Notes on CALCOS

Philip E. Hodge, Charles (Tony) Keyes

August 27, 2009

1 Introduction

CALCOS can be run in any of three ways: (1) from PyRAF using the hstcos package, (2) from Python or PyRAF by importing calcos, (3) from the Unix command line by typing 'calcos filename', where 'filename' is the name of an association file or a raw file. These are described in greater detail below.

For TIME-TAG data most calibration steps are applied to the photon events list, applied by adjusting the pixel coordinates, by setting the data quality flag, or by setting the weight. The weight accounts for the flat field and deadtime corrections. ACCUM data are converted to a pseudo-events list with the same format as for TIME-TAG data, and the calibration steps are applied in a similar manner.

TIME-TAG data will be binned to a pair of images, "effective" counts (taking weight into consideration) and gross counts (just the counts, ignoring the weight). One-dimensional spectra are extracted from these images. Auto and GO wavecal spectra are handled the same way; however, tagflash wavecal spectra are extracted directly from the events list.

The offset determined from a wavecal is applied by adjusting the positions (both x and y) in the corrtag table, with a linear slope between wavecal measurements if there is more than one.

2 Calling Sequence

This section describes how to run CALCOS and gives the calling sequence, whether using the 'hstcos' package, importing CALCOS from Python, or running CALCOS from the Unix command line.

2.1 Running CALCOS from PyRAF using hstcos

If you start PyRAF (type ‘pyraf’) and load the hst_calib and hstcos packages, you can run calcos as if it were any other IRAF task. That is, you can ‘lpar calcos’, ‘epar calcos’, or just run it. The parameters are:

argument	*default*	*description*
`input`	`""`	association table or single raw file name
`verbosity`	`1 (0 to 2)`	set more or fewer messages
`savetmp`	`False`	save temporary files?
`outdir`	`""`	optional output directory name
`shift_file`	`""`	file to override wavecal shifts
`csum`	`False`	create ‘calcos sum’ image?
`compress`	`False`	compress the csum image?
`comp_param`	`"gzip,-0.01"`	compression parameters for csum
`binx`	`1`	bin factor in X for csum
`biny`	`1`	bin factor in Y for csum
`stimfile`	`""`	append stim locations to this file
`livefile`	`""`	append deadtime factors to this file
`burstfile`	`""`	append burst info to this file

For example:

```
--> calcos rootname_asn.fits outdir=new
```

Further details are given in the next section.

2.2 Running CALCOS from Python or PyRAF

You can do the following to run CALCOS from Python (or PyRAF):

```
% python
>>> import calcos
>>> calcos.calcos("rootname_asn.fits")
```

The following is a list of all the arguments and their default values.

argument	*default*	*description*
asntable	`""`	association table or raw file to be processed
outdir	None	the name of the output directory
verbosity	None	if not None, set verbosity to this level (0, 1, 2)
create_csum_image	False	if True, write an image of the counts detected at each pixel; used by OPUS for updating the "cumulative image"
binx	None	binning factor in X for the csum image
biny	None	binning factor in Y for the csum image
compress_csum	False	compress the csum image?
compression_parameters	`"gzip,-0.01"`	compression type, quantization level
shift_file	None	text file to override wavecal shift1
save_temp_files	False	save temporary files?
stimfile	None	if specified, the stim positions will be written to (or appended to) a text file with this name
livetimefile	None	if specified, the deadtime factors will be written to (or appended to) a text file with this name
burstfile	None	if specified, burst information will be written to (or appended to) a text file with this name

Use keyword=value syntax for parameters other than asntable; for example:
`calcos.calcos ("l61h23nvs_asn", outdir="july10")`.
The 'asntable' parameter may include a directory, in which case CALCOS will look for all the input files for the dataset in that directory. If an output directory 'outdir' is not specified, all the output files will be written to the default directory. By default, the _x1d_a.fits, _x1d_b.fits, _lampflash_a.fits and _lampflash_b.fits files (if FUV) will be deleted after concatenating to the _x1d.fits or _lampflash.fits file. Specify save_temp_files=True to keep these _a.fits and _b.fits files.

The "calcos sum" file (suffix _csum) is used by OPUS to add to a cumulative image, to keep a record of counts extracted at different regions of each detector. See "Requirements for COS Cumulative Images," by David Sahnow, for full details. A csum image for NUV or for FUV ACCUM will be 2-D, 1024x1024 for NUV and 16384x1024 for FUV. For FUV TIME-TAG, however, the image will be 3-D, with the pulse height as the third axis (length 32). The FUV TIME-TAG image is large enough to cause problems with I/O or disk space, so options have been implemented for reducing its size. The csum image (either FUV or NUV) may be binned in either the X (dispersion) or Y axis (but not in the PHA axis). If 'binx' or 'biny' is not specified (i.e. is None), the default binning for that axis will be used; currently the default binning is 1, i.e. no binning. The binning factors must divide the respective axis lengths. Independently of binning, the csum image may be compressed (this uses the CompImageHDU class in pyfits). The compression parameters are given as a string consisting of two components separated by a comma. The first part is the compression type, which may be "gzip", "rice", or "hcompress". The second part is the quantization level, which specifies how the floating-point image values are to be converted to integer before compression. This conversion results in a loss of information, so the quantization level should be set with the intention that the quantization will be in the noise. If the quantization level is positive, it is interpreted as a value relative to the RMS noise level in the image background; a value of 16, for example, means that the quantization level will be 1/16 of the noise level. If the quantization level is negative, it's the actual floating-point increment that corresponds to a difference of one in the scaled integer image. The compression in pyfits supports other parameters such as tile shape, and parameters for hcompress, but these are currently not included in calcos.

