A Guide to Reducing Near-IR Images

Kenneth Hinkle
NOAO

Including contributions from
Richard Joyce
NOAO

ABSTRACT

This document reviews the steps needed to reduce near-infrared imaging data. Data from the Gemini NIRI imager will be used as an example but the techniques are the same for all instruments of this type. The document starts with background on FITS and IRAF. Then a detailed discussion is presented on reductions using NOAO IRAF. The final chapter reviews the more advanced and sophisticated Gemini IRAF tools. Pipeline reductions are not discussed.

Keywords: IRAF, NIRI, near-infrared imaging, data processing

DRAFT 1.1 August 2017

Table of Contents

1. Introduction

This manual discusses the reduction of near-IR imaging data for the Gemini imager NIRI. Data from the Gemini spectrograph GNIRS used in imaging mode will be similar. Both of these instruments have a 1024 x 1024 InSb array detector at the focal plane. Typically J - H - K broadband filters are used with NIRI although a selection of narrow-band filters are available. Near-IR imagers are a standard observatory instrument and many instruments will have similar data to NIRI. In the course of writing this Guide we have taken material, sometime nearly (or perhaps exactly) word for word, from the Gemini web pages and several presentations on line that discuss how to reduce Gemini data. Material also has been drawn from the NOAO NEWFIRM, SQIID, and WHIRC manuals. The WHIRC manual by Dick Joyce was especially helpful. In many but not all cases a citation is made to these sources.

There are several differences between infrared and optical data. The cause stems from the contribution of background radiation in the near-infrared and the physics of near-IR versus optical detectors. In the infrared at wavelengths longer than about 1.6 microns sky can produce significant signal. In this manual by sky we mean the radiation received at the detector from the sky, the telescope, and non-cooled parts of the instrument. This background signal can be larger than the astronomical signal.

In addition to sky a major difference between optical and infrared observing stems from the detectors. Infrared arrays pixels are read independently instead of using charge transfer techniques that are used with CCDs. The read noise is many times higher in the IR than in the visible and there is measurable dark current for IR arrays. Since the pixels are read independently the dead (or hot) pixels are isolated features. The sky and detector blemishes are removed from infrared images by taking multiple images with the telescope moved a small distance between images. This technique is called nodding or dithering. A very useful discussion of infrared arrays can be found in an article by R. Joyce from 1992 called “Observing with Infrared Arrays” in Astronomical CCD observing and reduction techniques edited by S. B. Howell (ASP Conference Series, 23, 258).

2. Data Requirements

Assuming that you have Gemini data calibrations should have been included automatically in the Phase II plan. The calibrations include flats and darks. These reference observations typically consist of a minimum of 10 flats and 10 darks for each filter used. The flats should have been taken with the same filter as the science observation. The flats and darks should have the same exposure time but (obviously) the filter is irrelevant for the dark.

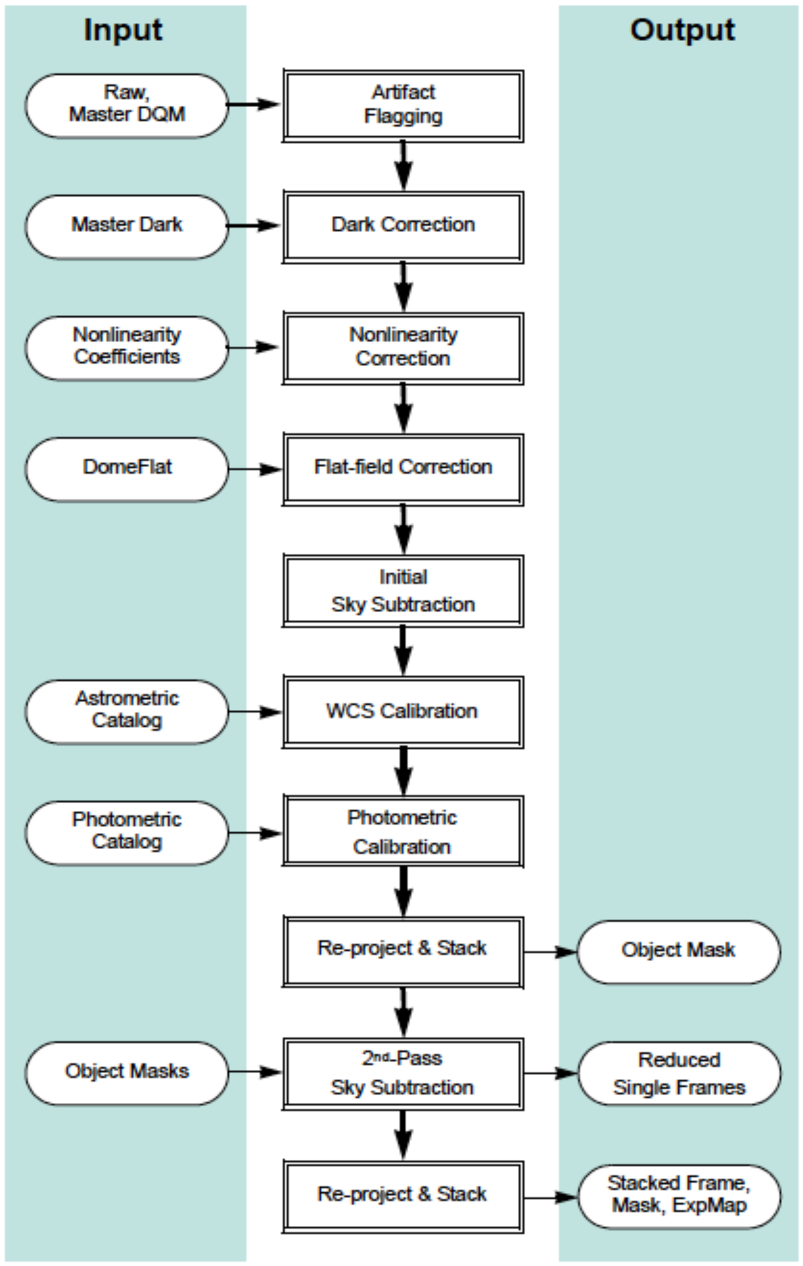


Figure 1 – Flow of data in reducing typical near-IR imaging data. The figure is taken from the NOAO Data Handbook Version 1.1 May 2009. The process shown here is for the NEWFIRM infrared array and is more complex than NIRI reductions but is illustrative.

Darks are also required with the same integration time as the science target. The same Fowler setting should be used as for the science observation. Joyce (1992) discusses Fowler settings (low noise reads = LNR). The science observations should consist of dithered images centered on the source field. An optional set of dithered

observations of off-source sky can also be taken. Figure 1 shows a typical reduction path. We will discuss this step-by-step providing simple IRAF instructions.
Raw files stored by the Gemini Data Handling System (DHS) are written with file names of the form N20080101S0001.fits. The date is the UT date at the beginning of the night. Because the DHS is writing data from multiple instruments, including the wavefront sensors, the NIRI file number sequence is frequently interrupted. Missing file numbers do not necessarily indicate missing NIRI data.

The sample data that will be discussed here is of a M31 bulge field observed during SV time 2003B. The data can be downloaded from the
NIRI data reduction example presented by Olsen and Stephans at the 2010 Gemini Data Workshop.

3. Multi- and Single-Extension FITS

Many Gemini instruments employ multiple science arrays. For instance, the most popular instrument, GMOS, uses three science arrays. FITS images consist of data and meta-data sections (Figure below). In order to describe observations with multiple images and associated meta-data Gemini employs multi-extension FITS (MEF). Of course not all Gemini instruments have multiple science arrays but for consistency Gemini distributes all data using multi-extension FITS. NIRI has a single detector and the raw NIRI images are MEF files with a single unnamed extension [1] that contains the image data (Example 1 below). Most of the important meta-data is in the primary header unit (PHU, extension [0]).

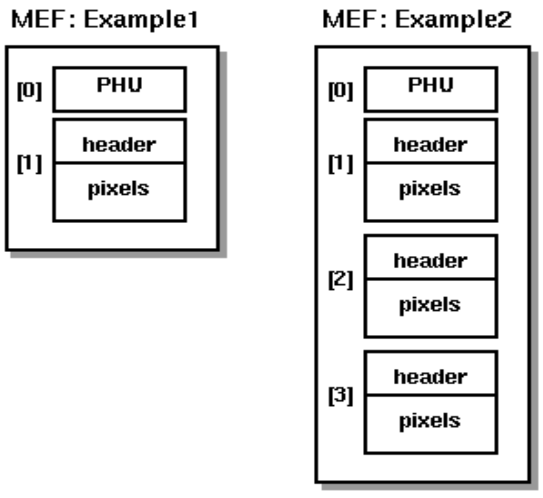


Figure 2 – A schematic showing how the Multi-Extension Fits (MEF) files are organized. (Figure taken from Gemini web pages). PHU=primary header unit.

Additional information on the naming of the files and the contents can be found in the NIRI web pages at
http://www.gemini.edu/sciops/instruments/niri/data-format-and-reduction/data-primer

In this manual we will do some exercises with NOAO IRAF that uses single-extension Fits (SEF).

4. IRAF

IRAF is an acronym for Image Reduction and Analysis Facility. IRAF provides a wide selection of data reduction tools that can be run through the command language (cl) interface. IRAF is free, widely distributed and contains an extraordinary selection of useful tools for reducing astronomical data. These and the fact that the majority of the astronomical community use IRAF are reasons to use IRAF. However, for many users IRAF is not an intuitive language and the learning curve can at times be steep. At times IRAF can be unforgiving. Beginners may wish to look at "The Beginner's Guide to Using IRAF" by Jeannette Barnes. Another useful reference is Peter Shames' and Douglas Tody's "A User's Introduction to the IRAF Command Language". Both manuals can be downloaded from the NOAO web site (search for them using the Goggle Custom Search box on the NOAO home page). This documentation is old and some details are out-of-date but it continues to have worth. Minimal knowledge of IRAF is assumed in this document.

IRAF has some features that seem strange on 21st century computers. These were built to match the environment of computers in the 1980s. The commands are in packages that have to be loaded. We will discuss this in more detail as we discuss various commands. Commands may be referred to by an abbreviated command name as long as the command name remains unique. If the name is ambiguous IRAF will tell you. The command to exit IRAF is logout.

Some UNIX commands work inside the IRAF command language (cl) and some do not. In particular the UNIX commands for moving and deleting files, rm, mv, and cp, do not work. This is because IRAF was written for machines with little memory and the images originally were not FITS. You can use the IRAF words del for rm, rename for mv, and copy for cp or you can use a ! to send the command to the UNIX shell.

To display images there are a number of routines compatible with IRAF. The routine available to the author is ds9. In the IRAF window type

```
! ds9 &
```

The IRAF command to show an image in the ds9 window is display filename. To measure features in the image use imexam. When using imexam a question mark in the ds9 window will generate help. Type q in the ds9 window to exit imexam.

IRAF commands are organized by packages. If a word is not present when you try to execute it you will have to load the package. To find the package name use help, for instance 'help imcopy'. At the top of the help page the package name appears, in this case images.imutil. Typing images.imutil in IRAF will load that package. Help also produces a listing of the package or parameters for each word, a description of the function, and examples. "?" lists the contents of the package. There are a very large number of tasks and the appendices to Barnes' Beginner's Guide listing the contents of IRAF packages are very helpful in sorting out the correct package to load.

Each IRAF command comes with a parameter set. For some commands the parameter sets will have to be edited. This is done with the command epar, as in epar imcopy. The epar editor is clunky and **not** vi. Esc-? produces a help screen. Use the up/down arrows to move from line to line. If you make a typing mistake the easiest thing to do is probable to down arrow to the next line, up arrow to the line that is messed up and re-type. Cntr-C exits without updating/changing the saved content. Cntr-D exits saving your input.

Standard NOAO IRAF requires images in FITS. As discussed above Gemini data is in Multi-extension FITS (MEF). MEF requires Gemini IRAF. With NOAO IRAF you can take a look at the extensions to your data using fxheader, a routine in the fitsutil package. The fitsutil package was written to handle MEF files. fxheader lists the type of the file in a multi-extension fits file and the dimension of the image. In the reduction process the images will be trimmed and it can be useful to know the dimensions.

In the example we will use standard IRAF to reduce NIRI data. To do this MEF files must be converted to standard FITS images using some IRAF copy command. For MEF images will be necessary to specify the MEF extension in square backets (e.g. N20031118S0753.fits[1]). A better command may be fxsplit that divides MEF files into separate files. A number of specific corrections to the images require routines found in Gemini IRAF or specialized Python routines. We will return to these Gemini IRAF routines in the last section of this manual.

A few syntactical features of IRAF are labor saving and will be used in examples in the following sections.

a) Lists of files

@filename is a text file (in the example with the name filename) that contains a list. @filename can be used as an argument in place of a file name. This is useful in processing lists of files rather than single files.

b) Pattern matching

As in unix * can be used to match zero or more characters. [...] will match any character in the class specified, like [a-z]. To match characters not in the class use an ^, like [^a-z]. A ? matches a single character.

