example6

May 7, 2018

1 Opening and saving files in an optimised way

One of the main speed ups is a simple change in the efficiency of how we open and close files (especially fits files).

Previously fits files were openned and closed at multiple points through out a single recipe, even when this was to/from the same fits file (i.e. when one wants to read/write the data or read/write fits header keys to a file.

Below shows how this has been stream-lined to increase running speeds.

Before we start we will set up a "mock" DRS recipe:

In [94]:
```
# imports
from __future__ import division
import numpy as np
import os
from collections import OrderedDict

from SpirouDRS import spirouConfig
from SpirouDRS import spirouCore
from SpirouDRS import spirouImage
from SpirouDRS import spirouLOCOR
from SpirouDRS import spirouStartup

# =============================================================================
# Define variables
# =============================================================================
# Name of program
__NAME__ = 'cal_BADPIX_spirou.py'
# Get version and author
__version__ = spirouConfig.Constants.VERSION()
__author__ = spirouConfig.Constants.AUTHORS()
__date__ = spirouConfig.Constants.LATEST_EDIT()
__release__ = spirouConfig.Constants.RELEASE()
# Get Logging function
WLOG = spirouCore.wlog
# Get plotting functions
sPlt = spirouCore.sPlt
```

```
# noteboook only
import sys
sys.argv = ['example6.py']

# setup some fake file names
night_name = '20170710'
input_file = 'flat_flat02f10.fits'
output_file = 'flat_out.fits'

# ------------------------------------------------------------------
# Set up
# ------------------------------------------------------------------
# get parameters from config files/run time args/load paths + calibdb
p = spirouStartup.Begin()
# deal with arguments being None (i.e. get from sys.argv)
pos = [0, 1]
fmt = [str, str]
names = ['inputfile', 'outputfile']
call = [input_file, output_file]
# now get custom arguments
customargs = spirouStartup.GetCustomFromRuntime(pos, fmt, names, calls=call)
# get parameters from configuration files and run time arguments
p = spirouStartup.LoadArguments(p, night_name, customargs=customargs,
                                mainfitsfile='inputfile')
# as we have custom arguments need to load the calibration database
p = spirouStartup.LoadCalibDB(p)
```

```
15:19:27.0 -   || ****************************************
15:19:27.0 -   || * SPIROU @(#) Geneva Observatory (0.1.032)
15:19:27.0 -   || ****************************************
15:19:27.0 -   ||(dir_data_raw)      DRS_DATA_RAW=/scratch/Projects/spirou_py3/data/raw
15:19:27.0 -   ||(dir_data_reduc)    DRS_DATA_REDUC=/scratch/Projects/spirou_py3/data/reduced
15:19:27.0 -   ||(dir_calib_db)      DRS_CALIB_DB=/scratch/Projects/spirou_py3/data/calibDB
15:19:27.0 -   ||(dir_data_msg)      DRS_DATA_MSG=/scratch/Projects/spirou_py3/data/msg
15:19:27.0 -   ||(print_level)       PRINT_LEVEL=all         %(error/warning/info/all)
15:19:27.0 -   ||(log_level)         LOG_LEVEL=all         %(error/warning/info/all)
15:19:27.0 -   ||(plot_graph)        DRS_PLOT=1            %(def/undef/trigger)
15:19:27.0 -   ||(used_date)         DRS_USED_DATE=undefined
15:19:27.0 -   ||(working_dir)       DRS_DATA_WORKING=/scratch/Projects/spirou_py3/data/tmp
15:19:27.0 -   ||                    DRS_INTERACTIVE is not set, running on-line mode
15:19:27.0 -   |example6|Now running : example6 with:
15:19:27.0 -   |example6|       -- inputfile=flat_flat02f10.fits
15:19:27.0 -   |example6|       -- outputfile=flat_out.fits
15:19:27.0 -   |example6|ICDP_NAME loaded from: /scratch/Projects/spirou_py3/spirou_py3/INTROOT/
15:19:27.0 -   |example6|Calibration file: 20170710_flat_flat02f10_badpixel.fits already exists
15:19:27.0 -   |example6|Calibration file: 20170710_flat_dark02f10_blaze_AB.fits already exists
15:19:27.0 -   |example6|Calibration file: 20170710_dark_flat02f10_blaze_C.fits already exists -
15:19:27.0 -   |example6|Calibration file: 20170710_dark_dark02d406.fits already exists - not co
```

```
15:19:27.0 -   |example6|Calibration file: 20170710_flat_dark02f10_flat_AB.fits already exists -
15:19:27.0 -   |example6|Calibration file: 20170710_dark_flat02f10_flat_C.fits already exists -
15:19:27.0 -   |example6|Calibration file: 20170710_flat_dark02f10_loco_AB.fits already exists -
15:19:27.0 -   |example6|Calibration file: 20170710_dark_flat02f10_loco_C.fits already exists -
15:19:27.0 -   |example6|Calibration file: 20170710_flat_dark02f10_order_profile_AB.fits already
15:19:27.0 -   |example6|Calibration file: 20170710_dark_flat02f10_order_profile_C.fits already
15:19:27.0 -   |example6|Calibration file: 20170710_fp_fp02a203_tilt.fits already exists - not c
15:19:27.0 -   |example6|Calibration file: spirou_wave_ini3.fits already exists - not copied
15:19:27.0 -   |example6|Calibration file: 2017-10-11_21-32-17_hcone_hcone02c406_wave_AB.fits al
15:19:27.0 -   |example6|Calibration file: spirou_wave_ini3.fits already exists - not copied
15:19:27.0 -   |example6|Calibration file: 2017-10-11_21-32-17_hcone_hcone02c406_wave_C.fits alr
```