The shift file (option `--shift filename`) may be specified in order to override values of the wavecal shift (shift1a, shift1b, shift1c) in the dispersion direction. This is a text file with five columns. (Actually, there can be a sixth column to specify shift2, but that value is currently not used.) The columns are:

rootname	fpoffset	flash number	segment/stripe	shift1

The first four columns are used for identifying which particular lamp exposure is to be overridden; however, any or all of those columns may be given as “any” (without quotes), which matches any value. All strings are case insensitive. Blank lines and lines beginning with “#” will be ignored. A single shift file can be used for an entire association; that's why rootname is included as a selection criterion. Rootname in this case means the portion of the file name that precedes the suffix. Note that this is the name of a particular exposure, not an association name. The rootname is taken from the actual name of the raw file, rather than from the ROOTNAME keyword, so that if the raw file has been renamed without changing the keyword, the name in the shift file will be the name that most people would expect. It is redundant to specify both rootname and fpoffset. Both are included to make it easier to set shift1 to the same value for all rootnames for a given fpoffset. If rootname is specified, fpoffset can be given as “any”. The flash number is one indexed. The keywords (LMP_ON1, etc.) for lamp flashes are one indexed, following the FITS convention, and the information about the flashes and the shifts that is written to the trailer file (and standard output) also show the flash number starting with one. For those reasons it was felt that a user would more likely expect one indexing than zero indexing for flash number. The segment or stripe name should be the complete string “FUVA”, “FUVB”, “NUVA”, “NUVB” or “NUVC”, not just the single letter “A”, “B” or “C”. The shift itself is the pixel offset from the template lamp spectrum for FPOFFSET=0 (FP-POS=3); a positive shift means the features in the observed spectrum are at higher pixel number than the features in the template.

2.3 Running CALCOS from the Unix Command Line

As an alternative, CALCOS can be run from the Unix command line, e.g.:

```
% calcos -o new --stim stim.txt rootname_asn.fits
% calcos -v -o new rootname_asn.fits > log.txt
```

The command-line options are:

`-q`	quiet
`-v`	very verbose
`-s`	save temporary files
`-o` outdir	output directory name
`--csum`	write "calcos sum" file
`--compress` parameters	compression parameters (e.g. 'gzip,-0.01')
`--binx nx`	X binning factor for csum image
`--biny ny`	Y binning factor for csum image
`--shift` filename	text file to specify shift1 values
`--stim` filename	append stim locations to filename
`--live` filename	append deadtime factors to filename
`--burst` filename	append burst info to filename

Following the options, list one or more association files (rootname_asn) or raw files (rootname_raw) or corrtag files (give the full file name). See the previous section for further information about the options.

The value of argument 'asntable' can be a rootname with either "_asn" (this is the normal case) or "_raw" appended, or it can be a complete file name. The "_asn" tells CALCOS to look for an association table with name rootname_asn.fits, and the "_raw" tells CALCOS to look for raw files and other files (_pht.fits and _spt.fits) with the specified rootname. The MEMNAME column in an association table normally contains rootnames rather than complete file names. However, MEMNAME can be a complete file name (give only one name for FUV data, i.e. not both segments), and this option would be used if the input file or files are corrtag rather than rawtag or rawaccum.

Here is an example association table.

MEMNAME	MEMTYPE	MEMPRSNT
L9V211C2S	EXP-FP	true
L9V211C4S	EXP-AWAVE	true
L9V211C6S	EXP-FP	true
L9V211C8S	EXP-AWAVE	true
L9V211CAS	EXP-FP	true
L9V211CCS	EXP-AWAVE	true
L9V211CES	EXP-FP	true
L9V211CGS	EXP-AWAVE	true
L9V211010	PROD-FP	false

MEMNAME is the rootname of the member (i.e. member of the association). MEMTYPE is the role that the exposure has in the association. For example, EXP-AWAVE is an auto or GO wavecal, EXP-FP is an FP-POS science exposure, and PROD-FP indicates the name of the "product" (some of the final results will be renamed to this rootname). MEMPRSNT indicates whether the file(s) for the exposure are actually present at the time that CALCOS begins running. Normally, all will be present except for the product, but if any rootname is flagged as not present, CALCOS will not attempt to read those files.

If asntable is the name of an association table (or truncated to rootname_asn), CALCOS will process each input file whose rootname is listed in the table. For FUV there will normally be two input files (one for each segment) for each rootname listed in the association table, and both will be processed. If auto or GO wavecals were used for an observation, the way to tell CALCOS that the wavecal information should be applied to the science data is to include both the science and wavecal rootnames in an association table, and give the appropriate value in the MEMTYPE column (e.g. EXP-FP or EXP-AWAVE). If a dataset includes multiple FP-POS observations, the way to tell CALCOS to combine them is to include all their rootnames in an association table.

If asntable is the name of a raw input file (possibly truncated to rootname_raw), that one file (or pair of files if FUV) will be processed. For FUV data, specify the name of the file for one segment; if data for the other segment is also present in the directory, both files will be processed.

3 COS Output Files and Naming Conventions

3.1 General Rules

For all output files (except final corrected image, _flt, or calibrated spectrum, _x1d) stemming from a single exposure at a particular spectral element and central wavelength combination: use exposure "rootname" as first portion of filename identifier. For the final corrected image and calibrated spectrum stemming from a single exposure, use "productname" as first portion of filename identifier.

For any output file that is a product of the combination of more than one exposure, use "productname" as first portion of filename identifier.

