IRSCLEAN User's Guide

Spitzer Heritage Archive Documentation

James Ingalls

Version 2.1, January 2011

Table of Contents

1 What is IRSCLEAN?

IRSCLEAN is an interactive IDL tool for creating bad pixel masks from Spitzer IRS BCD (and pre-BCD) image data, and "cleaning" the masked pixels in a set of data. The interactive version of the program, IRSCLEAN_MASK, is an outgrowth of the non-interactive IRSCLEAN package developed by members of the SSC IRS instrument support team and the Cornell IRS Instrument team. IRSCLEAN_MASK takes care of file input/output and queries you when it needs information.

1.1 Should I Always Run IRSCLEAN On My Data?

If your source is bright enough that the rogue pixels do not make a significant contribution to the flux, or if you were lucky enough to get a good match to the rogue pixels in background subtraction, you may choose not to clean any pixels. This is of course the best option, since "cleaning" a pixel in effect throws away the data for that pixel. The tradeoff should always be improvement in the extracted spectrum.

You should always check the match between a given rogue mask and your data. In particular the pixels in the rogue mask should be significantly brighter than their surroundings (NOTE: for background-subtracted data, rogue pixels can also be much fainter than surrounding pixels). Sometimes this is obvious from visual inspection. You can use irsclean_mask to overlay a rogue mask on your data, examine the brightness of pixels, and add or remove pixels from the mask.

1.2 What is This Document?

This document is a complete description of how to use IRSCLEAN. It tells how to install the package, explains the various kinds of data and masks that you will use, and runs through the various stages of the program. In the Appendices, we describe some extra functionality. Throughout the document you will see examples of IDL commands. Once IRSCLEAN is installed you will be able to run most of these commands using the sample data that come with the package.

1.3 What's New In IRSCLEAN

Previous Version: 2.0
Current Version: 2.1

Changes to IRSCLEAN

- (Bug Fix) Error message "`READCOL: Illegal format A10 in field 0`" when current version of IDLASTRO `readcol.pro` is used.
- (Bug Fix) `/HELP` keyword now triggers IDL internal `doc_library` command, instead of missing `print_header.pro`.
- (Incompatibility fix) Error message "`Colorbar::Draw Method: Attempt to call undefined method: 'COLORBAR::POSITION'`" under IDL 8.0

Changes to documentation:

- Updated "2.0" to "2.1".
- Documented all new features.

1.4 System requirements

In order to run IRSCLEAN you need the following:

1. IDL version 6.2 or later, which is free to download. **Note:** IDL 8.0 users should ensure that they are running the patched 8.0.1 version. (http://www.ittvis.com/UserCommunity/UserForums/tabid/58/forumid/27/threadid/10962/scope/posts/Default.aspx).
2. The most up to date version of the IDL Astronomy User's Library, which can be downloaded from http://idlastro.gsfc.nasa.gov/
3. A 3-button mouse, or its equivalent.

2 Installing IRSCLEAN

2.1 Downloading

The IRSCLEAN distribution can be downloaded as a .tgz (tar-gzip) file from

http://irsa.ipac.caltech.edu/data/SPITZER/docs/dataanalysistools/tools/irsclean/downloadirsclean/

2.2 Unpacking

Unpack using **`tar -xz`** or another appropriate software into a directory that is either in your IDL path, or in a directory that will be added to your IDL path (see below). The distribution will create its own directory (eg., **`IRSCLEAN2.1/`**). For example, to unpack in **`/home/myself/idl/IRSCLEAN2.1`** (assuming the directory **`/home/myself/idl`** exists):

```
unix% tar -xvzf irsclean2.1.tgz -C /home/myself/idl/
```

Note: Campaign Rogue Pixel Masks The campaign rogue pixel masks are automatically packaged with irsclean, so you no longer have to install the files yourself. The files are located in a subdirectory **`all_campaigns_roguemasks/`** under the **`IRSCLEAN2.1/`** distribution, and will be picked up automatically when keyword **`/CampaignRMask`** is invoked. The appropriate campaign mask is also automatically pre-loaded as the default input rogue mask file (**`inRmaskFile`**), if no file is given at the command line.

2.3 Setting the IDL Path

For IDL to be able to automatically compile and run the IRSCLEAN distribution when the routines are called, either your current IDL working directory has to be the directory where IRSCLEAN resides, or the distribution must be in the IDL path. If IRSCLEAN is placed in a directory which is already in the IDL path, then make sure the path entry is set such that subdirectories are searched, via the "+" symbol.

To add the exact directory where IRSCLEAN resides to the IDL path, you have three options:

2.3.1 Outside of IDL, Unix-type Computers

Add the directory where IRSCLEAN resides to the $IDL_PATH environment variable in your Unix startup file.

For example (csh or tcsh):

```
unix% setenv IDL_PATH ${IDL_PATH}:"+/home/me/idl/IRSCLEAN2.1"
```

The "**+**" at the beginning of the path means "search subdirectories".

2.3.2 Inside of IDL

Use the command **`PREF_SET`** in IDL.

Under Unix:

```
IDL> PREF_SET, 'IDL_PATH', '<IDL_DEFAULT>:+/home/me/idl/IRSCLEAN2.1',$
/COMMIT
```

Under Windows:

```
IDL> PREF_SET, 'IDL_PATH', '<IDL_DEFAULT>;+C:\home\me\idl\IRSCLEAN2.1'
IDL> PREF_COMMIT
```

Note the use of the colon (";") as a path separator for Windows, and the separate call to PREF_COMMIT.

You can find out the current value of IDL_PATH by typing

```
IDL> PRINT, PREF_GET('IDL_PATH')
```

2.3.3 All IDL Versions, all Operating Systems

Use the Preferences Dialog in IDLDE.

The IDL Workbench (was the IDL Development Environment) (command line: **`idlde`**) graphical user interface can be used to set permanent IDL preferences. Choose **`File->Preferences->PATHS`** and add an entry to the path similar to one of the strings above. Instead of the plus sign, a check box is used to search all subdirectories.

2.4 Mac OS X: Enabling Mouse Cursor Input

By default, "click through" events are not enabled in Mac OS X. As a result, interaction between the mouse cursor and the IDL graphics window (the heart of the mask editing process) will not occur. To fix this, type the following in an X11 command line terminal:

2.4.1 In OS X 10.4 (Tiger) and prior:

```
unix% defaults write com.apple.x11 wm_click_through -bool true
```

Then close X11 and restart.

If for any reason you need to undo this, simply type

```
unix% defaults delete com.apple.x11 wm_click_through
```

Then close X11 and restart. Your original default will be restored.

2.4.2 In OS X 10.5 (Leopard) and later:

After OS X Leopard, use the command:

```
unix%  defaults write org.apple.x11 wm_click_through -bool true
```

to invoke click through events, and

```
unix%  defaults delete org.apple.x11 wm_click_through
```

to turn them off. (The difference between Tiger and Leopard is the string "**com**" in Tiger and "**org**" in Leopard.)

2.4.3 Emulate a three-button mouse

Another problem that Mac users may encounter is the single-button mouse. Fortunately, X11 for the Mac allows you to emulate a three-button mouse. Under X11 choose **X11->Preferences->Input->Emulate three button mouse**.

3 The Sample Files

The IRSCLEAN distribution comes with a set of sample IRS data files, located in the subdirectory **sampledata/**, that can be used to test the program. The sample files are publicly available through the Spitzer archive and are observations of the calibration star HR 6348. There are files for both the Long Low (**sampledata/r13349376/ch2/bcd/**) and Long High (**sampledata/r13349376/ch3/bcd/**) modules. In this guide, most examples will use the sample file **sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits**. Feel free to try out IRSCLEAN using any of the **.fits** files.

We have included the trio of files usable during spectral extraction in SPICE, **bcd.fits**, **bmask.fits**, and **func.fits**. You can use the **bcd.fits** file in **Stage 1** (mask creation), and the **bmask** and **func** (uncertainty) files during cleaning. The sample files also include a file which is itself a list of bcd FITS files, **sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_clean_batch_list.txt**. As will be seen, such **.txt** files can be used in **Stage 2** to prescribe a set of data files to be cleaned.

4 Masks Used or Created by IRSCLEAN

IRSCLEAN is all about making and using masks. There are three main kinds of masks that you will deal with in running IRSCLEAN:

4.1 RMASK: The Rogue Pixel Mask

Bad or "rogue" pixel masks used and created by IRSCLEAN are binary images with bad pixels indicated by the value 1 and normal pixels indicated by 0. These masks are for use within the IRSCLEAN distribution and are not interchangeable with the masks created by the IRS pipeline.

When we refer to an RMASK, we usually are talking about a previously-derived mask, stored as a fits file, that you will either edit or use to clean data (as opposed to a "found" or "user-defined" mask; see below).

The most problematic kind of IRS bad pixels are the rogue pixels, which have elevated dark currents that change unpredictably with time. See Chapter 7 of the IRS Instrument Handbook (http://irsa.ipac.caltech.edu/data/SPITZER/docs/irs/irsinstrumenthandbook/) for more on rogue pixels.

One set of pre-derived campaign-based rogue masks are available on the Rogue Pixel Masks website. These masks are automatically installed with the IRSCLEAN distribution, under **`all_campaigns_roguemasks/`**. In this subdirectory there will be one mask per IRS module and campaign number. The file **`all_campaigns_roguemasks/campaigns.txt`** relates the campaign number (eg. **`IRS1`**) with the SSC campaign ID (eg. **`IRSX002500`**) that is stored in the fits header of your data files.

4.2 FMASK: The "Found" Mask

IRSCLEAN has a built-in algorithm for analyzing your input image and finding rogue pixels automatically. If you run IRSCLEAN's rogue-finding algorithm (using **`/getFmask`**) you are then allowed to edit the FMASK in Mask-Edit mode and save the mask to a FITS file for use later as an RMASK.

4.3 UMASK: The User-Defined Mask

You can specify an external function (using maskFunction) that will independently analyze your input image and find rogue pixels based on your own algorithm. This mask can also be edited in Mask-Edit mode and saved to a FITS file for use later as an RMASK.

4.4 BMASK: The Pipeline BCD Mask

The final products of the IRS Basic Calibrated Data pipeline include the processed observation, **`bcd.fits`**, the corresponding uncertainty plane, **`func.fits`**, and the BCD Mask, **`bmask.fits`**. The BMASK file is a 16-bit integer mask with bits set based on the status of BCD image pixels (listed in Table 4.1; for more information, see the IRS Instrument Handbook).

Table 4.1: Pipeline bmask bits

Bit #	Condition

0	Cleaned with IRSCLEAN
1	Latent-image flag
2	Digital saturation detected along the ramp (bit 3 in dmask).
3	RADHIT detection along ramp (bit 9 in dmask).
4	Non-linearity correction could not be computed (bit 12 in dmask).
5	Not used
6	Droop removed using questionable value (bit 6 in dmask).
7	Flat-field applied using questionable value (FLATAP)
8	Flat-field could not be applied (FLATAP)
9	Stray-light removal or cross-talk correction not applied
10	At least one sample exceeds 280000 e−. (bit 13 in dmask).
11	Data missing in downlink (bit 14 in dmask).
12	Only one usable plane
13	No usable planes
14	Pixel masked in pmask
15	Reserved: sign bit

The BMASK file determines which pixels to ignore during IRSCLEAN processing. Pixels that are "fatally" masked are colored blue during rogue mask editing. The default fatal value is decimal 28800 (bits 7 and 12-14). This can be adjusted using either the keyword **`bMaskVal`** or **`bMaskBits`**. These keywords allow you to enter fatal BMASK bits as either a single 16-bit number expressing the sum of the active bits (eg., **`bMaskVal=28800`**), or as a vector of the bits to be set to fatal (eg., **`bMaskBits=[7,12,13,14]`**). A fatally masked pixel can be toggled into a rogue pixel and cleaned (*not recommended*), in which case, IRSCLEAN will automatically update the BMASK for use in spectrum extraction with SPICE.

NOTE: Earlier versions of IRSCLEAN allowed the use of the BMASK as a bad pixel mask, but this led to unstable results. The current approach is to ignore fatally masked pixels.

NOTE: If a pixel is cleaned using IRSCLEAN, bit 0 of the pixel's BMASK is turned on.

