-dependent bandpasses
Levels: 1
Functionality: N
Useability: N
Documentation: N
Comments: JEH: Not present. See comment at beginning of section about documentation.

3.2-R10.3 WVR corrections to subsets of baselines (if available)
Levels: 2
Functionality: I2
Useability: I2
Documentation: I2
Comments: GBT: Not present

3.2-R11 Gain corrections will be made based on differences between observed and modeled data quantities, possibly with iteration (e.g. self-calibration and determination of gains using calibration sources). Where solutions are discrepant or poor, automatic editing shall be possible.

Levels: 1

Functionality: A

Useability: A

Documentation: A

Comments: JEH: This is how the ME formalism works. Self-cal is available, and I think the documentation is ok. The editing requirement doesn't belong here, since it is covered in 3.1.R9.

3.2-R12 Calibration quantities (possibly stored in tables or data structures) shall be transferable between sources and/or frequency bands, after any necessary interpolation, extrapolation or smoothing. This will be the primary method of phase calibration transfer using fast-switching between source and calibrator.

Levels:

Functionality:

Useability:

Documentation:

Comments: JEH: I didn't test this. Documentation says its possible to transfer the solutions, but I don't see how. I can find nothing on interpolating solutions. ALMA audit suggests that it might be sufficient to transfer the model from one dataset to another, and calibrate on that, but that is not straight forward. So E1/high?

3.2-R13 Determination of the time-variable, complex bandpass using calibration source observations, and transfer to target sources, shall be simple and robust.

Levels: 1,2

Functionality: A1,N2

Useability: A1,N2

Documentation: E1,N2

Comments: JEH: Level 1: You can calculate BP solutions at different times, but how they are applied to the data are not described (nearest solution, according to GM). How phases and amplitudes are averaged is not described or under user control. Level2: want more interpolation options.

3.2-R14 Determination of polarization calibration quantities such as leakage (D-term or Jones matrix) and complex gain difference shall be an integral part of the Package, with the capability of performing full matrix calculations.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: GBT

3.2-R15 Calibration and display of calibration quantities shall be largely graphical and intuitive.

Levels: 1

Functionality: A1,E2

Useability: A1

Documentation: E1

Comments: GBT

3.2-R16 Determination of, correction for, examination of, and flagging based on closure errors shall be straightforward to carry out within the calibrator tool.

Levels: 2

Functionality: I2

Useability: I2

Documentation: I2

Comments: GBT: Not present

Comments: JEH: So N, not I right?

3.2-R17 Redundancy (e.g. same, similar, or crossing baselines) shall be used wherever possible to increase accuracy of or to check calibration solutions. Editing based on this comparison shall be possible.

Levels: 2

Functionality: I2

Useability: I2

Documentation: I2

Comments: GBT

Comments: JEH: N, not I right?

3.2-R18 An integrated deconvolution, self-calibration, and editing/filtering tool shall be available, especially for novice users with data taken in commonly used modes.

Levels: 2

Functionality: I2

Useability: I2

Documentation: I2

Comments: GBT

3.2-R19 The ability to self-calibrate based on catalog source positions shall be possible.

Levels: 2

Functionality: I2

Useability: I2

Documentation: I2

Comments: GBT

Comments: JEH: I think this may be possible via the synthesis.componentmodels package, but its not clear. See comments to 3.2-R6.

3.2-R20 Solving for calibration solutions can be time consuming. The calibrator tool should provide runtime estimates to allow the user advance warning if a process is likely to take more than 10 minutes. The user should have the option to cancel the start of processing in the time estimate is too long.

Levels: 2

Functionality: I2

Useability: I2

Documentation: I2

Comments: GBT

Comments: JEH: N, not I right?

3.3 Apriori Calibration

Apriori calibration is based on data provided by, e.g., the VLA online system or weather inputs. To perform the calibration, AIPS++ must be able to read and interpret ancillary data:

3.3-R1 The Package shall be able to read VLA data obtained in special observing modes, including holography, baseline data, and tipping scans.

Levels: 2

Functionality: N2/med

Useability: N2

Documentation: N2

Comments: CJC: this capability does not currently exist in aips++.

3.3-R2 Antenna positions shall be calculated from data obtained in standard VLA baseline observations.

Levels: 2

Functionality: N2/med

Useability: N2

Documentation: N2

Comments: CJC: this capability does not currently exist in aips++.

3.3-R3 The Package shall be able to calculate gain-elevation curves from VLA data obtained in standard observing modes.

Levels: 2

Functionality: N2/med

Useability: N2

Documentation: N2

Comments: CJC: this capability does not currently exist in aips++.

Apriori calibration itself has further requirements:

3.3-R1 Apriori flags attached to an input data set should be read and properly handled.

Levels: 1,2

Functionality: A1,I2/low

Useability: A1,N2/low

Documentation: I1,N2/low

Comments: CJC: Level 1: Currently VLA online system flags are handled in a reasonable way. There is no documentation on what level of VLA online flags are included in aips++ data, but just a short sentence is all that is needed to fix this. Level 2: There is no control over whether to keep data at various levels of online flagging.

3.3-R2 Apriori calibration attributes attached to an input data set should be read and properly handled.

Levels: 1

Functionality: N1/high

Useability: N1

Documentation: N1

Comments: CJC: Various apriori data, such as Tsys or weather information, are not currently available in aips++ measurement sets.

3.3-R3 Corrections to the data based on a priori known values shall be possible, including:

3.3-R3.1 antenna position errors;
Levels: 1
Functionality: I1/high
Useability: I1
Documentation: I1
Comments: STM: Not currently available, though in development (target Jan 2003).

3.3-R3.2 antenna gain-elevation curves;
Levels: 1
Functionality: I1/high
Useability: I1
Documentation: I1
Comments: STM: Not currently available, except by custom glish script (available at NAUG website). In development (target Jan 2003).

3.3-R3.3 atmospheric optical depth with elevation;
Levels: 1
Functionality: I1/high
Useability: I1
Documentation: I1
Comments: STM: Not currently available, except by custom glish script (available at NAUG website). In development (target Dec 2002).

3.3-R4 The Package shall be able to derive atmospheric opacity at various VLA observing frequency bands based on:

3.3-R4.1 tipping scan data
Levels: 2
Functionality: N2/med
Useability: N2
Documentation: N2
Comments: CJC: Not currently available in aips++.

3.3-R4.2 measured system temperatures
Levels: 2
Functionality: N2/med
Useability: N2
Documentation: N2
Comments: CJC: Not currently available in aips++.

3.3-R4.3 measured atmospheric parameters at the VLA site: temperature, pressure, humidity
Levels: 2
Functionality: N2/med
Useability: N2
Documentation: N2
Comments: CJC: Not currently available in aips++.

3.3-R5 Atmospheric modeling shall provide the conversion factors between WVR data and the water contribution to the astronomical phase in the astronomical bands if WVR data are available.

Levels: 2

Functionality: N2/low

Useability: N2

Documentation: N2

Comments: CJC: These capabilities do not currently exist in aips++, but then neither do WVR data.

3.3-R6 Output from the atmospheric monitoring instrumentation (e.g., phase information from the atmospheric phase interferometer, or WVR, if available), provided in VLA format, shall be importable to the calibration software, for example to be used in flagging.

Levels: 2

Functionality: N2/low

Useability: N2

Documentation: N2

Comments: CJC: These capabilities do not currently exist in aips++.

3.3-R7 The Package shall be able to calculate and apply phase corrections and Faraday rotation corrections based on ionospheric TEC measurements.

Levels: 1

Functionality: N1/high

Useability: N1

Documentation: N1

Comments: CJC: this capability does not currently exist in aips++.

4 Imaging
Debra (general and mosaicing), Ed (general and wide-field), Bryan, Crystal and Frazer (wide-field, low frequency)

Applicable tools for imaging include: `imager` (e.g. functions to do CLEAN, MEM, & MSC deconvolution, wide-field imaging, & continuum subtraction), `viewer` interface. Applicable documentation should include Getting Results, volumn 2, chapters 2: Imaging, Deconvolution and Self-calibration & 3: Wide-field imaging.

Comments: JEH: Use: I would grade the Usability of the imager as I for everything. Its these damn "set" functions. By and large, these are not functions, but inputs. A much better model (IMO) would be to have the functions that actually depend on the data selection, weighting scheme, etc, embedded as roll-up menus in the relevant functions. If the roll-down/up could work fast, then users could simply pop them down to look at them to see if they want to change them. This probably impacts the whole bigger question of GUI redesign, but it deserves some mention here. It's just not clear (especially to novices) what functions require what setfunctions to have been run previously.

4.1 Interferometer Imaging Requirements

Because VLA is inherently a multi-channel instrument, spectral cube mapping shall be built in as the primary mode from the beginning. Also, due to the high volume of data that can be produced by the VLA, it is imperative that the imaging and deconvolution tools in the Package be user-friendly, efficient, and flexible. This is the workhorse of the Package as far as most users will be concerned, and suitability and success of the Package will be judged with this in mind.

4.1-R1 Imaging of data taken from any combination of VLA exported data, or VLA archive data must be provided. **An updated list of supported data and formats will be provided with each AIPS++ release.**

Levels: 1,2

Functionality: A1,I2/high

Useability: A

Documentation: I/low

Comments: DSS: **Level 1:** Imaging of standard VLA data formats is supported. **Level 2:** Two epochs of VLA data that have been filled directly in AIPS++ can't be imaged unless they are first converted to FITS files, reconverted back into AIPS++ Measurement Sets, then concatinated. **Documentation:** There is no list of supported data and formats available on the Web.

Levels: 1

Functionality: E (cannot do file $>$ 2.4 Gb)

Useability: E (Read and interpret files FG, TY, GA, etc.)

Documentation: A

Comments: EF: comments and grades above

4.1-R2 Efficient selection of subsets of the imaging data must be provided.

Levels: 1

Functionality: I1/med

Useability: A

Documentation: A

Comments: DSS: **Level 1:** Random channel selection during imaging is not available (useful to select line-free channels on both sides of the line). Field_IDs should allow source names to make them more user-friendly. The rest of the selection criteria are adequate. If you want something exotic, you have to resort to msselect TAQL strings. But forcing users to learn TAQL is not is not optimal and will (I think) cause more problems that it is worth.

Comments: JEH: I would request that FIELD_ID allow a "From ms" option from the wrench menu for this and all tools (calibrator too). I could imagine this being a plot of source name vs. time, and you would select whatever sources you want with the cursor; like antennae selection works.

Levels: 1

Functionality: E (more defaults, no TAQL, setimage, setdata overlap)

Useability: A

Documentation: A

Comments: EF comments and grades above

4.1-R3 Provision must be made for the utilization and development of a variety of imaging, deconvolution, and analysis algorithms, including:

4.1-R3.1 raw ("dirty") images with selectable weighting (natural, uniform, Briggs robust)
Levels: 1,2
Functionality: A1,I2/med
Useability: A
Documentation: I/high
Comments: DSS: **Level 1** Dirty images can be made of single fields using selectable weighting with the makeimage function. It is easy to run. **Level 2** Makeimage doesn't work with mosaic fields. **Documentation:** Documentation doesn't mention that makeimage uses the weights settings (if you have set them up with imager.weights) and doesn't mention the limitation that mosaics aren't handled. Documentation is also sparse and has mistakes: a description of `compleximage` is not given, examples say to run makeimage with imgr.image but it should be imgr.makeimage. Of course, you can always create a dirty image by cleaning it for just one iteration and this is the only way you can make a mosaic dirty image. But this is also not mentioned in the documentation.
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: EF comments and grades above

4.1-R3.2 residual images after model subtraction
Levels: 1,2
Functionality: A1,A2
Useability: A
Documentation: E
Comments: DSS: **Documentation:** Ref Manual documentation says that 'residual' is the name of the residual image but there is no further explanation. There should be a brief mention of what a residual image is and what the dimensions are (e.g. only the size of the cleaned region).
Levels: 1
Functionality: A
Useability: A
Documentation: A
Comments: EF comments and grades above

4.1-R3.3 single-scale CLEAN (Hogbom, Clark, Cotton-Schwab (wfclark))
Levels: 1
Functionality: A1

Useability: A
Documentation: A
Comments: DSS
Levels: 1
Functionality: A
Useability: E (too slow)
Documentation: I (Need more discussion about the differences)
Comments: EF comments and grades above

4.1-R3.4 maximum entropy/emptiness method (MEM)
Levels: 1
Functionality: E1
Useability: E
Documentation: A
Comments: DSS: **Level 1** MEM exists but it doesn't have the capability to do interactive cleaning. Thus, you have to make the mask separately, deconvolve, check the image, create a new mask, deconvolve again. It is not nearly as user-friendly as as the CLEAN deconvolution tool.
Levels: 2
Functionality: I (Could not get this to execute satisfactorily
Useability: I
Documentation: I
Comments: EF comments and grades above

4.1-R3.5 non-negative least-squares (NNLS)
Levels: 2
Functionality: U
Useability: U
Documentation: U
Comments: DSS: NNLS is a relatively new deconvolution algorithm that is still in the development stage. Extensive testing has not been done. It is not yet available for mosaics, and interactive masking is not an option.
Levels: 2
Functionality: I (Could not get this to execute satisfactorily
Useability: I
Documentation: I
Comments: EF comments and grades above

4.1-R3.6 multi-scale CLEAN
Levels: 1,2
Functionality: A1,I2/med
Useability: A
Documentation: A
Comments: DSS: **Level 1:** Multi-scale CLEAN on single fields works well in the testing I've done (2 datasets). Good documentation is available (and needed). **Level 2:** MS-CLEAN only cleans a quarter of the image. If a mask larger than a quarter of the image is set, then only the quarter starting at the bottom left corner is used - this is not a useful default because it can produce crazy images without error messages. Instead, the algorithm should warn the user about the problem and suggest that they use a larger image size if there is emission is outside of the inner quarter. Since MS-CLEAN takes signficantly longer than CLEAN, the algorithm should be optimized for speed, if at all possible.
Comments: JEH: I disagree that the documentation on multiscale CLEAN is good. The documentation in the URM is much too sparse; How are components cleaned with different resolutions added back in? With a guassian of the appropriate size, or with a single gaussian set by setbeam, or are delta functions simply added to cells in the model image? If so, how are they subtracted from the data to make the residual image, and what are the units of the residual and restored

maps? All appropriate multiscale documentation in the URM should point to the appropriate section in GRvol2.
Comments: JEH: The multiscale documentation in GRvol2 is confusing. What is the difference between imagermultiscale and imager.clean with algorithm=multiscale? Upon careful reading, it seems that imagermultiscale is an particular implementation of multiscale clean (and it should spell out what conditions it is best for; since it cleans largest scales first then smaller, presumably you would use it if you have a lot of flux in large scale structures and relatively little flux on the largest baselines. One could imagine cases in which the opposite is the case). If a novice user came in wanting to do a multiscale clean, I think the GR documentation implies they should run imagermultiscale, instead of imager.clean with alogrithm=multiscale.
Levels: 2
Functionality: I (Could not get this to execute satisfactorily)
Useability: I
Documentation: I
Comments: EF comments and grades above

4.1-R4 Interactive graphical selection of deconvolution region masks shall be available for all deconvolution algorithms.

