SFIT Processing Environment

Ivan Ortega, James Hannigan, Eric Nussbaumer

SFIT4 workshop

Nov 4-6, 2019; Boulder, CO

Introduction

The sfit processing environment is the machinery/tools surrounding the sfit core code. The ultimate goal is to:

- Create a directory structure to organize the output data
- Generate the necessary input files to run SFIT core code →Pre-Processing
- Execute the SFIT core code and error analysis on output → Processing
- Plotting results, HDF creation, analysis of retrievals → Post-Processing

The majority of the processing environment is written in python!

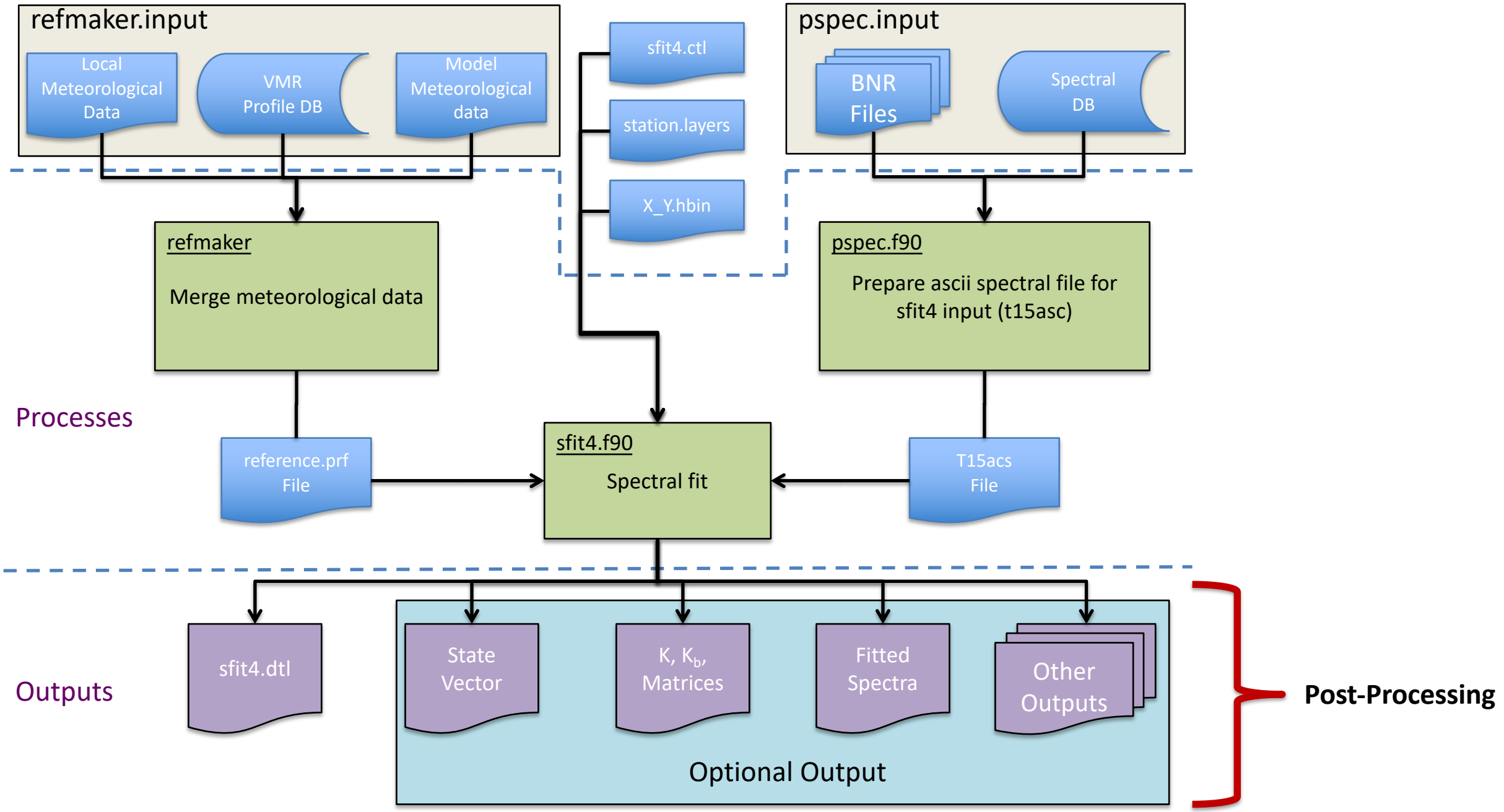
We should use Python 3x going forward. Python 2 will be in EOL as of Jan 2020.