4.5 OMASK: The Order Mask

Each IRS detector array images a set of spectral orders. The mask that specifies which pixels are in which order is given by the order mask, a file produced by the SSC, and delivered with the IRSCLEAN distribution in the subdirectory **`cal/.`** Each pixel in the order mask file has a number specifying which order it belongs to. Order mask pixels that are not in an order have the value 0. See Appendix 2 for more information on the locations of the IRS spectral orders. The order mask defines which pixels are cleaned by IRSCLEAN. The default order mask is the "narrow" order mask. You can select a "wide" order mask by setting the keyword **`/wide_omask`** at the IRSCLEAN command line. You can also specify no order mask by setting **`/noOrderMask`**. This allows you to clean any pixel on the array. Finally, you can define your own order mask with an external file using the keyword **`orderMask_File=.`**

4.6 UNCLEAN_MASK: The Mask of Unsuccsessfully Cleaned Pixels

Not all pixels are always successfully cleaned by IRSCLEAN. This is usually due to one of three reasons:

1. The pixel does not need to be cleaned, i.e., its "cleaned" value is nearly equal to the original value.
2. There are not enough "good" pixels (pixels not in the bad pixel mask) in surrounding rows to be used to build the row profile.
3. The average row profile is too noisy.

When one or more pixels are not cleaned, IRSCLEAN will produce a special **`unclean_mask.fits`** file for each processed data file (as of version 1.9). Similar to the rogue mask, the "unclean mask" is an image of 1's and 0's indicating whether or not a pixel has been successfully cleaned.

See the hint on "a second pass at cleaning data using the unclean_mask file" in the Appendix.

5 How to Run IRSCLEAN

Now that you have installed the IRSCLEAN distribution and have configured your system, you are ready to start using it to make rogue pixel masks.

5.1 General Syntax

The full calling syntax of IRSCLEAN_MASK is

```
IDL> irsclean_mask, dataFile, inRmask_File=inRmask_File,$
           [/getFmask or getFmask=1 or 2], AGGRESSIVE=aggressive,$
            noise_Floor=noise_Floor, rogue_Thresh=rogue_Thresh,
            /ABS_VAL, /NEGATIVE_ONLY, /NAN, /CampaignRMask, $
            maskFunction=maskFunction, Bmask_File=Bmask_File, bMaskVal=bMaskVal, $
            bMaskBits=bMaskBits,$
            orderMask_File=orderMask_File, /wide_OMask, /noOrderMask,$
            ORDERS=ORDERS, /peakUp,$
            outRmask_File=outRmask_File, outClean_File=outClean_File,$
            /autoSave, filesToClean=filesToClean,
            dataRange=dataRange,winSize=winSize,$
            Directory=Directory,dataRange=dataRange,$
            /noStage2, /noMaskEdit,$
            colorNames=colorNames, /HELP, _EXTRA=extra
```

NOTE: IDL is **not** case-sensitive. In this document we use mixed case for IDL keywords and variables solely for readability. Feel free to use all upper-case, all lower-case, or whatever case combination you like.

All inputs to IRSCLEAN_MASK are optional. The simplest mode of running the program is to type **`irsclean_mask`** at the IDL command line without any inputs:

```
IDL> irsclean_mask
```

If no arguments are given, the program will walk you through the required steps. (NOTE: This is the only way of running in Virtual Machine mode.)

The bare-bones mask editing and data cleaning procedure consists of

Stage 1: Creating and Editing Masks

1. reading in data (`dataFile`),
2. reading a rogue mask if desired (`inRmask_File`),
3. editing the mask (skip if `/noMaskEdit` is set),
4. saving the result (`outRmask_File, /autoSave`),
5. saving the cleaned version of `dataFile` (`outClean_File, /autoSave`).

Stage 2: Cleaning a Set of Data
(skip if `/noStage2` is set)

1. picking a set of `.fits` files to clean and/or one or more list (`.txt`) files (`filesToClean`),
2. reviewing and editing the list of files to clean,
3. cleaning the data (output files are automatically saved).

Relevant keywords that control a given step are given in `computer type` above. If any one of the keyword arguments is missing, you will be queried by the program. In Virtual Machine mode, all required inputs are requested.

There is also the option to find rogue pixels automatically (`getFmask`), in which case the program does not by default ask for a rogue mask (you can still set the `inRmask_File` keyword; the program will read in a rogue mask and combine it with the FMASK).

For all filename arguments (`dataFile, inRmask_File, Bmask_File, orderMask_File, outRmask_File, outClean_File, filesToClean`), if you don't want to type out the file name but want to make sure that IRSCLEAN asks for a file (or set of files in the case of `filesToClean`), simply enter a null value for the filename and the program will ask you to browse your computer for the file(s). For parameter `dataFile`, enter the null string `''` (two single quotes with no space in between). For keywords, use the syntax `inRmask_File=''`. In most cases, however, this will not be necessary, since the default is for IRSCLEAN to ask for a file if missing. For example

```
IDL> irsclean_mask,'',inRmask_File=''
```

is the same as typing

```
IDL> irsclean_mask
```

If the `getFmask` option is used, however, the default is not to ask for an external rogue mask file. To force IRSCLEAN to request an external rogue mask file, the `inRmask_File=’’` syntax is required.

5.2 Running the Virtual Machine Version of IRSCLEAN

For those without an IDL license, the virtual machine (VM) version of IRSCLEAN can be run from the Unix command line. First, IDL must be downloaded from http://www.ittvis.com/ProductServices/IDL.aspx. Running in VM mode is akin to typing “`irsclean_mask`” at the command line and letting the program query you for all inputs. The syntax is as follows

```
unix% idl -vm=[path to IRSCLEAN]/irsclean_mask.sav
```

Note that there are a few features of IRSCLEAN that will not be accessible from VM mode. The most important of these is the ability for IRSCLEAN to find a mask automatically (**`getFmask, Rogue_Thresh, Noise_Thresh, AGGRESSIVE, /ABS_VAL, /NEGATIVE_ONLY, ORDERS, /peakUp).`** The following keywords will not be changeable from their default values: **`Bmask_File, bMaskVal, bMaskBits`**, and **`orderMask_File`**. In addition, the following keywords represent functionality that is not accessible via the virtual machine: **`/NAN, maskFunction, /wide_OMask, /noOrderMask, winSize, Directory, colorNames, /HELP`**. All other keywords have values that will be set interactively at runtime. (Currently many NaN pixels are already fatally masked by the BMASK and those that are not will be cleaned during spectral extraction with SPICE, so there is no longer more than cosmetic benefit to the **`/NAN`** keyword).

6 Stage 1: Creating And Editing Masks

6.1 Reading in Data

```
IDL> irsclean_mask,$
          'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
           DataRange=[13,53]
```

The first step in an IRSCLEAN session is to read in some FITS data. If you do not enter a value for **`dataFile`** (the first argument) then you will be asked to browse your directories and pick a file. If you hit "Cancel" you will end your session.

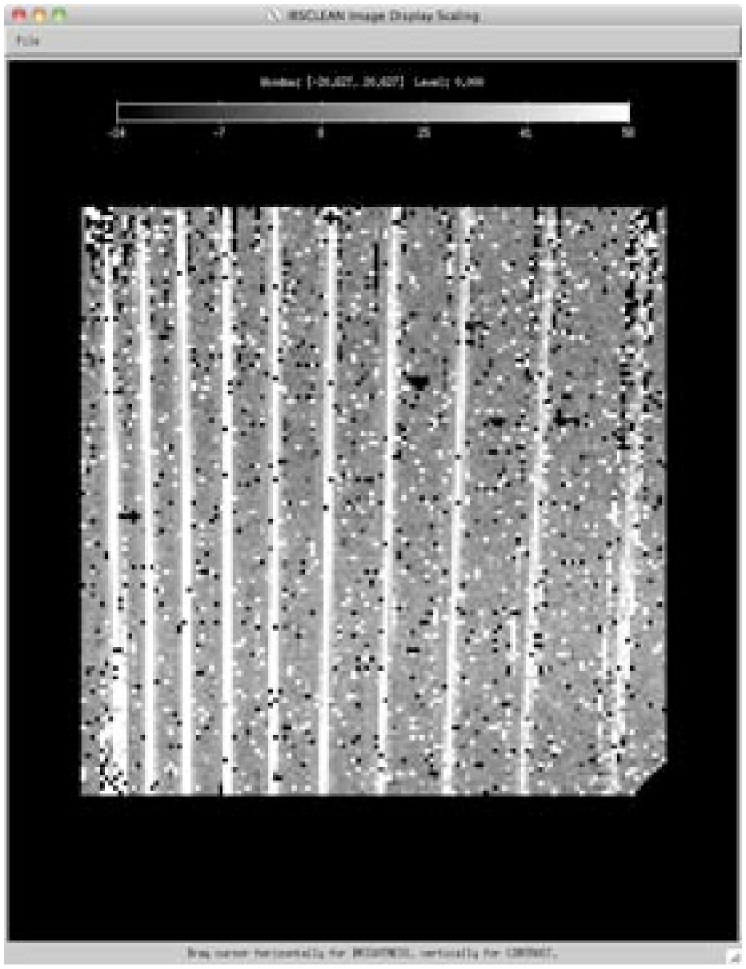


Figure 6.1 The image scaling widget.

6.2 Scaling Image Display

The keyword **`DataRange`** sets the image scaling when displaying during mask-editing mode. (If you leave out the keyword, the default scaling is based on the histogram of image values, which usually looks fine.) **`DataRange`** should be either set to actual known values, or **`[0,0],`** which will activate the manual display scaling widget (Figure 6.1)[1]. (This pops up just prior to mask editing.) Simply drag the cursor across the image with the left mouse button depressed. Dragging up will increase the contrast (decrease the spread of displayed brightnesses) and dragging to the right will increase the midpoint of the displayed brightnesses. When you're satisfied with the image scaling, click on **`File->Return to IRSCLEAN`** to close the window.

The next step is to obtain or build a mask of bad pixels. The input image is an aid to help you either judge the validity of a preexisting mask or build a new mask, as well as a guide to editing the mask.

[1] The image display scaling widget is based on **`windowimage.pro`** by David Fanning.

6.3 Reading a Preexisting Mask (**`inRmask_File, /CampaignRmask`**)

One way to use the program is to take a preexisting rogue mask and edit it. If you run **`irsclean_mask`** without either inputting a rogue mask or asking the program to find its own mask (using **`/getFmask`**), you will be asked to browse your computer for a rogue mask. IRSCLEAN will automatically choose the appropriate campaign-based rogue mask and enter it in a file picker dialog, asking you to pick a rogue mask to edit. You can select that mask, or use the dialog to browse your computer to another location and pick another mask. Alternately, you can forego having to select the campaign mask by entering **`/CampaignRmask`** at the command line. This will automatically load the campaign mask.

You can force the program to use a specific rogue mask by adding (eg) **`inRmask_File='rmask.fits'`** (to specify a filename) or **`inRmask_File=''`** (to be queried for a filename) to the calling sequence.

```
IDL> irsclean_mask,
          'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
           inRmask_file='all_campaigns_roguemasks/b3_rmask_IRS18.fits',$
           DataRange=[13,53]
```

Combining Rogue Masks: You can input or select more than one rogue mask by either entering a vector of file names at the command line (eg., **`inRmask_File=['mask1.fits','mask2.fits']`**) or selecting multiple files in the file picker dialogue. (On most platforms, multiple files can be selected using the Control key along with the left mouse button. They are, however, limited to being in the same directory when chosen in this way.) You can also select the campaign rogue mask (**`/CampaignRmask`**) as well as one or more rogue masks (**`inRmask_File`**). Masks will be combined using logical OR.

The rogue dialogue will always pop up unless you have set **`/getFmask, /CampaignRmask`**, or **`inRmask_File`** equal to a nonblank string. In any case, keep in mind that **you can always hit "Cancel" under the Rogue Mask dialogue and choose not to use an externally-derived rogue mask.** (See Creating a Mask Manually From Scratch below.)

If you input a file that doesn't exist, IRSCLEAN will ask you to choose a different file.

Be careful! IRSCLEAN does not care what file you input as a rogue mask. If you get strange results, then check the text output of your session. You will get a WARNING message if your mask file is either not made up of integers, or from the wrong (or unknown) IRS module, but IRSCLEAN will still try to process the data.

6.4 Creating a Mask Manually from Scratch

You do not have to use a preexisting mask. In fact you can create a mask from scratch interactively using the mouse. Leave out **`inRmask`** from the calling sequence and hit "Cancel" when asked to choose a rogue mask file. If no other mask is given or derived, you will enter Mask-Edit mode with the image of your data having no rogue mask overlayed. See Mask-Edit Mode below to learn how to build the mask.

```
IDL> irsclean_mask,$
           'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
            DataRange=[13,53]
```

6.5 Letting IRSCLEAN Find a Mask Automatically (**`getFmask`**)

IRSCLEAN can also find a mask for you if you set **`getFmask`**. This runs the WCLEAN algorithm on your data and produces a mask "on-the-fly" for you to edit. If **`getFmask`** is set, the default is not to require an external rogue mask too, unless **`inRmask`** or **`/CampaignRmask`** is also set.