3.2 Spectroscopy

For each individual spectroscopic exposure (rootname), i.e., for each individual FP-POS exposure, the pipeline produces the following output files:

For all exposures regardless of detector and mode:

rootname_asn.fits (association file to control calibration processing)
rootname_spt.fits (support file containing primary engineering information)

For NUV data:

rootname_rawtag.fits (for TIME-TAG) or rootname_rawaccum.fits (for ACCUM)
rootname_corrtag.fits (corrected EVENTS table)
rootname_flt.fits (corrected detector image)
rootname_counts.fits (detector image, not flat fielded)
rootname_lampflash.fits (only for TIME-TAG with TAGFLASH)
rootname_x1d.fits (calibrated spectrum file, one row for each stripe)

For FUV data:

For segment A:

rootname_rawtag_a.fits (for TIME-TAG) or rootname_rawaccum_a.fits (for ACCUM)
rootname_pha_a.fits (pulse-height distribution for segment A)
rootname_corrtag_a.fits (corrected EVENTS table for segment A)
rootname_flt_a.fits (corrected detector segment A image)
rootname_counts_a.fits (detector image, not flat fielded)

For segment B:

rootname_rawtag_b.fits (for TIME-TAG) or rootname_rawaccum_b.fits (for ACCUM)
rootname_pha_b.fits (pulse-height distribution for segment B)
rootname_corrtag_b.fits (corrected EVENTS table for segment B)
rootname_flt_b.fits (corrected detector segment B image)
rootname_counts_b.fits (detector image, not flat fielded)

Final combined calibrated spectrum (both segments together):

rootname_lampflash.fits (only for TIME-TAG with TAGFLASH)
rootname_x1d.fits (calibrated spectrum file, one row for each segment)

If only one exposure is taken for a grating and central wavelength combination, then the x1d file is copied to productname_x1dsum.fits, with changes in some header keywords, and with values in FLUX, ERROR, GROSS, NET and BACKGROUND columns set to zero where the DQ_WGT is zero.

Next, if more than one FP-POS exposure is taken in any order without changing the grating and central wavelength combination, the "rootname" files listed above are produced for each individual exposure, and one or more "product" files containing summations of individual exposures are also produced. Grand total calibrated "x1dsum" spectra are produced by combining data from all exposures from all FP-POS. Additionally, all FP-POS=1 exposures are weighted by their exposure times and combined into an "x1dsum1" calibrated extracted spectrum file, and so on for the other FP-POS positions utilized in the exposure sequence.

For calibrated spectrum files:

productname_x1dsum1.fits (sum of all FP-POS=1 exposures)
productname_x1dsum2.fits (sum of all FP-POS=2 exposures)
productname_x1dsum3.fits (sum of all FP-POS=3 exposures)
productname_x1dsum4.fits (sum of all FP-POS=4 exposures)
productname_x1dsum.fits (sum of all FP-POS positions; ultimate calibrated product)

3.3 Imaging

For NUV imaging observations all of the same types of files as for NUV spectroscopy are produced except that there are no _x1d files. A productname_fltsum.fits file is the final product. If multiple image exposures were taken consecutively, the fltsum file is the average of those exposures. If only one exposure was taken, the rootname_flt.fits file will be copied to productname_fltsum.fits; it's still an average, but an average of just one exposure.

3.4 Target Acquisition

ACQ/IMAGE data have two image sets (imsets), and the output flt and counts files will also contain two image sets.

rootname_rawacq.fits
rootname_flt.fits (corrected detector images)
rootname_counts.fits (detector images, not flat fielded)

Target acquisition observations other than ACQ/IMAGE produce the following much more limited set of output:

rootname_rawacq.fits

4 Keywords Read or Written by CALCOS

The following keywords are read from the primary or extension (EVENTS or SCI) header:

primary header keywords		
switches	*reference files*	*other keywords*
DQICORR	BPIXTAB	SEGMENT
RANDCORR	RANDSEED	OBSTYPE
TEMPCORR	BRFTAB	OBSMODE
GEOCORR	GEOFILE	EXPTYPE
IGEOCORR		OPT_ELEM
DEADCORR	DEADTAB	CENWAVE
FLATCORR	FLATFILE	APERTURE
DOPPCORR		TARGNAME
HELCORR		TAGFLASH
PHACORR	PHATAB	
BRSTCORR	BRSTTAB	FPPOS
BADTCORR	BADTTAB	FPOFFSET
X1DCORR	XTRACTAB	RA_TARG
WAVECORR	WCPTAB	DEC_TARG
	LAMPTAB	SUBARRAY
	DISPTAB	DEVENTA
BACKCORR		DEVENTB
FLUXCORR	FLUXTAB	MEVENTS
TDSCORR	TDSTAB	
PHOTCORR	IMPHTTAB	
STATFLAG		

extension header keywords	
DISPAXIS	NSUBARRY
PLANTIME	CORNERiX
EXPTIME	CORNERiY
EXPSTART	SIZEiX
EXPEND	SIZEiY
DOPPON	NUMFLASH
DOPPMAG	LMP_ONi
DOPPZERO	LMPOFFi
ORBITPER	LMPDURi
SDQFLAGS	

For the CORNER and SIZE keywords, i is an integer running from 1 to NSUBARRY. For the LMP keywords, i runs from 1 to NUMFLASH.

The calibration switches listed above for the primary header will normally be changed (in the output) from PERFORM to COMPLETE; in addition, a few more primary header keywords can be updated. Most other keywords that are updated are in the extension header. These keywords (in addition to the calibration switches) can be updated:

primary header keywords
NEXTEND
FILENAME
MINWAVE
MAXWAVE
BANDWID
CENTRWV
APERTURE
RANDSEED

extension header keywords				
V_HELIO	SHIFT1A	PHA_BADA	STIMA_LX	STIMB_LX
GLOBRATE	SHIFT1B	PHA_BADB	STIMA_LY	STIMB_LY
BSCALE	SHIFT1C	PHALOWRA	STIMA_RX	STIMB_RX
BZERO	DPIXEL1A	PHALOWRB	STIMA_RY	STIMB_RY
BUNIT	DPIXEL1B	PHAUPPRA	STIMA0LX	STIMB0LX
EXPTIME	DPIXEL1C	PHAUPPRB	STIMA0LY	STIMB0LY
EXPSTART	SHIFT2A	SP_LOC_A	STIMA0RX	STIMB0RX
EXPEND	SHIFT2B	SP_LOC_B	STIMA0RY	STIMB0RY
EXPSTRTJ	SHIFT2C	SP_LOC_C	STIMASLX	STIMBSLX
EXPENDJ	NUMFLASH	SP_SLP_A	STIMASLY	STIMBSLY
PLANTIME	LMP_ONi	SP_SLP_B	STIMASRX	STIMBSRX
NGOODPIX	LMPOFFi	SP_SLP_C	STIMASRY	STIMBSRY
GOODMEAN	LMPDURi	SP_WIDTH		
GOODMAX	LMPMEDi	X_OFFSET		

For the LMP keywords, i runs from 1 to NUMFLASH.

