CONTENTS

1 Making IMPHTTAB Tables 3
   1.1 Examples ........................................... 3

2 Compare IMPHTTAB Tables 7

3 Compare Synphot and Pysynphot 11

4 Calculate Photometry Keywords from IMPHTTAB 15

5 Make PCTETAB Reference File 17
   5.1 Primary Header Parameters .......................... 17
   5.2 Table Extensions .................................... 17

6 Indices and tables 21

Python Module Index 23

Python Module Index 25

Index 27
MAKING IMPHTTAB TABLES

Making IMPHTTAB tables requires a connection to CDBS and that the PYSYN_CDBS environmental keyword point to CDBS.

Use `reftools.mkimphttab.create_table()` and provide the name of your output file and the base observation mode from which to build a set observation modes for which photometry keywords will be calculated.

Examples of a base observation mode include "acs,hrc", "wfc3,uvis1", or "cos".

1.1 Examples

```python
>>> from reftools import mkimphttab
>>> mkimphttab.create_table('acs_wfc1', 'acs,wfc1', 'WFC1', ...
...   'Mar 01 2002 00:00:00', clobber=True, verbose=False)
... mkimphttab.create_table('acs_sbc', 'acs,sbc', 'SBC', ...
...   'Mar 01 2002 00:00:00', clobber=True)
```

reftools.mkimphttab.\texttt{create\_table}(\texttt{output}, \texttt{basemode}, \texttt{detector}, \texttt{useafter}, \texttt{tmgtab=None}, \texttt{tmctab=None}, \texttt{tmttab=None}, \texttt{mode\_list=\texttt{[]}}, \texttt{nmodes=None}, \texttt{clobber=False}, \texttt{verbose=True})

Create an IMPHTTAB reference file for a specified base configuration, \texttt{basemode}.

Parameters

- **output** : str
  - Output IMPHTTAB filename. (_imp.fits will be appended if only prefix is given.)

- **basemode** : str
  - Base observation mode for which to generate IMPHTTAB (e.g., \texttt{acs,hrc}). This is ignored if \texttt{mode\_list} is given.

- **detector** : str
  - Detector name.

- **useafter** : str
  - Useafter date in the format of MMM DD YYYY HH:MM:SS.

- **tmgtab**, **tmctab**, **tmttab** : str, optional
  - Graph (TMG), component (TMC), and thermal component (TMT) tables to use. If \texttt{None}, the most recent version in CDBS is used.

- **mode\_list** : list of str, optional
A list of observation modes which should be used to make an IMPHTTAB reference file. If given, basemode is ignored.

**nmodes**: int, optional
Set to limit the number of modes to calculate. This is for testing only, otherwise set to None.

**clobber**: bool, optional
Overwrite existing IMPHTTAB?

**verbose**: bool, optional
Display extra information.

```python
def create_table_from_table(output, detector, useafter, imphttab, **kwargs):
    Use a previously created IMPHTTAB reference file to generate a new IMPHTTAB reference file.
    Parameters
    output, detector, useafter : str
        See `create_table()`.
    imphttab : str
        File name of _imp.fits IMPHTTAB table from which to take observation modes.
    kwargs : dict
        Keywords accepted by `create_table()`, except `mode_list`.
```

```python
def create_nicmos_table(output, detector, useafter, pht_table, **kwargs):
    Use a NICMOS _pht.fits table to generate an IMPHTTAB table for observation modes listed in the given table.
    Parameters
    output, detector, useafter : str
        See `create_table()`.
    pht_table : str
        File name of _pht.fits table from which to take observation modes.
    kwargs : dict
        Keywords accepted by `create_table()`, except `mode_list`.
```

```python
def compute_values(obsmode, component_dict):
    Compute the 3 photometric values needed for a given observation mode string using pysynphot.
    Values calculated:
    • PHOTFLAM - Unit response in FLAM.
    • PHOTPLAM - Pivot wavelength.
    • PHOTBW - Bandwidth.
    Parameters
    obsmode : str
        Observation mode for which to calculate values.
    component_dict : dict
```

Chapter 1. Making IMPHTTAB Tables
A dictionary in which to cache opened component objects. May be empty.

**Returns**

\[ \text{valdict} : \text{dict} \]

Dictionary with photometry keywords as keys.

`reftools.mkimphottab.compute_synphot_values(obsmode)`

Calculate the same values as `compute_values()` but using IRAF SYNPHOT.

**Note:** This is replaced by `compute_values()` but kept for debugging.

**Parameters**

\[ \text{obsmode} : \text{str} \]

Observation mode for which to calculate values.