The code for deriving rogue masks from an image is given in the IDL procedure **`wclean_multires.pro`**, which is part of the IRSCLEAN distribution. This code attempts to detect rogue pixels as extremely bright lone pixels or clusters of pixels in a small-scale component of the image, using one of two approaches, invoked by setting either **`getFmask=1`** (or, equivalently, **`/getFmask`**); or **`getFmask=2.`**

6.5.1 getFmask=1: the Hierarchical Iterative Method **`(/getFmask, AGGRESSIVE,/ABS_VAL, /NEGATIVE_ONLY,ORDERS,/peakUp`***)*

Prior to IRSCLEAN 2.0 this was the only built-in method of finding rogue pixels. It starts from the multi-median transform[2] of the input image, decomposing it into a sum of images representing successively larger resolution scales (kernel diameter 3, 5, 9, 17, and 33 pixels, plus a smooth remainder. The algorithm measures the noise level in either the interorder region of the image or, if the **`ORDERS`** keyword is set, within each order separately. The program then iteratively attempts to find bright pixels in the first component of the multi-median transform, as compared with neighboring pixels in the first and the second multi-median components. Depending on the value of the **`AGGRESSIVE`** keyword (see below), certain constraints are imposed on whether or not a multi-median transform feature is considered a rogue pixel or a member of a group of rogue pixels. If you find that the default setting finds too many or too few rogue pixels, use the keyword **`AGGRESSIVE`** to adjust the constraints on the rogue-finding algorithm (default is **`AGGRESSIVE=0`**).

```
IDL> irsclean_mask, $
     'sampledata/ r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
      DataRange=[13,53],/getFmask,AGGRESSIVE=0.1
```

The default value of **`AGGRESSIVE`** is 0. Lower values will yield fewer pixels identified as rogue, higher values will yield more. You can set **`AGGRESSIVE`** to any negative or positive floating point value, but most datasets only show an effect for values between about -2 and +2. The keyword controls three things:

1. The number of iterations of the rogue-finding procedure. The first iteration is performed on the input data. Subsequent iterations are performed on an image which has been cleaned of the previous iteration's rogues. If **`AGGRESSIVE`** is less than or equal to 0, there will be only one iteration. As **`AGGRESSIVE`** increases above zero, the number of iterations increases by 9 for every unit increase in **`AGGRESSIVE`**.

2. The sigma threshold above which a pixel is accepted as a candidate rogue. The value of sigma is determined in the first multi-median component of the original image (smoothing kernel 3 pixels). For **`AGGRESSIVE=0`**, a pixel is a rogue candidate if its value in this transform component is greater than 4 times sigma. The sigma threshold decreases by a factor of 2 per unit increase in **`AGGRESSIVE`**, reaching 0 at **`AGGRESSIVE=2`**. Thus, for **`AGGRESSIVE`** greater than or equal to 2, there is no sigma constraint on a pixel being considered for rogue status (although other constraints must be met, such as no more than 2 adjacent pixels in either direction are allowed to be rogues).

[2] Starck, J.-L, Murtagh, F., & Louys, M. 1995, in *Astronomical Data Analysis Software and Systems IV*, eds. R. A. Shaw, H. E. Payne, and J. J. E. Hayes, ASP Conf. Ser. Vol 77

3. The detection of rogues in clusters. For values of **`AGGRESSIVE`** less than or equal to 0.5, a pixel which exceeds the sigma threshold can only be considered a rogue if it has no neighbors that also fulfil this criterion. For **`AGGRESSIVE`** greater than 0.5, however, rogues can be found in clusters, so long as a candidate rogue cluster is no larger than 2 pixels in either direction. For **`AGGRESSIVE`** greater than 0.5, in addition to accepting single rogue pixels that are next to already-found rogues, the program will deliberately search for additional clusters of rogues using a hierarchical segmentation technique[3] to identify continguous significant regions in the first two multimedian components.

The consequences of changing **`AGGRESSIVE`** are summarized in Table 6.1. Note: underlines indicate a value is included in the range.

Table 6.1 Changing the AGGRESSIVE Parameter

Aggressive (getFmask=1)	**Number of Iterations**	**Sigma Threshold**	**Clusters?**
$\le \underline{0}$	1	$\ge \underline{4.0\sigma}$	No
0 to $\underline{0.5}$	1 to $\underline{6}$	4 σ to $\underline{3.0\sigma}$	No
0.5 to 2.0	6 to 19	3σ to 0	Yes
$\ge\underline{2.0}$	$\ge\underline{19}$	$\underline{0}$	Yes

NOTE: To Include Negative Pixels in the Search, use the keywords **`/ABS_VAL`** or **`/NEGATIVE_ONLY`** in conjunction with **`getFmask`**. Normally the rogue-finding algorithm only looks for positive-valued rogue pixels. But sometimes there are high negative pixels that you also want to mask. (This is often the case when you are using background-subtracted data where the source and background images both had rogue pixels and the subtraction does not cancel out the rogues exactly.) The keyword **`/ABS_VAL`** tells IRSCLEAN that you want to look for *both* "positive" and "negative" rogue pixels. The keyword **`/NEGATIVE_ONLY`** tells IRSCLEAN to only look for "negative" rogues.

6.5.2 getFmask=2: Double Unsharp-Mask Flattening Method (`getFmask=2, Rogue_Thresh,Noise_Floor, AGGRESSIVE,/ABS_VAL,/NEGATIVE_ONLY`*)*

This method, new with IRSCLEAN 2.0, attempts to remove slowly varying trends from the data (“flattening”) through a double application of unsharp-masking, which (in theory) leaves only single or a few-pixel spikes (rogue pixels). The rogues can be found by thresholding the flattened data.

```
IDL> irsclean_mask, $
     'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
      DataRange=[13,53],getFmask=2,Noise_Floor=0,Rogue_Thresh=3
```

[3] Starck, J.-L., Bijaoui, A., Valtchanov, I., & Murtagh, F. 2000, A&A Supp., 147, 139

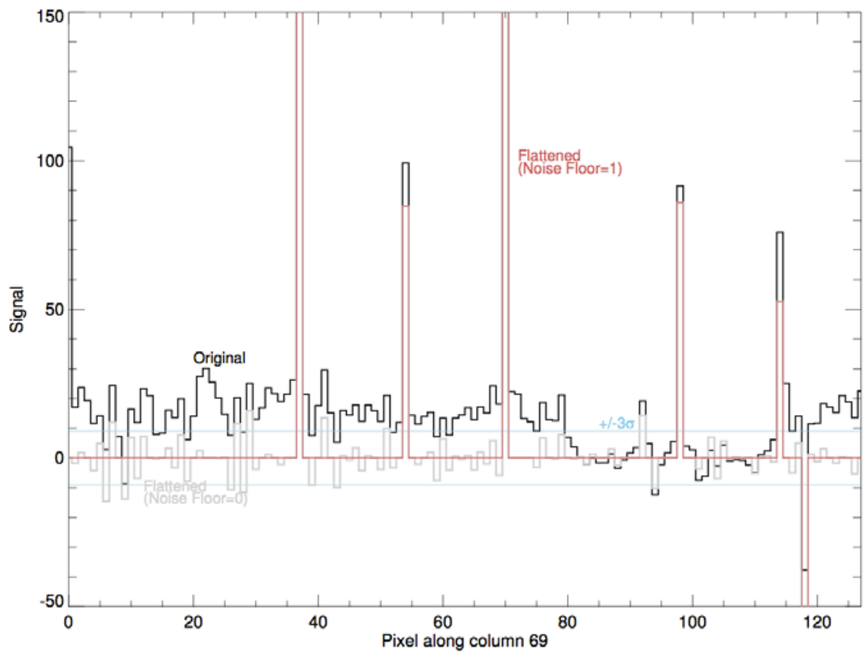


Figure 6.2 Double unsharp-mask flattening method of rogue pixel detection, Long-Low data along row 69.

We show here (Figure 6.2 and Figure 6.3) two examples of the double unsharp-mask flattening method of rogue pixel detection as applied to sample data files located in the IRSCLEAN distribution. The first example (Figure 6.2) uses Long Low data located in **`sampledata/r13349376/ch2/bcd/SPITZER_S2_13349376_0008_0000_8_bcd.fits`**. The plot shows data along colum 69 of this image (black), and double unsharp-mask flattened versions of it, for two different values of keyword **`Noise_Floor: 0`** (gray) and **`1`** (red). Rogue pixels are clearly visible to the eye as "spikes", single pixels with anomalously high signal, but given the presence of gradients, it is difficult to identify them automatically from the original data. The algorithm performs a first unsharp masking by subtracting a two-dimensional median-filtered image (smoothing kernel of 3 pixels) from the original data. (This is the first component of the multi-median transform described above.) The progam clips the resulting data, setting to zero all values below a noise floor given by the keyword **`Noise_Floor`** times the Gaussian standard deviation. The default is **`Noise_Floor=1.`** Setting **`Noise_Floor=0`** turns off clipping, and allows all data to be considered signal. The next step is to unsharp mask the *columns* of the resulting image, approximating the removal of behavior along the dispersion (wavelength) direction with characteristic scales greater than 3 pixels (i.e., continuum and resolved spectral features). The resulting "flattened" image usually has very few noticeable gradients or bumps, but retains "spiky" rogue pixels. A simple threshold will then enable the identification and masking of the rogues. Use the **`Rogue_Thresh`** keyword to tailor the multiple of the standard deviation above which a pixel in the flattened image is considered a rogue pixel (the default is **`Rogue_Thresh=3`**). The flattened profiles in the figures above display what happens when **`Noise_Floor`** is varied. The gray curves (**`Noise_Floor=0`**)show many non-rogue pixels that exceed the $\pm 3\sigma$ threshold (as shown by the blue lines), but the red curves (**`Noise_Floor=1`**)appear to isolate the rogue pixels quite well.

The second example (Figure 6.3) is Long High data located in s**`ampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits`**. As for the previous example, setting a **`Noise_Floor=0`** (gray) erroneously allows non-rogue pixels to pass the **`Rogue_Thresh=3`** criterion, whereas the red curve shows that the only pixels with absolute value greater than $\pm 3\sigma$ are rogues.

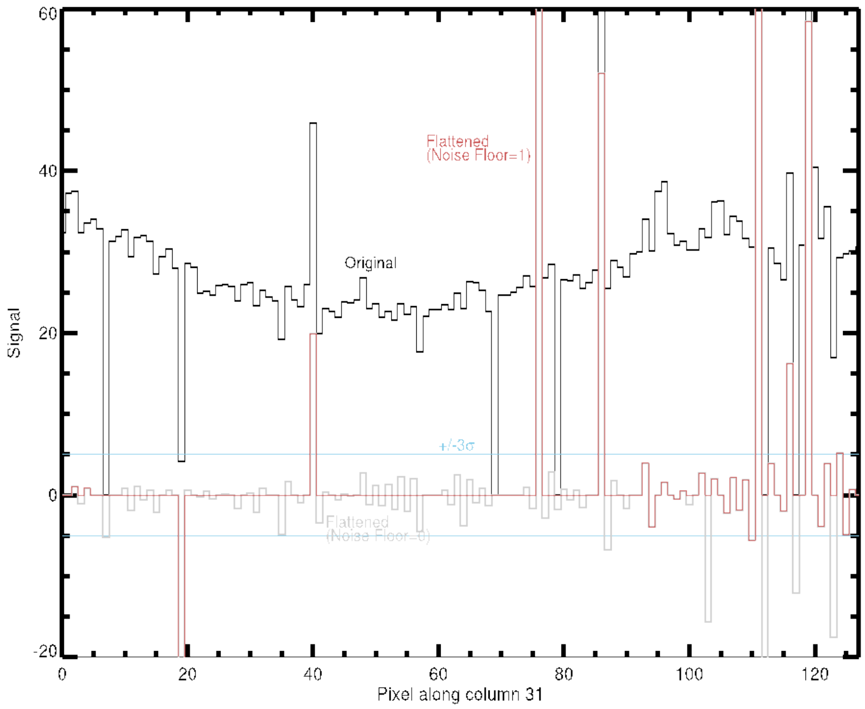


Figure 6.3 Double unsharp-mask flattening method of rogue pixel detection, Long-High data along row 31.

