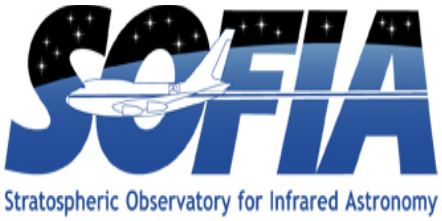

FORCAST Redux Users Manual

SCI-US-HBK-OP10-2003

Date: January 7, 2015
Revision: A

AFRC
Armstrong Flight Research Center
Edwards, CA 93523

ARC
Ames Research Center
Moffett Field, CA 94035

German Space Agency, DLR
Deutsches Zentrum für Luft und Raumfahrt

FORCAST Redux Users Manual

SCI-US-HBK-OP10-2003

AUTHOR:

William Vacca, USRA, SOFIA Senior Scientist	Date
Melanie Clarke, USRA, SOFIA Pipeline Engineer	Date

CONCURRENCE:

R.Y. Shuping, USRA, SOFIA DPS Lead	Date

APPROVAL:

William Reach, USRA, SOFIA Associate Director for Science	Date
William Szolnoki, USRA Business Manager	Date

REVISION HISTORY

REV	DATE	DESCRIPTION
-	12/30/13	Initial Release
A	1/7/15	Updated for Redux 1.0.1; re-organized to streamline distribution to GIs. Updates for Redux 1.0.3 OC2-2 release: Grism processing and grouping rules.

FORCAST Redux Users Manual
SCI-US-HBK-OP10-2003

Table of Contents

FORCAST Redux Users Manual
SCI-US-HBK-OP10-2003

1. INTRODUCTION

The SI Pipeline Users Manual (OP10) is intended for use by both SOFIA Science Center staff during routine data processing and analysis, and also as a reference for General Investigators (GIs) and archive users to understand how the data in which they are interested was processed. This manual is intended to provide all the needed information to execute the SI Level 2 Pipeline, flux calibrate the results, and assess the data quality of the resulting products. It will also provide a description of the algorithms used by the pipeline and both the final and intermediate data products.

A description of the current pipeline capabilities, testing results, known issues, and installation procedures are documented in the SI Pipeline Software Version Description Document (SVDD, SW06, DOCREF). The overall Verification and Validation (V&V) approach can be found in the Data Processing System V&V Plan (SV01-2232). Both documents can be obtained from the SOFIA document library in Windchill at location: / Software Management Development or Verification / Pipelines (DPS).

This manual applies to FORCAST Redux version 1.0.1 and later.

2. SI OBSERVING MODES SUPPORTED

2.1. FORCAST observing techniques

Because the sky is so bright in the mid-infrared (MIR) relative to astronomical sources, the way in which observations are made in the MIR is considerably different from the more familiar way they are made in the optical. Any raw image of a region in the MIR is overwhelmed by this sky "background" emission. The situation is similar to trying to observe in the optical during the day. The bright daylight sky swamps the detector and makes it impossible to see astronomical sources in the raw images.

In order to remove the background from the MIR image and detect the faint astronomical sources, observations of another region (free of sources) are made and the two images are subtracted. However, the MIR is highly variable, both spatially and – more importantly – temporally. It would take far too long (on the order of seconds) to reposition a large telescope to observe this "sky background" region: by the time the telescope had moved and settled at the new location, the sky background level would have changed so much that the subtraction of the two images would be useless. In order to avoid this problem, the secondary mirror (which is considerably smaller than the primary mirror) of the telescope is tilted, rather than moving the entire telescope. This allows observers to look at two different sky positions very quickly (on the order of a few to ten times per second), because tilting the secondary by an angle θ moves the center of the field imaged by the detector by θ on the sky. Tilting the secondary between two

positions is known as “chopping”. FORCAST observations are typically made with a chopping frequency of 4 Hz. That is, every 0.25 sec, the secondary is moved between the two observing positions.

Chopping can be done either symmetrically or asymmetrically. Symmetric chopping means that the secondary mirror is tilted symmetrically about the telescope optical axis (also known as the boresight) in the two chop positions. The distance between the two chop positions is known as the chop throw. The distance between the boresight and either chop position is known as the chop amplitude and is equal to half the chop throw (see Figure 1).

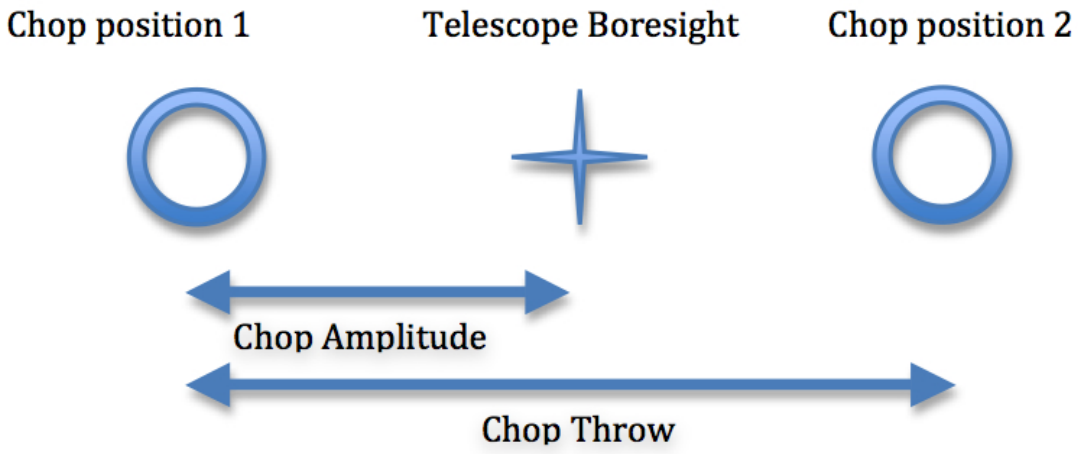

Figure 1: Symmetric Chop

Asymmetric chopping means that the secondary is aligned with the telescope boresight in one position, but is tilted away from the boresight in the chop position. The chop amplitude is equal to the chop throw in this case (see Figure 2).

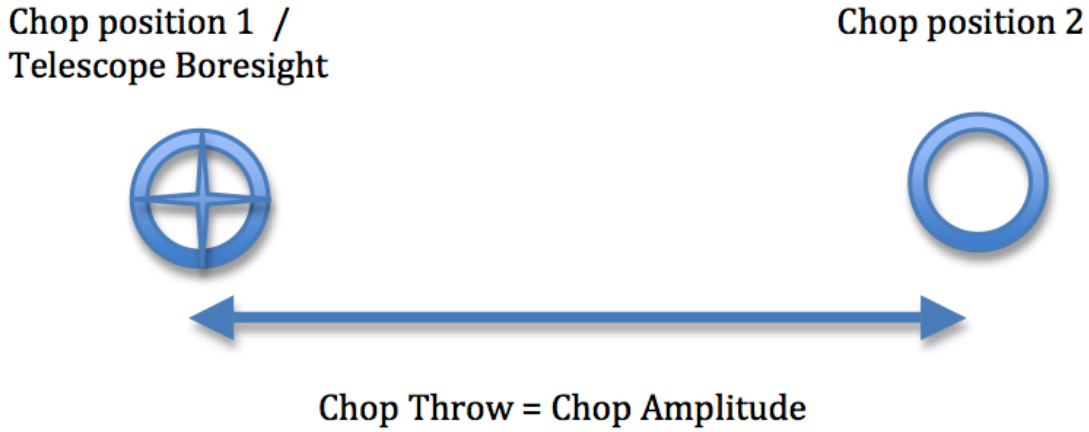

Figure 2: Asymmetric Chop

Unfortunately, moving the secondary mirror causes the telescope to be slightly misaligned, which introduces optical distortions (notably the optical aberration known as coma) and

additional background emission from the telescope (considerably smaller than the sky emission but present nonetheless) in the images. The optical distortions can be minimized by tilting the secondary only tiny fractions of a degree. The additional telescopic background can be removed by moving the entire telescope to a new position and then chopping the secondary again between two positions. Subtracting the two chop images at this new telescope position will remove the sky emission but leave the additional telescopic background due to the misalignment; subtracting the result from the chop-subtracted image at the first telescope position will then remove the background. Since the process of moving to a new position is needed to remove the additional background from the telescope, not the sky, it can be done on a much longer timescale. The variation in the telescopic backgrounds occurs on timescales on the order of tens of seconds to minutes, much slower than the variation in the sky emission.

This movement of the entire telescope, on a much longer timescale than chopping, is known as nodding. The two nod positions are usually referred to as nod A and nod B. The distance between the two nod positions is known as the nod throw or the nod amplitude. For FORCAST observations, nods are done every 5 to 30 seconds. The chop-subtracted images at nod position B are then subtracted from the chop-- subtracted images at nod position A. The result will be an image of the region, without the sky background emission or the additional emission resulting from tilting the secondary during the chopping process. The sequence of chopping in one telescope position, nodding, and chopping again in a second position is known as a chop/nod cycle.

Again, because the MIR sky is so bright, deep images of a region cannot be obtained (as they are in the optical) by simply observing the region for a long time with the detector collecting photons continuously. As stated above, the observations require chopping and nodding at fairly frequent intervals. Hence, deep observations are made by "stacking" a series of chop/nod images. Furthermore, MIR detectors are not perfect, and often have bad pixels or flaws. In order to avoid these defects on the arrays, and prevent them from marring the final images, observers employ a technique known as "dithering". Dithering entails moving the position of the telescope slightly with respect to the center of the region observed each time a new chop/nod cycle is begun, or after several chop/nod cycles. When the images are processed, the observed region will appear in a slightly different place on the detector. This means that the bad pixels do not appear in the same place relative to the observed region. The individual images can then be registered and averaged or median-combined, a process that will eliminate (in theory) the bad pixels from the final image.

2.2. Available chopping modes

2.2.1. Symmetric chopping modes: C2N and C2ND

FORCAST acquires astronomical observations in two symmetric chopping modes: two-position chopping with no nodding (C2) and two-position chopping with nodding (C2N). Dithering can be implemented for either mode; two-position chopping with nodding and dithering is referred to as C2ND. The most common observing methods used are C2N and C2ND. C2ND is conceptually very similar the C2N mode: the only difference is a slight movement of the telescope position after each chop/nod cycle.

FORCAST can make two types of C2N observations: Nod Match Chop (NMC) and Nod Perp Chop (NPC). The positions of the telescope boresight, the two chop positions, and the two nod positions for these observing types are shown below (Figures 3 and 4).

2.2.2. C2N: Nod Match Chop (NMC)

In this case, the telescope is pointed at a position half of the chop throw distance away from the object to be observed, and the secondary chops between two positions, one of which is centered on the object. The nod throw has the same magnitude as the chop throw, and is in a direction exactly 180 degrees from that of the chop direction. The final image generated by subtracting the images obtained for the two chop positions at nod A and those at nod B, and then subtracting the results. This will produce three images of the star, one positive and two negative, with the positive being twice as bright as the negatives.

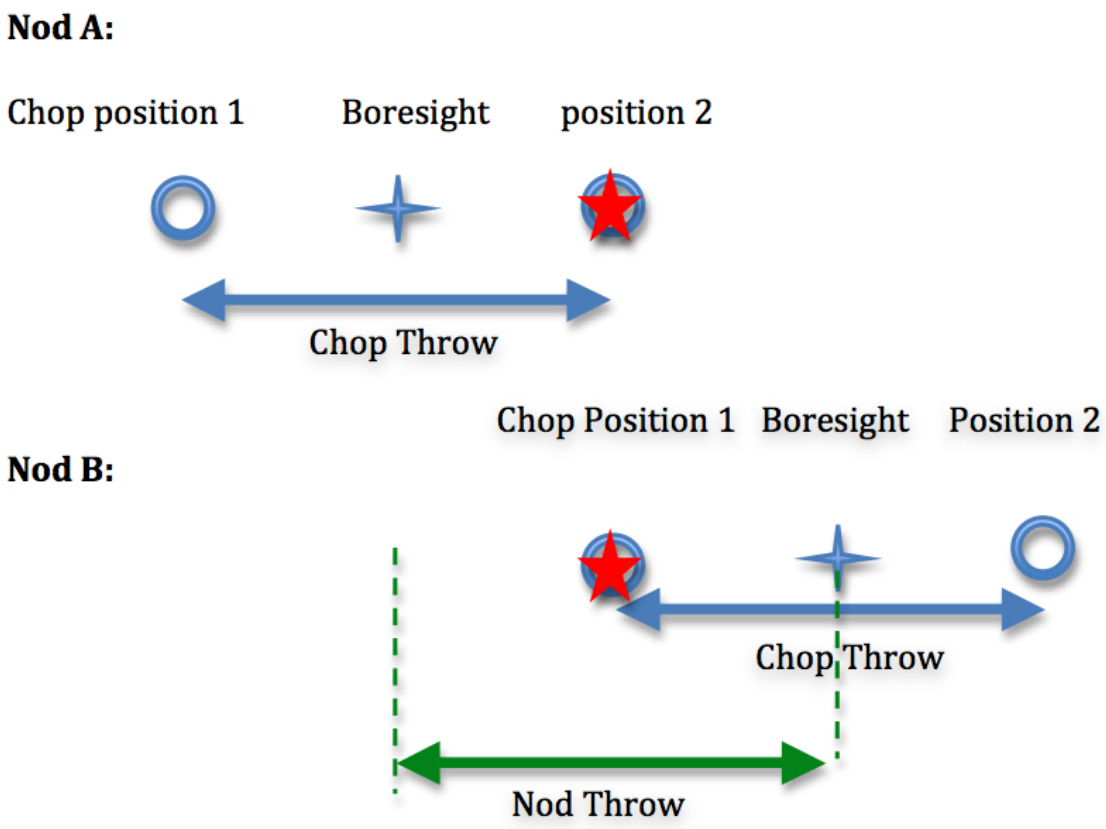

Figure 3: Nod Match Chop mode

2.2.3. C2N: Nod Perp Chop (NPC)

In this case, the telescope is offset by half the nod throw from the target in a direction perpendicular to the chop direction, and the secondary chops between two positions. The nod throw usually (but not necessarily) has the same magnitude as the chop, but it is in a direction perpendicular to the chop direction. The final image is generated by subtracting the images obtained for the two chop positions at nod A and those at nod B, and then subtracting the results.

This will produce four images of the star in a rectangular pattern, with the image values alternating positive and negative.

