Wide-field Infrared Survey Explorer (WISE)

Flatcal Subsystem Design Document

Version 2.0

12 August 2009

Prepared by: John W. Fowler & Frank. J. Masci

Infrared Processing and Analysis Center
California Institute of Technology

WSDC D-D012

Concurred By:

Roc Cutri, WISE Science Data Center Manager

Tim Conrow, WISE Science Data Center System Architect

John Fowler, WISE Science Data Center Cognizant Engineer

Frank Masci, WISE Science Data Center Cognizant Engineer

Revision History

Date	Version	Author	Description
2 September 2008	1.0	J. W. Fowler & F. J. Masci	Initial Draft
6 August 2009	1.1-1.41	J. W. Fowler & F. J. Masci	Added “-o9” frame-median table file output, updated error message handling, installed no-prior-uncertainty processing, and FRSETID processing; switched from FITS header keyword UTCS_OBS to UNIXT for time order
12 August 2009	2.0	J. W. Fowler & F. J. Masci	Added frame partitioning

Table of Contents

1 Introduction

1.1 Subsystem Overview

This document presents the requirements, design, algorithms, and state of implementation of the Flatcal (Flat Field Calibration) subsystem of the WSDC data processing system. Flatcal runs offline on all or part of a single scan.

1.1.1 Requirements

The Flatcal subsystem is required to compute for each WISE survey band a flat-field image, i.e., a relative pixel responsivity map, which is a calibration product that stores *relative* pixel-to-pixel responsivity variations for an array. This product will be created dynamically, i.e., when a desired number of (good quality) frames become available. It will then be applied to frames in the instrumental calibration pipeline.

The method is essentially the same as that used on 2MASS where flat-fields were derived from the twilight sky. On WISE, the zodiacal background will be used as the uniform illumination source. The method entails measuring the *change* in intensity in a pixel in response to the variation in overall (e.g., median) intensity in a frame as uniform illumination of the array changes. The relative responsivity for a pixel is given by the slope of a *robust* linear fit to the pixel versus overall intensity data. This will use scans where the background is known *a priori* to vary approximately uniformly over the array by enough to make a linear fit numerically stable. It is expected to vary by ~40-50% in bands 1 and 2, and ~30% in bands 3 and 4 from an ecliptic pole to equator (~ a quarter orbit). An important assumption for this method to work is that *every* pixel in an array must see the same intrinsic *change* in the background. A scenario where this may fail is described in section 5.3. We shall refer to this method as the *slope method*.

The slope method helps mitigate incomplete knowledge of a *static* absolute dark/bias bias level, since this effectively cancels out. This is generally not true for the classic flat-field estimation method based on combining frames in a stack and then normalizing. In the slope method, short-term fluctuations will still contribute to the scatter in the intensity measurements, and hence uncertainties in the final responsivity estimates.

Estimates by the WISE Project Office show that at least 15 orbits (i.e., 60 quarter orbit scans or ~thousands of frames) will need to be combined to obtain flats to better than 1% accuracy in bands 1 and 2. The reason is that the background signal in bands 1 and 2 will be low and many frames are needed to get a good signal-to-noise ratio. For bands 3 and 4, at least one orbit worth (i.e., 4 quarter orbits) will give responsivity maps to the desired accuracy to meet sensitivity requirements. These are considerations needed for memory allocation planning, especially for bands 1 and 2.

The Level 4 requirements supported by this processing are as follows.

L4WSDC-012: Flux measurements in the WISE Source Catalog shall have a SNR of five or more for point sources with fluxes of 0.12, 0.16, 0.65 and 2.6 mJy at 3.3, 4.7, 12 and 23 micrometers, respectively, assuming 8 independent exposures and where the noise flux errors due to zodiacal foreground emission, instrumental effects, source photon statistics, and neighboring sources (*traceable to Level-1*).

L4WSDC-013: The root mean square error in relative photometric accuracy in the WISE Source Catalog shall be better than 7% in each band for unsaturated point sources with SNR>100, where the noise flux errors due to zodiacal foreground emission, instrumental effects, source photon statistics, and neighboring sources. This requirement shall not apply to sources that superimposed on an identified artifact (*traceable to Level-1*).

L4WSDC-037: The WSDC Pipelines subsystem shall convert raw WISE science and engineering data into calibrated images and extracted source lists from which the preliminary and final WISE data products will be derived.

L4WSDC-039: Within 3 days from receipt of a given data set at the WSDC all data shall be processed through the WSDS Scan/Frame pipeline which performs basic image calibration and source extraction from on images from individual orbits. The results of this processing step shall be Level 1 source extractions and image data, which are loaded into the WISE Level 1 extracted Source Working Database (L1WDB) and Image Archive allowing access by the WISE Science Team for external quality assessment.

L4WSDC-042: The WSDS Pipeline processing shall remove the instrumental signature from Level 0 image frames.

L4WSDC-062: The WSDC shall perform quality analysis of all WISE science data and make reports available on a regular basis.

The Flatcal module is run offline on a list of preselected survey data images. Input frames will have already been dark-subtracted and linearized. The units of the pixel values are native WISE "*scaled*-slope" units as computed onboard for detector ramps. More specifically, they will be in units of *scaled* DN/SUR, where DN means Data Number and SUR means Sample Up the Ramp.

1.1.2 Liens

- Second-pass fitting with outlier rejection is under consideration.