Levels: 1

Functionality: I1/med

Useability: A

Documentation: E/high

Comments: DSS: **Level 1:** Interactive masking is available for CLEAN-based algorithms. It is not available in MEM or NNLS. Interactive masks don't work with thresholds set. You also can't interactively set a mask on one spectral channel and then apply that mask to all channels. The only way you can do this is to interactively define a mask on one channel, stop the cleaning process, then use that mask as an input to all channels when CLEAN is restarted. **Documentation:** Interactive masking is only described in the VLA cookbook. It is not in http://aips2.nrao.edu/weekly/docs/cookbook/cbvol2/node3.html or in the Ref Manual tool help.

Levels: 1

Functionality: I (only available for imager.clean. should be in deconvol)

Useability: E

Documentation: I (can't find documentation for this)

Comments: EF comments and grades above

4.1-R5 Multiple input datasets shall be supported directly in the tools (e.g. concatenation of the data, although it should be supported, shall not be required).

Levels: 1

Functionality: N1/high

Useability: U

Documentation: U

Comments: DSS: Not possible at this time.

Levels: 2

Functionality: U (I think that concatenate is all that's needed for now)

Useability: U (No need to have multiple data set inputs to most tools)

Documentation: U

Comments: EF comments and grades above

4.1-R6 *uv* data with different coordinate systems, equinoxes, or projections shall be transformed, merged and compared appropriately.

Levels: 1

Functionality: U

Useability: U

Documentation: U

Comments: DSS: Since multi-epoch data can only be merged in concatination algorithms at this time, this requirement cannot be assessed in the imaging section.

Levels: 2

Functionality: U

Useability: U

Documentation: U (make part of concatenation)

Comments: EF comments and grades above

4.1-R7 *uv* data with different velocity definitions and velocity frames shall be transformed and merged correctly.

Levels: 1

Functionality: U

Useability: U

Documentation: U

Comments: DSS: Since multi-epoch data can only be merged in concatination algorithms at this time, this requirement cannot be assessed in the imaging section.

Levels: 2

Functionality: U

Useability: U

Documentation: U (make part of concatenation)

Comments: EF comments and grades above

4.1-R8 Imaging of direct polarization products (e.g. RR, LL, RL, LR) or Stokes polarization states (e.g. I, Q, U, V) must be selectable and interchangeable where possible given the data.

Levels: 1

Functionality: I1/med

Useability: A

Documentation: I

Comments: DSS: **Level 1:** Currently you can only choose I, IV, or IQUV to image using imager.setimage. **Use:** For the choices you have, it is very easy to select the data in imager.setimage. **Documentation:** (http://aips2.nrao.edu/weekly/docs/user/SynthesisRef/node152.html) says IQU is also available but it is not available in the GUI. Documentation doesn't mention the stokes parameter choices nor does it discuss how polarization images are processed (e.g. simultaneously or separately) or how setting different masks in I, Q, and/or U affects processing.

Levels: 1

Functionality: E (Need more options) (IQU, Q, QU, U, V

Useability: A

Documentation: A

Comments: EF comments and grades above

4.1-R9 There must be straightforward and seamless integration of data from multiple epochs and configurations.

Levels: 1

Functionality: U

Useability: U

Documentation: U

Comments: DSS: Multi-epoch/ configuration data can only be combined in concatination algorithms at this time, this requirement cannot be assessed in the imaging section.

Levels: 1

Functionality: E (most problems arise from AIPS not supporting proper AN files)

Useability: A

Documentation: A

Comments: EF comments and grades above

4.1-R10 Subtraction of the continuum level from spectral data is required, in both the Fourier and image domain. In the case of *uv*-plane subtraction, flexible setting of the frequency channel ranges for the calculation of the continuum level and selection of the type of fitting (polynomial, spline) should be available in both GUI and CLI.

Levels: 1

Functionality: N1/high

Useability: I/high

Documentation: U

Comments: DSS: **Level 1:** *uv* continuum subtraction is not available (but should be soon). It is possible but no easy to subtract continuum in the image plane. **Use:** To remove the continuum in the image plane, you have to use complicated Glish scripts (as described in the VLA Cookbook data analysis section) to select line-free channels on both sides of the line. This, also, will be available very soon. **Documentation:** Although there is some documentation in the VLA cookbook about how to do image plane continuum subtraction, it is temporary at best and will change as soon as the new code is available for testing.

Comments: JEH: I think this is the same as requirements 5-R2,R3 in the Analysis section. DSS's comments should be transfered there (and I have done so in this version of the document). The present uv continuum scripts, and also UVLSF I believe, overwrite the CORRECTED data column. This sounds dangerous. Is anything done the MODEL column? You can imagine that you might want two model columns for continuum subtracted uvdata - one channel independent (for the continuum), and one channel dependent (for the deconvolved line data).

Levels: 1

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.1-R11 There shall be the choice of FFT and DFT imaging (especially for small datasets).

Levels: 1

Functionality: N2/low

Useability: U

Documentation: U

Comments: From the ALMA requirements: STM: There does not appear to be this option in `imager`, e.g. in `imager.setoptions` only the gridded Fourier Transform machine gridft can currently be specified.

Levels: 2

Functionality: N

Useability: N

Documentation: N

Comments: EF comments and grades above

4.1-R12 Interactive control of the cleaning process shall be possible, including stopping cleaning at any time, prolonging the cleaning after the maximum number of iterations is reached, and continuing to clean down to a specified threshold with out changing the clean mask.

Levels: 1

Functionality: I1/high

Useability: A

Documentation: E

Comments: DSS: **Level 1:** Interactive control is only available in CLEAN-based deconvolution (not MEM). In CLEAN, you can stop at any time, but you cannot prolong the cleaning after the max iterations is reached. Cleaning doesn't work right now with the threshold set so this function cannot be tested. **Documentation:** How to specify an interactive mask should be in the on-line tool help. It is not. It is only in the VLA cookbook.

Levels: 1

Functionality: E (Should be also generalized in deconvolver)

Useability: E (A bit clunkly to use)

Documentation: E (Hard to fine documentation on interactive mask)

Comments: EF comments and grades above

4.1-R13 Continuation of a stopped cleaning process shall be possible and should make use of the intermediate clean model produced before stopping.

Levels: 1

Functionality: A1

Useability: A

Documentation: A

Comments: DSS: If you want to restart a CLEAN process, set the model input to the model that was previously produced by CLEAN.

Levels: 1

Functionality: A (This is present. Just continue cleaning)

Useability: A

Documentation: E (not obvious from documentation)

Comments: EF comments and grades above

Comments: STM: This req is implied in the previous one - delete req.

4.1-R14 Application of a *uv* taper should be supported (e.g. filtering)

Levels: 1

Functionality: A1

Useability: E

Documentation: E

Comments: DSS: **Use/Documentation:** Tapering is supported but it is called filtering - a non-radio astronomer word. This makes it confusing to figure out how to do a simple taper. Tapering should be in the weights tool and be called 'taper' - not filter.

Levels: 1

Functionality: E (not a major point. just generalize filtering)

Useability: E

Documentation: E

Comments: EF comments and grades above

4.1-R15 Imaging is often time consuming, the imaging tool should provide run-time estimates to allow the user advance warning if a process is likely to take more than 10 minutes. The user should have the option to cancel the start of processing if the time estimate is too long.

Levels: 1

Functionality: N1/med

Useability: U

Documentation: U

Comments: DSS: Not available at this time.

Levels: 1

Functionality: E (This is handled by some above criteria)

Useability: E

Documentation: E

Comments: EF comments and grades above

4.2 Wide-Field Imaging Considerations

4.2-R1 Wide-field, multi-faceted imaging must be straightforward to set up and run.

Levels: 1

Functionality: E

Useability: E (slow in execution)

Documentation: E (facets should be an odd integer)

Comments: EF comments and grades above

4.2-R1.1 It should be possible to set up a central region of specified size which is imaged, e.g. the "fly's eye")
Levels: 2
Functionality: ?
Useability: ?
Documentation: ? (what does this mean?)
Comments: EF comments and grades above

4.2-R1.2 It should be possible to specify "outrigger" facets, either manually, or by reading in information from a catalog (e.g., NVSS, WENSS, etc...), or from multiple catalogs for frequency interpolation, and specifying facets at the locations of sources above a given flux density cutoff (accounting for the primary beam response).
Levels:
Functionality:
Useability:
Documentation:
Levels: 2
Functionality: E
Useability: E
Documentation: E
Comments: EF comments and grades above

4.2-R2 High-fidelity imaging of the entire primary beam in all Stokes parameters is the primary goal — therefore, incorporation of the polarized primary beam response of the array is required.

Levels: 2

Functionality: I

Useability: I

Documentation: I

Comments: EF comments and grades above

4.2-R3 It should be possible to stitch together all of the facets to form a single final image.

Levels: 1

Functionality: I

Useability: I (should be able to glue contiguous images together)

Documentation: I

Comments: EF comments and grades above

4.2-R4 The package should know the location and sky brightness structure of the strongest low frequency sources ($>$ 100 Jy)

Levels: 2

Functionality: E

Useability: E

Documentation: E

Comments: EF comments and grades above

4.2-R5 It should be possible to subtract the contribution of strong sources in the far sidelobes of the primary beam from the *uv* data, accounting for a time variable primary beam response.

Levels: 1

Functionality: E (can be done using low resolution image and outlier fields)

Useability: E

Documentation: E

Comments: EF comments and grades above

4.2-R6 It should be possible to derive variable source differential refraction effects across the primary beam, and account for them properly in imaging. (this is a technique for imaging when the isoplanatic patch is small, via fitting a zernike polynomial across the primary beam for source location.)

Levels: 2

Functionality: E (Also needed for VLA beam squint variation.

Useability: E (tough to do. Can do snapshots and add later)

Documentation: E

Comments: EF comments and grades above

4.3 Mosaicing Considerations

Applicable tools for mosaic imaging include: `imager` (e.g. functions to do CLEAN, MEM, & MSC deconvolution), `viewer` interface, combining single dish (e.g. GBT) with VLA interferometer data. Applicable documentation should include Getting Results, volumn 2, chapter 4: Mosaicing (multi-field imaging).

4.3-R1 Linear mosaics of images should be supported

Levels: 2

Functionality: U

Useability: U

Documentation: E

Comments: DSS: There is a linearmosaic tool to linearly mosaic multiple images together. Its a good tool to have but there is no discussion in the documentation as to when this should be used. The Ref Manual description is sparse at best. Note: Extensive testing has not been done on this algorithm.

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R2 Multi-field mosaics with joint deconvolution should be supported

Levels: 2

Functionality: 2E

Useability: A

Documentation: A

Comments: DSS: **Level 2:** The Linear mosaic with an image plane weighting function (for constant noise across the mosaic) and joint deconvolution using an approximate PSF subtraction in iterative steps works well. It would be useful to have even a rough time estimate of how long gridding will take BEFORE the algorithm begins (with the option to change imaging parameters if it is going to take many hours). Also, multi-field MEM (emptiness & entropy) and multi-scale CLEAN for mosaics cannot treat stokes I & V simultaneously so this will slow the deconvolution down when processing circular polarization.

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R3 Careful cross calibration of the flux scales between VLA interferometric data and single dish data is required for high fidelity imaging. There must be tools to cross-check and correct the relative calibration between mosaics and different component observations.

Levels: 2

Functionality: U

Useability: U

Documentation: U

Comments: DSS: You can correct the relative calibration by scaling the input images before feathering (or before the GBT image is converted to *uv* data prior to combination when this becomes available). But, I don't think we really have a careful understanding of how to determine this scaling. It requires development by the GBT, by the AIPS++ group, and by the NAUG.

Comments: JEH: See Tim's Newsletter article.

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R4 There must be the ability to include "zero-spacing" values in imager.

Levels: 2

Functionality: I2/high

Useability: A

Documentation: A

Comments: DSS: **Level 2:** 1) MEM: Zero-spacing values can be controlled with the parameters `targetflux/constrainflux`. 2) CLEAN: Not available but needs to be. **Use/Documentation:** In MEM, it is relatively easy to specify the zero-spacing flux. Once this capability is added to CLEAN-based deconvolution, the use and documentation should be at about the same level as in MEM.

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R5 GBT data must be integrally supported by the Package and there must be a straightforward way to combine GBT and VLA data in the image plane (e.g. feathering):

Levels: 2

Functionality: I2/high

Useability: I

Documentation: I

Comments: DSS: **Level 2:** Feathering is available in imager.feather. There are only 3 GUI inputs but Glish CLI has an additional 4 parameters - Glish/GUI operations should be reconciled. The GBT beam is not currently in AIPS++ (my fault - I just sent the beam data to AIPS++) so you have to approximate the beam with a Gaussian before feathering (this should be fixed). Image.feather won't work unless you edit the header of the GBT data (AOCso03256) - this should be fixed. The

image.regrid function (very useful to get SD and VLA data on the same dimensions) is not working with BiCubic interpolation (AOCso03290) - this should also be fixed. **Documentation:** The documentation (http://aips2.nrao.edu/weekly/docs/user/SynthesisRef/node172.html) is not detailed enough for users to figure out how to use this tool effectively, e.g. many inputs are not described.

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R6 There must be a straightforward way to combine GBT and VLA data in the *uv* plane with selectable weighting (e.g. combining the data in imager and cleaning the resulting image with any chosen deconvolution algorithms).

Levels: 2

Functionality: N2/high

Useability: U

Documentation: U

Comments: DSS: **Level 2:** Not available at this time. There is no tool that can convert single-dish data to *uv* data (although the code is buried in other tools). Once *uv* data can be constructed from a single-dish image, and imager can accept two separate data sets, then each data set will have to be selectively weighted before FFT and joint-deconvolution. This is not currently possible. Ideally, it would be good to have an algorithm that determines the optimal weighting between SD and VLA data - this may be in the EVLA realm. Note: there is a current limitation that only one telescope voltage pattern can be applied to a a particular MeasurementSet. This needs to be addressed before GBT and VLA data can be combined in the *uv* plane.

Comments: JEH: See Tim's Newsletter article.

4.3-R7 Careful (polarized) primary beam correction and pointing correction is critical for high fidelity mosaic imaging and must be incorporated into the mosaicing algorithms.

Levels: 2

Functionality: I2/low

Useability: A

Documentation: E

Comments: DSS: Pointing corrections cannot be made with current VLA data. This is in the realm of the EVLA (e.g. not a VLA requirement). **Level 2:** However, beam-squint corrections are possible in AIPS++. They just haven't been tested and "could be backwards in orientation." As soon as a source with strong, known polarization is observed and processed with and without beam-squint corrections, we will know if the correction is correct. As soon as the functionality is tested, it will be easy to implement (setvp.dosquint=T). **Documentation:** Ref Manual for setvp should discuss how to use the parallactic angle increment for the squint application (setvp.parangleinc).

Levels: 2

Functionality: ?

Useability: ?

Documentation: ?