Lets test to see what this gives us in the parameter dictionary "p":

```
In [66]: # parameters in "p" starting with the letter "r"
         WLOG('', p['log_opt'], 'p.startswith("r") = {0}'.format(p.startswith('r')))
         # the input file
         WLOG('', p['log_opt'], 'p["inputfile"] = {INPUTFILE}'.format(**p))
         # the output file
         WLOG('', p['log_opt'], 'p["outputfile"] = {OUTPUTFILE}'.format(**p))

15:04:12.0 -   |example6|p.startswith("r") = ['REDUCED_DIR', 'RAW_DIR']
15:04:12.0 -   |example6|p["inputfile"] = flat_flat02f10.fits
15:04:12.0 -   |example6|p["outputfile"] = flat_out.fits
```

Note here that these two variables were setup as custom inputs and do not have any information about the absolute path associated with the file. Also we used the startswidth command in p to look at which variables in p "start with" the letter "r".

1.1 Reading fits files

In general this is done as before. However when opening a fits file for the first time as well as reading the data the header is also read (and stored for later use).

This means any time a HEADER key is needed there should be thought to whether the header already exists in the code (with the only exception being when we need a HEADER key from a previously unopened file).

The main read functions are shown below:

1.1.1 ReadImage

Read image is a simple function which reads the input image and header.

It returns the data, the header and comment dictionaries (header dictionary contains the header keys and their values, the comment dictionary contains the header keys and their comment values) and the dimensions of the image.

"kind" keyword arguement sets the log name (i.e. "Reading {KIND} Image {filename}")

The logging can be turned off in this function by using the log=False keyword argument.

```
In [55]: # construct file name
         flatfilename = os.path.join(p['RAW_DIR'], p['inputfile'])
         WLOG('', p['log_opt'], 'Flat file name is: {0}'.format(flatfilename))

         # Read Image (basic version)
         flat_data, fhdr, fcmt, nx1, ny1 = spirouImage.ReadImage(p, flatfilename, kind='FLAT')
```

```
15:02:41.0 -   |example6|Flat file name is: /scratch/Projects/spirou_py3/data/raw/20170710/flat_
15:02:41.0 -   |example6|Reading FLAT Image /scratch/Projects/spirou_py3/data/raw/20170710/flat_
15:02:41.0 -   |example6|FLAT Image 2048 x 2048 loaded
```

```
In [56]: WLOG('', p['log_opt'], 'flat_data shape is {0}'.format(flat_data.shape))

         print('\nThe header has the following keys:')
         print(list(fhdr.keys()))
```

```
15:02:43.0 -   |example6|flat_data shape is (2048, 2048)
```

```
The header has the following keys:
['SIMPLE', 'BITPIX', 'NAXIS', 'EXTEND', 'NEXTEND', 'FASTMODE', 'REFOUT', 'REFPIXEL', 'ACQTIME',
```

1.1.2 ReadImageAndCombine

This is a special version of the read function which combines all the fits files in 'arg_file_names' (if there are more than one) by using the "framemath" keyword argument.

i.e. in cal_loc_RAW_spirou.py the user can define multiple files at run time as follows:

