



## **A Tigger User's Guide, by Oleg Smirnov**

# A Tigger User's Guide, by Oleg Smirnov

---



## 1. Introduction

Logged on 05/02/2012 07:13:25 PM

Tigger is both a standalone FITS image viewer and an LSM (local sky model) management tool. Since an LSM is just a source catalog, Tigger can also be pressed into service as a catalog viewer.

I developed Tigger because our LSMs were getting increasingly more complicated, and needed to represent a rich variety of source types -- point sources, Gaussians, FITS images, shapelets, etc. -- so the simple text-based formats we were using were clearly becoming inadequate to the task. Tigger defines a hierarchical OO-based LSM structure, which may be extended with arbitrary source types (point sources, Gaussians and FITS images are the ones currently supported). To manage LSMs, Tigger provides:

- \* a GUI-based LSM plotter & manager;
- \* a model conversion and manipulation script (tigger-convert), which supports (among other things) user-defined ASCII model formats;
- \* an API that allows arbitrary external model formats to be converted to/from Tigger models;
- \* a Python scripting interface to LSMs, for more elaborate (programmatic) manipulations of models;
- \* a MeqTrees interface, which allows any Tigger LSM to be directly mapped into a predict tree.

Since one of the most useful things one can do with an LSM is to plot it on top of a sky image (e.g. one derived from calibration using that LSM), I originally provided simple FITS viewing in early versions of Tigger. This quickly snowballed into a full-featured FITS viewer that drew heavily on kvis and DS9 for inspiration, but wrapped this in a modern Qt-based interface. As a result, Tigger can be enormously useful as a standalone FITS image viewer, even if you never deal with LSMs per se.

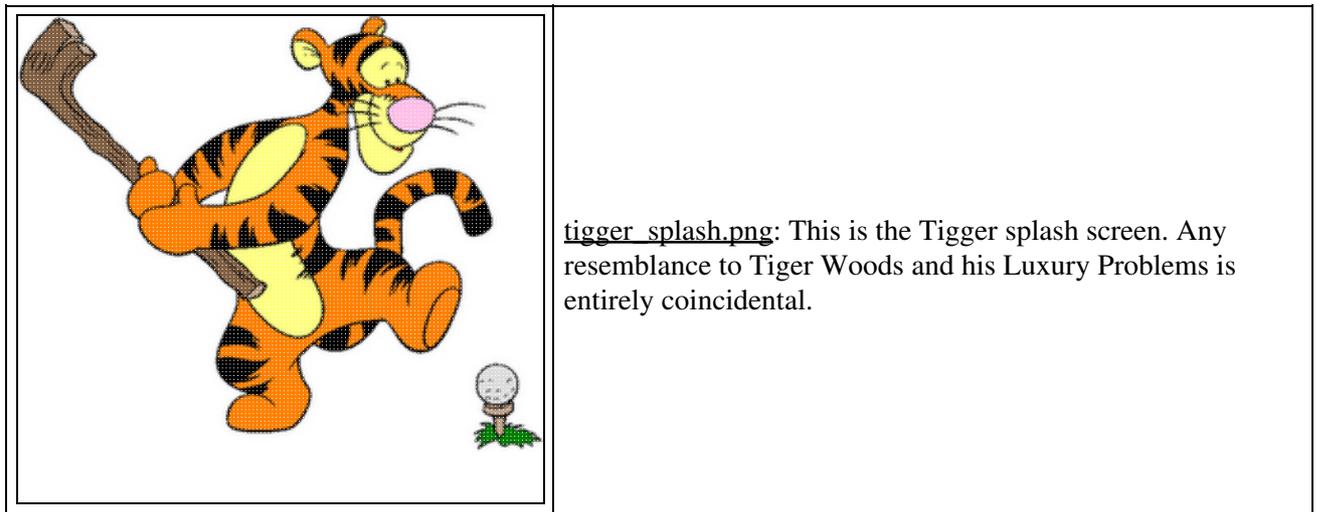
Tigger is written entirely in Python. A packaged version is distributed together with MeqTrees, but can be installed entirely independently of the latter (see <http://www.astron.nl/meqwiki>, follow the repository installation instructions, and install the "tigger" package). You may also choose to follow the latest development version by checking out the source from our Subversion repository (see <http://www.astron.nl/meqwiki/Tigger> for details).

Note that this User's Guide was written for Tigger svn revision 8753 or later (release 1.3); minor differences with older versions may exist -- and probably with future versions, if you're reading this in the distant future -- as Tigger is still being quite rapidly developed. With respect to older versions, most of the differences revolve

around mouse behaviour and menu items layouts. Most of the r8753 features are also available in somewhat older Tiggers, but may be harder to find or less logically laid out.

This User's Guide was produced with the PURR tool (PURR is Useful for Remembering Reductions -- and writing User's Guides, apparently), which is another standalone tool shipped via the MeqTrees repositories. In other words, this Guide is also a purrlog. A purrlog is an HTML document composed of a number of entries (of which this "Introduction" is one); each entry consists of a title (top of page), a text body (you're reading it), and, optionally, a number of "data products" or simply files (bottom of page). Data products may be downloaded simply by clicking on them. As you'll see below, this is actually a handy way to distribute a User's Guide, as it lets us ship things like demo FITS images and LSMs embedded directly in the guide. If you're reading a PDF-rendered version of the Guide, the links won't work -- in this case you may go to <http://www.astron.nl/meqwiki/Tigger> to find a "live HTML" version of the Guide.

## Data products



## 2. Loading FITS images

Logged on 05/02/2012 08:00:47 PM

To use Tigger as a FITS viewer, simply type

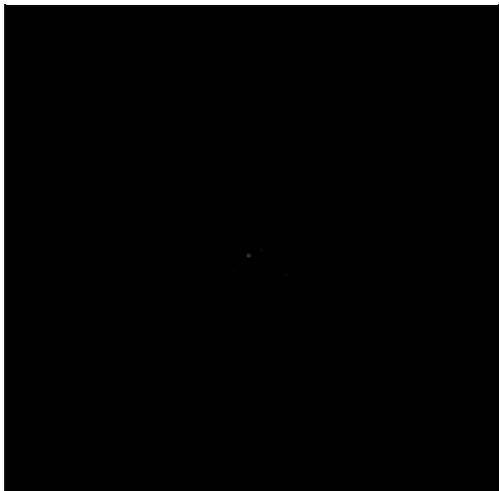
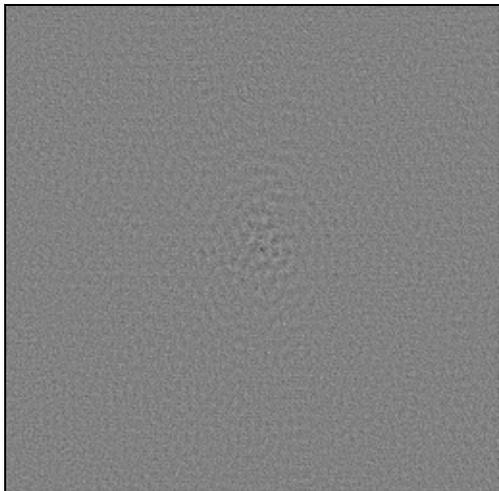
```
$ tigger filename.fits
```

An example FITS file is attached to this purrlog entry -- use the "3C147.fits" link below to download it to your current directory, then run "tigger 3C147.fits". The main display window of Tigger should come up, with

the famous 1.6 million dynamic range 21cm image of 3C147 in all its glory.

When used as a FITS viewer, the Tigger main window is laid out as per Fig 1 below. There is a menubar at the top of the window, a toolbar on the left, and a list of loaded images (in this case, just one) on the bottom.

## Data products

<p><u>3C147.fits</u> (<u>header</u>): A FITS image of the 3C147 field</p>	
<p>4096x4096x2x1 FITS cube, 2 planes are given below.</p>	
	<p>Image plane #0.</p> <p>data range: -0.00018127,21.5652  mean: 2.61289e-05  sigma: 0.0166717  clipping: 95%  clip range: -0.00018127,0.0419386</p>
	<p>Image plane #1.</p> <p>data range: -0.00479037,0.00048924  mean: -3.70499e-09  sigma: 1.27368e-05  clipping: 95%  clip range: -2.6347e-05,2.52118e-05</p>

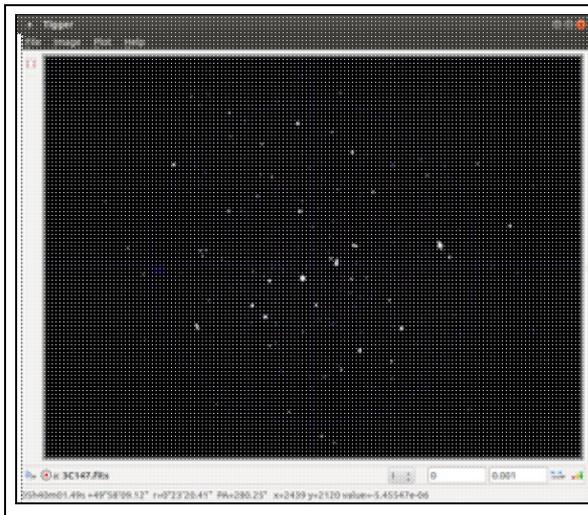


Fig2-1.png: Fig 2.1: Tigger main window, in FITS viewer mode

### 3. Mouse controls for image viewing

Logged on 05/29/2012 10:38:46 PM

When viewing a FITS image, various useful mouse functions are available for zooming and examining the image. Tigger is at its most convenient when used with a normal three-button mouse (and even better if it has a scroll wheel). However, because some laptops have two-button trackpads, some have three buttons but use the middle one to emulate a scrollwheel, and some cultist equipment provides just one very expensive (but extraordinarily elegant) button, the latest versions of Tigger support multiple, user-definable schemes for mapping commands onto mouse buttons and keyboard modifiers such as Shift, Ctrl and Alt.

You can switch between mouse schemes via the Plot | Mouse modes menu. The current scheme is saved to a config file and thus remembered across Tigger sessions. Pressing F1 at any time shows a quick reference for the current scheme (Fig 3.1).