Comments: EF comments and grades above

4.3-R7.1 A set of **VLA standard beam images** shall be distributed with the Package, with updates available for download when appropriate.
Levels: 1
Functionality: A1
Useability: A
Documentation: E
Comments: DSS: **Documentation:** http://aips2.nrao.edu/weekly/docs/user/SynthesisRef says there are "several voltage patterns for the VLA, including the appropriate beam squint for each observing band." This should be explained in more detail (e.g. what they are and what bands they are appropriate for). This documentation exists in the `imager.vpmanager` description but this is not where users will look if they just want to use the defaults and make sure their inputs are correct.
Levels: 1
Functionality: E
Useability: E
Documentation: E
Comments: EF comments and grades above

4.3-R7.2 The user shall be able to specify the primary beam in a number of forms, both analytic and tabular, in addition to the VLA provided primary beam.
Levels: 1
Functionality: A1
Useability: A
Documentation: I
Comments: DSS: **Level 1:** A voltage pattern (VP) table can be made with the `imager.vpmanager` tool. `vpmanager.setcpbnumeric` function creates a VP from a tabular list. Analytic functions can be used to create VPs from polynomials and Gaussians. **Documentation:** descriptions at http://aips2.nrao.edu/weekly/docs/user/SynthesisRef/node220.html and relavent function links are minimal. Most inputs are not explained. Documentation should be expanded. This web site also gives an example of how to create a VP using the `setpbairy` function. The setpbairy documentation says this will crash imager because it "tickles an egcs bug" - this should be fixed.
Levels: 1
Functionality: E
Useability: E
Documentation: E
Comments: EF comments and grades above

4.3-R8 Image plane and uv-plane gridding parameters and interpolation shall be selectable based on desired image criteria (e.g. field-of-view, accuracy)

Levels: 2

Functionality: I2/high

Useability: E

Documentation: E

Comments: DSS: **Level 2:** Imaging parameters are set in `imager.setimage`, gridding parameters in `imager.setoptions` and weighting in `imager.weight`. The function `imager.advise` will suggest appropriate parameters such as cellsize, pixels, and facets. For mosaicing, there are 2 changes required: 1) tiled gridding becomes very slow once a max is reached in the image cell number (e.g. A array). Speed should be optimized for large-field tiled gridding (modified gridding in the filler may help but the imager.gridding code should be examined too); 2) Weighting of mosaic data should be done field-by-field before gridding - this has to be done manually now (very tedious with GUIs). It should be automatically done in the `imager.weight` function (e.g. set a parameter weight(mosaic=T) and weights will be calculated for each field separately, before image gridding. **Use:** Because of the

weighting problem, mosaicing in the GUIs can be time-consuming. **Documentation:** This should be upgraded to talk about tiled-gridding problems, how to weight the data from the GUI interface.

Levels: 2

Functionality: E

Useability: E

Documentation: E

Comments: EF comments and grades above

4.3-R9 The Package must be able to produce an image by combining data with different image centers.

Levels: 2

Functionality: A2

Useability: A

Documentation: A

Comments: DSS: Standard mosaic practice. No problem.

Levels: 2

Functionality: I

Useability: I

Documentation: I

Comments: EF comments and grades above

4.3-R10 The imaging tools shall allow the option for the mitigation of the effects of non-coplanar baselines and sky curvature.

Levels: 2

Functionality: I2/med

Useability: A

Documentation: A

Comments: DSS: There are wide-field Hogbom, and Clark algorithms in the `imager.clean` function. Aliasing control is in `imager.setoptions`. Wide-field options are not available for MEM, multi-scale CLEAN, or NNLS algorithms. It does not appear to be an option to mosaic very large fields and apply wide-field imaging corrections in a mosaic image. This will be needed to mosaic low-frequency VLA data.

Levels: 1

Functionality: A (This is what normal wide field imaging does)

Useability: A

Documentation: A

Comments: EF comments and grades above

5 Data Analysis
Crystal, Walter, Mark (all), Bryan (uv fitting)

Applicable tools for data analysis include: `viewer` (e.g. statistics function), `image` (e.g. profile fitting, regridding, moment maps), & figure generation functions (e.g. skycatalogs, annotate function in viewer). Applicable documentation should include Getting Results, volume 1, chapters 3: Display of data and 4: Image analysis.

5.1 General Analysis Requirements

5.1-R1 Seamless transformation between image-plane and uv-plane analysis is necessary:

5.1-R1.1 Analysis based on goodness-of-fit to models, in both uv-plane (for interferometry) and image plane, shall be available.
Levels:
Functionality:
Useability:
Documentation:
Comments: CLB: For interferometry I can imagine this means something like, for example, VPLOT in aips where you can plot a self-cal solution on top of the raw data. Some of this can be done in msplot, but this requirement is too vague.

5.1-R1.2 Fourier transform of images between angular and uv domains shall be available.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: Fourier transforms both ways are possible. `imager.ft()` transforms from angular to uv domain at the UV values sampled in a given MS. Additionally, an image can be FFTed with `image.fft()`. `imager.makeimage()` can be used to go the other way, making a dirty image.

5.1-R2 Spectral line image plane continuum subtraction capabilitiy is required:

5.1-R2.1 Polynomial and linear continuum fitting over multiple spectral windows shall be supported
Levels: 1
Functionality: I1/high
Useability: I1/high
Documentation: A1
Comments: CLB: Linear subtraction is possible through a fairly long process of creating a region from the line free channels and then using the `imS.moments` tool to average them into a continuum, then you can subtract this pseudo continuum from your line with imagecalc. A function needs to be created that can do all this in one easy step and also allowing for polynomial subtraction.
Comments: JEH: this requirement also appears as 4.1-R10, so I repeat DSS's comments here:
Comments: DSS: (copied by JEH from S4.1-R10) **Level 1:** It is possible but not easy to subtract continuum in the image plane. **Use:** To remove the continuum in the image plane, you have to use complicated Glish scripts (as described in the VLA Cookbook data analysis section) to select line-free channels on both sides of the line. This, also, will be available very soon. **Documentation:** Although there is some documentation in the VLA cookbook about how to do image plane continuum subtraction, it is temporary at best and will change as soon as the new code is available for testing.

5.1-R2.2 Flagging based on continuum subtraction based on image statistics (e.g. median filtering) shall be supported
Levels: 1

Functionality: N1/med
Useability: U1
Documentation: U1
Comments: CLB: This capability doesn't yet exist.

5.1-R2.3 Fourier analysis of standing waves and their removal from spectra shall be available.
Levels: 2
Functionality: N2/low
Useability: U
Documentation: U
Comments: CLB: This is more an issue for EVLA and single dish.

5.1-R3 Spectral line UV plane continuum subtraction capabilitiy is required:

5.1-R3.1 Polynomial and linear continuum fitting over multiple spectral windows shall be supported
Levels: 1
Functionality: E1
Useability: E1
Documentation: I1/high
Comments: CLB: As far as I can tell this works. Some further testing is probably needed. The only documentation so far is in the user ref. manual and needs elaboration. The location of this function in the ms tool is pretty obscure. There needs to be a way to simultaneously output both the continuum subtracted line and the continuum as "corrected data". Currently you can only do one or the other. You have to use calibrator.correct to get back the uncorrected line data (if you choose this option). Perhaps both the continuum subtracted line and continuum should be made into new datasets.
Comments: JEH: this requirement also appears as 4.1-R10. UVSUB exists as a function (subcontinuum.g) that Kumar sent around (no docmentation). ms.uvlsf allows polynomial fitting. As CLB notes, I believe that both overwrite the CORRECTED data column. This sounds dangerous. Is anything done the MODEL column? To elaborate on her suggestion, you can imagine that you might want two model columns for continuum subtracted uvdata - one channel independent (for the continuum), and one channel dependent (for the deconvolved line data).

5.1-R3.2 Flagging based on continuum subtraction based on image statistics (e.g. median filtering) shall be supported
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1
Comments: CLB: This capability doesn't yet exist.
Comments: JEH: I think "image" in "image statistics" is a typo (should be UV data). In both msplot and viewer, you can flag continuum based on data-model, or data/model, which allows you to get outliers. But you cannot flag on statistics. autoflag is supposed to allow flagging on floating median over channels, but I couldn't get it to work.

5.1-R4 Translation between various astronomical quantities and units shall be available, including:

5.1-R4.1 flux density — Jy, mJy, μJy, mag, erg s^{-1} cm^{-2}
Levels: 1
Functionality: E1
Useability: A1
Documentation: E1

5.1-R4.2 temperature — K, mK, eV (Rayleigh-Jeans, Planck)
Levels: 1
Functionality: I1/low
Useability: A1

Documentation: E1

5.1-R4.3 surface brightness — Jy/beam, Jy/sr, MJy/sr, mag/arcsec^2

Levels: 1

Functionality: I1/low

Useability: A1

Documentation: E1

5.1-R4.4 frequency — Hz, MHz, GHz, cm^{-1} (wave number)

Levels: 1

Functionality: I1/low

Useability: A1

Documentation: E1

5.1-R4.5 velocity — km/s, m/s, z (redshift)

Levels: 1

Functionality: E1

Useability: A1

Documentation:

Comments: JEH: There is a problem with the DOPPLER subtable not being filled, resulting in the rest frequency being entered as 1 kHz. As a result, you can't get viewer to plot in km/s, and moment does not run properly (see bugs AOCso03764 and AOCso04273). So while it LOOKS like all the velocity conversion stuff is there, there is something fundementally wrong with the handeling of the doppler table. Sent this to STM for incorporation into Sec2 requirements.

5.1-R4.6 wavelength — m, cm, mm

Levels: 1

Functionality: A1

Useability: A1

Documentation: E1

Comments: WB: unit conversion can be done with quanta (e.g., `dq.convert(dq.quantity(1.0, "m2"), "cm2")` results in `[value=10000, unit=cm2]`). This is fine for direct unit conversions but doesn't apply to conversions requiring a reference such as redshift-velocity mapping. Many of these conversions can be done through measures. Many of the high level tools (e.g., profile fitter and viewer) allow choice of units, including redshift or entry of values in a variety of units. I have seen no evidence that magnitudes (what a silly system) or eV are anywhere supported.

5.1-R5 The astronomer must have the capability to develop their own tools or tasks, with easy access to data and images, and straightforward interface with the Package.

Levels: 2

Functionality: E2

Useability: E2

Documentation: I2/med

Comments: WB: Glish allows scripting language access to tables, upon which measurement sets and images, among other structures, are based. While it takes some learning, gaining familiarity with glish is no more difficult than for any other language. There is plenty of documentation full of examples.

Comments: JEH: There is a seemingly straightforward way to wrap glish around user-supplied (pre-compiled?) fortran or c programs (Joe gave us a demo on this), but I can find no documentation on it. I haven't used it, though, so can't comment on useability. I think it is generally not straightforward to get simple ascii output e.g. from a single pixel along the spectral axis, or from image statistics. One would need to learn how to use glish records. I don't think that is as easy as many other scripting languages. This is probably a contentious issue, but I find the use not that intuitive so would not grade it as A2 here. Documentation is also lacking - there should be a chapter on these capabilities in GRvol2, with links to the appropriate sections of the glish documentation.

5.2 Spectral Line Analysis

The ease and flexibility of spectral line analysis for VLA data is a driver for the Package. Automatic and user-controlled fitting routines are needed. Note that there is necessarily some overlap with the image cube analysis, but here we concentrate on issues relevant to traditional 1-D spectra.

5.2-R1 Automatic and user-controlled measurement of line parameters shall be available.

Levels: 1,2

Functionality: E1/N2/low

Useability: A1/U2

Documentation: A1/U2

Comments: WB: GUI-based fitting is set as a level 1 requirement, and automatic (automated?) fitting as level 2. The `imageprofilefitter` tool is a nice GUI for fitting line profiles. It appears that completely automatic fitting is not yet possible.

5.2-R2 It shall be possible for the user to specify a velocity or frequency windowing region for line fitting.

Levels: 1

Functionality: I1/med

Useability: I1/med

Documentation: U1

Comments: WB: The tools themselves do not seem to support explicit channel ranges. A sub-image made by selecting the appropriate channel range could be used to achieve the results, however interactively setting frequency range to fit is desired.

5.2-R3 Available line fitting parameters and profiles shall include:

5.2-R3.1 Gaussian line parameters (central and integrated intensity, line width, line center) for single or multiple lines
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1
Comments: WB: only line center amplitude appears to be produced, not integrated intensity. Multiple lines are supported.

5.2-R3.2 Damping profiles (Lorentzian)
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1
Comments: WB: only Gaussian forms are currently supported.

5.2-R4 Setting of fit constraints (e.g. spacing for multiple lines) shall be available and flexible (i.e. by GUI or by parameter list).

Levels: 2

Functionality: N2/med

Useability: U2

Documentation: U2

Comments: WB: multiple line constraints are listed in the "Future Improvements" section of the documentation of `imageprofilefitter` .

Comments: JEH: Any of the fit parameters may be fixed via the GUI in imageprofilefitter. Cannot specify via parameter list.

5.2-R5 Export of fit results in ASCII-format is needed.

Levels: 1

Functionality: N1

Useability: N1

Documentation: U1

Comments: WB: The fit parameters can be extracted into glish with `imageprofilefitter.getstore()`. This is fairly clunky. Ideally a 4 column ascii file could be produced with columns for freq, raw profile, fit profile, and residual profile.

5.2-R6 Export of publication quality postscript format of spectral line profiles is necessary. The user should be able to choose data, fitting profiles, axis scaling, and annotations.

Levels: 1

Functionality: N1/high

Useability: A1

Documentation: U1

Comments: WB: postscript plots can be made, though they are not publication quality. There is almost no configurability in the plotting and the output often has overlapping words, coordinate axes with rediculous numerical labels, and the linewidth cannot be chosen. Also, there is no control over the color. The lack of options make this very easy to use. I suppose documentation will come with the features.

Comments: JEH: While the plot cannot be directly edited from imageprofilefitter, but it can be saved as a glish plot, read into pgplotter, and then edited in any way desired (axis changed, colors changed, line type changed, etc). But a comment on that: the pgplotter editing tool is too atomic - every individual element must be edited individually. I don't see a big problem with this in terms of profile fitting. Its more of a problem with plotcal, when if you rescale the plot, the antenna labeling falls of the plot, and you would have to edit each labels x,y position to get it to plot properly.

Comments: JEH: I would add a requirement to export line profiles, combined fit, and residuals as simple ascii datafiles.

Comments: STM: Isn't this the previous requirement?

5.3 *uv* Analysis

The following capabilities should be supported:

5.3-R1 Derive visibility statistics (mean, rms [complex], with data selection, as function of channel, polarization, time, etc...)

Levels: 2

Functionality: N2/low

Useability: U2

Documentation: U2

Comments: WB: I don't think this functionality exists yet.

Comments: JEH: You can basically do this using the statistics tool of the viewer when viewing a ms, but data-selection is not allowed. If the msplot tvflg-like viewing had a statistics tool, that could work.

5.3-R2 Simple models should be available to fit to the uv data:

5.3-R2.1 Point source
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R2.2 Gaussian (circular or elliptical)
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R2.3 Limb-darkened disk (circular or elliptical)
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R2.4 Optically thin sphere
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R2.5 Ring
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R2.6 Cone
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: UV modeling is not yet available.

5.3-R3 Models should have the ability to fit for or hold constant the following parameters:

5.3-R3.1 Flux density
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R3.2 Position (2 axes)
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R3.3 Size (2 axes if non-circular)
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.3-R3.4 Position angle (if non-circular)
Levels: 2
Functionality: N2/low

Useability: U2
Documentation: U2

5.3-R3.5 Limb-darkening parameter
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: UV modeling is not yet available.