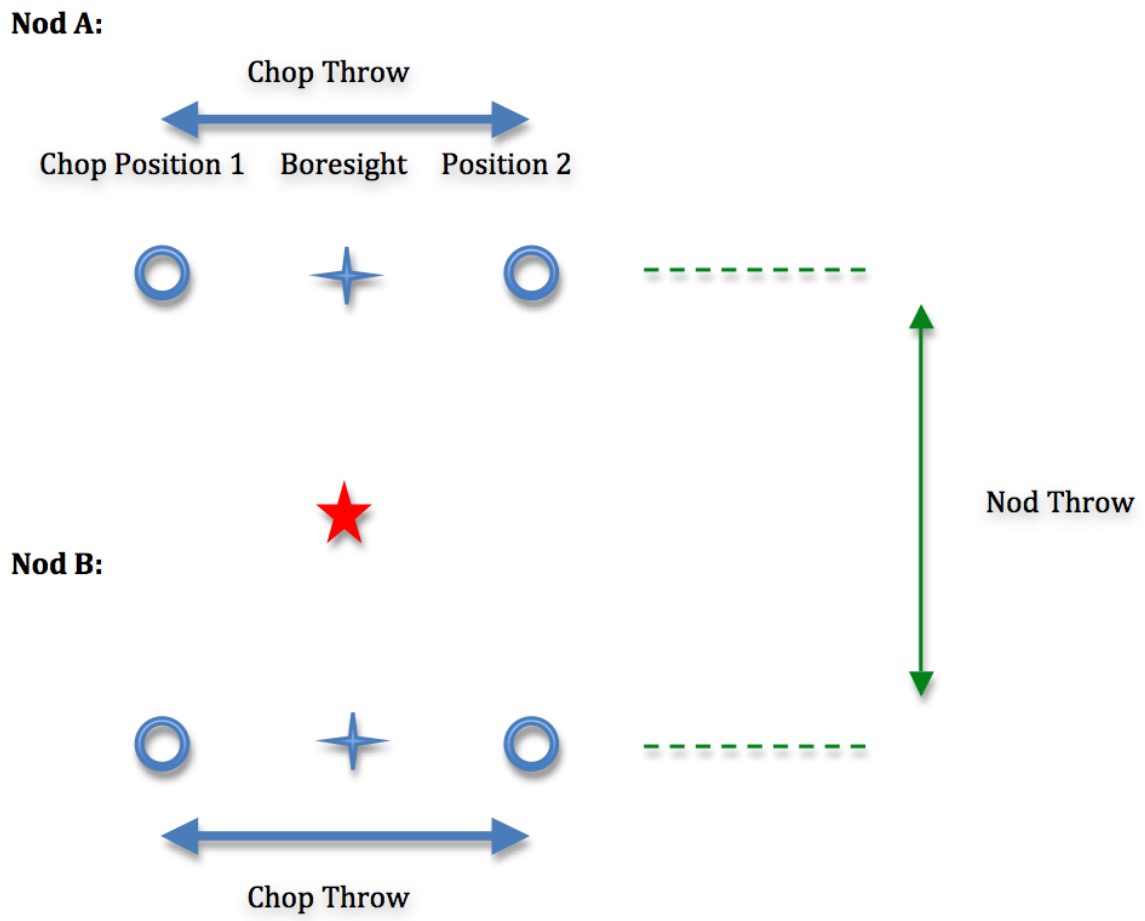

Figure 4: Nod Perp Chop mode

2.2.4. Asymmetrical chopping mode: C2NC2

FORCAST also has an asymmetrical chop mode, known as C2NC2. In this mode, the telescope is first pointed at the target (position A). In this first position, the secondary is aligned with the boresight for one observation and then is tilted some amount (often 180-480 arcseconds) for the second (asymmetrically chopped) observation. This is an asymmetric C2 mode observation. The telescope is then slewed some distance from the target, to some sky region without sources (position B), and the asymmetric chop pattern is repeated. The time between slews is typically 30 seconds.

2.2.5. Nod not related to Chop, Asymmetric Chop: NXCAC (Grism only)

This replaces C2NC2 mode when the GI wants to use C2NC2 mode with grisms **only**. This is ABBA, like C2N mode (not ABA, like C2NC2). The nods are packaged together, so data from this mode will reduce just like the C2N mode. The reason for adding this mode stems from the need to define our large chops and nods in ERF (equatorial reference frame), and dither in SIRF (science instrument reference frame) along the slit.

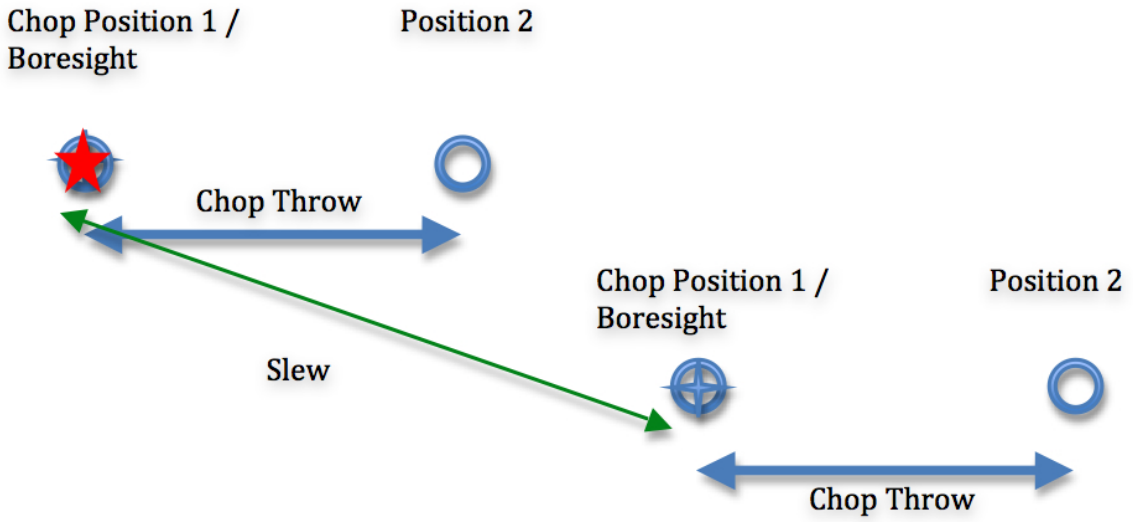

Figure 5: C2NC2 mode

2.3. Calibration files

Calibration files for FORCAST may include dark frames and flat field frames. FORCAST flat field images are either images of blank sky or of the on-board calibration source, which is a hot plate imaged onto the camera pupil. Dark frames may be used to correct the dark current in a flat field, but are not necessary for science frames, since the dark current will be removed when chop/nod subtraction is performed. It is not expected that flats and darks will change significantly over short time scales, so the FORCAST SI team will provide standard flats and darks to be used in every reduction. These will be distributed along with the reduction software code.

3. ALGORITHM DESCRIPTION

3.1. Overview of data reduction steps

Redux applies a number of corrections to each input file, regardless of the chop/nod mode used to take the data. The initial steps used for imaging and grism modes are nearly identical; points where the results or the procedure differ for either mode are noted below. After preprocessing, individual images or spectra of a source must be combined to produce the final data product. This procedure depends strongly on the chop/nod mode.

All raw files are first processed as images, with algorithms developed for the DRIP reduction package. Spectroscopy files then undergo spectral extraction and combination of the resulting one-dimensional spectra, using algorithms from the FSpextool reduction package.

See Figure 6 for a flowchart of all processing steps used by the pipeline.

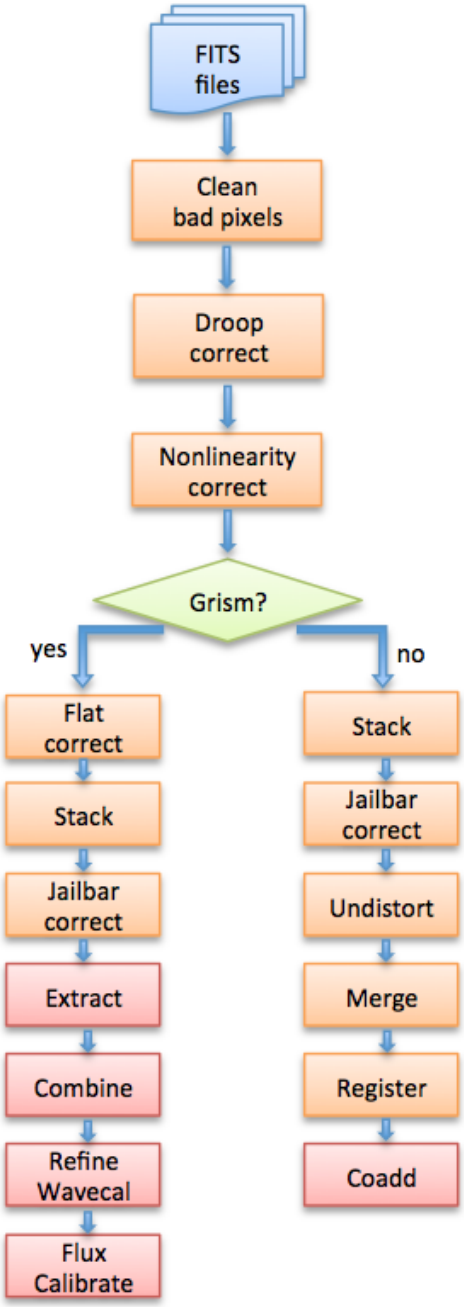

Figure 6: Processing steps for imaging and grism data. Boxes shown in red are algorithms that come from the FSpextool package; all others come from the DRIP package.

3.2. Reduction algorithms

The following subsections detail each of the data reduction pipeline steps:

- Cleaning of bad pixels
- Droop effect correction
- Nonlinearity correction
- Flat field correction
- Background subtraction (chop/nod stacking)
- Jailbar removal (crosstalk correction)
- Optical distortion correction
- Image shift and rotation (imaging only)
- Spectral extraction (grism only)
- Image registration (imaging only)
- Coadding multiple observations

3.2.1. Clean

Bad pixels in the FORCAST arrays take the form of hot pixels (with extreme dark current) or pixels with very different response (usually much lower) than the surrounding pixels. The pipeline minimizes the effects of bad pixels by using a bad pixel mask to identify their locations and then replacing the bad pixels with values derived from the surrounding operational pixels. The DRIP clean function (`drip_clean.pro`) is built around the IDL procedure `MASKINTERP`, written by J. Harrington, which fits a 2-dimensional surface to an aperture in the image centered on the bad pixel(s) while ignoring the bad pixel(s) identified in the mask. `MASKINTERP` then replaces the bad pixels with the corresponding values of the surface fit. `MASKINTERP` is set to use a planar surface with an aperture radius of 6 pixels.

The bad pixel map for both FORCAST channels is currently produced manually, independent of the pipeline. The mask is a 256x256 image with pixel value = 0 for bad pixels and pixel value = 1 otherwise.

3.2.2. Droop correction

The FORCAST arrays and readout electronics exhibit a linear response offset caused by the presence of a signal on the array. This effect is called ‘droop’ since the result is a reduced signal. Droop results in each pixel having a reduced signal that is proportional to the total signal in the 15 other pixels in the row read from the multiplexer simultaneously with that pixel. The effect, illustrated in Figure 7, is images with periodic spurious sources spread across the array rows. The droop correction removes the droop offset by multiplying each pixel by a value derived from the sum of every 16th pixel in the same row all multiplied by an empirically determined offset fraction: droopfrac = 0.0035. The pipeline gets droopfrac from the dripconf.txt configuration file; the droop correction algorithm can be found in the `drip_droop.pro` function.

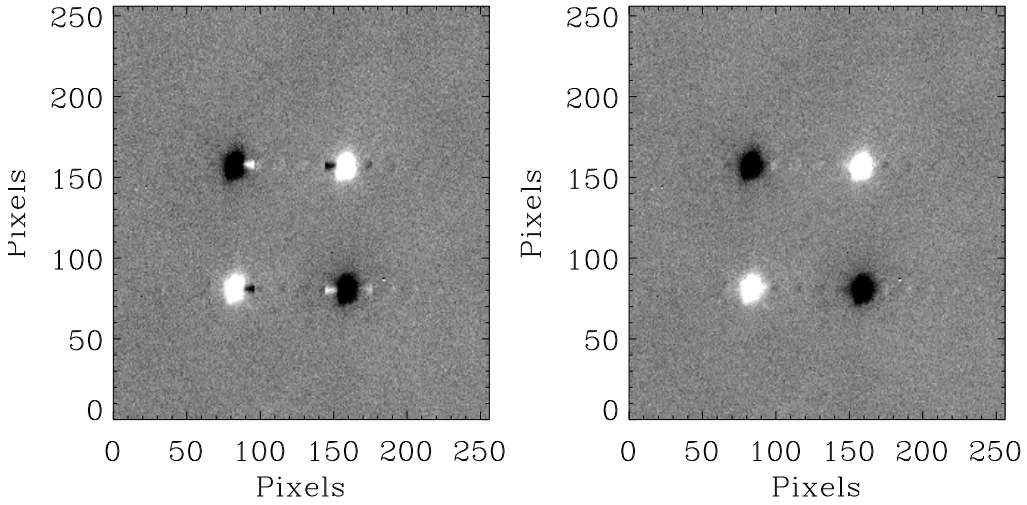

Figure 7: Background-subtracted FORCAST images of a star with droop (left) and with the droop correction applied (right).

3.2.3. Nonlinearity correction

In principle, the response of each of the pixels in our detector arrays should be linear with incident flux. In practice, the degree to which detector linearity depends on the level of charge in the wells relative to the saturation level. Empirical tests optimizing signal-to-noise indicate that signal levels in the neighborhood of 60% of full well for a given detector capacitance in the FORCAST arrays have minimal departures from linear response and optimal signal-to-noise. For a given background level we can keep signal levels near optimal by adjusting the detector readout frame rate and detector capacitance. Since keeping signals near 60% of saturation level is not always possible or practical, we have measured response curves (response in analog-to-digital units (ADU) as a function of well depth for varying background levels) that yield linearity correction factors. These multiplicative correction factors linearize the response for a much larger range of well depths (~15% – 90% of saturation). The linearity correction is applied globally to FORCAST images prior to background subtraction. The pipeline first calculates the background level for a sub-image defined in the `dripconf.txt` configuration file (`drip_background.pro`). This level is then used to look up the linearity correction factor (also located in `dripconf.txt`) and the pipeline applies the correction factor to the entire image. The code for this correction is in `drip_imgnonlin.pro`.

This global-level correction should remove non-linearity effects to first order, but it is also possible that non-linearity effects may vary from one pixel to another. The DRIP package includes an algorithm for applying a non-linearity correction individually to each pixel (`drip_nonlin.pro`). However, these corrections have not yet been determined for the FORCAST array, so the pixel-to-pixel non-linearity correction is not currently applied.

3.2.4. Flat field correction

The flat field correction should, in principle, remove pixel-to-pixel variations in gain, dark current, and responsivity and also variations in illumination across the detector array due to uneven illumination or image distortion. The pipeline (`drip_flatsum.pro`) uses raw

FORCAST flat fields to produce a master flat: a single image plane containing linear offsets. The offsets are calculated by dividing the image (pixel by pixel) by the median pixel value of the image. Thus a perfectly flat image would have a master flat containing values of 1.0. The flat field correction algorithm (`drip_flat.pro`) divides raw image frames by the master flat image.

The FORCAST data reduction pipeline produces a master flat image from raw flat files differently depending on the number of frames in the raw flat field data cube. A cube of four image planes is assumed to be composed of a hot blackbody source (two frames) and a cool blackbody source (two frames), which are differenced when generating the master flat field image. This is the default format produced by invoking the 'FLAT' button/function in the FORCAST data acquisition software. Any other number of frames (including a single frame) are assumed to be composed of identical blackbody sources and are averaged when generating the master flat field image.

In the mid-infrared, it is often difficult to produce a flat field frame that improves photometric precision, rather than worsens it. The imaging flats presently available for FORCAST do not improve image quality, so flat field correction is currently disabled for the imaging mode of the pipeline.

For spectroscopic modes, the flat field is normalized by fitting a low-order 2D surface to the image, . The flat field is applied via the `drip_flat` function. Note that versions of the FORCAST data reduction pipeline prior to Redux 1.0.1 normalized flats by simply dividing by the median value of the pixels: data reduced with flats normalized in this way should not be combined with data reduced by Redux 1.0.1 and later.

3.2.5. Background subtraction (chop/nod stacking)

Background subtraction is accomplished by subtracting chopped image pairs and then subtracting nodded image pairs. This happens in the `drip_stack.pro` function. For C2N/NPC imaging mode with chop/nod on-chip (ie. chop throws smaller than the FORCAST field of view), the four chop/nod images in the raw data file are reduced to a single stacked image frame with a pattern of four background-subtracted images of the source, two of them negative. For chop/nod larger than the FORCAST field of view the raw files are reduced to a single frame with one background-subtracted image of the source. For the C2N/NPC spectroscopic mode, either the chop or the nod is always off the slit, so there will be two traces in the subtracted image: one positive and one negative. If the chop or nod throw is larger than the field of view, there will be a single trace in the image.

In the case of the C2N/NMC mode for either imaging or spectroscopy, the nod direction is the same as the chop direction with the same throw so that the subtracted image frame contains three background-subtracted images of the source. The central image or trace is positive and the two outlying images are negative. If the chop/nod throw is larger than the FORCAST field of view, there will be a single image or trace in the image.