% is a substring substitution operator. If the sequence %a%b% is used in a file name the sting "a" is replaced with "b" in the generated file name. The example used in the User's Introduction to IRAF Command Language is
*files *%%_o%.com*
produces a list of files with an o added to the end of the filename. There is also a sting concatenation operator // that would do the same thing using
*files *.com//*_o

c) Executing commands in background and in the shell

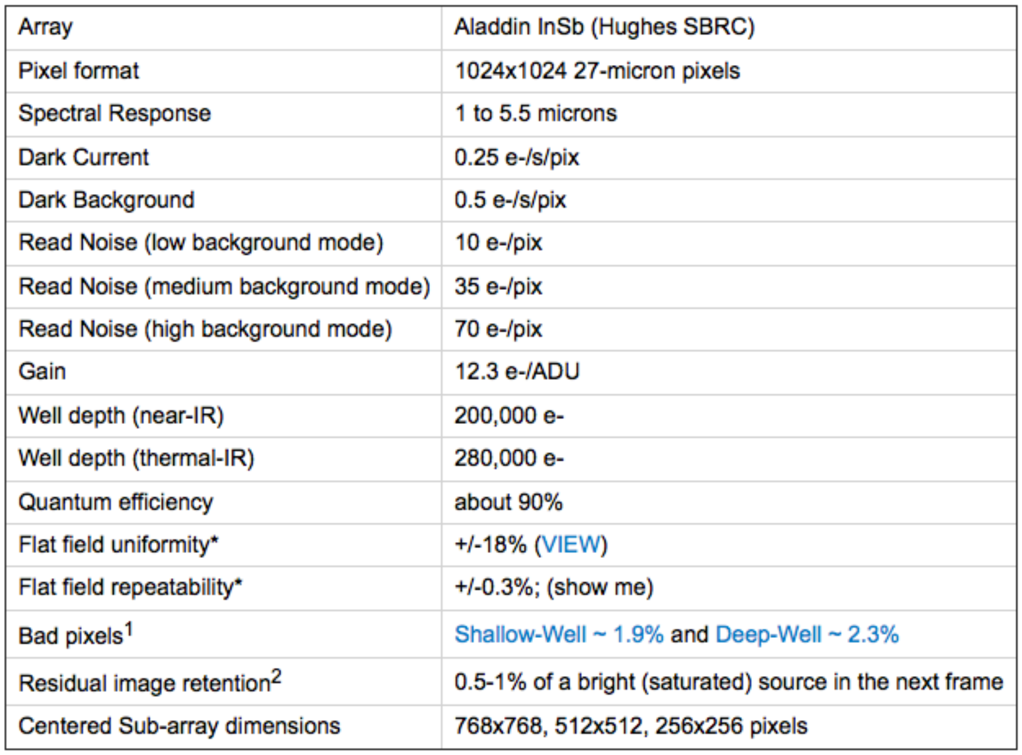
An & allows a command to run in back ground. An ! executes the command in the unix shell.

ADVICE TO THE IRAF BEGINNER: It can be difficult to undo IRAF commands. Some commands generate files in subdirectories and change information in the image header. Make a copy of all the data to a different subdirectory before starting any reductions.

5. NIRI

The Gemini web pages provide many details as well as references to articles by the builders. We have extracted and edited this material in the subsections below. NIRI has three imaging modes, f/6, f/14, and f/32 with pixel scales 0.1171, 0.0499, and 0.021 seconds of arc per pixel, respectively. f/32 is largely used with the Altair AO system. For f/32 the pixel scale changes slightly depending on the use of the Altair field lens (in=0.0219, out=0.0214 arc seconds per pixel). Figure 3 is a table giving the NIRI detector characteristics.

There is slight barrel distortion in the f/32 mode. A mathematical description of the distortion can be found at
https://www.gemini.edu/sciops/instruments/niri/imaging?q=node/10058
Over a distance of 500 pixels with distortion images appear about 3 pixels closer than in a distortion free image. For those with an IDL license Gemini provides on the above web site an IDL program for undertaking a correction.



Array	Aladdin InSb (Hughes SBRC)
Pixel format	1024x1024 27-micron pixels
Spectral Response	1 to 5.5 microns
Dark Current	0.25 e-/s/pix
Dark Background	0.5 e-/s/pix
Read Noise (low background mode)	10 e-/pix
Read Noise (medium background mode)	35 e-/pix
Read Noise (high background mode)	70 e-/pix
Gain	12.3 e-/ADU
Well depth (near-IR)	200,000 e-
Well depth (thermal-IR)	280,000 e-
Quantum efficiency	about 90%
Flat field uniformity*	+/-18% (VIEW)
Flat field repeatability*	+/-0.3%; (show me)
Bad pixels[1]	Shallow-Well ~ 1.9% and Deep-Well ~ 2.3%
Residual image retention[2]	0.5-1% of a bright (saturated) source in the next frame
Centered Sub-array dimensions	768x768, 512x512, 256x256 pixels

Figure 3. Table of NIRI detector characteristics from https://www.gemini.edu/sciops/instruments/niri/imaging/detector-array

5.1 Read Modes

Three read modes are provided. These correspond to changes in the array bias and the number of Fowler samples. For high background environments the array is read once at the beginning and once at the end of the exposure and the difference is recorded. In medium background situations, typically f/6 broad band JHK imaging, the beginning and end reads are digitally averaged 16 times. In low-background observations the array is read 16 times at the beginning and the end of the exposure with the digital averaging also taking place during each read.

5.2 Detector Saturation

The saturation level for a single coadd, low-noise read pair is about 16,000 ADU. The array works by doing a reset-read-read. So saturated sources can appear to have low count levels when the array begins to saturate in the time between the reset and the first read. Saturated stars often show a central hole that sometimes can be negative in the core.

5.3 Linearity

The array performs closest to linearity, with deviations from linearity of a few percent, at low- to medium-flux levels and with exposures longer than a few seconds. When the exposures are very short (<1s) or when the well reaches approximately >70% of the full well the non-linearity becomes severe.

5.4 First Frame

After changing the detector configuration (well-depth, read mode, exposure time, etc.) the first exposure of each new sequence will show poor background subtraction and should not be used. If the exposures were short it is possible the first exposure is usable.

5.5 Quadrant Boundaries

Quadrant boundaries are sometimes visible due to a mismatch in the background level of exposures that are not background dominated.

5.6 Crosstalk

Column crosstalk exists in the bottom right quadrant (rows 1 to 512) between columns 534 and 535. Stars falling on these columns appear slightly elongated or boxy.

5.7 Image persistence

Image persistence is a characteristic of InSb arrays. The NIRI web sight says persistence is `small' in the NIRI array.   The web site says that saturated or nearly saturated images will leave a ghost in one or two subsequent frames. The persistence of a very bright source is typically 0.5 to 1% in the next frame and less than 0.1% by the third frame.

6. Standard IRAF Near-IR Image Reduction

In the following sections we will use standard IRAF tools to demonstrate the basic steps involved in converting the raw images into scientifically useful data.

6.1 Array cosmetics

The raw images contain various artifacts. Figure 4 shows a raw flat. There are a number of obvious features. The diagonal line in the lower right is a crack in the array.

The circle of bad pixels with the vertical line below is the result of a bright pixel having been removed in the manufacturing process (a photo-emitting defect or PED). Since the PEDs both draw current and emit light they are removed as a step in the array manufacturing. They are 'cauterized' with a laser with the resulting signature circle of reduced sensitivity and sometimes a bright annulus. The same is true of the short vertical line near the center of the image. In the NIRI array the adjoining pixels in the column are dead.

The arc pattern is a "tree ring" pattern. Tree rings are a ubiquitous feature of InSb arrays resulting from lapping during manufacturing. There are several groups of bad pixels. In the upper left there is a feature that looks like a thumbprint. There are some other region with bad pixels and random bad pixels across the array. The array has uneven response with large-scale brighter areas. The quadrants have different sensitivities. There is a frame (either bright or dark and depending on f number) and for f/32 the corners are vignetted.

The tree rings, thumbprint, and bright or vignetted areas will be removed by the flat field. Combining the dithered observations will remove the bad pixels. **In addition to these features for NIRI data the first frame in the series (flat, dark or data) is generally bad and should be rejected.**

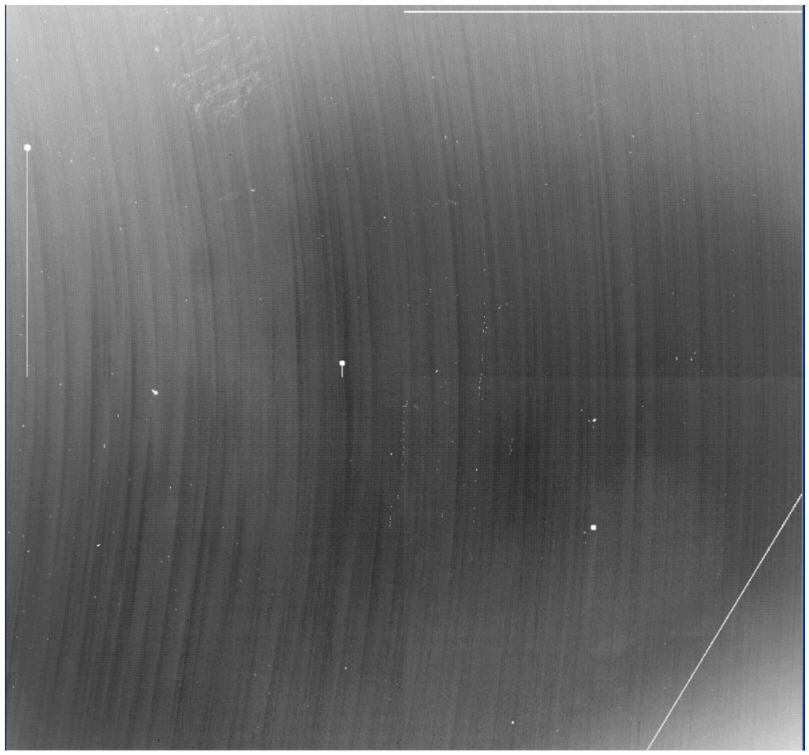


Figure 4 - A typical NIRI flat showing the cosmetic problems of the array. This image is from the Gemini web page http://www.gemini.edu/sciops/instruments/niri/NIRIJFlat.gif

6.2 Trimming

For many instruments including NIRI dead or masked rows and columns are present at the edge of the image. For NIRI there is a strip of bad pixels on the right and above the image. The raw NIRI images are 1024 x 1024. Use imexam to examine the edges of your images. I find a bad strip no broader than 20 pixels. In this example we will remove these by copying a sub-array by specifying the rows and columns. For the purposes of this example we will also copy from multi-extension fits (MEF) to single extension fits (SEF). For a single image an example of the copy is

```
imcopy  N20031118S0804.fits[1][1:1004,1:1004] flats_804.fits
```

Note again, we are not copying the MEF header in this example. As noted above fxsplit can be used to cleanly separate the MEF header and data. Then, if you are so inclined, an MEF file can be remade later using fxcopy.

All the data (science, sky, flats, darks) must be trimmed to the same size.

If you have a good memory, it is possible for some of the data to skip copying the data and simply specify the subarray when combining the images (see below). IRAF will produce an error message if you try to use imarith on arrays of different size.

6.3 Scaling - LNRS and coadds

Commands that either Fowler sample or coadd data can result in images that are scaled by a multiplicative factor. Whether or not this is a problem depends on how the electronics are configured. For NIRI data the web pages warn that data taken prior to November 2001 using Fowler samples (multiple non-destructive reads = low noise reads = LNRS) must be divided by the number of reads.

All NIRI data need to be divided by the number of coadds. The number of coadds can be found in the image header using the l+ extension, for instance

```
imhead N20031118S0804.fits[0] l+
```

The images can be rescaled with imarith, for instance for 2 coadds

```
Imarith  mydata.fits / 2. mydata_rescaled.fits
```

IRAF provides a routine, hselect, that lists the value of a header item.

A series of commands to correct your entire data set would be

```
hselect  @listofspectra.txt coadds yes > coadd_list.txt
```

that produces a value for all the coadds in the listofspectra followed by

```
imar @listofspectra.txt /  @coadd_list.txt @names_for_corrected_spectra.txt
```

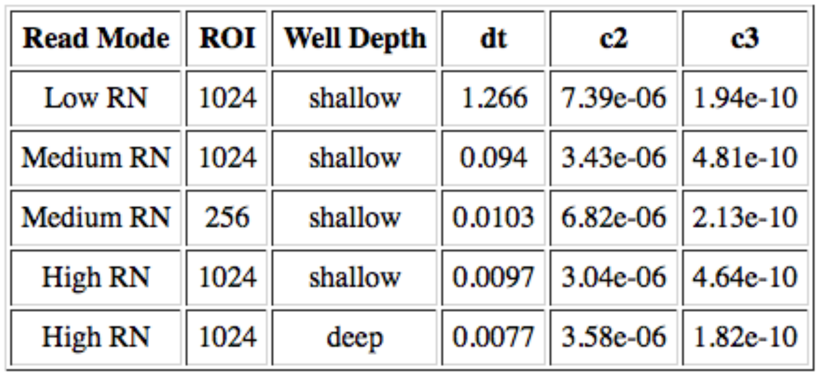
that produces corrected spectra.

6.4 Linearity correction

The gain in infrared arrays depends on the number of electrons in the wells. If the observation was planned so the wells were less than half filed the non-linearity should be very small. If the wells are near filled a correction is necessary. A linearity correction can be derived from flats at a variety of exposure times. Note that this must be done for the same bias, i.e. read mode, that was used for the data. As a Gemini user you will not have access to data of this type but the NIRI instrument team through the NIRI web pages provides information on the non-linearity correction. If a non-linearity correction is going to be applied it must be performed on the raw data prior to any other corrections but after scaling for Fowler sampling or coadds.

The information on non-linearity for the NIRI detector comes from work done by N. Stephens and discussed on the Gemini web page
http://staff.gemini.edu/~astephens/niri/nirlin/
A Python routine called NIRLIN is available that uses a cubic fit to correct the linearity. The table below (Figure 5) gives the fit parameters.

There is an IRAF task called irlincor that will apply a linearity correction. For the current example the exposures are not deep and we will ignore this correction.
I have not investigated if the coefficients in Figure 5 are in fact the correct ones for irlincor. The magnitude of the coefficients is similar to those found by Joyce for WHIRC (see WHIRC data reduction manual). To experiment with this it will be necessary to enter the coefficients into the parameter set for irlincor. This is done with epar irlincor.



Read Mode	ROI	Well Depth	dt	c2	c3
Low RN	1024	shallow	1.266	7.39e-06	1.94e-10
Medium RN	1024	shallow	0.094	3.43e-06	4.81e-10
Medium RN	256	shallow	0.0103	6.82e-06	2.13e-10
High RN	1024	shallow	0.0097	3.04e-06	4.64e-10
High RN	1024	deep	0.0077	3.58e-06	1.82e-10

Figure 5 – From http://staff.gemini.edu/~astephens/niri/nirlin/

6.5 Artifact Removal -- Bad pixels and pattern noise

Infrared images contain a number of different types of artifacts. Removing these will improve the overall quality of the final science image but also artifacts will be largely cancelled by the flats and sky corrections.