>> cal_loc_RAW_spirou.py 20170710 flat_dark02f10.fits flat_dark03f10.fits flat_dark04f10.fits flat_dark05f10.fits flat_dark06f10.fits

These files are then combined (via adding them together):

```
In [57]: # set up user inputted variables via sys.argv
         import sys
         sys.argv = ('cal_loc_RAW_spirou.py 20170710 flat_dark02f10.fits flat_dark03f10.fits fla
                     'flat_dark05f10.fits flat_dark06f10.fits'.split(' '))
         # set the main function arguments to None (set from runtime via sys.argv)
         night_name = None
         files = None
```

```
In [58]: # ----------------------------------------------------------------------
         # Set up
         # ----------------------------------------------------------------------
         # get parameters from config files/run time args/load paths + calibdb
         p = spirouStartup.Begin()
         p = spirouStartup.LoadArguments(p, night_name, files)
         # run specific start up
         params2add = dict()
         params2add['dark_flat'] = spirouLOCOR.FiberParams(p, 'C')
```

```
        params2add['flat_dark'] = spirouLOCOR.FiberParams(p, 'AB')
        p = spirouStartup.InitialFileSetup(p, kind='localisation',
                                           prefixes=['dark_flat', 'flat_dark'],
                                           add_to_p=params2add, calibdb=True)
```

```
15:02:46.0 -   || ****************************************
15:02:46.0 -   || * SPIROU @(#) Geneva Observatory (0.1.032)
15:02:46.0 -   || ****************************************
15:02:46.0 -   ||(dir_data_raw)      DRS_DATA_RAW=/scratch/Projects/spirou_py3/data/raw
15:02:46.0 -   ||(dir_data_reduc)    DRS_DATA_REDUC=/scratch/Projects/spirou_py3/data/reduced
15:02:46.0 -   ||(dir_calib_db)      DRS_CALIB_DB=/scratch/Projects/spirou_py3/data/calibDB
15:02:46.0 -   ||(dir_data_msg)      DRS_DATA_MSG=/scratch/Projects/spirou_py3/data/msg
15:02:46.0 -   ||(print_level)       PRINT_LEVEL=all         %(error/warning/info/all)
15:02:46.0 -   ||(log_level)         LOG_LEVEL=all         %(error/warning/info/all)
15:02:46.0 -   ||(plot_graph)        DRS_PLOT=1            %(def/undef/trigger)
15:02:46.0 -   ||(used_date)         DRS_USED_DATE=undefined
15:02:46.0 -   ||(working_dir)       DRS_DATA_WORKING=/scratch/Projects/spirou_py3/data/tmp
15:02:46.0 -   ||                    DRS_INTERACTIVE is not set, running on-line mode
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Now running : cal_loc_RAW_spirou on file(s): flat
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|On directory /scratch/Projects/spirou_py3/data/ra
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|ICDP_NAME loaded from: /scratch/Projects/spirou_p
15:02:46.0 - * |cal_loc_RAW_spirou:02f10+[...]|Correct type of image for localisation (dark_flat
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_flat_flat02f10_badpixe
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_flat_dark02f10_blaze_A
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_dark_flat02f10_blaze_C
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_dark_dark02d406.fits a
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_flat_dark02f10_flat_AB
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_dark_flat02f10_flat_C.
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_flat_dark02f10_loco_AB
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_dark_flat02f10_loco_C.
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_flat_dark02f10_order_p
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_dark_flat02f10_order_p
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 20170710_fp_fp02a203_tilt.fits
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: spirou_wave_ini3.fits already e
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 2017-10-11_21-32-17_hcone_hcone
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: spirou_wave_ini3.fits already e
15:02:46.0 -   |cal_loc_RAW_spirou:02f10+[...]|Calibration file: 2017-10-11_21-32-17_hcone_hcone
```

The files and night_name were added in the standard way (i.e. without customisation) therefore the filenames are loaded into "fitsfilename" and "arg_file_names" as is default with the DRS:

```
In [59]: # check the file names
         WLOG('', p['log_opt'], 'fitsfilename = {FITSFILENAME}'.format(**p))
         WLOG('', p['log_opt'], 'arg_file_names = {ARG_FILE_NAMES}'.format(**p))
```

```
15:02:48.0 -   |cal_loc_RAW_spirou:02f10+[...]|fitsfilename = /scratch/Projects/spirou_py3/data/
15:02:48.0 -   |cal_loc_RAW_spirou:02f10+[...]|arg_file_names = ['flat_dark02f10.fits', 'flat_da
```