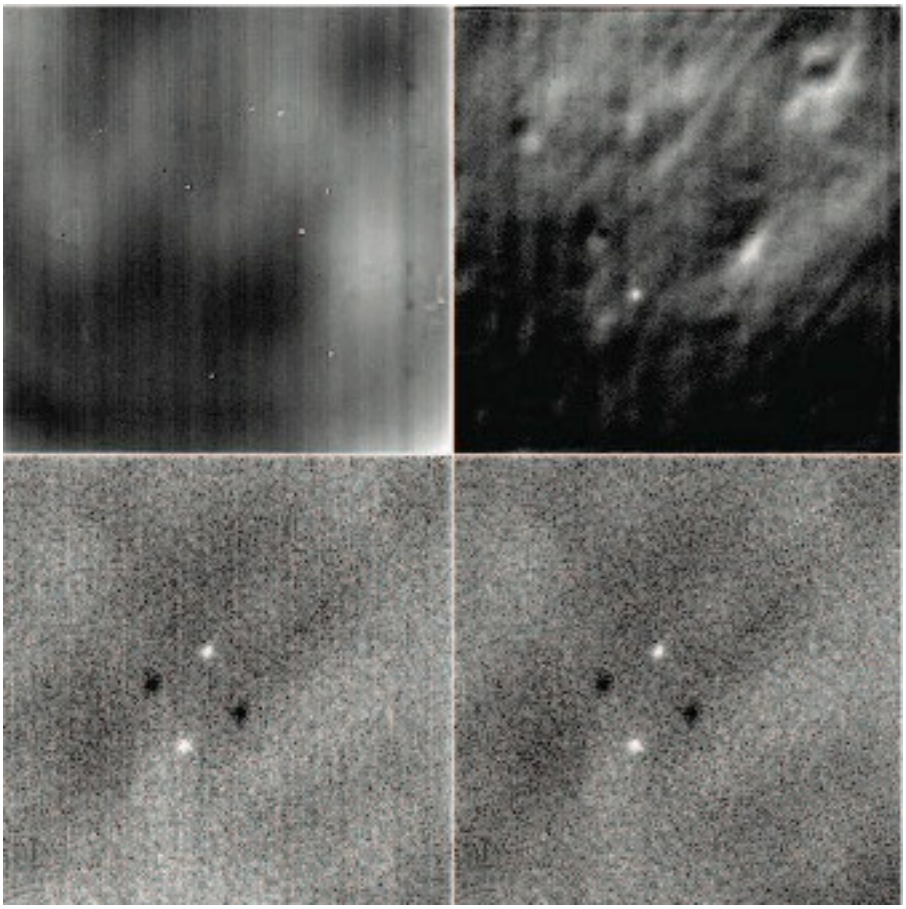

Figure 8. Images at four stages of background subtraction: raw frames (upper left), chop-subtracted (upper right), chop/nod-subtracted for a single nod sequence (nod positions AB, lower left), and chop/nod-subtracted for a full nod cycle (nod positions ABBA, lower right).

C2NC2 raw data sets for imaging or spectroscopy consist of a set of 5 FITS files, each with 4 image planes containing the chop pairs for both the on-source position (position A) and the blank sky position (position B). The four planes can be reduced in the same manner as any C2N image, by first subtracting chopped image pairs for both and then subtracting nodded image pairs. The nod sequence for C2NC2 is $A_1B_1A_2A_3B_2A_4A_5B_3$, where the off-source B nods are **shared** between some of the files (shared B beams shown in bold):

File 1 = $A_1\mathbf{B_1}$
File 2 = $\mathbf{B_1}A_2$
File 3 = $A_3\mathbf{B_2}$
File 4 = $\mathbf{B_2}A_4$
File 5 = A_5B_3

At this point, the background in the chop/nod-subtracted stack should be zero, but if there is a slight mismatch between the background level in the individual frames, there may still remain some small residual background level. After stacking, the pipeline estimates this residual

background by taking the mode of the image data in a central section of the image, then subtracts this level from the stacked image.

The last step in the imaging stack pipeline step is to convert pixel data from analog-to-digital units (ADU) per frame to mega-electrons per second (Me^-/s) using the gain and frame rate used for the observation.

For grism data, this conversion is applied as well. Then, individual frames taken at the same dither position may be combined together to increase the signal-to-noise in the two-dimensional spectral image. This aids in the extraction of faint sources, but is not necessary if the source is bright. This combination will generally be done for science targets, but not for telluric standards.

3.2.6. Jailbar removal (Crosstalk correction)

The FORCAST array readout circuitry has a residual, or latent, signal that persists when pixels have high contrast relative to the surrounding pixels. This can occur for bad pixels or for bright point sources. This residual is present not only in the affected pixels, but is correlated between all pixels read by the same one of sixteen multiplexer channels. This results in a linear pattern of bars (every 16 pixels the pattern repeats) known as "jailbars" in the background-subtracted (stacked) images. Jailbars can interfere with subsequent efforts to register multiple images since the pattern can dominate the cross-correlation algorithm used in image registration. The jailbars can also interfere with photometry in images and with spectral flux in spectroscopy frames. Particularly troublesome is the fact that the jailbar intensity appears to depend on the intensity of signal on the array so that flat-fielding it out will not work.

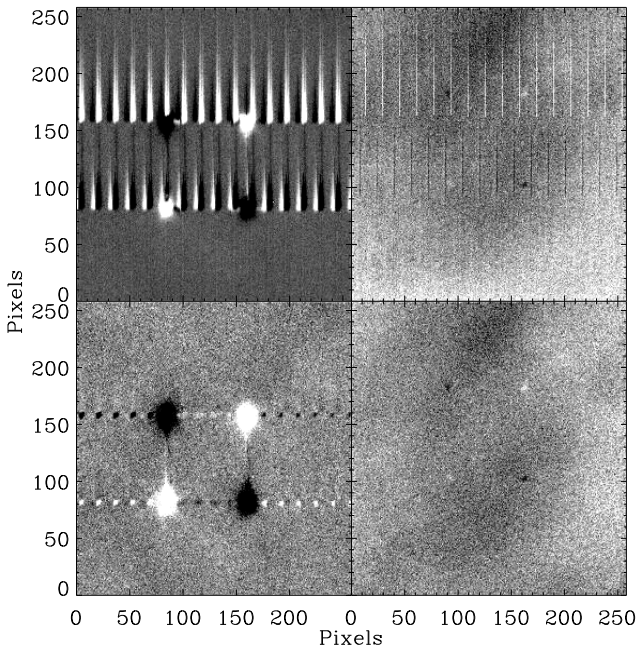

Figure 9. Cross talk correction for a bright point source on left, and faint source on right. Images on the top are before correction; images on the bottom are after correction.

The pipeline attempts to remove jailbar patterns from the background-subtracted images by replacing pixel values by the median value of pixels in that row that are read by the same multiplexer channel (i.e. every 16th pixel in that row starting with the pixel being corrected). The jailbar pattern is located by subtracting a 1-dimensional (along rows) median filtered image from the raw image.

The jailbar correction (`drip_jbclean.pro`) can be toggled on or off by setting the 'JBCLEAN' keyword in the `dripconf.txt` file prior to running the pipeline: `jbclean = 'MEDIAN'` switches the correction ON, `jbclean = 'N'` switches the correction OFF. There is currently no way to automatically determine, prior to data reduction, whether the jailbar correction will be necessary. If the jailbar correction is desired, it will be applied at the end of the stack step, to the chop/nod-subtracted image.

3.2.7. Optical distortion correction

The FORCAST optical system introduces anamorphic magnification and barrel distortion in the images. The distortion correction uses pixel coordinate offsets for a grid of pinholes imaged in the lab and a 2D polynomial warping function (IDL's `polywarp.pro`, and `poly_2d.pro`) to resample the 256x256 pixels to an undistorted grid. The resulting image is 262x247 pixels with image scale 0.768"/pixel for a corrected field of view of 3.4x3.2 arc minutes. The distortion-corrected image is centered in a 656x656 pixel array to accommodate the distortion correction and to make room for subsequent shifting and adding of chop/nod images and for image rotation prior to the final coaddition step of the reduction process.

There is no distortion correction for the grism mode since the extracted spectra have a wavelength and spatial calibration applied directly to the array rows.

3.2.8. Image shift and rotation (merging)

The stack step of the pipeline in imaging mode produces images with multiple positive and negative source images depending on the chop/nod mode used for data acquisition. These positive and negative sources must be merged (`drip_merge.pro`) by copying, shifting, and re-combining the image. The final image must then be rotated to the nominal sky angle (NORTH = UP, EAST = LEFT in the displayed image).

The merge pipeline step (implemented in `drip_merge.pro`) makes a number of copies of the stacked image, shifts them by the chop and nod throws used in data acquisition, and adds or subtracts them (depending on whether the image is a positive or negative background-subtracted image). DRIP can use three different methods for registration in the merge process (selected in the `dripconf.txt` configuration file with the keyword `CORMERGE`):

- centroid of the brightest point source in the stacked images (`CORMERGE='CENT'`)
- cross-correlation, usually best for extended or nebulous sources (`CORMERGE='COR'`)
- chop/nod data from the FITS header (`CORMERGE='N'`)

Then, the merged image is rotated using the `SKYANGLE` FITS keyword value. Thus, the final merged image consists of a positive image of the source surrounded by a number of positive and

negative residual source images left over from the merge shift and add process. The central image is the source to use for science.

For the NPC imaging mode with chop/nod amplitude smaller than the field of view, the stack step produces a single stacked image frame with a pattern of four background-subtracted images of the source, two of them negative. The merge step makes four copies of the stacked frame, then shifts each using the selected algorithm. It adds or subtracts each copy, depending on whether the source is positive or negative.

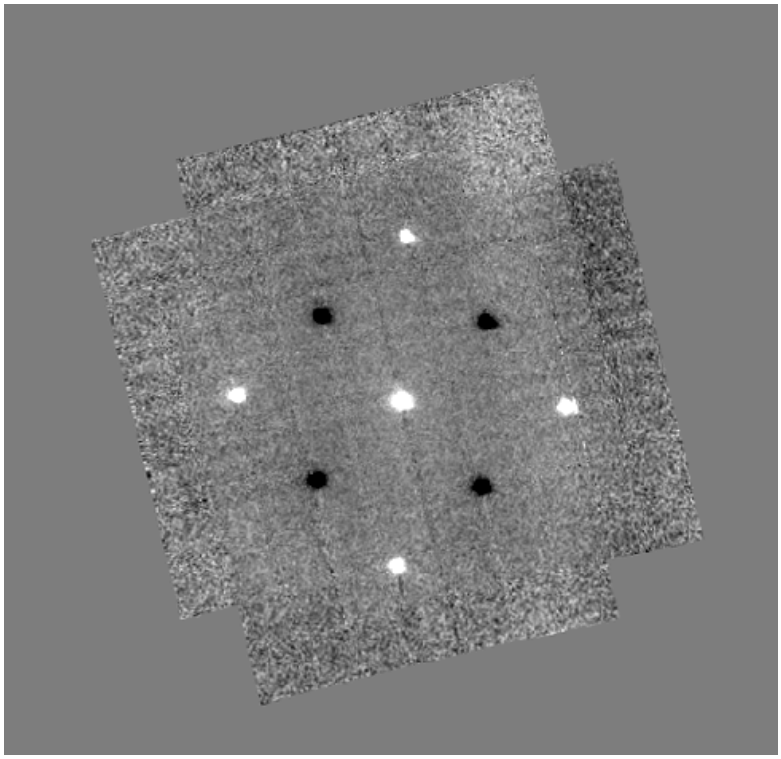

Figure 10. NPC observation after merging. Only the central source should be used for science; the other images are artifacts of the stacking and merging procedure.

For the NMC imaging mode with chop/nod amplitude smaller than the field of view, the stacked image contains three background-subtracted sources, two negative, and one positive. The positive source has double the flux of the negative ones, since the source falls in the same place on the detector for two of the chop/nod positions. The merge step for this mode makes three copies of the image, shifts the two negative sources on top of the positive one, then subtracts them.

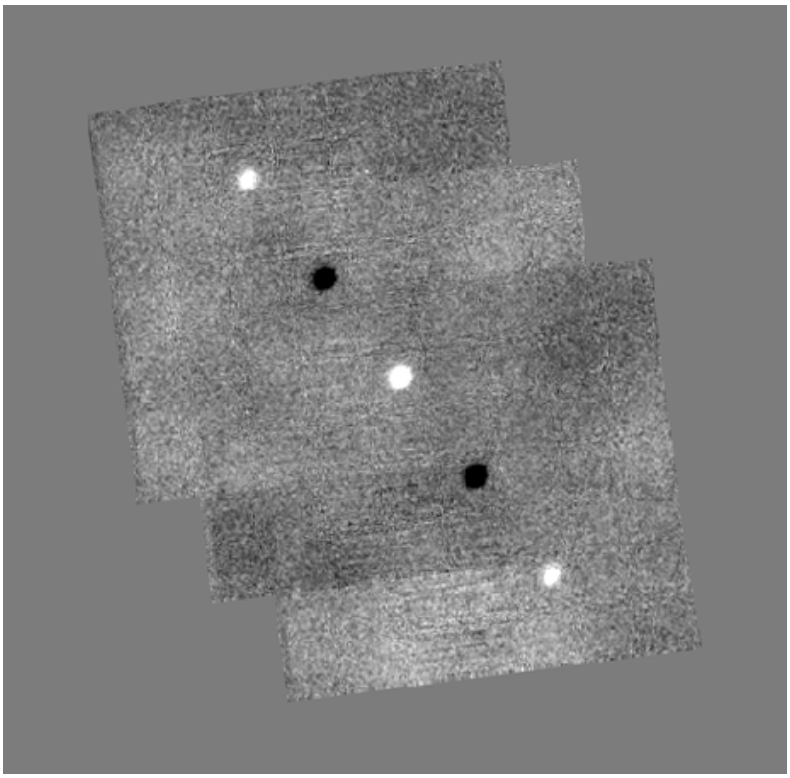

Figure 11. NMC observation after merging. As with NPC, only the central source should be used for science.

While performing the merge, the locations of overlap for the shifted images are recorded. For NPC mode, the final merged image is normalized by dividing by the number of overlapping images at each pixel. For NMC mode, because the source is doubled in the stacking step, the final merged image is divided by the number of overlapping images, plus one. In the nominal case, if all positive and negative sources were found and coadded, the signal in the central source, in either mode, should now be the average of four observations of the source. If the chop or nod was relatively wide, however, and one or more of the extra sources were not found on the array, then the central source may be an average of fewer observations.

For either NPC or NMC imaging modes, with chop/nod amplitude greater than half of the array, there is no merging to be done, as the extra sources are off the detector. However, for NMC mode, the data is still divided by 2 to account for the doubled central source. For C2NC2 mode, the chops and telescope moves-to-sky are always larger than the FORCAST field of view, so merging is never required. Merging is also not required for spectroscopy observations, as the spectra in the stacked image are extracted separately, and then coadded directly.

3.2.9. Spectral extraction

The FSpextool spectral extraction algorithms used by Redux offer two different extraction methods depending on the nature of the target source, as defined by the `SRCTYPE` FITS keyword. For point sources, Redux uses an optimal extraction algorithm, described at length in the Spextool paper (see the Other Resources section, below, for a reference). For extended

sources, Redux uses a standard summing extraction, which simply sums the flux over an aperture, which can be specified directly by the user or determined automatically from the spatial distribution of the flux over the slit (the spatial profile).

For the NPC grism mode, with chop/nod amplitude less than the field of view, there will be a positive and a negative spectral trace in the stacked image. Redux extracts both, multiplying the negative spectrum by -1 to make it positive. It then merges the spectra by coadding them and dividing by 2.

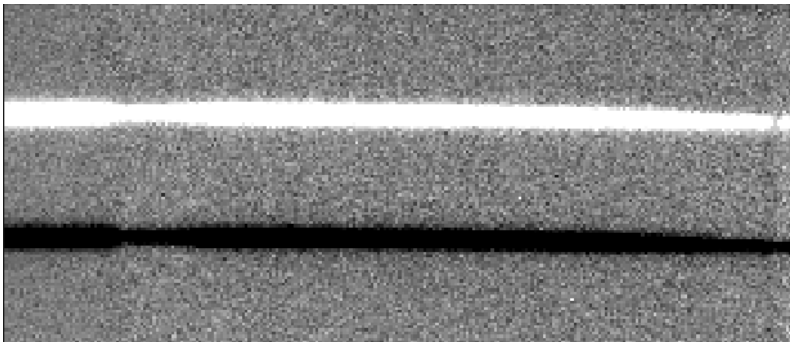

Figure 12. NPC spectrum after stacking. Both spectra will be extracted.

