SFIT Processing Environment

Ivan Ortega, James Hannigan, Eric Nussbaumer

SFIT4 workshop

Nov 4-6, 2019; Boulder, CO

Updated May, 2020

NCAR
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH

ACOM

Introduction

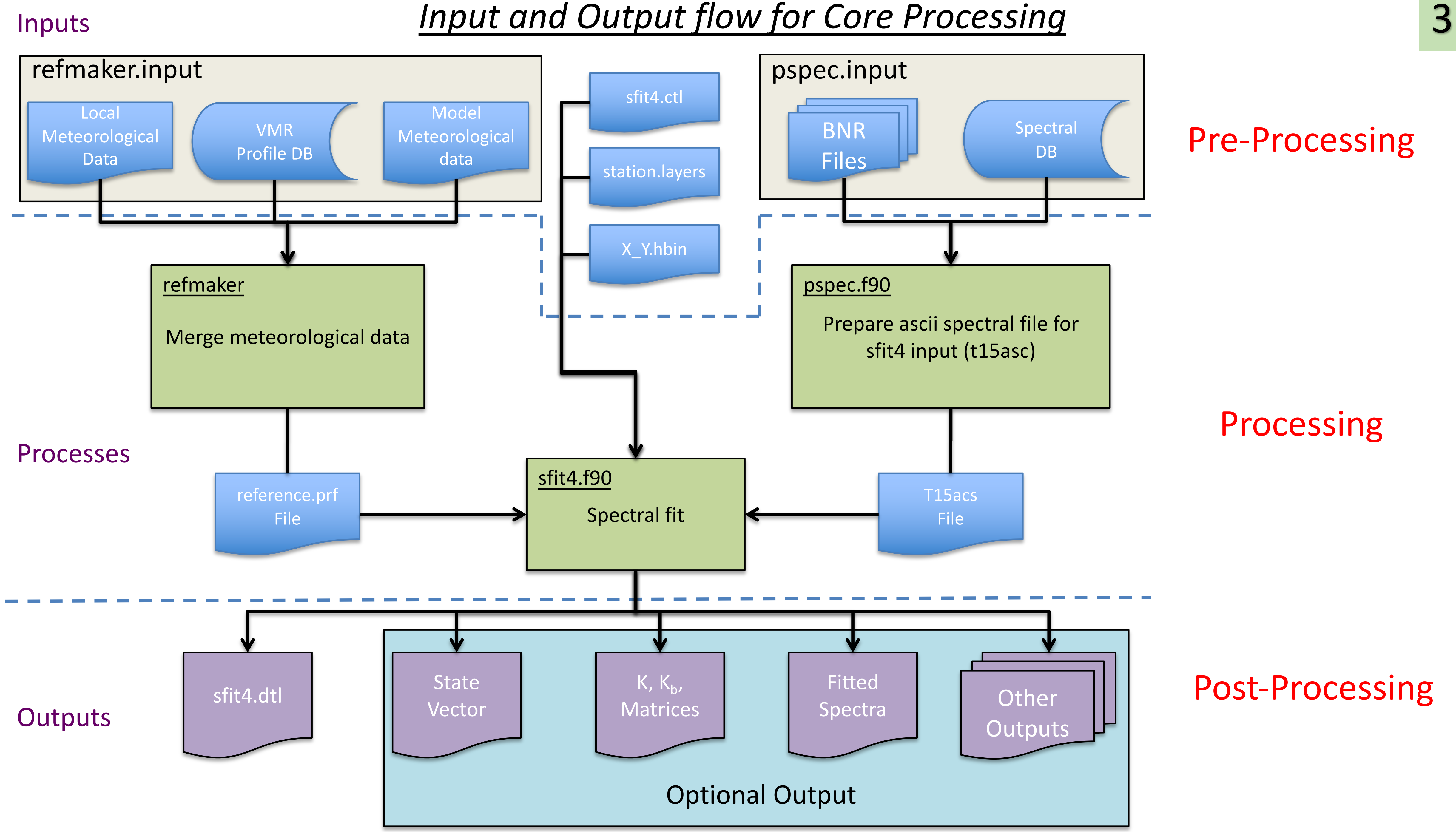
The sfit processing environment is the machinery/tools surrounding the sfit core code. The ultimate goal is to:

- 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

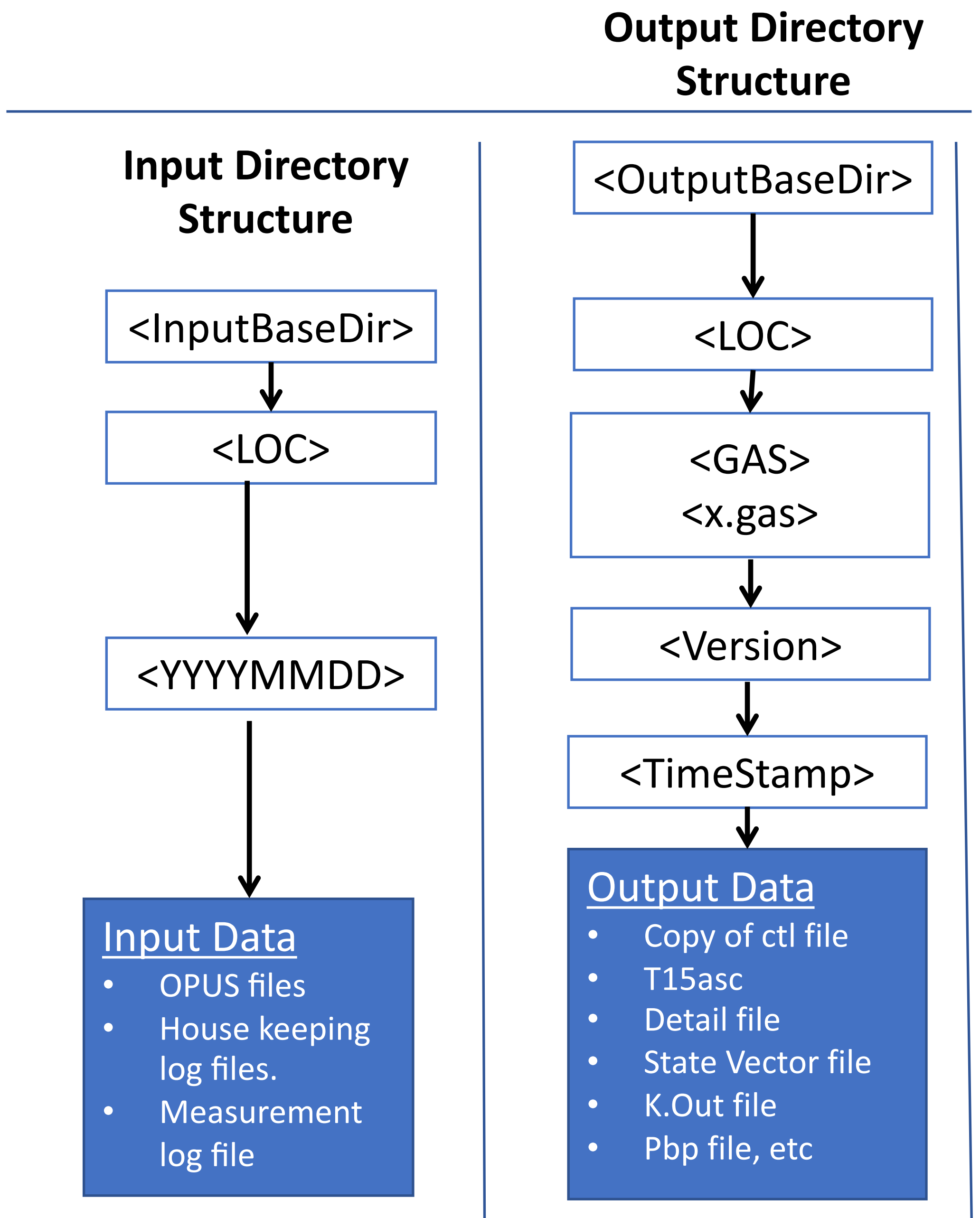
The majority of the processing environment is written in python!