The following mouse operations are available, with the button mappings given here corresponding to the default three-button "zoom/unzoom/measure" scheme:

- \* Zoom: click and drag the Left button to zoom into a window. Double-click the left button to zoom 2x at a specific point.
- \* Zoom in/out: click the Middle button to zoom back out to the previous view. You may then Shift+click the Middle button to zoom in again. Tigger maintains a "zoom stack" of viewing areas, so Middle button and Shift+Middle button effectively walks you up and down the zoom stack. Ctrl+Middle button zooms all the way back out -- note also the "Zoom out" button on the toolbar that does the same thing.

\* Mouse wheel zoom: if you have a mouse wheel (and "Mouse wheel zoom" is enabled in the Plot | Mouse mode menu), then rolling the wheel up will zoom x2 at the current pointer position, and rolling the wheel down will zoom out up the zoom stack. Note that there is a deliberate small delay before the zoom takes effect, during which time you may roll the wheel up or down some more to change the zoom level. This allows you to zoom by x4, x8, etc. in a single operation.

\* Stats: click and drag the Right button to select a window on the image. Tigger then displays some statistics for the pixels in that window. Note that when a stats box is selected, a "colour zoom" button on the left toolbar becomes visible (Fig 3.2). Clicking on this button will change the displayed intensity range of the image to the min/max of the selected stats box, and will set the current "data subset" to that box. This is explained in more detail in the section on "Histogram and intensity controls" below.

\* Measure: Shift+click and drag with the Right button to measure angular separation and position angle between two points (Fig 3.3). A simple Shift+click of the Right button displays the coordinates of the point you clicked on.

\* Select: these operations are available when an LSM is loaded, and will be documented in the "Manipulating LSM sources" section below.

Note that the information displayed by the Stats and Measure tools is also echoed on the text terminal where you ran Tigger, and copied to the clipboard. This makes it easy to paste it into a document if needed.

Finally, with some mouse schemes (notably, the fashionable single-button mouse) it is impossible to map all of the functions above onto the available button(s) and keyboard modifiers. Tigger gets around this by splitting the functions into a number of "modes", with the same key/button combination responsible for different things depending on which mode is in effect. The current mode is indicated by (and can be changed via) buttons on the toolbar on the left (Fig 3.4). You may also use F4 to cycle through available modes; the contents of the quick reference displayed with F1 are updated accordingly.

Mouse schemes may be redefined by editing a config file. The default schemes are read from a `tigger.conf` file located in some systemwide location (check the contents of your Tigger package or installation); you may override this by your own schemes defined in `.tigger.conf` in your home directory. To do this, first copy-and-paste the contents of the global `tigger.conf` into your local `.tigger.conf`, then start editing.

## Data products



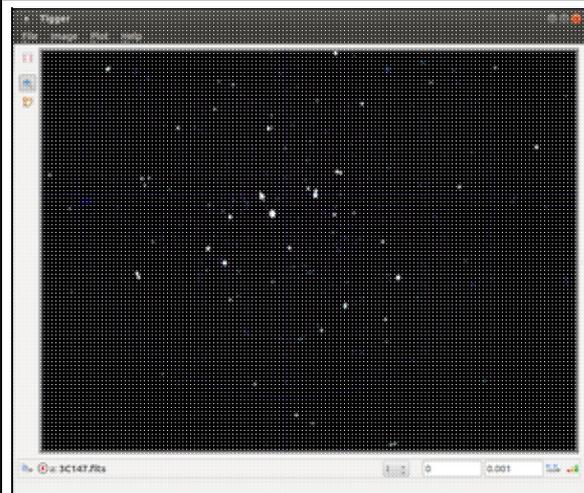
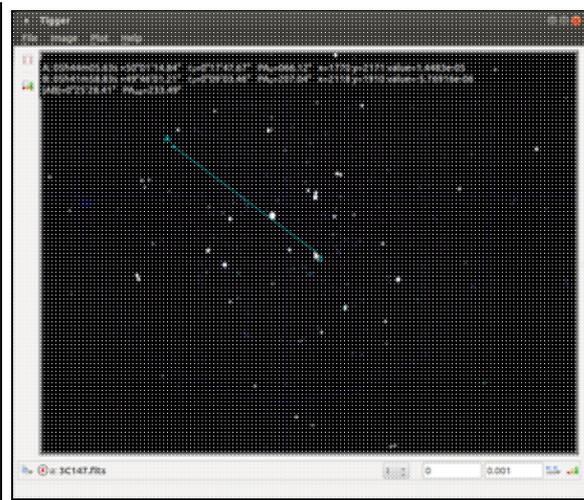


Fig3-4.png: Fig 3.4: Mouse modes -- mode selection buttons appear on the toolbar.

tigger.conf: config file defining mouse schemes

## 4. Working with multiple images

Logged on 06/01/2012 05:08:45 PM

We are now going to load a couple more images into Tigger. Download the spw0\_dirty.fits and spw1\_dirty.fits images attached to this purrlog entry. Then, use the Image | Load image... menu command (or press Ctrl+L) to load first one, then the other image. Your Tigger window should now look like Fig 4.1 below.

Tigger can display multiple images (possibly with different sizes, centres and resolutions) on the same plot, using their true projected coordinates. By default, only a single "topmost" image is actually rendered, while the other images are indicated by a rectangular frames (see frame for 3C147.fits -- which is bigger than the other two images -- on Fig 4.1). You may cycle through all the images by pressing F5, or use F6 to flip back-and-forth between the current image and the previous image. Note that Tigger uses a rendering cache -- cycling through the images is much faster the second time 'round, once they've all been rendered to the cache. Finally, you may enable the "Image | Display all images" option via the menu -- when this is enabled, Tigger will attempt to render all visible images simultaneously (rather than use frames). This rendering mode is somewhat slower, so it is off by default.

The image list at the bottom of the Tigger window contains a number of useful controls. From left to right, these are:

- \* A "Raise" button, which will raise the associated image to the top of the plot. The topmost image is indicated by bold font in the image list. Holding the mouse button down over the Raise button will display an image operations menu (note that the same per-image menus are available via "Image" on the menu bar). Note that pressing Alt+A for the first image, Alt+B for the second image, etc. is an alternative way to raise an image to the top.
- \* A roundrel icon (before the name of the image) indicating the current plot WCS (world coordinate system). Tigger can display multiple images having different WCSs; there is always one WCS defining the plot per se, and the others are projected into it. The roundrel icon indicates which image this "plot WCS" is associated with. Initially this is the first loaded image. If you go to the image operations menu for image spw0\_dirty.fits (either via Image | b: spw0\_dirty.fits, or by holding down the "Raise" button) and select "Center plot on image", roundrels will appear next to images B and C -- these have the same WCS, which is now the "plot WCS". Note that all three images share the same centre, so this actually makes little difference -- the difference is far more pronounced when displaying images with different centres.
- \* The name of the image. Hovering the mouse over the name displays some detailed information about the image.
- \* A Stokes plane selector, which you can use to flip between available Stokes planes (F7/Shift+F7 does the same). The selector is context sensitive, and only shows up when the images does have a Stokes axis. For images with e.g. a frequency or velocity axis, a channel selector will be shown as well.
- \* Two text fields showing the current intensity range. You can change the intensity range by entering new values directly into these fields, and pressing enter. Try changing the range of the spw1\_dirty.fits image to 0 to 0.0001 (Fig 4.2).
- \* An "Intensity unzoom" button. Clicking this will reset the intensity range to the full image min/max. Clicking and holding this button down shows a little "intensity zoom" menu, which provides shortcuts for setting the intensity range to span a certain proportion of pixel values: 99.99%, 99.9%, etc.

\* A "Lock" button. This locks together the intensity ranges of multiple images. Try locking together `spw0_dirty.fits` and `spw1_dirty.fits`, then changing the intensity range of one of them. Holding down the "Lock" shows a little menu with options for locking all images together, or unlocking all images.

\* Finally, the rightmost button is the "Colours & intensities" button. Pressing this will bring up the Histogram & Intensity control dialog, which is the subject of the next section. Note that pressing F9 does the same thing for the currently topmost image.

Last but certainly not least, a great feature of Tigger is the ability to do on-the-fly image math. Select "Image | Compute image..." (or press Ctrl+M). In the next dialog box (Fig 4.3), enter "b-c". You will immediately see the result of image B minus image C -- this will be added to the image list (as image D), and rendered in the plot. This is a temporary image that is only held in memory at this point -- but you can save it into a FITS file of its own by clicking the "save" button in the image list at the bottom of the Tigger window.

Image math is enormously powerful, as arbitrary numpy expressions are supported. Whatever you enter in the dialog is evaluated using the Python interpreter, with images mapped to the variables "a", "b", "c", etc., and with all symbols from numpy imported into the local namespace. If the result evaluates to a numpy array, this is treated as a new image. Any errors (exceptions) are caught and reported back to the user. In particular, this implies that:

\* All symbols and functions defined in the numpy module are available. Thus, to take the cosine of an image, simply use "cos(a)" rather than "numpy.cos(a)".

\* The various sub-modules of numpy, such as `numpy.fft`, are directly available. Try computing the expression "fft.fft2(b)" to take a quick FFT of an image.

\* The arrays involved in the expression must have the same shape, or an error will be reported. For example, with the images loaded here, "b-c" works but "a-c" doesn't, since image A has a different size. You can use numpy slicing syntax, though, to extract a properly-sized subset of A: "a[1024:3072,1024:3072]-b" will do the job just fine.

\* Speaking of numpy slicing, if you know the syntax, you can do some really powerful things. For example, "b-b.mean()[:,newaxis]" will subtract from each row of the image the mean value across that row.