5 Calibration Steps

Here is a list of all the keywords for calibration switches, shown approximately in the order in which the steps are performed for TIME-TAG data. STATFLAG is actually done multiple times, separately for different output files.

switch	*description*
BRSTCORR	search for bursts (FUV)
BADTCORR	flag bad time intervals that are listed in bad-times ref table (FUV)
PHACORR	filter based on pulse height (FUV)
RANDCORR	add pseudo-random numbers to pixel coordinates (FUV)
TEMPCORR	linear, temperature-dependent geometric correction (FUV)
GEOCORR	geometric correction (FUV)
IGEOCORR	interpolate within geo-correction file when doing geocorr (FUV)
DQICORR	initialize DQ extension from bad-pixel table
DOPPCORR	apply Doppler correction to pixel coordinates (TIME-TAG)
FLATCORR	divide by flat field
DEADCORR	correct for dead time
WAVECORR	correct pixel coordinates based on wavecal
X1DCORR	extract 1-D spectrum
HELCORR	apply heliocentric correction to wavelengths
BACKCORR	subtract background from 1-D spectrum
FLUXCORR	apply flux calibration to 1-D spectrum
TDSCORR	time-dependent sensitivity correction to the flux
no switch	average repeated observations
STATFLAG	compute statistics and populate header keywords with results

5.1 Pseudo-Corrtag Files

For data taken in ACCUM mode, the image will be converted to a "pseudo-corrtag table" and will be written to disk with the same format as if the input were TIME-TAG. Most calibration steps will be done the same for TIME-TAG and ACCUM data. FLATCORR differs because of the Doppler correction, which is done on-board for ACCUM exposures. DEADCORR is not quite the same for ACCUM data, and PHACORR is not done at all for ACCUM.

The number of rows in the pseudo-corrtag table will be the total number of counts in the input image. The RAWX and RAWY columns will be set based on the counts in the input image. For each pixel in the image, the X and Y coordinates of that pixel will be assigned to the RAWX and RAWY columns in the pseudo-corrtag table, and they will be included N times (if N is greater than zero), where N is the number of counts in the pixel. The values in the TIME column will all be the same, half the exposure time. The PHA column will be zero.

5.2 Buffer Zones for NUV Spectroscopic Images

For spectroscopic data, the FP-POS optional parameter can be used to take exposures with offsets of from -1 to +2 steps of the optic select mechanism (OSM) from the nominal OSM position. For NUV data one step results in a shift of the spectrum by about 50 pixels on the detector. These shifts are measured during wavecal processing (see below), and the XFULL column in the corrtag table includes the shift to the nominal position (keyword FPOFFSET = 0). The pixel values in the flt and counts images are based on the XFULL and YFULL columns, so those images can be shifted by of order 50 to 100 pixels from nominal. In order to avoid loss of data due to pixels being shifted off the image, the output images (for NUV spectroscopic data only) are made larger than the detector; the image size is 1274×1024 rather than 1024×1024. The buffer zone is 100 pixels on the left side of the image and 150 pixels on the right side. The size of the buffer zone on the left is given by the keyword X_OFFSET in the first extension of the FITS file. For NUV data X_OFFSET will be either 0 or 100 (0 in raw data and for imaging data). For FUV data the pixel offset for each step is about 250 pixels, but the active area of the detector is so far from the edges that no data are lost due to FP-POS offsets. X_OFFSET is therefore 0 for FUV data.

5.3 Screening for FUV Bursts

Screening for bursts (BRSTCORR) is done only for FUV. The burst detection algorithm in CALCOS is very similar to the one used in cf_screen_burst.c for FUSE. The reference file BURSTTAB specifies the parameters used for burst detection. In addition, the BRFTAB specifies the location of the FUV active region. Calcos looks for bursts within the active region but excluding the region near where the target spectrum is expected to be found.

Burst detection is based on the counts in two temporary arrays that contain the number of background and source events in uniformly spaced intervals of length delta_t seconds (these would be count rates if delta_t were 1 s). delta_t is either the value of DELTA_T (15 s) from the BURSTTAB, or for high count rate data it would be DELTA_T_HIGH (1 s). The distinction is based on the overall count rate, the number of events divided by the exposure time from the EXPTIME keyword, which includes stim counts as well as source and background counts and bursts. The array bkg_counts of background counts is the number of events within the active region but not close to the target spectrum. The array src_counts of source counts is the number of events close to the target spectrum minus the expected number of background counts within that region. Most of the tests for bursts (described below) are based just on bkg_counts, but one also uses src_counts.

The source region is based on B_SPEC (rounded to an integer), not taking the slope into account, and HEIGHT. The source region is from B_SPEC - HEIGHT/2 to B_SPEC + HEIGHT/2. There are two background regions. The lower background region runs from the low limit of the active area (301 or 371) to HEIGHT/4 below the low limit of the source region (i.e. to B_SPEC - (3/4)·HEIGHT). The location of the upper background region depends on whether the exposure includes a tagflash wavecal. If tagflash is not used, the upper background region runs from HEIGHT/4 above the upper limit of the source region (i.e. from B_SPEC + (3/4)·HEIGHT) to the upper limit of the active area (750 or 770). For tagflash data, the lower limit is raised (to avoid the wavecal spectrum) to (3/4)·HEIGHT above the nominal location of the wavecal spectrum.