5.3-R4 Simple models above should be able to be combined into a more complex model.

Levels: 2

Functionality: N2/low

Useability: U2

Documentation: U2

Comments: WB: UV modeling is not yet available.

5.3-R5 Models should be able to be plotted on top of visibility data

Levels: 2

Functionality: N2/low

Useability: U2

Documentation: U2

Comments: WB: UV modeling is not yet available.

Comments: JEH: Once models are in the MODEL column, viewer and msplot can plot data-model or data/model, but this is not the same.

5.3-R6 You should be able to import an image of a structure (model) and then fit for the flux density and position.

Levels: 2

Functionality: N2/low

Useability: U2

Documentation: U2

Comments: WB: UV modeling is not yet available.

5.4 Image Cube Analysis and Manipulation

5.4-R1 The Package shall support the construction and analysis of image cubes with a variety of axis choices, including:

5.4-R1.1 time
Levels: 1
Functionality: I1/med
Useability: I1/med
Documentation: I1/med
Comments: WB: it seems that time is not known to the image type. `image.adddegaxes()` has provisions for adding direction, spectral, linear, and stokes axes only. The basic image structure can however accommodate user defined axes, but with some additional work.

5.4-R1.2 position (e.g. general X,Y)
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1
Comments: WB: X and Y linear are supported, but I don't think a third (Z) linear coordinate is fully supported.

5.4-R1.3 Fourier axes (u,v)
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R1.4 channel (frequency, velocity, channel number, band, IF)
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R1.5 polarization
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R2 Basic cube rotation and transposition operations shall be available, including rotation not orthogonal to cube faces.

Levels: 1,2

Functionality: A1,N2/low

Useability: A1,U2

Documentation: A1,U2

Comments: WB: axis selection can be done in viewer, satisfying Level 1 requirements (orthogonal 90 deg rotations). Level 2 (arbitrary rotation) is not supported.

Comments: JEH: Can't do true transpose via viewer. e.g., have frequency decreasing to the right. Arbitrary rotations is a high priority - eg., for rotating galaxies along major axis for doing position velocity plots.

5.4-R3 It shall be possible to define multi-dimensional regions of the cube for further processing, both interactively and by specifying coordinates manually.

Levels: 1,2

Functionality: A1,N2

Useability: A1,N2

Documentation: E1,N2

Comments: JEH: I would say manual specification is level1, A1. Documentation should include link to newsletter article. Graphical specification is level 2, not there.

Comments: WB: simple 2-dim regions can be entered graphically, but more complex regions require manual input. Although there is documentation under "region" that is helpful, it takes a bit of searching to find what is needed. `drm.gui()` allows creation and editing capabilities for regions. `regionmanager.gui()` documentation is minimal, but enough to figure things out. It would be nice to be able to visualize a constructed region.

5.4-R4 Identification and reporting of image features (e.g. as determined in processing operations) shall be available, and interactive (where appropriate). Features shall include:

5.4-R4.1 output in pixel coordinates (e.g. row, column)
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1

5.4-R4.2 output in world coordinates (e.g. RA, Dec, Vel)
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1

5.4-R4.3 convertible to other supported coordinate systems (e.g. precessed to B1950)
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1

5.4-R4.4 exportable (e.g. as an ASCII file)
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1

Comments: WB/CLB: There is a function in the image tool called `findsources` that seems to fulfill these requirements. We couldn't figure out how to look at the component list or do these conversions although it seems possible. `fitsky` seems to be a more complex version (not tested). Serious ease of use issues after the component list is made (which is straightforward).

Comments: JEH: I am not sure what this refers to. Output from what processing? Why is exporting to ascii file graded as A1 in use and documentation? I find a simple thing like trying to print the intensity value for all frequencies at a given position in a cube impossible to find from the documentation, and very cumbersome and unintuitive when an aips++ expert tells you how to do it.

5.4-R5 The ability to extract same or lower-dimensional structures from higher-dimensional data cubes efficiently is required:

5.4-R5.1 Extraction of "cubical" sub-structures aligned with the original cube axes must be straightforward.
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: WB/CLB: This basic functionality is in `imagefromimage` and it works fairly well with an interactive display of the image and then selection of the region. It doesn't work well for image cubes since it only grabs the plane that is displayed instead of the whole cube. You could get around this if the region parameters were displayed and you could adjust the 3rd axis by hand, but it is currently unclear how to even see the region selected. Another way is to use `im2 := im1.subimage(outfile='im2', region=drm.box([50,50,1,1],[100,100,1,1]) )` by hand but this looses the interactivity.

5.4-R5.2 User selection of extraction criteria must be possible through the GUI as well as scriptable.
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1

Comments: WB:`drm.gui()` can be used for graphical entry, which is entirely analogous to the script form. Documentation is sufficient, but could use some concrete examples.

5.4-R5.3 User-selectable sub-structures with arbitrary orientation within the parent cube, with appropriate transformation or interpolation, shall be possible.
Levels: 2
Functionality: N2
Useability: U2
Documentation: U2
Comments: JEH: I would think the previous requirements - arbitrary image transformations followed by extraction - would suffice.

5.4-R5.4 Extraction of data structures based on standard database (e.g. SQL) queries shall be available.
Levels: 2
Functionality: A2
Useability: I2/med
Documentation:
Comments: WB/CLB: In principle you can do this with TAQL but without functions like an automatic source finder, it isn't clear how useful it is. The usability of TAQL is quite poor and cumbersome.

5.4-R6 The ability to collapse or integrate over sub-dimensions of data cubes to form "moments" is required. This shall be possible along any direction(s) in the cube aligned with the axes.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: WB: This is fairly straight forward. `image.moments()` can compute any moment about any image axis. regions/masks can be applied.

5.4-R7 Moments along arbitrary user-specified directions in the cube shall be possible.

Levels: 2

Functionality: I2/low

Useability: U2

Documentation: U2

Comments: WB: It appears that `image.moments()` does not allow moments to be made along directions not parallel to an image axis. However, the adept user could write a *simple* glish script to do this.

5.4-R8 The Package must have the capability of assembling lower-dimensional data structures into higher-dimension cubes.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: WB: there is an image constructor `imageconcat()` which takes as input any number of input images and the axis of concatenation. `image.adddegaxes()` may need to be used to make an extra axis, or the image's coordinate system can be changed.

5.4-R9 Blanking of pixels must be maintained through the analysis process:

5.4-R9.1 It must be possible to turn on and off different blanking (mask) levels, when blanking is set within the Package.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: The masking system is sophisticated. The default mask associated with an image can be changed. Masks can be combined. See aips++ note 223 on Masks. There is also a `maskhandler` in the image tool that allows manipulation of multiple masks.

5.4-R9.2 Blanking shall not be destructive, and the original pixel value is retained (if defined).
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R9.3 Interactive and automatic facilities for setting of any blanking parameters shall be provided.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: What is meant by blanking parameters?
Comments: CLB: I believe they just mean things like blank above a certain threshold value (automatic) or draw a region outside of which is blanked (interactive). You can certainly do the former and I believe the later. Many functions allow input of mask expression. There is also a `maskhandler` in the image tool that allows manipulation of multiple masks.

5.4-R10 A variety of image processing and filtering operations on the cube shall be available, including:

5.4-R10.1 Spatial smoothing and convolution, with kernels including:

5.4-R10.1.1 Uniform (box-car or top-hat) kernel
Levels: 1
Functionality: I1/med
Useability: A1
Documentation: A1
Comments: WB: `image.sepconvolve()` allows square/rectangular boxcar smoothing. Disc-shaped convolution would need to be done with `image.arrconvolve()` and a user-defined array of the appropriate size.

5.4-R10.1.2 Elliptical Gaussian kernel
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R10.1.3 Symmetric polynomial
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2

5.4-R10.1.4 User-supplied
Levels: 2
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: `image.convolve2D()` currently only allows gaussian convolutions, but it is clear that other types could be easily added. User-supplied kernels can be used with

`image.arrconvolve()`. A glish array is needed for this.

5.4-R10.2 Spectral smoothing and convolution, with kernels including:

5.4-R10.2.1 Uniform (box-car or top-hat) kernel
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R10.2.2 Gaussian kernel
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R10.2.3 Hanning
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R10.2.4 User-supplied
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

Comments: WB: Uniform(boxcar), Gaussian, and Hanning smoothing in the spectral axis can be performed with `image.sepconvolve()`. User-supplied can be done with `image.arrconvolve()`.

5.4-R10.3 Deconvolution
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: WB: The deconvolver tool does this (in the image plane). It would be convenient if a glish array could be passed as the PSF, but instead this tool requires that a PSF image resides on disk.

5.4-R10.4 Spectral regridding and averaging
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: see below

5.4-R10.5 Spatial regridding (ie. change cell size, image center, pad to larger size)

5.4-R10.5.1 user supplied
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

5.4-R10.5.2 to match template image
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: `image.regrid` can do this and the above specral regridding. Coordsys conversion can be done as well. This is very well thought out in aips++.

Comments: JEH: How does one do a simple magnify/subsample? It doesn't seem like regrid will allow this (since the cs will be different). You would want to do this e.g. when taking an image and simulating what it would look like when it is X times further away.
Comments: WB: I believe the answer for JEH is to simply modify the cs associated with an image, i.e., the pixel scales.

5.4-R10.6 Temporal resampling after processing
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1
Comments: WB: Time axes are not predefined (they should be) for images, however, the image.regrid function could still be used for this.

5.4-R10.7 Clipping
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: This is an application for LEL. Note that the requirement does not specify what happens to values that get clipped. LEL will allow the construction of a mask based on a clip level, or can treat the clip value as a saturate value with the `min( , )` or `max( , )` function used with one scalar argument.

5.4-R10.8 Windowing and boxing, with definitions of regions of interest recordable and passable for subsequent use
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: CLB: I think this means that you should be able to define a region to imagefromimage for example and use the same parameters again later. This is possible by defining regions using the region tool and then saving them.

5.4-R10.9 Arithmetical operations, including:

5.4-R10.9.1 addition of uniform level
Levels: 1
Functionality: E1
Useability: E1
Documentation: I1/med

5.4-R10.9.2 multiplicative scaling
Levels: 1
Functionality: E1
Useability: E1
Documentation: I1/med

5.4-R10.9.3 logarithm
Levels: 1
Functionality: E1
Useability: E1
Documentation: I1/med

5.4-R10.9.4 exponential
Levels: 1
Functionality: E1
Useability: E1
Documentation: I1/med

Comments: WB: See the note on LEL for this : aips++ note 223. The `imagecalc()` constructor for image can perform almost arbitrary arithmetic on images, arrays, and numbers.
Comments: JEH: I would grade usability and documentation as I. Note 223 is too long and not very readable; help on image.calc only has 2 examples. Functionality I think I would give a E - it would be nice to have a tool to construct LEL expressions.
Comments: WB: I changed usability and functionality to E and doc to I1/med as a compromize. This seems like one of those features that just needs to be learned (and needs more documentation!), but is not difficult once it is understood.

5.4-R10.10 Statistical operations over a defined region, including:

5.4-R10.10.1 mean
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1

5.4-R10.10.2 rms
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1

5.4-R10.10.3 standard deviation from mean
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1

5.4-R10.10.4 median
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1

5.4-R10.10.5 mode
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1
Comments: WB: `image.statistics` returns all of this information and allows the use of a region and/or a mask. To get some of the values, `robust=T` must be passed to the function.
Comments: JEH: I would grade documentation as E. Should include a link to documentation on how to extract information from a Glish record.

5.4-R10.11 Fourier and correlation operations on cube or cubical sub-regions, including:

5.4-R10.11.1 Fourier transform
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: `image.fft()` does this.

5.4-R10.11.2 power spectrum
Levels: 2
Functionality: A2
Useability: E2
Documentation: U2
Comments: WB: The 2-D power spectrum is just the square of the

FFT, which can be achieved by `image.fft(amp="amp.im")` and then `imagecalc(pixels="amp.im*conj(amp.im)")`. This should be easier to do. If the intent was the 1-D power spectrum, then there isn't an easy solution.

5.4-R10.11.3 autocorrelation
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: Not there?

5.4-R10.11.4 structure function
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: Missing?

5.4-R10.12 Scalar arithmetic between different regions (including treatment of masked regions and differently shaped regions), including:

5.4-R10.12.1 sum (difference)
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1

5.4-R10.12.2 product (division)
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1

5.4-R10.12.3 spectral index
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1

5.4-R10.12.4 median
Levels: 1
Functionality: N1/low
Useability: U1
Documentation: U1

5.4-R10.12.5 maximum (minimum)
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1
Comments: WB: Again, these (excepting median) can be computed with LEL and `imagecalc()`. If the regions are not on the same grid, then one must first be regridded, hence the E on usability. Not all of the above modes were tested, but all should behave the same way

5.4-R10.13 Construction and comparison of vector quantities in the cube, including:

5.4-R10.13.1 polarization (E) vector at each cube pixel
Levels: 1
Functionality: A1
Useability: A1

Documentation: A1

5.4-R10.13.2 rotation measure between different frequencies

Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB/CLB: These exist and appear to work well. Extensive testing was not done, particularly on rotation measure since we didn't have an appropriate data set.

5.4-R10.14 Vector arithmetic between different regions (including treatment of masked regions and differently shaped regions), including:

Comments: WB/CLB: As far as we can tell none of the requirements from here down exist as stand alone capabilities. The `imagepol` tool must be able to do some of these operations in order to create for example the polarized intensity. Nothing equivalent to `imagecalc` or `calc` seems to exist.

5.4-R10.14.1 sum (difference)

Levels: 1
Functionality:
Useability:
Documentation:

5.4-R10.14.2 dot product

Levels: 2
Functionality:
Useability:
Documentation:

5.4-R10.14.3 cross product

Levels: 2
Functionality:
Useability:
Documentation:

5.4-R10.15 Interpolation across blanked or masked regions

Levels: 2
Functionality:
Useability:
Documentation:

5.4-R11 Calculus on scalar and vector fields in cube, including:

5.4-R11.1 differentiation along paths

Levels: 1
Functionality:
Useability:
Documentation:
Comments: CLB: taking the derivative of a spectrum would be very useful for Zeeman work.

5.4-R11.2 integration over regions

Levels: 1
Functionality:
Useability:
Documentation:

5.4-R11.3 gradient of scalar fields

Levels: 2
Functionality:
Useability:
Documentation:

5.4-R11.4 divergence of vector fields
Levels: 2
Functionality:
Useability:
Documentation:

5.4-R11.5 curl of vector fields
Levels: 2
Functionality:
Useability:
Documentation:

5.4-R11.6 Laplacian of vector fields
Levels: 2
Functionality:
Useability:
Documentation:

5.4-R12 There shall be the capability to manipulate data cubes as general data structures, so that arithmetical and logical operations can be applied as object methods.

Levels: 2

Functionality: A2

Useability: A2

Documentation: A2

Comments: WB: This is what LEL is for.

