```
coordsystest()
```

Runs all tests.

Example

```
coordsys(4)
```

Runs only test 4.

1.2.14 imagefitter - Tool

Package general
Module images

Fit models to images of the sky
include "imagefitter.g"

Constructors

imagefitter	Construct an Imagefitter

Functions

componentlist	Recover the fitted components
done	Destroy this Imagefitter tool
gui	Start the GUI interface
nfits	Return number of accepted fits
regions	Recover fitted regions
setmaxpixels	Set maximum number of pixels to fit without query
summary	Summarize the image being fitted
type	What type is this tool ?

Synopsis
The Imagefitter `tool` offers interactive fitting of 2D models (or components) to an image of the sky. You control the process from a GUI interface. The `tool` is self contained. Thus, the display is built into the interface.
Current functionality includes

- The image is displayed with the Viewer providing full control over the display (zooming, colour tables, transfer function, labels etc) and easy saving to postscript of the display.

- The basic interface allows only Gaussian fits of single components with an automatic estimate. A more advanced interface allows simultaneous fits of multiple components with interactive or automatic estimates.
- Choice of models with the advanced interface.
- Model parameters can be held fixed in the fitting process with the advanced interface.
- Interactive specification of `regions-of-interest` (box or polygon) to be fitted
- Accepted fits are stored and returned Componentlist tools via function componentlist. The fitted components can be optionally deconvolved from the restoring beam in this step. The Componentlist can be used as a model in imager.
- The `regions-of-interest` used for the accepted fits are stored internally and can be recovered.
- Fit parameters are listed and the fits can be graphically overlaid on the display.
- Fit residuals are displayed via an image, histogram and statistics
- The fitted model can be subtracted/added to the image
- `Regions-of-interest` can be recovered from a Table and automatically refit.
- You can step through sky planes of a multidimensional image fitting as you go.

Restrictions which will be removed with time are currently

- Only Gaussian and Point (a Gaussian of the same shape as the restoring beam) models are available
- There is no joint Stokes fitting
- There is no automatic source finding

There will be some interface changes as these things are accomodated.

More Details
It is useful for you to know a little about how the Imagefitter manages the resultant fitted parameters. Each 'fit' that you make may contain multiple components. For example, you might fit three gaussians simultaneously. The Imagefitter stores the result of each accepted fit in its own Componentlist `tool`. In reality it maintains a list of Componentlist tools and you can

manipulate this list in a few ways. You can display (make graphical overlay on display) any or all of the Componentlist tools in the list (via the Show/Hide buttons on the main GUI). You may delete any or all of the Componentlist tools in the list (via the Delete button on the main GUI). You can recover all or any of the Componentlist tools in the list (via function componentlist).
Here is an example of the basic Imagefitter GUI.
Under the "File" menu at the top you find the following items.

- Save regions : when a fit is accepted, the region that was fit is stored as well. You can save those regions to a Table if you wish (this invokes the Saveregions widget).
- Restore regions and fit : this enables you to recover previously stored regions from a Table and automatically fit them (single Gaussian component only).
- Dismiss : dismiss the GUI (you can recover it with the gui function).
- Done : destroy the Imagefitter tool

The GUI shows a display of the image to the right. Beneath the image is a listbox showing the parameters of the model fits. To the left are a range of displays for the residuals of the fits. At the top is a histogram and some statistics. Below that is an image of the residuals in the fitted region.
There are two rollups beneath the data displays.
The first is labelled "Pixel Selection". With this widget, you can select which pixels, by intensity range, will be fit - it is generally good to exclude noise pixels. You can set an inclusion range, an exclusion range, or use an auto pixel selection. This last mode selects pixels depending on the context. For Stokes I, positive pixels are selected if the absolute peak in the fitting region is positive. It selects negative pixels if the peak is negative. For other Stokes parameters, all pixels are selected. You may wish to fine tune the pixel selection range so that only pixels above the noise level are selected.
Also in this rollup is an entry box labelled 'Maximum pixels'. If you ask to fit a large number of pixels (perhaps you forgot to set a region), the fit may go for a semi-infinite time. To protect against this, there is a built in maximum number of pixels allowed. If you ask to fit more pixels than this, you are queried whether you wish to proceed through a choice GUI. You can set this maximum pixels query limit to something different here (perhaps make it larger).
The second rollup is labelled "Fine Control". This is where you can activate the advanced interface. You select the number of simultaneous components you wish to fit (either type it in or move the slider) and then press the "Go" button.
Finally, a message line below the rollups provides you with information on what to do next. The Imagefitter constrains you as much as it can so that the only things you can do are valid.