**Returns**

\[ \text{valdict} : \text{dict} \]

Dictionary with photometry keywords as keys.
Tools for comparing two IMPHTTAB tables from the same instrument and detector.

class reftools.imphtcomp.ImphttabComp(tab1, tab2)

Class for comparing two IMPHTTAB tables from the same instrument and detector.

Parameters

- tab1 : str
  Filename of first IMPHTTAB for comparison.

- tab2 : str
  Filename of second IMPHTTAB for comparison.

Attributes

- tab1_name : str
  Filename of first IMPHTTAB.

- tab2_name : str
  Filename of second IMPHTTAB.

- modes : ndarray
  Obsmodes present in both input files.

- flam1 : ndarray
  PHOTFLAM values from tab1 for obsmodes in modes.

- plam1 : ndarray
  PHOTPLAM values from tab1 for obsmodes in modes.

- bws1 : ndarray
  PHOTBW values from tab1 for obsmodes in modes.

- flam2 : ndarray
  PHOTFLAM values from tab2 for obsmodes in modes.

- plam2 : ndarray
  PHOTPLAM values from tab2 for obsmodes in modes.

- bws2 : ndarray
  PHOTBW values from tab2 for obsmodes in modes.

- flamdiff : ndarray
  Percent differences between flam1 and flam2 calculated as (flam1 - flam2) / flam1.

- plamdiff : ndarray
  Percent differences between plam1 and plam2 calculated as (plam1 - plam2) / plam1.

- bwdiff : ndarray
  Percent differences between bws1 and bws2 calculated as (bws1 - bws2) / bws1.

make_plot (outname='imphttab_comp.pdf')

Make a plot with histograms of the percent differences between PHOTFLAM, PHOTPLAM, and PHOTBW for the IMPHTTAB tables.

Differences plotted are 100 * (table1 - table2) / table1.
Parameters

outname : str, optional

Filename of output plot, including extension. Defaults to ‘imphttab_comp.pdf’.

print_diffs (orderby='photflam', lines=25)

Print obsmodes and parameters ordered by orderby parameter, with the largest absolute differences in that parameter at the top. This is for seeing which obsmodes have the largest difference in the specified parameter. Prints the number of modes given in the lines parameter.

Differences shown are calculated as 100 * (table1 - table2)/table1.

Parameters

orderby : str, optional

The parameter by which to order the printed results, with modes having the largest absolute difference in this parameter printed at the top.

May be one of: ‘photflam’, ‘photplam’, or ‘photbw’. An ImphtcompError is raised if the input does not match one of these.

Defaults to ‘photflam’.

lines : int, optional

The number of lines to print. Defaults to 25.

Raises

ImphtcompError :

If orderby does not match a valid option.

reftools.imphtcomp.plot_table_diffs (table1, table2, outname='imphttab_comp.pdf')

Make a plot with histograms of the percent differences between PHOTFLAM, PHOTPLAM, and PHOTBW for the IMPHTTAB tables.

Differences plotted are 100 * (table1 - table2) / table1.

Parameters

table1 : str

Filename of first IMPHTTAB for comparison.

table2 : str

Filename of the second IMPHTTAB for comparison.

outname : str, optional

Filename of output plot, including extension. Defaults to ‘imphttab_comp.pdf’.

reftools.imphtcomp.print_table_diffs (table1, table2, orderby='photflam', lines=25)

Compare two IMPHTTAB tables and print their differences to the terminal. Prints any obsmodes which are in either table but not in both.

Also prints the obsmodes and parameters for the modes which most differ in the parameter given in orderby. This is for seeing which obsmodes have the largest percent difference in the specified parameter. Prints the number of modes given in the lines parameter.

Differences shown are calculated as 100 * (table1 - table2)/table1.

Parameters

table1 : str

Filename of first IMPHTTAB for comparison.
table2 : str
   Filename of the second IMPHTTAB for comparison.
orderby : str, optional
   This specifies one of ‘photflam’, ‘photplam’, ‘photbw’, or ‘all’. The printed results are
   ordered according to the absolute difference in the specified parameter, with the mode
   with the largest absolute difference at the top of the list.
   Specifying ‘all’ will print 3 tables, one ordered by each of the parameters.
   Defaults to ‘photflam’.
lines : int, optional
   Number of lines of differences to print. Defaults to 25.