Screening for bursts is done in two stages, a search for "large" bursts and a search for "smaller" bursts. In both stages, a burst is indentified as an outlier in bkg_counts. For each such burst, all events within that time interval will be flagged in the data quality column (DQ in the corrected TIME-TAG events table) with data quality bit DQ_BURST = 64. They will also be flagged in bkg_counts by setting the value to a negative number, to make it easy to ignore in subsequent processing and easy to identify later for writing info to an optional output file. This output file (specify 'burstfile=filename' when running CALCOS) contains four columns, one row per delta_t time interval:

(1) the time (seconds) at the middle of the time interval
(2) the background counts for that time interval
(3) 1 if the current time interval contains a large burst, else 0
(4) 1 if the current time interval contains a small burst, else 0

The search for large bursts is straightforward. First the median value of bkg_counts is determined. Then any element in bkg_counts that is greater than median_n times the median is flagged as a burst, where median_n is currently set to 4.0 in the BURSTTAB. Note that this test is not based on the expected variation from the median; the cutoff is simply a factor times the median.

The search for smaller bursts is done iteratively, up to max_iter (10) times. This search also uses a median, but it's a boxcar filter that takes the median within the box. For each element of bkg_counts, a box of length (median_dt / delta_t) elements is taken, centered on the element, and rounded up to an odd number if the quotient is even. (The box is truncated on one side as an endpoint is approached.) The default value of median_dt is 600 seconds. The median of non-negative elements within the box is taken as the filtered value. The rejection criteria are based on the difference delta_counts between an element of bkg_counts and the filtered value. For element i to be flagged as a smaller burst, the following three criteria must all be met:

delta_counts > burst_min · delta_t
delta_counts > stdrej · sqrt (bkg_counts[i])
delta_counts > source_frac · src_counts[i]

The default values of these parameters are: burst_min = 5, delta_t = 15 (or 1 for high count rate data), stdrej = 5, source_frac = 0.001.

There are (at least) three differences between what is done in the COS code and what is done in cf_screen_burst.c for FUSE data. (1) Calcos does not attempt to exclude counts due to geocoronal emission. If this is necessary it can be added, based on segment, opt_elem and cenwave, and the dispersion relation. (2) The FUSE code checks for zeros in the background counts array and interprets these as dropouts. [Does COS suffer from dropouts? If so, testing for zeros would be easy to include.] (3) The last of the above criteria for smaller bursts (delta_counts > source_frac · src_counts[i]) differs in detail from the FUSE algorithm, but the intent should be the same; the test is a simple interpretation of the description of source_frac in the FUSE code. cf_screen_burst.c uses an indirect method of estimating the source count rate, based on the total count rate, background count rate and the sizes of the source region (aperture) and the active area.

5.4 Doppler Correction

This section describes DOPPCORR and HELCORR. DOPPCORR is a correction for the orbital motion of HST around the Earth, while HELCORR is a correction for the orbital motion of the Earth around the Sun. The heliocentric correction will be applied by adjusting the wavelengths for the extracted spectrum.

For ACCUM mode, the Doppler correction for HST orbital motion is done on-board during the exposure. This means, however, that the pixel coordinates of a feature in a spectrum are not necessarily the same as the location where the photons hit the detector. This distinction makes a difference for the data quality initialization and the flat field correction. Calcos will therefore shift the pixel coordinates in the bad pixel table and the flat field image by the amount of the Doppler shift (or, in general, convolve with a histogram of the shift during the exposure) before applying the correction to the data. DOPPCORR for ACCUM data makes use of the keywords shown in the following table. Keywords dopzerot, dopmagt and orbtpert are the values that were used on-board for the correction during the exposure.

expstart	exposure start time (MJD)
exptime	duration (seconds) of the exposure
dopzerot	time (MJD) when the Doppler shift was zero and increasing, i.e. when HST was closest to the target
dopmagt	amplitude (pixels) of the Doppler shift
orbtpert	period (seconds) of HST orbit

For TIME-TAG mode, the raw events table will have the actual detector coordinates of each photon event. The orbital Doppler correction will be applied by adding an offset to the pixel coordinates in the events table, with the corrected coordinates being written to column XDOPP. DOPPCORR for TIME-TAG data depends on the values of the following four keywords and table column. Keywords doppzero, doppmagv and orbitper are calculated from the orbital elements by OPUS during the calibration.

expstart	exposure start time (MJD)
doppzero	time (MJD) when the Doppler shift was zero and increasing, i.e. when HST was closest to the target
doppmagv	amplitude (km/s) of the Doppler shift
orbitper	period (seconds) of HST orbit
time	column of the time of each event (seconds since expstart)

For both FUV and NUV, wavelengths increase toward larger pixel number, so the shift to be subtracted from the TIME-TAG pixel coordinates is:
$shift = doppmagv/c \cdot \lambda/\mathrm{d}\lambda \cdot \ sin(2\pi \cdot t/orbitper)$
where t is the time of each event in seconds since doppzero:
$t = (expstart - doppzero) \cdot 86400 + time$

The radial velocity of a target due to the Earth's orbital velocity around the Sun is computed as follows. The sign is taken to be positive if the distance to the target is increasing.

The components $\dot{x}$, $\dot{y}$, $\dot{z}$ of the Sun's velocity vector (in astronomical units per day) with respect to the Earth are computed using the expressions below. Angular constants are shown in degrees, though of course they are converted to radians in calcos.

