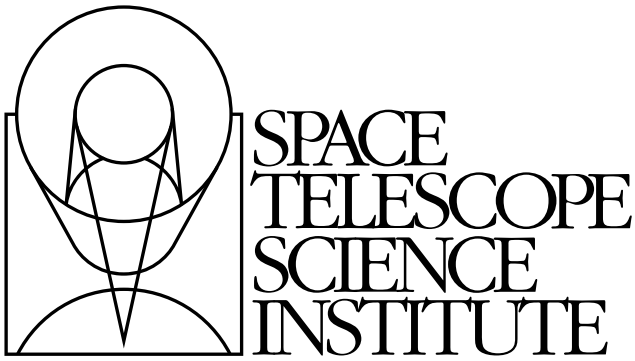

wfpc2tools Documentation

Release 1.0.0

STScI

May 03, 2013

CONTENTS

This package contains various functions for analyzing and processing HST/WFPC2 images.

Modules:

CHAPTER

ONE

WFPC2CTE

WFPC2CTE - Module for computing the CTE degradation for WFPC2 images

This module updates the header of the input WFPC2 image with standardized computations of the effect of CTE based on the algorithm published by Dolphin (2004, http://purcell.as.arizona.edu/wfpc2_calib/2004_12_20.html).

ASSUMPTIONS for the COMPUTATION

1. The CTE gets computed for a source at the chip center.
2. **The background (in electrons) gets defined by the clipped mode of the** central 200x200 pixels from the image.
3. **The source is assumed to have 100 electrons, 1000 electrons and** 10000 electrons in the aperture.
4. **The reported CTE is the sum of the XCTE and YCTE computed from** Dolphin's algorithm.

INPUT

The sole input for this task is the filename of the WFPC2 image.

If the input image is in GEIS format, it will convert it to a multi-extension FITS formatted file, then update the FITS file while leaving the GEIS image un-modified.

If the input image is already multi-extension FITS, it will update the header directly.

If the input image is waivered FITS, it will quit with a message telling the user to first convert the file to GEIS. The user can then provide the GEIS image as input.

OUTPUT

The keywords which get updated are:

```
CTE_1E2  - CTE for a source with an intensity of 100 electrons
CTE_1E3  - CTE for a source with an intensity of 1000 electrons
CTE_1E4  - CTE for a source with an intensity of 10000 electrons
```

SYNTAX

This task can be run on an input WFPC2 image using either of the following calls:

```
wfpc2cte.compute_CTE(filename,quiet=True)
-or-
wfpc2cte.run(filename,quiet=True)
```

where the filename is the name of the input WFPC2 image.

EXAMPLE

The syntax for running this task on a WFPC2 file named 'u40x0102m.c0h':

```
import wfpc2cte
wfpc2cte.run('u40x0102m.c0h')
```

The command to print out this help file:

```
wfpc2cte.help()
```

FUNCTIONS

wfpc2tools.wfpc2cte.**compute_CTE**(*filename*, *quiet=True*, *nclip=3*, *update=True*)

Compute the CTE correction for a 100, 1000 and 1e+4 DN source in a WFPC2 chip. These correction values will be written to the WFPC2 image header as the CTE_1E2, CTE_1E3 and CTE_1E4 keywords respectively.

Parameters

filename : str

Name of WFPC2 image

quiet : bool, optional [Default: True]

Specifies whether or not to print verbose messages during processing

nclip : int [Default: 3]

Number of clipping iterations for computing the chip's pixel values

update : bool [Default: True]

Specifies whether or not to update the input image header with the computed CTE correction values

wfpc2tools.wfpc2cte.**compute_XCTE**(*xpos*, *bg*)

wfpc2tools.wfpc2cte.**compute_YCTE**(*chip_values*, *yr*, *xcte*)

wfpc2tools.wfpc2cte.**compute_chip_values**(*extn*, *gain*, *nclip=3*)

wfpc2tools.wfpc2cte.**help**()

wfpc2tools.wfpc2cte.**run**(*filename*, *quiet=True*, *nclip=3*)

wfpc2tools.wfpc2cte.**update_CTE_keywords**(*hdr*, *cte*, *quiet=False*, *update=True*)

CHAPTER

TWO

WFPC2DESTREAK

This module implements the destreak correction for WFPC2 images. WFPC2Destreak - Module for performing destreak correction on WFPC2 images

Outline

1. **In the 'interior' image region (starting to the right of the pyramid region), eliminate the CRs, and**

   calculate the mean (im_mean) and sigma (im_sigma):