Now one can use ReadImageAndCombine to combine all the files in "arg_file_names":

```
In [60]: # ------------------------------------------------------------------
         # Read image file
         # ------------------------------------------------------------------
         # read the image data
         data, hdr, cdr, nx, ny = spirouImage.ReadImageAndCombine(p, framemath='add')
```

```
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Reading Image /scratch/Projects/spirou_py3/data/r
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Image 2048 x 2048 loaded
15:02:50.0 - * |cal_loc_RAW_spirou:02f10+[...]|Adding 4 frame(s)
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Reading File: /scratch/Projects/spirou_py3/data/r
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Reading File: /scratch/Projects/spirou_py3/data/r
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Reading File: /scratch/Projects/spirou_py3/data/r
15:02:50.0 -   |cal_loc_RAW_spirou:02f10+[...]|Reading File: /scratch/Projects/spirou_py3/data/r
```

Note that currently to be consistent with the old DRS this process updates the value of "fitsfilename" to the last file called in the combining process (this is probably not wanted but is here for consistency):

```
In [61]: WLOG('', p['log_opt'], 'fitsfilename = {FITSFILENAME}   <------ Here'.format(**p))
         WLOG('', p['log_opt'], 'arg_file_names = {ARG_FILE_NAMES}'.format(**p))
```

```
15:02:51.0 -   |cal_loc_RAW_spirou:02f10+[...]|fitsfilename = flat_dark06f10.fits   <------ Here
15:02:51.0 -   |cal_loc_RAW_spirou:02f10+[...]|arg_file_names = ['flat_dark02f10.fits', 'flat_da
```

1.2 Writing fits file and header keys

This was overhauled as it represented one of the slowest steps of any recipe.

Some simple rules should be followed:

1) generate the data to be saved

2) generate a dictionary to save the key's and the value/comments to:

   Note hdict entries should be in the following format: >> hdict['KEY'] = ['value', 'comment']

3) save the data **and** hdict one time before the recipe finishes (if at all possible)

This process is shown below for a fake output file:

```
In [103]: # reset sys.argv
          import sys
          sys.argv = ['example6']

          # fake data
          WLOG('', p['log_opt'], 'creating fake data')
```

```
        fakedata = np.random.random(size=100**2).reshape(100, 100)
        WLOG('', p['log_opt'], 'fake data shape = ({0} x {1})'.format(*fakedata.shape))

        # check file name
        WLOG('', p['log_opt'], 'output file = {OUTPUTFILE}'.format(**p))
```

```
15:28:23.0 -   |example6|creating fake data
15:28:23.0 -   |example6|fake data shape = (100 x 100)
15:28:23.0 -   |example6|output file = flat_out.fits
```

1.2.1 Adding a header

Now we can create hdict (the header dictionary).
There are several functions that can be used to help us here.

- One can import keys from another HEADER (CopyOriginalKeys)
- One can add a single key (AddKey)
- One can add a 1DList of keys
- One can add a 2DList of keys

These are all shown below:

In [96]:
```
# create the hdict (can be left for the functions to generate)
hdict = OrderedDict()

# lets copy the keys from the flat file
#  (recall the header files were called fhdr and fcmt)
hdict = spirouImage.CopyOriginalKeys(fhdr, fcmt, hdict=hdict)

# lets print the keys:
print(list(hdict.keys()))

# and lets print one entry
print('\n key = (value, comment)')
print(' gain = {GAIN}'.format(**hdict))
```

```
['NEXTEND', 'FASTMODE', 'REFOUT', 'REFPIXEL', 'ACQTIME', 'ACQTIME1', 'ASIC_NUM', 'SCA_ID', 'MUXT

 key = (value, comment)
 gain = (1.3, '[(e-/ADU)] gain')
```