Note: **`Noise_Floor`** and **`Rogue_Thresh`**, relate to two different measures of the spread of the data. The noise standard deviation (**`Noise_Floor`** units) is derived assuming Gaussian statistics, so this value is usually a few times higher than the robust estimate of σ used to threshold rogue pixels (**`Rogue_Thresh`** units). Default values of these parameters (**`Noise_Floor=1`** and **`Rogue_Thresh=3`**) work well for the sample data. If you are not having any luck with these settings, however, the best approach is to vary them slowly around the defaults. Remember, reducing either **`Noise_Floor`** or **`Rogue_Thresh`** enables the detection of *more* rogue pixels.

The /**`ABS_VAL`** and **`/Negative_Only`** keywords have the same meaning under **`getFmask=2`** as under **`getFmask=1`**. Furthermore, the **`AGGRESSIVE`** keyword can be used in place of **`Rogue_Thresh`** when **`getFmask=2`**. They are related via the equation:

$$\mathrm{AGGRESSIVE} = 1 - \frac{\mathrm{Rogue\_Thresh} - 2}{2}.$$

6.6 Specifying Your Own Subroutine for Automatic Mask Finding (**`maskFunction=, AGGRESSIVE, /ABS_VAL, /NEGATIVE_ONLY`**)

Some users find that the built-in rogue-finding algorithm invoked by **`getFmask`** is either too aggressive, or not aggressive enough in finding rogue pixels. You may separately (or additionally) specify your own subroutine to create a rogue pixel mask from the input image using the keyword **`maskFunction = [function name]`**. The **`[function name]`** is the name (or array of names) of a user-supplied IDL function (or set of functions) that independently creates a rogue mask from your data. Each function should be in its own file in your IDL_PATH, with the same name (plus "**.pro**") as the function itself. Each user-defined function will be called from inside the **irsclean_mask** main program. The function declaration statement must be of the following form.

```
FUNCTION irsclean_threshall,orig_img,omask,AGGRESSIVE=AGGRESSIVE,$
         NEGATIVE_ONLY=NEGATIVE_ONLY,ABS_VAL=ABS_VAL
```

The first entry after the **FUNCTION** statement is of course the function name. This is followed by the following local IRSCLEAN variables:

- **orig_img**: the fits data used to find the mask (eg., input using **dataFile**)
- **omask**: the order mask image, obtained automatically by **irsclean_mask**, specifying the order number as a function of pixel. (See Appendix 2 for an image showing the mapping of order number to pixel in the IRS arrays).
- **AGGRESSIVE, NEGATIVE_ONLY, and ABS_VAL**: the values of these keywords that you input to **irsclean_mask**

All user-defined functions should have the same definition statement, with the exception of the function name. (This admittedly inflexible calling sequence may be relaxed in later releases.) The output of your function call should be a 128 x 128 byte array of 1's and 0's. If it is not, the result will be ignored.

The sample function **irsclean_threshall.pro** (the first line of which is copied above) is included with the IRSCLEAN distribution. This function designates as bad all pixels greater than a quantity of sigma, dependent on the value of the **AGGRESSIVE** keyword. Sigma is measured as the standard deviation in the inter-order region. You can invoke this function by adding **maskFunction = 'irsclean_threshall'** to the **irsclean_mask** call. Keywords **/ABS_VAL** and **/NEGATIVE_ONLY** work as described above (6.5.1).

```
IDL> irsclean_mask, $
    'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
     maskFunction='irsclean_threshall',AGGRESSIVE=0.5,DataRange=[13,53]
```

6.7 Combining Masks

An input rogue mask (**inRmask=**), the found mask (**getFmask=**), and any user-defined mask (**maskFunction=**) will be combined together in one super-mask if you invoke more than one of these mask input keywords. IRSCLEAN will combine the binary rogue, found, and user-defined masks using the Boolean relational operator OR.

```
IDL> irsclean_mask, $
    'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
     inRmask='all_campaigns_roguemasks/b3_rmask_IRS18.fits',$
     maskFunction='irsclean_threshall',getFmask=2,DataRange=[13,53]
```

6.8 Mask-edit Mode

The heart of the mask-making process is Mask-edit Mode, where IRSCLEAN will overlay the current mask as built from the rogue mask, found mask, and/or user-defined mask (or no mask at all if you have decided to build a mask from scratch), and you have the opportunity to accept or reject masked pixels, and even add new ones.

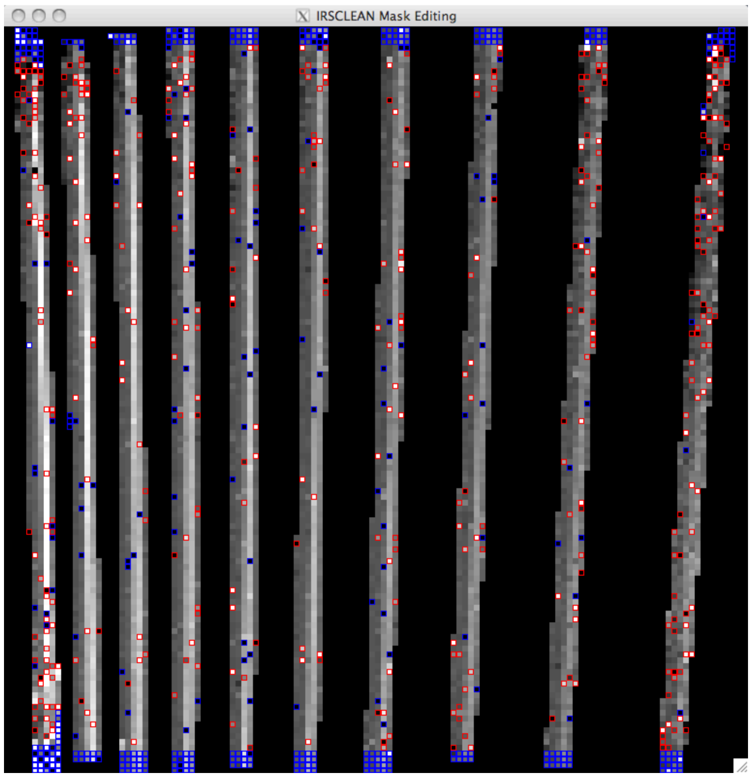


Figure 6.4 Entering mask edit mode.

For example, the result of typing

```
IDL>  irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val
```

is shown in Figure 6.4. Red squares indicate pixels in the rogue mask (derived using the double unsharp-mask method via `getFmask=2`). Blue squares indicate pixels that have been masked fatally by the pipeline BMASK. Interorder pixels are blanked out (in display only), since these pixels will not get cleaned and do not contribute to the extracted spectrum. The automatic rogue-finding algorithm did a decent job of finding most of the rogue pixels, although there are a few missed rogues and some false positives.

Now you have the chance to select pixels with the mouse cursor. Clicking on a pixel with the left mouse button will toggle its mask status "on" or "off". Alternatively, you can toggle the mask status of a pixel by typing '**m**' at the keyboard.

NOTE: while it is possible to toggle the status of a blue BMASK pixel to rogue and clean it, thus bringing the pixel back "into the fold," *we recommend against this*. The IRS spectral extraction software can handle the few BMASK pixels that occur in the middle of an order; they will not affect your spectrum. And there is not enough information to clean properly the BMASK pixels on the edges of a spectral order.

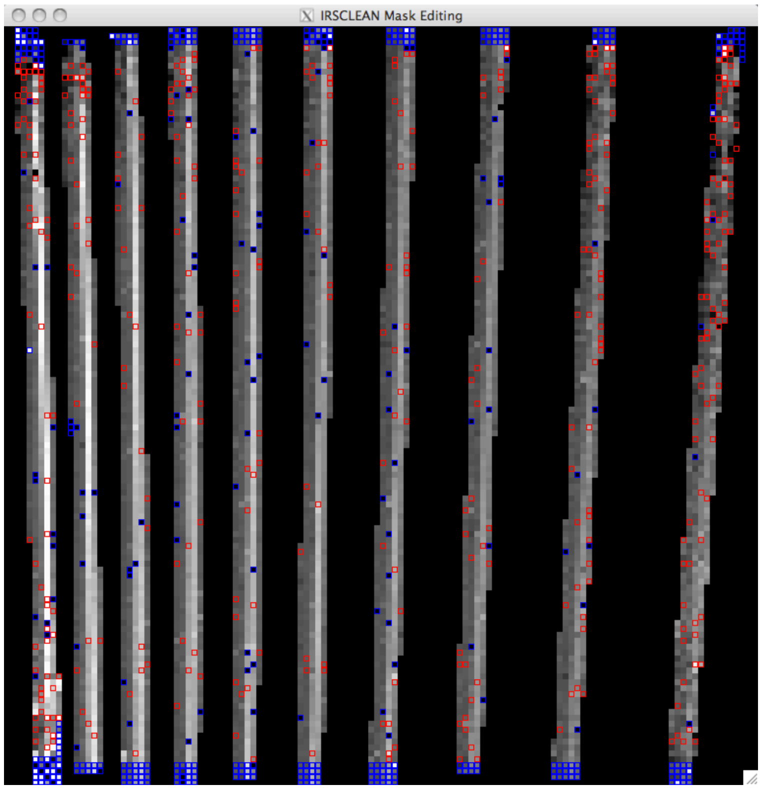


Figure 6.5 Mask-edit mode, after cleaning.

If you need to rescale your image during mask-editing, you can type ‘**r**’ to open the manual display scaling widget (Section 6.2).

Clicking the right mouse button in the graphics window (or typing ‘**c**’) will clean the data using the current mask, and overlay the mask on the cleaned image for your examination. After editing the mask above to remove illuminated pixels and add missed hot (or cold) bad pixels, we click the right mouse button or type ‘**c**’, and obtain the image in Figure 6.5.

NOTE: Pixels in between spectral orders (defined by the order mask) will not get cleaned. They are "blanked out" in display.

NOTE: for keyboard shortcuts to work, the cursor focus needs to be on the *terminal* in which you invoked IDL (the cursor should be active). This is usually the case at the beginning of your session, but the focus can move to another window if you go through more than one mask-editing iteration (for example, when you click ‘**No**’ on the “Done editing mask” dialogue or rescale your image by typing ‘**r**’). **Keyboard shortcuts will *not* work from the IDL Workbench.**

Table 6.2 IRSCLEAN Keyboard Shortcuts (Mask-edit mode)

Button	**Keyboard Shortcut**	**Action**

LEFT	**m**	Toggle mask pixels on and off
MIDDLE	**q**	Quit without saving
RIGHT	**c**	Clean data using current mask
	r	Rescale image manually (retaining current mask)

The mask we produced in this example is saved in **`sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_4_rmask.fits`** under the IRSCLEAN distribution.

You are then asked if you want to save the mask and exit Mask Edit Mode. If you choose '**No**' you will return to Mask-Edit mode and be allowed to add or remove pixels from the mask again. **NOTE: You must choose "Yes" to proceed to Stage 2, data cleaning. If you do not wish to save the mask, then select "Cancel" in the mask-saving dialog.** Only pixels in the order mask are cleaned, so inter-order pixels do not get cleaned even though they might be picked as rogue pixels.

In Mask-Edit Mode, you can also click the middle mouse button in the graphics window (or type '**q**') and quit the whole program. The mouse and keyboard choices available during Mask-Edit Mode are listed in Table 6.2.

6.9 Saving Your Mask (`outRmask_File`)

The result of a successful IRSCLEAN session is a new rogue mask. When you click the right mouse button or type '**c**' in Mask-Edit mode, the program will ask you are done editing your mask. If you are satisfied, hit '**Yes**' and you will be allowed to pick an output filename to which to save the mask, unless you have already set the value (using the keyword **`outRmask_File`**). The default file name is of the form **`b[chan]_rmask.fits`**, where **`chan`** indicates the IRS channel number (**`0`**=Short Low, **`1`**=Short High, **`2`**=Long Low, **`3`**=Long High).