   - Over the entire c0 image, cosmic rays are identified and masked in the c0 data
   - For the entire image, the global mean and the (clipped) sigma is calculated for all unmasked pixels
   - For each row, the mean is calculated for all unmasked pixels
   - For each row, the difference between the mean and the global mean is subtracted from the c0 data
2. The modified c0 data is written to the file <dataset>_bjc_<chip>.fits ('bjc' stands for 'bias jump corrected')

Command Line Options

Linux command line short options and defaults (set in wfpc2util.py):

```
-g: group (default = 4)
-b: bias_thresh (default = 100000.)
-r: row_thresh (default = 0.1)
-v: verbosity (default = verbose)
-m: input_mask (default = None)
-i: niter (default = 5)
```

Examples

1. For a dataset with multiple groups, to process group 4 using a bias threshold=280 and row threshold=0.2:

   ```
   hal> ./wfpc2destreak.py "u96r0603m_c0h.fits"  -g 4 -b 280. -r 0.2  -v
   ```

   This can also be specified using the 'long options':

   ```
   hal> ./wfpc2destreak.py "u96r0603m_c0h.fits" --group=4 --bias_thresh=280. --row_thresh=0.2
   ```
2. To allow the routine to run with all of the defaults:

   ```
   hal> ./wfpc2destreak.py "u96r0603m_c0h.fits"
   ```
3. For a dataset with a single group, using defaults for the thresholds:

```
hal> ./wfpc2destreak.py "u96r0603m_c0h.fits"  -g 0
```

4. Same as example F, but specifing an input mask to use:

```
hal> ./wfpc2destreak.py "u96r0603m_c0h.fits"  -g 0  -m "mask_u8zq0104.fits"
```

5. Same as example F, but specifing 3 iterations for the CR rejection

```
hal> ./wfpc2destreak.py "u96r0603m_c0h.fits"  -g 0  -i 3
```

6. Run the routine for group 3 of a geis image

```
hal> ./wfpc2destreak.py "ub080106m.c0h" -g 3
```

Example 'A' under pyraf:

```
--> wfp = wfpc2destreak.Wfpc2destreak( "u96r0603m_c0h.fits", group=4, bias_thresh=280, row_thresh=0.2
--> wfp.destreak()
```

Example 'A' under stsdas (after loading hst_calib and wfpc):

```
--> from wfpc2tools import wfpc2destreak
--> wfp = wfpc2destreak.Wfpc2destreak( "u96r0603m_c0h.fits", group=4, bias_thresh=280, row_thresh=0.2
--> wfp.destreak()
```

or

```
--> import wfpc2tools
--> wfp = wfpc2tools.wfpc2destreak.Wfpc2destreak( "u96r0603m_c0h.fits")
--> wfp.destreak()
```

Functions

class wfpc2tools.wfpc2destreak.**Wfpc2destreak**(*input_file*, *input_mask=None*, *group=None*, *verbosity=0*, *bias_thresh=None*, *row_thresh=None*, *niter=None*)

Calculate magnitude of and remove streaks from specified group of wfpc2 data.

Parameters

input_file : str

name of the c0h file to be processed

input_mask : str

name of the input mask

group : int

number of group to process

verbosity : str

verbosity level (0 for quiet, 1 verbose, 2 very verbose)

bias_thresh : float

bias threshold (no correction will be performed if this is exceeded by im_mean)

row_thresh : float

row threshold (no correction will be performed if this exceeds the calculated row correction)

niter : int

number of iterations for CR rejection

Examples

```
>>> wfpc2_d = wfpc2destreak.Wfpc2destreak( filename, input_mask=input_mask, group=group, verbosi
       bias_thresh=bias_thresh, row_thresh=row_thresh,  niter=niter)
>>> wfpc2destreak.Wfpc2destreak.destreak(wfpc2_d)
```

destreak()
Method to perform destreak correction.

print_pars()
Print parameters method.

wfpc2tools.wfpc2destreak.**check_cl_pars**(*input_file*, *group*, *bias_thresh*, *row_thresh*, *input_mask*, *niter*)
When run from linux command line, verify that each parameter is valid.

Parameters

input_file : str

name of input file

group : int

number of group to process

bias_thresh : float

bias threshold (no correction will be performed if this is exceeded by im_mean)

row_thresh : float

row threshold (no correction will be performed if this exceeds the calculated row correction)

input_mask : str

name of input mask file

niter : int

number of CR rejection iterations

Returns

group : int

row_thresh : float

niter : int

wfpc2tools.wfpc2destreak.**check_neighbors**(*new_cr*, *residual*, *cutoff*, *sub_shape*)
Check for cosmic rays in neighboring pixels.