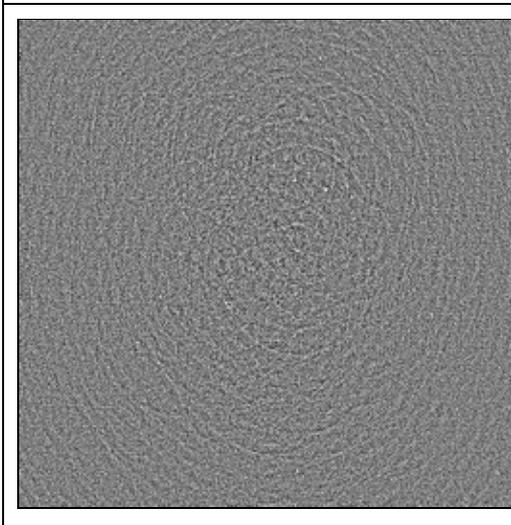
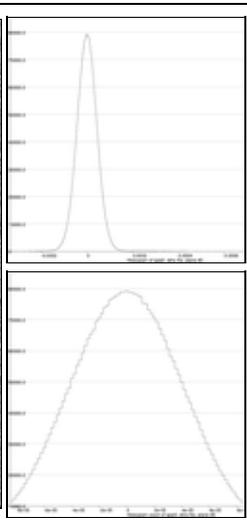
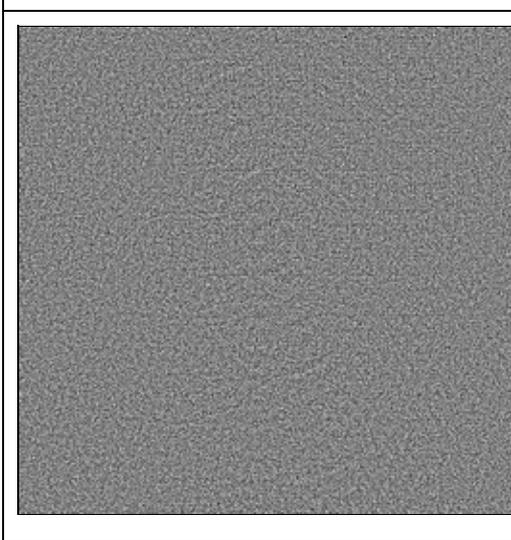
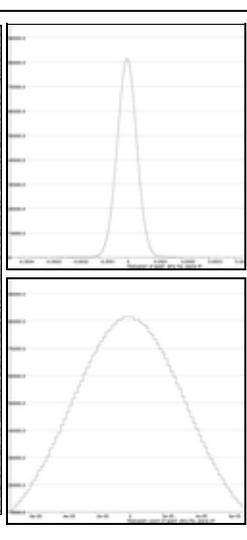
\* Tigger tries to set the WCS of the resulting image from another existing image with the same dimensions. If multiple images with different WCSs are loaded, it will bring up a dialog letting you choose which WCS to use.

## Data products

---

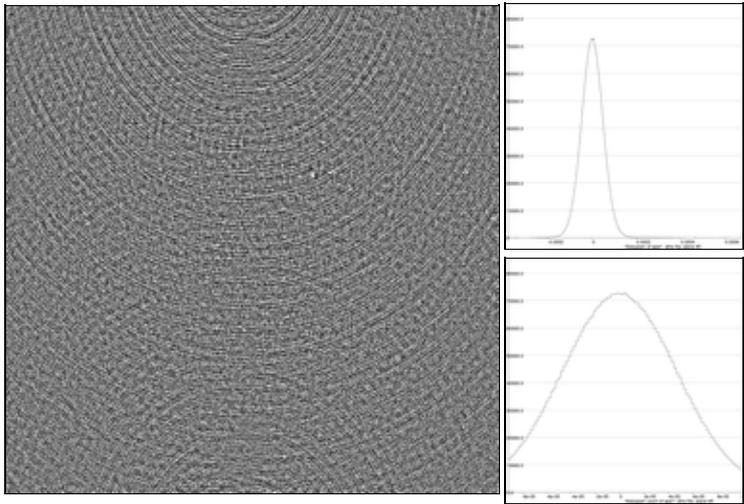
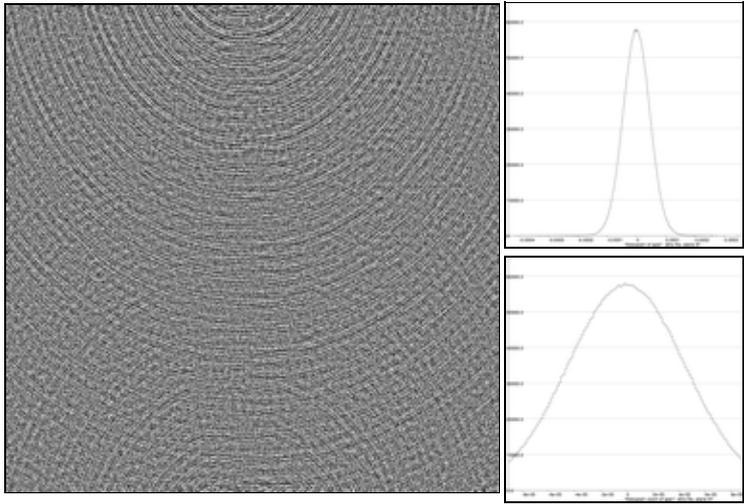
`spw0_dirty.fits (header)`: dirty image, band 0

2048x2048x2x1 FITS cube, 2 planes are given below

			<p>Image plane #0.</p> <p>data range: -0.000335587,0.00068</p> <p>mean: 1.02594e-10</p> <p>sigma: 4.27722e-05</p> <p>clipping: 95%</p> <p>clip range: -8.45253e-05,8.48419e-05</p>
			<p>Image plane #1.</p> <p>data range: -0.000431637,0.00043</p> <p>mean: 3.57251e-10</p> <p>sigma: 3.47935e-05</p> <p>clipping: 95%</p> <p>clip range: -7.1085e-05,6.87552e-05</p>

spw1\_dirty.fits (header): dirty image, band 1

2048x2048x2x1 FITS cube, 2 planes are given below

	<p>Image plane #0.</p> <p>data range: -0.000379942,0.00067</p> <p>mean: 5.42751e-09</p> <p>sigma: 4.82299e-05</p> <p>clipping: 95%</p> <p>clip range: -9.15289e-05,0.00010</p>
	<p>Image plane #1.</p> <p>data range: -0.000437787,0.00035</p> <p>mean: 3.13532e-09</p> <p>sigma: 4.53562e-05</p> <p>clipping: 95%</p> <p>clip range: -9.07831e-05,8.8165e-</p>
	<p><u>Fig4-1.png</u>: Fig 4.1: Tigger window with three images loaded.</p>

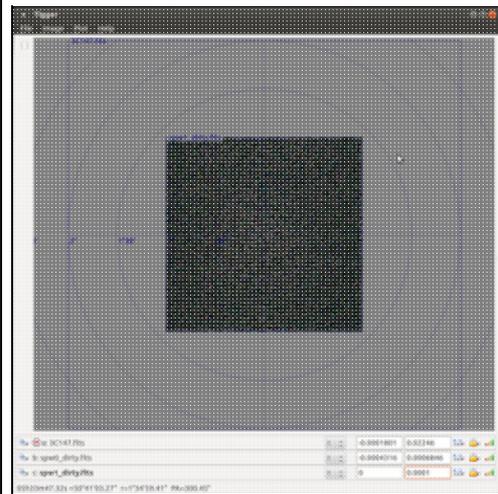
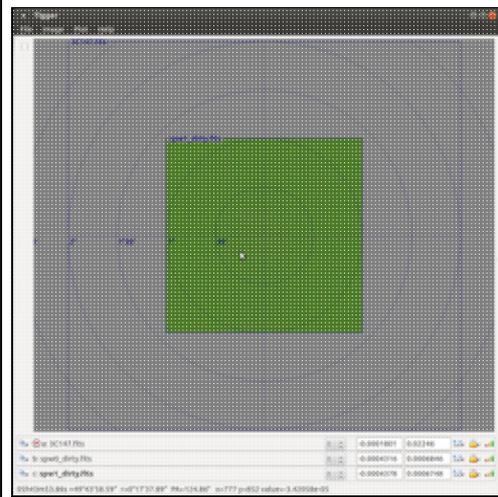


Fig4-2.png: Fig 4.2: Changing the intensity range for the third image

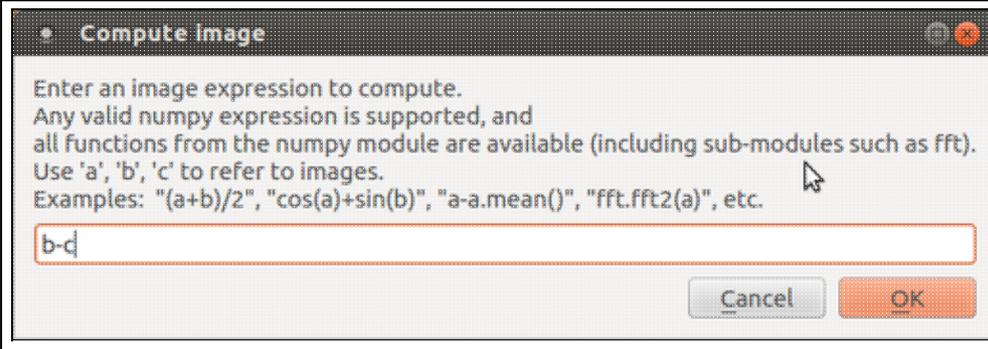


Fig4-3.png: Fig 4.3: Image arithmetic dialog.

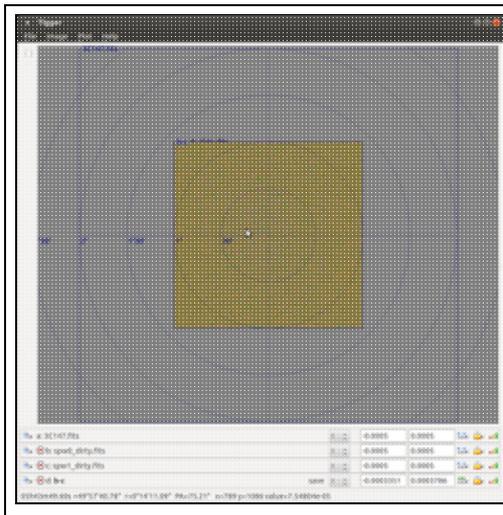


Fig4-4.png: Fig 4.4: "b-c" image appears in image list.

## 5. Colour and intensity controls

Logged on 06/01/2012 07:50:40 PM

The colour control dialog can be accessed by pressing the "Colours & intensities" button next to the image name (this is the rightmost button). Note that every image has its own independent dialog. Fig 5.1 shows this dialog for the spw0\_dirty.fits image.

The top part of the dialog is occupied by a histogram of pixel values. This offers a number of interesting features:

- \* the histogram can be shown on a linear or logarithmic Y axis: use the "log Y" checkbox to change this.
- \* the mean, max bin, rms and half-max values are indicated by markup on the histogram plot.
- \* the current intensity transfer function (ITF) is indicated by a blue line plotted over the histogram, sharing its X axis (pixel value).
- \* the current colour map (as a function of pixel value) is indicated by a colour bar over the histogram, sharing its X axis.
- \* the currently rendered intensity range is indicated by a grey box on the histogram plot. You can change this range by clicking directly on the histogram: the left button sets the low end of the range, and the right button (or Ctrl+Left, on single-button mice) the high end. Now click left and right on two points around the main body of the distribution.