Input and Output flow for Core Processing

Inputs

Processes

Outputs



Post-Processing

The post-processing step involves plotting and analyzing the results of one or more retrievals.
We have programs to show standard plots: fits, AKs, Jacobian, profiles, errors, summary outputs

Program	Code	Purpose
pltRet.py	python	Program to plot individual retrieval using command line arguments. Check pltRet.py -? for options
pltSet.py	python	Program to plot multiple retrievals using an input file
setInput.py	python	Editable input file for pltSet.py

There are no filtering options for a single retrieval; however, for a set of retrieval there are multiple parameters that one can filter on such as RMS, DOFs, dates, etc. The program `pltRet.py` creates plots for a single retrieval and only requires command line arguments. Using the option `pltRet.py -S` would save plots into a pdf file.

The program `pltSet.py` plots an entire set of retrievals and requires an input file (`setInput.py`).

Both working with python 2.7 and 3x

Plot individual retrieval (single measurement)

```
>> pltRet.py -?

pltRet.py [-i <str> ]
-i <dir>      Data directory. Optional: default is current
working directory
-S            Flag to save results in pltRet.pdf. Optional:
default is False
```

Purpose:
This program is use to plot individual results of sfit4
-- Jacobian Matrix
-- Fit retrievals/residuals in all micro-windows
-- Averaging Kernels (Matrix, vmr, and unitless)
-- Profiles of all gases in mixing ratios
-- Profile error are shown if error are calculated
-- Cumulative sum of DOF profile
-- Summary Files, including error summary if present, are printed in terminal
-- Optional to save PDF file

Needed modules:
import dataOutClass
import matplotlib

Check out the pdf example provided

- These routines are meant as diagnostic tools.
- Easily expanded/add plots
  - e.,g, LOS
  - What else?
- Can easily be adapted for high quality figure for articles.

Plot set of retrievals (multiple measurement)

```
>> pltSet.py -?

pltSet.py [-i <str> -? ]
-i <file> : Run pltSet.py with specified input file
-?        : Show all flags
```

Tip: a setinput.py file is located in every gas folder, so one can run this routine easily every time.

What are the important inputs in this routine?

Example of setinput.py

```
#-----------------------------------------------------------------------------
# Name:
#       setInput.py
#
# Purpose:
#       This is the input file for pltSet.py
#
#-----------------------------------------------------------------------------
loc             = 'fl0'             # Name of station location
gasName   = 'co'              # Name of gas
ver             = 'Current_v3'          # Name of retrieval version to process
ctlF            = 'sfit4_v3.ctl'          # Name of ctl file

#------
# Flags
#------
saveFlg     = True              # Flag to either save data to pdf file (saveFlg=True) or plot to screen (saveFlg=False)
errorFlg      = True             # Flag to process error data
fltrFlg         = True             # Flag to filter the data
byYrFlg       = False            # Flag to create plots for each individual year in date range
szaFlg        = True             # Flag to filter based on min and max SZA
dofFlg        = True             # Flag to filter based on min DOFs
pcNegFlg   = True             # Flag to filter profiles with negative partial columns
```

```
tcNegFlg   = True              # Flagsag to filter profiles with negative total columns
tcMMFlg    = False             # Flag to filter based on min and max total column amount
cnvrgFlg   = True              # Flag to filter profiles that did not converge
rmsFlg     = True              # Flag to filter based on max RMS
chiFlg     = False             # Flag to filter based on max CHI_2_Y

maxRMS     = 1.5                   # Max Fit RMS to filter data. Data is filtered according to <= maxrms
minDOF     = 0.9               # Min DOFs for filtering
minSZA     = 0.0               # Min SZA for filtering
maxSZA     = 90.0              # Max SZA for filtering
maxCHI     = 2.0               # Max CHI_y_2 value
maxTC      = 5.0E24            # Max Total column amount for filtering
minTC      = 0.0               # Min Total column amount for filtering
sclfct     = 1.0E9             # Scale factor to apply to vmr plots (ppmv=1.0E6, ppbv=1.0E9, etc)
sclfctName = 'ppbv'            # Name of scale factor for labeling plots

#----------------------
# Date range to process
#----------------------
iyear      = 2010
imnth      = 1
iday       = 1
fyear      = 2019
fmnth      = 12
fday       = 31
```