1.2 Applicable Documents

This subsystem conforms to the specifications in the following project documents:

- WISE Science Data Center Functional Requirements Document, WSDC D-R001
- WISE Science Data System Functional Design, WSDC D-D001
- Software Management Plan, WSDC D-M003
- Instrumental Calibration Plans and Considerations:
  http://web.ipac.caltech.edu/staff/fmasci/home/wise/SingleOrbit_Cal.html
- Infrastructure and Instrumental Calibration Scan Pipeline:
  http://web.ipac.caltech.edu/staff/fmasci/home/wise/ScanPL_instrumental_cal.pdf
- Software Interface Specification (ICL01) Frame Processing Status Mask:
  http://web.ipac.caltech.edu/staff/fmasci/home/wise/InstruCal01.txt
- Software Specifications for the *flatcal* module:
  http://web.ipac.caltech.edu/staff/fmasci/home/wise/flatcal_specs.pdf
- IRAS Zodiacal Model:
  http://irsa.ipac.caltech.edu/IRASdocs/issa.exp.sup/Ap/G.html

1.3 Acronyms

2MASS	Two-Micron All-Sky Survey
DN	Data Number
FITS	Flexible Image Transport System
FRD	Functional Requirements Document
IOC	In-Orbit Checkout
NAXIS1	Number of columns in an image
NAXIS2	Number of rows in an image
Nframes	Number of science images to be processed
SDS	Subsystem Design Specification
SIS	Software Interface Specification
SUR	Sample Up the Ramp
W1	WISE wavelength channel 1, 3.4 microns
W2	WISE wavelength channel 2, 4.6 microns
W3	WISE wavelength channel 3, 12 microns
W4	WISE wavelength channel 4, 22 microns
WISE	Wide-field Infrared Survey Explorer
WSDC	WISE Science Data Center
WSDS	WISE Science Data System

2 Input

2.1 Control Input

Flatcal reads control input in the form of Fortran Namelist files and command-line parameters. For control parameters included in both command line and namelist inputs, the command line inputs override.

2.1.1 Flatcal Command-Line Parameters

The command-line parameters for flatcal are given by its tutorial display:

```
flatcal version: 2.0  A90812 - execution begun on 28-08-09 at 15:31:26

Usage: flatcal

-f1 <inp_img_list_fname>    (Required; list of input pre-calibrated frames in
                             FITS format)

-f2 <inp_mask_list_fname>   (Optional; list of input bad-pixel masks in 32-bit
                             INT FITS format; only values 0 -> 2^31 are used)

-f3 <inp_unc_list_fname>    (Optional; list of input uncertainty images in
                             FITS format)

-lf <min_frame_signal>      (Optional; minimum median frame signal to retain;
                             units = native image units; Default = no limit)

-hf <max_frame_signal>      (Optional; maximum median frame signal to retain;
                             units = native image units; Default = no limit)

-lt <lower_threshold>       (Optional; lower-tail SNR threshold for in-frame
                             outlier trimming; Default = 5)

-ut <upper_threshold>       (Optional; upper-tail SNR threshold for in-frame
                             outlier trimming; Default = 5)

-ng <num_grids>             (Optional; number of partitions per axis for
                             robust outlier detection; Default = 1)

-m <inp_mask_bits>          (Optional; mask template [decimal] specifying
                             bits to flag/omit from processing; Default=0)

-r                          (Optional switch; if set, inflate or deflate input
                             uncertainties to obtain reasonable chi-squares
                             and parameter uncertainties; default = off)

-o1 <out_slope_img>         (Required; output flat-field [slope-fit]
                             image FITS filename)

-o2 <out_slope_unc_img>     (Required; output flat-field [slope-fit]
                             uncertainty image FITS filename)

-o3 <out_intercept_img>     (Optional; output intercept-fit image FITS
                             filename)

-o4 <out_intercept_unc_img> (Optional; output intercept-fit uncertainty
                            image FITS filename)

-o5 <out_co-std-dev_img>    (Optional; output co-standard deviation [between
                             slope and intercept] image FITS filename;
                             pixel value = sign[cov] sqrt[|cov|])

-o6 <out_mask_img>          (Optional; output 8-bit mask FITS filename
                             flagging pixels with unreliable/bad responsivity
                             estimates)
```

```
-o7 <outchisq_img>           (Optional; output chi-square-fit image FITS
                              filename)

-o8 <numfits_img>            (Optional; output #points-fit image FITS
                              filename)

-o9 <frame_median.tbl>       (Optional; output frame medians table-file name)

-n <namelist_file_name>      (Optional; name of file containing fltcalin
                              namelist specifications)

-rm <rel_min_sig>            (Optional; minimum value of residual sigma when
                              no priors are input, fraction of median pixel
                              value; Default = 0.001)

-d                           (Optional; switch to print debug statements
                              to stdout, ancillary QA to ascii files)

-v                           (Optional; switch to increase verbosity to stdout)
```

2.1.2 Flatcal Namelist Parameters

The Flatcal module optionally reads a NAMELIST file. The name of this file must be given on the command line via the "-n" option. The name of the NAMELIST is fltcalin. The parameters defined in the NAMELIST are as follows, where corresponding command-line flags, if any, are shown parenthetically inside quotation marks).

Name	Description	Dim	Type	Units	Default
BadFlat	Value used for flat when valid estimate is not possible	1	R*4	-	1.0e-10
Debug	If T, debug mode is turned on	1	L*4	-	F
DetMin	Minimum value for a deter-minant encountered during chi-square minimization flat estimation	1	R*8	-	1.0d-50
FlatSNmin	Minimum S/N for flat not to have flat mask bit #2 set	1	R*4	-	2.0
FramMedMax	Maximum frame median for frame to be used in flat estimation (see "-hf" " in section 2.1.1)	1	R*4	-	9.9e25
FramMedMin	Minimum frame median for frame to be used in flat estimation (see "-lf" in) section 2.1.1)	1	R*4	-	-9.9e25
IDump	Array of column numbers for pixels whose stacks are to be dumped to a table file (see section 4.10)	10	I*4	-	10*0
Infl8	If T, flat uncertainty will be inflated if chi-square is too large or deflated if chi-square is too small (see "-r" in section 2.1.1)	1	L*4	-	F
JDump	Array of row numbers for pixels whose stacks are to be dumped to a table file (see section 4.10)	10	I*4	-	10*0