The Basic Interface
If you don't activate the advanced interface, you get the basic interface. This means that

- Only one Gaussian component is fitted
- You cannot fix any parameters of the fit
- The intitial estimate is automatic

The fitting process is triggered by you generating a region (rectangle or polygon) and double clicking within it to signify it is ready. After the fit is done, the parameters are displayed in the list box, and the residual image, histogram and statistics are displayed. At this point, you can press the "Accept" button to accept and store the fit internally. Otherwise you can just do another fit.
At any point you can press the "Subtract" button. This subtracts the last fitted model from the data and the main image display is updated to reflect this. Once the "Subtract" button is pressed, it becomes an "Add" button, as there is not much point to you subtracting a model more than once, fun though it might be.
The fit parameters are displayed in the listbox under the main display. Here you may select the current fit or any of the previously accepted fits from the list. Its components (a fit may have multiple components in the advanced interface) will be listed. You may also show/hide graphically the selected fit on the display. You may also delete fits from the list if you decide you erroneously accepted a fit earlier.

The Advanced Interface
If you activate the advanced interface (via the "Fine Control" rollup and the "Go" button), then

- You can fit many simultaneous components
- In principle those components can be different types, but currently only types Gaussian and Point (a Gaussian with the same shape as the restoring beam) are implemented.
- You can fix some parameters of the fit
- The intitial estimate can be interactive or automatic

With the advanced interface, the order in which you do things is not as linear as with the basic interface. However, the idea is that you first establish an estimate for each component, and then do the fit.
Here is an example of the advanced Imagefitter GUI with three components available.
You select a particular component by clicking on the numbered (TAB) button at the top of the GUI (this is a pseudo-tab widget interface). This brings to

the front a GUI for this particular component. You then choose the type of component you wish to fit (optionmenu in the top left). Then you establish an estimate for each component and then fit them simultaneously in a specified region.

At the bottom of the advanced GUI, there is a Region widget. This houses any region you create interactively with the main display (via the rectangle or polygon region maker). This region is used in two ways. Firstly, this is the region that is used when you press the "Fit" button. Secondly, when you make an automatic estimate for a particular component, the pixels are only selected from this region. Thirdly, when you make an automatic estimate for all components simultaneously, the region must embrace all of the sources of interest (it would be the same region as when doing the actual fit). Thus the region has a context dependent use - one component specific (auto individual component estimate) and one global (the fit and auto estimate for all components). If you have not explicitly set a region, it defaults to the unzoomed full region currently being displayed.

To aid you in using this Region widget, there is a check button (labelled "Inserted") which is set on for half a second when a region is captured from the main display. There is also a button labelled "Show". This lists the bounding box of the region and redisplay the region on the image.

Now the main thing you must do is establish an estimate for that component. You can do this in a variety of ways. Firstly, you can enter manually the values for each parameter. Secondly, you can press the "Auto" button and an automatic estimate will be made for you. Thirdly, you can press the "Inter" button and use the displayed overlay to establish an interactive estimate.

It is also possible to make automatic estimates for *all* of the components (rather than one by one). You do this by establishing a region that embraces all of the components. Then press the "Est" button at the bottom and it will make an estimate for each component requested. It won't work terribly well if the components are heavily overlapping.

Once you are happy with the estimates for each component, you press the "Fit" button. You can then click the "Accept" button on the main GUI display (as in the Basic interface) if you want to keep the fit, or you can just do another one.

The buttons labelled "Fixed" for each parameter enable you to fix that parameter at its estimated value during the fit. This must be an active value to be useful. To aid in setting an estimated value which you wish to fix, the left-pointing arrows transfer the values from the fit entry boxes to the estimate boxes.

The "Pixel Selection" rollup is active for the Advanced as well as the Basic interface.

Finally please note that for your convenience, the flux density is presented to you by the Imagefitter GUI interfaces as a peak flux density. Inside the Componentlist, the flux density is stored in an Integral representation. Therefore conversions back and forth go on behind the scenes.

Events
The Imagefitter emits one event called 'accept'. Whenever the user presses the 'Accept' button to accept a fit, this event is emitted. The value of the event is the fit number.

imagefitter - Constructor

Package	**general**
Module	**images**
Tool	**imagefitter**

Construct an Imagefitter

Synopsis
imagefitter(infile, region, parent, residual, maxpix, widgetset)

Arguments

infile	in	The image file name or image tool Allowed:The file name or an Image tool
region	in	The region of interest to display Allowed:Region tool Default: Whole image
parent	in	Parent frame to embed GUI in Allowed:Tk frame agent Default: F
residual	in	Residual image file name Allowed:String Default: Temporary image
maxpix	in	Maximum number of pixels to fit without query Allowed:Integer Default: 256*256
widgetset	in	The widgetset to use for the GUI Allowed:A widgetserver tool Default: ddlws