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val,$
outRmask_File='SPITZER_S3_13349376_0012_0000_4_rmask.fits'
```

NOTE: You are not required to save a mask file. Simply choose "Cancel" in the file saving dialogue to move on to the next stage.

6.10 Saving Your Cleaned Data (`outClean_File`)

After saving your mask, you are also given the opportunity to save the cleaned image that is currently displayed in your IDL graphics window. If you answer "**Yes**" to the query, then a file-saving dialog will open, unless you have already set the **`outClean_File`** keyword. As usual, setting **`outClean_File=''`** (blank string) will automatically load the file picker. The default file name is taken from the input data file, with the suffix "**`_clean`**" inserted before the "**`.fits`**" extension. A HISTORY line to the effect that the file has been cleaned using **`irsclean_mask.pro`** will be added to the FITS header. **NOTE: If they exist, the corresponding uncertainty and mask files will also be "cleaned" automatically** (see "Cleaning Uncertainty and Mask Files," below).

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val,$
outRmask_File='SPITZER_S3_13349376_0012_0000_4_rmask.fits',$
outClean_File='SPITZER_S3_13349376_0012_0000_4_bcd_clean.fits'
```

6.11 Automatically Saving the Rogue Mask and Cleaned Data (**/Autosave**)

The **/Autosave** keyword will skip file picker dialogs for **outRmask_File** and **outClean_File**, and save the mask and cleaned images in files with default names (see Sections 6.9 and 6.10 for details).

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val,/AutoSave
```

6.12 Skipping Mask-Edit Mode (**/noMaskEdit**)

It is possible to skip interactive editing of the rogue mask, using the **/noMaskEdit** keyword. This is useful when you are processing a data file with a predefined rogue mask (or the Campaign rogue mask), or if you are calling **irsclean_mask** from another program as a batch process.

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits', $
inRmask_File='sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_rmask.fits
',/noMaskEdit,/autoSave
```

6.13 Quitting After Stage 1 (**/noStage2**)

In general, the purpose of Stage 1 is to create a bad pixel mask to apply to one or more BCD images in Stage 2. In many cases, however, a mask is only applicable to the BCD image that was used to create it. By default, the program normally flows directly to Stage 2 after saving the cleaned image used to make the mask. To quit instead after making the mask and/or cleaning its “parent” BCD image, use the keyword **/noStage2**.

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val,/AutoSave
```

6.14 Full Batch Mode (**getFmask, /CampaignRMask, inRmask_File, /noMaskEdit, /autoSave, /noStage2**)

To run IRSCLEAN in a fully noninteractive manner, invoke **getFmask**, /**CampaignRMask**, or **inRmask_File** (or any combination thereof—the goal is to define a rogue mask outside of Mask-Edit Mode); and set **/noMaskEdit**, **/autoSave**, and **/noStage2**:

```
irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits', $
```

```
inRmask_File='sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_rmask.fits
',/noMaskEdit,/autoSave,/noStage2
```

7 Stage 2: Cleaning A Set of Data

Once you have a rogue mask that you are satisfied with, IRSCLEAN moves on to "Stage 2", (unless you have set the keyword **`/noStage2`**). Now you will be given the chance to apply the mask to other data files from the same IRS array. **NOTE: Since rogue pixels can vary on timescales of hours, the effectiveness of a rogue mask for cleaning data is usually limited to data from the same AOR used to derive the mask.**

NOTE: The pixel cleaning algorithm is found in the IDL function **`imbadfix.pro`**, which is part of the IRSCLEAN distribution. Pixel cleaning is accomplished by deriving profiles for the 3 rows above and below the bad pixel, predicting the profile for the bad pixel's row, scaling that profile by the good pixels in the bad pixel's row, and substituting the value for the pixel in question from the predicted and scaled profile.

7.1 Picking Files to Clean (`filesToClean`)

NOTE: If you want to skip Stage 2 but forgot to set `/noStage2`, you can quit by choosing "Cancel" when asked to choose files to clean.

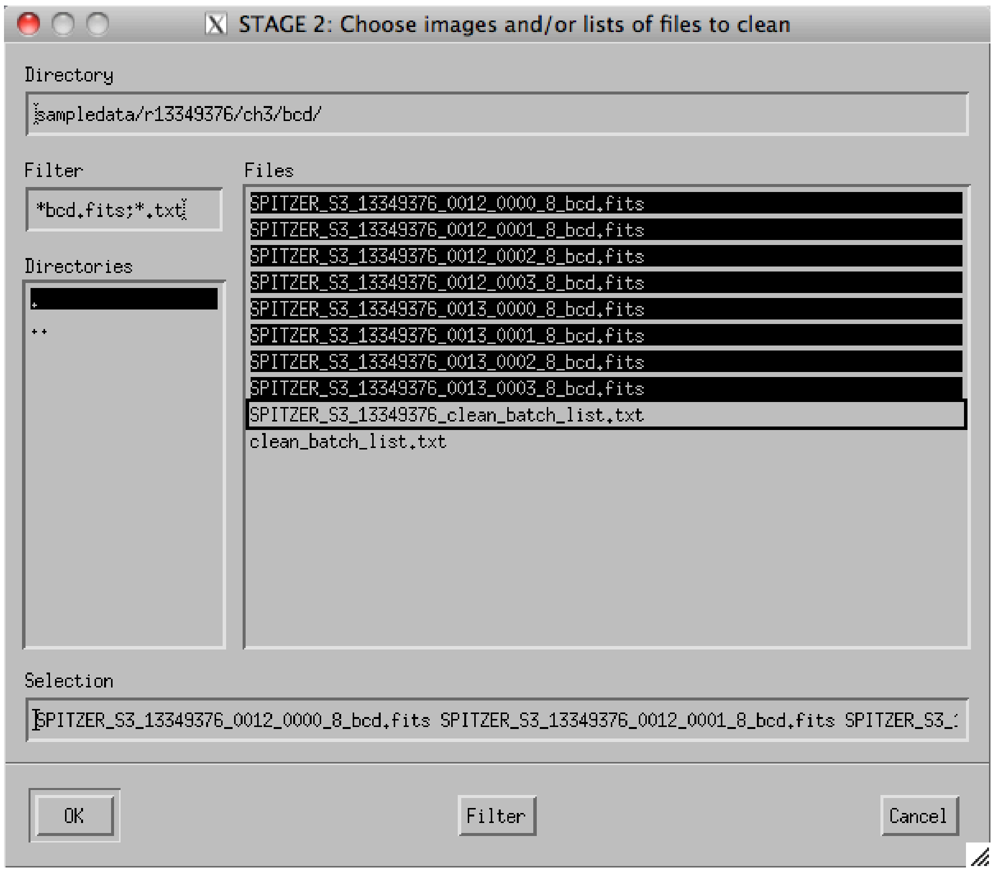
If you have not set the **`filesToClean`** keyword, IRSCLEAN opens a file picker dialog, which you can use to pick any (or all) of the following:

1. One or more **`.fits`** files.
2. One or more **`.txt`** (ASCII text) files that list a set of fits files, one per line.

An example **`.txt`** file is as follows:

```
#  an example .txt file
SPITZER_S3_13349376_0012_0000_8_bcd.fits
SPITZER_S3_13349376_0012_0001_8_bcd.fits
SPITZER_S3_13349376_0012_0002_8_bcd.fits
SPITZER_S3_13349376_0012_0003_8_bcd.fits
SPITZER_S3_13349376_0013_0000_8_bcd.fits
SPITZER_S3_13349376_0013_0001_8_bcd.fits
SPITZER_S3_13349376_0013_0002_8_bcd.fits
SPITZER_S3_13349376_0013_0003_8_bcd.fits
```

NOTE: Comments may be embedded in `.txt` files by putting the pound sign (#) at the beginning of a line.


STAGE 2: Choose images and/or lists of files to clean
Directory
sampledata/r13349376/ch3/bcd/
Filter
bcd.fits;.txt
Files
Directories
.
..
Selection
OK
Filter
Cancel

```
SPITZER_S3_13349376_0012_0000_8_bcd.fits
SPITZER_S3_13349376_0012_0001_8_bcd.fits
SPITZER_S3_13349376_0012_0002_8_bcd.fits
SPITZER_S3_13349376_0012_0003_8_bcd.fits
SPITZER_S3_13349376_0013_0000_8_bcd.fits
SPITZER_S3_13349376_0013_0001_8_bcd.fits
SPITZER_S3_13349376_0013_0002_8_bcd.fits
SPITZER_S3_13349376_0013_0003_8_bcd.fits
SPITZER_S3_13349376_clean_batch_list.txt
clean_batch_list.txt
```

```
SPITZER_S3_13349376_0012_0000_8_bcd.fits SPITZER_S3_13349376_0012_0001_8_bcd.fits SPITZER_S3_1
```

Figure 7.1 Choosing files to clean (Stage 2).

The program is flexible enough to allow you to choose a combination of **.fits** and **.txt** files at the same time. The files will be parsed to create a master list of all **.fits** files to be processed. For Unix-type operating systems (including Mac OS X), multiple files can be selected by holding down the '**Control**' key and clicking on the files. An example of a user choosing the sample files listed above is displayed in Figure 7.1.

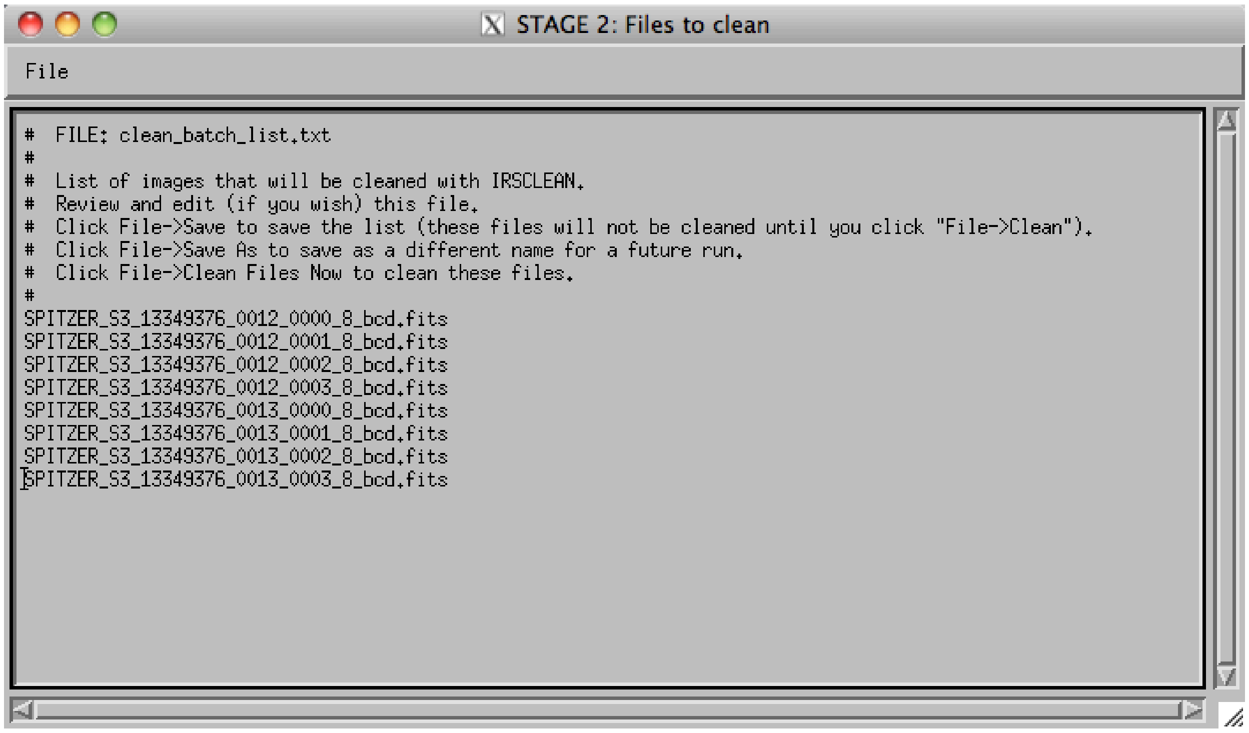
The file selection process is subject to the following rules:

- All **.fits** and **.txt** files chosen via the dialogue must be in the same directory. However, **.fits** files listed inside a **.txt** file can come from anywhere, so long as the full path to the file is specified inside the **.txt** file.
- For **.fits** files listed inside a **.txt** file, the default path is the source directory of the **.txt** file. If you want to use a different directory, you should specify it explicitly in the entry.
- If you accidentally choose the same file (full path) more than once, it will only be processed once. On the other hand, files with the same base-names that reside in different directories are not considered duplicates and will be processed separately.

To skip interactive file-picking, set the **filesToClean** keyword at the command line. For example,

```
IDL> irsclean_mask, $
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits', $
inRmask_File='sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_rmask.fits
',/noMaskEdit,/autoSave,$
filesToClean='sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_clean_batch_list.txt'
```

If the path is not given explicitly, files are assumed to be in the current working directory.