Name	Description	Dim	Type	Units	Default
Mask	Mask value for omitting a pixel from flat computation (see “-m” in section 2.1.1)	1	I*4	-	0
MaskDump	Flat Mask value for second second type of pixel dump (see section 4.10)	1	I*4	-	126
MinPix	Minimum number of pixels for flat or frame median computation	1	I*4	-	5
NamWrt	If T, this namelist will be written to stdout during initialization	1	L*4	-	F
NDump	Number of pixels to be dumped of the two types of dump criteria (see section 4.10)	2	I*4	-	2*0
Ng	Number of grid divisions per image axis for partitions in which spatial outliers are detected	1	I*4	-	1
RelSigMin	Minimum value of residual sigma when no priors are input, fraction of median pixel	1	R*4	-	0.001
ThrshLo	Lower-tail #sigma for outlier rejection for trimming pixel stack for flat computation (see “-lt” in section 2.1.1)	1	R*4	-	5.0
ThrshHi	Upper-tail #sigma for outlier rejection for trimming pixel stack for flat computation (“-ut”)	1	R*4	-	5.0
Verbose	If T, verbose processing mode is turned on (“-v”)	1	L*4	-	F
ZSig	Maximum number of sigma for chi-square of flat not to trigger uncertainty inflation if Infl8 = T	1	R*4	-	3.0

2.2 ASCII Input

Flatcal reads one to three ASCII (text) file lists of FITS file names corresponding to FITS images. One is required, and its name is specified on the command line via the "-f1" flag; this is the list of names of science images. If the "-f2" flag was used, then a list of mask images is also read, and if the "-f3" flag was used, then a list of uncertainty images is read. These last two lists must be in one-to-one correspondence with the first list, i.e., the n^{th} mask image and the n^{th} uncertainty image must correspond to the the n^{th} science image. Each list is read, the number of lines is counted, and all must have the same number of lines; this number is used for memory allocation. Then the lists are rewound, each line is read, and the corresponding file is read into memory. The file name on each line must be left-justified and is case-sensitive. Path names may be included and are required if the FITS files are not in the working directory.

2.3 FITS Input

For each line in each ASCII file described in section 2.2 above, flatcal reads the FITS file whose name is given. All FITS images must have the same values for the following header parameters: NAXIS (must be 2), NAXIS1, NAXIS2, and BAND. Each frame must have its own parameter UNIXT, which is used to report the TIME range of data in output FITS headers. If the FRSETID parameter is in the header of the first input image file, then it must be in all input image headers and is used to report the range of frameset IDs in out put images via the FRMIDSEQ parameter.

3 Processing

3.1 Flatcal Processing

The flatcal module initializes itself by performing the following tasks.

A.) Reading and processing its control inputs.

B.) Verifying that all required inputs were given.

C.) Reading the lists of FITS files and allocating memory.

D.) Reading in all specified FITS images.

E.) Computing the partition limits; arrays of pixel coordinates are set up to control loops over partitions in terms of frame pixel coordinates. These arrays are named Ig0, IgF, Jg0, and JgF. A partition is identified by its index on each axis, Ig and Jg. For example the grid corresponding to the 2nd partition in the direction of increasing column number and the 3rd partition in the increasing row direction has Ig = 2 and Jg = 3. A nested loop over this partition would run from J = Jg0(3) to JgF(3) for the outer loop and from I = Ig0(2) to IgF(2) for the inner loop. These partition limits are computed as follows.

```
$\Delta C = \frac{N_{cols}}{N_g}$ , $\Delta R = \frac{N_{rows}}{N_g}$
$Ig0(1) = 1$, $Jg0(1) = 1$
$loop\ on\ N = 1\ to\ N_g$ :
    $IgF(N) = round(N \times \Delta C)$
    $JgF(N) = round(N \times \Delta R)$
    $if\ \ N > 1\ then\ Ig0(N) = IgF(N-1) + 1$
    $if\ \ N > 1\ then\ Jg0(N) = JgF(N-1) + 1$
$end\ loop$
```

Since we have $N_{cols} = N_{rows}$, the partitions will never be more than one row or column off from being square, and the entire frame will be covered.

3.2 Flat Computation

3.2.1 Abscissa Computation

The first step is to compute a robust median pixel value for each science image, i.e., a robust estimate of the representative sky background, the median signal that would be observed if there were no point-like objects or radiation hits. This will be called simply the “abscissa” for brevity, since this is the parameter to which each individual pixel’s output will be fit as a linear function to determine that pixel’s “flat” value, namely the slope of the linear fit..

During initialization, all frames to be processed were loaded into a three-dimensional data cube in memory. The amount of memory required for this is `4*NAXIS1*NAXIS2*Nframes` for the science images, and the same each for optional uncertainty images and optional mask images. For W1, W2, and W3, `NAXIS1` = `NAXIS2` = 1016, so for these science images, the amount of memory required in bytes is 4.12904e6*`Nframes`. For W4, `NAXIS1` = `NAXIS2` = 508, so for W4 science images, the amount of memory required in bytes is 1.032256e6*`Nframes`.

An example of a minimal processing run would be 50 W4 frames with no masks or uncertainties. The total input-image memory for this case is 51.6 MB. An example of a typical expected case would be 100 W1 frames including science, uncertainty, and mask images. The total input-image memory for this case is 1.24 GB.

A nested loop over all frames and partitions is executed, and in this loop, the following processing is performed on each partition (which is the whole frame if `Ng` = 1).

- A one-dimensional stack is loaded with all pixel values that are neither masked nor NaN.
- If the number of values in the stack is less than `MinPix` (see section 2.1.2), a partition abscissa value of NaN is returned (this will not be used unless `Ng` = 1); otherwise processing continues.
- The stack is sorted in ascending value, and the median value is found.
- The standard deviation of the lower 50%-tile about the median is computed, σ_{50}.
- All values in the stack that are either less than median–`ThrshLo`*σ_{50} or greater than median+`ThrshHi`*σ_{50} (see section 2.1.2) are compressed out of the stack and set to NaN in the image frame itself to prevent their being used later in pixel stacks.
- The remaining stack is re-sorted into ascending order, and the new median is found; this is the frame abscissa if `Ng` = 1.

If the number of partitions is greater than 1 (`Ng` > 1), then another loop over all frames is executed, this time without partitioning, and the following processing is performed on each frame, which may now have some pixels set to NaN by the nested loop above.

- A one-dimensional stack is loaded with all pixel values that are neither masked nor NaN.