Raises
   ImphtcompError :
      If orderby does not match a valid option.
Tools for comparing pysynphot and synphot photometry calculations.

```python
class reftools.synpysyncomp.SynPysynComp(imphttab)
    Compare synphot and pysynphot values for obsmodes listed in the input IMPHTTAB. Only obsmodes with three or fewer listed components are compared. For example, acs,wfc1,f555w would be processed but not acs,wfc1,f555w,MJD#.

    This can be a long process for large numbers of obsmodes so it is recommended that users use the CSV functionality of this class to save the results to a .csv file and then use the supporting functions print_synpysyn_diffs and plot_synpysyn_diffs in this module to investigate them.

    Parameters
        imphttab : str
            Filename of IMPHTTAB from which to take obsmodes for comparison.

    calculate_diffs(verbose=False)
    Calculate diffs for all obsmodes and return.

    Parameters
        verbose : bool, optional
            If True, print obsmodes as they are processed. Defaults to False.

    Returns
        res : dict
            Dictionary containing lists of all pysynphot and synphot calculated values and their differences calculated as (pysynphot - synphot)/synphot.

    comp_synpysyn(mode)
    Returns a dictionary of pysynphot and synphot values and their percent differences calculated as (pysynphot - synphot)/synphot.

    Parameters
        mode : str
            Observing mode.

    Returns
        comp : dict
            Dictionary containing calculated values.

    get_pysyn_vals(mode)
    Get comparison values from pysynphot.
```
Parameters
   mode : str
       Obsmode string.

Returns
   ret : dict
       Dictionary containing calculated values.

**get_syn_vals** *(mode)*
Get comparison values from synphot.

Parameters
   mode : str
       Obsmode string.

Returns
   ret : dict
       Dictionary containing calculated values.

**write_csv** *(outfile, verbose=False)*
Write a CSV file containing the pysynphot and synphot values for all obsmodes and their percent differences calculated as (pysynphot - synphot)/synphot.

Parameters
   outfile : str
       Name of file to write to.
   verbose : bool, optional
       If True, print obsmodes as they are processed. Defaults to False.

**class** *reftools.synpysyncomp.SynPysynPlot*(csvfile)*
Make a plot from a CSV file created by SynPsynComp.write_csv illustrating differences between synphot and pysynphot calculated products.

The plots show differences between 7 parameters calculated by both synphot and pysynphot. The differences are shown as histograms. The difference plotted is calculated as 100 * (pysynphot - synphot)/synphot.

Parameters
   csvfile : str
       Filename of input CSV file containing comparison results. Should be a file created by SynPsynComp.write_csv.

Attributes :
   fig : matplotlib.figure.Figure
       Useful for setting manipulating figure properties such as the title, which otherwise defaults to csvfile.

**save_plot** *(outname='synpysyn_comp.pdf')*
Save plots to a file.

Parameters
   outname : str, optional
       Name of file to save to, including extension. Defaults to 'synpysyn_comp.pdf'.
reftools.synpysyncomp.plot_synpysyn_diffs (csvfile, outname=`synpysyn_comp.pdf`)  
Make and save a plot illustrating differences between parameters calculated by both synphot and pysynphot.  
Data are taken from a CSV file made by SynPysynComp.write_csv.  

Differences are shown as histograms. The difference plotted is calculated as 100 * (pysynphot - synphot) / synphot.  

Parameters  

csvfile : str  
Filename of input CSV file containing comparison results. Should be a file created by SynPysynComp.write_csv.  

outname : str, optional  
Name of file to save to, including extension. Defaults to ‘synpysyn_comp.pdf’.  

reftools.synpysyncomp.print_synpysyn_diffs (csvfile, orderby=`photflam`, lines=25)  
Print synphot/pysynphot comparison results from a CSV file produced by SynPysynComp.write_csv to the terminal.  

Prints the obsmodes and parameters for the modes which most differ in the parameter given in orderby. This is for seeing which obsmodes have the largest percent difference in the specified parameter. Prints the number of modes given in the lines parameter.  

Only prints data for the parameters PHOTFLAM, PHOTPLAM (PIVOT WV), and PHOTBW (RMSWIDTH). Differences are given as 100 * (pysynphot - synphot) / synphot.  

Parameters  

csvfile : str  
Filename of input CSV file containing comparison results. Should be a file created by SynPysynComp.write_csv.  

orderby : str, optional  
This specifies one of ‘photflam’, ‘photplam’, ‘photbw’, or ‘all’. The printed results are ordered according to the absolute difference in the specified parameter, with the mode with the largest absolute difference at the top of the list.  
Specifying ‘all’ will print 3 tables, one ordered by each of the parameters.  
Defaults to ‘photflam’.  

lines : int, optional  
Number of lines of differences to print. Defaults to 25.  

Raises  

SynPysynCompError :  
If orderby does not match a valid option.  

reftools.synpysyncomp.read_synpysyn (csvfile)  
Read a CSV file created by SynPysynComp.write_csv into numpy arrays.  