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

Input and Output flow for Core Processing



Directory structure of input and outputs that are employed within this environment.



- <InputBaseDir> : Base directory for input file structure
- <OutputBaseDir> : Base directory for output file structure (can be same as <InputBaseDir>
- <LOC> : Three letter site location abbreviation
- <YYYYMMDD> : Year, month, day of observation
- <GAS> : Primary gas of interest for retrieval
- <X.gas> : All inputs files & data for this gas
- <TimeStamp> : UTC time stamp of observation HHMMSS.SS
- <Version> : User defined description of ctl file used for processing

e.g., (Input directory): /data/MLO/20191001

e.g., (output directory):
/data/MLO/ch4/Current/20181231.212342
/data/MLO/ch4/x.ch4/sfit4.ctl
Note: the above can be applied also to airborne measurements

Pre-Processing (offline)

Pre-processing involves creating the spectral database file which has information regarding a spectral observation, extracting relevant HITRAN line lists, and preparing ZPTW profiles (altitude, pressure, temperature, water vapor) from other sources such as NCEP/ERA.

- Prepare spectral database
- Prepare ZPTW (altitude, pressure, temperature, and water vapor)
- Prepare WACCM to reference (every group might have this already, see wiki, or ask Jim)
- Prepare HITRAN hbin file
  - Linelist (provided)
  - Prepare sfit4.ctl file
  - Prepare isotope.ctl file
- Prepare ils data?

Pre-Processing: Spectral database

There are several steps in creating the spectral database:

1. **Creating the initial spectral database (info from OPUS)**
2. Re-formatting the house keeping log files
3. Re-formatting the external station weather data
4. Appending the initial spectral database with house an external station weather data

Note that not all sites have house or external station weather data. Only step 1 is carried out. However, they are highly recommended, especially pressure and temperature values… and for airborne measurements GPS information

Do we create a database for all spectra recorded?

We recommend to do an initial quality check of the spectra, i.e., remove low quality spectra.

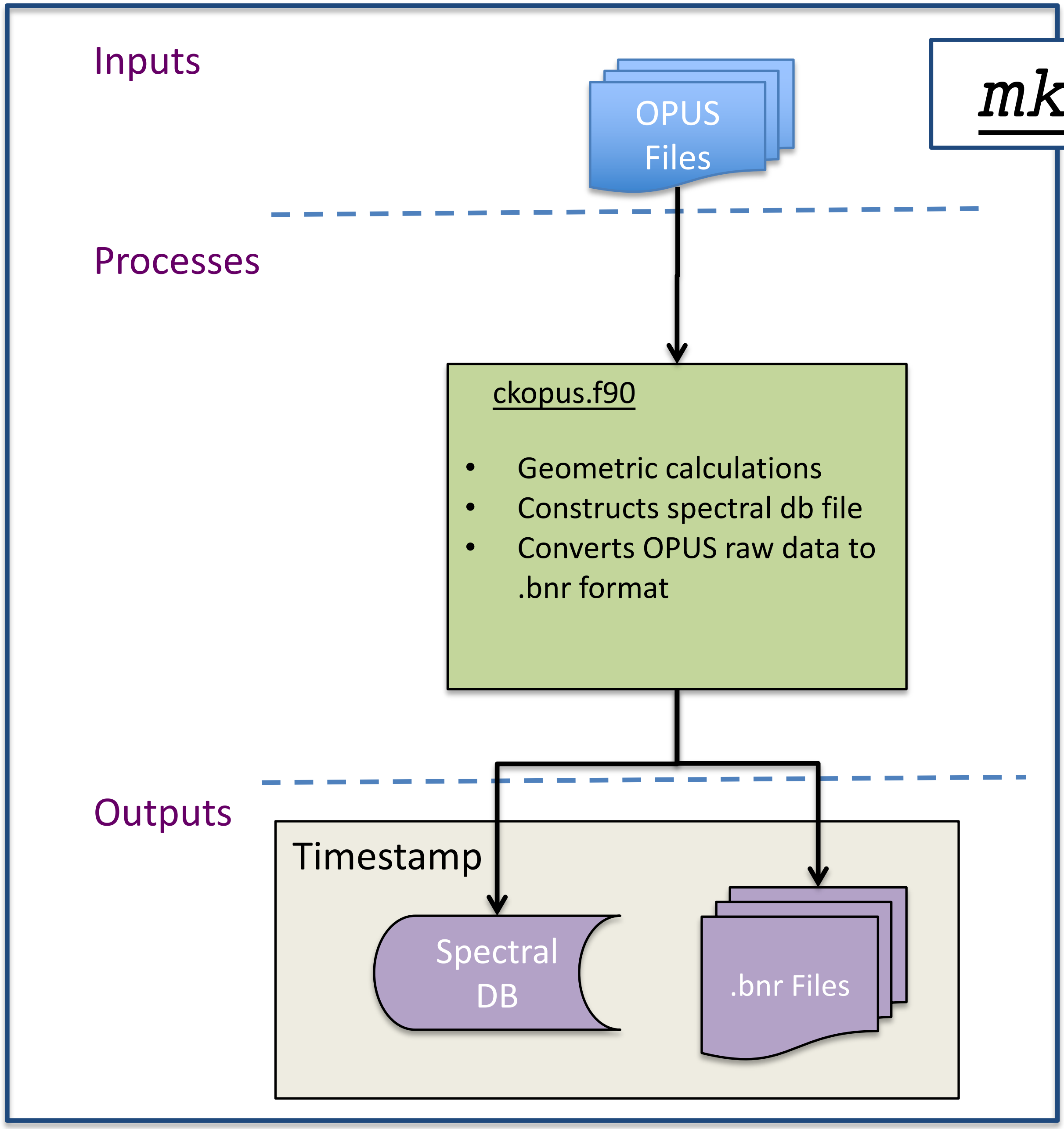
Pre-Processing : Initial quality check of opus files

We currently have two tools:

1. An IDL program (`ckop.pro`), which allows the user to look through each individual spectra and discard or keep it.

2. A GUI written in python (`ckopPy.py`). This python script uses a python Class to read opus format (nicely provided by Wolfgang Stremme, CCA-UNAM, Mexico). This GUI calculates a SNR based on out of band noise (or any other band) and maximal signal. Additionally, a proxy is created to integrate positive and negative values to create a ratio as a second quality check for each spectra. Furthermore, we can plot time series of SNR, and or log HK files (available upon request).

Input and Output Flow for spectral database



`mkSpecDB.py`

The python program **`mkSpecDB.py`** and the C program **`ckopus.c`** are used to create the initial spectral database. The `ckopus.c` program also the ability to convert OPUS files to regular binary files. The spectral database file catalogs the measurements and associates important meta-data with each. Meta-data includes: time-stamp, solar zenith angle, etc.

There are two options to run `mkSpecDB.py`:

(1) With an input file: **`specDBInputFile.py`** → dates, paths input/output, ckopus path/flags, bnr format.

(2) Command lines.