```
#  FILE: clean_batch_list.txt
#
#  List of images that will be cleaned with IRSCLEAN.
#  Review and edit (if you wish) this file.
#  Click File->Save to save the list (these files will not be cleaned until you click "File->Clean").
#  Click File->Save As to save as a different name for a future run.
#  Click File->Clean Files Now to clean these files.
#
SPITZER_S3_13349376_0012_0000_8_bcd.fits
SPITZER_S3_13349376_0012_0001_8_bcd.fits
SPITZER_S3_13349376_0012_0002_8_bcd.fits
SPITZER_S3_13349376_0012_0003_8_bcd.fits
SPITZER_S3_13349376_0013_0000_8_bcd.fits
SPITZER_S3_13349376_0013_0001_8_bcd.fits
SPITZER_S3_13349376_0013_0002_8_bcd.fits
SPITZER_S3_13349376_0013_0003_8_bcd.fits
```

Figure 7.2 Editing the list of files to clean (Stage 2).

7.2 Reviewing Files to Clean

After all `.fits` file names have been extracted from the input `.txt` files and combined with any `.fits` files given explicitly, the parsed list will be saved to the file `clean_batch_list.txt` in the same directory as `dataFile`. Then IRSCLEAN will open a file-editing dialog, so you can review and edit (if necessary) the complete list. An example editing window is shown above (*previous page*).

For your convenience, the full path to each file will be listed, but you are allowed to add files to the list without giving the path, provided the file is in the current working directory. (File names have been edited in the display above to remove paths.)

The FILE dropdown menu of the file-editor allows you three choices:

1. **`File->Save`** Allows you to save the current list as `clean_batch_list.txt`. **`File->Save`** defines the set of files to be cleaned.

2. **`File->Save As`** Allows you to save the current list under a different name. **NOTE: A list saved in this way will not be cleaned in this run, but can be used as input to a future run of IRSCLEAN (either by picking it in the dialog window, or entering it using the command line keyword `filesToClean`).**

3. **`File->Clean Files Now`** Exits the file editing window and cleans the set of files in the currently saved `.txt` file `clean_batch_list.txt`.

NOTE: Each session of IRSCLEAN will recreate `clean_batch_list.txt`, so if you wish to keep your list of files to clean for future sessions, remember to use ‘`File->Save As`’ to write a unique file.

7.3 Cleaning Files

After you have edited the list to your satisfaction and select **`File->Clean Files Now`**, the files in your list will be cleaned using your rogue mask and IRSCLEAN. The cleaned images will be stored in **`.fits`** files with the same name as the original images, but with **`_clean`** inserted before the **`.fits`** suffix. For example the clean image corresponding to **`'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits'`** is stored in **`'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd_clean.fits'`**. A **`HISTORY`** keyword is added to the FITS header in the clean file, specifying the date and time that IRSCLEAN was run on the file.

NOTE: If a file in your list does not exist or is not a fits file, you will see a message to that effect. No action will be taken for that entry. Also, if a file is not from the same IRS array as the rogue mask, it will not be cleaned.

7.4 "Cleaning" Uncertainty and Mask Files

Table 7.1 Uncertainty and Mask file names

Data	**Uncertainty**	**BMASK**
`[prefix]_bcd.fits` `[prefix]_dks_bcd.fits`	`[prefix]_func.fits`	`[prefix]_bmask.fits`
`[prefix]_droop.fits`	`[prefix]_drunc.fits`	
`[prefix]_rsc.fits`	`[prefix]_rscu.fits`	
`[prefix]_f2ap.fits`	`[prefix]_f2unc.fits`	
`[prefix]_[*]_coa2d.fits`	`[prefix]_[*]_c2unc.fits`	`[prefix]_[*]_c2msk.fits`
`[prefix]_[*]_bksub.fits` `(Low Resolution only)`	`[prefix]_[*]_bkunc.fits`	`[prefix]_[*]_bkmsk.fits`

For every data file in your list, IRSCLEAN will automatically look for a corresponding uncertainty and bmask file and produce “clean” versions of those files as well. The relationship between the name of a data file and that of its uncertainty and bmask files is shown in Table 7.1. To be processed, uncertainty and bmask files need to be in the same directory as the processed image. Otherwise no action is taken.

Just as for the data files, the “cleaned” uncertainty and bmask images will be stored in **`.fits`** files with the same name as the original files, but with **`_clean`** inserted before the **`.fits`** suffix. For example, the cleaned version of **`'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_func.fits'`** is stored in **`'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_func_clean.fits'`**. A **`HISTORY`** keyword is added to the FITS header in the clean file, specifying the date and time that the file was processed.

If found, the uncertainty and bmask files are altered by IRSCLEAN as follows:

- **The uncertainty value in cleaned pixels is interpolated from neighboring pixels** using the same algorithm as for data. This ensures that the pixel has minimal impact on the computation of the uncertainty in the corresponding wavelength bin of the extracted spectrum. In earlier versions, uncertainties of cleaned pixels were set to **`NaN`** (not a number). This had the inadvertent effect of eradicating (setting to **`NaN`**) the error estimate for the entire wavelength bin of a cleaned pixel.
- **The bmask value in cleaned pixels is set to `1` (one).** In other words, we turn on the zeroth bit and turn off any other bits that may have been set in the bmask by the pipeline. Only pixels cleaned using IRSCLEAN will have the zeroth bit set in the bmask. Setting the bmask value of cleaned pixels to **`1`** also ensures that cleaned pixels are always included in spectral extraction in SPICE, since any fatal masking of rogue pixels by the BCD pipeline will be cleared.

8 Using IRSCLEAN Results in SPICE

Once you have one or more cleaned images, you can use the results in SPICE. Use the "**`clean`**" versions of each of the three files in the "Input" tab of SPICE, the same way you would use their non-cleaned counterparts. As noted previously, these will be produced automatically by IRSCLEAN, provided the originals were available in the same directory as the image file to be cleaned.

NOTE: To extract a spectrum properly from a cleaned image, you must use the cleaned versions of the uncertainty and bmask files, in addition to the clean image itself. If you use the original uncertainty and/or bmask files, some cleaned pixels may not be used, and the output uncertainty spectrum may incorporate unrealistic errors for cleaned pixels.

We show below a comparison between the spectrum extracted from the original Long High image **`SPITZER_S3_13349376_0012_0000_8_bcd.fits`** using uncertainty **`SPITZER_S3_13349376_0012_0000_8_func.fits`** and mask **`SPITZER_S3_13349376_0012_0000_8_bmask.fits`**. and the spectrum extracted from the cleaned image **`SPITZER_S3_13349376_0012_0000_8_bcd_clean.fits`** using uncertainty **`SPITZER_S3_13349376_0012_0000_8_func_clean.fits`** and mask **`SPITZER_S3_13349376_0012_0000_8_bmask_clean.fits`**. (The image was cleaned using the rogue mask depicted in earlier examples above.) The blue curve is the original spectrum and the red curve is the spectrum of the cleaned data. Most of the spikes in the red spectrum have been removed by cleaning.

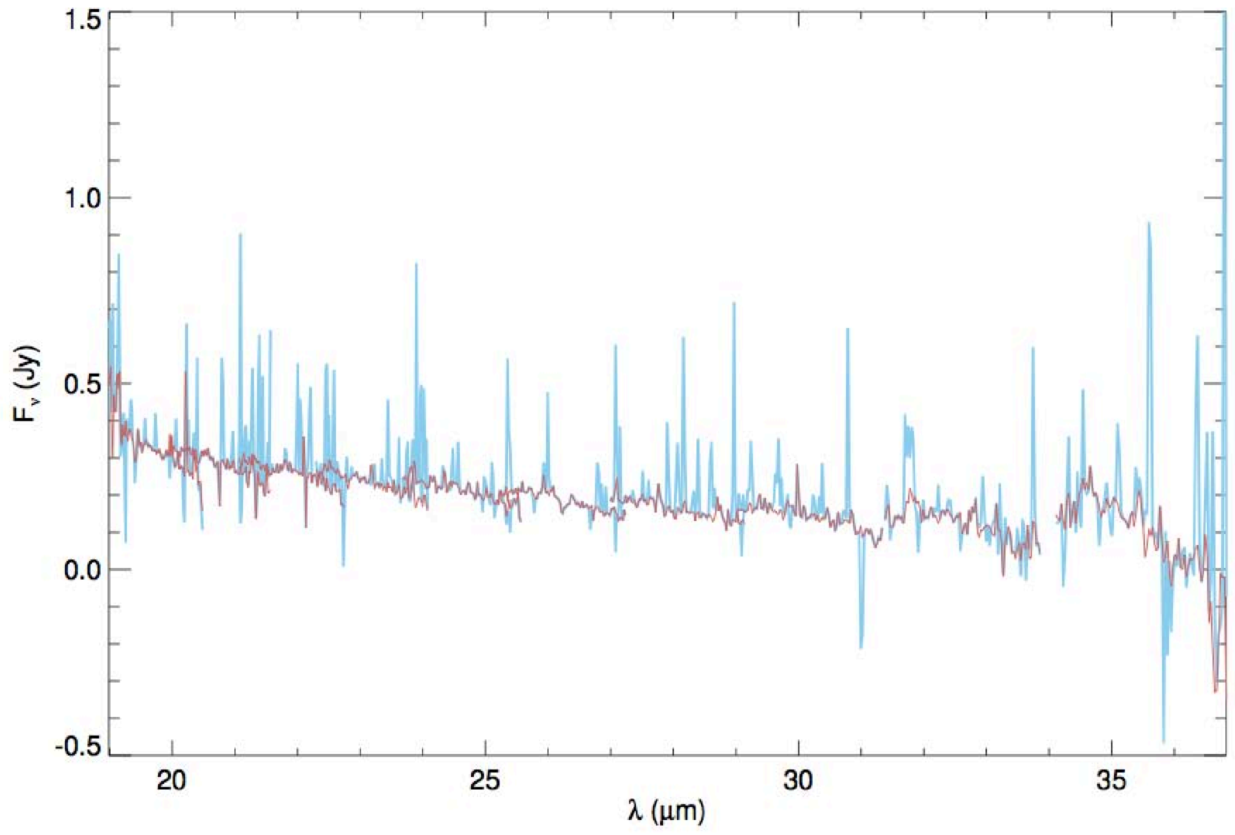


Table 8.1 Extracted spectra for sample bcd data. Shown are the original data (blue) and the cleaned data (red).

That's it! We hope IRSCLEAN helps improve the quality of your data. Feel free to report problems to us as you find them.

9 Appendix 1: Hints

9.1 Automatic Command Line Suggestion

Every session of IRSCLEAN will result in a string being printed to the IDL terminal that gives the equivalent IDL command that you could have typed to get the same set of options non-interactively. You might find this helpful for subsequent runs where you want to vary some but not all parameters and don't want to have to type a long path or pick a file using the dialog again. It is also useful to build a template command for inclusion in other IDL procedures.

9.2 Getting Help (`/Help`)

To get full documentation on the input and output parameters of IRSCLEAN, set the keyword **`/HELP`**:

```
IDL> irsclean_mask,/HELP
```

9.3 Setting the Working Directory (**`Directory`**)

The working directory for IRSCLEAN is the directory from which files are chosen and to which outputs are written (unless specified otherwise by a full path file name). By default, the starting working directory of IRSCLEAN is the current working directory. If you haven't done anything to change directories during your IDL session, this is the directory you started up IDL in. If you want to set the value explicitly, use the keyword **`Directory`**:

```
IDL> irsclean_mask,DIRECTORY='/home/me/idl/IRSCLEAN2.1/sampledata'
```

9.4 Changing the Default Size of the Displayed Image (**`WinSize`**)

The default image display size for Mask-Editing mode is 768 x 768 pixels. This seems to fit in most desktop and laptop screens and still allow for easy mouse editing. If you need to shrink the size of the image or would like to make it bigger, set the WinSize keyword to a two-element vector with the new size in x and y pixels**. We recommend that you use the same numbers for both x and y, and stick to multiples of 128.**

```
IDL> irsclean_mask,WinSize=[1024,1024]
```

9.5 Changing the Default Brightness Scaling of the Displayed Image (**`dataRange`**)

The default for image scaling in Mask-Editing mode is to display the inner 96% of pixel values. In many cases this allows sufficient contrast to identify rogue pixels. To change this from the command line, set the **`dataRange`** keyword.

```
IDL> irsclean_mask,dataRange=[0,100]
```

You can scale the image manually using the mouse. To do this prior to Mask-Editing mode, set **`dataRange=[0,0]`** at the command line. You can bring back the manual scaling widget from within Mask-Edit mode by typing '**r**'. See Section 6.8.

9.6 Changing the Default Colors for the Rogue Mask and BMASK (**`colorNames`**)

Normally rogue pixels are highlighted red and BMASK pixels are highlighted blue, but these colors can be changed at the command line by setting the **`colorNames`** keyword to a 2-element string array listing the rogue and BMASK color names, respectively. Available color names can be found by typing

```
IDL> print,FSC_COLOR(/names)
```

Or to visually choose a color name, type

```
IDL> print,PICKCOLORNAME()
```

The result of typing the following command is shown in Figure 9.1.