- If the number of values in the stack is less than `MinPix` (see section 2.1.2), an abscissa value of NaN is returned; otherwise processing continues.
- The stack is sorted in ascending value, and the median value is found.
- The standard deviation of the lower 50%-tile about the median is computed, σ_{50}.
- All values in the stack that are either less than median–`ThrshLo`*σ_{50} or greater than median+`ThrshHi`*σ_{50} (see section 2.1.2) are compressed out of the stack and set to NaN in the image frame itself to prevent their being used later in pixel stacks; on the same pass, the dispersion about the current median is computed for the data retained in the stack; this is available in the table-file output ("-o9").
- The remaining stack is re-sorted into ascending order, and the new median is found; this is the frame abscissa.

3.2.2 Pixel Flat Computation

A loop over all array pixels is performed, and inside this loop each pixel is fit to a linear function of the frame abscissa. The slope of this line is the flat value, and the intercept is also computed. The processing is described below.

- If the frame abscissa value is NaN or less than `FramMedMin` or more than `FramMedMax` (see section 2.1.2), then this frame cannot be used.
- The pixel's value in each usable science frame in the data cube is examined; if no uncertainties were input, then if the pixel value is not NaN and not masked, then the value is put into a pixel stack; if uncertainties were input, then the same mask conditions must be met by the science and uncertainty values, and in addition the uncertainty must be greater than zero (including not NaN); if and only if these conditions are met, the science data value and the uncertainty value are both placed into separate stacks; an index array is filled which specifies from which frame number in the data cube the pixel came.
- If the number of values in the stack is less than 1, `BadFlat` (see section 2.1.2) is returned for the flat value, and zero is returned for the intercept; if `BadFlat` is not zero, then 1/`BadFlat` is returned as the uncertainty, otherwise 1.0e10; if mask output was selected, the pixel's value in the mask has bit 5 set.
- If the number of values in the stack is greater than 0, processing continues; if the number of values in the stack is less than `MinPix`, the same "failed" values are returned as in the previous item, except that the mask bit that is set is bit number 4 instead of 5.
- All values in the stack are used in the linear fit; this computation is described in Appendix A; if the number of values accepted for the linear fit is less than `MinPix`, then the same "failed" values described above are returned, and the mask bit is number 4; if the determinant of the linear system is less than `DetMin` (see section 2.1.2), then the same "failed" values are returned, and the mask bit is number 3; if the ratio of the flat to its uncertainty is less than `FlatSNmin`, then mask bit number 2 is set if mask output was requested.

- After the linear fit, sanity checking of the pixel flat is performed as described in Appendix A, which also discusses the uncertainty computation.

4 Output

4.1 Flat FITS Output File

The main output ("-o1" command-line parameter) is a FITS file containing the flat image. An example header is below. The angle brackets indicate information whose literal content depends on actual execution or generation circumstances.

```
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                  -32 / number of bits per data pixel
NAXIS   =                    2 / number of data axes
NAXIS1  =                 1016 / length of data axis 1
NAXIS2  =                 1016 / length of data axis 2
BAND    =                    1 / WISE band number (1, 2, 3 or 4)
NUMINP  =                   70 / Number of input frames used
UTCSBGN =           1260864418 / Earliest UTCS in frame stack [sec]
UTCSEND =           1261051979 / Latest UTCS in frame stack [sec]
FRMIDSEQ=         <'Num..Num'> / Range of frameIDs used
COMMENT  slope for WISE flat calibration
COMMENT  Generated by flatcal vsn 1.0  A80616 on 17-06-08 at 12:19:20
```

Notes:

- BITPIX = -32 refers to single precision floating point.
- For bands 1, 2 and 3, the input frames will have NAXIS1 = NAXIS2 = 1016. For band 4, NAXIS1 = NAXIS2 = 508.
- The UTCSBGN, UTCSEND keywords specify the start/end observation time of frames in the input ensemble. These time tags will be available in the input frame headers [e.g., UNIXT].
- The FRMIDSEQ specifies the range of input frame IDs: a string composed of the *min* and *max* ID delimited by two dots, e.g., '31412..31505'.

4.2 Flat Uncertainty FITS Output

The "-o2" command-line parameter is required to be specified, and a corresponding FITS file containing the flat uncertainties is generated. The header is similar to that of section 4.1 except that the COMMENT fields state that this is a slope 1-sigma uncertainty image.

4.3 Optional Intercept-Fit FITS Output File

If the "-o3" command-line parameter was specified, a FITS file containing the intercepts of the linear fits is generated. The header is similar to that of section 4.1 except that the COMMENT fields state that this is an intercept image.

4.4 Optional Intercept-Fit Uncertainty FITS Output File

If the “-o4” command-line parameter was specified, a FITS file containing the uncertainties of the intercepts of the linear fits is generated. The header is similar to that of section 4.1 except that the COMMENT fields state that this is an intercept uncertainty image.

4.5 Optional Slope-Intercept Co-Standard Deviation FITS Output File

If the “-o5” command-line parameter was specified, a FITS file containing the co-sigma values for the slopes and intercepts of the linear fits is generated. The header is similar to that of section 4.1 except that the COMMENT fields state that this is a co-sigma image.

4.6 Optional Flat Mask FITS Output File

If the “-o6” command-line parameter was specified, a FITS file containing the flat mask values is generated. The header is similar to that of section 4.1 except that BITPIX = 8, and the COMMENT fields state that this is a flat mask image. The bit definitions are as follows.

Bit #	Condition
0	uncertainty needed to be rescaled downward (not fatal)
1	uncertainty needed to be rescaled upward (not fatal)
2	slope/sigma_slope below threshold (probably not fatal, TBD)
3	determinant too close to zero (fatal)
4	lost too many data points to outliers (fatal)
5	no data to fit (fatal)

4.7 Optional Chi-Square FITS Output File

If the “-o7” command-line parameter was specified, a FITS file containing the reduced chi-square values for the linear fits is generated (see Appendix A). The header is similar to that of section 4.1 except that the COMMENT fields state that this is a reduced chi-square image.

4.8 Optional Sample-Size FITS Output

If the “-o8” command-line parameter was specified, a FITS file containing the number of points used in the linear fits is generated. The header is similar to that of section 4.1 except that the COMMENT fields state that this is an image of the number of fitting points.