The initial spectral databases should be made for individual years. The output files have the names spDB_loc_YYYY.dat; e.g., **spDB_MLO_2019.dat**

mkSpecDB.py

```
>> mkSpecDB.py -?

mkSpecDB.py [-i <File> -D <Directory> -s tab/mlo/fl0 -d 20180515 -?
•  There are two options to run mkSpecDB.py:
•  (1) mkSpecDB.py -i <File>. In this case the input file needs to be modified accordingly.
•  (2) mkSpecDB.py -s tab/mlo/fl0 -d 20180515 -?
•    -i             : input File
•    -D             : only creates a processed folder list with opus files
•    -s             : Flag Must include location: e.g., mlo/tab/fl0
•    -d <20180515> or <20180515_20180530>  : Flag to specify input Dates. If not Date is
   specified current date is used.
•    -?             : Show all flags'

Note: if input file is provided the location, dates, etc need to be modified accordingly
Note: if input file is not provided the location, dates, are taken from -s -d, and additional
hardcoded inputs are in mkSpecDB.py
```

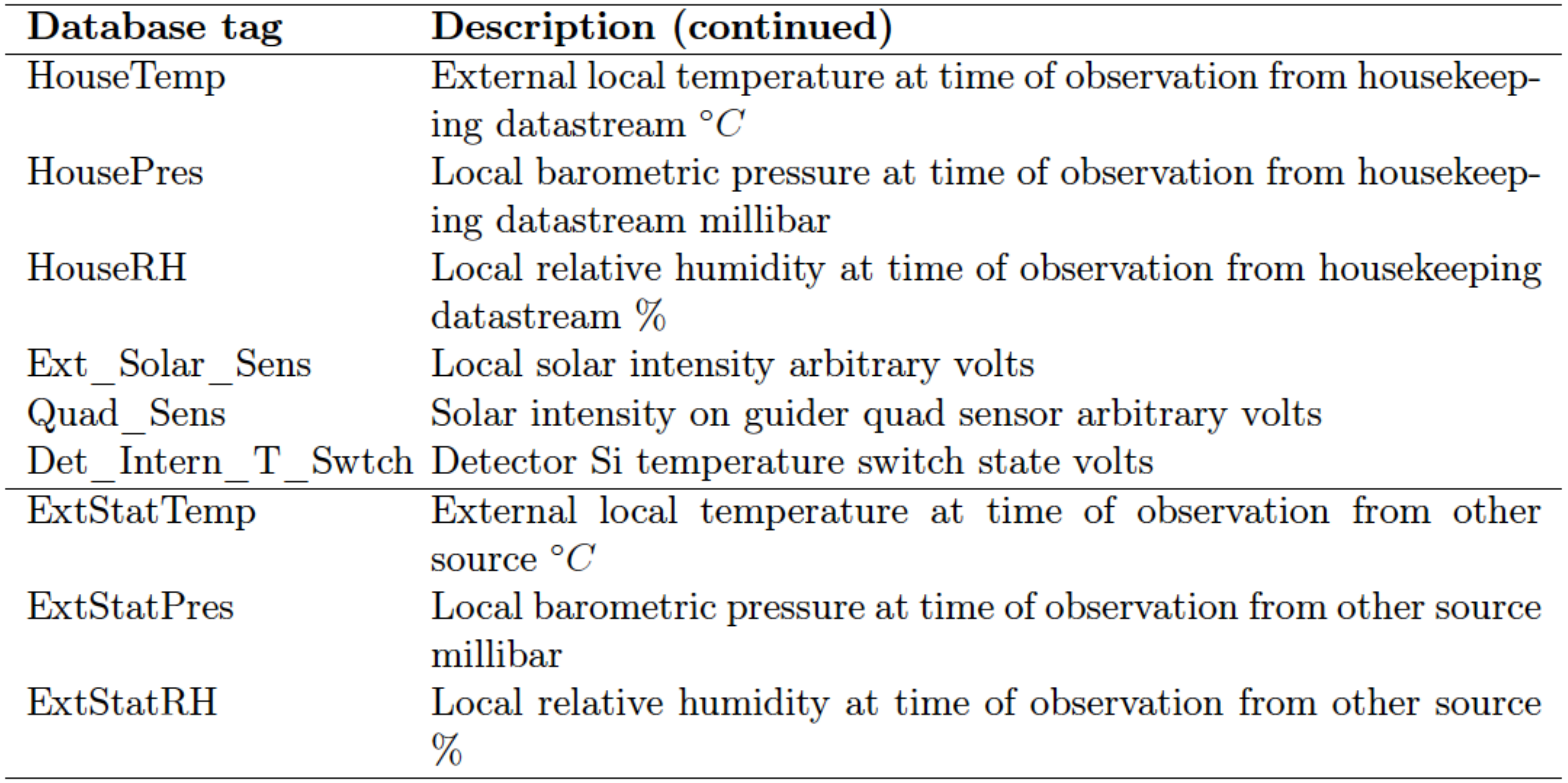
Output example

```
Filename    Site   SBlock  TOffs   TStamp  Date      Time     SNR   N_Lat  W_Lon   Alt     SAzm    SZen   ROE    Dur        Reso    Apd     FOV  LWN     HWN    Flt       MaxY      MinY       FLSCN       EXSCN  GFW  GBW
s1ifml1a.0  MLO    SNGC    0.0284  182459  20191001  18:24:59 0.0   19.54  155.57  3396.0  285.86  60.22  6377.6738  204.70  0.0035  BX   1.9139  0.998  4349.998  1  5.568e+00  -3.222e+00  2   2    1    1
s1ifml1a.1  MLO    SNGC    0.0284  200925  20191001  20:09:25 0.0   19.54  155.57  3396.0  303.90  37.80  6368.6706  204.70  0.0035  BX   1.9139  0.998  4349.998  1  1.503e+01  -1.349e+01  2   2    1    1
s1ifml1a.2  MLO    SNGC    0.0284  210348  20191001  21:03:48 0.0   19.54  155.57  3396.0  321.98  28.34  6356.9242  204.70  0.0035  BX   1.9139  0.998  4349.998  1  2.159e+01  -1.650e+01  2   2    1    1
```

List and description of database tags

Database tag	Description
Filename	OPUS path and filename
Site	3 lettersite name specifier (see constant.c)
SBlock	OPUS data block name e.g.EMIS, with OPUS transmissionsolar spectra
TOffs	Time offset in seconds required for ZPD correction (decimal)
TStamp	UT time of ZPD after UT, misc. and ZPD corrections HHMMSS
Date	UT date of ZPD YYYYMMDD
Time	UT time hh:mm:ss
SNR	Signal-to-noise ratio from stored value in OPUS file
N_Lat	Latitude of observation site, positive north, decimal degrees
W_Lon	Longitude of observation site, positive west, decimal degrees
Alt	Altitude of observation site, meters asl
SAzm	Azimuth angle of solar position at ZPD calculated in ckopus positive west of south, decimal degrees
SZen	Zenith angle of solar position at ZPD calculated in ckopus, decimal degrees
ROE	Radius of Earth at SAzm kilometers
Dur	Total integration time of observation seconds
Reso	Spectral resolution of spectrum as calculated in OPUS
Apd	Apodization function applied to spectrum in block SBlock by OPUS
FOV	Full field of view of spectrum using aperture and fore optic focal length milliradians
LWN	Low wavenumber in spectrum in block SBlock $\mathrm{cm}^{-}1$
HWN	High wavenumber in spectrum in block SBlock $\mathrm{cm}^{-}1$
Flt	Filter ID code from OPU via ckopus.c:filterid() 1 char
MaxY	Maximum spectral point value in block SBlock
MinY	Minimum spectral point value in block SBlock
FLSCN	Number of requested scans
EXSCN	Number of recorded scans
GFW	Number of good forward scans
GBW	Number of good backward scans

What housekeeping info can be appended to the initial database?