\* As soon as this happens, the grey box is adjusted, and the histogram is automatically zoomed into approximately the selected area (Fig 5.2). Note that the auto-zoom behaviour may be disabled via the checkmark on the top.

\* Other ways to zoom in and out of the histogram are the two "+" and "-" buttons to the top of it, or else the zoom wheel between them. Finally, the rightmost button zooms all the way out to the full extent of the histogram.

The next section of the dialog is called "Data subset". This indicates both the currently visible image slice (Stokes I in this case), and the data subset for which the histogram is rendered (and for which statistics are shown). In this case the current subset is the same as the current slice, i.e. "Stokes I". Using the "->full" and "->slice" buttons to the right, the subset may be switched between the full image cube, and the current slice. Note that the histogram, statistics, and the intensity range is updated accordingly (if the subset is too large, Tigger will not immediately compute the mean and std values, as that may take a few seconds -- a "more..." button is displayed instead, which you can press to perform the computation).

Note that this offers an interesting interaction with the "Stats box" operation discussed in section 3 above. Use the appropriate mouse button (right button, by default) to select a box in the image -- you do not need to close the Colour Controls for this. You will see some stats for that image region, but also a "colour zoom" button will appear on the toolbar to the right. Pressing this button will set the current data subset to that image region (Fig 5.3) -- the histogram will be redrawn to match, and the intensity range will be set to the min/max of the selected box. (You can then use the "->full" or "->slice" buttons to reset the data subset back.)

The next section of the dialog is called "Intensity mapping". This contains:

\* a couple of text fields for the low and high end of the intensity range, which work exactly like the fields in the image list of the main Tigger window.

\* In addition to this, a "->0" button resets the low end to zero; a "to histogram" button resets it to the currently displayed area of the histogram, and the "Reset to" drop-down menu offers a number of other predefined intensity ranges.

\* An "intensity policy" control selects between linear, logarithmic and histogram-equalized intensity transfer functions. Note how the ITF curve in the histogram plot is updated when you change the ITF here.

\* The "lock display range", "lock all to this", and "unlock all" buttons work exactly like their counterparts in the image list of the main Tigger window -- see the previous section for details.

The "Colourmap" selector offers a number of colour mapping schemes. When set to the wonderful "CubeHelix" scheme developed by Dave Green (<http://www.mrao.cam.ac.uk/~dag/CUBEHELIX/>), a number of extra slider controls below the colourmap selector allow the parameters of the colourmapping to be adjusted. Note that the colourmap shown in the dialog is updated continuously as you drag the sliders around,

but is only applied to the image when you release a slider.

## Data products

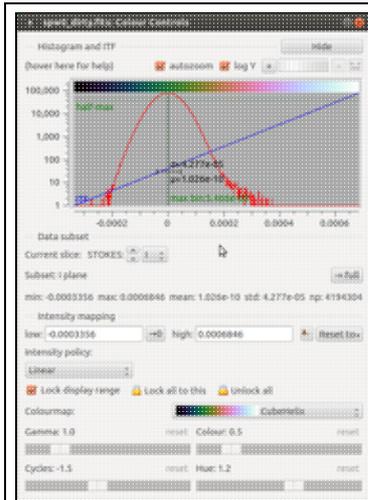


Fig5-1.png: Fig 5.1: Colour controls dialog.

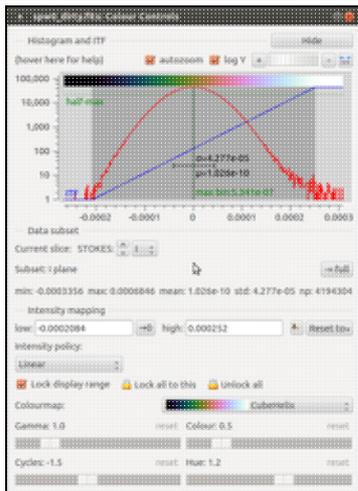
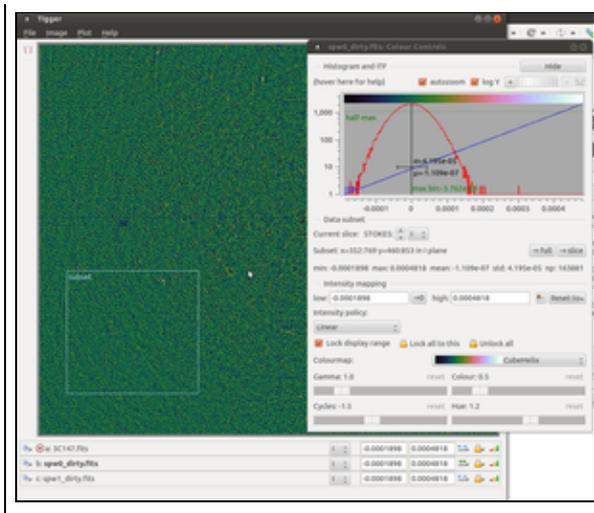


Fig5-2.png: Fig 5.2: Changing the intensity range and zooming into the histogram.

Fig5-3.png: Fig 5.3: Focusing on a subset of the image.



## 6. Other image viewer features

Logged on 06/01/2012 08:21:21 PM

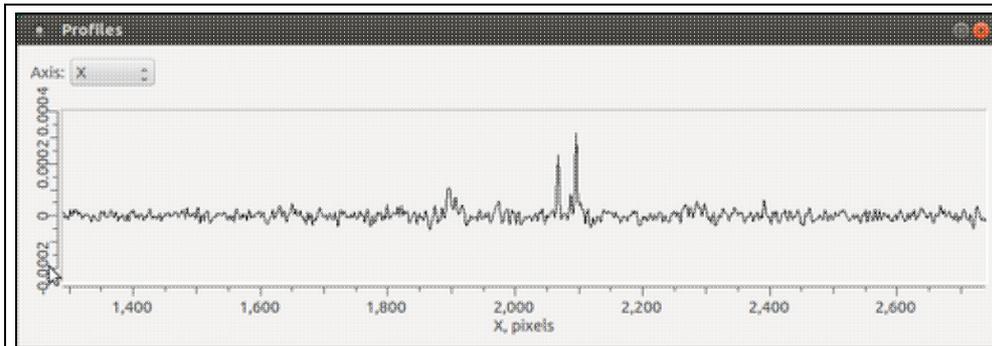
A few more useful Tigger features are:

- \* Press F3 to show or hide a "Live profile" window. This is continuously updated to show a cross-section through the image at the current mouse position. The axis along which the cross-section is taken can be selected at the top of this window (note that this can be a frequency axis, if your FITS image contains one).
- \* Press F2 to show or hide a "Live zoom" window. This is continuously updated by a close-up of the image area under the mouse cursor, along with X and Y cross-sections through that area.
- \* The "Plot | Fix aspect ratio" menu option enforces a fixed aspect ratio (i.e. square image pixels). Normally this is on. Turn it off if you want to zoom into a rectangular area of the image, and stretch the pixels accordingly.
- \* The "Plot | Show PSF" option shows a rendering of the PSF size (aka beam size), if known from the FITS header, as an ellipse in the top-right corner of the plot.
- \* The "Plot | Show grid circles" option determines the stepping of the angular distance indicators, or whether they're shown at all.
- \* The "Export plot to PNG file" (Ctrl+F12) command can be used to save the current plot to a PNG file. Note that this renders the current plot contents exactly as they appear in the plot window (i.e. at the current zoom level, with grid circles, with perhaps an LSM plot on top, etc.) You may also export individual images to

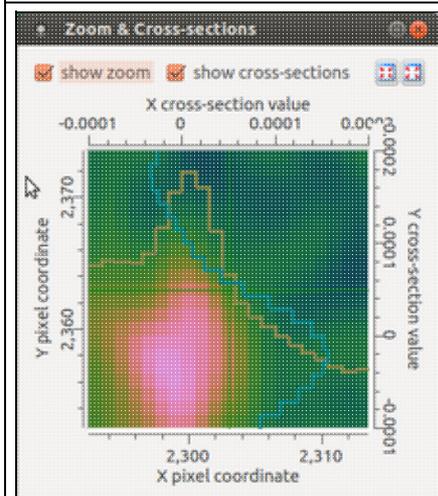
PNG, via the "Image | x: foo.fits | Export image to PNG file" option. The latter will export an image by itself with nothing else, at its native resolution, using the current intensity and colour mapping.

This concludes our tour of the FITS viewer features. This, however, is only half of what Tigger can do -- to see the other half, we need to start working with sky models (LSMs).

## Data products



[Fig6-1.png](#): Fig 6.1:  
Live profile display



[Fig6-2.png](#): Fig 6.2:  
Live zoom &  
cross-section display

## 7. Working with local sky models

Logged on 06/02/2012 12:13:34 PM

A local sky model is essentially just a structured sky catalog. One such model (for the 3C147 field) is attached to this entry. You may download it to your local directory, then view with Tigger:

```
$ tigger 3C147.lsm.html
```

Note that a Tigger-format sky model is just an HTML file at the same time, so you can view it with a web browser (or place it online) if nothing else is handy.

When viewing sky models, the Tigger window looks somewhat different (Fig 7.1). What's even more useful is combining a sky model with an image. Use the "Image | Load image" menu (or press Ctrl+L), and load the 3C147.fits image (alternatively, we could have specified both an image and an LSM file on the command line when starting Tigger). The combined view contains both the rendered image, and a plot of the sky model (Fig 7.2), which is a view that is especially useful when doing calibration or other data reduction.

In LSM mode, the main Tigger window is divided into three panels. The dividers between the panels feature grab handles which can be used to resize them relative to one another. The three panels are:

- \* A table of LSM sources on the top.
- \* A plot window on the lower-left.
- \* A groupings/styles table on the lower-right (combined with an optional image list at the bottom), which will be documented in a later section.

The LSM table shows the main attributes of all the LSM sources, such as name, position, flux, apparent flux ("I(app)"), etc. Click on the column headings at the top of the table to re-sort it by that attribute. You can also use the standard click/Shift+click/Ctrl+click techniques to select rows of the table; the sources you select in this manner are also indicated in the plot. Note that a count of selected sources is given in the Source groupings table (third row).