Parameters  

csvfile : str  
Filename of CSV to read. Should be a file created by SynPysynComp.write_csv.  

Returns  

res : dict  
Contains one field for each column of the CSV file with a numpy array in that field.
getphotpars includes utilities for calculating the photometry keywords PHOTZPT, PHOTFLAM, PHOTPLAM, and PHOTBW for a given obsmode and IMPHTTAB. The calculations are performed in the same way here as they are in hstcal pipelines.

To calculate a single set of keywords use the function `get_phot_pars`.

If you are calculating for several obsmodes from a single IMPHTTAB file it’s best to use the `GetPhotPars` class. For example:

```python
get_phot = GetPhotPars(imphttab)
for obs in obsmodes:
    photzpt, photflam, photplam, photbw = get_phot.get_phot_pars(obs)
... get_phot.close()
```

**exception reftools.getphotpars.ImphttabError**  
Class for errors associated with the imphttab file.

**class reftools.getphotpars.GetPhotPars(imphttab)**  
This object can be used to get photometry parameters from a given IMPHTTAB reference file. Initialize with the name of an IMPHTTAB reference file and then call this class or the get_phot_pars method with a complete obsmode to get the photometry parameters.

**Example obsmodes are:**  
`'acs,wfc1,f625w,f660n' 'acs,wfc1,f625w,f814w,MJD#55000.0' 'acs,wfc1,f625w,fr505n#5000.0,MJD#55000.0'`

**Parameters**  

<table>
<thead>
<tr>
<th>imphttab</th>
<th>str</th>
</tr>
</thead>
</table>

Filename and path of IMPHTTAB reference file.

**Attributes**

<table>
<thead>
<tr>
<th>imphttab_fits</th>
<th>pyfits.HDUList</th>
</tr>
</thead>
</table>

Open pyfits.HDUList object from imphttab.

**close()**  
Close imphttab_fits attribute.

**get_phot_pars(obsmode)**  
Return PHOTZPT, PHOTFLAM, PHOTPLAM, and PHOTBW for specified obsmode.
Parameters

    obsmode : str
    obsmode string

Returns

    photzpt : float
    PHOTZPT from imphttab_fits header.

    photflam : float
    Interpolated PHOTFLAM for obsmode.

    photplam : float
    Interpolated PHOTPLAM for obsmode.

    photbw : float
    Interpolated PHOTBW for obsmode.

reftools.getphotpars.get_phot_pars(obsmode, imphttab)
Return PHOTZPT, PHOTFLAM, PHOTPLAM, and PHOTBW for specified obsmode and imphttab.

Parameters

    obsmode : str
    Complete obsmode string including any parameterized values.

Example obsmodes are:
    'acs,wfc1,f625w,f660n'           'acs,wfc1,f625w,f814w,MJD#55000.0'
    'acs,wfc1,f625w,fr505n#5000.0,MJD#55000.0'

    imphttab : str
    Path and filename of IMPHTTAB reference file.

Returns

    photzpt : float
    PHOTZPT from IMPHTTAB header.

    photflam : float
    Interpolated PHOTFLAM for obsmode.

    photplam : float
    Interpolated PHOTPLAM for obsmode.

    photbw : float
    Interpolated PHOTBW for obsmode.
MAKE PCTETAB REFERENCE FILE

The PCTETAB reference file contains data in both its primary header and in several table extensions. The parameters that go into the primary header are single numbers that must be specified in the call to MakePCTETab.

The data that goes into the table extensions is kept in text files. The names of these text files are given to MakePCTETab, which reads them to populate the table extensions. See the documentation for MakePCTETab for more detailed descriptions, argument names, and default values.

5.1 Primary Header Parameters

- Number of times the readout is simulated to arrive at the corrected image.
- Number of times the pixels are shifted per readout simulation.
- The read noise, in electrons, of the image. This is technically different for each amp but here we pick a single representative value.
- The default value selecting a model for how read noise is handled before CTE correction.
- Threshold for re-correcting over-subtracted pixels. Sometimes the CTE correction removes too much flux from a trail leaving a large divot.

5.2 Table Extensions

Functions for ACS PCTETAB reference file.

Authors
  Pey Lian Lim, Matt Davis

Organization
  Space Telescope Science Institute

History

- 2010-08-31 PLL created this module.
- 2010-11-09 PLL added RN2_NIT keyword and updated documentation.
- 2011-04-25 MRD updated for new CTE algorithm parameters
- 2011-07-18 MRD updated to handle time dependence
- 2011-11-29 MRD updated with column-by-column CTE scaling
• 2013-08-13 PLL removed depreciated PyFITS calls and cleaned up codes.