6.5.1 Dead Pixels

Since bad pixels are not light sensitive a bad pixel map can be made by comparing flats of different signal levels. A suitable set of flats is not provided with the sample

Gemini data set being used in this example. The process is to divide two flatfield images taken with the same filter, bias, and integration time but that observed different brightness flats. Twilight sky flats would be appropriate. Then use imreplace to replace the pixels that deviate from the typical value. With imreplace a file can be produced with bad pixels at a value of 1 and good pixels at a value of 0. This file can be used with the IRAF routine fixpix to remove the dead pixels. Note that fixpix writes over the input images so it is a good plan to make a copy of the raw data before running fixpix. In addition to a image of bad pixels fixpix will also accept a special bad pixel file that specifies the bad regions. Each line in the file has limits identified by x1, x2, y1, y2. Using a file of this type allows the size of the bad regions to be controlled and is especially helpful with the PED areas.

Bad pixels are discussed on the Gemini NIRI web site at http://www.gemini.edu/sciops/instruments/niri/data-format-and-reduction/data-primer but a bad pixel map is not provided. An image of the bad pixel map is provided and we show that in Figure 6.

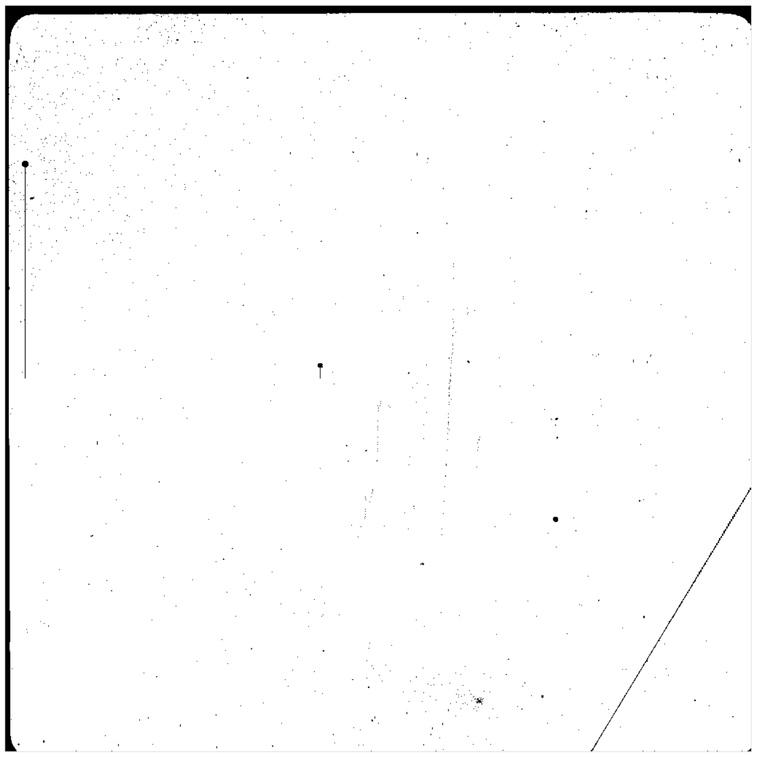


Figure 6 – A typical bad pixel map for the NIRI detector. The frame edges are very obvious in this image. These should have already by trimmed off the image.

Many poor performing pixels can be corrected by the flat fielding and the dithering technique removes most of the individual bad pixels. While not ideal we will forge on for now ignoring the bad pixels and see how things turn out.

6.5.2 PEDS

As noted above the NIRI array contains several photo-emitting defects (PEDs discussed above). The PEDs encompass regions are not completely dead so the technique of generating a bad pixel image by the flatfield ratio technique does not encompass the entire PED. More can be removed by adding the region to a bad pixel file.

6.5.3 Stripping

Some NIRI data has problems with stripping. Gemini does provide a routine CLEANIR that is designed to remove the vertical striping, horizontal banding, and quadrant offsets that plague some NIRI data. The routine is described at http://www.gemini.edu/sciops/instruments/niri/data-format-and-redution/cleanir
Fortunately striping is confined to a subset of NIRI data the test data does not have this problem.

6.6 Flats-off (darks) and flat-on exposures

The naming and numbering system for Gemini data gives no indication of the contents of the files. The header contains fields called OBJECT and OBSTYPE that will identify the observation. You can see the entire header for each file with the command

```
imhead filename.fits[0] l+
```

imhead without the l+ just lists the object. For the longer header option the header will scroll off the screen so it may be best to store the header to a file (follow the command above with >file_name.txt). We have previously introduced the useful IRAF tool, hselect. hselect lists the content of specific header words for multiple files. Typically this would look like

```
hselect filename*.fits[0]
```

where the * is a wild card for the file number. hselect will prompt for the header word. When asked about the boolean expression the answer should be 'yes'. Another option with MEF data is fxheader, also discussed above.

The goal is to make a flat field image. A set of observations should have been taken with equal exposure time with the flat field lamp on and off. Sometimes the flats-off exposures are called darks but if there is thermal emission at the wavelength of the observation these exposures will contain signal.

After the flats-on and flats-off images have been identified you should make up list of these files. A convenient tool is

```
files *.fits >flats_list.txt.
```

Lists of flats-off and flats-off files can be made by copying the .txt file and editing with vi. New users of IRAF should note that if you want to use the unix command cp you need to put an IRAF escape character (!) before the cp. IRAF provides the word copy that does the same task. Similarly unix mv is IRAF rename. The vi editor is recognized by IRAF.

An interesting check on the images is to do image statistics on the individual images. This can be done with the command imstat.

```
imstat @flats-off-list.txt
```

The @ sign tells IRAF to use the contents of the file, not the file name, as the input to command. We will use this trick repeatedly below.

With NIRI the first image of any sequence is not good. This should show up in the stats and you should remove this image from the list. In the case of the example data the bad images have already been removed. The images should all have the same number of pixels and the flats-on set and the flats-off set should have similar mean values, standard deviations, etc. This being checked we can combine the images into single flats-off and flats-on images by using

```
imcombine
```

My preference with imcombine is to enter the input and output file information by editing the imcombine parameters. The command is

```
epar imcombine
```

In this example I have used @flat_off_list.txt for the input parameter and flat_off_comb.fits for the input parameter. Imcombine will propt for these two parameters so this is not necessary. However, some of the other parameters need to be checked.

```
combine=median
statsec=[200:800,200:800]     Feel free to play around with this.
scale=median     (for flats,  scale=none for flat-off)
```

To exit epar without saving changes (if you are fighting with the funky editor) ^c.
To exit and save changes ^d.

The dark is subtracted from the flat by using imarith

```
imar flat_on_comb.fits – flat_off_comb.fits flat_dark.fits
```

I recommend normalizing the flat minus dark image. Do this by using imstat and selecting a fairly large sub-array region, perhaps the 200:800 region as above. Then divide the image by this value.

```
imstat flat_dark.fits[200:800,200:800]    my example gives a mean of 4331, so
imar flat_dark.fits / 4331. Flat_dark_norm.fits
```

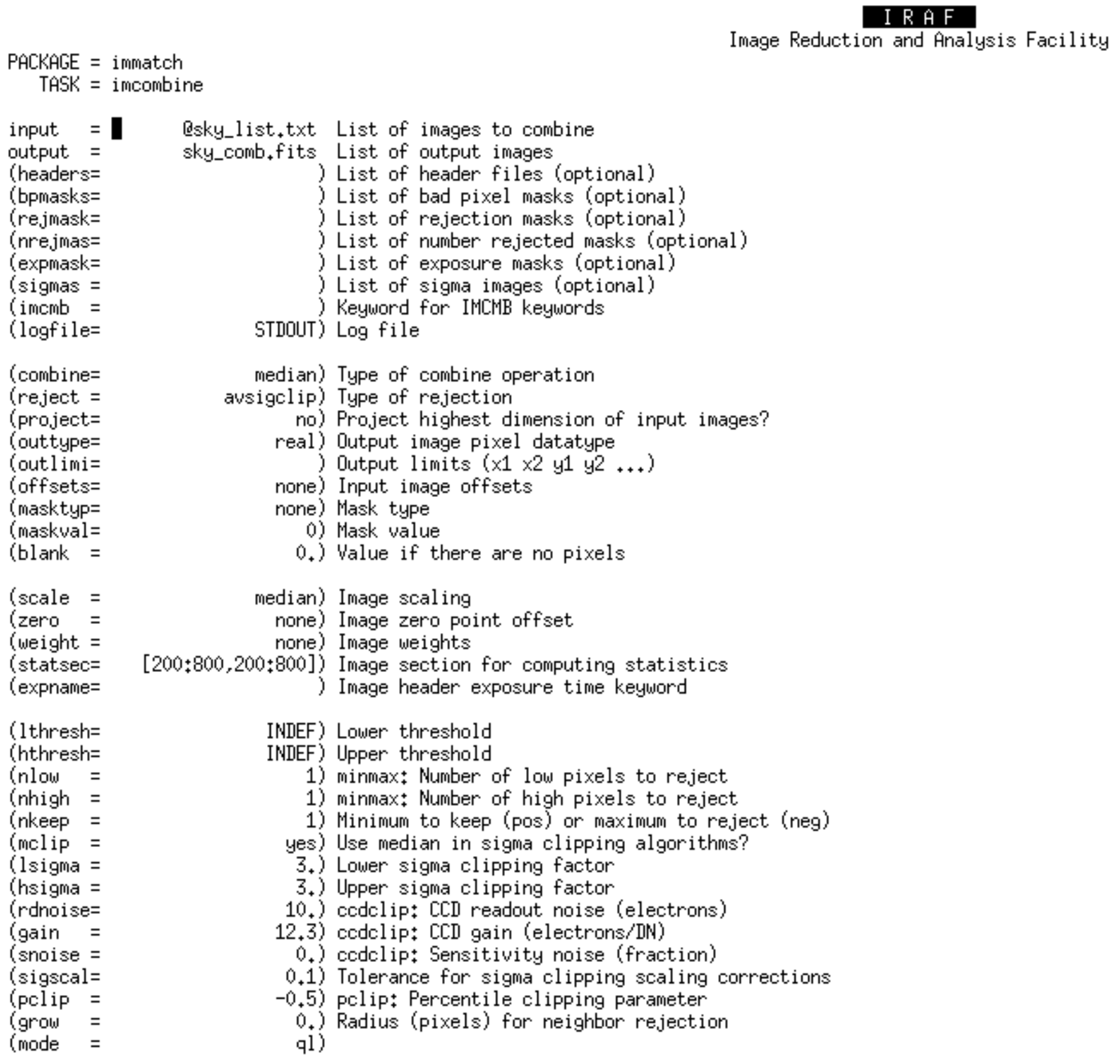
Next use imreplace to remove the very small or negative numbers from this image. A replacement value of 1.0 and an upper limit of 0.05 seem good. Keep the lower level at INDEF. Use epar imreplace to set these then type imreplace.

You should look at the flat_dark_norm file using displ.

6.7 Sky

Create a sky image by using imcombine on all of the images in each filter on each target field using median filtering (scale=median, see Figure 7). In the infrared we expect that the sky level will be changing, so scale the image using the median value of a large sub-array such as [200:800,200:800]. Use displ to image the result. Ideally, all the stars should have been removed from the sky image. If the science observation was of a very crowded fields or contained an extended source the target will not completely average out. In these cases dedicated sky observations should have been taken by nodding off target.

It is important to visually inspect the sky image.



```
                                                                       I R A F
                                                         Image Reduction and Analysis Facility
PACKAGE = immatch
   TASK = imcombine

input   =        @sky_list.txt  List of images to combine
output  =        sky_comb.fits  List of output images
(headers=                    ) List of header files (optional)
(bpmasks=                    ) List of bad pixel masks (optional)
(rejmask=                    ) List of rejection masks (optional)
(nrejmas=                    ) List of number rejected masks (optional)
(expmask=                    ) List of exposure masks (optional)
(sigmas =                    ) List of sigma images (optional)
(imcmb  =                    ) Keyword for IMCMB keywords
(logfile=              STDOUT) Log file

(combine=              median) Type of combine operation
(reject =            avsigclip) Type of rejection
(project=                  no) Project highest dimension of input images?
(outtype=                real) Output image pixel datatype
(outlimi=                    ) Output limits (x1 x2 y1 y2 ...)
(offsets=                none) Input image offsets
(masktyp=                none) Mask type
(maskval=                   0) Mask value
(blank  =                  0.) Value if there are no pixels

(scale  =              median) Image scaling
(zero   =                none) Image zero point offset
(weight =                none) Image weights
(statsec=    [200:800,200:800]) Image section for computing statistics
(expname=                    ) Image header exposure time keyword

(lthresh=               INDEF) Lower threshold
(hthresh=               INDEF) Upper threshold
(nlow   =                   1) minmax: Number of low pixels to reject
(nhigh  =                   1) minmax: Number of high pixels to reject
(nkeep  =                   1) Minimum to keep (pos) or maximum to reject (neg)
(mclip  =                 yes) Use median in sigma clipping algorithms?
(lsigma =                  3.) Lower sigma clipping factor
(hsigma =                  3.) Upper sigma clipping factor
(rdnoise=                 10.) ccdclip: CCD readout noise (electrons)
(gain   =                12.3) ccdclip: CCD gain (electrons/DN)
(snoise =                  0.) ccdclip: Sensitivity noise (fraction)
(sigscal=                 0.1) Tolerance for sigma clipping scaling corrections
(pclip  =                -0.5) pclip: Percentile clipping parameter
(grow   =                  0.) Radius (pixels) for neighbor rejection
(mode   =                  ql)
```

Figure 7 – epar imcombine set up for sky frames. Note that the parameter combine is set to median to smooth the stars from the dithered images. If dark images are being processed the scale parameter should be 'none'.

6.8 Science Images

Data sets typically consist of several images of the field with different telescope pointings. Depending on the sky stability, the sky level may be somewhat different in each of the images. We already created a sky image (above).