Note also that when you move your mouse over the rows of the table, the corresponding source marker is highlighted in the plot. Conversely, you can move the mouse over source markers in the plot while holding down Ctrl (in the default 3-button mouse scheme -- see Section 3 above for details) to highlight those sources and show their table entries.

Finally, a few words on source names. These can be arbitrary, unique identifiers. However, Tigger has a preferred naming convention (which may be imposed via `tigger-convert --rename`, see Section 10 below) called COPART, for "cluster ordering, P.A., radial distance, type". The 3C147.lsm.html model provided here follows this convention. Under COPART, a source name consists of:

- \* A letter group indicating how bright the source (or its cluster -- see below) is, relative to others in the model. The 26 brightest sources are A to Z, respectively, followed by aa to az, followed by ba to bz, etc.
- \* Two digits indicating the position angle of the source relative to the centre of the LSM, in degrees\*10.

\* One digit indicating radial distance of the source from the centre, in user-defined steps (the step size is defined on a per-model basis, in case of the 3C147 LSM here it is 10').

\* An optional lowercase character suffix indicating the order of the source in its "source cluster". Closely-grouped LSM sources are normally assumed to be components of the same complex sky source, and are normally "clustered" together by giving them the same name with different suffixes. Thus the brightest source in the cluster may be called A182, the next-brightest A182a, A182b, etc.

\* An optional uppercase suffix indicating the type of the source. This is "G" for Gaussians, and nothing for point sources.

COPART allows names to carry some meaningful information about the position of a source in the field. Thus, a name like "C182a" can immediately be attributed to the third-brightest source cluster in the model -- and the second-brightest source in the cluster -- located due South of field centre (PA ~180 degrees), 20' to 30' from centre.

Note that source clustering is quite important in the calibration process. In particular, if direction-dependent gain solutions are obtained, it is useful (for all sorts of reasons) to apply a single solution to the entire cluster. MeqTrees can be asked to make use of clustering information by telling it to e.g. "Use a unique xx-Jones term per 'cluster'". In this case, the "cluster" tag of each source (see the discussion on tags below) is used to determine how to assign unique Jones terms to sources, so that sources with the same value of "cluster" use the same Jones term.

## Data products

[3C147.lsm.html](#): 3C147 LSM file

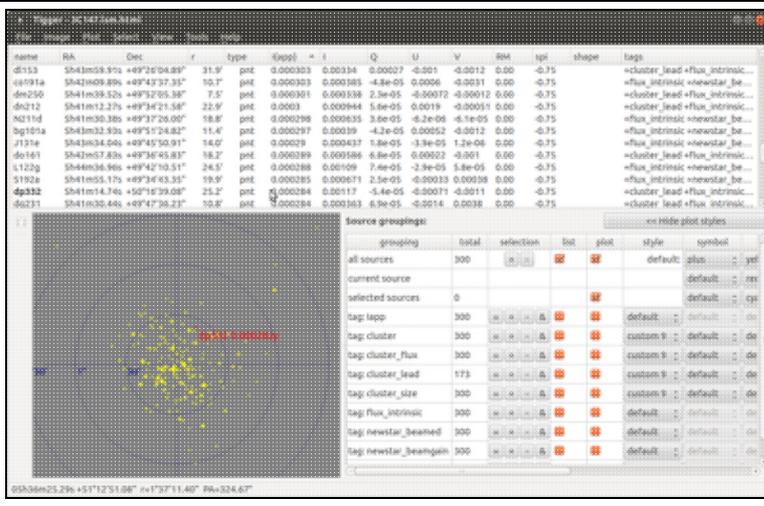


Fig7-1.png: Fig 7.1: Tigger as an LSM viewer.

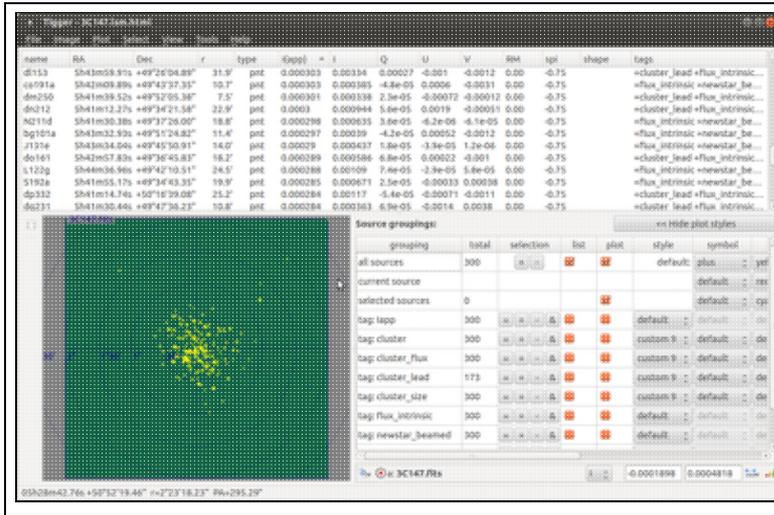


Fig7-2.png: Fig 7.2: LSM view combined with image view.

## 8. Selecting sources and working with source tags

Logged on 06/04/2012 12:54:35 PM

Source tags are arbitrary "name=value" pairs associated with model sources. They can be used both to provide additional information on a source (e.g. "r" for distance from field centre, "newstar\_beamgain" for NEWSTAR-derived sky models with a primary beam applied, etc.), and to identify subsets of sources for special processing. The latter feature is especially useful in MeqTrees, where a common use case is to tell a calibration script to e.g. "apply a differential gain term to all sky model sources with the tag 'dE'", etc. The use of source tags is prevalent throughout MeqTrees scripts -- every time a subset of sources needs to be specified, it may be done through the use of tags.

You can add a tag to a group of sources by selecting them in Tigger, then using the "LSM | Tag selection" menu option. Likewise, you may remove tags via the "Untag selection" option. For example, let's can select the sources B\*, C\* and D\* (click on the "name" column heading to sort sources by name, then click on source B232 to select it, then scroll down to D141c and Shift+click on it to select all the sources in between, see Fig 8.1), then use "LSM | Tag selection" (or press Ctrl+T) to open the dialog, enter "dE" for the tag name, and press OK. You will now see an entry for "tag: dE" appear in the Source groupings table, showing you that 23 model sources now have the "dE" tag (Fig 8.3).

Note that sources can also be selected with the mouse. The following selection operations are available (see also the discussion on mouse scheme in Section 3 above):

\* select sources in a rectangular window (using Shift+LeftButton in the default 3-button mouse scheme). This new selection replaces the current selection, if any.

\* select sources and add them to the current selection (Shift+Ctrl+LeftButton).

\* deselect sources in a window (Shift+Alt+LeftButton).

Now note how you can use the Source grouping table to manipulate the current selection. Every row of the table has four buttons marked "=", "+", "-" and "&". The "=" button selects all the sources -- and only the sources -- in that group. The "+" button extends the current selection with the given group. The "-" button deselects all the sources in the group. The "&" button intersects the current selection with the group -- i.e. deselects all sources NOT in the group.

For example, click "-" on "all sources" to deselect all sources. Now click "+" next to "cluster\_lead" -- this selects 173 sources with the "cluster\_lead" tag (more on what this tag means later). Now click "+" next to "dE" -- you have now selected 193 sources with either of the tags set. Click on "=" next to "cluster\_lead" to select the 173 sources with that tag again. Now, click on "&" next to "dE" -- you have now selected the 3 sources that have both tags set. Now, use "Select | Add tag to selection" to give these sources a tag called "cluster\_lead\_dE".

As a more useful exercise, here's how you select sources based on a combination of attributes. Activate the "LSM | Select by attribute" command in the menu. In the resulting dialog, select "r" and ">". Then start dragging the slider around -- this selects sources based on their distance from field centre (the units for "r" are radians, which is admittedly not very convenient, and needs to be fixed in a future version). With the slider set at 50%, you have selected the "outer" half of the model (Fig 8.4). Now use "LSM | Add tag to selection", and assign these sources a new tag, call it e.g. "outer\_half".

Next, use the "LSM | Select by attribute" dialog to select source with an apparent flux of greater than 1 mJy ( $I_{app} > 0.001$ ). This selects about 67 sources. Now click on "&" next to the "outer\_half" entry in the Source groupings table. You now have selected the 32 sources that are brighter than 1 mJy, and lie in the outer part of the field. Use the "LSM | Add tag" dialog to tag them with e.g. the tag "outer\_1mJy".

Finally, the Source groupings table may be used to set up a range of custom plot styles based on model plot tags. This is controlled by widgets in the right half of the table (Fig 8.5), as follows:

\* The following style attributes may be changed: plot symbol, plot symbol colour, label, label colour, label size.

\* The top row of the table ("all sources") determines the default plot style for model sources.

\* The second and third row determines the plot style of the selected sources, and the currently highlighted source, respectively.

\* The remaining rows may be used to set up custom per-group styles. For a source grouping to be displayed in its own style, first of all the selector in the "style" column must be set to "custom #". (As long as the style

selector is set to "default", the group is plotted in the default style, and the other controls are disabled). Then, the other controls may be used to change various plot attributes.

\* If a source is present in multiple groupings, then its plot attributes are a combination of the custom styles set up in the table, with lower-numbered "custom #" styles having priority.

In Fig. 8.5, we have set up custom plot style for the "clusted\_lead\_dE", "dE", and "outer\_half" groupings that we created earlier.

Finally, the "list" and "plot" columns of the Source groupings table may be used to plot and list only a subset of the model sources. Each grouping has a pair of associated list/plot checkboxes with three states: ON, OFF, and neutral. A source will be listed in the table / included in the plot if it belongs to a grouping whose list/plot checkbox is ON, and does not belong to a grouping whose list/plot checkbox is OFF. The two "all sources" checkboxes control global visibility. The default initial state is for "all sources" to be list ON, plot ON, and all the other checkboxes to be neutral. This causes all sources in the model to be listed and plotted. By setting the "all sources" checkboxes to OFF, and then setting a few groups to be selectively ON, we can plot and list subsets of sources (Fig 8.6).