$\Delta t = t_m - 51544.5$, the time in days since 2000 Jan 1.5, where t_m is the time (MJD) at the midpoint of the exposure. ϵ is the obliquity of Earth's axis. L and R are approximate values for the longitude of the Sun and the distance from the Earth to the Sun, respectively.

$$\begin{aligned}
\epsilon &= 23.439 - 0.0000004 \cdot \Delta t \\
g &= 357.528 + 0.9856003 \cdot \Delta t \\
l &= 280.461 + 0.9856474 \cdot \Delta t \\
\dot{g} &= 0.9856003 \\
\dot{l} &= 0.9856474 \\
L &= l + 1.915 \cdot sin(g) + 0.02 \cdot sin(2 \cdot g) \\
\dot{L} &= \dot{l} + 1.915 \cdot cos(g) \cdot \dot{g} + 0.04 \cdot cos(2 \cdot g) \cdot \dot{g} \\
R &= 1.00014 - 0.01671 \cdot cos(g) - 0.00014 \cdot cos(2 \cdot g) \\
\dot{R} &= 0.01671 \cdot sin(g) \cdot \dot{g} + 0.00028 \cdot sin(2 \cdot g) \cdot \dot{g} \\
\dot{x} &= \dot{R} \cdot cos(L) - R \cdot sin(L) \cdot \dot{L} \\
\dot{y} &= \dot{R} \cdot cos(\epsilon) \cdot sin(L) + R \cdot cos(\epsilon) \cdot cos(L) \cdot \dot{L} \\
\dot{z} &= \dot{R} \cdot sin(\epsilon) \cdot sin(L) + R \cdot sin(\epsilon) \cdot cos(L) \cdot \dot{L}
\end{aligned}$$

The Earth's velocity around the Sun is then (converting from astronomical units per day to kilometers per second):

$$\vec{V} = -1731.4568 \cdot \begin{vmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{vmatrix}$$

α and δ are the right ascension and declination of the target, and $\vec{T}$ is a unit vector pointing toward the target:

$$\vec{T} = \begin{vmatrix} cos(\delta) \cdot cos(\alpha) \\ cos(\delta) \cdot sin(\alpha) \\ sin(\delta) \end{vmatrix}$$

The radial velocity is then $radvel = -\vec{T} \cdot \vec{V}$. This value is saved as keyword V_HELIO.

The radial velocity is applied (heliocentric correction) by modifying the wavelengths in the x1d table. The change to the wavelength is $-\lambda \cdot radvel/c$, where c is the speed of light.

5.5 Wavecal Processing

Conventional wavecals (i.e. auto or GO) are separate lamp exposures with keyword EXPTYPE = "WAVECAL". These are calibrated nearly the same as are science data. For FUV data, the A and B segments are processed independently. The first step is to determine the location of the spectrum (or spectra, if NUV) in the cross-dispersion direction. This is done from the corrtag (corrected TIME-TAG) table. Locations of events near the nominal location of the wavecal spectrum will be collapsed along the dispersion direction to get a 1-D array with the cross-dispersion profile; that array will be boxcar smoothed, and the location of the maximum in the smoothed array is taken as the location. The offsets from nominal for segments A and B (if FUV) or stripes A, B and C (if NUV) are recorded as the values of keywords SHIFT2A, SHIFT2B, and (if NUV) SHIFT2C. The sign of a SHIFT2 keyword will be positive if the spectrum was found at a larger pixel number than the nominal location.

To determine the offset in the dispersion direction, the 1-D extracted wavecal lamp spectrum will be compared to a template wavecal (LAMPTAB) taken with the same grating, central wavelength, and possibly FP-POS offset. The shift of the wavecal spectrum will be determined by minimizing chi square. For NUV data, a "global" shift will first be found by adding all three spectral stripes together, adding the template spectra together, and finding the offset between those summed spectra. Then the shift of each stripe will be found independently by comparing with its template. If the shift of any stripe differs too much (currently 15 pixels) from the global shift, the shift of that stripe will be flagged as not found. Other tests on the validity of the shift are that the total number of counts in a wavecal spectrum be at least 50, and that chi square divided by the number of degrees of freedom be no larger than 6 and no smaller than 1/6. For NUV data, if some stripes are found but others are not, the shifts of the missing stripes will be assigned by using an average of the shifts of the found stripes, including a rough estimate of the relative shifts between stripes. More accurate values for the relative shifts between stripes will be incorporated when such data become available.

The offset in the dispersion direction will be recorded in keywords SHIFT1A and SHIFT1B, and also SHIFT1C if NUV. The sign of a SHIFT1 keyword will be positive if a feature in the observed wavecal is at larger pixel number than the corresponding feature in the template wavecal.

If a science observation is bracketed by auto or GO wavecal exposures, the SHIFT1[a-c] and SHIFT2[a-c] values from the wavecals will be linearly interpolated at the times of the events (TIME-TAG) or the middle time (ACCUM) of the science observation, and the interpolated values will be assigned to the SHIFT1[a-c] and SHIFT2[a-c] keywords in the science data header. If there is just one wavecal observation in a dataset, or if there is more than one but they don't bracket the science observation, the SHIFT1[a-c] and SHIFT2[a-c] keywords will just be copied from the nearest wavecal to the science data header.

For a science exposure (but not for an auto/GO wavecal), these shifts (in both axes) will be subtracted from the pixel coordinates, and the corrected coordinates will be saved in the XFULL and YFULL columns. The flt and counts images are then created by binning the XFULL and YFULL columns. For science data, the 1-D spectrum or spectra will be extracted from the flt and counts images. There's a problem here! If the offset in the dispersion direction has a fractional part that happens to be constant (e.g. only one wavecal), then the information about this systematic offset would be lost when the pixel coordinates are rounded off when binning to images. This information is therefore saved in DPIXEL1[a-c] keywords, which are used later when computing wavelengths for the 1-D extracted spectra. DPIXEL1[a-c] is obtained by taking the difference between XFULL and the nearest integer to XFULL, then taking the average of those differences. When finding the difference between XFULL and the nearest integer to XFULL, only events within the active area (if FUV) or over the region for the PSA or WCA for a given stripe (if NUV) will be included.

A tagflash wavecal is a lamp exposure that is taken concurrently with a TIME-TAG science exposure, and the photon events for both the wavecal lamp and science target are mixed together in the same events table. In some respects, tagflash wavecals are handled differently from auto and GO wavecals.