```
IDL> irsclean_mask,
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',getFmask=2,
DataRange=[-22.7,52.36],/abs_val,colornames=['dark orchid','green']
```

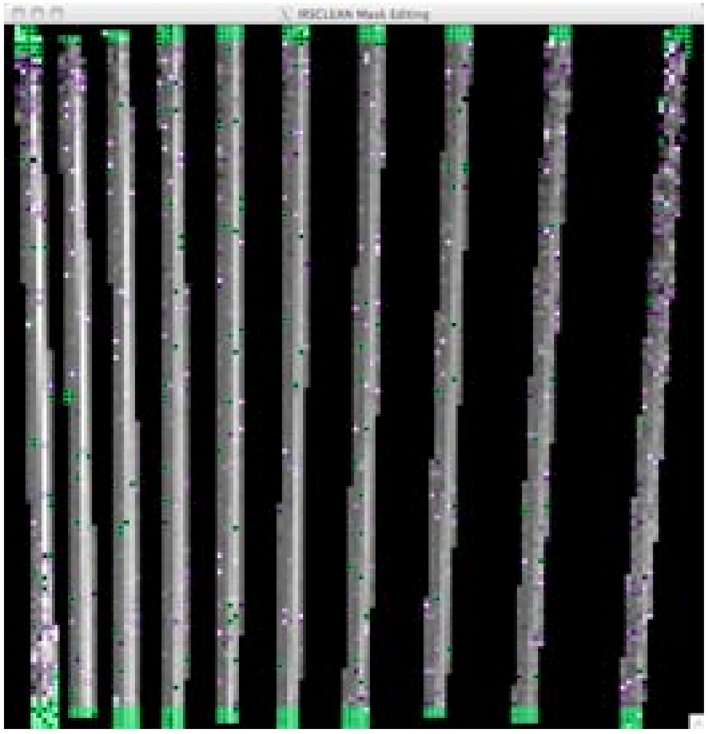


Figure 9.1 Using different colors for rogue and bmask pixels (dark orchid for rogues and green for bmask pixels).

9.7 Adding NaN Pixels to the Rogue Mask (`/NAN`)

If you set the **`/NAN`** keyword in IRSCLEAN, you can include in the rogue mask pixels with their values set to "not a number," or **`NaN`**. If other masks are input or derived, the **`NaN`** mask will be combined with them.

```
IDL> irsclean_mask,/NAN
```

NOTE: As of the S18.7 pipeline processing version, most `NaN` pixels are already fatally masked in the BMASK, so these pixels will not by default be allowed in the rogue mask unless you choose them deliberately during Mask-Edit Mode. Furthermore, SPICE and the online pipeline automatically clean `NaN` pixels during spectral extraction, so it is no longer necessary to add `NaN's` to the rogue mask.

If you do choose to use IRSCLEAN to clean **`NaN`** pixels, make sure that pixels were actually cleaned properly, because cleaning removes the "fatal" flag from the bmask. Poorly-cleaned pixels will add noise to the extracted spectrum. In particular, watch out for clusters of **`NaN`**'s on the edges of an order, which are very difficult to clean. It is probably better to leave such pixels to SPICE.

9.8 Using the Cursor Value to Help Find Rogues by Eye

During Mask-Editing mode the cursor position, pixel value, and wavelength are printed to the screen. Sometimes the pixel value is useful for confirming suspected rogues found by eye. If all surrounding pixels have much lower values, it is likely that the pixel is a rogue. (Be careful, however, of mistaking spikes due to unresolved emission lines or faint marginally resolved point sources as rogues.)

9.9 A second pass at cleaning data using the **`unclean_mask`** file.

Not all pixels in your bad pixel mask are always successfully cleaned by IRSCLEAN. See the description of the "unclean" mask above.

It is possible that once the majority of rogue pixels have been cleaned, the unsuccessfully cleaned pixels will be easier to clean. The unclean mask can be used as an input bad pixel mask (**`inRmask_File`**) to a subsequent run of IRSCLEAN on the cleaned image.

```
IDL> irsclean_mask,
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd_clean.fits',$
DataRange=[-24,54],$
inRmask_File='sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd_unclea
n_mask.fits'
```

If after a second pass, a pixel still does not clean, it is likely that there is not enough information to clean the pixel properly.

10 APPENDIX 2: Limiting Which Orders IRCLEAN_MASK Examines (`/getFmask, ORDERS=`)

Under **`getFmask=1`** (hierarchical iterative rogue finding method), you can force IRSCLEAN to detect rogues only in specific spectral orders using the **`ORDERS`** keyword in conjunction with **`/getFmask`**. You may find this useful if the combination of signal, noise, and background varies from order to order and you need to tune your rogue-finding to order-to-order variations. **Note that the procedure for detecting rogue pixels here is different than the standard approach.** The noise will be measured in each order independently , and rogue pixels will be detected using a thresholded multiple of the noise level in the order. [The default algorithm without setting **`ORDERS`** measures the noise level in either the interorder regions (**`droop.fits, rsc.fits, f2ap.fits, bcd.fits`**) or all orders together (**`coa2d.fits,bksub.fits`**) and searches for rogue pixels in the whole array at once.]

Set **`ORDERS`** equal to an IDL vector listing the order numbers to examine. Each order in the list will be analyzed separately to look for rogue pixels, and the discovered rogues will be combined into the found mask.

```
IDL> irsclean_mask,
'sampledata/r13349376/ch3/bcd/SPITZER_S3_13349376_0012_0000_8_bcd.fits',$
DataRange=[-24,54],/getFmask,ORDERS=[2,3]
```

We show the order numbers for IRS arrays in Figure 10.1. Note: the IRS peakup sub-arrays are designated short low "orders" 4 (Red Peakup) and 5 (Blue Peakup). The keyword **`/PeakUp`** must be set to find rogues in the peakup sub-arrays (see below), with or without using the **`ORDERS=`** functionality.

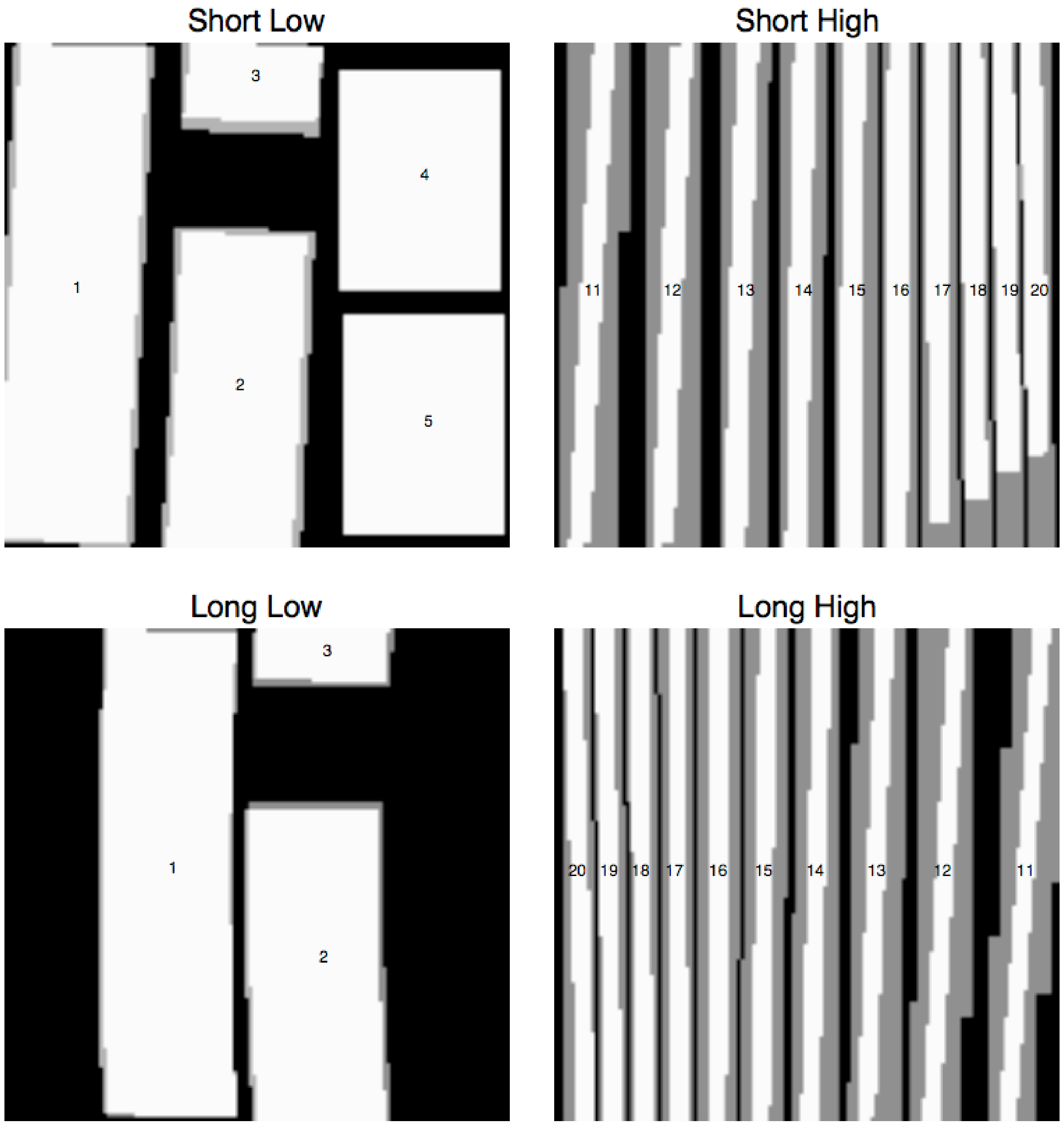


Figure 10.1 IRS Orders. We show the default order masks in white. Wide order masks accessible via keyword /wide_omask are shown in gray. Although not officially numbered in the IRS pipeline, the peakup sub-arrays are labeled Short-Low orders 4 (Red) and 5 (Blue) in IRSCLEAN.

11 APPENDIX 3: Finding Rogues in and/or Cleaning The Peakup Sub-arrays (/PeakUp)

If your input data file is from the Short Low array, the rogue-finding and cleaning methods ignore the peakup imaging sub-arrays and concentrates only on the spectral orders. You can override this default using the **/PeakUp** keyword. The peakup sub-arrays are designated short low “orders” 4 (Red Peakup) and 5 (Blue Peakup). If you set **/PeakUp**, the peakup apertures will be combined with the other three SL spectral apertures and rogues will be found and/or cleaned over the entire array. Under **getFmask=1**, if you set **/PeakUp** and designate one or both peakup “orders” using **ORDERS**, then the analysis is and/or cleaning will operate in turn on each peakup “order” selected (as well as any spectral orders chosen).

12 APPENDIX 4: Complete Listing of Command Options

We list below the full set of command line options, plus online help, for IRSCLEAN, extracted from the file **irsclean_mask.pro**.

IRSCLEAN_MASK Options