## Data products

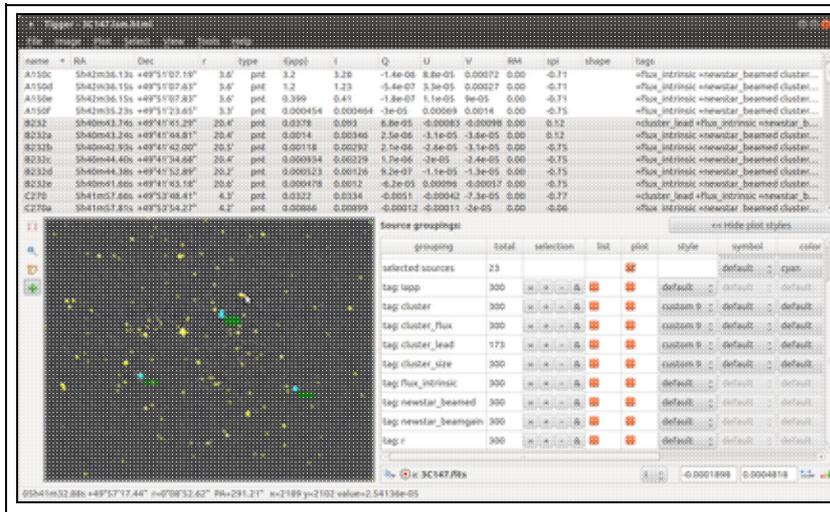


Fig8-1.png: Fig 8.1: Selected sources B\* -- D\*

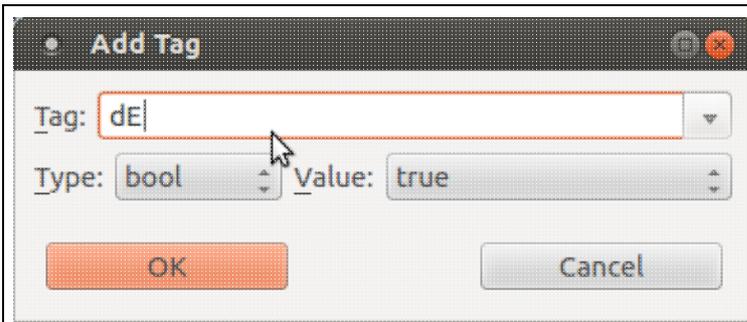


Fig8-2.png: Fig 8.2: Tag selection dialog.

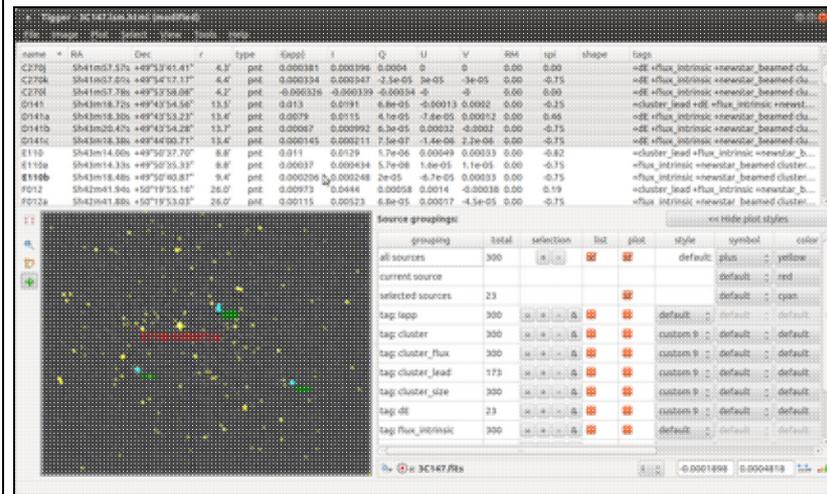


Fig8-3.png: Fig 8.3: Source groupings now has a dE entry

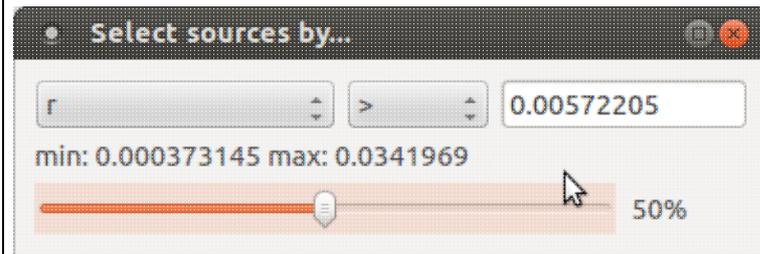


Fig8-4.png: Fig 8.4: Selecting "outer" sources

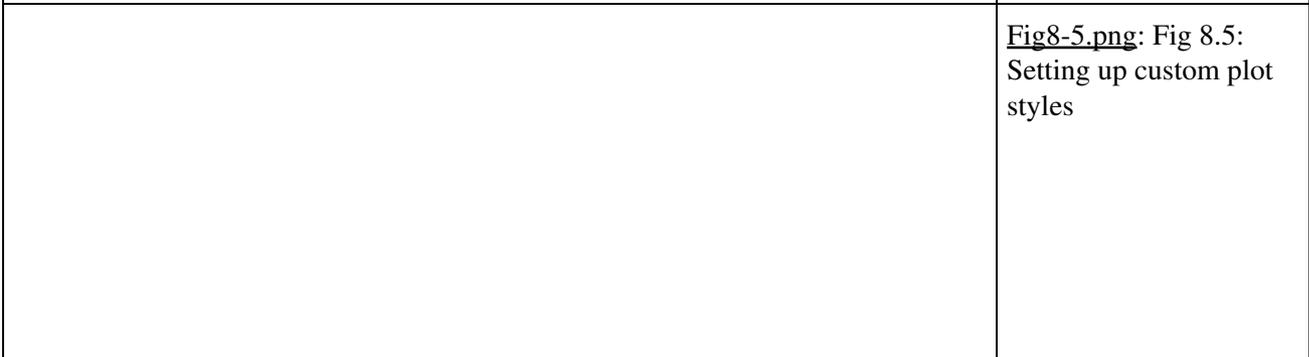


Fig8-5.png: Fig 8.5: Setting up custom plot styles



Fig8-6.png: Fig 8.6: Displaying subsets of sources

## 9. Bricks, Gaussians and hybrid models

Logged on 06/05/2012 06:10:34 PM

A crucial feature of Tigger is support for "hybrid" sky models containing other source types besides point sources. This is illustrated by the A773 sky model attached to this log entry. This LSM was borrowed from a real-life reduction of some WSRT low-frequency data.

Copy the A773 tarball to your directory and untar it, then load the sky model therein into Tigger. You will see that this LSM contains three types of sources (look at the "type" and "shape" columns of the LSM table, and note also the three "type" entries in the Source groupings table):

- \* point sources, which are familiar from the previous sections

- \* Gaussian sources

- \* FITS images (also called "bricks" in MeqTrees parlance), which for this particular LSM come in two flavours -- a traditional single large image of clean components, and a number of small "postage stamp" images corresponding to individual sources.

Of course being able to represent this information in an LSM is only half the functionality -- the other half is being able to use it in calibration. At present only MeqTrees has support for all three sources types. A Tigger model such as this one, when loaded into MeqTrees, will cause proper predict trees to be generated automatically. In the future this will be extended with other source representations such as shapelets.

In the meantime, Tigger provides a number of useful tools to manage FITS image-based models. These are accessed via the Tools menu. To demonstrate them, reload Tigger with the 3C147.lsm.html model from before, and copy the 3C147brick.fits file attached to this entry into your current directory. We will now examine the available tools.

The "Add FITS brick" option adds a FITS image to your model. Use this if you have e.g. a ready-made image of clean components, or a deconvolved image of an extended source. You need to specify a FITS file, a name (identifier) under which this brick will be added to the model, and a padding factor (use a factor of 1.5~2 if your brick has significant flux towards the edges, as this reduces aliasing. If your brick only has flux in e.g. the inner half of the image, then a padding factor of 1 is perfectly fine). See Fig 9.2 for an example. If you click "OK" in the dialog, the specified FITS image will be added to the LSM as a brick source. Do not save this LSM -- we only did this as a demonstration -- instead reload the original 3C147.lsm.html and examine the next tool.

The "Make FITS brick from selected sources" tool implements the rather common use case of moving a number of faint model sources into a brick, for prediction en masse (as pixels of an image). The rationale for this is that a brick provides a much more efficient (FFT-based) means of predicting source visibilities, compared to the DFT-based approach of using individual delta functions. In the case of the 3C147 field, we could probably predict all the sub-mJy sources (which form a large part of the model) via an image, with very little loss of accuracy.

To demonstrate this feature, proceed as follows. First, load the original 3C147.lsm.html model. Next, prepare a FITS image into which the sources will be moved. (The data in this FITS image will be overwritten, so its real purpose is to provide the "scaffolding" of a coordinate grid. Tigger is at present unable to make new FITS images from scratch.) We'll use the 3C147brick.fits image for this. Load it into Tigger alongside the LSM using "Image | Load image", or Ctrl+L. Next, use the "LSM | Select by attribute" tool to select all sources with  $I_{app} < 0.001$  (Fig 9.3). As you can see, some sources lie outside the area spanned by the image. Deselect these sources using Shift+Alt+LeftButton (if you have the default 3-button mouse scheme enabled -- otherwise use F1 to look up how to do "Deselect sources" -- you may need to put the mouse into "Selection mode" with F4.

See also the discussion on mouse schemes in Section 3.), along with a few of the sources lying at the very edge of the image. You should now have about 228 sources selected (Fig 9.4).

Now, activate the "Tools | Make FITS brick from selected sources..." dialog (Fig 9.5). You need to supply the following information:

- \* The FITS file to use (3C147brick.fits). Note that this can have Stokes and frequency channels; the spectral index and polarisation of LSM sources should be handled correctly.
- \* If the "Apply primary beam" option is checked, the flux of the sources will be converted into apparent flux before placing them into the brick. MeqTrees does not apply a primary beam when doing FFT-based model predictions, so the image needs to be pre-scaled by a power beam, which is what this option does. The additional information it requires is an analytic expression for the primary beam, and a reference frequency to use therein. If the LSM already contains a PB expression as an optional attribute (which the 3C147 LSM we use here does), this expression is helpfully copied into the dialog. Note that the "I(app)" attribute of sources is ignored during this operation -- it is the proper IQUV fluxes that are used with the supplied PB expression.
- \* If "overwrite image" is in effect, the FITS image will be nulled before filling it with sources. Use "add into image" to preserve the old contents instead.
- \* If "add image to sky model" is checked on, the new brick will be automatically added to the LSM. We usually want this on.
- \* If "remove from the sky model sources that go into the brick" option is in effect, sources converted into image pixels will be removed from the LSM automatically. We usually want this on.