For the NMC grism mode, with chop/nod amplitude less than the field of view, and chopping or nodding along the slit, there will be a positive and two negative spectral traces in the stacked image. In this case, Redux extracts all three spectra, multiplying the negative ones by -1. It then merges the spectra by coadding them and dividing by four (to account for the doubled central source). If the chop/nod amplitude for the NMC mode is larger than the field of view or it is chopping off the slit, there will be only a single spectral trace. In this case, Redux extracts this spectrum, then simply divides it by two to account for the doubled source.

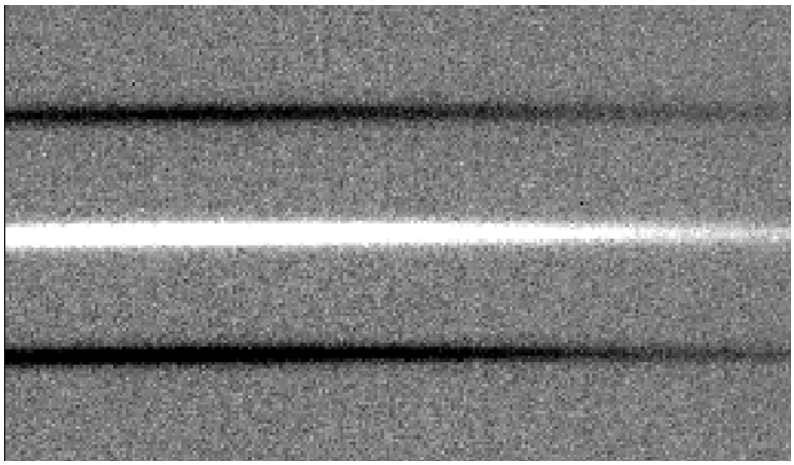

Figure 13. NMC spectrum after stacking. All three spectra will be extracted.

All other grism modes produce a single positive spectral trace in the stacked image, which Redux extracts directly.

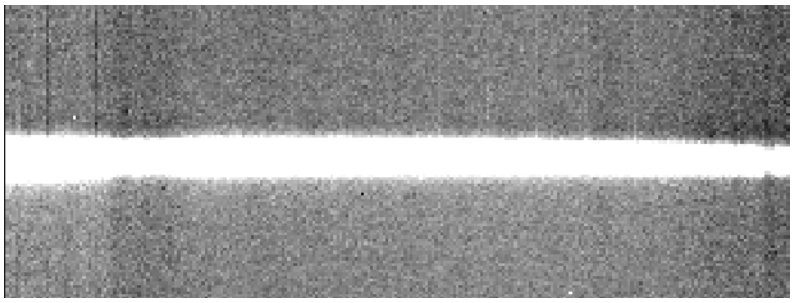

Figure 14. NMC spectrum with wide chop after stacking.

All spectral extractions require a 2D wavelength calibration map, identifying the wavelength associated with each pixel in the array, to extract spectra along lines of constant wavelength. This simultaneously corrects for any distortion in the spatial or spectral directions, and wavelength-calibrates the output spectrum. We generate wavelength calibration maps from identifications of sky emission and telluric absorption lines and a polynomial fit to centroids of those features in pixel space for each row (i.e. along the dispersion direction). Specification of a wavelength calibration is an interactive process, but application of the derived wavelength calibration is automatic and part of the data reduction pipeline. We expect the wavelength calibration to be good within approximately one pixel.

3.2.10. Image registration

In order to combine multiple imaging observations of the same source, each image must be registered to a reference image, so that the pixels from each image correspond to the same location on the sky. This step is performed by `drip_register.pro`. This algorithm uses the same three options for registration of images as the merge step (centroid, cross-correlation, or FITS header data), which are set in the `dripconf.txt` configuration file with the keyword `CORCOADD`. The first image in the set is treated as the reference image; the algorithm uses header data to shift this image to account for any initial dither offset. For all subsequent images, the specified algorithm is used to find the shift required to register it to the first image. The interpolation order of the shift may be set in the configuration file, using the keyword SHIFTORD. SHIFTORD=0 indicates that integer pixel shifts should be used, SHIFTORD=1 uses bilinear interpolation to do sub-pixel shifts, and SHIFTORD=3 (the default) uses a cubic interpolation to do sub-pixel shifts.

3.2.11. Coadding multiple observations

The final pipeline step is coaddition of multiple observations of the same source with the same instrument configuration and observation mode.

For imaging, the image combination is performed by an FSpextool algorithm that allows rejection of outlying values. The default statistic is a robust mean.

For spectroscopy, the FSpextool algorithm that combines the individual spectra scales them to a median value, by default, before combining them with a robust weighted mean statistic. It may also optionally correct each spectrum's spectral shape to the first spectrum in the set before combination.

3.2.12. Grism correction

For spectroscopy, the final produced spectrum may have additional corrections for instrumental calibration. The wavelength calibration may be adjusted by identifying a feature in the extracted spectrum: from this identification, a zero-point shift is calculated and applied to the output spectra. Finally, the spectrum will be corrected for atmospheric transmission and instrumental response, producing a flux-calibrated spectrum: see the section on flux calibration, below.

3.3. Uncertainties

Redux calculates the expected uncertainties for raw FORCAST data as a variance image associated with the input data. It then propagates this variance image along with the data through each processing step. This variance image is written to disk as an extra plane in all FITS images produced at intermediate steps. For the 1D extracted spectra written to disk, the uncertainty is saved as a standard deviation (the square root of the propagated variance) in an extra dimension in the image file.

FORCAST raw data is recorded in units of ADU per coadded frame. The variance associated with the *i,j*th pixel in this raw data is calculated as:

$$V_{ij} = \frac{N_{ij} \cdot \beta_g}{FR \cdot t \cdot g} + \frac{RN^2}{FR \cdot t \cdot g^2}$$

where N is the raw ADU per frame in each pixel, β_g is the excess noise factor, FR is the frame rate, t is the integration time, g is the gain, and RN is the read noise in electrons. The first term corresponds to the Poisson noise, and the second to the read noise. Since FORCAST data are expected to be background-limited, the Poisson noise term should dominate the read noise term.

The variance for the standard extraction is a simple sum of the variances in each pixel within the aperture. For the optimal extraction algorithm, the variance on the *i*th pixel in the extracted spectrum is calculated as:

$$V_i = \sum_j \frac{1}{{P_{ij}}^2 \cdot V_{ij}}$$

where P_{ij} is the spatial profile, V_{ij} is the variance at each pixel, and the sum is over all pixels j in the extraction aperture. This equation comes from the Spextool paper, describing optimal extraction.

3.4. Other Resources

For more information on how to run the FSpextool interactive tools (*xspextool, ximgtool*, *xvspec*, *xwavecal2d, xcombspec*, *xtellcor*, and *xcleanspec*), see the help files distributed with the FSpextool code, under *fspextool/Helpfiles*.

For more information about the Redux, DRIP, and FSpexool software architecture, see the Redux Developer's Manual, located in *redux/helpfiles*.

For more information on the reduction algorithms used in FSpextool, see the Spextool papers:
Spextool: A Spectral Extraction Package for SpeX, a 0.8-5.5 micron Cross-Dispersed Spectrograph Michael C. Cushing, William D. Vacca and John T. Rayner (2004, PASP 116, 362).
A Method of Correcting Near-Infrared Spectra for Telluric Absorption William D. Vacca, Michael C. Cushing and John T. Rayner (2003, PASP 115, 389).
Nonlinearity Corrections and Statistical Uncertainties Asociated with Near-Infrared Arrays William D. Vacca, Michael C. Cushing and John T. Rayner (2004, PASP 116, 352).

4. FLUX CALIBRATION

4.1. Imaging Flux Calibration

The reduction process generates Level 2 images with data values in units of Me-/sec. After Level 2 imaging products are generated, the pipeline derives the flux calibration factors (in units of Me-/s/Jy) and applies them to each image. The calibration factors are derived for each FORCAST filter configuration from observations of calibrator stars. Standards are observed in a specific instrument configuration (filter and dichroic) as well as in an observation configuration (altitude and zenith).

After the calibration factors have been derived, they are written to the headers of the Level 2 merged and coadded files in the FITS keyword CALFCTR. It should be noted that nothing is done to the actual data values in the images between Level 2 and Level 3; the Level 3 data values still have units of Me^-/s. The only differences between the Level 2 and Level 3 data products are the addition of keywords to the headers of the Level 3 files containing the calibration factor, its uncertainty, and the reference wavelength. Conversion to Jy must be carried out by the user, by dividing the values measured in the image by the calibration factor given in the header.

4.1.1. Reduction Steps

The calibration is carried out in several steps that in practice are performed iteratively. The first step consists of measuring the photometry of all the standard stars for a specific mission. Calibration factors are then derived from the measured photometry and the known fluxes of the standards. For each object, the calibration factors from all the standards on a flight are adjusted to account for the differences between the target airmass and altitude and those of the standards, and then averaged. The pipeline then inserts this value and its uncertainty into the headers of the Level 2 data files. After all calibration factors are derived for a mission, the final step requires studying the calibration values. The calibration factor for each instrument configuration should be consistent within a mission and even between consecutive missions. Values that are not consistent may come from bad observations of a standard star. Bad standard stars are removed, and the reduction process is repeated.

The determination of the calibration factors begins with the analysis of the reduced Level 2 images of the standard stars observed on a given flight. Photometry is done on these images using an aperture of 12 pixels and a background region of 25-35 pixels. The measured count rate is then divided by the predicted flux in Jy for each star in each filter. The predicted fluxes were computed by multiplying the model stellar spectrum by the overall filter+instrument+telescope+atmosphere response curve and integrating over the filter passband to compute the mean flux in the band. The adopted filter throughput curves are those provided by the vendor or measured by the FORCAST team, modified to remove regions (around 6-7 microns and 15 microns) where the values were contaminated by noise. The instrument throughput is calculated by multiplying the transmission curves of the entrance window, dichroic, internal blockers, and mirrors, and the detector quantum efficiency. The telescope throughput value is assumed to be constant (85%) across the entire FORCAST wavelength range. The atmospheric transmission is computed using the ATRAN code (Lord 1992) for a range of observatory altitudes (corresponding to a range of overhead precipitable water vapor values) and telescope elevations.

For most of the standard stars, the adopted stellar models were obtained from the *Herschel* calibration group and consist of high-resolution theoretical spectra, generated from the MARCS models (Gustafsson et al. 1975, Plez et al. 1992), scaled to match absolutely calibrated observational fluxes (Dehaes et al. 2011). For β UMi we scaled the model by a factor of 1.18 in agreement with the results of the *Herschel* calibration group (J. Blommaert, private communication; the newer version of the model from the *Herschel* group has incorporated this factor).

Using the measured photometry of the standard, N_e^{std}(in Me/s), and the predicted mean fluxes of the standards in each filter, $\langle F_\nu^{std} \rangle$ (in Jy), the flux of a target object is

$$F_\nu{}^{nom,obj}(\lambda_{ref}) = \frac{N_e^{obj}}{C}$$

where $N_e^{obj}$is the count rate in Mega-electrons/s detected from the source, C is the calibration factor (Me/s/Jy), and $F_\nu{}^{nom,obj}(\lambda_{ref})$ is the flux in Jy of a nominal, "flat spectrum" source (for which $F_\nu \sim \nu^{-1}$) at a reference wavelength λ_{ref}. The calibration factor, C, is computed from

$$C = \frac{N_e^{obj}}{F_\nu^{nom,obj}(\lambda_{ref})} = \frac{N_e^{std}}{\langle F_\nu^{std} \rangle} \frac{\lambda_{piv}^2}{\langle \lambda \rangle \lambda_{ref}} \frac{R_\lambda^{obj}/R_\lambda^{ref}}{R_\lambda^{std}/R_\lambda^{ref}}$$

with an uncertainty given by

$$\left(\frac{\sigma_C}{C}\right)^2 = \left(\frac{\sigma_{N_e^{std}}}{N_e^{std}}\right)^2 + \left(\frac{\sigma_{\langle F_\nu^{std} \rangle}}{\langle F_\nu^{std} \rangle}\right)^2 .$$

Here, $\lambda_{piv}$is the pivot wavelength of the filter, $\langle \lambda \rangle$ is the mean wavelength of the filter, and the ratio $R_\lambda / R_\lambda^{ref}$ accounts for differences in system response (transmission) between the actual observations and those for a 'reference' altitude of 41K and a telescope elevation of 45°.

The values of C, σ_C , and λ_{ref} are written into the headers of the Level 3 data as the keywords CALFCTR, ERRCALF, and LAMREF, respectively. The reference wavelength λ_{ref} for these observations was taken to be the mean wavelengths of the filters, $\langle \lambda \rangle$. Note that σ_C currently

assumes no uncertainty on the stellar models and the values of $\left\langle F_\nu^{std} \right\rangle$. The uncertainties on the stellar model fluxes are expected to be on the order of 5-10% (Dehaes et al. 2011). If we assume that the only uncertainties in the values of $\left\langle F_\nu^{std} \right\rangle$ are those arising from the theoretical models, and adopt a 10% value, we have $\sigma_C \approx 0.1$C, because in general $\sigma_{N_e^{std}} / N_e^{std} << 0.1$. Based on the variations seen in the calibration factors across multiple flights we estimate the overall statistical uncertainty in our flux calibrations is ~6% (see Herter et al. 2013).

Each observation of a standard provided a value of the calibration factor in the various filters. The values of C were examined across all of the Cycle 1 flights to check for consistency. Discrepant values signaled problems with the standard star data and those images were then excluded from the calibration process.

An observer often wishes to determine the true flux of an object at the reference wavelength, $F_\nu^{obj}(\lambda_{ref})$, rather than the flux of an equivalent nominal, "flat spectrum" source. To do this, we define a color correction *K* such that

$$K = \frac{F_\nu^{nom,obj}(\lambda_{ref})}{F_\nu^{obj}(\lambda_{ref})},$$

where $F_\nu^{nom,obj}(\lambda_{ref})$ is the flux density one obtained by measurement on a data product. Divide the measured values by *K* to obtain the "true" flux density. In terms of the wavelengths defined above,

$$K = \frac{\langle\lambda\rangle\lambda_{ref}}{\lambda_{piv}^2} \frac{\langle F_\nu^{obj}\rangle}{F_\nu^{obj}(\lambda_{ref})}.$$

For most filters and spectral shapes, the color corrections are small (<10%). Tables listing K values and filter wavelengths are available from the SOFIA website.

4.2. Spectrophotometric Flux Calibration

The common approach to characterizing atmospheric transmission for ground-based infrared spectroscopy is to obtain, for every science target, similar observations of a spectroscopic standard source with as close a match as possible in both airmass and time. Such an approach is not practical for airborne observations, as it imposes too heavy a burden on flight planning and lowers efficiency of science observations. Therefore, we employ a calibration plan that incorporates a few observations of a calibration star per flight and a model of the atmospheric absorption for the approximate altitude and airmass at which the science objects were observed. Telluric absorption models have been computed, using ATRAN, for the entire set of FORCAST grism passbands for every 1000 feet of altitude between 38K and 43K feet, and for every 5 degrees of zenith angle between 30 and 70 degrees. These values correspond to the typical observing limits of SOFIA. Instrumental response curves have been generated for each grism and slit combination from observations of standard stars and stellar models provided by the Herschel Calibration Program.

Flux calibration of FORCAST grism data for a science target is currently carried out in a two-step process:

1. For any given observation of a science target, the closest telluric model (in terms of

altitude and airmass of the target observations) is selected and then smoothed to the observed resolution and sampled at the observed spectral binning. The observed spectrum is then divided by the smoothed and re-sampled telluric model.
2. The telluric-corrected spectrum is then divided by the response function corresponding to the observed instrument mode to convert Mega-electrons/s to Jy at each pixel.

In order to account for any wavelength shifts between the models and the observations, an optimal shift is estimated from the peak of the cross-correlation of the observed spectrum and the correction curves. At each step the correction curve is then shifted and the observed spectrum is then divided by the result.