To reduce science images, next generate sky-subtracted images by subtracting the sky image from each of the original images. Remember that for the example that started above the image are SEF so it will be necessary to copy the MEF to SEF.

Also remember for NIRI to divide by the number of coadds before subtracting the sky. If the sky level had changed during the observation series, some of the sky levels may be positive, some negative. This gets fixed below.

Here are the steps I used:

```
imcopy N20031118S0135.fits[1][1:1004,1:1004] N135_trimmed.fits
imcopy N20031118S0136.fits[1][1:1004,1:1004] N136_trimmed.fits
imcopy N20031118S0137.fits[1][1:1004,1:1004] N137_trimmed.fits
imcopy N20031118S0138.fits[1][1:1004,1:1004] N138_trimmed.fits
imcopy N20031118S0139.fits[1][1:1004,1:1004] N139_trimmed.fits
imcopy N20031118S0243.fits[1][1:1004,1:1004] N243_trimmed.fits
imcopy N20031118S0244.fits[1][1:1004,1:1004] N244_trimmed.fits
imcopy N20031118S0245.fits[1][1:1004,1:1004] N245_trimmed.fits
imcopy N20031118S0246.fits[1][1:1004,1:1004] N246_trimmed.fits
imcopy N20031118S0247.fits[1][1:1004,1:1004] N247_trimmed.fits
files *_trimmed.fits >trimmed_list.txt   #made up a list of the trimmed files
copy trimmed_list.txt trimmed_div_list.txt
vi trimmed_div_list.txt   #made up a list of files called trimmed_div
imar @trimmed_list.txt / 2. @trimmed_div_list.txt   #divided by the number of coadds
copy trimmed_div_list.txt trimmed_div_skysub_list.txt
vi trimmed_div_skysub_list.txt  #made up a list of files ending in sky
imar @trimmed_div_list.txt - sky_comb.fits @trimmed_div_skysub_list.txt #subtract sky
```

Divide each sky-subtracted image by the normalized flat minus dark for that filter. This should result in images in which the (non-zero) sky level is uniform across the image. For most purposes it is best to subtract the non-zero base line (sky) offset from the images at this point. If you are doing aperture photometry, the sky level should be automatically subtracted during that process. Here is a procedure for removing the sky.

Use epar to set the imstat task so that the median or midpoint value or mean or what ever you want to take for the base line value is one of the output columns. For this example we assume that the median is column 4 of the output from imstat.

```
imstat @inlist.txt > statlist.txt    #runs imstats and puts results in file statlist
fields statlist.txt 4 > sslist.txt   #writes column 4  to list sslist
imar @inlist.txt - @sslist.txt @outlist.txt    #subtracts the mode from each image
```

6.9 Combining Images

At this point in the reduction the results can be analyzed in each image separately or in a combined image. For large mosaics distortion correction will be needed to ensure good overlap of the stars in different images. Most infrared observers will want to combine the dithered images into a single image.

It will come as not surprise to the careful reader that there are numerous ways to approach aligning and combining the images into a mosaic with IRAF. Sets of standard IRAF tasks are presented in the noao.imred.irred IRAF package as well as in the images.immatch IRAF package.

If the fields to be reduced have stars in common it is possible to use some routines written by M. Merrill for the SQIID infrared imager. These routines are part of the upsqiid package. SQIID was a simultaneous multi-color (either three or four color) imager but the routines work for single color imaging. A description of the upsqiid package, as well as instructions for downloading the package into IRAF, can be found at

http://www.noao.edu/kpno/sqiid

After downloading the various .cl scripts it will be necessary to tell IRAF that these exist. The command for this is task as in

```
task xyget = xyget.cl
```

xyget uses some other words and you will also need nircombine so here is the list of the scripts that need tasking:

xyget.cl, expandnim.cl, locate.cl, closure.cl, nircombine.cl

For reasons known only to IRAF gurus my IRAF package knows about the upsqiid package and will list all the routines if I type a ?. However, the individual routines do not have parameter sets. To fix this problem I downloaded the .cl and used the IRAF command redefine. Note that if you use task or redefine the .cl script must exist in the same directory with the data.

To use the upsqiid scripts to combine images there must exist an alignment star common to all of the image fields. This limits use to fields that occupy no more than about 50% more than the un-dithered field of view of NIRI. Extended fields (maps) require a different tool set. The noao.imred.irred package is one choice for producing maps but still requires stars in common in adjacent frames.

Single star photometry does not require the correction of optical distortion of the field.

6.10 Distortion Correction

NIRI used at f/32 has radial barrel distortion. According to the Gemini web page https://www.gemini.edu/node/10058 the distortion moves images 3 pixels closer to the center at 500 pixels. The distortion scales with square of the radial distance so will have negligible impact close to the center of the image. Programs utilizing deep imaging of a small field using small dither amplitudes likely do not require correction. f/32 is typically used with the Altair AO system so the offset and field used are probably both small. Imaging over a more extended field at f/32, even with small dither amplitudes, may benefit from distortion correction when using PSF-fitting photometry.

The formula for removing the NIRI f/32 distortion is r' = r + kr^2, where k = (1.32 ± 0.02) x 10^-5 . r is the uncorrected distance from the field center in pixels and r' is the corrected distance from the center in pixels. Gemini provides an IDL program to do this correction. The writer of this manual is indeed aware that this is not helpful for many IRAF users. IRAF provides two routines, geomap and geotran, to remove distortion. Geomap produces a text file containing the coordinate transformation that is used by geotran to remove distortion from the images. The upsqiid routine xyget also includes distortion correction (next section).

A distortion correction remaps each pixel slightly so should be carried out on images that have already been processed by sky subtraction and flat fielding. There can be an impact on surface brightness measurements of extended sources. The IRAF geotran routine will preserve flux when the 'fluxconserve' flag is set to 'yes'. This serves to warm that the distortion correction can change the photometry.

6.11 Small Field Mosaics

Using the upsqiid package two routines are required to combine images, xyget and nircombine. xyget finds common stars in the images and creates a registration database. nircombine combines the registration database into a composite image. This discussion draw heavily from the WHIRC Data Reduction Manual by R. Joyce.

upsqiid xyget

The task xyget is an interactive task in which each of the images is displayed in turn and a star common to all of the images is identified on the display. The measured centroid of each of the star images is put into a database file of the sort '<inlist>.xycom', where 'inlist' is the list of input images. The '<inlist>.xycom' file is then used as the input to the nircombine task which will combine the input images into a single output image.

Reduce the images to be mosaicked (sky subtraction, flatfielding, etc.) as above and make a list of their names.

As discussed above it is probably best to set the sky levels for all of the images to zero before running xyget. This is in fact optional since removing the sky level can be carried out automatically within the nircombine task. However, the two approaches will give slightly different sky levels in the combined image so I prefer to go with what I understand.

Run xyget. Figure 8 illustrates the parameter set for xyget without distortion correction.

'images' parameter is the list containing the images to be analyzed.

'frame_nu' parameter has an unusual, non-IRAF syntax. The first number is the total number of images to be analyzed. The second number is the number of the reference image in the list of images. In the example, there are five images in the list and the first will be used as the reference.

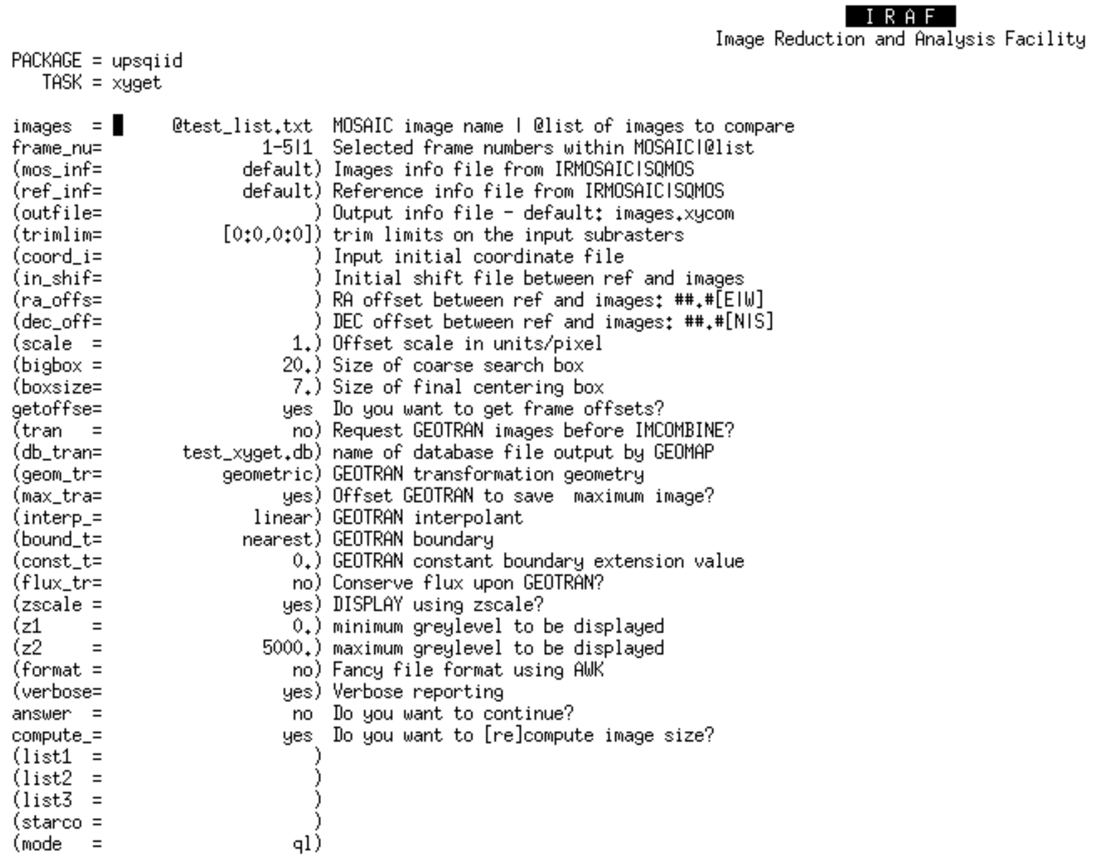
'bigbox' is the initial search box size.

'boxsize' is the final centering box. You will need to look at the output. If the '<list>.xycom' is missing one or more of the images (i.e., couldn't centroid on the star), then make the 'boxsize' slightly larger. This can happen if the seeing varies.

When you run xyget, the first image in the list will be displayed on the image display. Identify a star in the field which is well isolated, unsaturated, and is in all of the images. Put the cursor on it and hit <spacebar>. The star coordinates will be displayed in the IRAF xgterm window. One star is usually sufficient for alignment, so enter 'q' to stop this. The program will return "Do you want to get frame offsets? (yes)". Answer affirmatively with <CR>. The program will return "Select star for reference frame 1". You can just hit the <spacebar> on the same star. The program will then display frame 1 in the list, which will be the same image again, if you selected frame 1 as the reference in the 'frame_nu' parameter. Center the cursor on the same star and hit <spacebar>. The succeeding images will be displayed in turn. Move the cursor to the same star in each frame and hit <spacebar>. After the last image, the program will display a summary of the centroid information (Figure 9).

Check to make sure that all images have output data. If one or more is missing from the list, you may need to adjust the 'boxsize' parameter and re-run xyget. In this case you will have to delete the '<list>.xycom' file.

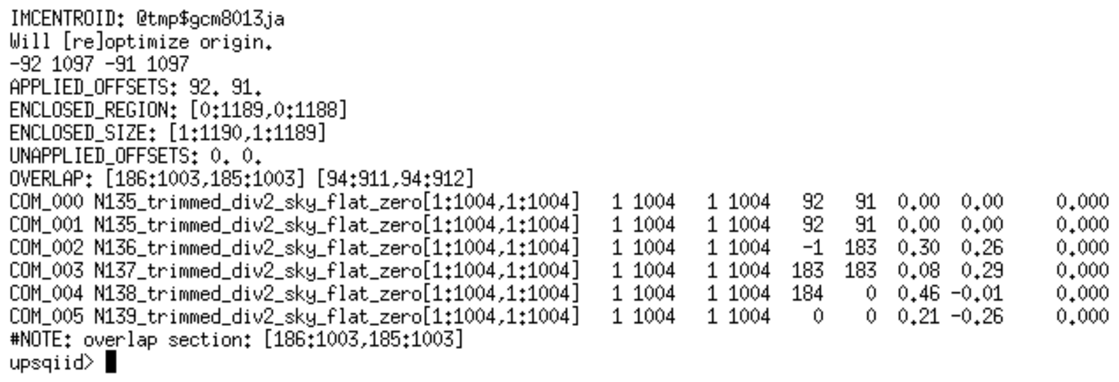
You may wish to rename the '<list>.xycom' file to something unique (e.g., 'M13.h.xycom') so that it is not overwritten by succeeding runs of xyget.



```
                                                                    I R A F
                                                      Image Reduction and Analysis Facility
PACKAGE = upsqiid
   TASK = xyget

images  =       @test_list.txt  MOSAIC image name | @list of images to compare
frame_nu=                1-5|1  Selected frame numbers within MOSAIC|@list
(mos_inf=             default) Images info file from IRMOSAIC|SQMOS
(ref_inf=             default) Reference info file from IRMOSAIC|SQMOS
(outfile=                    ) Output info file - default: images.xycom
(trimlim=           [0:0,0:0]) trim limits on the input subrasters
(coord_i=                    ) Input initial coordinate file
(in_shif=                    ) Initial shift file between ref and images
(ra_offs=                    ) RA offset between ref and images: ##.#[E|W]
(dec_off=                    ) DEC offset between ref and images: ##.#[N|S]
(scale  =                  1.) Offset scale in units/pixel
(bigbox =                 20.) Size of coarse search box
(boxsize=                  7.) Size of final centering box
getoffse=                 yes  Do you want to get frame offsets?
(tran   =                  no) Request GEOTRAN images before IMCOMBINE?
(db_tran=       test_xyget.db) name of database file output by GEOMAP
(geom_tr=           geometric) GEOTRAN transformation geometry
(max_tra=                 yes) Offset GEOTRAN to save  maximum image?
(interp_=              linear) GEOTRAN interpolant
(bound_t=             nearest) GEOTRAN boundary
(const_t=                  0.) GEOTRAN constant boundary extension value
(flux_tr=                  no) Conserve flux upon GEOTRAN?
(zscale =                 yes) DISPLAY using zscale?
(z1     =                  0.) minimum greylevel to be displayed
(z2     =               5000.) maximum greylevel to be displayed
(format =                  no) Fancy file format using AWK
(verbose=                 yes) Verbose reporting
answer  =                  no  Do you want to continue?
compute_=                 yes  Do you want to [re]compute image size?
(list1  =                    )
(list2  =                    )
(list3  =                    )
(starco =                    )
(mode   =                  ql)
```

Figure 8 – Parameter set for the task xyget without distortion correction. The parameter 'tran' is set to 'no'.