6 Visualization
Crystal, Walter, Mark, John (all)

This is intended as the purely graphical part of data analysis. There is by necessity some overlap with the functionality discussed under Data Analysis, particularly that for image cube manipulation, and it would in fact be ideal if visualization and analysis were so closely integrated that there were no effective difference. The intention here is that the user is not only able to display pre-calculated images (processed using tools from the Data Analysis suite), but also has the capability of doing some processing and display on-the-fly as an integral part of the visualization.

6.1 General Visualization and Plotting Requirements

6.1-R1 Plotting and display capabilities shall be integrated into the GUI tools throughout the Package.

Levels: 1

Functionality: I1/high

Useability: E1

Documentation: U2

Comments: WB: This is a very general requirement that spans all of the package. In general, plotting capabilities exist in the places that the user would want plotting capability (e.g. plotting of calibration tables). The score of I1 is based on the fact that the plotting facilities generally exist, however there are limitations to their usefulness. For example, plotting a 3x3 array of VLA bandpasses results in text too small to read and some useful formats are omitted (e.g. plotting both phase and amplitude on the same plot).

Comments: JEH: I'll repeat a comment I made in Sec5: although any plot file generated in aips++ can be edited in any way desired via the pgplotter tool, that editing is too atomic - every individual element must be edited individually. It is desirable to be able to edit things in a global sense - eg, changing the text font for all labels in a 3x3 plot. Also, labels should be written using relative coordinates instead of absolute, so that if you zoom into the plot with the cursor, they don't fall off the page.

6.1-R2 Plotting and display capabilities shall be integrated into the CLI tools throughout the Package.

Levels: 1

Functionality: E1

Useability: A1

Documentation: A1

Comments: WB: The CLI tools suffer from the same deficiencies as the GUI tools (see above) and the cause of the deficiencies is the same, so the two requirements should be combined.

Comments: CLB: I think this requirement has to do with whether you can access the plotting cabibilities from the command line. As far as I can tell all the abilities of the GUI tools are available and hence this requirement is met. All of the problems mentioned throughout this section with the display tools still hold.

Comments: JEH: I would grade this as E on functionality. While you can generate a ps file directly from the CLI (e.g., cal.plotcal with psfile set), you have no control over how it is plotted. e.g., you can't plot bandpass solutions with a display range from 0.95-1.05, and I don't see how you could change this from the CLI using the pgplotter tool. You could read the desired table directly using table.getcol and make whatever plot you wanted with the pgplotter directly, but there should be more user control on producing plots with native plotting functions like plotcal. Maybe via a drop-down menu.

6.1-R3 Images in standard formats must be readable by the display tool.

Levels: 1

Functionality: I1/high

Useability: I1

Documentation: I1

Comments: CLB: The viewer can read in both AIPS++ format images as well as those written to FITS format by AIPS. However, if the image has been convolved in AIPS, the viewer no longer recognizes the beam size and hence the image statistics functions are useless.

6.1-R4 Standard type of plots must be supported, such as:

6.1-R4.1 X-Y plots
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.1-R4.2 histograms
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.1-R4.3 contour plots
Levels: 1
Functionality: A1
Useability: E1
Documentation: A1
Comments: WB: In the pgplotter, it is a bit obnoxious that the user must define the coordinate transform to plot a contour. Other than that, it is simple enough.

6.1-R4.4 vector plots
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.1-R4.5 2D images
Levels: 1
Functionality: E1
Useability: A1
Documentation: A1
Comments: WB: Greyscale images/arrays can be easily plotted. Color images can be plotted only with a color lookup table, (i.e. one cannot provide RGB at each pixel.)

6.1-R4.6 wireframe 3D surfaces
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: This functionality seems missing, but I don't see it being important.

Comments: WB: Since the pgplotter tool forms the foundation for plotting in aips++, my above evaluation was based on the functionality of pgplotter. It was not simple finding the complete documentation for pgplotter. Much of the useful documentation is not linked from the pgplotter tool page, but rather from the glish reference manual.

Comments: JEH: Grades hold for viewer too. Viewer documentation is pretty complete, however it is hard to find a given functionality if you don't already know how to get to it. e.g., if you want to make a plot of a spectra through a cube, its very hard to find out how to do this unless you know that you need to use the bullseye button. A useful table of comments or index to the viewer documentation would really help.

6.1-R5 "Blinking" between two different images must be possible.

Levels: 1

Functionality: N1/high

Useability: U1

Documentation: U1

Comments: WB: I assume that the viewer was the intended target for this requirement. Currently the viewer does not have blinking capability.

Comments: JEH: Why just 2 images? I think the user should be able to blink between a number of images (e.g., assign images to number buttons on the keyboard, and depressing the number select which image is displayed. This is possible in several other visualization tools.

6.1-R6 Stepping through a set of channels, at a user-selected rate and with adjustable transfer functions must be possible.

Levels: 1

Functionality: E1

Useability: E1

Documentation: I1/med

Comments: WB: The viewer can step through a third axis of an image manually or at a set rate, forward or backward. Transfer functions can be selected with the Adjust menu. Documentation specific to stepping though an image was not found, although it is quite straight forward.

Comments: CLB: You should also be able to select the channel by a draggable bar or clicking on the spectrum to update channel display. The current tape player design is overly complicated and cumbersome.

6.1-R7 Stepping through a set of ordered images (possibly stored in different cubes), at a user-selected rate and with adjustable transfer functions must be possible.

Levels: 1

Functionality: N1/high

Useability: U1

Documentation: U1

Comments: WB: I separated this and the previous requirements into two – the former for channels within one cube, and this for separate images.

Comments: JEH: I don't think this means viewing a cube along the 3rd axis, but something more like viewing a set of images in succession. eg., radio continuum, optical, Halpha, etc. Gipsy, DS9, saotng all let you do this in various ways. If it does mean viewing along a third axis, it should be written that way.

6.1-R8 Standard plotting formats shall be supported, both displayed (e.g. X window) and hardcopy:

6.1-R8.1 There must be at least one designated standard output format (e.g. postscript) that can be converted by the user to a variety of formats using easily obtainable tools.
Levels: 1
Functionality: E1

Useability: A1
Documentation: U1
Comments: WB: Both pgplotter and the viewer can save the plots as postscript. it is unfortunate that the postscript generated is not easily editted by hand.

6.1-R8.2 The Package shall also support the output of a variety of commonly used formats such as FITS, postscript, pdf, gif and/or jpeg.
Levels: 1
Functionality: I1/low
Useability: A1
Documentation: A1
Comments: WB: the image tool supports export to FITS (with `image.tofits()` and to ascii text (with `image.toascii()`), but not to any other format. FITS files can be conveniently converted to jpeg or other standard formats outside of aips++. Neither the viewer nor pgplotter have the capability of exporting to formats other than postscript.
Comments: CLB: The viewer can save images in XPM format which can then be converted to other formats without losses, but it would be better to have a more readily accessable format like gif or jpeg without the extra step of conversion outside of AIPS++.

6.1-R8.3 The Package shall also support the output of a movie or animation format (i.e. mpeg).
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1
Comments: WB: no functionality exists to meet this requirement.

6.1-R9 Identification of cursor position shall be available for interactive plots. Where appropriate, this information shall be recordable and exportable. If you "see-it" you should be able to figure out where it came from.

Levels: 1

Functionality: E1

Useability: A1

Documentation: U1

Comments: WB: The viewer displays the cursor position for each layer below the image. By default it is in world coordinates, however the Adjust menu allows this to be changed to pixels or other units. pgplotter displays the world coordinates of the cursor. I don't believe that the cursor position can be recorded/exported in either case.

Comments: JEH: Curson position is also reported beneath the image in the viewer, and is reported and can be copied to the clipboard using the viewer "Positions" option of the Imageanalysis tools'. I find it strange that you can not use X-windows to copy/paste the world coordinate position from either of these places to a separate xterm.

6.1-R10 An extra "axis" of information shall be encodable on the standard plot types using color and/or intensity.

Levels: 1

Functionality: E1

Useability: A1

Documentation: U1

Comments: WB: Many tools (msplot comes to mind) use color to represent amplitude. pgplotter allows color, size and shape choice for markers. I believe that many tools could take better advantage of this. Using intensity to encode amplitude and color to encode phase could allow for effective editing tools.

6.1-R11 The displays shall have similar look and feel to reduce the plotting learning curve.

Levels: 1

Functionality: A1

Useability: U1

Documentation: U1

Comments: WB: many of the core display elements are reused in many places.

6.2 Display Appearance and Interactivity

Comments: WB: My evaluation of this subsection regards the viewer exclusively.

6.2-R1 Plot selection parameters (axes, limits, colormap) shall be conveniently controllable.

Levels: 1

Functionality: I1/high

Useability: I1/high

Documentation: A1

Comments: WB: The Adjust menu allows control of axes and colormap. The limits are controlled interactively by zooming. The slow display of the adjust menu causes the low grade on usability. The limits of the animation axis cannot be adjusted, hence the low grade in functionality.

6.2-R2 User should have full control over the transfer function, including through a user defined color map or function.

Levels: 1

Functionality: I1/med

Useability: I1/med

Documentation: I1

Comments: WB: User-defined color maps are not supported. Moderate control over a reasonable set of pre-defined color maps is supported.

Comments: CLB: I give lower marks for useability due to the vague nature of the power scaling function and dragging the mouse to change the transfer function. It should be possible to draw or manipulate a line that describes the transfer function i.e. aipsview, XV.

6.2-R3 There shall be interactive display zooming and unzooming capability within plot windows.

Levels: 1

Functionality: E1

Useability: E1

Documentation: I1

Comments: WB: Interactive zooming is easy to use. It can be very slow on large images. Incremental un-zooming is not supported and is desirable. Documentation on zooming wasn't found by me.

6.2-R4 The plot update speed shall not be a bottleneck. Speed shall be benchmarked, and should be commensurate with comparable plotting packages.

Levels: 1,2

Functionality: E1,I2/med

Useability: U

Documentation: U

Comments: CLB: If the viewer supports FITS it shouldn't take a lot longer to modify or load these files than aips++ format images.

Comments: JEH: Viewer tool has made significant progress in this area, but can still be several times slower than comparable packages. Level1: movie rates and color map adjustments are now up to par to other systems. Live cursor plots (eg., plotting spectra as cursor moves around map) still lag behind other systems. Level2: The main remaining bottlenecks are in the speed of gui's to appear - initially for viewer to come up, and for adjust panel to appear. Takes forever for the viewer to load FITS images when used in stand-alone mode.

6.2-R5 Different line styles, sizes, thicknesses and colors must be available.

Levels: 1

Functionality: E1

Useability: A1

Documentation: A1

Comments: WB: This requirement is relevant to vector and contour maps. Different line thicknesses and colors are available, however the only option for a different line style is the option of encoding negative contours with dashed lines.

Comments: JEH: These are available in pgplotter for basic line plots

6.2-R6 The user shall be able to manipulate intensity and color scales, interactively for graphical displays. The setup shall be saveable and reloadable.

Levels: 1

Functionality: E1

Useability: A1

Documentation: I1/med

Comments: WB: Under the appropriate viewer image layer's Adjust menu, the current configuration can be saved. Documentation for these options was not found. The Help menu on the viewer has several documentation links, but some (e.g. Viewer Introduction) drive the web browser to documentation not relevant to the viewer.

6.2-R7 Basic axis transformations shall be built in to plotting, such as:

6.2-R7.1 Logarithmic amplitude and intensity scale.
Levels: 1
Functionality: A1
Useability: E1
Documentation: E1
Comments: WB: Logarithmic compression of the intensity scale is performed with the cryptically named "Power cycles" slidebar. Tool tips (the hovering help boxes) for this functionality are absent. The hovering help should indicate that 0.0 means linear scaling and explain the scale.

6.2-R7.2 Different time and coordinate units and formats (e.g. hours, hhmmss, radians, ddmmss.s).
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1
Comments: WB: It appears that axis coordinate format cannot be adjusted.

6.2-R8 The user shall be able to augment plots and produce overlays of different data sets of standard formats:

6.2-R8.1 Images with same axes, size and orientation shall be superposable directly, with basic control of colors and symbols.
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1
Comments: WB: The Viewer data manager allows this. Images do not need to have a common size and only the display axes need to be the same. I'm not sure what control over symbols implies.

6.2-R8.2 Overlay layer style shall be selectable, e.g. contours, greyscale, colormapped (RGB or HSV), or single color (i.e. one layer gets assigned intensity scales of red, another one of green, and one of blue).
Levels: 1
Functionality: I1/high
Useability: A1
Documentation: A1
Comments: WB: The viewer currently lacks the ability to allow HSV or RGB displays to be generated by 3 single-color input images. Contours, greyscale/colormap, and vector maps are currently allowed.

6.2-R8.3 Overlay of selectable coordinate grids (e.g. J2000, B1950, galactic, ecliptic, pixel number) shall be available. It must also be possible to overlay multiple grids.
Levels: 1,2
Functionality: E1,I2/low
Useability: A1,I2/low
Documentation: E1,E2
Comments: WB: All of these coordinate grids except "pixel" are supported. Full grids or ticks can selected and the display is quite configurable with sensible defaults. Documentation is sparse. Note that most of the options in the Adjust menu are documented as, *You might have noticed all the other options you have in the Adjust panel. Just play around and look what they do.* It is true that the best way to learn the viewer's functionality is by poking around, but a bit more documentation is waranted.
Comments: JEH: pixel grids are supported via the Axis Label Properties of the adjust gui. It does not appear to be possible to overlay more than one grid. I would consider that a level 2/low option.

6.2-R8.4 The user must be able to overlay functional fits (e.g. polynomials) or points read in from standard tabular files.
Levels: 1
Functionality: I1/high
Useability: I1/high
Documentation: I1/high
Comments: CLB: This does exist at some level through table overlays called a skycatalog. A tool exists inside of aips++ to help create such a table the documentation is quite poor. Also, only if an appropriate table file already exists in the directory that you are looking in does the "overlay table" option appear as a display data type. Therefore, if you don't already know about it, you would never find it.

6.2-R8.5 It shall be possible to place data sets in "layers" be which can be interactively colormapped, and switched on and off.
Levels: 1
Functionality: E1
Useability: E1
Documentation: E1
Comments: WB: This is functionally complete, I believe, but it would be more convenient if the

status bars (one displayed below the image per layer) would contain the register checkbox for that layer, and if it were easier to reorder layers.

6.2-R8.6 It shall be possible to display and overlay data with different coordinate systems, i.e., the coordinate system of the display can be chosen independent of the system the data were observed in and the data transformed appropriately with pre-computation.
Levels: 1
Functionality: A1
Useability: A1
Documentation: U1
Comments: WB: This happens behind the scenes. The concept of registering the image allows this to happen. However, if a J2000 image needs to be displayed on a B1950 pixel grid (this is different than simply drawing a B1950 coordinate grid) I believe the image needs to be resampled first.

6.2-R9 It should be possible to display and view multiple images simultaneously (i.e. not just in layers).

Levels: 1

Functionality: A1

Useability: A1

Documentation: U1

Comments: CLB: This is essential for comparing multiple images of the same region.

Comments: JEH: Doesn't this just mean that you can have several viewers up at once? This is possible.

Comments: WB: Grade set to A assuming JEH's comment is correct.

6.2-R10 Users shall be able to synchronize multiple display windows, for example such that zooming to a given pixel in one image window will select the equivalent pixel in the slaved windows.