The nominal start and stop times for each lamp flash (concurrent wavecal exposure) are read from keywords in the corrtag table. The actual start and stop times could differ from the nominal times, so CALCOS will determine the actual times (restricted to being

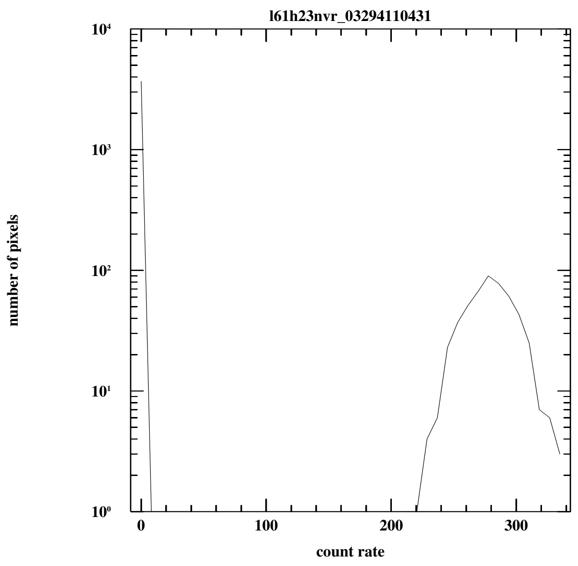

Figure 1: Lamp histogram for l61h23nvr_03294110431

within the nominal start-to-stop intervals) by examining the count rates in the wavecal region. The count rate is taken as the number of photon events within each one-second interval, but considering only events that have a pixel position that is within a region on the detector where the wavecal spectrum should be found. A histogram of the count rate is constructed. The histogram is expected to have two peaks, one near zero corresponding to dark counts (and many pixels may actually have zero counts), and another at high count rate due to the lamp illumination. The average count rate when the lamp is on is taken to be the count rate for that second peak of the histogram. The lamp turn-on and turn-off times are then taken to be the times when the count rate rises above or falls below half the lamp-on count rate. Figure 1 shows a plot of such a histogram as an example.

A lamp "flash" actually lasts a finite length of time (e.g. 10 seconds); therefore, some time within the lamp-on interval must be chosen to represent the moment of the flash. The lamp brightness will likely vary during a flash, so rather than using the midpoint of the lamp-on interval, CALCOS uses the time of the median photon event within that interval. Keywords LMP_ONi and LMPOFFi (i is the one-indexed flash number) will be updated with the actual lamp turn-on and turn-off times, in seconds since the beginning of the science exposure. Keywords LMPDURi and LMPMEDi will be updated with the actual duration and median time of the flash.

The location of the wavecal spectrum will first be measured in the cross-dispersion direction by collapsing the spectrum along the dispersion direction, resulting in a cross-dispersion profile, as is done for auto and GO wavecals. If the spectrum cannot be found, a comment to that effect will be printed to the standard output. If the spectrum was found, the processing to find the shifts in the dispersion direction is nearly the same as for auto and GO wavecals, except that the wavecal spectra will be extracted from the XDOPP and YCORR (or RAWY if NUV) columns of the corrtag table.

Depending on the length of the exposure and the time since the OSM was last moved, there will often be more than one wavecal flash during a science exposure, and in that case the shifts will be linearly interpolated between flashes. After the last flash, the shift will be extrapolated with zero slope.

5.6 1-D Spectral Extraction

The 1-D spectrum is extracted from the counts and flt images. Both files are required, based on the algorithm specified in AV-03. The extraction region is a parallelogram, with the shorter edges at the image boundaries and the longer (sloping) edges parallel to the spectrum. For each pixel in the dispersion direction, the values within the parallelogram will be added together in the cross-dispersion direction to get one element of the spectrum. (Note that in some cases the spectral lines are obviously tilted, but the sum is done without regard to any tilt.) The location of the parallelogram in the cross-dispersion direction is taken from column B_SPEC in the XTRACTAB. For wavecal exposures (auto or GO, not tagflash), the offset (keyword SHIFT2A, SHIFT2B or SHIFT2C) will be added to the nominal extraction and background locations; the exposure is taken to be an auto or GO wavecal if the EXPTYPE keyword is WAVECAL. For wavecals, the pixel coordinates themselves will not be corrected for the wavecal offsets, but the wavelengths in the x1d table will be recomputed after wavecal processing to account for the offset (SHIFT1A, etc.) in the dispersion direction. For science exposures, the pixel coordinates should already have been corrected for the wavecal offsets in both axes; this is the case both for tagflash exposures and for exposures with associated auto or GO wavecals. There are two background regions, one on either side of the spectrum, and they are extracted in a similar manner. The values in the two background regions will be added together, boxcar smoothed in the dispersion direction, and scaled by the sizes of the extraction regions before being subtracted from the science spectrum.

Here is a table showing the relevant program variables for 1-D spectral extraction and what they mean. Variables beginning with a capital letter are saved in the output x1d table. The "_i" means array element i in the dispersion direction.

variable	*description*
e_i	effective count rate, extracted from _flt.fits
GC_i	gross count rate, extracted from _counts.fits
BK_i	smoothed background count rate, extracted from _counts.fits
eps_i	effective count rate / gross count rate
N_i	net count rate = eps_i · (GC_i - BK_i)
ERR_i	error estimate for net count rate
DQ_i	data quality flags, bitwise OR of input DQ array
DQ_WGT_i	data quality weight array
snr_ff	the value of keyword SNR_FF from the flat field ref image
extr_height	the number of pixels in the cross-dispersion direction that are added together for each pixel of the spectrum
bkg_height1	the number of pixels in the cross-dispersion direction in the first background region
bkg_height2	the number of pixels in the cross-dispersion direction in the second background region
bkg_smooth	the number of pixels in the dispersion direction for boxcar smoothing the background data
bkg_norm	float (extr_height) / (2. · float (bkg_extr_height))