If you turn on the correct flags xyget will apply a distortion correction. You will need to know the same kind of information needed when the distortion correction was discussed above. If you want to apply the distortion correction you must provide a data base file with the geomap output. Then set the parameter 'tran' to 'yes.' The other parameters can remain as in Figure 8, i.e. the geotran transformation geometry 'geom_tr' set to 'geometric' , 'max_tra' to yes, 'interp' to linear, 'bound_t' to 'nearest' and 'const_t' to 0.



```
IMCENTROID: @tmp$gcm8013ja
Will [re]optimize origin.
-92 1097 -91 1097
APPLIED_OFFSETS: 92. 91.
ENCLOSED_REGION: [0:1189,0:1188]
ENCLOSED_SIZE: [1:1190,1:1189]
UNAPPLIED_OFFSETS: 0. 0.
OVERLAP: [186:1003,185:1003] [94:911,94:912]
COM_000 N135_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004   92   91  0.00  0.00     0.000
COM_001 N135_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004   92   91  0.00  0.00     0.000
COM_002 N136_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004   -1  183  0.30  0.26     0.000
COM_003 N137_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004  183  183  0.08  0.29     0.000
COM_004 N138_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004  184    0  0.46 -0.01     0.000
COM_005 N139_trimmed_div2_sky_flat_zero[1:1004,1:1004]   1 1004   1 1004    0    0  0.21 -0.26     0.000
#NOTE: overlap section: [186:1003,185:1003]
upsqiid> █
```

Figure 9 – Output at completion of the xyget task. Note that all of the five images have offset data.

upsqiid nircombine

The task nircombine (Figure 10) combines the input images using the '<list>.xycom' output of xyget.

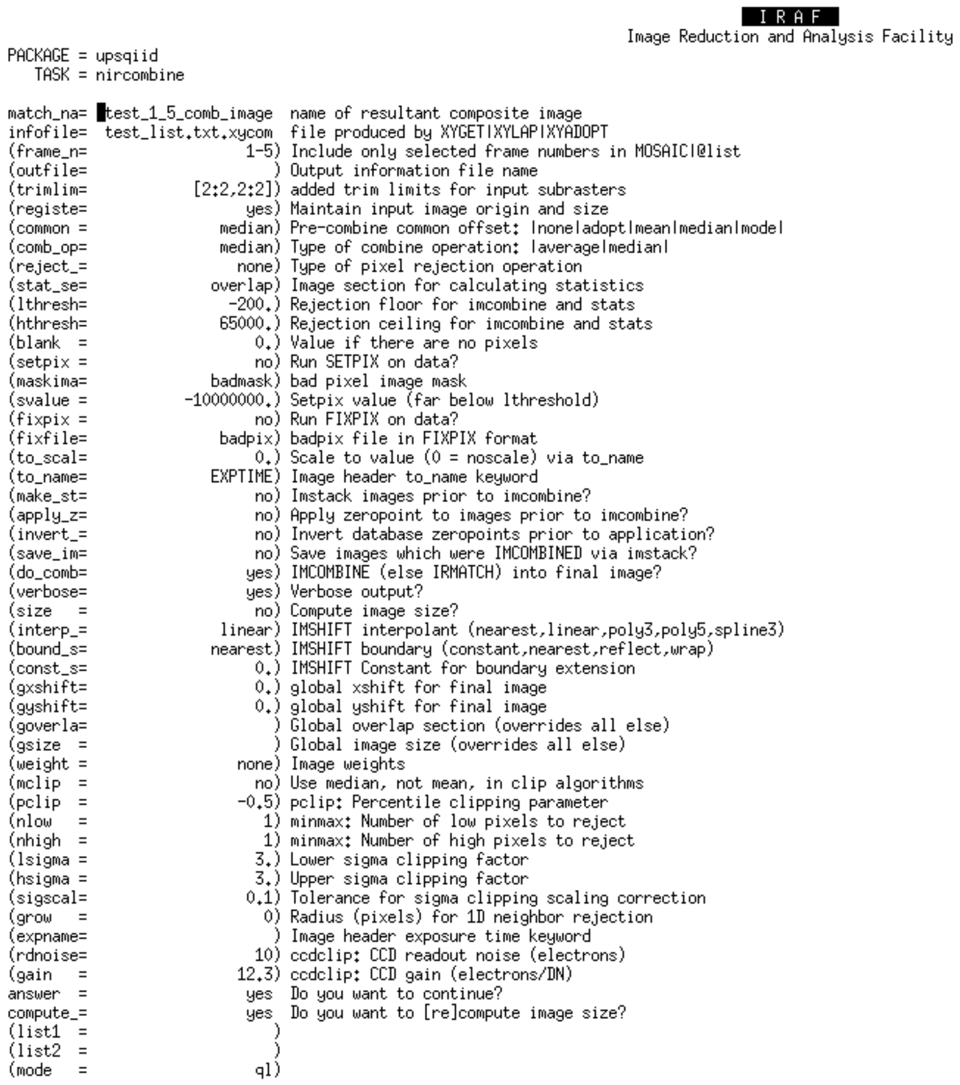
'match_na' will be the name of the resultant image.

'infofile' is the '<list>.xycom' output file from xyget.

'frame_n' is the list of images to be combined. Normally, this would be the entire set of input images, but if for some reasons (bad seeing, bad read, etc.) an image needs to be excluded it can be done here.

'comb_op' specifies the method used for combining the images. median combination is aesthetically pleasing because it usually results in the removal of fixed image artifacts which survived the earlier bad pixel repairs. Better photometric accuracy may result from 'average' to avoid signal clipping that 'median' can introduce. If in doubt run the task with both modes and compare the results.

'apply_z' will normalize the sky levels in the images before combining them.



```
                                                                    I R A F
                                                      Image Reduction and Analysis Facility
PACKAGE = upsqiid
   TASK = nircombine

match_na= test_1_5_comb_image  name of resultant composite image
infofile=  test_list.txt.xycom  file produced by XYGET|XYLAP|XYADOPT
(frame_n=                 1-5) Include only selected frame numbers in MOSAIC|@list
(outfile=                    ) Output information file name
(trimlim=          [2:2,2:2]) added trim limits for input subrasters
(registe=                 yes) Maintain input image origin and size
(common =              median) Pre-combine common offset: |none|adopt|mean|median|mode|
(comb_op=              median) Type of combine operation: |average|median|
(reject_=                none) Type of pixel rejection operation
(stat_se=             overlap) Image section for calculating statistics
(lthresh=               -200.) Rejection floor for imcombine and stats
(hthresh=              65000.) Rejection ceiling for imcombine and stats
(blank  =                  0.) Value if there are no pixels
(setpix =                  no) Run SETPIX on data?
(maskima=             badmask) bad pixel image mask
(svalue =          -10000000.) Setpix value (far below lthreshold)
(fixpix =                  no) Run FIXPIX on data?
(fixfile=              badpix) badpix file in FIXPIX format
(to_scal=                  0.) Scale to value (0 = noscale) via to_name
(to_name=             EXPTIME) Image header to_name keyword
(make_st=                  no) Imstack images prior to imcombine?
(apply_z=                  no) Apply zeropoint to images prior to imcombine?
(invert_=                  no) Invert database zeropoints prior to application?
(save_im=                  no) Save images which were IMCOMBINED via imstack?
(do_comb=                 yes) IMCOMBINE (else IRMATCH) into final image?
(verbose=                 yes) Verbose output?
(size   =                  no) Compute image size?
(interp_=              linear) IMSHIFT interpolant (nearest,linear,poly3,poly5,spline3)
(bound_s=             nearest) IMSHIFT boundary (constant,nearest,reflect,wrap)
(const_s=                  0.) IMSHIFT Constant for boundary extension
(gxshift=                  0.) global xshift for final image
(gyshift=                  0.) global yshift for final image
(goverla=                    ) Global overlap section (overrides all else)
(gsize  =                    ) Global image size (overrides all else)
(weight =                none) Image weights
(mclip  =                  no) Use median, not mean, in clip algorithms
(pclip  =                -0.5) pclip: Percentile clipping parameter
(nlow   =                   1) minmax: Number of low pixels to reject
(nhigh  =                   1) minmax: Number of high pixels to reject
(lsigma =                  3.) Lower sigma clipping factor
(hsigma =                  3.) Upper sigma clipping factor
(sigscal=                 0.1) Tolerance for sigma clipping scaling correction
(grow   =                   0) Radius (pixels) for 1D neighbor rejection
(expname=                    ) Image header exposure time keyword
(rdnoise=                  10) ccdclip: CCD readout noise (electrons)
(gain   =                12.3) ccdclip: CCD gain (electrons/DN)
answer  =                 yes  Do you want to continue?
compute_=                 yes  Do you want to [re]compute image size?
(list1  =                    )
(list2  =                    )
(mode   =                  ql)
```

Figure 10 – Parameter set for the IRAF task 'nircombine'. If the 'apply_z" parameter is set to 'yes' nircombine will normalize offsets in the sky level of the input images.

6.12 Larger Mosaics

NIRI was designed to provide detailed images of regions of small spatial extent. It is likely that large mosaics would be undertaken by other instruments that were designed for wide-field surveys. The most general approach is probably to generate a world coordinate system. There is documentation at
http://iraf.noao.edu/projects/ccdmosaic/astrometry/astrom.html
Infrared maps covering a larger area can be generated using the generic IRAF tasks in the noao.imred.irred package.

irmosaic, apphot.center and irmatch2d can be used in a manner analogous to the upsqiid tasks discussed above. As with the upsqiid tasks the images must have spatial overlap. Furthermore with the IRAF tasks the data must be taken in a regular N × M mosaic pattern.

The process is a bit cumbersome in that one must manually generate and edit a file containing the relative coordinates of reference stars in adjacent images. irmosaic combines the input images into a single N × M mosaic in the same geometry that they were taken on the sky. apphot.center task is used following irmosaic to measure the centroids of pairs of stars which are common to adjacent images in the mosaic image. irmatch2d task shifts the images in the original mosaic, using this coordinate file so that the adjacent stars match up, creating a single composite image. Irmosaic and apphot.center both required editing of input and output files.

An example of the use of these routines can be found in the WHIRC Data Reduction manual that is available on the NOAO/KPNO web site.

7. Analysis

Various analysis packages are available to do photometry in the reduced images. The simplest, most limited technique is to use imexam on the displayed image. The r or a commands measure the area of point sources. Users wishing additional sophistication should consult “A User’s Guide to Stellar Photometry with IRAF” by Phil Massey and Lindsey Davis. A commonly used tool is DAOPHOT (Stetson 1987 PASP 99 191). This software package is available in the IRAF digiphot.doaphot package. A guide, “A Reference Guide to the IRAF/DAOPHOT Package”, by Lindsey Davis is available.

8. Reduction with Gemini IRAF tools

8.1 General use tools

Gemini IRAF is written to use MEF files. There is a brief introduction to Gemini IRAF available on line:

https://www.noao.edu/meetings/gdw/files/Labrie_Gemini_IRAF.pdf

The Gemini IRAF package can be accessed by loading “gemini” (see 3.2 of the Beginner’s Guide) and for NIRI data “niri”.

A small set of general use tools can be found in the gemini.gemtools package. Some of the most useful tasks are:

gemarith This is the MEF version of imarith.

gemcombine   This is the MEF version of imcombine

gemhead This is the same as header but for MEF files

wmef   Builds a MEF file from SEF files.   Combined with fxsplit this provides a path between SEF and MEF.

8.2 NIRI Package

In addition to the general use routines each Gemini instrument has its own package. NIRI reduction routines exist to do the following tasks.

8.2.1 Pre-reduction corrections

Should you decide to use either of the following routines they need to be executed before any additional reduction steps.

cleanir – Vertical strip correction Python routine This is not always needed.   It is designed to take out the vertical striping and horizontal banding patterns and bias offsets that sometimes result from the array controller.   The code is discussed on and can be downloaded at
https://www.gemini.edu/sciops/data-and-results?q=node/11650

nirlin – Non-linearity correction Python routine. Non-linearity correction is typically important when the wells are nearly full. This was discussed with the SEF reduction.   The non-linearity correction code can be downloaded atq
http://www.gemini.edu/sciops/instruments/niri/data-format-andreduction/nirlin

8.2.2 Preparing the images – nprepare

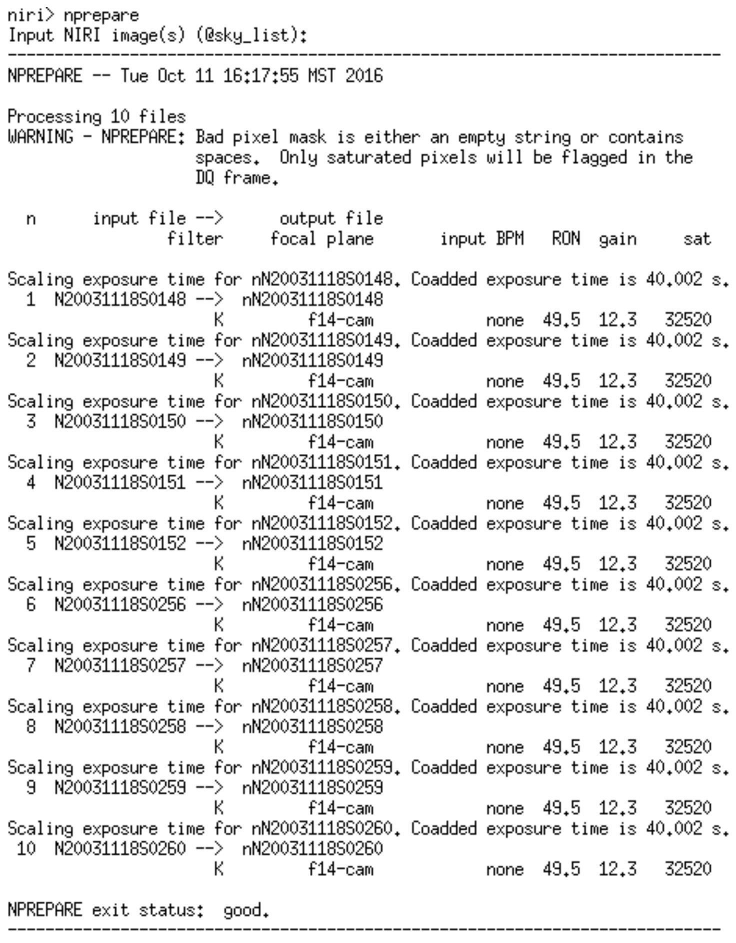
nprepare must be executed if you are going to use the packaged NIRI reduction routines.   nprepare adds header keywords to the PHU that the other routines require.   WARNING—There is a similar routine for GNIRS called nsprepare. Do not type the wrong name.   The NIRI routine is nprepare.