```
;+
; NAME: IRSCLEAN_MASK
;
;
;
; PURPOSE: Spitzer IRS rogue mask editing and cleaning software.
;
;
; CATEGORY: Image Processing
;
;
;
; CALLING SEQUENCE:  irsclean_mask [, dataFile] [,inRmask_File=inRmask_File]
;                                  [,/getFmask or getFmask=1 or 2]
;                                  [,AGGRESSIVE=aggressive]
;                                  [,NOISE_FLOOR=noise_floor]
;                                  [,ROGUE_thresh=rogue_thresh]
;                                  [,/ABS_VAL] [,/NEGATIVE_ONLY=NEGATIVE_ONLY]
;                                  [,/NAN] [,dataRange=dataRange]
;                                  [,Bmask_File=Bmask_File]
;                                  [,bMaskVal=bMaskVal] [,bMaskBits=bMaskBits]
;                                  [,orderMask_File=orderMask_File]
;                                  [,/wide_OMask] [,/noOrderMask]
;                                  [,/CampaignRMask]
;                                  [,outRmask_File=outRmask_File]
;                                  [,outClean_File=outClean_File]
;                                  [,/autoSave]
;                                  [,filesToClean=filesToClean] [,WinSize=WinSize]
;                                  [,Directory=Directory]
;                                  [,ORDERS=ORDERS] [,/peakUp]
;                                  [,/noStage2] [,/noMaskEdit]
;                                   [,maskFunction=maskFunction]
[,colorNames=colorNames]
;                                   [,/HELP] [,_EXTRA=extra]
;
; INPUTS:
;    All inputs are optional.  The program queries the user for
;    anything it needs.
;
;
;
; OPTIONAL INPUTS:   dataFile - File with IRS image data (droop, rsc, bcd,
;                                f2ap, bksub, or coa2d) to be used to produce
;                                the rogue pixel mask.  You will have
;                                the option to clean this file using
;                                the mask that you develop.   If not
;                                given, a file picker widget will
;                                appear. If the filename is a standard
;                                IRS pipeline name, then IRSCLEAN will
;                                search for associated bmask and
;                                uncertainty files.
;
;
; KEYWORD PARAMETERS:
;;;;;;;;   INPUT MASKS
;                   inRmask_File - File name(s) for the input rogue
;                                      pixel mask.  This mask will be
;                                      combined with other rogue masks
;                                      derived from getFmask
;                                      and /NAN, and will be editable
;                                      during the mask-editing stage.
;                                      If you do not set getFmask,
;                                      or CampaignRMask,
;                                      then you will be queried
;                                      for inRmask_File (you can cancel
;                                      out of this dialog and start
;                                       mask-editing with no mask).
;                                       Can be a list of file names.
;                          /CampaignRMask - Set this keyword to
;                                           automatically load the
;                                           campaign rogue mask
;                                           corresponding to your
;                                           data.  IRSCLEAN will
;                                           combine this with other
;                                           rogue pixel masks, if
;                                           keywords getFmask,
;                                           inRmask_File, or /NaN are set.
;                          Bmask_File   - The pipeline bmask, a 16-bit
;                                         image.  If no file is given, IRSCLEAN will
;                                         automatically search for the
;                                         bmask associated with the
;                                         input image file using the
;                                         standard SSC naming convention. During
;                                         mask-editing, bmask-ed pixels
;                                         (pixels with appropriate
;                                         bmask bits as given in
;                                   bmaskVal) are highlighted in
;                                   blue.  These pixels will not
;                                   be considered in rogue
;                                   finding, nor will they be
;                                   used in computing the cleaned
;                                   pixel values.  You can turn a
;                                   bmask-ed pixel into a rogue
;                                   pixel (and back) by clicking
;                                   on it during mask-editing.
;                         bMaskVal   - An integer representation of
;                                   the bmask bits for which a pixel is
;                                   removed from consideration.
;                                   Default is 28800 (bits
;                                   7,12-14).
;                                   These pixels will be
;                                   highlighted in blue during
;                                   mask-editing.  To avoid
;                                   considering the bmask, set
;                                   bMaskVal to zero (0).  If both bMaskVal
;                                   and bMaskBits are set, then bMaskVal has
;                                   precedence.
;                         bMaskBits - Explicit listing of
;                                   the bmask bits for which a pixel is
;                                   removed from consideration.
;                                   Default is [7,12,13,14].
7,12-14).
;                                   These pixels will be
;                                   highlighted in blue during
;                                   mask-editing.  To avoid
;                                   considering the bmask, set
;                                   bMaskVal to zero (0).
;                        /NaN   -   set this keyword to add NaN pixels to the rogue
mask
;                        orderMask_File - Name of file to use to
;                                    specify the spectral orders.  If not given
;                                    (and neither /wide_OMask or /noOrderMask
are set) then use
;                                    the default (narrow) order mask delivered
with IRSCLEAN.
;                       /wide_OMask -    Use the wide order mask delivered with
IRSCLEAN.
;
;                       /noOrderMask   - Set this keyword to define
;                                    the entire image as "Order 1" and ignore
the
;                                    need for an order mask.  Useful if you want
;                                    to create masks of pixels in between
;                                    orders,or construct masks using non-IRS
;                                    images.  Be warned that results of cleaning
may not be reliable.
;                         /peakUp -  Set this keyword to enable
;                                   mask-finding and cleaning of the IRS
;                                   peak-up arrays (only valid for Short Low
;                                   data).
```

```
;*******************************
;;;;;;   ROGUE-FINDING KEYWORDS
;                         getFmask -  Use a rogue-finding algorithm
;                                     to find rogue pixels in your
;                                     dataFile.  You can set this keyword
;                                     to 1 (or invoke /getFmask) to find rogues
;                                     using the standard algorithm used
;                                     in IRSCLEAN versions up to 1.9
;                                     (also called the "complex"
;                                     algorithm).
;                                     As of IRSCLEAN 2.0, you can also run an
;                                     alternative "basic" algorithm by
;                                     setting getFmask=2.
;                                       In summary set getFmask=
;                                        1 - Hierarchical Iterative method (same as in
;                                            IRSCLEAN versions up to
;                                            1.9). Sigma thresholding
;                                            using first two levels of
;                                            multi-median transform,
;                                            allowing clusters of
;                                            rogues.  (Equivalent to
;                                            /getFmask.)
;                                        2 - Double Unsharp-Mask
;                                            Flattening method.
;                                            Unsharp-mask of image,
;                                            followed by a second
;                                            unsharp-mask of each
;                                            column (flattening).
;                                            NOISE_FLOOR sets the
;                                            minimum nsigma baseline to be considered
signal, whereas ROGUE_THRESH sets the
;                                            unsharp-mask of Simple
;                                            sigma thresholding of
;                                            first level of
;                                            multi-median. transform, with
;                                            rejection of clusters of
;                                            three or more pixels.
;                           /ABS_VAL - Search for positive and negative
;                                      rogues.  Normally the image is
;                                      thresholded such that candidates
;                                      with values greater than a
;                                      certain multiple of sigma are
;                                      considered rogues.  This method
;                                      allows for a dual thresholding
;                                      of both the image and its
;                                      inverse.
;                          /NEGATIVE_ONLY - Search only for negative
;                                           rogues.  This is useful
;                                           when the background
;                                           observation had a lot of
;                                           rogues that didn't
;                                           cancel out in bg
;                                           subtraction.
;                         AGGRESSIVE - A number between -2.0 and +2.0
;                                      specifying the sigma threshold
;                                      above which a candidate pixel is
;                                      considered a rogue (see
;                                      online documentation
;                                      for details).  If getFmask=1,
;                                      this number also defines the
;                                      number of rogue-finding
;                                      iterations and whether or not
;                                      clusters of pixels are allowed
;                                      in the rogue mask.  Default is AGGRESSIVE=0.
;                         ROGUE_THRESH - The minimum sigma
;                                        for a rogue.  If getFmask=2,
;                                        then setting ROGUE_THRESH=0 and NOISE_FLOOR=0
;                                        will result in all pixels being detected as
rogues.
;                                        This keyword is interchangeable with
AGGRESSIVE:
;                                        ROGUE_THRESH = 2.0 + 2.0 * (1.0-AGGRESSIVE) >
0
;                         NOISE_FLOOR   - When getFmask=2 is set, the minimum sigma to
be
;                                        considered signal.
;                                        Alternately, this is the multiple of sigma in
the
;                                        small-scale component image of the multi-
median
;                                        transform below which a pixel value is
;                                        considered noise and removed from
consideration.
;                                        Default is NOISE_SIGMA=1.
;                         maskFunction - the name of a user-supplied
;                                        function to find rogue
;                                        pixels.  This function allows
;                                        as inputs the original fits
;                                        data, the order mask image,
;                                        and the keywords /ABS_VAL,
;                                        /NEGATIVE_ONLY, and
;                                        AGGRESSIVE.  The result of
;                                        this function will be combined
;                                        with any other rogue masks
;                                        input (inRmask_File) or
;                                        derived (via /getFmask).  See
;                                        documentation for details.
;                             orders    - List of orders to detect
;                                         rogues in, under /getFmask.  Setting this
;                                         keyword results in the independent
;                                         analysis of each of the orders given in
;                                         the list.  See documentation for the
;                                         order numbering and layout of each IRS
;                                         module.

;********************************
;;;;;;;;;;;;;;OUTPUT CONTROL
;
;                         outRmask_File - Output rogue mask file.  Set
;                                        this keyword to the name of the file for
;                                        storing the rogue mask
;                                        that has been created.  If not
;                                        given and autoSave is not set, you will be
queried.
;                        outClean_File - Output cleaned data file from
;                                        Stage 1.
;                                        Set this keyword to the name of the file for
;                                        storing the cleaned image that corresponds
to

;                                        the input image (dataFile).  If not given,
;                                        and /autoSave is not set, you will be
queried.
;                        filesToClean - List of files (*.fits) and/or
;                                       file lists (*.txt) to which the derived bad
;                                       pixel mask will be applied.  A string array.
;                                       List files have one FITS file per line.  The
;                                       clean data will  be saved in files with
;                                       similar names, but with "_clean" inserted
;                                       before ".fits".  If not given, and /noStage2
is

;                                       not set, you will be queried.

;;;;;;;;;;PROGRAM FLOW
;                         /noMaskEdit - Set this keyword to skip
;                                         interactive editing of the
;                                         rogue mask.
;                           /autoSave -  Set this keyword to
;                                       automatically save the rogue
;                                       mask and cleaned image from
;                                       Stage 1, using a best guess at
;                                       the file name (unless
;                                       outRmask_File and/or
;                                       outClean_File are set)
;                          /noStage2 - Set this keyword to skip
;                                        Stage 2 of the process (multiple file
;                                        cleaning using the Stage 1
;                                        mask)
;;;;;;;;;;MISCELLANEOUS
;                         Directory   - Set this keyword to the initial
;                                      working directory (will change if you move
;                                      to another directory in file picker).
;                                      Default is current IDL working directory
;                                      (usually the directory that you start IDL
;                                      in unless you have changed directories during
your session).
;                         dataRange   - set this keyword to the range
;                                      in data values over which to scale the
;                                      images for display (a 2-element vector).  If
;                                      not given or set to 0
;                                      (1-element scalar), the display
;                                      will be scaled to the min and
;                                      max of the image.  Setting
;                                      dataRange=[0,0] will allow you
;                                      to interactively choose the
;                                        display range prior to mask editing.
;                          WinSize    - This keyword (a 2-element
;                                        vector) sets the x,y size of the image
;                                        window (a small pad will be added for
;                                        display purposes).  Default is
;                                        [768,768].  Maximum is [2048,2048].
;                          colorNames - Two element string array giving
;                                        the names of the colors for the
;                                        overlay symbols drawn on rogue
;                                        and bmask pixels during the
;                                        mask editing stage.  Default is
;                                        ['red','blue'].  Available color names
;                                        can be found by typing
;                                        IDL> print,FSC_COLOR(/names)
;                                        Or to visually choose a color
;                                        name, type
;                                        IDL> print,PICKCOLORNAMES()
;
;                           /Help      - Print the irsclean_mask.pro documentation
header
;
; OUTPUTS:             All input arguments, if passed as named variables, will
;                      be returned with their current values as of the
;                      termination of the program.  In particular, file
;                      names if passed as variables that equal the
;                      null string (''), or the dataRange keyword if
;                      passed as a variable that equals [0,0], will
;                      return in the respective variables the actual
;                      values chosen during execution of the program.
;
;                      On exit, IRSCLEAN prints the command line that
;                      you could have typed to obtain the same result,
;                      incorporating all changes to input arguments and
;                      other choices made during your session.
;
;
; OPTIONAL OUTPUTS:  If set, Keyword DIRECTORY will be reset to the final working
directory.
;
;
; COMMON BLOCKS:
;                      IRSCLEAN_SHARE contains image display scaling information
derived from IRSCLEAN_WINDOWIMAGE.
;
;
; SIDE EFFECTS: This program uses interpolation to edit the values of "rogue" pixels
and their associated uncertainties.
;               It also resets the bmask for these pixels to zero.
;
;
; RESTRICTIONS: Currently runs only on 128x128 pixel images.
;
;
;
```

```
; PROCEDURE:
;
;
;
; EXAMPLE:
;
;
;
; MODIFICATION HISTORY:
;
;    Copyright (C) <2011>  <J. Ingalls/SSC>
;    This program is free software: you can redistribute it and/or modify
;    it under the terms of the GNU General Public License as published by
;    the Free Software Foundation, either version 3 of the License, or
;    any later version.
;
;    This program is distributed in the hope that it will be useful,
;    but WITHOUT ANY WARRANTY; without even the implied warranty of
;    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
;    GNU General Public License for more details.
;
;    You should have received a copy of the GNU General Public License
;    along with this program.  If not, see <http://www.gnu.org/licenses/>.
;
;-
```