Database tag	Description (continued)
HouseTemp	External local temperature at time of observation from housekeeping datastream $°C$
HousePres	Local barometric pressure at time of observation from housekeeping datastream millibar
HouseRH	Local relative humidity at time of observation from housekeeping datastream %
Ext_Solar_Sens	Local solar intensity arbitrary volts
Quad_Sens	Solar intensity on guider quad sensor arbitrary volts
Det_Intern_T_Swtch	Detector Si temperature switch state volts
ExtStatTemp	External local temperature at time of observation from other source $°C$
ExtStatPres	Local barometric pressure at time of observation from other source millibar
ExtStatRH	Local relative humidity at time of observation from other source %

Any other important information can be appended, e.g., for mobile platforms, lat/lon/altitude, etc

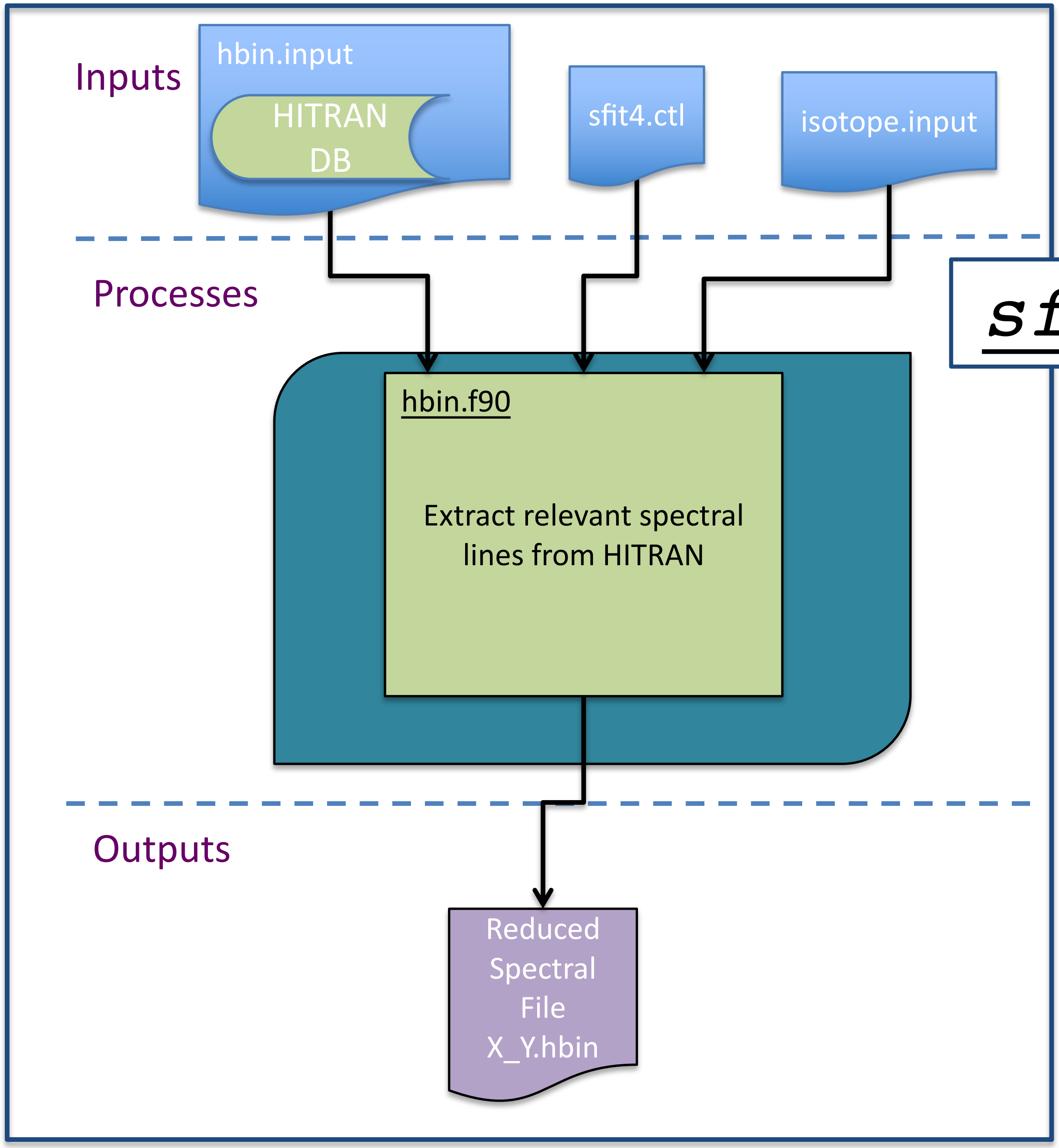
Py programs to append data

Program	Code	Purpose
appendSpecDB.py	python	Program to create the append spectral database file
appndSpecDBInputFile.py	python	Editable input file for appendSpecDB.py

```
>> appendSpecDB.py -?

appendSpecDB.py [-i <File> -D <Directory> -s tab/mlo/fl0 -y 2019 -?
There are two options to run appendSpecDB.py:
(1) appendSpecDB.py -i <File>. In this case the input file needs to be modified accordingly.
(2) appendSpecDB.py -s tab/mlo/fl0 -y 2018 -?
  -i             : input File
  -s             : Flag Must include location: e.g., mlo/tab/fl0
  -y      <YYYY> : Flag to specify year.
  -?             : Show all flags
Note: if input file is provided the location, dates, etc need to be modified accordingly
Note: if input file is not provided the location, dates, are taken from -s -d, and additional hardcoded inputs
are in appendSpecDB.py
```

Note: there is a previous step to read site specific format files. Modifications/edits need to be accomplished to read properly different formats

Input and Output Flow for HITRAN hbin file


Inputs
hbin.input
HITRAN DB
sfit4.ctl
isotope.input
Processes
hbin.f90
Extract relevant spectral lines from HITRAN
Outputs
Reduced Spectral File X_Y.hbin

The HITRAN hbin file can be created calling either hbin directly or with sfit4Layer0.

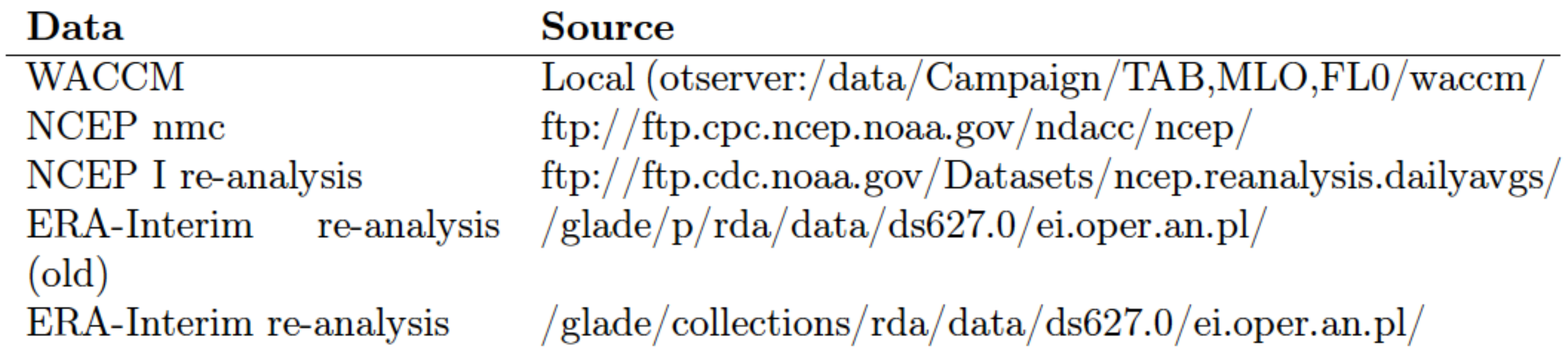
sfit4Layer0.py

```
>> sfit4Layer0.py -?
sfit4Layer0.py -f <str> [-i <dir> [-b <dir/str> -?]


-i <dir> Data directory. Optional: default is current directory
-f <str> Run Flags: Necessary: h = hbin, p = pspec, s = sfit4, e =
error analysis, c = clean
-b <dir/str> Binary sfit directory. Optional: default is hard-
coded in main(). Also accepts v1, v2, etc.

             v1:/data/ebaumer/Code/sfit-core-code/src/
             v2:/data/tools/400/sfit-core/src/
             v3:/Users/jamesw/FDP/sfit/400/sfit-core/src/
             v4:/home/ebaumer/Code/sfit4/src/
             v5:/Users/jamesw/FDP/sfit/400/src/src-irwg14-mp
             v6:/data/ebaumer/Code/ sfit-core-code-1.0.5/src/
```

Input and Output Flow for ZPTW profiles

Temperature and pressure profiles are taken from NCEP nmc data. Available for NDACC sites:
ftp://ftp.cpc.ncep.noaa.gov/ndacc/ncep

Currently water vapor profiles are taken from NCEP (daily) I and ERA-Interim (6h) re-analysis data. Both NCEP and ERA-Interim data are interpolated with WACCM data to reach 120km vertical height.