Check out the pdf example provided

HDF creation

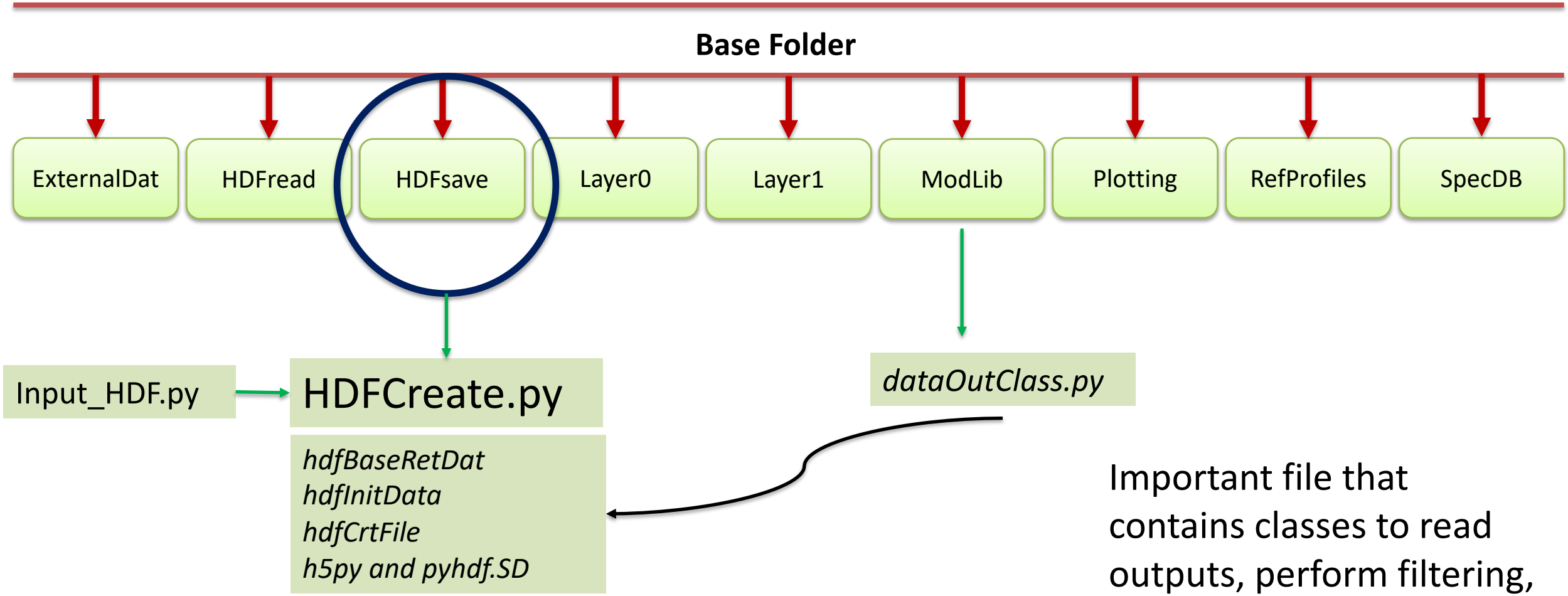
Hierarchical Data Format (**HDF**) is a set of file formats (**HDF4**, **HDF5**) designed to store and organize large amounts of data.