4.9 Optional Frame Abscissa Table File Output

If the “-o9” command-line parameter was specified, a table file containing the frame abscissa values, the frame noise in non-outlier samples, and corresponding UNIXT values is generated.

4.10 Optional Pixel Stack Dump Table File Output

Pixels may be selected to have their stacks dumped in two ways: (a.) by column and row coordinates, up to ten pairs; (b.) by bits set in the flat mask, up to a user-specifiable number of pixels. Type (a.) is accomplished by setting the namelist parameters `IDump`, `JDump`, and `NDump(1)` (see section 2.1.2); for example to dump three pixels, those with coordinates (17,342), (625,922), and (472,562), the namelist would contain:

```
      IDump = 17, 625, 472,  JDump = 342, 922, 562,  NDump(1) = 3
```

Type (b.) is acomplished by setting the namelist parameter `MaskDump` to the bitwise OR of all flat mask bits of interest, and setting a limit for the number of pixels to be dumped via the parameter `NDump(2)` ; for example, to dump the first 50 pixels whose flat mask contains bits 3 or 5 in the “on” state, the namelist would contain:

```
      MaskDump = 40,  NDump(2) = 50
```

where 40 = 101000 binary, i.e., bits 3 and 5 are on. When a pixel’s stack is dumped by either means, a table file is generated named PixDump***_I_J***.tbl, where ***I*** and ***J*** are 4-digit decimal numbers equal to the column and row coordinates of the pixel, respectively. For example, the stack for pixel (472,562) would be dumped in a file named PixDump_0472_0562.tbl. The header would provide the values for the number of frames in the stack, the flat mask value, the reduced chi-square of the linear fit, the flat (slope of the linear fit), and the intercept of the linear fit. The data rows contain the pixel value and its uncertainty (if no priors were provided, then the latter is retro-fit from the dispersion and a robust estimate of population sigma), the linear-fit values of the abscissa and fit slope, the fit residual, and the frame number corresponding to the abscissa..

The first 14 and last 8 lines from a sample file are shown below.

```
\NPixStack =  251
\MaskFlat =  2
\ChiSq = 1.288E+00
\Flat = 1.002E+00
\Intercept =-6.945E+00
|   Pixel  |  SigPix |   Absc    |    Fit    |  Resid   |NFrm|
|   real   |   real  |   real    |   real    |   real   | int|
  1.274E+03 3.096E+01  1.311E+03  1.307E+03 -3.370E+01    1
  1.277E+03 3.097E+01  1.316E+03  1.312E+03 -3.498E+01    2
  1.368E+03 3.128E+01  1.317E+03  1.313E+03  5.494E+01    3
  1.330E+03 3.115E+01  1.310E+03  1.306E+03  2.317E+01    4
  1.326E+03 3.114E+01  1.310E+03  1.306E+03  2.003E+01    5
  1.290E+03 3.102E+01  1.311E+03  1.307E+03 -1.708E+01    6
  1.323E+03 3.113E+01  1.314E+03  1.310E+03  1.294E+01    7
    .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
  1.507E+03 3.174E+01  1.516E+03  1.512E+03 -5.201E+00  244
  1.559E+03 3.192E+01  1.516E+03  1.513E+03  4.624E+01  245
  1.490E+03 3.169E+01  1.517E+03  1.514E+03 -2.411E+01  246
  1.554E+03 3.190E+01  1.518E+03  1.515E+03  3.921E+01  247
  1.481E+03 3.166E+01  1.519E+03  1.516E+03 -3.509E+01  248
  1.528E+03 3.181E+01  1.521E+03  1.518E+03  1.056E+01  249
  1.481E+03 3.166E+01  1.522E+03  1.519E+03 -3.797E+01  250
  1.542E+03 3.186E+01  1.524E+03  1.521E+03  2.107E+01  251
```

5 Testing and Parameter Tuning

5.1 Flatcal Testing

The flatcal module has been unit-tested with simulated data hand-edited to force the various processing paths to be traversed. It has also been tested with simulated FITS data generated by N. Wright for a mid-ecliptic-latitude region around zero degrees longitude and 30 degrees latitude, with a background ramp artificially added to simulate a zodiacal gradient.

5.2 Flatcal Parameter Tuning

The flatcal input parameters which control robust estimation of the sky offset and transient pixel behavior will require tuning prior to their use in routine pipeline processing. The chi-square image should prove useful in this activity, since effective outlier rejection should leave a set of pixel samples which is well behaved with respect to prior uncertainty and local background noise. The two most important parameters are (see section 2.1.2) `ThrshHi` and `ThrshLo.` Also of special interest are `MinPix` and `ChiSqMax`. Finally, the optimal number of consecutive frames has to be determined.