Analysis of the calibrated spectra of standard stars observed during Cycle 2 indicates that the average RMS deviation over the G063, G227, and G329 grism passbands between the calibrated spectra and the models is on the order of ~5%. For the G111 grism, the average RMS deviation is found to be on the order of ~10%; the larger deviation for this grism is due primarily to the highly variable ozone feature at 9.6 microns, which the ATRAN models are not able to reproduce accurately. The Level 3 data product for any grism includes the calibrated spectrum and an error spectrum that incorporates these RMS values. The adopted telluric absorption model and the instrumental response functions are also provided.

As for any slit spectrograph, highly accurate absolute flux levels from FORCAST grism observations (for absolute spectrophotometry, for example) require additional photometric observations to correct the calibrated spectra for slit losses that can be variable (due to varying image quality) between the spectroscopic observations of the science target and the calibration standard.

5. DATA PRODUCTS

5.1.1. Filenames

Output files from Redux are named according to the convention:
FILENAME = F[flight]_FO_IMA|GRI_AOR-ID_SPECTEL1|SPECTEL2_Type_FN1[-FN2],
where flight is the SOFIA flight number, FO is the instrument identifier, IMA or GRI specifies that it is an imaging or grism file, AOR-ID is the 8 digit AOR identifier for the observation, SPECTEL1|SPECTEL2 is the keyword specifying the filter or grism used, Type is three letters identifying the product type (listed in Tables 5 and 6, below), FN1 is the file number corresponding to the input file. FN1-FN2 is used if there are multiple input files for a single output file, where FN1 is the file number of the first input file and FN2 is the file number of the last input file.

5.1.2. Pipeline Products

Table 5 and 6 lists all intermediate products generated by Redux for imaging and grism modes, in the order in which they are produced. By default, for imaging, the undistorted, merged, registered, and coadded products are saved; for grism, the stacked, mrgspec, combspec, and calspec products are saved. All products can be saved by specifying the appropriate option in either the automatic or interactive modes.

Step	Description	PRODTYPE	Identifier	Saved by default?	Size
Clean	Bad pixels cleaned	cleaned	CLN	N	256x256x2
Droop Correct	Corrected for droop effect	drooped	DRP	N	256x256x2
Nonlinearity Correct	Corrected for detector nonlinearity	linearized	LNZ	N	256x256x2
Stack	Chop/Nod stacked	stacked	STK	N	256x256x2
Stack	Common dither positions combined (not usually used for imaging)	stackeddithers	SKD	N	256x256x2
Undistort	Corrected for optical distortion	undistorted	UND	Y	656x656x2
Merge	Chop/nod images merged into single source	merged	MRG	Y	656x656x3
Register	Multiple observations registered to a reference image	registered	REG	Y	656x656x3
Coadd	Multiple observations averaged	coadded	COA	Y	656x656x3

Table 1: Intermediate data products for imaging reduction

Step	Description	PRODTYPE	Identifier	Saved by default?	Size
Make Dark	Processed dark frame	masterdark	DRK	N	256x256x2
Make Flat	Processed flat frame	masterflat	FLT	N	256x256x2
Clean	Bad pixels cleaned	cleaned	CLN	N	256x256x2
Droop Correct	Corrected for droop effect	drooped	DRP	N	256x256x2
Nonlinearity Correct	Corrected for detector nonlinearity	linearized	LNZ	N	256x256x2
Flat Correct	Divided by flat field	flatted	FTD	N	256x256x2
Stack	Chop/Nod stacked	stacked	STK	Y	256x256x2

Stack	Common dither positions combined	stackeddithers	SKD	Y	256x256x2
Extract	Rectified image, produced during extraction	rectified	RIM	N	XxYx3 (size varies)
Extract	Raw extracted spectra	spec	SPC	N	Xx3xNa (NAXIS1 varies, Na=number of orders * apertures)
Extract	Merged spectra	mrgspec	MRG	Y	Xx3xN (NAXIS1 varies, N=number of orders)
Combine	Combined spectra	combspec	CMB	Y	Xx3xN
Flux Calibrate	Flux calibrated spectra	calspec	CAL	Y	Xx5xN

Table 2: Intermediate data products for grism reduction

5.1.3. Data Format

All files produced by the pipeline are FITS single-extension image files. All imaging products, are 3-D arrays of data, where the first plane is the image and the second plane is the variance associated with each pixel in the image. Take the square root of the variance plane to get the uncertainty estimate associated with each pixel in the image. The third plane in the merged, registered, and coadded imaging products is an exposure map, indicating the number of exposures stacked or coadded at each pixel. Multiply the exposure map by the value of the keyword DETITIME, divided by 2.0, to get the total integration time at each pixel in the image.

The stacked and rectified grism products, like the imaging products, are 3-D arrays of data, where the first plane is the image and the second is the variance. The third plane in the rectified image is a bad pixel mask. The rectified image also contains the wavelength calibration, encoded in the WCS in the header of the FITS file. The merged and combined grism products are one-dimensional spectra, stored in three rows of data. The first row is the wavelength, the second is the flux, the third is the error (standard deviation). If there were multiple orders in the spectrum (eg. the G1xG2 mode), then the spectrum for each order is stacked into a different plane. The length of the row varies depending on the data, but is typically around 240-250 pixels. For the calibrated grism product, two additional rows are added, for reference: the fourth is the fractional atmospheric transmission curve, the fifth is the instrumental response curve, in Me/s/Jy.

The final uncertainties in the calibrated spectrum contain both the statistical uncertainties due to the noise in the extracted spectrum and the systematic uncertainties due to the calibration (added in quadrature). This quantity represents our best estimate of the total uncertainty on the flux at any given pixel. Investigators who wish to investigate the signal-to-(statistical) noise of their spectra should retrieve the Level 2 merged spectrum; the uncertainties in these products do

not include the systematic values arising from the calibration process.

6. GROUPING LEVEL_1 DATA FOR PROCESSING

In order for a group of imaging data to be reduced together usefully, all images must have the same target object and be taken in the same chop/nod mode. They must also have the same detector, filter, and dichroic setting. In order to be calibrated together, they must also be taken on the same mission, with similar altitudes and zenith angles. Optionally, it may also be useful to separate out data files taken from different observations, or on different flight legs, or line-of-sight rewinds.

For spectroscopy, all the same rules hold, with the replacement of grism element for filter, and with the additional requirement that the same slit be used for all data files.

These requirements translate into a set of FITS header keywords that must match in order for a set of data to be grouped together. These keyword requirements are summarized below, in the tables below.

Grouping Criteria: Imaging

Keyword	Datatype	Match Criterion
OBSTYPE	STR	Exact
OBJECT	STR	Exact
INSTCFG	STR	Exact
DETCHAN	INT	Exact
SPECTEL1 / SPECTEL2*	STR	Exact
DICHROIC	STR	Exact
MISSN-ID	STR	Exact
ALTI_STA	FLT	+/- 500
ALTI_END	FLT	+/- 500
ZA_START	FLT	+/- 2.5
ZA_END	FLT	+/- 2.5
PLANID	STR	Exact
AOR_ID (optional)	STR	Exact
LASTREW (optional)	STR (DATE/TIME)	Exact

Grouping Criteria: Grism

Keyword	Datatype	Match Criterion
OBSTYPE	STR	Exact
OBJECT	STR	Exact
INSTCFG	STR	Exact
DETCHAN	INT	Exact
SPECTEL1 / SPECTEL2*	STR	Exact

DICHROIC	STR	Exact
MISSN-ID	STR	Exact
ALTI_STA	FLT	+/- 500
ALTI_END	FLT	+/- 500
ZA_START	FLT	+/- 2.5
ZA_END	FLT	+/- 2.5
PLANID	STR	Exact
SLIT**	STR	Exact
AOR_ID (optional)	STR	Exact
LASTREW (optional)	STR (DATE/TIME)	Exact

* SPECTEL1 is used if the detector is the SWC (DETCHAN=0), SPECTEL2 is used for LWC (DETCHAN=1)
** If SLIT is in use (value != "NONE" or "UNKNOWN"), **always** include group in the grism plan, regardless of INSTCFG. This ensures that slit images get reduced with the spectroscopic data and placed in the same preview.

Table 3: Keyword requirements for FORCAST imaging and spectroscopy grouping

7. CONFIGURATION AND EXECUTION

7.1. Installation

Redux is a software package written in IDL that is designed to be a framework for executing any number or combination of data reduction algorithms. For FORCAST, it has been developed to support seamlessly running image processing algorithms from the DRIP package and spectral extraction algorithms from the FSpextool package. Redux can run in an automatic batch mode, integrated with the SOFIA Data Pipeline System (DPS), or it can run with a graphical front end as a quick-look data viewer in flight or during manual data reduction and analysis. Redux with DRIP and FSpextool was developed under Linux and MacOS X operating systems, running IDL 8.1. Other operating systems and later versions of IDL may also work, but have not been tested.

Running Redux requires IDL 8.1 or later, as well as the latest version of the IDL Astronomy User's Library, the Coyote graphics library, the DRIP package, the FSpextool package, and the Redux code. DRIP, FSpextool, and Redux are under SOFIA DPS revision control and can be obtained directly from git repositories there. The IDL Astronomy User's Library (astrolib) is publicly available, and can be downloaded from the website at the following URL:
http://idlastro.gsfc.nasa.gov/homepage.html.
The Coyote graphics library (coyote) is also publicly available and can be downloaded from:
http://www.idlcoyote.com/documents/programs.php.
When these packages have been installed, their locations should be added to the IDL_PATH environment variable, so that their procedures are accessible to Redux.

When you have the gzipped tar file of the Redux, DRIP, and FSpextool codes, unpack them, as, for example:

```
tar xvzf redux.tar.gz
```

```
tar xvzf drip.tar.gz
tar xvzf fspextool.tar.gz
```

This will create directories called `redux`, `drip`, and `fspextool`, which will contain a number of subdirectories. Each of these package directories should be added to the IDL_PATH as well.

7.2. Configuration

For DRIP algorithms, default options are specified in the `dripconf.txt` configuration file, located in the drip package directory, which is read when the pipeline is initiated. This file contains a list of keywords and their values that are used in the data reduction process. See Appendix B for a sample of this configuration file.

For FSpextool algorithms, default options are specified in a FORCAST.dat configuration file, located in the fspextool package directory. This file also contains keyword-value pairs, in the format `parameter=value`. The parameters must all be present and in the correct order, but can have any number of spaces or comments between them. Comment lines begin with the % or # character. See Appendix B for a sample of this configuration file as well.

In automatic pipeline mode, parameters set in the configuration files are the values actually used by the pipeline. In interactive mode, the configuration files set the default values, but the parameter values used can be modified at run-time.

7.3. Input data

Redux takes as input raw FORCAST FITS data files, which are image cubes composed of 256x256 pixel image arrays. The number of frames per raw data cube depends on the chop/nod mode used to acquire the data (Table 2). FITS headers contain data acquisition and observation parameters and, combined with the pipeline configuration files, comprise the information necessary to complete all steps of the data reduction process.

Chop/Nod Mode	Number of frames	Comments
C2N, NMC	4	Two-Position Chop with Nod Matched in throw and parallel to the chop direction 2 chop positions in each of 2 nod positions
C2N, NPC	4	Two-Position Chop with Nod perpendicular to the chop direction 2 chop positions in each of 2 nod positions
C2NC2	4	Extreme asymmetric chop and telescope move to blank sky: two chop positions per sky position. Typically 5 input files corresponding to ABAABAAB pattern; see Section 3.2.5 for

		details.
N	2	Two-position Nod only, may be used for grism spectroscopy
SLITSCAN	4 or C2NC2	Spectral map of an extended source, most likely using C2NC2 but could use C2N

Table 4: Contents of FORCAST raw data files by observing mode

The pipeline reads a raw FORCAST data file, identifies the observing mode used and associates the file with a set of ancillary and calibration data files (Table 3). The default files to be used are defined in a lookup table that reads the DATE-OBS keyword from the raw file, then chooses the appropriate calibrations for that date.

Ancillary data file	Data type	Comments
Configuration file (eg. `dripconf.txt`)	ASCII	Contains initial configuration of pipeline parameters and nonlinearity coefficients
Bad pixel mask (eg. `swc_badpix.fits` or `lwc_badpix.fits`)	FITS	Single 2D image containing locations of bad pixels in short wave or long wave camera
Dark frame (eg. `swc_dark_10262012.fits` or `lwc_dark_10262012.fits`)	FITS	Dark frame for short wave or long wave camera
Flat frame (eg. `g5-8_l3_rawflat_300K.fits`)	FITS	Flat field frame appropriate for grism and slit used
`pinhole_locs.txt`	ASCII	Pinhole locations file for distortion correction (imaging only)
Spectral order definition file (eg. `G1_narrow_flat.fits`)	FITS	Image file containing header keywords that define the edges of all orders in a 2D spectral image
Wavelength calibration map (eg. `G1_wavecal.fits`)	FITS	Two frame image associating a wavelength value and a spatial distance across the slit with each pixel (grism only)
Atmospheric transmission curve (eg. `atran_41K_45deg_4-50mum.sav`)	IDL SAV	An IDL array with wavelength and transmission values for a particular altitude and zenith angle (grism only)
Instrumental response curve (eg. `G063_LS24_Cy2_response.out`)	ASCII	Table containing the response (Me/s/Jy) at each wavelength for a particular grism/slit mode

Table 5: Auxiliary files

7.4. Required input keywords

In order for Redux to complete processing successfully, there are a few keywords that must be correct in the input FITS headers. These are described in Appendix A.

7.5. Automatic mode execution

Redux is an object-oriented program whose basic unit is a reduction object (`forcast_imaging_reduction__define.pro` or `forcast_grism_reduction__define.pro`). To run the pipeline from the IDL command line as a DCS black box pipeline, we run the pipeline wrapper (`forcast_pipe.pro`). This wrapper takes as input the path to an input manifest file. This text file should contain a line specifying the number of input files, then the relative path to each input file, one per line. The script then reads these input files, instantiates the appropriate reduction object according to the mode specified in the input FITS headers, then calls the object's reduce method. This method calls each processing step in order, as appropriate for the given mode. Finally, the wrapper script will write an output manifest called `outfiles.txt` containing the names of the produced data files.

This wrapper can be invoked from the IDL prompt, as

```
IDL> forcast_pipe, 'infiles.txt'
```

or directly from a terminal as

```
localhost$ echo "forcast_pipe, 'infiles.txt'" | idl
```

The wrapper accepts a single input parameter on the command line, which, if set, will save all intermediate output. This option is specified as, for example:

```
IDL> forcast_pipe, 'infiles.txt', /save_intermediate
```

7.6. Manual mode execution

It is also possible to run the pipeline interactively, using a graphical user interface. The IDL command `redux`, called without arguments, will launch the Redux GUI.

7.6.1. Basic workflow

To start an interactive reduction, open a set of FORCAST files, using the File menu (**File->Open New Reduction**). This will bring up a file dialog window (Figure 10). All files selected will be reduced together as a single reduction set.