5.2.1 Examples

```python
>>> from reftools import pctetab
>>> pctetab.MakePCTETab(...
...    'pctetab_pcte.fits', 'pctetab_dtdel.txt',
...    ['pctetab_chgleak-1.txt', 'pctetab_chgleak-2.txt'],
...    'pctetab_levels.txt', 'pctetab_scaling.txt',
...    'pctetab_column_scaling.txt', history_file='pctetab_history.txt')
```

exception reftools.pctetab.PCTEFileError

Generic exception for errors in this module.

reftools.pctetab.MakePCTETab("out_name, dtdel_file, chg_leak_file, levels_file, scale_file, column_file, sim_nit=7, shft_nit=7, read_noise=5.0, noise_model=1, oversub_thresh=-10, useafter='Mar 01 2002 00:00:00', pedigree='INFLIGHT 01/03/2002 22/07/2010', creatorName='ACS Team', history_file='', detector='WFC')

Make the CTE parameters reference file.

Parameters

- `out_name` : str
  Name of pctime fits file being created. May include path.

- `dtdel_file` : str
  Path to text file containing dtdel data.
  The file should have 2 columns with the following format:

  ```
  DTDE  Q
  float  int
  ...
  ...
  ```
  Lines beginning with # are ignored.

- `chg_leak_file` : str or list of str
  Path to text file(s) containing charge leak data. If passed as a string the string may contain wild cards so that multiple files are specified.
  The input file should contain 5 columns with following format:

  ```
  NODE LOG_Q_1 LOG_Q_2 LOG_Q_3 LOG_Q_4
  int  float  float  float  float
  ...
  ...
  ```
  Lines beginning with # are ignored.

- `levels_file` : str
  Text file containing charge levels at which to do CTE evaluation.
  The input file should have a single column with the following format:

  ```
  LEVELS
  int
  ...
  ```
Lines beginning with # are ignored.

**scale_file**: str

Text file containing CTE scaling parameters

The input file should have two columns with the following format:

```
MJD     SCALE
float   float
...     ...
```

Lines beginning with # are ignored.

**column_file**: str

Text file containing CTE column-by-column scaling.

The input file should have 5 columns with the following format:

```
COLUMN AMPA AMPB AMPC AMPD
int    float float float float
...    ...    ...    ...    ...
```

Lines beginning with # are ignored.

**sim_nit**: int, optional

Number of iterations of readout simulation per column.

**shft_nit**: int, optional

Number of shifts each readout simulation is broken up into. A large number means pixels are shifted a smaller number of rows before the CTE is evaluated again.

**read_noise**: float

Value for RN_CLIP keyword in PCTEFILE EXT 0. This is the maximum amplitude of read noise used in the read noise mitigation. Unit is in electrons.

**noise_model**: {0, 1, 2}

Select the method to be used for readnoise removal:

0: no read noise smoothing
1: standard smoothing
2: strong smoothing

**oversub_thresh**: float

Value for SUBTHRSH keyword in PCTEFILE header. CTE corrected pixels taken below this value are re-corrected. Unit is in electrons.

**useafter**: str, optional

Value for USEAFTER keyword. Defaults to ‘Mar 01 2002 00:00:00’

**pedigree**: str, optional

Value for PEDIGREE keyword. Defaults to ‘INFLIGHT 01/03/2002 22/07/2010’

**creatorName**: str, optional

Name of the person generating this fitsFile. Defaults to ‘ACS Team’

**historyFile**: str, optional

5.2. Table Extensions
ASCII file containing HISTORY lines for EXT 0. Include path. Each row will produce one HISTORY line. Defaults to ‘’

detector : str, optional

Supported detector. Defaults to ‘WFC’

Examples

Saving file pctetab_pcte.fits with the command:

```python
>>> MakePCTETab(
... 'pctetab_pcte.fits', 'pctetab_dtdel.txt',
... ['pctetab_chgleak-1.txt', 'pctetab_chgleak-2.txt'],
... 'pctetab_levels.txt', 'pctetab_scaling.txt',
... 'pctetab_column_scaling.txt', history_file='pctetab_history.txt')
```
INDICES AND TABLES

• genindex
• modindex
• search
PYTHON MODULE INDEX

r
reftools.getphotpars, 15
reftools.imphcomp, 7
reftools.pctetab, 17
reftools.synpysyncmp, 11
Python Module Index

r
reftools.getphotpars, 15
reftools.imphcimp, 7
reftools.pctetah, 17
reftools.synpysyncomp, 11