5.3 Potential Caveat

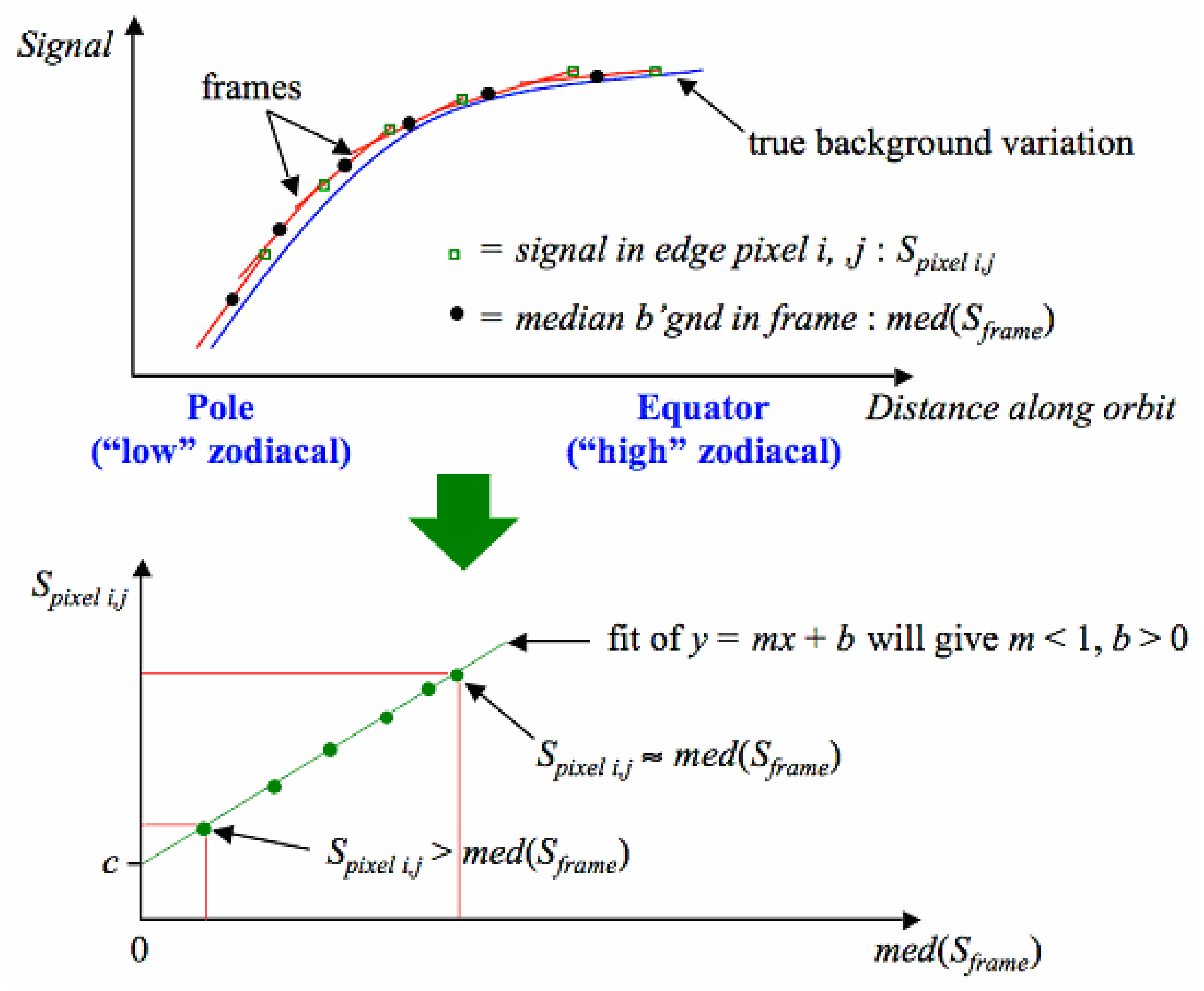
It was mentioned above that for the *slope method* to be reliable, *every* pixel in an array must see the same intrinsic *change* in the background on average. An example where this may fail is if the background intensity exhibits some curvature along a scan. This implies that there will be a gradient that changes from frame-to-frame. The signals in pixels (*relative* to the frame median background) located at the extreme edges of a frame along the *in-scan* direction will change in a systematic manner along the scan. This will bias their slope (hence relative responsivity) estimates towards either the low or high side depending on where the pixels are located. A schematic of this is shown in Figure 1.

Whether this effect is significant will not be known until in-orbit checkout. It may be that the dynamic range (i.e., ~40-50% background variation from an ecliptic pole to equator, irrespective of longitude) is too small for curvature effects to dominate over the background fluctuations. If not, an additional prefiltering step will be needed to ensure the input frames don't exhibit gradients that *depend systematically* on the total signal, e.g., only use frames from the (linear) portion of a scan, assuming such exists, or use data that sample opposite curvatures symmetrically. In other words, any *gradients* in frames along a scan must be more or less random and unbiased and independent of signal.

To check if global curvature effects are biasing the slope (hence responsivity), one can test either of the following hypotheses for the slope (m) and intercept (b) for pixels located respectively at the maximum/minimum extremities of a frame along the in-scan direction:

$$(b > 0 \text{ and } m < 1) \text{ and } (b < 0 \text{ and } m > 1)$$

where "maximum" here is defined as that frame edge *closest* to the ecliptic equator. If these conditions are found at a level that is *significant*, i.e., unlikely to have occurred by chance given the errors in b and m, then the data are not adequate for the purposes at hand. As a quick check, one can also examine the m and b images directly for patterns consistent with the above. It may also help to check that the median background does indeed exhibit *curvature* in the expected direction (i.e., is *concave-down* as in Fig. 1 where $d^2 med(S)/dt^2 < 0$, and t is time measured for a frame sequence going from pole to equator). If these conditions are satisfied, then further preprocessing of the input data is needed. These checks are beyond the scope of the flatcal module. This caveat is described here for future reference in case it becomes a problem. The table-file output ("-o9"), however, should aid in judging background curvature in the input data.



Signal
frames
true background variation
▫ = signal in edge pixel i, j : $S_{pixel\ i,j}$
● = median b'gnd in frame : $med(S_{frame})$
Pole
("low" zodiacal)
Equator
("high" zodiacal)
Distance along orbit
$S_{pixel\ i,j}$
fit of $y = mx + b$ will give $m < 1, b > 0$
$S_{pixel\ i,j} \approx med(S_{frame})$
$S_{pixel\ i,j} > med(S_{frame})$
c
0
$med(S_{frame})$

Figure 1. Schematic showing exaggerated curvature in the background and its impact

The effects of curvature on pixels at different frame locations might be mostly mitigated by symmetric sampling of the zodiacal emission about the ecliptic, since for example, a leading-edge pixel will suffer opposite effects approaching the ecliptic and receding from it. A first-order analysis of this possibility was performed by using two extremely simple zody models which are grossly similar to the latitude distribution found by IRAS at 12 and 25 microns (see Figure 2).