Data	**Source**
WACCM	Local (otserver:/data/Campaign/TAB,MLO,FL0/waccm/
NCEP nmc	ftp://ftp.cpc.ncep.noaa.gov/ndacc/ncep/
NCEP I re-analysis	ftp://ftp.cdc.noaa.gov/Datasets/ncep.reanalysis.dailyavgs/
ERA-Interim re-analysis (old)	/glade/p/rda/data/ds627.0/ei.oper.an.pl/
ERA-Interim re-analysis	/glade/collections/rda/data/ds627.0/ei.oper.an.pl/

Table 13: *Reference profiles web sources.*

We have a script that pulls raw data from the above sites every day under crontab.

Pressure and temperature profiles in the ZPT.nmc.120 files come from NCEP nmc data. The NCEP nmc data is vertically interpolated with WACCM data to reach 120km. In the event that the NCEP NMC data is not available for a particular day, the WACCM data is substituted.

Program	Code	Purpose
NCEPnmcFormat.py	python	Program to format the NCEP nmc data
NCEPinputFile.py	python	Editable input file for NCEPnmcFormat.py program
MergPrf.py	python	Main program to create ZPT and water files from WACCM data
mergprfInput.py	python	Input file for MergPrf.py program

NCEP I & ERA Interim Water Profiles

Program	Code	Purpose
cnvrtNC.py	python	Program to convert ERA-Interim GRIB files to NetCDF files
ERAwaterPrf.py	python	Program to extract daily averaged and 6h water profiles from ERA-Interim
NCEPwaterPrf.py	python	Program to create daily water profiles from NCEP I

Note: ERA5 provides hourly estimates of a large number of atmospheric parameters and might need to be considered in the near future.

Retrieved Water Profiles

- For all sites water vapor is retrieved when available. This water can be used as a prior for other retrievals and preferably for NDACC archive dataset.

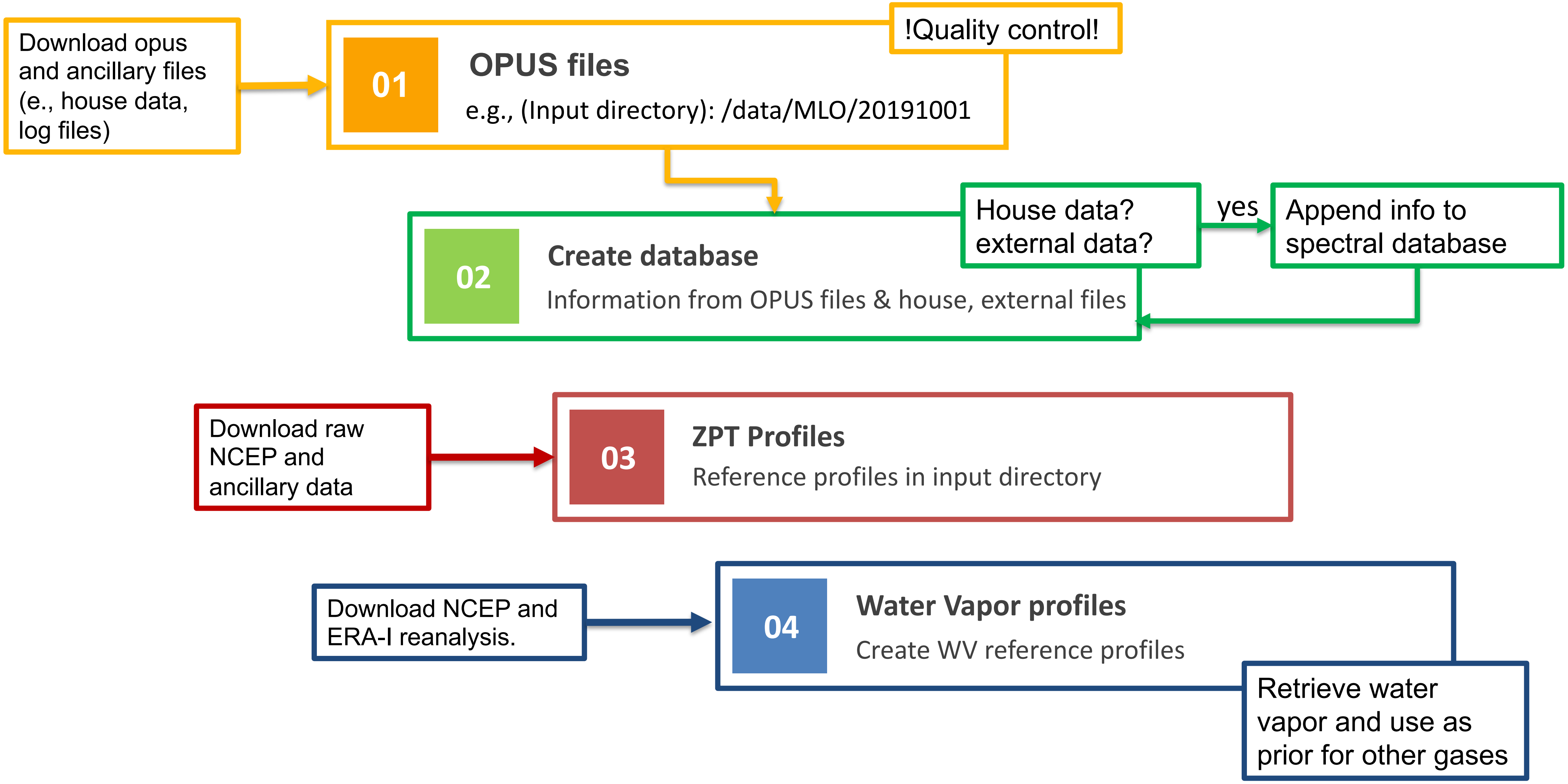
- The program `retWaterPrf.py` creates w-120.YYYYMMDD.HHMMSS.v99 for each retrieval. These files are stored in the data directories.

- A daily average of these profiles can be created using the program `retWaterPrfDaily.py`. These daily averages are also stored in the main data directories.

All profiles reside in the data directories (e.g., /data/MLO/20191001)

Profile Type	**Source**	**File Name**
Temperature	NCEP nmc	ZPT.nmc.120
Pressure	NCEP nmc	ZPT.nmc.120
Water Vapor	WACCM	w-120.v1
Water Vapor	NCEP I	w-120.v3
Water Vapor	ERA-Interim	w-120.v4
Water Vapor	ERA-Interim-6h	w-120.YYYYMMDD.HH0000.v66
Water Vapor	Retrieved	w-120.YYYYMMDD.HHMMSS.v99
Water Vapor	Retrieved Daily	w-120.v5

Overview: Steps for Pre-Processing


Download opus and ancillary files (e., house data, log files)
!Quality control!
01
OPUS files
e.g., (Input directory): /data/MLO/20191001
02
Create database
Information from OPUS files & house, external files
House data? external data?
yes
Append info to spectral database
Download raw NCEP and ancillary data
03
ZPT Profiles
Reference profiles in input directory
Download NCEP and ERA-I reanalysis.
04
Water Vapor profiles
Create WV reference profiles
Retrieve water vapor and use as prior for other gases

Multiple and single Processing

Layer0

The purpose of Layer0 is to run a single retrieval.