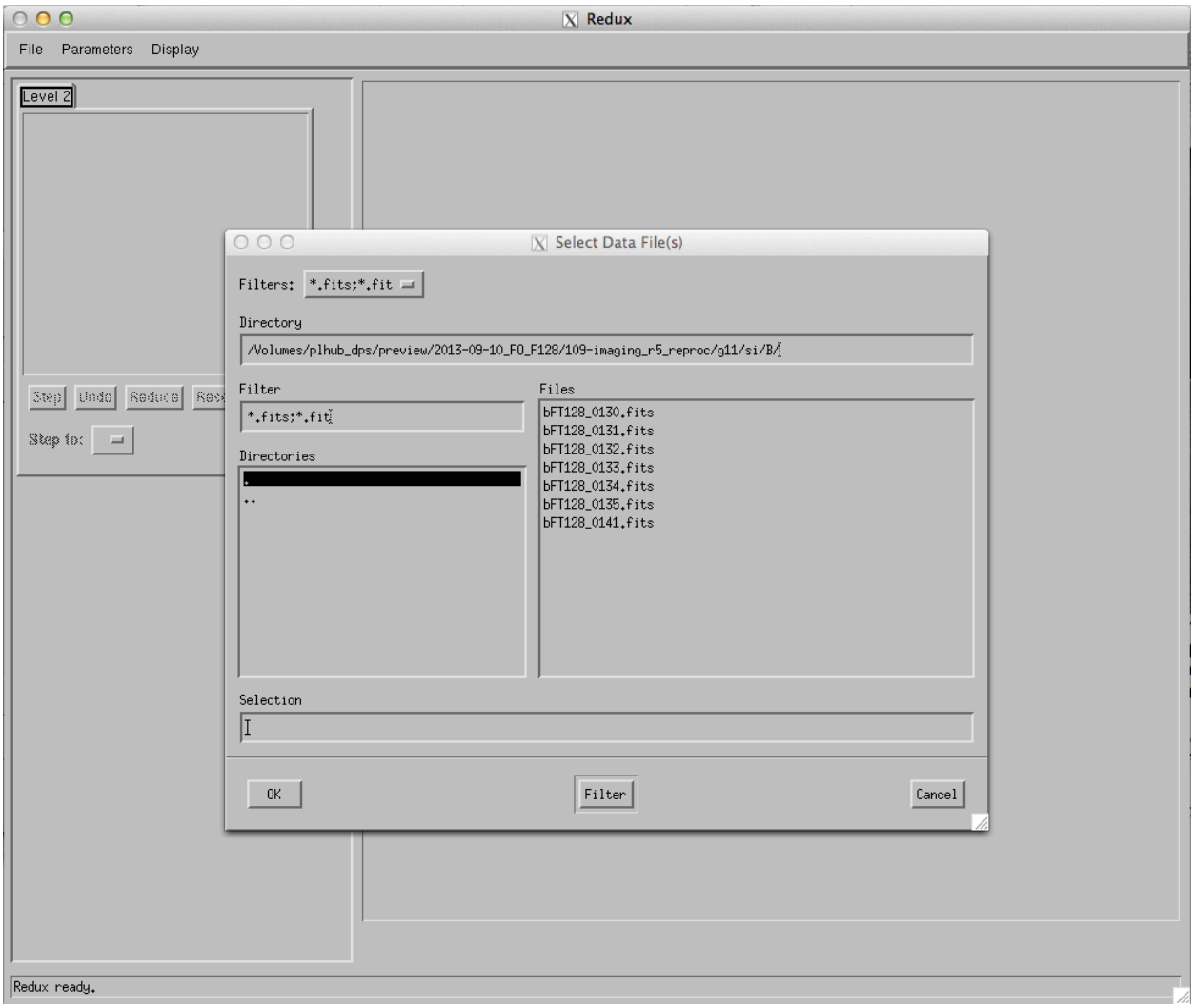

Figure 15: Open New Reduction

Redux will decide the appropriate reduction steps from the input files, and load them into the GUI. The steps for imaging inputs will differ from the steps for grism inputs (Figures 11 and 12), following the flowchart in Figure 6.

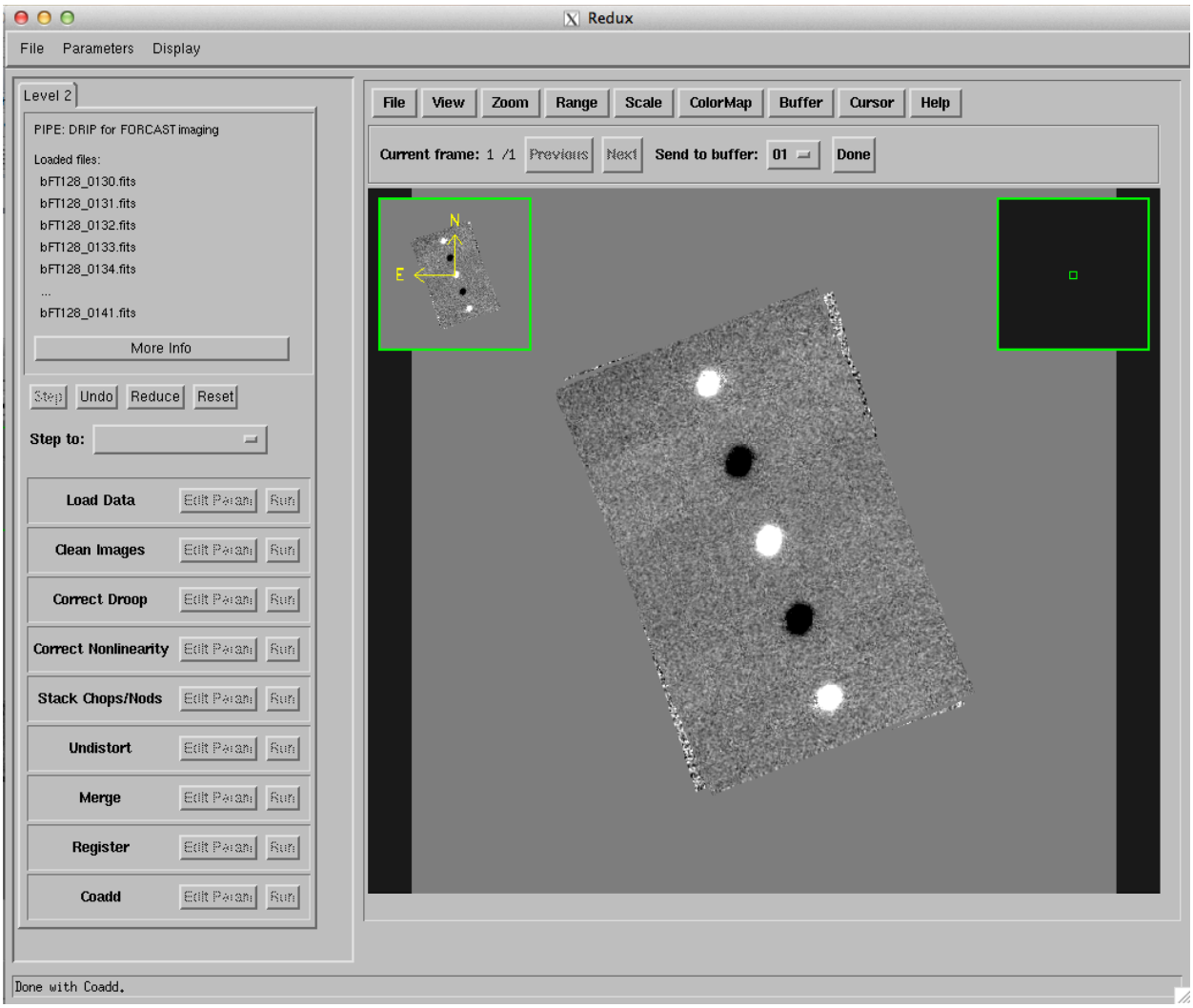

Figure 16: Imaging reduction

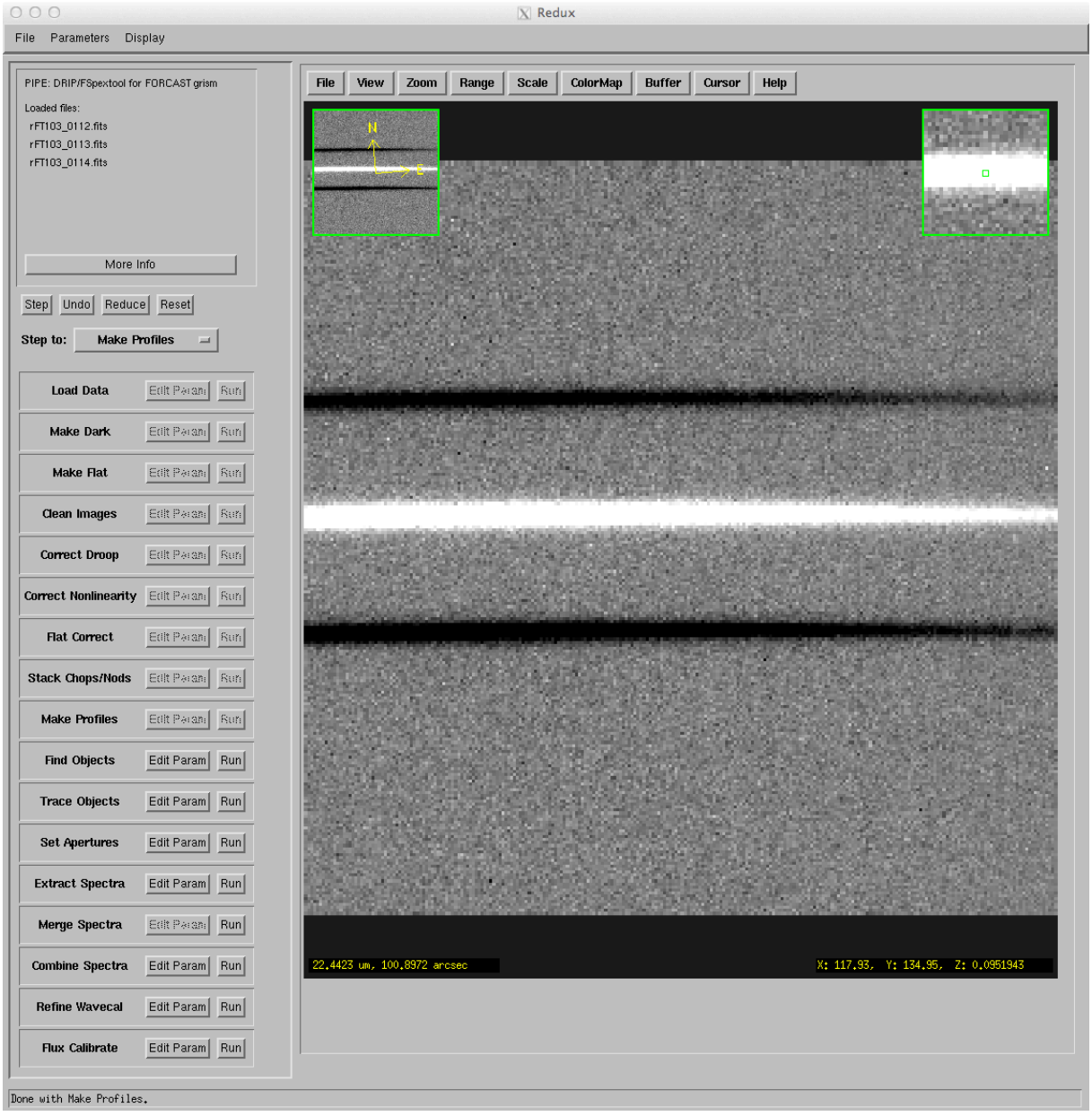

Figure 17: Grism reduction

In either mode, each reduction step has a number of parameters that can be edited before running the step. To examine or edit these parameters, click the **Edit Param** button next to the step name to bring up the parameter editor for that step (Figure 13). DRIP parameters are read from the `dripconf.txt` configuration file the first time the **Load Data** step is run, so parameters for later steps should not be edited prior to running this step. Within the parameter editor, all values may be edited, but will not be used unless **Save** or **Done** is selected. Clicking **Save** will leave the parameter editor window open; clicking **Done** will save values and close the window. Clicking **Reset** will restore any edited values to their defaults; clicking **Cancel** will discard all changes to the parameters.

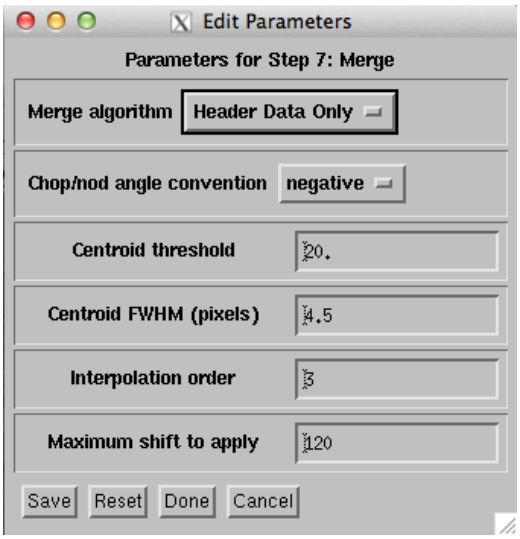

Figure 18: Sample parameter editor (for imaging Merge step)

After all parameters for a step have been examined and set to the user's satisfaction, a processing step can be run on all loaded files either by clicking **Step**, or the **Run** button next to the step name. Each processing step must be run in order, but if a processing step is selected in the **Step to:** widget, then clicking **Step** will treat all steps up through the selected step as a single step. When a step has been completed, its buttons will be grayed out and inaccessible. It is possible to undo one previous step by clicking **Undo**. All remaining steps can be run at once by clicking **Reduce**. After each step, the results of the processing will be displayed in the display window. Clicking **Reset** will restore the reduction to the initial state.

It is possible to reduce a single file at a time by opening a new reduction on a single file, reducing it, then adding in a new file (**File->Add Files**). Files can also be removed from the reduction set (**File->Remove Files**), but this will reset the reduction for all loaded files. Selecting **Display->Display File Information**, or the **More Info** button, will pull up a table of information about the currently loaded files (Figure 14). The table rows displayed can be filtered by entering a search string into the **Filter** text box.

The current set of parameters can be displayed, saved to a file, or reset all at once using the **Parameters** menu. A previously saved set of parameters can also be restored for use with the current reduction (**Parameters->Load Parameters**).

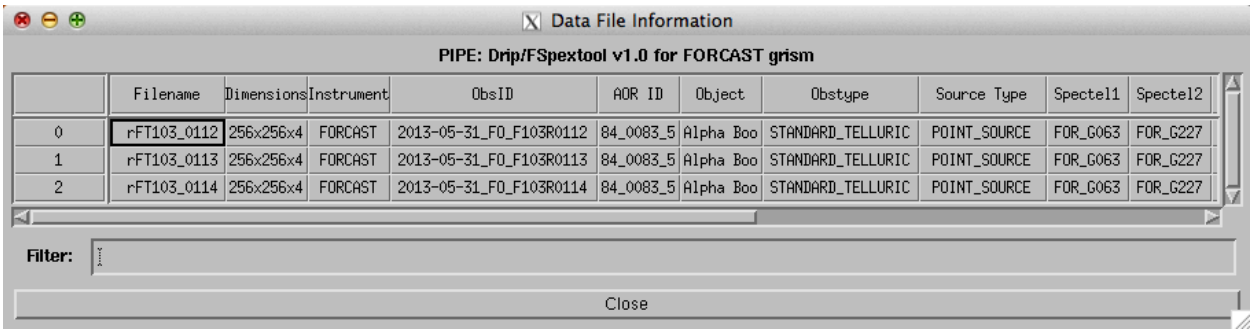

Figure 19: File information table

7.6.2. Display features

Redux displays images using ximgtool, a full-featured display tool distributed with FSpextool. For more information, see the ximgtool help file, available from Redux via the **Help** button just above the display. See Table 4 for a quick listing of the most useful ximgtool features.

Feature	Menu button	Keyboard shortcut
Load new file	**File->Load FITS**	--
Load file into new frame	**File->New Frame**	--
View FITS header	**File->View Header**	--
Zoom	**Zoom->Zoom In**, **Zoom Out**, **Zoom To Fit**	Press **z** to enter zoom mode, then **i** to zoom in, **o** to zoom out, or **t** to zoom to fit
Color stretch	**Cursor->Stretch**	Press **s** to enter stretch mode, click and drag to change brightness and contrast
Set display range	**Cursor->Range**	Press **r** to enter range mode, click and drag to select the box that sets the display range
Display distance	--	Press **d** to enter distance mode, then click and drag to identify start and end points
Line cut	--	Press **l** to enter line cut mode, then click and drag to identify start and end points
Display image statistics	--	Press **m** to enter moments mode, then click and drag to identify box for which the statistics should be calculated
Photometry	--	Press **a** over a star to do basic photometry.
Clear current mode	--	Press **c**
Buffer select	**Buffer->Buffer 1, Buffer 2…**	Press **f** to move to the next buffer, **b** to move to the previous buffer.
Buffer math	**Buffer->Buffer Math**, then select buffers and arithmetic operation to perform	--
Blink buffers	**Buffer->Blink Buffers**	--
Cycle frames	**Buffer->Cycle Frames**	Press **n** to move to the next frame, **p** to move to the previous frame.