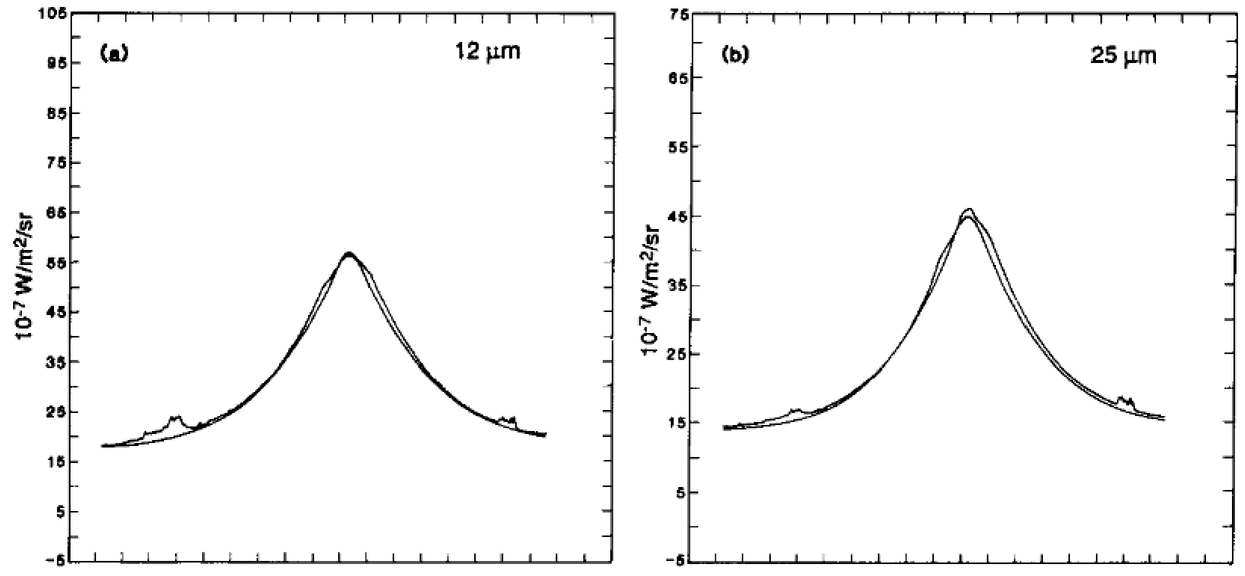


Figure 2. Zodiacal emission at 12 and 25 microns, solar elongation 90 degrees (IRAS); the smooth curves are the J. Good models

The simplified zody models used in the analysis herein were powers of the cosine of latitude scaled by 10,000 DN; the powers were 4 and 16. The latter produced a narrower concentration, which should aggravate curvature effects. Sampling was simulated by taking frame centers every one degree in latitude from -90 to +90. A leading-edge pixel was taken to be half a degree in front of the frame center. This results in the frame center and leading-edge pixels never seeing exactly the same zody, which also aggravates curvature effects. To isolate the latter, noiseless data were assumed. The 181 samples for each pixel were used in the fitting algorithm desribed in Appendix A. The true values of the flat field and intercept for the leading-edge pixel were 1.0 and 0.0, respectively. The sampling is illustrated in Figure 3.

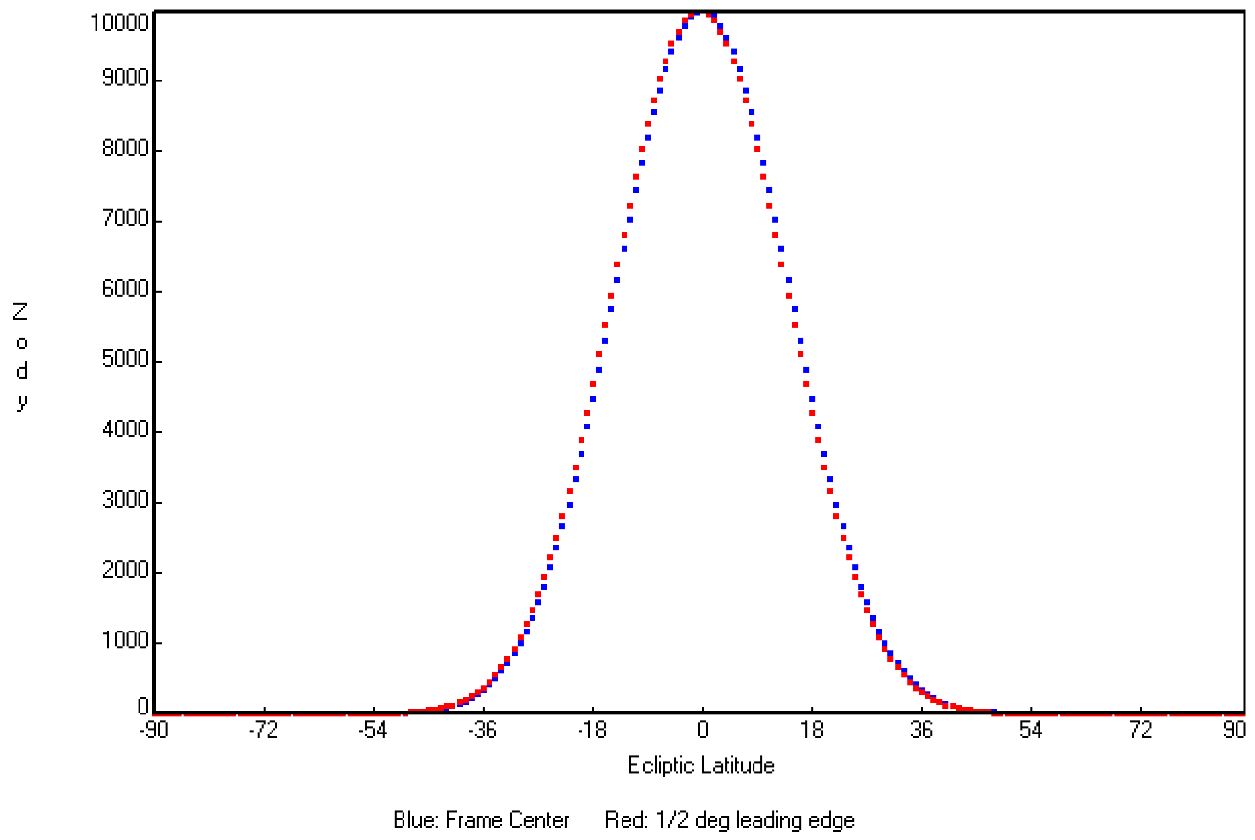


Figure 3. Simple Zody Model varying as $\cos^{16}$(latitude) with pixels sampled a half-degree apart every degree.