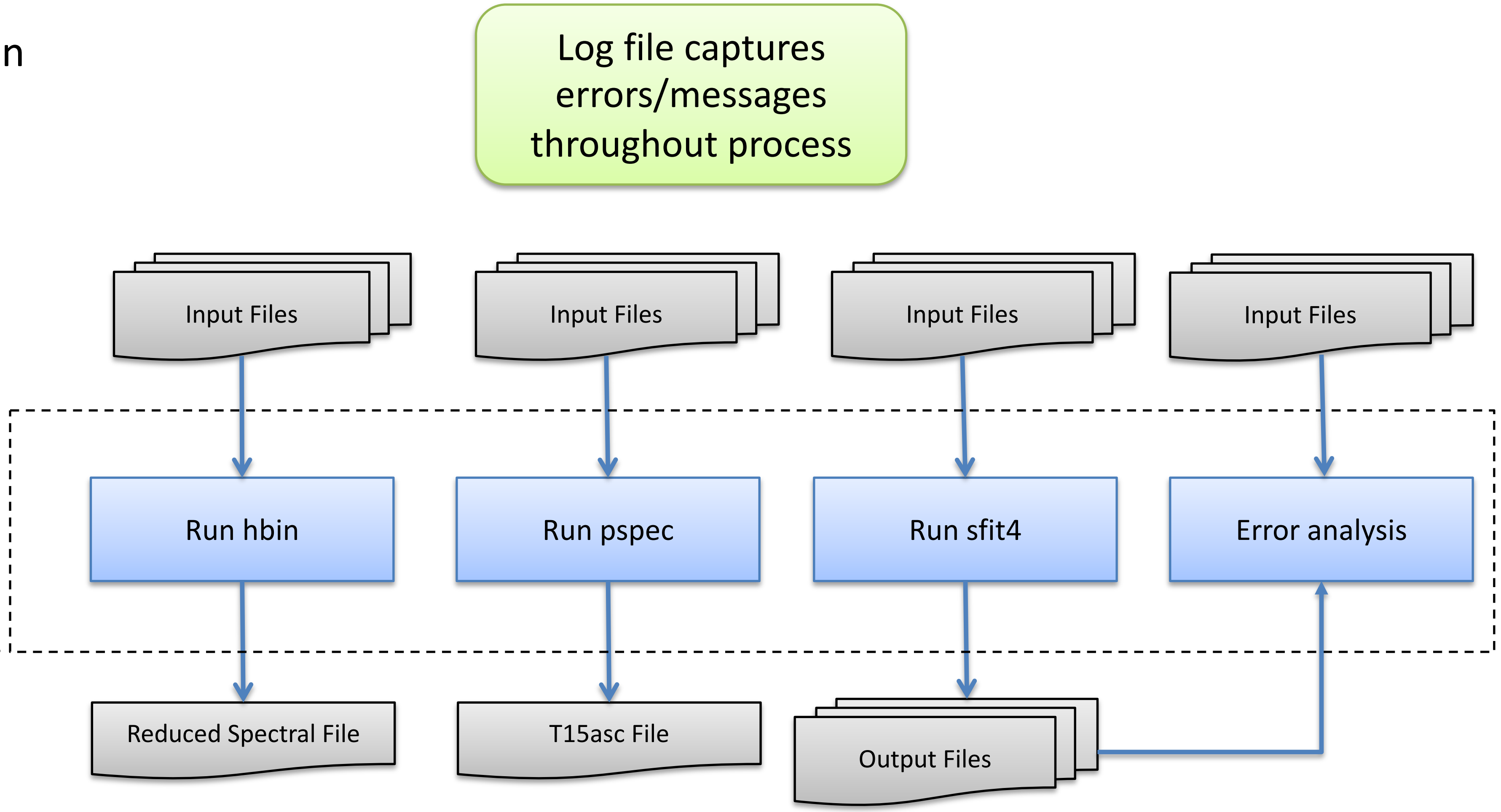
The program sfit4Layer0.py runs layer 0.

This program is called with command line arguments.

There is no input file.

It can run hbin, pspec, or sfit4 independently.

sfit4Layer0.py



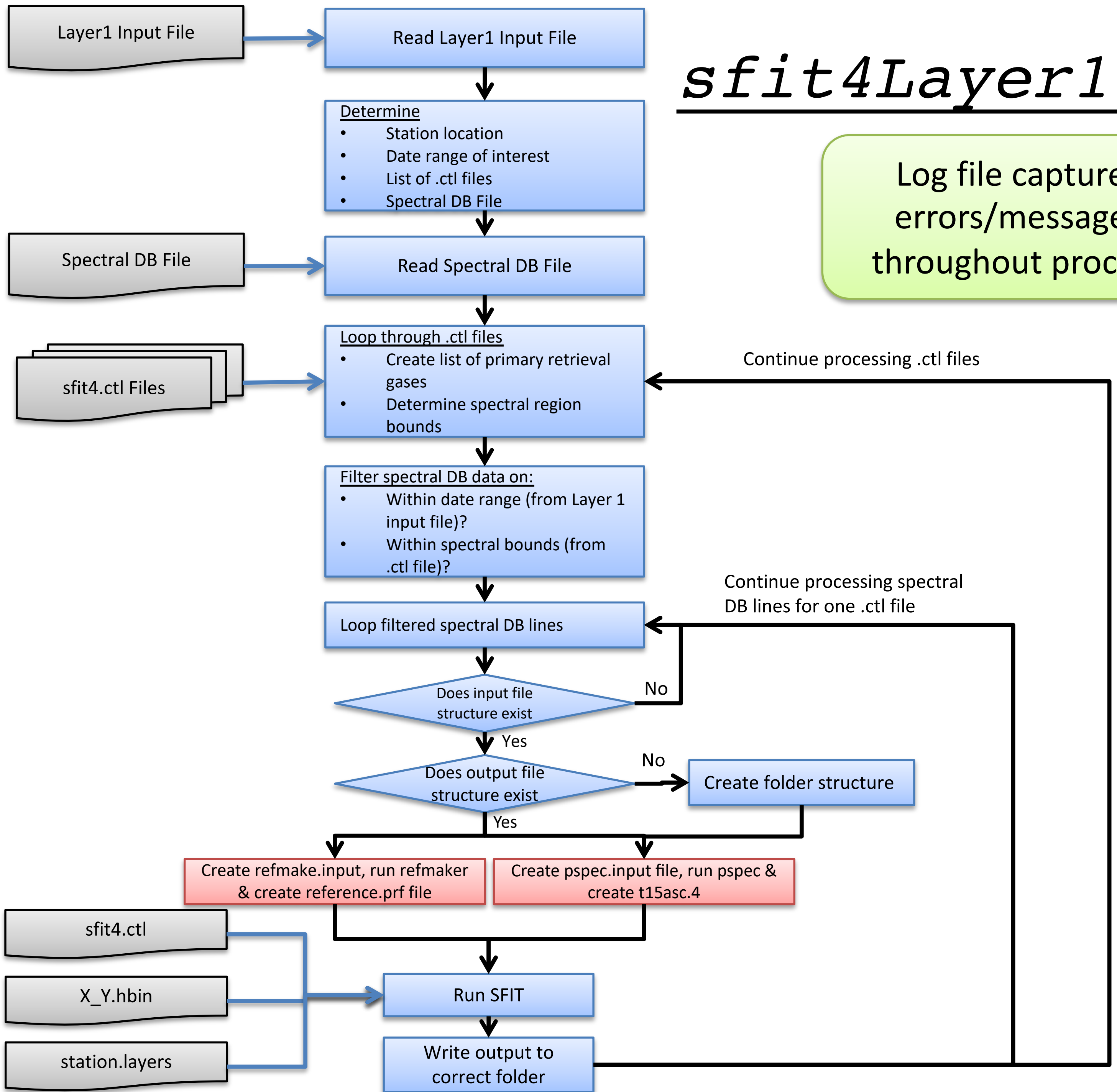
Layer1

The purpose of Layer1 is to batch process multiple or many retrievals.