Returns
imagefitter tool

Description

This is the constructor. You must specify the `image file` either by its name or by giving an `image tool`. You may optionally embed the display in an existing Tk frame (`parent`) if you wish. This constructor does a fair amount of work, so be a little patient if need be.
The input image is copied (and any specified region applied); this copied image is used should you choose to subtract any model fits (via the GUI). If you specify the argument `residual`, then the copied image is saved in a file of this name. If you leave `residual` unspecified, a temporary image will be created and used for model subtractions, but it will be destroyed when you terminate the Imagefitter.
See the description at the top of this document to find details on operating the Imagefitter interface.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.done()
- im := image('ngc1313.cube')
- myfitter := imagefitter(infile=im, residual='ngc1313.resid')
```

These examples show the two ways to use the `infile` argument. In the first case we indicate that should we subtract models from the image, that they should be subtracted from the `image file` called *ngc1313.cube*. In the second case we indicate that should we subtract models from the image, that the input image is not changed, and the residuals are written to the `image file` called *ngc1313.resid*.

---

imagefitter.componentlist - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Recover the fitted components

Synopsis
componentlist(which, concatenate, deconvolve)

Arguments

which	in	Which fit ? Allowed:Unset or integer Default: All
concatenate	in	Concatenate all Componentlists into one ? Allowed:T or F Default: F
deconvolve	in	Deconvolve components from restoring beam Allowed:T or F Default: F

Returns

Record of Componentlists, Componentlist or fail

Description

The Imagefitter stores the fitted Sky components in an internal record. For each accepted fit, there is one entry in the record (indexed 1, 2, 3 ...) and each entry holds a Componentlist `tool`; there may be any number of components in each Componentlist (depends upon how many components you fitted). If there are no accepted fits, it returns an empty record and issues a warning.
By default, this function returns the record holding all of the separate Componentlists. If you do specify `which` (the integer index of the accepted fit) then it returns the Componentlist `tool` for that index directly.
If you leave `which` unset (its default), but you specify `concatenate=T`, then all the Componentlists are concatenated into one, and that concatenated Componentlist `tool` is returned directly. The `concatenate` argument is meaningless if you do specify a particular index with `which` (see above).
You may also optionally deconvolve the components from the restoring beam of the image if there is one. The default is not to do this.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- list := myfitter.componentlist()                     # Recovers record of Componentlist tools
- print list[1].length()                               # Length (number components) of
3                                                      # first Componentlist
#
- cl := myfitter.componentlist(which=3)                # Recovers specific Componentlist
- print cl.length()
2
-
- cld := myfitter.componentlist(which=2, deconvolve=T)    # FIt 2 and deconvolve from beam
```

imagefitter.done - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Destroy this Imagefitter tool

Synopsis
done()

Returns
Bool or fail

Description
This function destroys the Imagefitter `tool`.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.done()
- myfitter
F
```

---

imagefitter.gui - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Start the GUI interface

Synopsis
gui()

Returns
Bool or fail

Description
If you dismiss the GUI (with the dismiss button), then you can recover it with this function. The state of the Imagefitter and the GUI is preserved. If the GUI is already present, then it does nothing.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.gui()
```

---

imagefitter.done - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Destroy this Imagefitter tool

Synopsis
done()

Returns
Bool or fail

Description
This function destroys the Imagefitter `tool`.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.done()
- myfitter
F
```

---

imagefitter.nfits - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Return number of accepted fits

Synopsis
nfits()

Returns
Integer

Description

This function retuns the number of fits that the user has accepted.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- print 'Number of fits = ', myfitter.nfits()
```

---

imagefitter.regions - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Recover fitted regions

Synopsis
regions(which)

Arguments

which	in	Which fit ? Allowed:Unset or integer Default: All

Returns
Record of regions, region, or fail

Description

As well as the internal fitlist (a record of Componentlist `tools`), there is also a record of regions, corresponding to the accepted fits that is stored internally. This function recovers those regions. By default, you get them all in an indexed record. You can specify a specific one if you wish, and then the function returns the region directly (rather than putting it in a record).

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- r := myfitter.regions()
- print 'number of fitted regions=', length(r)
- im := image('ngc1313.cube')
- im.view(region=r[1])
```

Here the region of the image that was fitted for the first component is displayed.

---

imagefitter.setmaxpixels - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Set maximum number of pixels to fit without query

Synopsis
setmaxpixels(maxpix)

Arguments

maxpix	in	Number of pixels

Returns
T

Description

If you ask to fit a large number of pixels (perhaps you forgot to set a region), the fit may go for a semi-infinite time. To protect against this, there is a built in maximum number of pixels allowed. If you ask to fit more pixels than this, you are queried whether you wish to proceed through a choice GUI.
This function lets you set the maximum pixels value.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube', maxpixels=512*512)
- myfitter.setmaxpixels(maxpixles=128*128)                          # My computer is slower than I thought
```

---

imagefitter.summary - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

Summarize the image being fitted

Synopsis
summary()

Returns
T or fail

Description
This function summarizes, to the logger, the image with which the fitter was constructed. It just invokes the Image `tool` function summary.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.summary()
```

---

imagefitter.type - Function

Package	**general**
Module	**images**
Tool	**imagefitter**

What type is this tool ?

Synopsis
type()

Returns
String or fail

Description
Returns the string “imagefitter”.

Example

```
- myfitter := imagefitter(infile='ngc1313.cube')
- myfitter.type()
imagefitter
```

1.2.15 imagefittertest - Function

Package general
Module images

Run test suite for Imagefitter tool

Synopsis
imagefittertest(which)

Arguments

which	in	Which test to run Allowed:Integer or vector of integers Default: All tests

Returns
T or F