In [109]:
```
# we need to use keyword stores to add keys
#   These are lists that should be located in spirouKeywords.py (loaded into "p")
# i.e.:
print('Example keyword store:')
print('\n\t kw_LOCO_NBO = {KW_LOCO_NBO}\n'.format(**p))
```

```
        # lets create a fake one to add to
        kw_inputfile = ['INFILE', 0.0, 'The input file']
        kw_fake1dlist = ['MY1LST', 0.0, 'The fake 1D list']
        kw_fake2dlist = ['MY2LST', 0.0, 'The fake 2D list']

        # lets also create a fake 2D list
        my1dlist = np.arange(0, 10)
        my2dlist = np.arange(0, 30).reshape(3, 10)

        # reset hdict from above
        hdict = OrderedDict()

        # Now lets add some keys
        hdict = spirouImage.AddKey(hdict, p['kw_version'])
        hdict = spirouImage.AddKey(hdict, kw_inputfile, value=p['inputfile'])
        hdict = spirouImage.AddKey1DList(hdict, kw_fake1dlist, values=my1dlist)
        hdict = spirouImage.AddKey2DList(hdict, kw_fake2dlist, values=my2dlist)
```

Example keyword store:

```
kw_LOCO_NBO = ['LONBO', 0, 'nb orders localised']
```

In [106]:
```
# lets print all entries in hdict:
print('Example header from hdict: ')
# loop around keys
for key in hdict:
    # get value and comment
    value, comment = hdict[key]
    # print
    print('\t{0} = {1} /{2}'.format(key, value, comment))
```

```
Example header from hdict:
        VERSION = SPIROU_0.1.032 /DRS version
        INPUTFILE = flat_flat02f10.fits /The input file
        MY1LST0 = 0 /The fake 1D list order=0
        MY1LST1 = 1 /The fake 1D list order=1
        MY1LST2 = 2 /The fake 1D list order=2
        MY1LST3 = 3 /The fake 1D list order=3
        MY1LST4 = 4 /The fake 1D list order=4
        MY1LST5 = 5 /The fake 1D list order=5
        MY1LST6 = 6 /The fake 1D list order=6
        MY1LST7 = 7 /The fake 1D list order=7
        MY1LST8 = 8 /The fake 1D list order=8
        MY1LST9 = 9 /The fake 1D list order=9
        MY2LST0 = 0 /The fake 2D list order=0 coeff=0
        MY2LST1 = 1 /The fake 2D list order=0 coeff=1
```

```
MY2LST2 = 2 /The fake 2D list order=0 coeff=2
MY2LST3 = 3 /The fake 2D list order=0 coeff=3
MY2LST4 = 4 /The fake 2D list order=0 coeff=4
MY2LST5 = 5 /The fake 2D list order=0 coeff=5
MY2LST6 = 6 /The fake 2D list order=0 coeff=6
MY2LST7 = 7 /The fake 2D list order=0 coeff=7
MY2LST8 = 8 /The fake 2D list order=0 coeff=8
MY2LST9 = 9 /The fake 2D list order=0 coeff=9
MY2LST10 = 10 /The fake 2D list order=1 coeff=0
MY2LST11 = 11 /The fake 2D list order=1 coeff=1
MY2LST12 = 12 /The fake 2D list order=1 coeff=2
MY2LST13 = 13 /The fake 2D list order=1 coeff=3
MY2LST14 = 14 /The fake 2D list order=1 coeff=4
MY2LST15 = 15 /The fake 2D list order=1 coeff=5
MY2LST16 = 16 /The fake 2D list order=1 coeff=6
MY2LST17 = 17 /The fake 2D list order=1 coeff=7
MY2LST18 = 18 /The fake 2D list order=1 coeff=8
MY2LST19 = 19 /The fake 2D list order=1 coeff=9
MY2LST20 = 20 /The fake 2D list order=2 coeff=0
MY2LST21 = 21 /The fake 2D list order=2 coeff=1
MY2LST22 = 22 /The fake 2D list order=2 coeff=2
MY2LST23 = 23 /The fake 2D list order=2 coeff=3
MY2LST24 = 24 /The fake 2D list order=2 coeff=4
MY2LST25 = 25 /The fake 2D list order=2 coeff=5
MY2LST26 = 26 /The fake 2D list order=2 coeff=6
MY2LST27 = 27 /The fake 2D list order=2 coeff=7
MY2LST28 = 28 /The fake 2D list order=2 coeff=8
MY2LST29 = 29 /The fake 2D list order=2 coeff=9
```