Levels: 1

Functionality: N1/high

Useability: U1

Documentation: U1

Comments: WB: This essential feature is absent.

6.2-R11 Users shall be able to save all plots and images in a publication quality image. They should be able to add annotation, both interactively and through scripts, including text with various fonts (including Greek letters), symbols (e.g. all the symbols provided by the LaTeX package with AMSTeX extension), arrows, geometrical figures like boxes and circles, etc.

Levels: 1

Functionality: I1/high

Useability: I1/high

Documentation: A1

Comments: WB: The output is of publication quality, however, the annotation feature lacks printing ability if you save in XPM format.

Comments: CLB: The default location of the color wedge is extrememly unappealing with a large gap between the image and the wedge. The wedge is also quite narrow and sticks up above the top of the image. Time consuming fiddling with the canvas manager geometry tool (right margin space etc.) can reduce some of these problems, but should not be neccessary.

6.2-R12 Overlay files should have control parameters which allow the user to choose each item symbol, size, position, and text. The text should allow spaces and various fonts. The position format should be allowed in pixels, relative pixels, or world coordinates (hms/dms).

Levels: 1

Functionality: N1/med

Useability: U1

Documentation: U1

Comments: WB: I was unable to find any functionality that meets this requirement. The "Annotating" dialog has an option for saving and restoring – could this be related? Also the viewer Data Manager window has a Marker Map option, but no documentation was found for this.

Comments: CLB: I couldn't find any way to get the skycatalog tool to open. Search in the tool manager suggests that opening the viewer will give you this option but there is nothing there.

6.3 Image-cube Manipulation

6.3-R1 Histograms of pixel values must be easily produced for selected regions of the cube.

Levels: 1

Functionality: A1

Useability: A1

Documentation: U1

Comments: WB: The viewer's "Tools" menu has an image analysis option. When open, a region can be drawn on the image and a double click causes statistics to be generated for the region. A "Plot" button allows the histogram to be displayed.

6.3-R2 It shall be possible to view subsets or slices of data cubes that shall be interactively selected in GUI mode:

6.3-R2.1 for axes aligned with cube faces.
Levels: 1
Functionality: E1
Useability: A1
Documentation: U1
Comments: WB: Zooming allows interactive selection of primary display axes. Non-displayed axes are either set to one value, or set as the animation axis. The animation controls certainly need limit buttons.

6.3-R2.2 for arbitrarily aligned axes.
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: I'm not sure this is even useful. It is absent. Someone crazy enough to want this could define a mask to his/her taste.

6.3-R3 It must be possible to plot values of the same pixel in different cube layers, or in different images, against each other.

Levels: 1

Functionality: N1

Useability: U1

Documentation: U1

Comments: WB: Not sure what this means, however I don't think any such feature exists.

Comments: CLB: An example is a color-color diagram between two images of different frequency.

Comments: JEH: You could do this with regions and image.pixelvalue, then plotting these against each other. This is an analysis requirement. I can't think of a standalone visualization tool which does this in other packages.

6.3-R4 Data cubes must be viewable as movies with selectable frame rates and layer step directions.

Levels: 1,2

Functionality: E1

Useability: A1,N2/high

Documentation: A1,U2

Comments: WB: The "Adjust" menu allows the selection of display and animation axes. The tape deck control menu (triangle button below the displayed frame number) allows frame rate (up to 25 fps) and the number to step over.

Comments: JEH: I would add requirement to restrict pixel range of movie as a level2/high requirement, and grade functionality as E1 - we would like a slide bar for stepping through movie to replace the VCR controls. Clicking in the slidebar area could allow stepping through the cube.

6.3-R5 Interactive display of spectra corresponding to a pixel or region in a displayed image shall be supported.

Levels: 1

Functionality: E1

Useability: A1

Documentation: E1

Comments: WB: The `imageprofilefitter` tool does this. It would be nice to have this better integrated with the viewer for quick usage.

Comments: CLB: The designation of the Image Analysis roll down of "Positions" does not intuitively lead you to believe this will display spectra.

Comments: JEH: Faster updating is needed. Documentation is E1 - this functionality is hard to find if you don't know where it is.

6.3-R6 Interactive display of a 1D slice taken from a 2D image, such as dragging the line on the map to bring up a position-velocity diagram, shall be available. The resulting PV diagram should be able to be saved in a publication quality image with annotations.

Levels: 1,2

Functionality: N1/med,N2/med

Useability: U1,U2

Documentation: N1/high,N2/med

Comments: WB: I believe no functionality exists.

Comments: CLB: I found a listing in the getting results chapter that suggests that there is such a thing as a "Viewer Slice Display Panel" but I could never locate these functions.

Comments: JEH: You can get a 3-D slice in the viewer by going under "File=>New Slice display", but this does not allow random positions to be viewed. I would add as a level 2 requirement the ability to do slices along line-segments, as in Gipsy.

6.3-R7 Plotting of spectra on a pseudo-grid corresponding to position on a raster (e.g. a "stamp map" or "profile map", basically thumbnail spectra in panels corresponding to position) shall be possible.

Levels: 1

Functionality: N1/med

Useability: U1

Documentation: U1

Comments: WB: I believe no functionality exists.

6.3-R8 Plotting of multiple channel maps in a thumbnail format with publication quality (e.g. axes only along outer most channel maps) shall be possible.

Levels: 1

Functionality: I1/high

Useability: I1/high

Documentation: I1/high

Comments: CLB: It is possible through the canvas manager to indicate how many channels you want to display in x and y in the viewer. However, in practice, the result is very unapealling with large gaps between the images and one giant color wedge extending the full size of the canvas. It needs to be possible (the default) to have the image axes only displayed for the outermost planes.

6.4 Plotting Visibility Data

6.4-R1 Plotting of commonly used basic data and calibration quantities must be straightforward and easily accessible from all relevant tools. Any quantity should be displayable versus any other (or any two plus another encoded as a z-axis or intensity) as long as these quantities are defined for the same visibility or calibration solution interval. These displayable quantities include:

6.4-R1.1 real and/or imaginary
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.2 amplitude and/or phase
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.3 time
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.4 hour angle
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1

6.4-R1.5 parallactic angle
Levels: 1
Functionality: N1/med

Useability: U1
Documentation: U1

6.4-R1.6 uv distance
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.7 u and/or v
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.8 baseline
Levels: 1
Functionality: I1/med
Useability: U1
Documentation: I1/low

6.4-R1.9 channel or frequency or velocity
Levels: 1
Functionality: A1
Useability: A1
Documentation: A1

6.4-R1.10 azimuth and/or elevation
Levels: 1
Functionality: N1/med
Useability: U1
Documentation: U1

Comments: WB: All of the above were graded on the msplot tool. Rate and Delay were removed from the requirements. The missing functionality above (hour angle, parallactic angle, az., el.) are all not strictly defined for an array of non-zero size, however, should be implemented for the VLA array center. Baseline is currently replaced with the probably less useful Ant1 and Anr2 parameters.

Comments: JEH: Maybe this needs to be split into two requirements, with the 3rd axis encoding separate. I do not believe it is possible to plot two of these quantities and encode another as intensity. Of particular interest would be Amp(u,v), Phase(u,v), or even the ability to encode two quantities (phase, amplitude) on plots of two other - eg., intensity-hue plots of amplitude base for u-v plots. level2/high.

6.4-R2 Data selection parameters shall be understandable (e.g. by antenna name or number instead of antenna table entry number, polarization name RR or I) and straightforward, using graphical browsers (in GUI mode) and/or standard selection language (e.g. SQL queries) in script mode.

Levels: 1

Functionality: A1

Useability: A1

Documentation: A1

Comments: WB: No exceptions were found.

6.4-R3 It shall be possible to interpolate or extrapolate any tabulated quantity onto a visibility or calibration solution point, and then manipulate these like extra visibility information.

Levels:

Functionality:

Useability:

Documentation:

Comments: WB: What?

Comments: CLB: I agree, what?

6.5 Plotting Ancillary VLA Data

Although the Package will not likely be the primary vehicle for the NRAO staff to assess the state of the array, it is intended that users (as well as staff) have the full capability of using ancillary data provided by the VLA to aid in the processing and understanding of their data. Therefore, the Package should be able to deal with this data in as user-friendly a manner as possible.

6.5-R1 The Package shall be able to plot standard VLA-format ancillary data, including:

6.5-R1.1 amplitude or single-dish power versus AZ and EL.
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: This is again penalized because of Az., El. not being supported.

6.5-R1.2 WVR output data (when available).
Levels: 2
Functionality: N2/low
Useability: U2
Documentation: U2
Comments: WB: Later...

6.5-R1.3 holography and beam map data.
Levels: 2
Functionality:
Useability:
Documentation:
Comments: WB: Not sure about the requirements here.

6.5-R1.4 monitor point values (e.g. temperatures).
Levels: 2
Functionality:
Useability:
Documentation:
Comments: WB: I don't believe this data is filled with VLA data – first a filler is needed.

6.5-R1.5 TIP data and solutions (if available)
Levels: 2
Functionality:
Useability:
Documentation:
Comments: WB: Not sure.

6.5-R1.6 Reference pointing solutions (if available)
Levels: 2
Functionality:
Useability:
Documentation:
Comments: WB: Not sure.

7 Special Features
Mark (simulations), Bryan (solar system), Walter (pulsar and astrometry), Steve (pipeline), Tim Bastian/Debra (solar observing)

7.1 Simulations

Note: Inclusion of at least moderate simulation capability in the Package will benefit the user. The main goals are to allow the user to simulate basic VLA modes based on input models or images and to replace existing data in VLA format with simulated data.

7.1-R1 The Package must be able to compose models or images using any combination of 3D Gaussians, point-sources, or other geometrical structures with a variety of intensity distributions.

Levels:

Functionality:

Useability:

Documentation:

7.1-R2 For specified source models, the Package should be able to simulate uv data or images with the desired array configurations.

Levels:

Functionality:

Useability:

Documentation:

7.2 Solar System Object Observing

The Sun and planets are important and interesting targets for VLA observing. The requirements are likely to be strongest on the actual hardware (e.g. high frequency and time resolution) but software compatibility must also be considered. In particular, solar and planetary observations require a special effort in tracking and handling of ephemerides.

7.2-R1 There shall be the provision for the near-field imaging of solar-system objects. This can be done through the introduction of phase corrections based upon the sphericity of the incoming wave front.

Levels:

Functionality:

Useability:

Documentation:

7.2-R2 The Package must be able to calculate and compensate for the position of moving objects in the solar system:

7.2-R2.1 Body-centered ephemerides must be provided for major solar system objects, including:

7.2-R2.1.1 Sun

Levels:
Functionality:
Useability:
Documentation:

7.2-R2.1.2 Moon
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.1.3 planets
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.1.4 major satellites (all with diameter > 50 km)
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.1.5 major asteroids (all known with diameter > 50 km)
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.1.6 known short period comets
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.2 Ephemerides must be calculated in all available reference frames.
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.3 The Package shall be able to import a user-supplied ephemeris in tabular form.
Levels:
Functionality:
Useability:
Documentation:

7.2-R2.4 The Package shall calculate positions from user-provided orbital elements.
Levels:
Functionality:
Useability:
Documentation:

7.2-R3 The Package must carry out astrometry for moving sources.

Levels:

Functionality:

Useability:

Documentation:

7.2-R4 It should be possible to compute and apply the visibility corrections for variable distance to the object (both amplitude and u, v).

Levels:

Functionality:

Useability:

Documentation:

7.2-R5 The imaging corrections for the near-field shall be included.

Levels:

Functionality:

Useability:

Documentation:

7.2-R6 It should be possible to take the observed positions as a function of time, and the known (or input) positions, and make a correction to the visibility phase to compensate for the difference.

Levels:

Functionality:

Useability:

Documentation:

7.2-R7 It should be possible to take out the body tracking in the visibility file, making background sources appear fixed, and then put the tracking back in.

Levels:

Functionality:

Useability:

Documentation:

7.2-R8 Any time that a different position is specified and visibilities modified (see the above 2 reqs), the new tracked position (which might be fixed) must be recorded with the data as well as the original observed position.

Levels:

Functionality:

Useability:

Documentation:

7.2-R9 During deconvolution, it should be possible to specify an initial model (limb-darkened elliptical disk) which is to be subtracted from the u-v data as the first step in the deconvolution.'

Levels:

Functionality:

Useability:

Documentation:

7.2-R10 It should be possible to plot on top of an image, a labelled grid of cartographic coordinates for the body:

7.2-R10.1 with viewing geometry (distance to body, subearth latitude, longitude, and PA of north pole) and physical body parameters (polar and equatorial radii) specified by the user
Levels:
Functionality:
Useability:
Documentation:

7.2-R10.2 with the above parameters calculated (or tabulated) internally
Levels:
Functionality:
Useability:
Documentation:

7.3 Solar Observing

To observe the Sun one must point at the Sun. This is done the same way that planetary observations are made. Tracking requirements for Solar System objects are covered in the previous section.

The Sun differs from the planets and comets in several important respects. First, it is large - larger than the primary beam in most bands - and it rotates. Hence, an observer might want to correct for both the Sun's apparent motion against the background sky *and* its rotation. Second, the Sun is bright, so bright that the signal must be attenuated. Finally, the Sun can be highly variable: the total flux can change by orders of magnitude. Specific requirements to reduce solar observations to within currently possible errors are:

7.3-R1 When filling VLA data offline, one must take care *not* to compute weights on the basis of the nominal sensitivity since they're often meaningless for many antennas. When solar data are filled, the Tsys correction applied by the VLA online system must be undone. Values of Tsys and the nominal sensitivity (real or dummy) must be made available to the observer to determine which antennas have solar cals and if any antennas or IFs are bad.

Levels: 2

Functionality: N2

Useability: U

Documentation: U

Comments: TB/DSS: Not available at this time.

7.3-R2 Once the "good" solar cal antennas have been identified, the a task analogous to the AIPS task SOLCL must be available to average the nominal sensitivities of the antennas (for each IF) for each integration time. The dummy nominal sensitivity of any antenna that had no solar cal must be replaced with this value. After this step, calibration proceeds just as it does for any other source.

Levels: 2

Functionality: N2

Useability: U

Documentation: U

Comments: TB/DSS: Not available at this time.

7.4 Pulsar Observing

There is no need for pulsar specific data reduction capability for the VLA. The time bins available to EVLA will change this.

7.5 Astrometry Considerations

7.5-R1 The facility to correct the baseline vectors (u, v, w) produced by the VLA correlator shall be present, either incorporated into the filler, or as a separate tool:

7.5-R1.1 Timestamps should be corrected to reflect the midpoint of the sample integration.
Levels: 2
Functionality: A2
Useability: A2
Documentation: U2
Comments: WB: This is done at the time of data filling. No user intervention is needed. Documentation should not be required.

7.5-R1.2 Special relativistic diurnal and annual aberration correction should be applied.
Levels: 2
Functionality: I2/low
Useability: U2
Documentation: U2
Comments: WB: There is no method to correct a ms for this effect. There is, however, functionality in the dm.uvw function that might allow for easy implementation.