Layer1 requires an input file to specify retrieval options such as date range, input/output directory, etc.

The layer one processing environment serves to do the following:

- Create a directory structure to organize the output data
- Generate the necessary input files to run SFIT core code
- Execute the SFIT core code
- Conduct error analysis on output


Layer1 Input File
Read Layer1 Input File
Determine
Station location
Date range of interest
List of .ctl files
Spectral DB File
Spectral DB File
Read Spectral DB File
sfit4.ctl Files
Loop through .ctl files
Create list of primary retrieval gases
Determine spectral region bounds
Continue processing .ctl files
Filter spectral DB data on:
Within date range (from Layer 1 input file)?
Within spectral bounds (from .ctl file)?
Continue processing spectral DB lines for one .ctl file
Loop filtered spectral DB lines
Does input file structure exist
No
Yes
Does output file structure exist
No
Create folder structure
Yes
Create refmake.input, run refmaker & create reference.prf file
Create pspec.input file, run pspec & create t15asc.4
sfit4.ctl
X_Y.hbin
station.layers
Run SFIT
Write output to correct folder

sfit4Layer1.py

Log file captures errors/messages throughout process

>> sfit4Layer1.py -?

```
sfit4Layer1.py -i <file> -l -L0 -P <int> -d <20190101_20191231> -?
  -i <file>                              : Flag to specify input file for Layer 1 processing.       <file> is full path and 
filename of input file
  -l                                     : Flag to create log files of processing. Path to write log files is specified in input 
file
  -L <0/1>                               : Flag to create output list file. Path to write list files is specified in input file
  -P <int>                               : Pause run starting at run number <int>. <int> is an integer to start processing at
  -d <20190101> or <20190101_20191231>   : Date or Date range.
                                          -d is optional and if used these dates will overwrite dates in input file for Layer 1 
processing

  -?                                     : Show all flags
```

Tip: >> `sfit4Layer1.py -i input.py –P1`

Will create all needed files to test/debug with Layer 0.

How does input layer 1 look?

```
#------------------------------------------------------------------------------------------
# Name:
#       TAB_CO_input.cpy
#
# Purpose:
#       This is the main input file for sfit4Layer1 processing. Contains directories, flags,
#       etc for processing Layer 1.

#---------
# Location
#---------
loc = 'tab'

#-----------------------------
# Date Range of data to process
#-----------------------------
# Starting
iyear   = 2018            # Year
imnth = 5                 # Month
iday     = 2              # Day

# Ending
fyear    = 2018           # Year
fmnth  = 12               # Month
fday    = 31              # Day

#------------
# directories
#------------
BaseDirInput     = '/data1/'                                          # Input base directory
BaseDirOutput    = '/data1/ebaumer/tab/co/'                           # Output base directory
binDir          = '/data/ebaumer/Code/sfit-core-code/src/'     # binary directory
#ilsDir          = '/data/Campaign/TAB/ilsFiles/ils/lft11/'        # ILS file(s). Options:
ilsDir          = ''        # ILS file(s). Options:
                                         #   1) Use empty string ('') to indicate no ILS file!!
                                         #   2) If string points to directory finds ils file closest in date (ils file name must be in format: *ilsYYYYMMDD.*)
                                         #   3) If string points to specific file, this ils file is used for all data processing

#RatioDir        = '/Users/ebaumer/Data/TestBed/fltrFiles/'       # Directory for ratio files ** Currently NOT used **
logDirOutput     = '/data1/ebaumer/tab/co/'                           # Directory to write log files and list files
```

```
ctlList   = [['/data1/ebaumer/tab/co/x.co/sfit4_v3.ctl','4','Current_B3'], ['/data1/ebaumer/tab/co/x.co/sfit4_v3.ctl','5','Current_B3']]  #Filter 4 and 5

spcdbFile = '/data/Campaign/TAB/Spectral_DB/HRspDB_tab_2015_2018.dat'     # Spectral DB File

WACCMfile   = '/data/Campaign/TAB/waccm/WACCMref_V6.TAB'                     # WACCM profile to use
WACCMfolder = '/data/Campaign/TAB/waccm/co/'                                 # WACCM folder with monthly profiles
```

~~sbCtlFile = '/data1/ebaumer/tab/co/x.co/sb_b3.ctl'   # Control file for error analysis~~