NIRI writes images taken as coadds as a summed, not averaged, image. This means that if one takes 10 coadds of 1 sec each, the final image will have a flux level in ADU equivalent to 10 sec. The task nprepare leaves the pixel values as written by NIRI but modifies the header keyword EXPTIME to be the total exposure (the original exposure time multiplied by the number of coadds). nprepare adds a keyword COADDEXP that contains the original exposure time for each individual frame. While this may seem confusing, it is consistent with one ADU in the output image representing a fixed number of detected electrons

The NIRI array controller averages the data based on the number of fowler reads so no modification is necessary for different LNRS or NDAVGS values. The flux value will correspond to the exposure time, and the noise will vary as a function of the number of reads. . In the example data set the flat and dark coadds and LNRS equal one.

In addition to these changes nprepare also adds header keywords GAIN, RDNOISE, SATURATI, NONLINEA, and BIAS. These words provide the gain (e-/ADU), read noise (e-), saturation level (ADU), the level at which the data start to be non-linear (ADU), and the array bias voltage, respectively. These header keywords will be used by other scripts in the NIRI package. nprepare can also add variance and data quality planes, as described in the help pages for nprepare.

nprepare refers to a data file named nprepare.dat that contains the information for adding these header keywords. The user is responsible for confirming that nprepare and nprepare.dat are appropriate for the data being reduced. nprepare.dat is included with the current release of the NIRI package. nprepare.dat will be modified from time to time to reflect updates in the array parameters or instrument configuration. The writer of this manual does not know how to confirm that the correct version is being used. Let me know if you find out.



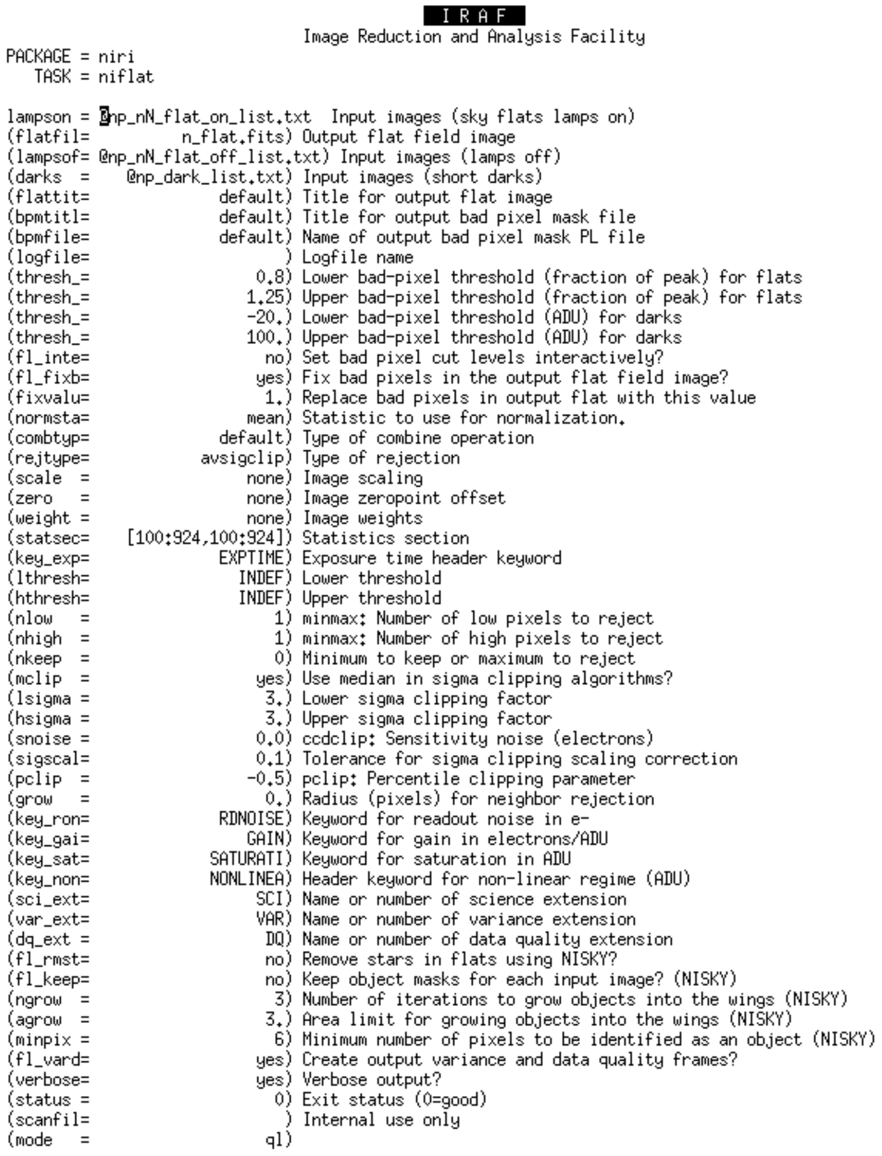
```
niri> nprepare
Input NIRI image(s) (@sky_list):
----------------------------------------------------------------------------
NPREPARE -- Tue Oct 11 16:17:55 MST 2016

Processing 10 files
WARNING - NPREPARE: Bad pixel mask is either an empty string or contains
                    spaces.  Only saturated pixels will be flagged in the
                    DQ frame.

  n      input file -->      output file
                filter     focal plane       input BPM   RON  gain     sat

Scaling exposure time for nN20031118S0148. Coadded exposure time is 40.002 s.
  1  N20031118S0148 -->  nN20031118S0148
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0149. Coadded exposure time is 40.002 s.
  2  N20031118S0149 -->  nN20031118S0149
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0150. Coadded exposure time is 40.002 s.
  3  N20031118S0150 -->  nN20031118S0150
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0151. Coadded exposure time is 40.002 s.
  4  N20031118S0151 -->  nN20031118S0151
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0152. Coadded exposure time is 40.002 s.
  5  N20031118S0152 -->  nN20031118S0152
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0256. Coadded exposure time is 40.002 s.
  6  N20031118S0256 -->  nN20031118S0256
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0257. Coadded exposure time is 40.002 s.
  7  N20031118S0257 -->  nN20031118S0257
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0258. Coadded exposure time is 40.002 s.
  8  N20031118S0258 -->  nN20031118S0258
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0259. Coadded exposure time is 40.002 s.
  9  N20031118S0259 -->  nN20031118S0259
                      K         f14-cam           none  49.5  12.3   32520
Scaling exposure time for nN20031118S0260. Coadded exposure time is 40.002 s.
 10  N20031118S0260 -->  nN20031118S0260
                      K         f14-cam           none  49.5  12.3   32520

NPREPARE exit status:  good.
----------------------------------------------------------------------------
```

Figure 11 – Output from nprepare. Note the data is renamed.

8.2.3 Flats and bad pixel map

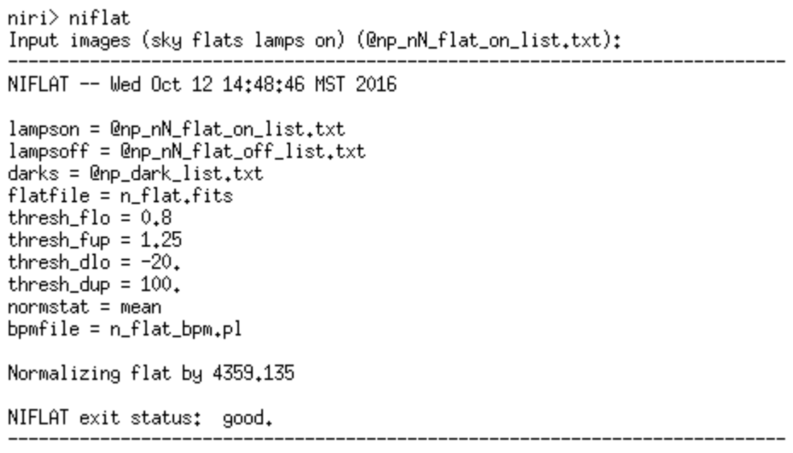
niflat derives the flat field from the flat on and off exposures and, if you have a separate set of darks, a bad pixel map. As with all the Gemini MEF routines the images must have previously been processed by nprepare.

If you simply type niflat it will want a list of the flat-on exposures. From this it computes an average flat-on. However, it is better to do an epar niflat. Then enter the lists of flat-on exposure, flat-off exposures, and dark exposures. There is also a parameter for the new flat field name. Figure 12 shows the input and Figure 13 the resulting output. The flat-on, flat-off, and darks were all copied into the same subdirectory.



```
                              I R A F
                    Image Reduction and Analysis Facility
PACKAGE = niri
   TASK = niflat

lampson = @np_nN_flat_on_list.txt  Input images (sky flats lamps on)
(flatfil=           n_flat.fits) Output flat field image
(lampsof= @np_nN_flat_off_list.txt) Input images (lamps off)
(darks  =    @np_dark_list.txt) Input images (short darks)
(flattit=               default) Title for output flat image
(bpmtitl=               default) Title for output bad pixel mask file
(bpmfile=               default) Name of output bad pixel mask PL file
(logfile=                      ) Logfile name
(thresh_=                   0.8) Lower bad-pixel threshold (fraction of peak) for flats
(thresh_=                  1.25) Upper bad-pixel threshold (fraction of peak) for flats
(thresh_=                  -20.) Lower bad-pixel threshold (ADU) for darks
(thresh_=                  100.) Upper bad-pixel threshold (ADU) for darks
(fl_inte=                    no) Set bad pixel cut levels interactively?
(fl_fixb=                   yes) Fix bad pixels in the output flat field image?
(fixvalu=                    1.) Replace bad pixels in output flat with this value
(normsta=                  mean) Statistic to use for normalization.
(combtyp=               default) Type of combine operation
(rejtype=              avsigclip) Type of rejection
(scale  =                  none) Image scaling
(zero   =                  none) Image zeropoint offset
(weight =                  none) Image weights
(statsec=    [100:924,100:924]) Statistics section
(key_exp=               EXPTIME) Exposure time header keyword
(lthresh=                 INDEF) Lower threshold
(hthresh=                 INDEF) Upper threshold
(nlow   =                     1) minmax: Number of low pixels to reject
(nhigh  =                     1) minmax: Number of high pixels to reject
(nkeep  =                     0) Minimum to keep or maximum to reject
(mclip  =                   yes) Use median in sigma clipping algorithms?
(lsigma =                    3.) Lower sigma clipping factor
(hsigma =                    3.) Upper sigma clipping factor
(snoise =                   0.0) ccdclip: Sensitivity noise (electrons)
(sigscal=                   0.1) Tolerance for sigma clipping scaling correction
(pclip  =                  -0.5) pclip: Percentile clipping parameter
(grow   =                    0.) Radius (pixels) for neighbor rejection
(key_ron=               RDNOISE) Keyword for readout noise in e-
(key_gai=                  GAIN) Keyword for gain in electrons/ADU
(key_sat=              SATURATI) Keyword for saturation in ADU
(key_non=              NONLINEA) Header keyword for non-linear regime (ADU)
(sci_ext=                   SCI) Name or number of science extension
(var_ext=                   VAR) Name or number of variance extension
(dq_ext =                    DQ) Name or number of data quality extension
(fl_rmst=                    no) Remove stars in flats using NISKY?
(fl_keep=                    no) Keep object masks for each input image? (NISKY)
(ngrow  =                     3) Number of iterations to grow objects into the wings (NISKY)
(agrow  =                    3.) Area limit for growing objects into the wings (NISKY)
(minpix =                     6) Minimum number of pixels to be identified as an object (NISKY)
(fl_vard=                   yes) Create output variance and data quality frames?
(verbose=                   yes) Verbose output?
(status =                     0) Exit status (0=good)
(scanfil=                      ) Internal use only
(mode   =                    ql)
```

Figure 12 – epar for niflat.



```
niri> niflat
Input images (sky flats lamps on) (@np_nN_flat_on_list.txt):
----------------------------------------------------------------------------
NIFLAT -- Wed Oct 12 14:48:46 MST 2016

lampson = @np_nN_flat_on_list.txt
lampsoff = @np_nN_flat_off_list.txt
darks = @np_dark_list.txt
flatfile = n_flat.fits
thresh_flo = 0.8
thresh_fup = 1.25
thresh_dlo = -20.
thresh_dup = 100.
normstat = mean
bpmfile = n_flat_bpm.pl

Normalizing flat by 4359.135

NIFLAT exit status:  good.
----------------------------------------------------------------------------
```

Figure 13 – Output from niflat.