7.5-R2 Image plane Gaussian fitting shall exist:

7.5-R2.1 Accurate position and amplitude should be computed for sources detected anywhere in a synthesized image.
Levels: 1
Functionality: A1
Useability: A1
Documentation: E1/low
Comments: WB: The imagefitter tool satisfies this requirement. Documentation is available on-line : `http://aips2.nrao.edu/released/docs/user/General/node205.html` . Measured amplitudes and positions agree with AIPS task JMFIT in both the low and high signal-to-noise cases. The advanced fitting options are a little complicated to use and the documentation is terse, although complete enough to figure it out.

7.5-R2.2 Accurate error bars shall be reported for each fit parameter.
Levels: 1
Functionality: E1
Useability: A1
Comments: WB: These error bars come for free with the above fitting. Error bars are consistent with those from JMFIT *except* for postion angle of the elliptic Gaussian, in which case imagefitter claims about 90 times the precision (could this be a radians-to-degrees conversion bug? Submitted as defect AOCso04226.

7.5-R2.3 Covariances between each pair of fit parameters shall be reported.
Levels: 2
Functionality: N2/med
Useability: U2
Comments: WB: This isn't available. At the time of defect AOCso02866 submission, internal structures did not allow off-diagonal covariances to reach the user.

7.5-R2.4 Results should be reported to sufficient precision.
Levels: 1
Functionality: E1
Useability: A1
Comments: WB: RA and Dec results are reported to 1 mas. High precision 43 GHz relative astrometry could require more precision than this. For VLA I'd like to see 1 more digit. VLBI will require even more.

7.6 Pipeline Considerations

AIPS++ should be able to flag, calibrate, and image current VLA data in an off-line pipeline. All components necessary to support off-line processing with a pipeline script should be available.

The AIPS++ package should meet all requirements outlined in section 1.5 to ensure the command line interface (CLI) is adequate to handle off-line processing. In addition, the following requirements should also be met:

7.6-R1 Automatic flagging capabilities should be available to flag bad data. The heuristics for identifying

and flagging bad data are to be developed for the Expanded VLA. If they are not ready for the current VLA, then the pipeline script should allow an individual to manually edit the data at specific points in the process, e.g. at the beginning of the pipeline, after calibration solutions have been derived, and after calibration has been applied.

Levels:

Functionality:

Useability:

Documentation:

7.6-R2 Automatic calibration should be available for:

7.6-R2.1 the flux density scale
Levels:
Functionality:
Useability:
Documentation:

7.6-R2.2 polarization calibration
Levels:
Functionality:
Useability:
Documentation:

7.6-R2.3 bandpass calibration
Levels:
Functionality:
Useability:
Documentation:

7.6-R2.4 phase and amplitude calibration
Levels:
Functionality:
Useability:
Documentation:

7.6-R3 Automatic imaging should be available for all sources in the project. There should be the capability to allow an individual to break into the process to create a mask interactively.

Levels:

Functionality:

Useability:

Documentation:

7.6-R4 The end products of the pipeline should include:

7.6-R4.1 the script used to flag, calibrate, and image the dataset
Levels:
Functionality:
Useability:
Documentation:

7.6-R4.2 plots of the calibration solutions
Levels:
Functionality:
Useability:
Documentation:

7.6-R4.3 visibility plots for each source
Levels:
Functionality:
Useability:
Documentation:

7.6-R4.4 deconvolved, restored images for each source
Levels:
Functionality:
Useability:
Documentation:

8 Help Facility and Documentation
Debra, Mark (all), John (VLA cookbook), Walter (review grades/comments)

Applicable tools for the help facility include the `bug` tool (e.g. reports and status via e-mail and web tracking). Applicable VLA documentation should include the VLA cookbook as well as Getting Started, Getting Results and the layout/organization of these documents on the web. Adequate information should be available on the main web page about who to contact for questions.

8-R1 There shall be a variety of help levels and documentation formats accessible from the UI and over the Internet, applicable to novices, experts, and technical users. These shall include:

8-R1.1 User cookbooks with extensive examples and detailed discussion about how to reduce interferometer data (e.g. AIPS cookbook level)
Levels: 1
Documentation: I/high
Comments: DSS: There is a VLA Cookbook being written that goes through 2 basic examples - it is not complete (awaiting continuum subtraction code changes) and latest version is not available on the web. Once all GUI instruction is complete, the next version should include data reduction wisdom like what is in the AIPS cookbook.

8-R1.2 Script library with adequate documentation and comments in the scripts for the intermediate-level user.
Levels: 1
Documentation: I/high
Comments: DSS: There are not near enough scripts/recipes to be useful: There are 7 general glish-use scripts in the recipe repository and only one example of VLA continuum data reduction exists for end-to-end synthesis scripts. There are also synthesis examples of msclean, and mstofits tool use. The VLA cookbook scripts are not in the recipes repository index - they should be. Instead they are in aips2.nrao.edu/stable/code/nrao/scripts/ - I can only find a link to these in the online VLA cookbook.

8-R1.3 An up-to-date dictionary mapping AIPS task functionality to AIPS++ tools and functionality.
Levels: 1
Documentation: E
Comments: DSS: the dictionary exists but it is not up-to-date and no date exists on the dictionary to indicate when this list was generated. This list (and the MIRIAD-AIPS++ cross listing) should be updated each release cycle.

8-R1.4 Application descriptions
Levels: 1
Documentation: E
Comments: DSS: CBvol2 has good, detailed descriptions of synthesis calibration & self-cal, imaging (including wide-field, and mosaicing), plotting, data display, and image analysis. An excellent start. Similar detailed documentation for the remainder of the end-to-end VLA data path should be provided: 1) data import/export and 2) interactive and non-interactive flagging.

8-R1.5 Reference manual (with all inputs to functions and tools)
Levels: 1
Documentation: I/high
Comments: DSS: The reference manual exists but many segments are written for the programmer, not the end users. Yet the Help buttons on the tools all drive the web browsers to the Ref manual. Thus, the Ref manual should describe all inputs in a language that is understandable for the beginning user. Examples of the Reference manual being non-outside user friendly: 1) Ref man

flagger.setpol says you can choose polarization indices (not names) but doesn't tell the user how to map indices to names (1=RR, 2=LL, 3=RL?, 4=LR?); 2) cal.setsolve says component type allowed = G,T,B, or D with default = ". At this level, there is no description of what G, T, B, D are (you have to go elsewhere to find this); 3) imager.setimage says distance = distance to object: usually ignore this! (so when should the user not ignore this?); 4) imagepol.linpolint: "return" parameter has no help documentation at all; 5) imager constructor compress parameter help says: Compress calibration columns? Allowed=Bool, Default=F (no indication how this will affect data or calibration). There are many examples like this peppered through the Ref manual. It should be reviewed thoroughly and descriptive statements for EACH input should be added.

8-R1.6 On-line help, FAQ, email contacts. These shall be updated on a timely basis (e.g. with each public release).

Levels: 1

Documentation: I/low

Comments: DSS: FAQs seem to be very out-of-date (e.g. FAQ: AIPS++ runs too slow. Answer: get a bigger machine; FAQ: AIPS++ just hangs, now what? Answer: check that lockd is running on your NSF server (so what do I do if it is?). These answers are not particularly helpful for users. FAQs should be reviewed and updated on a regular basis. I cannot find a web link to On-line help where a user can e-mail a question in and get an answer back (I've looked in Contact Us, Getting started, Learn More, & Documentation sites.) The facility does exist from the `Tool manager` **help** button, **ask a Question** option. I knew this facility was available but it took me 2 hours of playing around in help options to find it - way too obscure to be useful. Now, if you do know someone in the project, then the 'Contact us' web site is up front and has adequate information to call or send e-mail to a specific person.

8-R1.7 Bug reports and tracking

Levels: 1

Functionality: A1

Useability: A

Documentation: E

Comments: DSS: Bug reporting is easy enough from the GUIs and Glish command line. It is mentioned up front in the VLA cookbook. How to submit a bug report is not mentioned in "getting started" documentation: it should be. Bug reports and submittal forms are buried in the "Contact Us" link, it should be its own high-level link. The list of known bugs (http://aips2.nrao.edu/docs/contactus/knownbugs.html) is woefully out of date - only 53 bugs are listed as of May 2001, the latest one is AOCso02544 - but we are up to AOCso04210 now, almost 2000 bugs have been submitted since this page was updated. The same condition exists for requested enhancements.

8-R1.8 Release history, patch descriptions

Levels: 1

Documentation: N/med

Comments: DSS/WB: I can find no documentation on the web that describes the Release history of AIPS++ or information about patch descriptions. The web states that we are in the 6th public release, and gives Version & Build numbers but nothing more that I can find. It would be particularly useful to have BRIEF patch descriptions added to the code startup message and a high-level web page so users can quickly identify where there are changes/improvements.

8-R1.9 Programmer references and guides

Levels: 1

Documentation: I/med

Comments: DSS/WB: The Reference manual is probably adequate for the programmer to see input structure. Programming manuals and help are also available on the web in an obvious location. However, it doesn't mention how outside users can set up their own development environment. If a user wants to do this now, they have to work with programmers closely to figure it out.

8-R1.10 Data format descriptions

Levels: 1

Documentation: E

Comments: DSS: The measurement set description is adequate but it is buried in AIPS++ Note 191 (referenced in the calibration section of http://aips2.nrao.edu/docs/gettingresults/grvol2/node2.html). This is not in an obvious location for users. It should be in a more obvious location. There is however, a summary and reference link in the VLA cookbook that gives users adequate information if they don't want to know too many details.

8-R1.11 Algorithm descriptions

Levels: 1

Documentation: I/high

Comments: DSS: None available on the web that I can find. Algorithm descriptions would be very useful to users who are trying to figure out exactly what is going on inside of a task.

8-R1.12 Newsletters, email exploders, notes series

Levels: 1

Documentation: A

8-R2 Documentation must be kept up-to-date and complete for all parts of the Package.

Levels: 1

Documentation: I/high

Comments: DSS: Some documentation is not kept up-to-date. Specifically: AIPS-AIPS++ dictionary, FAQs, lists of known bugs and enhancement requests, & release history/patch descriptions. Thus, it appears that a formal documentation review mechanism is not comprehensive enough or not done often enough.

8-R3 Help materials shall be available in printable formats, including:

8-R3.1 standard document formats (pdf, postscript)

Levels: 1

Documentation: E

Comments: DSS: postscript files are available but pdf is not provided.

8-R3.2 printer-friendly versions of HTML pages

Levels: 1

Documentation: A

8-R4 Help shall be context-sensitive where relevant. In GUI mode, fly-over banners should indicate use of buttons and fields, and clickable help buttons should be available on all GUI windows.

Levels: 1

Documentation: E

Comments: DSS: Flyover banners exist for most GUI inputs but there are some that have no popup help (e.g. calibrater.calave(mode), or that have the message: 'No help available' (e.g. viewer.datamanager & displaypanel inputs). Click-able help buttons are available on most GUIs that drive the web browser to User Reference Mannual chapters; the only exception I've found is the viewer Annotation GUI. Popup help should be reviewed for *ALL* inputs and added where necessary. Annotation GUI should have help documentation added.

8-R5 In GUI mode, help functions should direct a browser to a Web page.

Levels: 1

Functionality: A1

Useability: A

Documentation: U

Comments: DSS: The basic functionality exists and is easy to use. **Documentation:** The documentation level that the web takes you to is not appropriate for beginning or intermediate-level users. However, this is assessed in the Ref Manual requirement so it is not graded here. IF it is decided that GUI Help buttons will no longer be driven to the Ref manual, then this requirement will have to be graded again.

8-R6 In CLI mode, the Package must support in-line text based help.

Levels: 1

Functionality: I1/med

Useability: I/med

Documentation: N/med

Comments: DSS: **Level 1 functionality & Use:** typing 'help' at the CLI gives glish code and errors, typing help(); doesn't work either, it requires an argument of 'general', 'Refman:' or 'Glish:' **Documentation:** The 'general' argument provides very little information. Refman and Glish, drive your browser, they do not provide text-based help. If I know exactly the module.tool path, I can type, e.g., help('synthesis.calibrater') and I still get no useful information until I type web(); again not text-based. Essentially, CLI-based help is not useful except as a conduit to the web help. I say this is only of Medium severity because there is Web-based documentation so the absence of Glish-access text-based documentation seems to be a lower priority.

8-R7 Both the GUI and CLI-based help levels should be appropriate for the beginning user and should be consistent (e.g. GUI and CLI help should provide the same documentation links or information).

Levels: 1

Functionality: I1/med

Useability: I/med

Documentation: I/med

Comments: DSS: **Level 1:** GUI and CLI-based help are very different. The GUI-based help is closer to being what the user needs.

8-R8 Full search capability must be built into the documentation library.

Levels: 1

Functionality: A

Useability: A

Documentation: E

Comments: DSS: There is no documentation when you click on Search to tell you the simple steps of how to run the search engine or find a specific function. Its really simple so even a few sentences added at the site http://aips2.nrao.edu/stable/docs/search/search.html should be adequate.

8-R9 The top-level organization of the help documentation on the web should be easy to navigate for the new user.

Levels: 1

Documentation: I/high

Comments: DSS: http://aips2.nrao.edu/docs/aips++.html (main NRAO web page) is a cute (but not very useful) page that gives the mission statement, and a general presentation on AIPS++. But, the practical links users will need are in a little bar on the top that is not that obvious. I suggest that the top-level web page should have clear, bulleted links to the things most important to users: Getting started; FAQs; Cookbooks & Documentation; Bug Reports; Release Notes & Patches; Newsletters, Memos, & Notes; Contact Us; Search. Then more detailed links at the bottom: E-mail exploders; Learn More about the AIPS++ Project; Glish; Programming. All of the above links should then be in the top bar(s) on all subsequent pages.

8-R10 Secondary-links or clear paths should be available for:

8-R10.1 VLA Cookbook
Levels: 1
Documentation: I/high
Comments: DSS: It is great that the primary help documentation is now called a Cookbook. However, a cookbook is supposed to be for beginning users. Having a 3 volumn cookbook with a total of 17 chapters (only one of which is applicable to VLA data reduction) is way too complicated a path for beginning users. Cookbooks should be listed separately, at a high-level and be a stand-alone document.

8-R10.2 Bug reporting
Levels: 1
Documentation: I/low
Comments: DSS: Bug reports and submittal forms are buried in the "Contact Us" link, it should be its own high-level link.

8-R10.3 Reference Manual
Levels: 1
Documentation: A

8-R10.4 Release history
Levels: 1
Documentation: N
Comments: DSS: Link is not available that I can find.

8-R10.5 On-line help, FAQs, e-mail contacts
Levels: 1
Documentation: I/med
Comments: DSS: On-line help link on web not available. FAQs and e-mail contact links are obvious.

8-R10.6 Newsletters, email exploders, notes series
Levels: 1
Documentation: E
Comments: DSS: Buried in the "Learn More" link, it should be more obvious such as: Newsletters, Memos, & Notes.

A VLA Observing Modes

We attempt to define the set of VLA observing modes that cover the range of observations that will be carried out with the current array (before EVLA). A set of **keywords** are defined which describe the modes.