Parameters

new_cr : ndarray

1-D array of (int) ones or zeros (1 indicates a cosmic ray)

residual : ndarray

1-D array of residuals (float64), subarray - fit

cutoff : float

criterion for flagging an outlier as a cosmic ray

sub_shape : tuple

numbers of lines and columns in subarray

Returns

new_cr : int

pixel position associated with identified cosmic ray,
possibly with additional cosmic rays flagged

wfpc2tools.wfpc2destreak.**check_py_pars**(*input_file*, *group*, *bias_thresh*, *row_thresh*, *input_mask*, *niter*)

When run under python, verify that each unspecified parameter should take the default value, and give the user the opportunity to change it.

Parameters

input_file : str

name of input file

group : int

number of group to process

bias_thresh : float

bias threshold (no correction will be performed if this is exceeded by im_mean)

row_thresh : float

row threshold (no correction will be performed if this exceeds the calculated row correction)

input_mask : str

name of input mask

niter : int

number of CR rejection iterations

Returns

group : int

bias_thresh : float

row_thresh : float

wfpc2tools.wfpc2destreak.**cr_reject**(*SubArray*, *niter*)

Identify and replace cosmic rays in the given subarray.

Parameters

SubArray : ndarray

subarray of the data

niter : int

the number of iterations used when rejecting cosmic rays

wfpc2tools.wfpc2destreak.**fitline**(*x*, *y*, *mask*)

Fit a straight line to y vs x, where mask is 0.

Parameters
x : ndarray

float64 array of independent-variable values

y : ndarray

float64 array of dependent-variable values

mask : ndarray

int32 array of ones or zeros (0 indicates a good value)

Returns
coeffs : tuple

coefficients of fit: tuple of the slope and intercept

wfpc2tools.wfpc2destreak.**median**(*y*, *mask*)
Return the median of the array y, ignoring masked elements.

Parameters
y : ndarray

array of values

mask : ndarray

array of (int32) ones or zeros (0 indicates a good value)

Returns
median : float

median of y, ignoring masked elements

wfpc2tools.wfpc2destreak.**update_header**(*self*, *hdr*)
update header from input c0 file with specified header, and updated data

Parameters
self : object

Wfpc2destreak object containing results to be recorded to header

hdr : object

PyFITS header object

wfpc2tools.wfpc2destreak.**write_mask**(*data*, *filename*)
write specified mask

Parameters
data : ndarray

mask array

filename : string

mask file name

wfpc2tools.wfpc2destreak.**write_to_file**(*data*, *filename*, *hdr*, *verbosity*, *im_mean*, *im_sigma*)
Write mean and sigma to file.

Parameters
data : ndarray

array of floats

filename : string

mask file name

hdr: object :

Pyfits header object

verbosity: int :

verbosity level (0 for quiet, 1 verbose, 2 very verbose)

im_mean : float

clipped mean of image region

im_sigma : float

clipped sigma of image region

CHAPTER

THREE

WFPC2UTIL

This module provides basic utilities required by this package for processing WFPC2 images.

wfpc2tools.wfpc2util.**all_printMsg**(*message*, *level=1*)
Print message as verbose message by default

Parameters
message : string

message be printed, if verbosity level is appropriate

level: int [Default: 1 (VERBOSE)] :

integer indicating the level of verbosity for printing this string

wfpc2tools.wfpc2util.**checkVerbosity**(*level*)
Return true if verbosity is at least as great as level.

Parameters
level: int :

level of verbosity to be checked against global value

wfpc2tools.wfpc2util.**printMsg**(*message*, *level=0*)
Print message based on verbosity level.

Parameters
message : string

message be printed, if verbosity level is appropriate

level: int [Default: 0 (QUIET)] :

integer indicating the level of verbosity for printing this string

wfpc2tools.wfpc2util.**setBias_thresh**(*bias_thresh_value*)
Copy bias_thresh to a variable that is global for this file.

Parameters
bias_thresh_value : float

value of bias_thresh

wfpc2tools.wfpc2util.**setGroup**(*group_value*)
Copy group to a variable that is global for this file.

Parameters
group_value : int

value of group

wfpc2tools.wfpc2util.**setInput_mask**(*input_mask_value*)
Copy input_mask to a variable that is global for this file.

Parameters
input_mask_value : string

value of input_mask

wfpc2tools.wfpc2util.**setNiter**(*niter_value*)
Copy niter to a variable that is global for this file.

Parameters
niter_value : int

value of niter

wfpc2tools.wfpc2util.**setRow_thresh**(*row_thresh_value*)
Copy row_thresh to a variable that is global for this file.

Parameters
row_thresh_value : float

value of row_thresh

wfpc2tools.wfpc2util.**setVerbosity**(*verbosity_level*)
Copy verbosity to a variable that is global for this file.

Parameters
verbosity_level: int :

an integer value indicating the level of verbosity

Note: the above only represents the primary user interface functions for this package

CHAPTER

FOUR

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

W

W