Now press OK; and use "Image | 3C147brick | Unload image" to refresh the brick image (as the underlying file will have changed). Set the image intensity limits to 0 and  $1e-5$ . You should now see something like Fig 9.6. Zoom into the image for a closer look -- all the fainter LSM sources have been turned into image pixels with the corresponding intensity. The image itself has been added to the LSM as a brick-type model source. You may now save this LSM (under a different filename if you prefer), and use it in a calibration with MeqTrees.

Note also that the "make brick" operation is also available as a standalone script, see "tigger-make-brick --help" for help.

The final image-related tool is called "Restore model into image". This is a common operation performed after calibration. The typical outputs of a MeqTrees calibration pipeline are corrected residual images -- these have had all the LSM sources subtracted out, and (hopefully) only show the faint background. To turn these into a final product, one needs to "restore" model sources back into the images, using a Gaussian restoring beam. This is the tool for doing this (also available as a standalone script, see "tigger-restore --help").



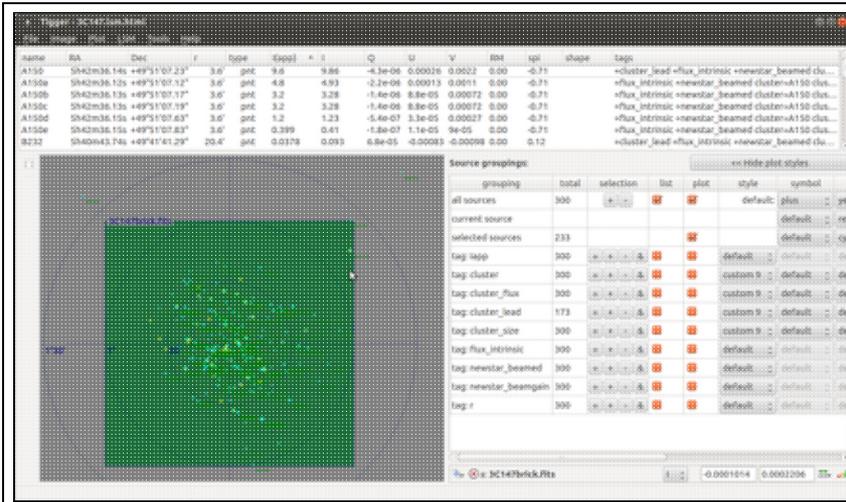


Fig9-3.png: Fig 9.3: Loaded 3C147.lsm.html and brick image of selected sources Iapp<0.001

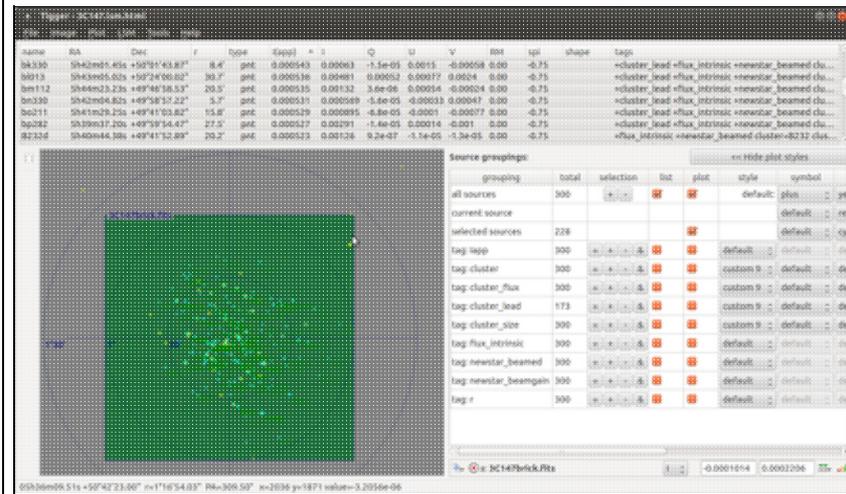


Fig9-4.png: Fig 9.4: Selected sources for moving to brick

Fig9-5.png: Fig 9.5: Convert sources to brick dialog

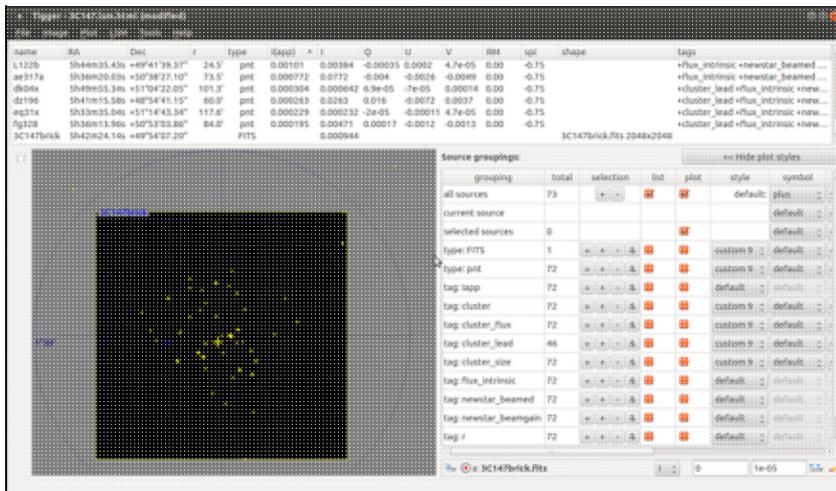
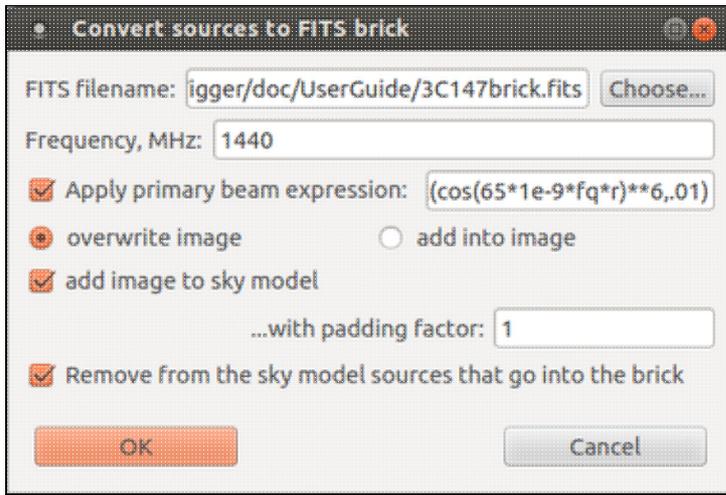


Fig9-6.png: Fig 9.6: Sources converted and brick loaded

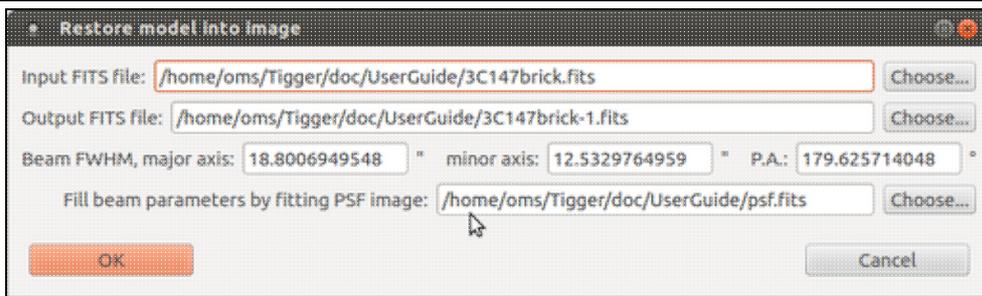
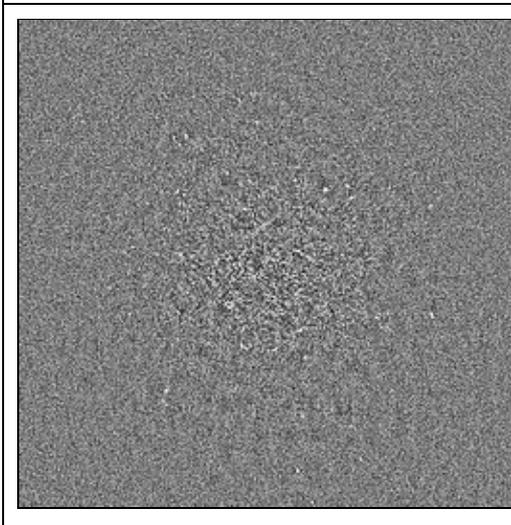
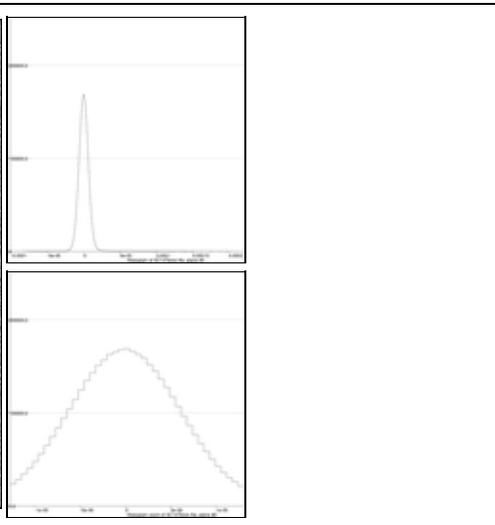
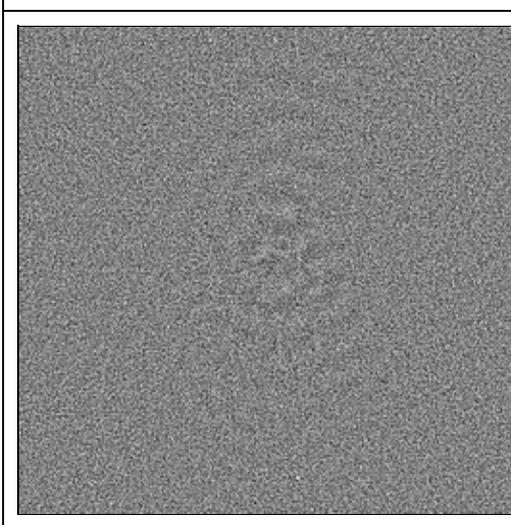
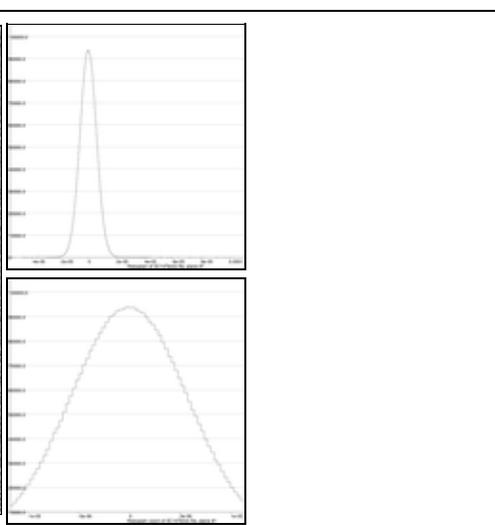
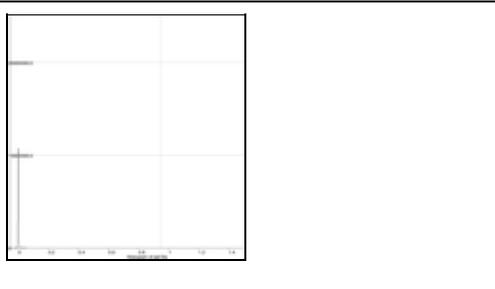