When the entire range of latitudes was used, the values of the flat and intercept were 0.999567 and 0.84578, respectively. The broader 4th -power cosine model came even closer to the true values. When only latitudes greater than or equal to zero were used in the fit, however, much larger errors were observed: the flat and intercept were 0.98372 and -22.431, respectively.

6 Example Command Lines

The following example reads in a list of image FITS file names (`-f1`), corresponding masks (`-f2`), and uncertainty images (`-f3`). The output flat image (`-o1`) will be named flat.fits, and the output flat uncertainty image (`-o2`) will be named sig_flat.fits. A flat mask image named mask_flat.fits will be generated (-o6), a reduced chi-square image (`-o7`) to be named chsq.fits is requested, and a sample-size image (`-o8`) named nused.fits is also requested. Samples with input mask bits 1 and 2 set will not be processed (`-m`) since the value 6 was specified, and 6 = $2^2 + 2^1$.

```
% flatcal -f1 intlist -o1 flat -o2 sig_flat –o6 mask_flat \
          -f3 unclist -f2 masklist -m 6 –o7 chsq –o8 nused
```

Appendix A Linear Fit

A.1 Chi-square Minimization

The linear fit of a pixel's value as a function of the frame abscissa is obtained via chi-square minimization if prior uncertainties have been provided (command-line option –f3). Given N usable values y_{in} for pixel i in frame n, and corresponding frame abscissa values x_n, n = 1 to N, and given one-sigma uncertainties σ_{in} for each y_{in}, the following summations are computed.

$$\begin{aligned} K &= \sum_{n=1}^{N} \frac{1}{\sigma_{in}^2} \\ K_x &= \sum_{n=1}^{N} \frac{x_n}{\sigma_{in}^2} \\ K_y &= \sum_{n=1}^{N} \frac{y_{in}}{\sigma_{in}^2} \\ K_{xx} &= \sum_{n=1}^{N} \frac{x_n^2}{\sigma_{in}^2} \\ K_{xy} &= \sum_{n=1}^{N} \frac{x_n y_{in}}{\sigma_{in}^2} \end{aligned} \tag{A.1}$$

The slope m_i and intercept b_i are obtained via standard chi-square minimization formulas as follows.

$$\begin{aligned} m_i &= \frac{K K_{xy} - K_x K_y}{D} \quad , \quad D \equiv K K_{xx} - K_x^2 \\ b_i &= \frac{K_{xx} K_y - K_x K_{xy}}{D} \end{aligned} \tag{A.2}$$

The one-sigma uncertainties in m_i and b_i, σ_{mi} and σ_{bi}, respectively, and their covariance, cosig(m_i,b_i), are obtained from:

$$\begin{aligned} \sigma_{mi}^2 &= \frac{K}{D} \\ \sigma_{bi}^2 &= \frac{K_{xx}}{D} \\ \mathrm{cov}(m_i, b_i) &= \frac{-K_x}{D} \\ \cos ig(m_i, b_i) &= sign(\mathrm{cov}(m_i, b_i)) \sqrt{|\mathrm{cov}(m_i, b_i)|} \end{aligned} \tag{A.3}$$

Note that these formulas treat the range of *n* from 1 to *N* as inclusive, with a total of *N* points used in the fit; in practice, the values of the *N* points are obtained via the index array that relates a given data point to its frame number.

Chi-square for the fit is computed from:

$$\chi_i^2 = \sum_{n=1}^{N} \frac{\left(y_{in} - m_i x_n - b_i\right)^2}{\sigma_{in}^2} \tag{A.4}$$

The number of degrees of freedom is $N_F = N - 2$. The expected value for chi-square is equal to the number of degrees of freedom, and the variance of this expectation is $2N_F$. It is intended that N be fairly large (e.g., > 30), and so the chi-square can be assumed to be approximately asymptotically Gaussian. The deviation of chi-square from its expectation value in units of standard deviation is:

$$Z_i = \frac{\left|\chi_i^2 - N_F\right|}{\sqrt{2N_F}} \tag{A.5}$$

If Z_i > `ZSig` (see section 2.1.2; default = 3), the fit is considered anomalous; if this is diagnosed and if uncertainty inflation has been selected (command-line option "–r"), then the uncertainties are adjusted according to:

$$\begin{aligned} R_i &\equiv \sqrt{\frac{\chi_i^2}{N_F}} \\ \sigma_{mi} &\Leftarrow R_i \sigma_{mi} \\ \sigma_{bi} &\Leftarrow R_i \sigma_{bi} \\ \cos ig(m_i, b_i) &\Leftarrow R_i \cos ig(m_i, b_i) \end{aligned} \tag{A.6}$$

If the ZSig test is failed and mask output was requested, independently of "-r", mask bit 0 or 1 is set if the uncertainty was overestimated or underestimated, respectively, The value of chi-square returned for optional output into an image (command-line option –o7) is the reduced chi-square

$$\chi_{ri}^2 = \frac{\chi_i^2}{N_F} \tag{A.7}$$

A.2 Least Squares

If no prior uncertainty images have been supplied, the linear fit must be done via ordinary least-squares fitting. This involves computing the values in Equation A.1 above with the sigma values all set to 1.0, then computing the slope and intercept as in Equation A.2.

In order to compute uncertainties, a substitute for input prior data uncertainties must be obtained. This is done as follows. The fit residuals $y_{in} - m_i x_n - b_i$ are computed and sorted into an array denoted S, and the standard robust estimate of the sample sigma is used:

$$\sigma_{in} \approx \frac{S(84.13447\%) - S(15.86553\%)}{2} \tag{A.8}$$

where the percentages in parentheses indicate percentile values in the sorted array, and we have retained the i subscript to be consistent with the preceding equations even though all pixel samples in the stack have the same uncertainty estimate. Now Equation A.1 can be recomputed using this estimate, after which Equation A.3 supplies plausible uncertainties for the slope and intercept. Because the estimate provided by Equation A.8 is robust against outliers in the stack, Equations A.4 through A.7 may now be used without the tautological implications that would follow from simply computing the sample sigma by the usual RMS formula.