Supported by many software, it also has a Java-based HDF Viewer (HDFView)

The current version, HDF5, differs significantly in design and API from the major legacy version HDF4.

The quest for a portable scientific data format began in 1987 by the Graphics Foundations Task Force (GFTF) at the National Center for Supercomputing Applications (NCSA). NSF grants received in 1990 and 1992 were important to the project. Around this time NASA investigated 15 different file formats for use in the Earth Observing System (EOS) project. After a two-year review process, HDF was selected as the standard data and information system (source: https://en.wikipedia.org/wiki/Hierarchical_Data_Format).

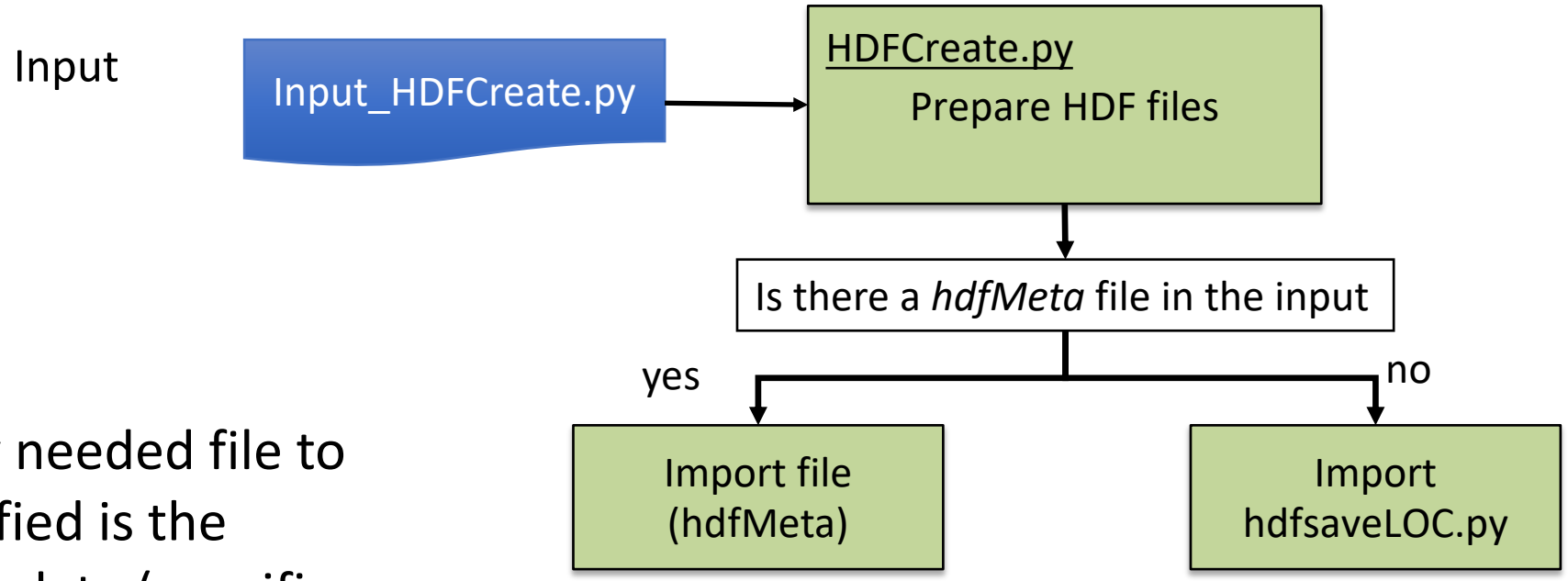
HDF creation



Input file with important information regarding the location, gas, paths, dates, and filtering.