Note by default the dimension names are 'order' and 'coeff' this can be changed with the "dim1name" and "dim2name" keyword arguments:

```
In [110]: # reset hdict from above
          hdict = OrderedDict()

          # Now lets add some keys
          hdict = spirouImage.AddKey(hdict, p['kw_version'])
          hdict = spirouImage.AddKey(hdict, kw_inputfile, value=p['inputfile'])
          hdict = spirouImage.AddKey1DList(hdict, kw_fake1dlist, values=my1dlist, dim1name='dim1
          hdict = spirouImage.AddKey2DList(hdict, kw_fake2dlist, values=my2dlist, dim1name='dim1

          # lets print all entries in hdict:
          print('Example header from hdict: ')
          # loop around keys
          for key in hdict:
              # get value and comment
              value, comment = hdict[key]
```

```
            # print
            print('\t{0} = {1} /{2}'.format(key, value, comment))

Example header from hdict:
        VERSION = SPIROU_0.1.032 /DRS version
        INFILE = flat_flat02f10.fits /The input file
        MY1LST0 = 0 /The fake 1D list dim1=0
        MY1LST1 = 1 /The fake 1D list dim1=1
        MY1LST2 = 2 /The fake 1D list dim1=2
        MY1LST3 = 3 /The fake 1D list dim1=3
        MY1LST4 = 4 /The fake 1D list dim1=4
        MY1LST5 = 5 /The fake 1D list dim1=5
        MY1LST6 = 6 /The fake 1D list dim1=6
        MY1LST7 = 7 /The fake 1D list dim1=7
        MY1LST8 = 8 /The fake 1D list dim1=8
        MY1LST9 = 9 /The fake 1D list dim1=9
        MY2LST0 = 0 /The fake 2D list dim1=0 dim2=0
        MY2LST1 = 1 /The fake 2D list dim1=0 dim2=1
        MY2LST2 = 2 /The fake 2D list dim1=0 dim2=2
        MY2LST3 = 3 /The fake 2D list dim1=0 dim2=3
        MY2LST4 = 4 /The fake 2D list dim1=0 dim2=4
        MY2LST5 = 5 /The fake 2D list dim1=0 dim2=5
        MY2LST6 = 6 /The fake 2D list dim1=0 dim2=6
        MY2LST7 = 7 /The fake 2D list dim1=0 dim2=7
        MY2LST8 = 8 /The fake 2D list dim1=0 dim2=8
        MY2LST9 = 9 /The fake 2D list dim1=0 dim2=9
        MY2LST10 = 10 /The fake 2D list dim1=1 dim2=0
        MY2LST11 = 11 /The fake 2D list dim1=1 dim2=1
        MY2LST12 = 12 /The fake 2D list dim1=1 dim2=2
        MY2LST13 = 13 /The fake 2D list dim1=1 dim2=3
        MY2LST14 = 14 /The fake 2D list dim1=1 dim2=4
        MY2LST15 = 15 /The fake 2D list dim1=1 dim2=5
        MY2LST16 = 16 /The fake 2D list dim1=1 dim2=6
        MY2LST17 = 17 /The fake 2D list dim1=1 dim2=7
        MY2LST18 = 18 /The fake 2D list dim1=1 dim2=8
        MY2LST19 = 19 /The fake 2D list dim1=1 dim2=9
        MY2LST20 = 20 /The fake 2D list dim1=2 dim2=0
        MY2LST21 = 21 /The fake 2D list dim1=2 dim2=1
        MY2LST22 = 22 /The fake 2D list dim1=2 dim2=2
        MY2LST23 = 23 /The fake 2D list dim1=2 dim2=3
        MY2LST24 = 24 /The fake 2D list dim1=2 dim2=4
        MY2LST25 = 25 /The fake 2D list dim1=2 dim2=5
        MY2LST26 = 26 /The fake 2D list dim1=2 dim2=6
        MY2LST27 = 27 /The fake 2D list dim1=2 dim2=7
        MY2LST28 = 28 /The fake 2D list dim1=2 dim2=8
        MY2LST29 = 29 /The fake 2D list dim1=2 dim2=9
```

1.2.2 Saving the file and header (ONCE)

After the data has be generated and the hdict dictionary made (as above) one can save both of these in one operation:

```
In [112]: # construct absolute filename
          fakeoutfile = os.path.join(p['REDUCED_DIR'], p['outputfile'])
          # write to file (filename, data, hdict)
          WLOG('', p['log_opt'], 'Writing file {0} to disk'.format(fakeoutfile))
          spirouImage.WriteImage(fakeoutfile, fakedata, hdict)
```

```
15:29:41.0 -   |example6|Writing file /scratch/Projects/spirou_py3/data/reduced/20170710/flat_ou
```