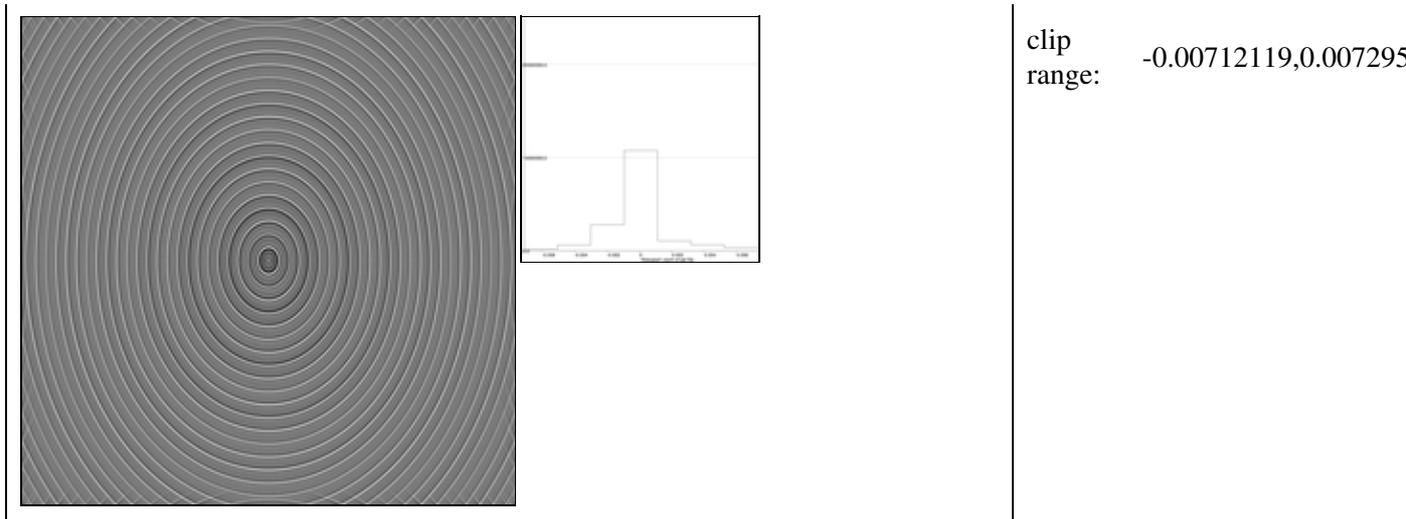
Fig9-7.png: Fig 9.7: Restore model into image dialog

[A773.tgz](#): A773 sky model

[3C147brick.fits](#) (header): FITS brick for 3C147 model

2048x2048x2x1 FITS cube, 2 planes are given below

		<p>Image plane #0.</p> <p>data range: -0.0001014,0.0002205</p> <p>mean: 1.16783e-07</p> <p>sigma: 6.81136e-06</p> <p>clipping: 95%</p> <p>clip range: -1.27315e-05,1.30515e-05</p>
		<p>Image plane #1.</p> <p>data range: -5.54082e-05,0.000110</p> <p>mean: 3.28845e-09</p> <p>sigma: 5.77172e-06</p> <p>clipping: 95%</p> <p>clip range: -1.1716e-05,1.12628e-05</p>
<p><u>psf.fits (header)</u>: PSF file for the "restore model" operation</p>		
		<p>data range: -0.0544909,1</p> <p>size: 4096x4096x1x1</p> <p>mean: -2.20103e-07</p> <p>sigma: 0.0030379</p> <p>clipping: 95%</p>



## 10. Scripting, data conversion, and Python API

Logged on 06/06/2012 03:29:58 PM

Advanced usage of Tigger involves using scripts and the Python API for manipulation of LSMs. A couple of scripts (`tigger-make-brick` and `tigger-restore`) were already mentioned in the previous section. A third script, `tigger-convert`, provides some more powerful model manipulation capabilities:

- \* conversion of Tigger-format models to and from external formats (currently ASCII tables, NEWSTAR and some BBS formats are supported)
- \* merging of models
- \* extraction of model subsets (based on their attributes)
- \* renaming of sources
- \* translation of model coordinates

A subset of these will be illustrated in this section; see "`tigger-convert --help`" for a full list of options.

Probably the first most useful capability of `tigger-convert` is to read a wide variety of whitespace-separated ASCII tables and convert them into Tigger LSMs. As an example, see the attached `cats-search.txt` file. This is an NVSS extract obtained via the SAO CATS catalogue query tool (<http://www.sao.ru/cats/> -- choose "Select in celestial area" -- "Radio" -- enable "NVSS" -- select RA from 0 to 1 hour, Dec from 0 to +20 degrees, flux

from 500 mJy up, press "Search", save the results file).

The catalog file in question contains one line per source, with a number of whitespace-separated columns. This is exactly the format expected by `tigger-convert` for ASCII files -- all that remains is to specify what column corresponds to what. The file may be converted to a model with the following command:

```
$ tigger-convert cats-search.txt cats.lsm.html --format "dum name ra_h ra_m ra_s dum dec_d dec_m dec_s dum dum i"
```

Here, the `--format` option specifies the order of the columns. The "dum" entries specify columns to be ignored; the other entries specify valid columns. Use `"tigger-convert --help"` to look up other valid column names, and note that `tigger-convert` will recognize the unit suffixes "\_d", "\_m", "\_s", "\_h", and "\_rad" for columns that deal with angles and angular extents.

Load the `cats.lsm.html` model into `tigger` to verify that it has been converted properly. Note that source names have been taken from the model file; use `--rename` to rename sources using the COPART conversion:

```
$ tigger-convert cats.lsm.html --rename -f
```

...and load it into `Tigger` again. Finally, here is how we could extract a subset of sources:

```
$ tigger-convert cats.lsm.html cats1.lsm.html --select I.gt.1
```

This selects sources with `"I>1"`, and writes them to the file `cats1.lsm.html`.

Finally, you may use the `Tigger` API to manipulate LSMs directly from Python. A full documentation of the API is beyond the scope of this User Guide, but the attached `tigscript.py` file provides a useful starting example. This contains the following statements:

```
# imports the Tigger API and loads a model

import Tigger

mod = Tigger.load("cats.lsm.html")

# mod.sources is a list of model sources. This makes a new list from this, of sources with I>1

sources = [ src for src in mod.sources if src.flux.I > 1 ];

# this loops over sources in the new list, and modifies their attributes

for src in sources:
```

```
. print src.name,src.pos.ra,src.pos.dec  
  
. src.flux.I *= 2  
  
. src.flux.Q = .1  
  
# this puts the new source list into the model object, and saves it under a new filename  
  
mod.setSources(sources)  
  
mod.save("cats2.lsm.html")
```

Now load the new cats2.lsm.html model into Tigger and observe the results.

As a very advanced use case of Tigger, consider the following real-life data reduction (this was applied to the A773 data set, an LSM for which we were looking at earlier).

- \* An initial LSM for the field is built up using NEWSTAR. This contains only point sources, which do not do a very good job of representing the slightly resolved sources in the field. The initial reduction is also contaminated by DDEs.

- \* The NEWSTAR model is converted into Tigger using tigger-convert, renamed according to COPART, and sources are clustered.

- \* "Bothersome" sources for which a differential gain solution is needed are assigned a "dE" tag. Even more bothersome slightly-resolved sources are assigned a "postage\_stamp" tag.

So far, these are preparation steps which are done manually, using the Tigger GUI and support scripts. Now, a fully automatic MeqTrees pipeline may be started, consisting of the following steps:

- \* a selfcal (G-Jones) solution is done on every MS (every night, every spectral window) of the observation

- \* a differential gain solution is performed on sources bearing a "dE" tag.

- \* for every spectral window, for every source bearing a "postage\_stamp" tag, the pipeline generates residual visibilities where the rest of the LSM (with the exception of that source) has been subtracted, and corrections for G and dE towards that source have been applied. A small but high-resolution image (the "postage stamp" per se) is then made (using all available data), centered on that source. The image is CLEANed. The resulting image of clean components is then inserted into the LSM as a FITS brick, in place of the source in question.

- \* this produces a "hybrid" LSM similar to the one we saw earlier in Section 8. This hybrid LSM is now used to re-do G and dE solutions.

\* the residual visibilities are imaged and CLEANed. The resulting image of (faint) clean components is inserted into the LSM.

\* a final round of calibration is done with the new LSM.

Note that every step of this pipeline is scripted and fully automatic, and makes use of the various unique Tigger features. Source tags play an absolutely crucial role: it is via tags that the user can designate some sources for the "special treatment" of dE solutions or postage stamps.

## Data products

<u>cats-search.txt</u> : CATS query, as an ASCII table
--

<u>figscript.py</u> : example script using the Tigger API
--

---

## Conclusion

Logged on 06/06/2012 04:20:45 PM

This concludes our Tigger User's Guide. I hope you find it useful. Further information may be obtained from the Tigger API Reference (under construction), the Tigger source code, or from me (Oleg Smirnov, oms AT ska DOT ac DOT za) directly.

This work has been supported by the European Community Framework Programme 7, Advanced Radio Astronomy in Europe, grant agreement no. 227290 (RadioNet FP7/ALBiUS).

---

 This log was generated by PURR version 1.1.