Important file that contains classes to read outputs, perform filtering, creates plots… etc.

Diagram of HDF creation



The only needed file to be modified is the hdfMeta data (specific for each site)

Needed modules (provided):

dataOutClass
hdfBaseRetDat
hdfInitData
hdfCrtFile
h5py and pyhdf.SD

Example of input file for HDFCreate.py

```
#-------------------------------------------------------------------------------
# Name:
#       input_HDFCreate.py
#
# Purpose:
#       This is the input file for HDFCreate.py
#
#-------------------------------------------------------------------------------
loc            = 'tab'              # Name of station location
gasName    = 'co'              # Name of gas
ver            = 'Current_B3_RD'       # Name of retrieval version to process
ctlF           = 'sfit4_v3.ctl'          # Name of ctl file

#------
#Some Meta-data for hdf file (Global Attributes) --> More in hdfsave.py
#------
sfitVer        = '0.9.4.4'
fileVer        = '004'             # Updated October 2017
projectID    = 'QA4ECV'

#------
#Python Flg: if True Use Python Interface; if False use IDL Interface
#------
pyFlg       = True

yrlFlg      = True              # If True will create yearly files from Jan 1 to Dec 31; if False will create use single file from date range below
```

```
spcDBFile    = 'HRspDB_tab_RD.dat'
statLyrFile  = 'station.layers'

#------
# If pyFlg is False the below IDL file is needed
#------
idlFname      = ' '

#---------------------
# Date range
#---------------------
iyear       = 2019
fyear       = 2019

#------
# Flags
#------
errFlg      = True
szaFlg      = True              # Flag to filter based on min and max SZA
dofFlg      = True              # Flag to filter based on min DOFs
pcNegFlg    = True              # Flag to filter profiles with negative partial columns
tcNegFlg    = True              # Flagsag to filter profiles with negative total columns
tcMMFlg     = False             # Flag to filter based on min and max total column amount
cnvrgFlg    = True              # Flag to filter profiles that did not converge
rmsFlg      = True              # Flag to filter based on max RMS
chiFlg      = False             # Flag to filter based on max CHI_2_Y
h2oFlg      = True              # Flag to filter Negative Water Vapor Columns
```

```
maxRMS      = 2.5              # Max Fit RMS to filter data. Data is filtered according to <= maxrms
minDOF      = 1.0              # Min DOFs for filtering
minSZA      = 0.0              # Min SZA for filtering
maxSZA      = 90.0             # Max SZA for filtering
maxCHI      = 2.0              # Max CHI_y_2 value
maxTC       = 5.0E24           # Max Total column amount for filtering
minTC       = 0.0              # Min Total column amount for filtering

#------
# OPTIONAL (dQuality,
#-----

dQuality = 'RD'    # Data_Source - Created for Rapid Delivery - dataStr['DATA_QUALITY']
hdfMeta = 'hdfsaveFL0.py'
```

Example of file name:
groundbased_ftir.o3_ncar001_thule_20191019t140535z_20191020t181338z_004

Latest version: dates can be included in the command line and will overwrite the dates in the input file (handy for near real time)

```
>> HDFCreate.py –i input_HDFCreate.py – d 20191001_20191015
```

To include soon:

- Flag to create either hdf4 or hdf5 (this is implemented but default is hdf4)
- Read from the database either SAzm or NAzm; W_Lon or E_Lon
- Any other important meta data that should be included here?

Plotting HDFs

One can plot a set of HDF files. The needed python files to plot HDF are in the HDFread folder.

The program pltHDF.py creates plots for a single or multiple years as specified in the input file.

The program pltHDF.py requires an input file (input_HDFRead.py). The inputs and flags needed in `input_HDFRead.py` are self explanatory. To create plots using HDF files use the following syntax:

$ python pltHDF.py -i input_HDFRead.py

- Expect to make these tools available

- Any feedback/recommendations would be ok