Table 6: Useful ximgtool features

Ximgtool has five buffers available for simultaneous display of images. If there are more than five images loaded into Redux, they can be viewed by selecting **Display->Quick Look** from the Redux menu. This will cycle through the data in its current processing state, allowing interaction and analysis with each image in turn. To move between images, click the **Next File** or **Previous File** buttons, below the image. Click **Cancel** to quit the quick look display.

7.6.3. Imaging Reduction

FORCAST imaging reduction with Redux is straightforward. Each processing step in the GUI corresponds to a single step in the flowchart of Figure 6 (with the exception of the Jailbar Correct step: this is applied as part of the Stack Chop/Nods processing step in the Redux GUI).

Some useful options to note for imaging mode:

- Intermediate files can be saved by selecting the **Save all intermediate files** button in the Load Data parameters. The output path for reduced data can also be set in the Load Data parameters.
- The default for imaging is to subtract any residual background after stacking chop/nod frames. This can be turned off by deselecting **Subtract residual background** in the Stack Chop/Nods parameters.
- The default method for both merging and registration steps is to use the header data to determine the shifts. This can be overridden to select either the **Centroid** or **Cross Correlation** methods in the Merge or Coadd parameters.
- The most common point of failure for the imaging mode is at the Register step. Therefore, it is sometimes useful to load in previously merged images and simply run the register and coadd steps. This can be done via Redux, using the **File->Open New Reduction** menu button, and selecting the merged products to coadd.

7.7. Grism Reduction

FORCAST grism reduction with Redux is slightly more complicated than for imaging. The GUI has additional steps to make processed dark and flat files, and breaks down the spectral extraction into six separate steps to give more control over the extraction process. These steps are:

- Make Profiles: Generate a smoothed model of the relative distribution of the flux across the slit (the spatial profile)
- Find Objects: Use the spatial profile to identify spectra to extract. By default, Redux attempts to automatically identify sources, but they can also be manually identified by entering a guess position to fit near, or a fixed position, in the Find Objects parameters. These manual positions should be comma-separated values, listed in arcseconds (refer to the spatial profile).
- Trace Objects: Identify the location of the spectrum across the array, by either fitting the continuum or fixing the location to the aperture center.
- Set Apertures: Identify the data to extract from the spatial profile (Figure 15). This is done automatically by default, but all aperture parameters can be overridden manually in the parameters for this step.
- Extract Spectra: Extract the one-dimensional spectrum from the identified apertures. By default, Redux will attempt optimal extraction for files with keyword SRCTYPE=POINT_SOURCE, and standard extraction for any other value. The method can be overridden in the parameters for this step.
- Merge Spectra: All apertures are merged into a single spectrum, and normalized according to the Chop/Nod mode, as detailed in section 3.2.9

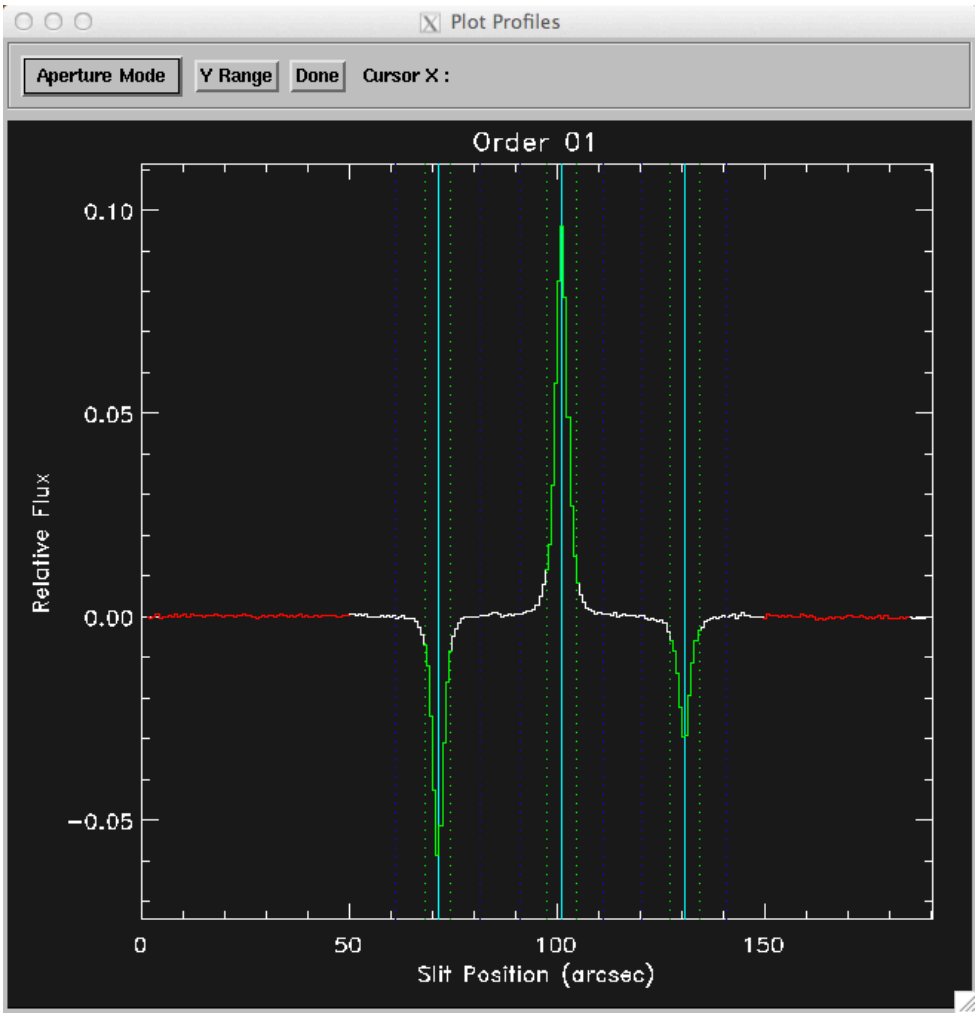

Figure 20: Aperture locations automatically identified and overplotted on the spatial profile. The three positions identified (light blue lines) correspond to the positive and negative spectra in Figure 12. Green lines indicated the extraction aperture, dark blue lines indicate the PSF radius (the point at which the flux goes to zero), and red lines indicate background regions.

Other important parameters to note:

- As for imaging reductions, intermediate files can be saved by selecting the **Save all intermediate files** button in the Load Data parameters. Output paths are also set in these parameters.
- Default dark, flat, and wavelength calibration files are determined after the Load Data step is run. They can be overridden in the parameters for the Make Dark, Make Flat, and Make Profiles steps.
- By default for all grism reductions except telluric standards, files that are taken at the same dither position are combined after chop/nod stacking, before extraction. This generally increases the signal to noise in the final spectra, as it results in a higher-quality spatial profile and spectral trace. This option can be turned off in the Stack Chops/Nods step by deselecting **Combine common dither positions**.

- Previously generated stacked images can be loaded directly into Redux for extraction using the **File->Open New Reduction** menu button, and selecting the stacked products. Previously generated merged spectra can be loaded similarly, for direct access to the combination step.

Extracted spectra are displayed using xvspec (Figure 16), a display tool packaged with FSpextool. Xvspec can only display one spectrum at a time, but each loaded spectrum can be examined by using the **Quick Look** feature, or else additional spectra may be directly loaded for display by clicking **Load Spextool FITS**. For more information on xvspec features, use the Help button in the xvspec window.

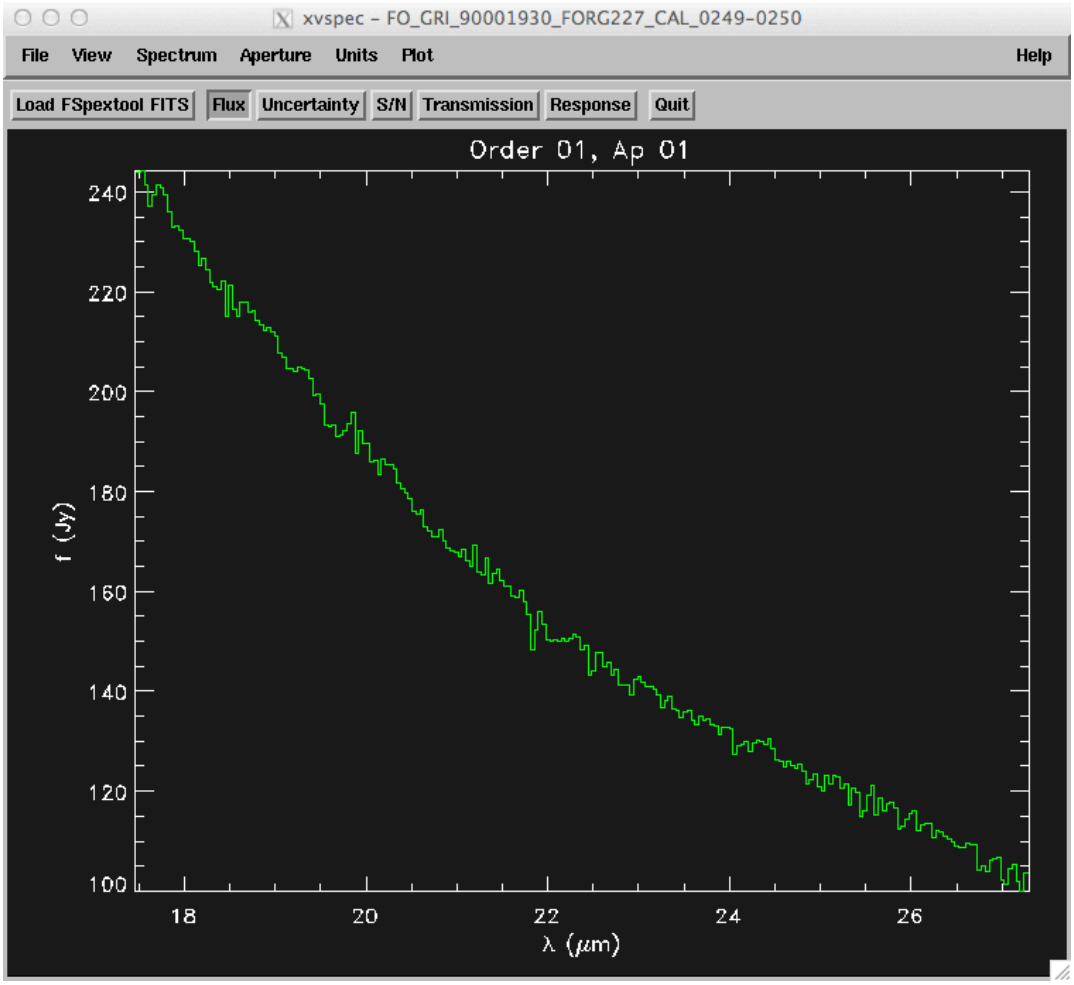

Figure 21: Final extracted spectrum, displayed in xvspec

7.8. Imaging Flux Calibration

The pipeline can apply a flux calibration. The calibration factors (e.g. Me-/sec per Jy) must be derived for each FORCAST filter configuration from observations of calibrator stars. The forcast pipeline includes a module to measure the photometry of the standard stars and apply the

calibration factor to the targets based on the theoretical flux of the standard star and the observed photometry. The module, namely pipecal, allows users to pertorm the photometry of the standard stars and derive the calibration factors based on comparing the photometry to the models of the standards. The final calibration factors of the targets account of the altitude and zenith angle differences. The reduction is carried by steps as follow:

1) **Copy** the directory containing the level 2 data (named pXXXX)
2) **Group** the data in separate directories according to their observation properties (filter/dichoric). The grouping can be performed manually, by DPS or using the groupl3 tool in this package as: ***groupl3 -g pXXXX.***
   Alternatively you can run the command with the obtion "-q FAIL" to avoid copying data that has DATAQUAL keyword value FAIL. However, since the calibration routines can also avoid those files, it is preferably to also stage the "FAIL" data to have them in the summary of the processing as bad flagged data. This will allow future reviewers of the processing to understand that the data was not missed by mistake but removed on purpose from the processing. The resulting tree will be as follow:

```
pXXXX/
        g1/
        g2/
        g3/
        ...
        fc_imagingswc_0_mirrorswc_forf066/
        fc_imagingswc_0_mirrorswc_forf077/
        ...
```

   where fc_* will contain the staged data. When grouping, the groupl3 tool will also stage the resources required for calibration such as response, standard fluxes and colors.
3) **Measure the photometry** of the standard stars. Use the ***dophot*** tool for this step. You can use the commad to perform the photometry recursively under all the staged data. Some examples of how to run the photometry are below.
   - dophot -r fc_imaging*
   - dophot -r fc_imagingswc_0_mirrorswc_forf066/
   - dophot fc_imagingswc_0_mirrorswc_forf066/standard_flux/gY
4) **Analyze the photometric** results by using ***checkphot*** in all the Run the command to perform the analysis of the resulting photometry. This routine will create calibration values at specific altitude and zenith angles for the input data. Use the "-f file" option to record the summary in an output file. If you require to display more keyword values in the output table, use "-k keywordName" as many times as additional keywords you require. The table will contain the calibration factors at a defaut altitude (41000 feet) and zenith angle (45 degrees). For any altitude/za combination the values of the calibration factor at a specific filter/dichroic should be consistent. A big difference indicates a problem with the standard star.
   - checkphot -r fc_imaging*
   - checkphot -r altitude za fc_imaging*

   The result of the photometry will be recorded in the headers of the standards.
5) **Perform a QA** on these stars and flag the one that have problems using the ***flagqual*** tool.

- flagqual file BADCAL
- flagqual -r fc_imagingswc_0_mirrorswc_forf066/standard_flux/g*
- flagqual -f list.txt BADCAL

6) **Perform calibration** of the targets using the ***docal*** tool.
   - docal -r fc_imaging*

   The final calibration factor, its error and the reference wavelength value are written in the headers of the target as CALFCTR, ERRCALF and LAMREF. After all calibration factors were derived for a mission, the final step requires studying the calibration values. The calibration factor for each instrument configuration should be consistent within a mission and even between consecutive missions. Values that are not consistent may come from bad observations of a standard star. The operator should remove bad standard stars and start again the reduction process from the first step for each mission that contains inconsistent values.
7) **Record the results** in text files using the ***dosummary*** tool. The results may include the photometry and/or the calibration factors. You can use the "-f file" option to record the values in a file. If necessary, you can concatenate the photometry and calibration files to be stored in the archive.
   - for calibration factors: dosummary fc_imaging*
   - for photometry: dosummary -s fc_imaging*

8. DATA QUALITY ASSESSMENT

8.1.1. Imaging

The first step in the quality analysis is to check the information from the pipeline. This information is available in the output log to the terminal and in the headers of the output files. Since the final product header contains all the information of the intermediate steps, a quick look at the header could tell us about any potential failure in the process. It is also important to look at the HISTORY of the headers of the final image and calibration files to identify messages from each step. Redux writes error messages in the header history when one of the steps is not performed correctly.

The second step of the quality analysis requires visualizing the images that were produced by the pipeline. First, check the final image for major problems. Some problems can be quickly identified by checking if the final image contains the expected pattern according to the observation mode. Then, check each of the intermediate products and calibration files to identify the following potential problems (see Table 7).

Product Type to Check	**Features to look for**	**Problem**
cleaned (or any	Bad pixels at the positions	Bad pixel removal was not performed correctly.

subsequent image)	indicated by the mask.	
drooped	Stellar profiles dip below background level at their edges	Droop correction was not performed correctly.
merged	Elongated stars	Indicates that: - the shift of the merging was not calculated correctly * N option: chop and nod keywords are wrong * CENT and COR options: algorithm did not work - or the distortion correction is wrong. Check if it was performed and which input file was used. Check also related keyword values.
stacked or merged	Jailbar structures	Jailbar correction was not performed correctly at drip_stack step for the data.
merged	Pattern of stars does not match the expected one for observation mode.	The shift when merging was wrong. Check the shift values and compare them with the distance between stars in the stacked image. If CENT algorithm was used, there may be a detection of a cosmic ray as a star.
coadded	Central star shows twice or there is an obvious mismatch of two coadded images	The shift was calculated wrong or the headers are wrong. Check the dither keyword values for N option or the shift values for CENT or COR. NOTE: A bad coadd may not be obvious in the final image when many fits are combined because the average may remove the mismatched pattern. It is sometimes better to check the coadded image after each dither position.