The keywords which define the VLA Standard Observing Modes are:

SCAN_MODE How are the pointing centers arranged?

single_field — a single pointing;
mosaic — multiple linked pointing centers (e.g. a raster);

FIELD_MODE Are wide-field corrections necessary?

narrow — no correction necessary;
wide — sky curvature and/or w-term correction necessary;

POLARIZATION How many parallel-hand and cross-hand products are there?

single_pol — one parallel hand (e.g. RR or LL) available, which will proxy for Stokes I;
dual_pol — both parallel hands present (e.g. RR and LL), allowing Stokes I and one other (e.g. V);
full_pol — both parallel hands and both cross hands present (e.g. RR, LL, RL, LR), allowing all Stokes parameters (IQUV);

SPECTRAL_MODE How many spectral channels are to be analyzed together?

continuum — independent (usually relatively wide) bands (e.g. IF sidebands);
spectroscopic — more than one spectral channel which must be analyzed together (possibly to be combined post-calibration into a pseudo-continuum channel, or kept separate);

Observation definitions are constructed using these keywords. For example the most common VLA continuum mode is given by: **single_field, narrow, dual_pol, continuum**. There are 24 VLA standard modes built from these keys.

There are also three VLA Special Observing Modes:

SPECIAL_MODE Are there special hardware or software modes used in the observation?

solar — for bright solar signals;
planetary — for moving sources with ephemerides;
pulsar — using pulsar gating backends etc.;

The default is for no special mode (if unspecified).

B Wide-Field Considerations (Frazer Owen)

This is an attempt to describe the steps necessary to make a wide-field, high resolution (A or B array), deep VLA survey at 20cm. The motivation is to encourage making this process possible in AIPS++. The process is complex and also requires steps outside of what one would think of as radio synthesis imaging. However, to arrive a a useful scientific result one needs to consider the entire process.

B.1 Filling the data: database efficiency

One major problem what making a deep survey is that it requires a lot of data. Presently, the AIPS++ database is very nice for small problems, carrying along all the information needed in a nice accessible way. However, the size of the databases for deep surveys (20–100 hours, 5s integrations (probably should be shorter), 351 baselines, 2pol x 2IFs x 7channels) makes a database too large practically to be swallowed by the AIPS++. Also while disk sizes will eventually absorb this database size, the new correlator will increase these numbers by about 100x, so AIPS++ needs a mode where it deals with the databases in as efficient a way as possible. It seems to me this means a 16bit compressed format and possibly fewer bits of information being carried along. AIPS++ at least needs to match AIPS efficiency here.

B.2 Flexible uv Editing

Throughout the process, starting just after filling at any time one needs to be able to investigate the uv data and apply one of many flagging processes. In AIPS I use TVFLG, UVFLG, CLIP, and FLGIT at various stages depending on what I encounter. Flexible flagging and display is a must. One must also have some way to reverse the process, either by keeping multiple copies of the database or by easy reversibility of the flagging. The latter is harder than the former but is more space efficient.

B.3 External Calibration

1. Bandpass Calibration:
    1. phase self-calibrate bandpass calibrator before any calibration is applied;
    2. calculate and display (check) bandpass corrections;
    3. apply corrections to original database, including unselfcalibrated bandpass calibrator since it may be used to external phase calibration.
2. Bootstrap amplitude calibration from flux calibrator to phase calibrator
3. Calculate and apply amplitude and phase calibration
4. Calibrate weights using external system temperatures. Currently I fit a polynomial to the variation of Tsys with elevation and apply a global Tsys vs elevation correction to all antennas. This is because I don't believe the noise tube calibrations. There are several other ways this might be done. The relative weights are often in error by factors of 5 or more and thus this is important and should be viewed as part of the necessary calibration.

B.4 Setting Up the Grid of Imaging Positions

For wide-field, 3D imaging, one needs an inner dense grid of of facets which cover the primary beam out to almost the first null calculated so that 3D effects are small enough to be ignored. In addition one needs a number of outlying fields to cover the bright confusing sources which lie in the first sidelobe or beyond. The first part of the problem is straightforward and I believe AIPS++ can do that. The outlying sources are another problem. One can find them by making a tapered image of a very large field and/or using external catalogs like NVSS. To do the latter one would need a program which has fairly detailed model of the primary beam beyond the first null. The first way is easier and that is what I do most of the time. AIPS does have a version of the second method which some people use.

Once one has an image one needs an easy way to pick out to pick out and record the confusing sources. I usually display a box on the TV which covers the area sampled by the dense inner grid, so I can see what parts of the image to ignore. Then I use TVMAX in AIPS to point at the outer confusing sources and measure their positions. AIPS uses a simple ASCII file as the input source for IMAGR to tell it where

to put the outlying fields as well as the dense grid. Thus one can simply use the mouse and an editor to move the TVMAX fitted positions into the list of coordinates for AIPS.

B.5 Picking the Weighting for IMAGR

Besides the external weights one also needs to pick the weighting schemes to use on the gridded uv data to get the best compromise between beam size and noise characteristics. This requires trying several different sets of parameters before one starts. The best solution depends on the detailed uv sampling, so it is not possible in general to decide this in advance.

Right now for me this is usually a combination of ROBUST, SUPERUNIFORM and a TAPER. There could be a bigger parameter space than this. One needs to do this to be sure one is getting the best combination before going through the very long reduction process to come.

To do this in AIPS++ right now as I understand it one must undo the imaging weights in the database each time and go re-apply all the different forms of weighting one wants each time. One must also remember what one has done. AIPS, on the other always recalculates all the imaging weights, so one must just change a single parameter in IMAGR and rerun the program to try another combination. This is computationally inefficient but that is a rather minor matter in total processing time for a deep survey.

There needs to be an user efficient way to do this, probably with a GLISH script and perhaps with a grid of trial beams and statistics produced. I also would prefer always to apply the weights on the fly to minimize bookkeeping but that is a close call.

B.6 Optimal Averaging of the Input uv Dataset

Before imaging to minimize imaging time, one should average the data in time and frequency to produce the smallest input dataset. Perhaps one should do this before step IV. In AIPS one can do this with the program UBAVG which runs through the data on each baseline and averages versus time in such a way as to produce less than an N percent error inside some radius from the phase/pointing center. This can make a significant difference in imaging time since the uv-based clean process is dominated by the number of uv points it must subtract a model from.

In principle there should be a step in the imaging process which does this optimization of the uv dataset for both time and frequency. With the current correlator at 20cm frequency sampling is not usually adequate, so only time averaging is important but at longer wavelengths now and in the future at 20cm and shorter wavelengths frequency averaging will be a issue.

However, one does need to be able to go back to the full time resolution in the selfcal step. Also when one is combining uv data from multiple days one needs the full time resolution.

B.7 Initial Imaging

Once one has the full grid set up, one can make the initial set of images. One needs to clean this image to a moderate level so one can see all the sources. AIPS++ should be able to do this now. For deep imaging one normally will have multiple days with the same or similar uv coverage. For this initial image, one uses just one day's data to minimize the processing time.

B.8 Initial Boxing

Without BOXES to limit the areas CLEAN can consider, CLEAN will scatter power outside the regions with real sources to optimize its fit to the data. This will bias the noise low, bias the source flux densities low, and take longer to run than if one restricts the algorithm. It will also produce a non-optimum model for the selfcal process. There are other ways to solve this problem besides BOXES such as rejecting isolated clean components by some method before restoring the image but BOXES seems to me the most straight-forward, even if it is expensive in real time for the user.

In theory, one could write program to find the boxes. However, in practice such a task has trouble distinguishing between residual sidelobes from strong sources and real sources which the human can do fairly easily. For big surveys, some such automatic program is needed but I suspect we may always need a user check and adjustment of the program's solution. Thus some sort of interactive environment is needed.

In AIPS right now the process is entirely interactive. One displays the image, or part of it, on the TV and uses the graphics overlays to mark where the BOXES should go. The interactive program, FILEBOX, writes the marked positions in the same ASCII file as it stores the field centers and sizes. This makes editing this file very flexible. This process is very time consuming, however.

In the ideal world there would be a program which sets up the first guess at the BOXES (or more general REGIONS). Kumar already has produced a first attempt at this. Then the user would display and correct interactively the automatically created REGIONS.

The most obvious thing missing from AIPS++ is the graphical overlay for the TV. This will come up again and again in the image analysis steps.

B.9 Second Imaging

After the BOXES, one reruns IMAGR, staring from scratch with the clean components and clean down to the current noise level.

B.10 Selfcal loop

One then selfcals the first days uv data using all the clean components and re-images the field. Usually, one does this the first time with a phase-only selfcal, re-images, and then does it again with a phase and amplitude selfcal and re-images again. After each re-imaging one checks the clean boxes, usually adding quite a few after the first selfcal. The amplitude and phase selfcal is usually done with a longer solution interval to increase the S/N since one is solving for more variables the second time. Besides fixing poor calibration the amplitude and phase step also forces the two IF's to be on approximately the same flux scale, which minimizes error for bright sources. This is a practical matter right now. In an era with the new VLA correlator this probably would be done differently to keep track of the differences in the primary beam with frequency and probably to solve for the spectral index as well as the average total intensity across the band. Usually two selfcal passes are adequate, so this is not really in the DIFMAP limit.

B.11 Further uv Clipping

Once one has a good model one can subtract out the sources and do a uv editing step on the uv residuals. Normally, I subtract out the model with UVSUB, use UVPLT to plot the data, and CLIP to remove remaining high points. Sometimes a pass through TVFLG is used here. Then one adds the MODEL make into the remaining data with UVSUB again. Exactly how this step fits with VIII can vary.

B.12 Calibration of the Rest of the Data

Once one has a good image for the first night, one can use this model to calibrate the rest of the nights. Often just one amplitude and phase "selfcal" will be good enough. Also one needs to go through step IX on the rest of the data at this point.

B.13 Compressing the Full uv Dataset

Once again one wants to average the data in time and frequency before imaging. However, in addition, one wants to average the multiple days together to minimize the size of this full dataset.

In AIPS right now this involves going through a series of steps not intended for this purpose. One converts the times into HA's, edits the headers to make the individual datasets look like they all were observed on the same day, DBCONs all the datasets together, sorts them in baseline-time order, averages them optimally versus time, and sorts then back in time-baseline order for IMAGR. I probably am leaving a few steps out. Clearly a TASK to do all this would be a better approach. However, this effort and bookkeeping is well worth it because one can reduce a big database by a factor of 10 or more and thus speed up the imaging by almost the same factor.

B.14 Imaging the Full Dataset

At this point one normally makes an image with the full dataset, adds and resets the boxes again, does a selfcal using the optimally averaged dataset, and then makes an "almost" final image.

B.15 RR/LL Pointing Errors

At this point the residual pointing errors usually limit the result. The beam squint produces pointing differences between the R and L feeds which are understood to first order. For sources near the field center this is not too big a problem; however, for bright sources further for the pointing center the effect can be large and can be the big problem, scattering sidelobes over large regions of the map and increasing significantly the average noise level. This can be helped by a good choice of weighting at step IV. The pointing error also varies with paralactic angle as the position of a bright source relative to the pointing error vector changes.

The only way in current AIPS to deal with this problem is to divide the data into small ranges of paralactic angle, by polarization and by IF. Then one images and selfcals each subset separately, forcing the same clean beam for each one. Then one measures the noise on each resulting image and stacks them together weighting them optimally. Usually if I have 6 hours for each uv-track (only six hours to minimize the system temperature increases at low elevation), I divide the data into two time intervals, two IFs and two polarizations, producing 8 images in the end I must stack up.

Eventually, in AIPS++ one hopes this problem will be reduced by smarter selfcal/imaging programs which can 1) take into account the RR/LL pointing offset from the start, 2) solve for pointing errors with time and include them in the processing, 3) solve for local pointing errors for bright confusing sources and treat them separately in the processing. However, this processing may complicate the data compression issues and require a rethinking of the sequence of events in the processing.

B.16 Making the Source Catalogs

After the images are made there is still a lot to do to produce usable scientific results. The first, most important, step is to make a source catalog. To do this one first runs an automatic source finding/fitting task. In AIPS this is SAD. After adjusting a number of parameters and running the program on the first facet of the image, one produces an ASCII catalog. However, the same problems exist with this task as with the automatic boxing procedure. When one is in a region affected by the residual sidelobes of a bright source, the program has trouble tell the difference between sidelobe ripple and real sources. Thus one is required to investigate each source to be sure it is real.

I do this by using the ASCII list, the mouse and an almost trivial AIPS script called COVTLOD. COTVLOD allows me to specify a center position and an image size and then load a region of that size on the TV centered on the position specified. While it would be fairly easy for each user to write such a script, it is very efficient for one to be available in the package. I similar script, COSTAR, will also plot a symbol on the TV using the graphics overlay. I often plot a circle centered on the position to make recognizing the source easy. At this point I usually re-fit the flux and position of the source with JMFIT and copy the answer with the mouse to my summary table. This last step is probably unnecessary and is partly driven by a less than ideal format for the SAD output.

For large sources, I usually measure a total flux density with TVSTAT, make a contour/greyscale image with KNTR, and measure a total size from two extreme positions on the image. If there is a central component or other bright feature, I measure its position using JMFIT, TVSTAT, or in extreme cases with the IMPOS, which just reads the positions which the cursor is pointed to.

AIPS++ clearly needs the source finding/fitting/cataloging task. It also needs the graphics overlay and should have a script to load a small region around a given position as well as being able to mark particular positions on the TV.

B.17 Comparing with other catalogs

At this point one normally wants to compare the radio image with other images: radio, optical, IR, X-ray etc. In general other images will not have coordinates will not be well registered with the radio system. One needs a way to improve this. This is done in AIPS with a program called, XTRAN, which can take a set of known positions on the optical (or other type) image and calculate a transformation to a new coordinate system. One can then use the SAD process to make a catalog for this image. One can also use HGEOM to transform the image again to register with the radio image.

However, normally it is easier to mark all the radio source positions on the optical (or other) image and look for matches. This is done in AIPS by making an ST file. The ST file is, in turn, made by running a task, STARS, which converts an ASCII file into an internal AIPS extension file. One can then use TVSTAR to mark the positions of a large number of objects on another image. For optical identifications, one usually uses the marks the smaller number of radio sources on the optical image. One can also plot the ST files on greyscale or contour output images. Another use is checking the final catalog one makes in step XIV.

COTVLOD and COSTAR are extremely useful at this point. One often has an interesting object in a given field that someone has found something interesting about at another wavelength. Using these two scripts one can investigate the radio image at this position quickly and precisely.

It also is very useful to be able to plot a contour map from one band on top of a greyscale from another. For catalogs of multiple types of objects, it is useful to make several ST files and display each one in a different color.

While most of the latter capabilities are not imaging, they are necessary for a complete processing package which can produce actual scientific results without going into another package. AIPS++ must be able to

deal with data from other bands to be successful.

B.18 Summary:

AIPS++ seems to need several improvements to make it useful for deep, widefield imaging and several more for to take over all the processing in this area. Many of these should be easy. However, the uvdatabase, and improvements to the viewer seem like the most fundamental. A few others like dealing with the pointing/gain errors are clearly difficult but may be conceptually straight-forward. Much of the rest may be doable with GLISH scripts but may take a lot of time to tune adequately to make useful. It also is possible I have left some things out and may need to revise this memo as I realize what I have forgotten. But this is the process I believe is necessary as of right now.