The error array is then calculated as follows:
term1_i = (N_i · exptime / (extr_height · snr_ff))^2
term2_i = eps_i^2 · exptime · (GC_i + BK_i · (bkg_norm / bkg_smooth))
ERR_i = $\sqrt{(\text{term1\_i} + \text{term2\_i})}/\text{exptime}$

The wavelength at each pixel of the 1-D spectrum is computed from the dispersion relation, read from the DISPTAB. For each grating and central wavelength, DISPTAB contains coefficients of a polynomial that gives the wavelength in Ångstroms as a function of pixel number. For ACCUM data the pixel numbers will be shifted (prior to evaluating the dispersion relation) by subtracting the value of the keyword written during wavecal processing (SHIFT1A, SHIFT1B, or SHIFT1C). The heliocentric correction (HELCORR) will be applied to the wavelengths by subtracting $\lambda \cdot V\_HELIO/c$, where V_HELIO is the radial velocity (computed earlier) due to the Earth's orbital motion around the Sun, and c is the speed of light.

The flux correction (FLUXCORR) is applied by dividing the NET and ERROR columns by the sensitivity read from the FLUXTAB. The NET divided by the sensitivity is written to the FLUX column, while the ERROR column is modified in-place. The sensitivity array will be interpolated to the wavelength scale of the observed spectrum. The sensitivity may vary with time, and this correction (using the TDSTAB) is applied as part of FLUXCORR. The sensitivity is described in the TDSTAB as a (not necessarily continuous) piecewise-linear function of time; there is one such function at each of several (e.g. 60) different wavelengths. The intercept at a reference time and the slope are given for each interval. For times of observation that are outside the range of times listed in the TDSTAB, the correction is extrapolated toward earlier or later times using a slope (with time) of zero.

6 Columns in Output Tables

This section describes the columns in the output tables. The corrtag table contains the calibrated events table, which is based on the input rawtag table. The x1d table contains the 1-D extracted spectra. The lampflash table is only written when the input was taken in tagflash mode; this table contains the 1-D extracted wavecals.

6.1 Columns in the Corrtag Table

The corrtag table contains the calibrated photon events list for TIME-TAG data, or a pseudo-events list for ACCUM data. The values in the PHA column will be zero for NUV data.

column	*description*
TIME	time of event in seconds after EXPSTART (float)
RAWX	X (column) pixel number of event in input table (short)
RAWY	Y (row) pixel number of event in input table (short)
XCORR	X (column) pixel number corrected for distortion (float)
YCORR	Y (row) pixel number corrected for distortion (float)
XDOPP	X pixel number corrected for Doppler shift and distortion (float)
XFULL	X pixel number corrected for offset in the dispersion direction (based on the wavecal spectrum) and distortion and Doppler shift (float)
YFULL	Y pixel number corrected for offset in the cross-dispersion direction (based on the location of the wavecal spectrum) and distortion (float)
EPSILON	weight for event, based on flat field and deadtime (float)
DQ	data quality flag (short)
PHA	pulse height amplitude (byte)

6.2 Columns in the x1d Table

The x1d table contains the 1-D extracted spectra. For a single exposure, there will normally be two spectra for FUV (one for each segment) and three for NUV (one for each spectral stripe).

column	*description*
SEGMENT	segment (or NUV stripe) name (string)
EXPTIME	exposure time in seconds, corrected for gaps (double)
NELEM	all the following columns are arrays of this length (int)
WAVELENGTH	array of wavelengths in Angstroms (double)
FLUX	array of fluxes in erg/s/cm^2/angstrom (float)
ERROR	error estimates for fluxes in erg/s/cm^2/angstrom (float)
GROSS	count rate in counts/s (float)
NET	rate in counts/s, background subtracted and corrected for flat field and deadtime (float)
BACKGROUND	background count rate in counts/s (float)
DQ	data quality flags, bitwise OR of input DQ within the extraction region (short)
DQ_WGT	weight (0 or 1) to use when averaging x1d tables (float)

An element in DQ_WGT will be 0 if any pixel in the extraction region is flagged as dead, hot, or out of bounds, and it will be 1 otherwise. DQ_WGT is used as a weight when combining FP-POS data.

6.3 Columns in the Lampflash Table

For tagflash data, a rootname_lampflash.fits table will be written that contains the extracted wavecal spectrum for each flash and for each FUV segment or NUV stripe.

column	*description*
SEGMENT	segment (or NUV stripe) name (string)
TIME	median time (seconds) of current flash (double)
EXPTIME	duration (seconds) of flash (double)
LAMP_ON	time (seconds) when the lamp turned on (double)
LAMP_OFF	time (seconds) when the lamp turned off (double)
NELEM	length of the wavelength and gross arrays (int)
WAVELENGTH	array of wavelengths in Angstroms (double)
GROSS	array of count rate (counts/s) (float)
SHIFT_DISP	shift in pixels in the dispersion direction (float)
SHIFT_XDISP	shift in pixels in the cross-dispersion direction (float)
SPEC_FOUND	true if the spectrum was actually found (boolean)
CHI_SQUARE	a measure of how well the spectrum matched the template (float)
N_DEG_FREEDOM	number of degrees of freedom for chi square (int)

Calcos reads the nominal lamp turn-on and turn-off times from keywords LMP_ONi and LMPOFFi, but it then determines the actual on and off times (with a resolution of about one second) from the count rate in the wavecal region of the detector. These actual times in seconds are recorded in columns LAMP_ON and LAMP_OFF; the zero point is the overall exposure start time, keyword EXPSTART. The value in the TIME column is the time of the median photon event within the interval LAMP_ON to LAMP_OFF, also in seconds since EXPSTART. EXPTIME is redundant because it is equal to LAMP_OFF - LAMP_ON, but it is provided for convenience. The spectrum itself is saved in the GROSS column (an array in each row); note that this is the gross count rate, i.e. uncorrected for flat field or deadtime.