Table 7: Potential problems with imaging reductions

8.1.2. Spectroscopy

Because all the products except for the extracted spectra are created in the same way as for imaging, the QA of spectroscopy data will be performed in a similar way as for imaging data. The difference will come from the fact that the features in the visualized data will be different (see Table 7).

Product Type to Check	Features to look for	Problem
cleaned (or any subsequent image)	Bad pixels at the positions indicated by the mask.	Bad pixel removal was not performed correctly
drooped	Spatial profile dips below background level near bright spectral trace	Droop correction was not performed correctly.
stacked	Jail bar structures	Jail bar correction was not performed correctly at drip_stack step for the data, or flat fielding was performed with a flat containing jailbar structures.
stacked	Position of spectra along the slit does not match the expected one for observation mode.	This may indicate a problem with the raw data, if a chop or nod failed.

stackeddithers	Multiple spectra appear where there should be only one	This may indicated that the source appeared in a different place in the slit for one or more input files
mrgspec, combspec	Features are duplicated or repeat a number of times.	The wavelength was calculated wrong or the headers are wrong.

Table 8: Potential problems with grism reductions

APPENDIX A: REQUIRED INPUT KEYWORDS

This table describes the type and expected value for all FITS keywords used by the DRIP/FSpextool pipelines.

Table 8: Required input keywords

Keyword	Type	Expected value
Image keywords		
ALTI_STA	float	0-60000.
ALTI_END	float	0-60000.
AOR_ID	string	
DATASRC	string	ASTRO, CALIBRATION, LAB, TEST, OTHER, FIRSTPOINT
DATE-OBS	string	yyyy-mm-ddThh:mm:ss[.sss]
DETCHAN	int	0, 1
DETECTOR	string	As-010, Sb-083
DETITIME	float	> 0
EPERADU	float	> 1
FRMRATE	float	> 0
ILOWCAP	bool	
INSTCFG	string	IMAGING_SWC, IMAGING_LWC, IMAGING_DUAL, GRISM_XD, GRISM_SWC, GRISM_LWC, GRISM_DUAL, GRISM_XD-LSV, GRISM-SSV, GRISM-LSV
INSTMODE	string	C2, C2N, C2NC2, N, SLITSCAN, NXCAC
INSTRUME	string	FORCAST
INTTIME	float	
MISSN_ID	string	
NAXIS1	int	256
NAXIS2	int	256
OBJECT	string	
OBS_ID	string	
OBSTYPE	string	OBJECT, STANDARD_FLUX, STANDARD_TELLURIC, LAMP, FLAT, DARK, BIAS, SKY
OTMODE	string	AD, SUR
OTSTACKS	int	>0
SPECTEL1	string	NONE, FOR_F054, FOR_F064, FOR_F066, FOR_F077, FOR_F111, FOR_F113, FOR_F197, FOR_F253, FOR_XG063, FOR_XG111, FOR_G063, FOR_G111
SPECTEL2	string	NONE, FOR_F086, FOR_F113, FOR_F118, FOR_F254, FOR_F315, FOR_F336, FOR_F348, FOR_F371, FOR_F242,

		FOR_G227, FOR_G329
TELESCOP	string	SOFIA
TIME-OBS	string	
WAVELNTH	float	0-40.
ZA_START	float	0-90.
ZA_END	float	0-90.
Shifting keywords		
DITHER	bool	
DITHERCS	string	SIRF, ERF
DTHINDEX	int	> 0
DITHERX	float	
DITHERY	float	
CHOPPING	bool	
CHPCRSYS	string	SIRF, ERF
CHPAMP1	float	≥ 0
CHPANGLR	float	
CHPANGLE	float	
CHPNPOS	int	> 0
NODDING	bool	
NODCRSYS	string	SIRF, ERF
NODAMP	float	≥ 0
NODANGLR	float	
NODANGLE	float	
NODBEAM	string	A, B
SKY_ANGL	float	
Instrument Mode keywords		
SKYMODE	string	C2NC2, NMC, NPC, NPCNAS, NPCCAS, SLITSCAN, NOD, NXCAC
Additional keywords for grism mode		
SRCTYPE	string	POINT_SOURCE, EXTENDED_SOURCE, OTHER, UNKNOWN
SLIT	string	FOR_SS24, FOR_LS24, FOR_LS47, NONE
Required dripconf keywords		
RN_HIGH	float	
RN_LOW	float	
BETA_G	float	
BORDER	int	
BGSUB	int	0, 1
ANGLCONV	string	‘positive’, ‘negative’
CHOPTSACONV	int	-1, 1
XYSHIFT	float	
SHIFTORD	int	0, 1, 3
CORMERGE	string	COR, CENT, N, NOSHIFT
CORCOADD	string	COR, CENT, N, NOSHIFT, WCS

MTHRESH	float	
CTHRESH	float	
JBCLEAN	string	MEDIAN, FFT, N
FRACDROOP	float	
MINDROOP	float	
MAXDROOP	float	
NRODROOP	int	
ORDER	int	
PIN_NPTS	int array	should have 2 elements
PIN_SPX	int array	should have 8 elements
PIN_SPY	int array	should have 8 elements
NLRSWCHI	float	
NLSSWCHI	float	
NLCSWCHI	float array	should have >1 element
LIMSWCHI	float array	should have 2 elements
NLRSWCLO	float	
NLSSWCLO	float	
NLCSWCLO	float array	should have >1 element
LIMSWCLO	float array	should have 2 elements
NLRLWCHI	float	
NLSLWCHI	float	
NLCLWCHI	float array	should have >1 element
LIMLWCHI	float array	should have 2 elements
NLRLWCLO	float	
NLSLWCLO	float	
NLCLWCLO	float array	should have >1 element
LIMLWCLO	float array	should have 2 elements
NLINSECTION	int array	should have 4 elements

APPENDIX B: SAMPLE CONFIGURATION FILES

This is a sample of the `dripconf.txt` configuration file, used to set reduction parameters for DRIP algorithms and override FITS keyword values as necessary.

```
; Calibration files for swc and lwc cameras
badfileswc='swc_badpix.fits'
badfilelwc='lwc_badpix.fits'
flatfileswc='b039_1202.fits'
flatfilelwc='r039_1202.fits'
darkfileswc='b039_1203.fits'
darkfilelwc='r039_1203.fits'

; Parameters for variance calculation
rn_high=2400.
rn_low=244.8
beta_g=1.0

; Parameters for resizing imaging
resize=1.0
border=128

; Parameters for stacking
; if bgsub=1, residual background will be subtracted after stacking
bgsub=1

; Parameters for merging and coadding
anglconv='negative'
xyshift=15.
shiftord=0
; cormerge='COR' triggers drip_merge to use cross correlation
; cormerge='CENT' triggers drip_merge to use centroid
; cormerge='N' triggers drip_merge to use nominal chop/nod positions
CORMERGE = 'N'
CORCOADD = 'N'
; jbclean = 'FFT' triggers cleaning jailbar pattern with fft
; jbclean='MEDIAN' trigger cleaning jailbar pattern with median filter
; jbclean='N' no jailbar cleaning
jbclean = 'MEDIAN'
; Find peaks
cthresh=15.
mthresh=15.
;mfwhm=20.

; distortion correction
order=3
pinhole_file='pinhole_locs.txt'
pin_npts=[12,12]
pin_spx=[3,3,3,2,5,5,6,6]
pin_spy=[1,2,3,3,5,6,5,6]

; droop correction
mindroop    = 0.0
maxdroop    = 65535.0
nrodroop = 16
fracdroop = 0.0035

; Global image correction
nlinsection=[128,128,190,190]
nlinrefs = 7000.
nlinscal = 7000.
nlcswclo = [1.0000000, 0.39280,-0.16786,-0.11277]
nlcswchi = [1.0000000, 0.39280,-0.16786,-0.11277]
nlclwclo = [0.99867,0.34156,-0.14747,-0.07846]
nlclwchi = [0.99867,0.34156,-0.14747,-0.07846]
limswclo  = [1900.0, 12100.0]
limswchi  = [1900.0, 12100.0]
limlwclo  = [2500.0, 11000.0]
limlwchi  = [2500.0, 11000.0]
```

This is a sample of the FSpextool configuration file for FORCAST, called FORCAST.dat, condensed into a single image. Page 1 lists parameters used by the pipeline. Pages 2 and 3 list all keywords that will be propagated from the input files to the output files.

Page 1

```
# This is the calibration file for the FORCAST spectrograph on SOFIA.
# Note the values must be in the correct order, but can have any number of
# spaces/comments between them.
#
#------------------------------------------------------------------------
#
 INSTRUMENT=FORCAST
 NCOLS=256
 NROWS=256
 STDIMAGE=256
 PLOTWINSIZE=700 512
 FILENAME=FILENAME
 EXPTIME=EXPTIME
 TIME=TIME_OBS
 POSANGLE=None
 HA=None
 AIRMASS=None
 NINT=4
 BADPIXMASK=None
%
%CAL BASE
%
 CALMODULE=mc_forcastcals1d
%
%FILE READ MODE
%
 FILEREADMODE=Filename
 IPREFIX=F
 OPREFIX=r
 SUFFIX=*.fits*
 FITSREADPROGRAM=mc_readforcastfits
 HEADCOMBPROGRAM=mc_forcastdcshdr
 YUNITS=Me/s
 YTITLE=f (!5Me s!u-1!n)
%
% Reduction Mode
%
 REDUCTIONMODE=A
%
% Combine Base Information
%
 COMBMODE=A
 COMBSTAT=Median (Median Error)
 COMBTHRESH=8.0
 COMBODIR=proc/
%
%  Sky Base Information
%
 SKYSTAT=Robust Weighted Mean
 SKYTHRESH=8.0
%
% Profile Parameters
%
 YBUFFER=2
 OVERSAMP=1
 ATMOSTHRESH=0.7
%
% Point Source Base
%
PSNAPS=1
PSPSFRAD=22.0
PSAPRAD=7.0
PSBGSUB=1
PSBGSTART=24.0
PSBGWIDTH=30
PSBGDEG=0
PSBGMULT=2.0
%
% Extended Source Base
%
XSBGSUB=1
XSBG=0-9,24-36,54-60
XSBGDEG=0
%
% Additional processing base
%
 ADDLMODULE=mc_forcastaddlproc
%
%Other Base Parameters
%
 TRACEDEG=2
 TRACESTEP=7
 TRACESUMAP=7
 TRACESIGTHRESH=1
 TRACEWINTHRESH=5
 BADPIXELTHRESH=7
 PLOTSATURATEDPIXELS=0
 SATURATION=3000
 CHECKSEEING=0
 SEEINGTHRESH=3
 LINCORRECT=0
 ERRORPROPAGATION=1
 FLATFIELD=0
 FIXBADPIXELS=1
 OPTIMALEXTRACTION=1
```

Page 2

```
%
% FITS Header Keywords to Grab
% ----------------------------
%
% Keywords required for processing, may also be required
% by SOFIA DCS
%
KEYWORD=ALTI_STA
KEYWORD=ALTI_END
KEYWORD=AOR_ID
KEYWORD=CZNCZ
KEYWORD=CHOPPING
KEYWORD=CHPAMP1
KEYWORD=CHPANGLE
KEYWORD=CHPANGLR
KEYWORD=CHPCOORD
KEYWORD=CHPNPOS
KEYWORD=DATASRC
KEYWORD=DATE-OBS
KEYWORD=DETCHAN
KEYWORD=DETECTOR
KEYWORD=DETITIME
KEYWORD=DITHER
KEYWORD=DITHERCS
KEYWORD=DITHERX
KEYWORD=DITHERY
KEYWORD=DTHINDEX
KEYWORD=DTHNPOS
KEYWORD=EPERADU
KEYWORD=FILENUM
KEYWORD=FRMRATE
KEYWORD=ILOWCAP
KEYWORD=INSTCFG
KEYWORD=INSTMODE
KEYWORD=INSTRUME
KEYWORD=INTTIME
KEYWORD=MISSN-ID
KEYWORD=NODAMP
KEYWORD=NODANGLE
KEYWORD=NODANGLR
KEYWORD=NODBEAM
KEYWORD=NODCOORD
KEYWORD=NODDING
KEYWORD=OBJECT
KEYWORD=OBS_ID
KEYWORD=OBSTYPE
KEYWORD=OTMODE
KEYWORD=OTNBUFS
KEYWORD=OTSTACKS
KEYWORD=SKY_ANGL
KEYWORD=SKYMODE
KEYWORD=SLIT
KEYWORD=SPECTEL1
KEYWORD=SPECTEL2
KEYWORD=SRCTYPE
KEYWORD=TELESCOP
KEYWORD=UTCSTART
KEYWORD=WAVELNTH
KEYWORD=ZA_START
KEYWORD=ZA_END
%
% Keywords required by SOFIA DCS keyword dictionary
%
KEYWORD=KWDICT
KEYWORD=IMAGEID
KEYWORD=AOT_ID
KEYWORD=PROCSTAT
KEYWORD=HEADSTAT
KEYWORD=FILEREV
KEYWORD=PLANID
KEYWORD=DEPLOY
KEYWORD=FLIGHTLG
KEYWORD=ORIGIN
KEYWORD=OBSERVER
KEYWORD=CREATOR
KEYWORD=OPERATOR
KEYWORD=FILENAME
KEYWORD=DATE
KEYWORD=UTCEND
KEYWORD=WVZ_STA
KEYWORD=WVZ_END
KEYWORD=TEMP_OUT
KEYWORD=TEMPPRI1
KEYWORD=TEMPPRI2
KEYWORD=TEMPPRI3
KEYWORD=TEMPSEC1
```

Page 3

```
KEYWORD=AIRSPEED
KEYWORD=GRDSPEED
KEYWORD=LAT_STA
KEYWORD=LON_STA
KEYWORD=LAT_END
KEYWORD=LON_END
KEYWORD=HEADING
KEYWORD=TRACKANG
KEYWORD=TELCONF
KEYWORD=TELRA
KEYWORD=TELDEC
KEYWORD=TELVPA
KEYWORD=TELEQUI
KEYWORD=LASTREW
KEYWORD=FOCUS_ST
KEYWORD=FOCUS_EN
KEYWORD=TELEL
KEYWORD=TELXEL
KEYWORD=TELLOS
KEYWORD=TSC-STAT
KEYWORD=FBC-STAT
KEYWORD=OBSRA
KEYWORD=OBSDEC
KEYWORD=EQUINOX
KEYWORD=TRACMODE
KEYWORD=TRACERR
KEYWORD=MAPPING
KEYWORD=SCANNING
KEYWORD=DATATYPE
KEYWORD=MCCSMODE
KEYWORD=EXPTIME
KEYWORD=RESOLUN
KEYWORD=DETSIZE
KEYWORD=PIXSCAL
KEYWORD=SIBS_X
KEYWORD=SIBS_Y
KEYWORD=CHPFREQ
KEYWORD=CHPPROF
KEYWORD=CHPSYM
KEYWORD=CHPAMP2
KEYWORD=CHPCRSYS
KEYWORD=CHPTIP
KEYWORD=CHPTILT
KEYWORD=CHPPHASE
KEYWORD=NODTIME
KEYWORD=NODN
KEYWORD=NODSETL
KEYWORD=NODPATT
KEYWORD=NODSTYLE
KEYWORD=NODCRSYS
KEYWORD=DTHPATT
KEYWORD=DTHOFFS
KEYWORD=MAPCRSYS
KEYWORD=MAPNXPOS
KEYWORD=MAPNYPOS
KEYWORD=MAPINTX
KEYWORD=MAPINTY
KEYWORD=SCNRA0
KEYWORD=SCNDEC0
KEYWORD=SCNRAF
KEYWORD=SCNDECF
KEYWORD=SCNRATE
KEYWORD=SCNDIR
%
% Pipeline keywords
%
KEYWORD=PIPELINE
KEYWORD=PIPEVERS
KEYWORD=PRODTYPE
KEYWORD=BAD_OID
KEYWORD=LINC_OID
KEYWORD=DARK_OID
KEYWORD=FLAT_OID
KEYWORD=HISTORY
```