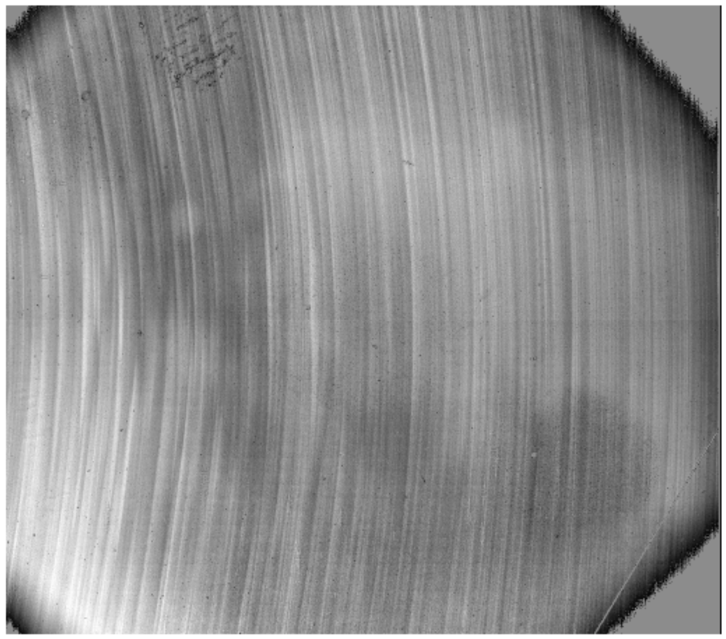


Figure 14 – Image of the resulting flat.

8.2.4 Sky

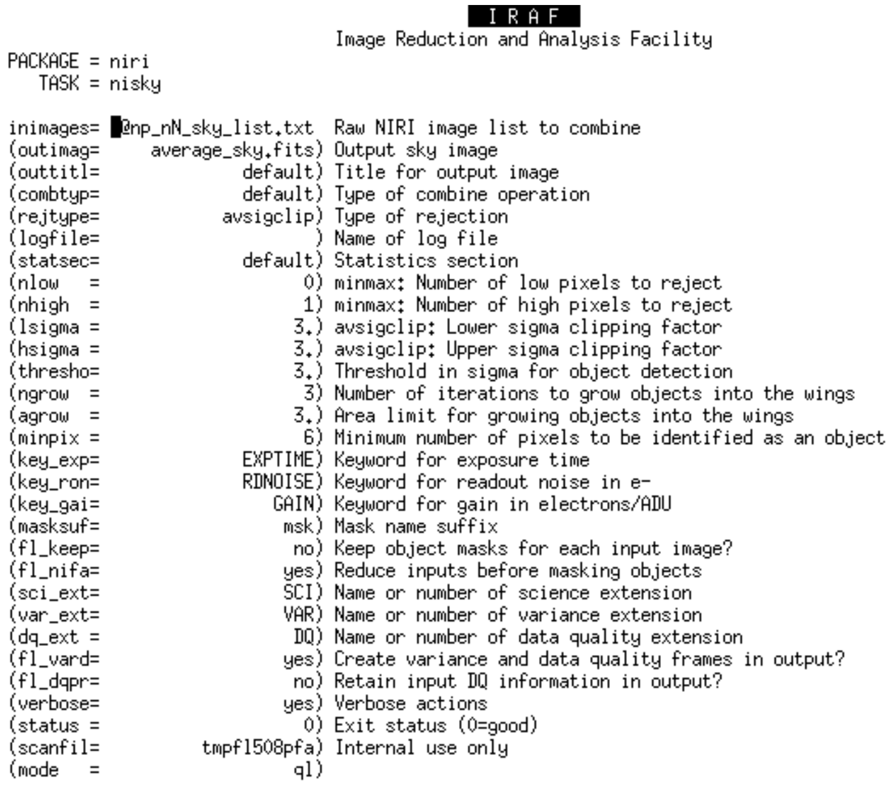
nisky - This is the standard routine for constructing a sky image from sky observations. As with all the Gemini MEF routines the images must have been previously run through nprepare. Here is what the IRAF help for nisky says happens:

nisky starts by generating a simple median sky image which it uses to reduce the input sky images. A flat field is generated by normalizing this quick sky image. The input images, sky subtracted and flattened, are then used to identify objects and create an object mask.

Using the object mask the routine objmasks identifies the objects in the field. The parameters threshold and minpix can be adjusted to fine tune how many faint objects are detected. The objmasks parameters ngrow and agrow dictate how identified objects are expanded in the mask to cover the faint extended wings. objmasks works best on fields with many smaller objects rather than large extended objects.

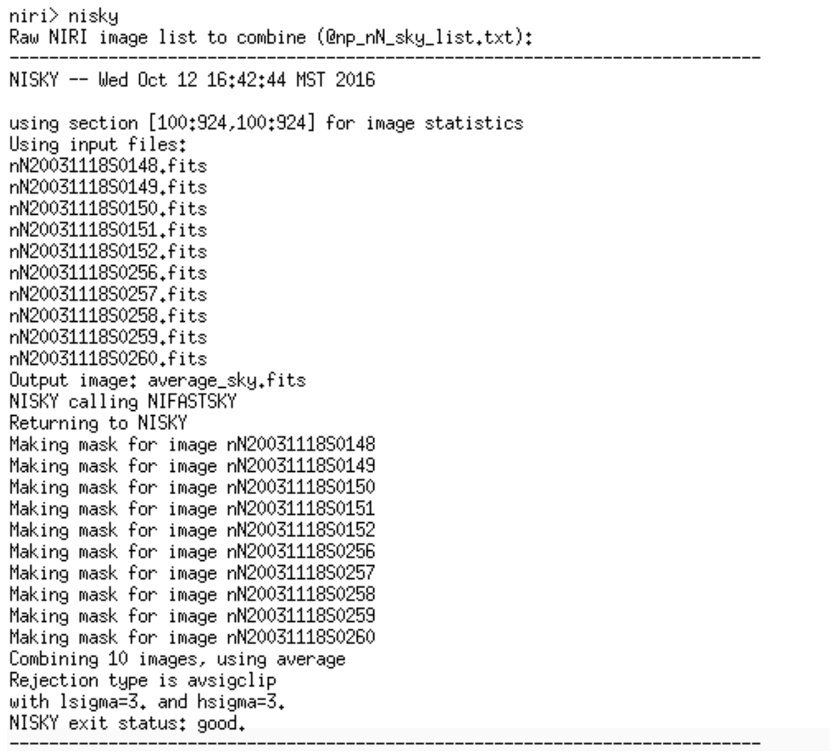
Once the masks are generated using objmasks, imcombine is used to generate the final sky image ('average' is selected if combtype='default') with the stars masked. The variance and data quality frames are derived from the imcombine output.

nisupersky – This should be used for deep imaging. After combining the final image using imcoadd, run nisupersky to create new mask files for NISKY. Then, re-run NISKY with the new masks. All subsequent steps in the reduction must be repeated.



```
                                   I R A F
                      Image Reduction and Analysis Facility
PACKAGE = niri
   TASK = nisky

inimages= @np_nN_sky_list.txt  Raw NIRI image list to combine
(outimag=     average_sky.fits) Output sky image
(outtitl=              default) Title for output image
(combtyp=              default) Type of combine operation
(rejtype=            avsigclip) Type of rejection
(logfile=                     ) Name of log file
(statsec=              default) Statistics section
(nlow   =                    0) minmax: Number of low pixels to reject
(nhigh  =                    1) minmax: Number of high pixels to reject
(lsigma =                   3.) avsigclip: Lower sigma clipping factor
(hsigma =                   3.) avsigclip: Upper sigma clipping factor
(thresho=                   3.) Threshold in sigma for object detection
(ngrow  =                    3) Number of iterations to grow objects into the wings
(agrow  =                   3.) Area limit for growing objects into the wings
(minpix =                    6) Minimum number of pixels to be identified as an object
(key_exp=              EXPTIME) Keyword for exposure time
(key_ron=              RDNOISE) Keyword for readout noise in e-
(key_gai=                 GAIN) Keyword for gain in electrons/ADU
(masksuf=                  msk) Mask name suffix
(fl_keep=                   no) Keep object masks for each input image?
(fl_nifa=                  yes) Reduce inputs before masking objects
(sci_ext=                  SCI) Name or number of science extension
(var_ext=                  VAR) Name or number of variance extension
(dq_ext =                   DQ) Name or number of data quality extension
(fl_vard=                  yes) Create variance and data quality frames in output?
(fl_dqpr=                   no) Retain input DQ information in output?
(verbose=                  yes) Verbose actions
(status =                    0) Exit status (0=good)
(scanfil=          tmpfl508pfa) Internal use only
(mode   =                   ql)
```

Figure 15 – The parameters for nisky.



```
niri> nisky
Raw NIRI image list to combine (@np_nN_sky_list.txt):
----------------------------------------------------------------------------
NISKY -- Wed Oct 12 16:42:44 MST 2016

using section [100:924,100:924] for image statistics
Using input files:
nN20031118S0148.fits
nN20031118S0149.fits
nN20031118S0150.fits
nN20031118S0151.fits
nN20031118S0152.fits
nN20031118S0256.fits
nN20031118S0257.fits
nN20031118S0258.fits
nN20031118S0259.fits
nN20031118S0260.fits
Output image: average_sky.fits
NISKY calling NIFASTSKY
Returning to NISKY
Making mask for image nN20031118S0148
Making mask for image nN20031118S0149
Making mask for image nN20031118S0150
Making mask for image nN20031118S0151
Making mask for image nN20031118S0152
Making mask for image nN20031118S0256
Making mask for image nN20031118S0257
Making mask for image nN20031118S0258
Making mask for image nN20031118S0259
Making mask for image nN20031118S0260
Combining 10 images, using average
Rejection type is avsigclip
with lsigma=3. and hsigma=3.
NISKY exit status: good.
----------------------------------------------------------------------------
```

Figure 16 – the output from nisky.

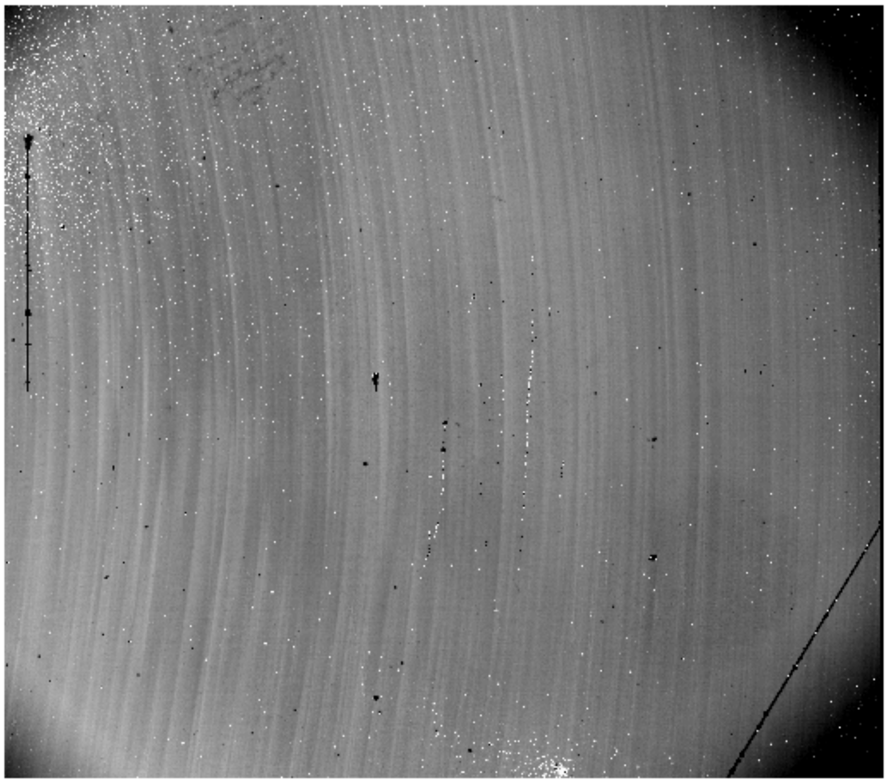
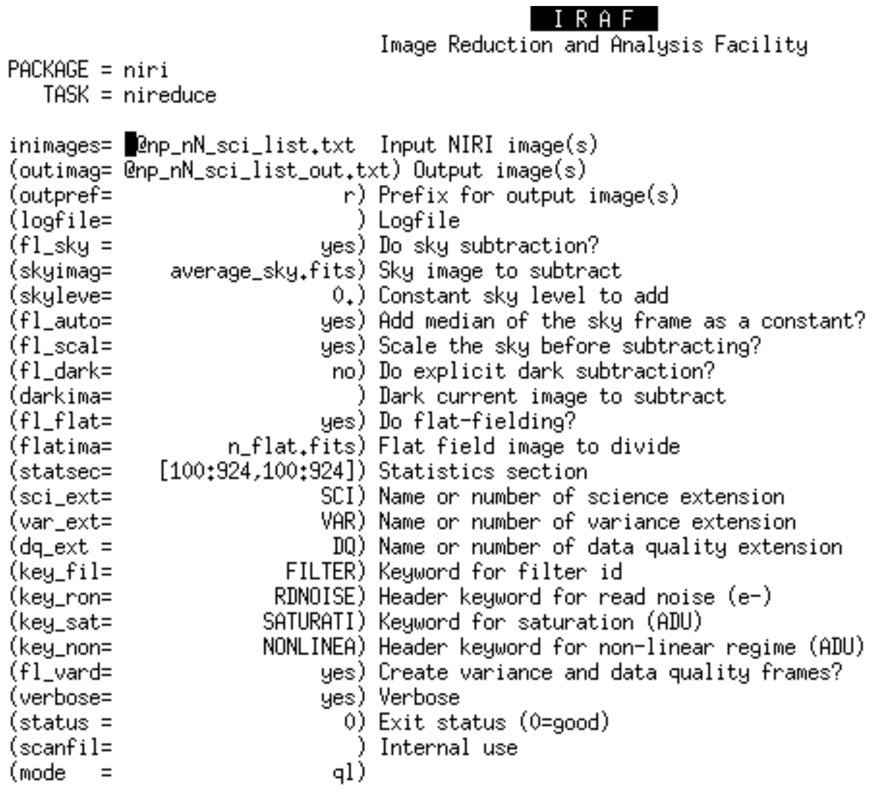


Figure 17 – Image of the resulting average ‘sky’.

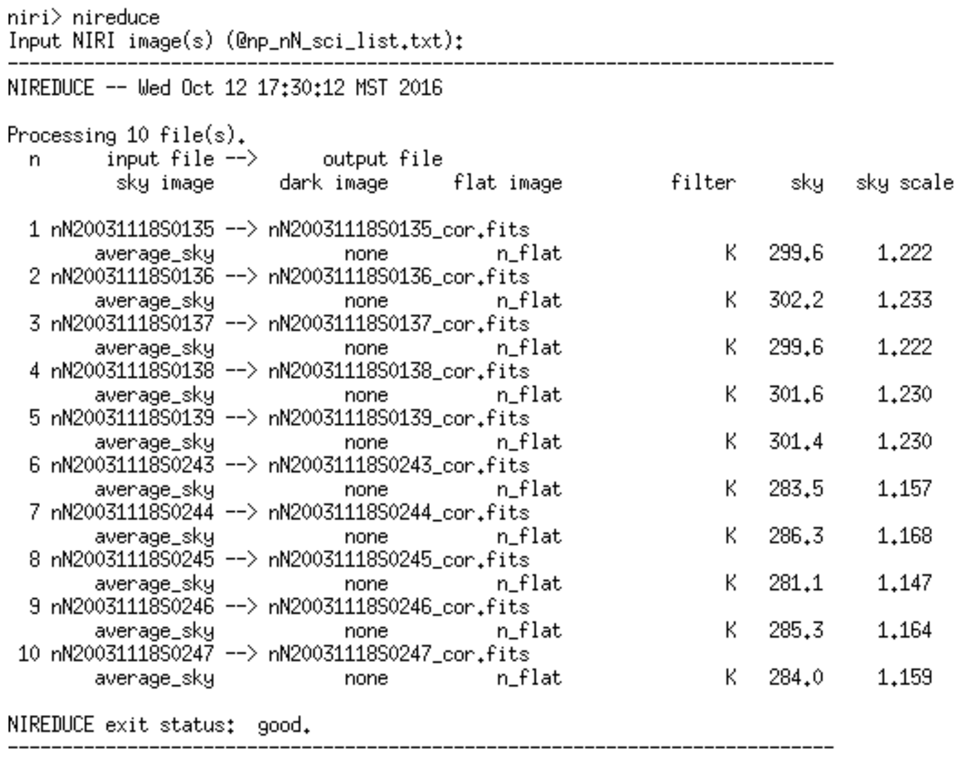
8.2.5 Removing the flat and sky from the science frames.

nireduce – This routine does the sky subtraction and flat fielding to the science frames. As for all the routines in this package the input images must be previously processed using nprepare. Sky and flat images made up by niflat and nisky are required. There are a number of inputs and in the example the sky and flat file names and the science frame input and output lists were entered using epar nireduce. The flat field images are assumed to be normalized flat fields. Nireduce does allow for the restoration of the sky level to the original level to maintain the noise characteristics of the image. help nireduce as well as help niflat and help nisky discuss this and other features that allow for the statistical analysis of the image. Users that require these features are directed to the help files. Similarly the help packages discuss how saturated pixels are processed.



```
                                   I R A F
                    Image Reduction and Analysis Facility
PACKAGE = niri
   TASK = nireduce

inimages= @np_nN_sci_list.txt  Input NIRI image(s)
(outimag= @np_nN_sci_list_out.txt) Output image(s)
(outpref=                    r) Prefix for output image(s)
(logfile=                     ) Logfile
(fl_sky =                  yes) Do sky subtraction?
(skyimag=      average_sky.fits) Sky image to subtract
(skyleve=                   0.) Constant sky level to add
(fl_auto=                  yes) Add median of the sky frame as a constant?
(fl_scal=                  yes) Scale the sky before subtracting?
(fl_dark=                   no) Do explicit dark subtraction?
(darkima=                     ) Dark current image to subtract
(fl_flat=                  yes) Do flat-fielding?
(flatima=          n_flat.fits) Flat field image to divide
(statsec=     [100:924,100:924]) Statistics section
(sci_ext=                  SCI) Name or number of science extension
(var_ext=                  VAR) Name or number of variance extension
(dq_ext =                   DQ) Name or number of data quality extension
(key_fil=               FILTER) Keyword for filter id
(key_ron=              RDNOISE) Header keyword for read noise (e-)
(key_sat=             SATURATI) Keyword for saturation (ADU)
(key_non=             NONLINEA) Header keyword for non-linear regime (ADU)
(fl_vard=                  yes) Create variance and data quality frames?
(verbose=                  yes) Verbose
(status =                    0) Exit status (0=good)
(scanfil=                     ) Internal use
(mode   =                   ql)
```

Figure 18 – epar input for nireduce.



```
niri> nireduce
Input NIRI image(s) (@np_nN_sci_list.txt):
------------------------------------------------------------------------------
NIREDUCE -- Wed Oct 12 17:30:12 MST 2016

Processing 10 file(s).
  n      input file -->      output file
         sky image      dark image      flat image         filter     sky   sky scale

  1 nN20031118S0135 --> nN20031118S0135_cor.fits
        average_sky           none          n_flat              K   299.6     1.222
  2 nN20031118S0136 --> nN20031118S0136_cor.fits
        average_sky           none          n_flat              K   302.2     1.233
  3 nN20031118S0137 --> nN20031118S0137_cor.fits
        average_sky           none          n_flat              K   299.6     1.222
  4 nN20031118S0138 --> nN20031118S0138_cor.fits
        average_sky           none          n_flat              K   301.6     1.230
  5 nN20031118S0139 --> nN20031118S0139_cor.fits
        average_sky           none          n_flat              K   301.4     1.230
  6 nN20031118S0243 --> nN20031118S0243_cor.fits
        average_sky           none          n_flat              K   283.5     1.157
  7 nN20031118S0244 --> nN20031118S0244_cor.fits
        average_sky           none          n_flat              K   286.3     1.168
  8 nN20031118S0245 --> nN20031118S0245_cor.fits
        average_sky           none          n_flat              K   281.1     1.147
  9 nN20031118S0246 --> nN20031118S0246_cor.fits
        average_sky           none          n_flat              K   285.3     1.164
 10 nN20031118S0247 --> nN20031118S0247_cor.fits
        average_sky           none          n_flat              K   284.0     1.159

NIREDUCE exit status:  good.
------------------------------------------------------------------------------
```

Figure 19 – Output from nireduce

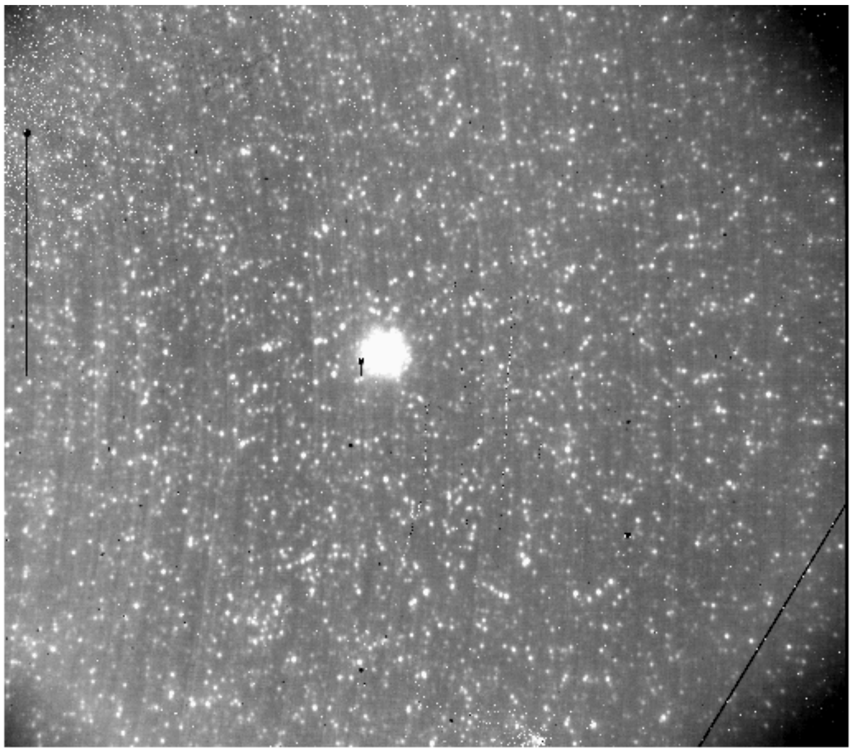


Figure 20 – Raw image of N20031118S0135

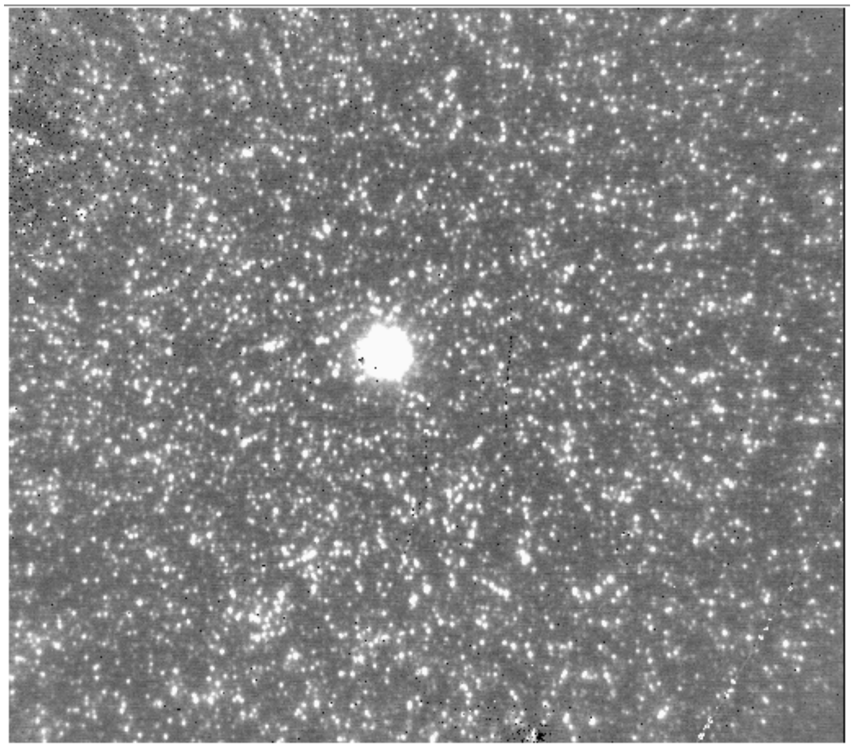


Figure 21 – The sky and flat corrected image N20031118S0135

8.2.6 Combining the images

imcoadd generates the final image by combining all the reduced science images. imcoadd uses geomap and geotran to register the images. This is a fully automatic process that does not involve identifying stars. My experience is that the upsqiid xyget and nircombine packages, that require the identification of a star or stars in the field, work much better on the crowded test data. I would strongly suggest that users of the Gemini IRAF package try both imcoadd and upsqiid xyget/nircombine. IRAF fxsplit can convert the MEF output from nireduce into SEF images suitable for use with upsqiid.

imcoadd employs imcombine to clean cosmic ray damage. The first image in the list will be used as the reference image. The first image will be the reference for geometry and used to scale magnitudes. The output image will have the same field as the first image. Obviously if the image quality varies in the data set it is advantages to select a first image of good quality. Users are encouraged to consult the help file for a complete discussion of this complex routine and a summary of how it functions.

imcoadd crops the output science image at the size of the first image. If you require a mosaic you should consult the packages discussed in section 6.11 and 6.12. The imcoadd documentation suggests that a minimum of five point sources are required for imcoadd to work well. If the science field is filled with diffuse emission the discussion in section 6.12 on making mosaics may be more helpful.

There are various outputs from this routine. Users should consult the information in the help file for details. Science-grade photometry should be undertaken from the cleaned average image [reference]_add. [reference]_add is on the same magnitude scale as the reference image, e.g. same exposure time and same airmass. If the reference image was obtained under photometric conditions, then photometry derived from [reference]_add can be calibrated in the same way as would be done for photometry from the reference image.

imcoadd is modular in design and each step has a flag in the parameter file related to it. When you run imcoadd expect that it will produce a large amount of output. Here is a summary of the reduction process.

1. Find the objects in the reference image (the first in the list of images) using DAOFIND: fl_find+
2. Derive the transformations using GEOMAP: fl_map+
3. Transform the images using GEOTRAN: fl_trn+
4. Derive the median image using IMCOMBINE: fl_med+
5. Derive the average image cleaned for cosmic-ray-events, using IMCOMBINE: fl_add+
6. Derive the average uncleaned image using IMCOMBINE: fl_avg+

imcoadd can be started from any intermediate result by setting the preceding flags to "no" and the flags for the following actions to "yes". For example, to rederive the cleaned average image rerun the task with the flags set to

```
fl_find- fl_map- fl_trn- fl_med- fl_add+ fl_avg+ fl_overwrite+
```

fl_overwrite=yes will delete the existing output files that are about the be rederived.

Input files:

The input images can be a comma separated list or a file that lists one image per line, e.g.

```
images=@imagelist
```

where "imagelist" contains the image names. Sophisticated bad pixel processing is possible. The inputs are described in the help pages.

If you run with fl_find- fl_map+, IMCOADD will look for the coordinate file [reference]_pos, where "reference" is the first image in the list of input images.

Output files:

The coadded images is in file [image name]_add.fits. If the input images are MEF this image will also be MEF. This is the only output image with the complete header.

There are many other output files generated. These are described in the help file.

8.2.7 NIRI Reduction Scripts

There is a tool in the NIRI package called 'niriexamples' . niriexmples is designed to be followed by the nature of the example, for instance 'niriexamples imaging'. Niriexamples provides a number of scripts that were set up for quality assessment. Advanced users may find these of use in building custom scripts to reduce their data.