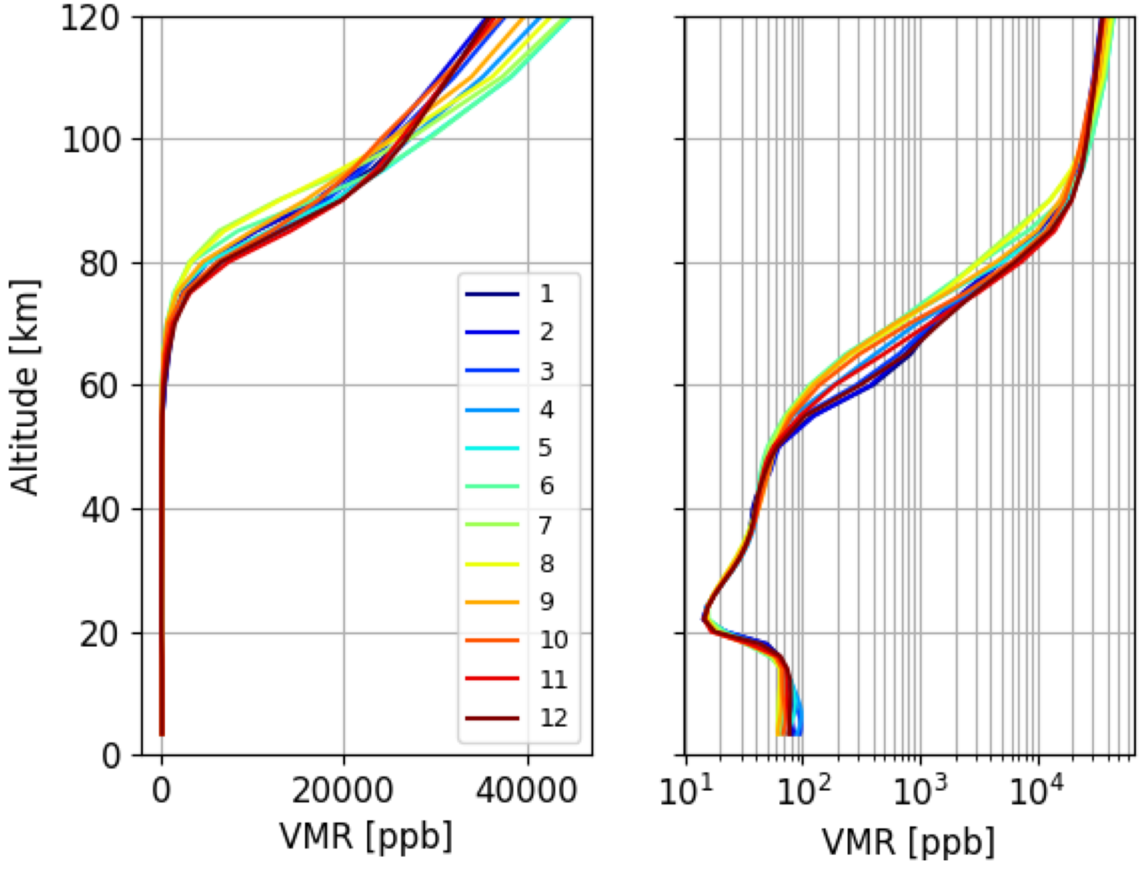
```
#--------------------
# Flags and Constants
#--------------------
waccmFlg    = 1                          # Flag to use WACCM profiles: 0 = Use single WACCM file defined above (WACCMfile)
                                         #                             1 = Use Monthly mean WACCM profiles in the folder defined above (WACCMfolder)

coaddFlg    = 0                          # Flag to indicate processing coadded spectra

ilsFlg      = 1                        # ILS file flag: 1 = Use ils file/directory specified in ilsDir string
                                     #              0 = No ils is specified in input file. What is specified in ctl file is used

scnFlg      = 0                        # Flag to use measurement files with only forward or only backward scans
                                     # 0 = Flag off - does not distinguish between forward and backward scans
                                     # 1 = Only use files with FOWARD scans
                                     # 2 = Only use files with BACKWARD scans

pspecFlg    = 1                          # 1 = run pspec,    0 = do not run pspec
refmkrFlg   = 1                          # 1 = run refmaker, 0 = do not run refmaker
sfitFlg     = 1                        # 1 = run sfit,     0 = do not run sfit
lstFlg      = 1                        # Flag to create list file. Output file which has meta data and a list of all directories processed
errFlg      = 1                        # 1 = run error analysis, 0 = do not run error analysis
zptFlg      = 1                        # 1 = Use new ZPT.nmc files, 0 = use old zpt-120 files

refMkrLvl   = 0                          # Version of reference maker to use.
                                     #    0 = Use pre-existing zpt file. Concatonate with water and WACCM profiles
                                     #    1 = Use pre-existing zpt file. Concatonate with water and WACCM profiles. Replace
                                     #       surface pressure and temperature with values in database file. If those values
                                     #       are not present, then default to original zpt file
```

```
wVer       = 99                                    # Version of water profile to use.
                                              #   <0 => Get the latest water version file
                                              #   >=0 => Get user specified water version file. Latest file is taken if unable to find user specified
#------------------
# Pspec input flags
#------------------
nBNRfiles = 1                                      # Number of BNR files to include in pspec input

outFlg    = 1                                      # Pspec output flag
                                              #    1 = output t15asc file (ascii)
                                              #    2 = output bnr file (binary)
                                              #    3 = output binary and ascii file

verbFlg   = 2                                      # Pspec verbosity output flag
                                              #    0 = no stdout from baseline correction or zero bnr or block output for plotting
                                              #    1 = stdout from bc and zeroed bnr but no blockout
                                              #    2 = stdout from zeroed bnr and blockout for plotting

nterpFlg  = 1                                      # nterp -  zero fill factor
                                              #    = 0 - skip resample & resolution degradation (regardless of sfit4.ctl:band.n.max_opd value)
                                              #    = 1 - minimally sample at opdmax
                                              #    > 1 - interpolate nterp-1 points upon minimal sampled spacing
                                              #        note: OPD is taken from sfit4.ctl:band.n.max_opd value

ratioFlg  = 0                                      # rflag - ratio flag, to ratio the spectra with another low resolution spectral file (eg spectral envelope)
                                              #    = 0 - no ratio
                                              #    = 1 - ratio, file is a bnr of same type as fflag below, expected to be resolution of ~10cm-1

fileFlg   = 0                                      # fflag - file open flag
                                              #    = 0 for fortran unformatted file
                                              #    = 1 for open as steam or binary or c-type file (gfortran uses stream)

zFlg      = 0                                      # zflag - zero offset
                                              #    = 0 no zero offset,
                                              #    = 1 try w/ baselincorrect,
                                              #    0 < z < 1 use this value,
                                              #    = 2 use optimized 2nd polynomial fit to fully absorbed regions in 10m region

#---------------------------------------------
# filter bands and regions for calculating SNR
# These values are used in creating the pspec
# input file. Edit at your own risk
#---------------------------------------------
fltrBndInputs = "1 \n\
f4  2300.000 2301.000 \n"
```

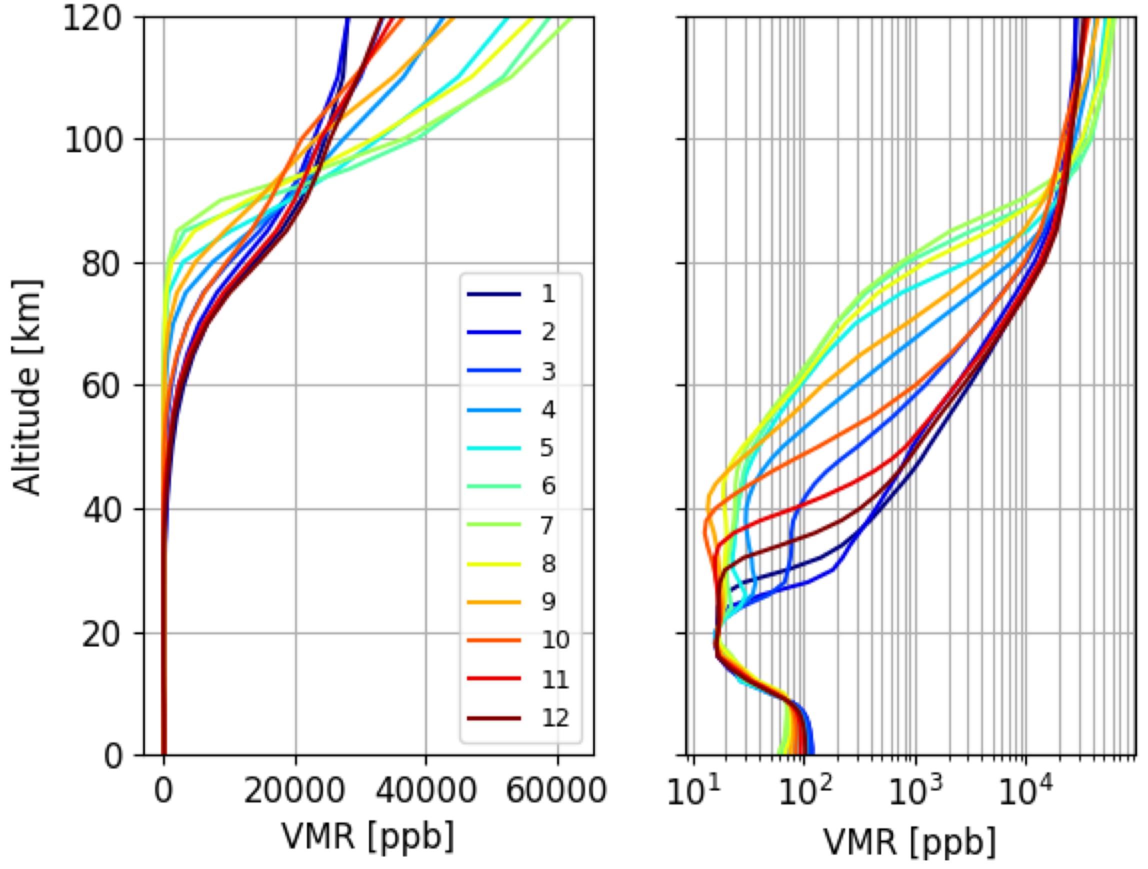
Do we need to include WACCM monthly profiles?

WACC CO monthly profiles

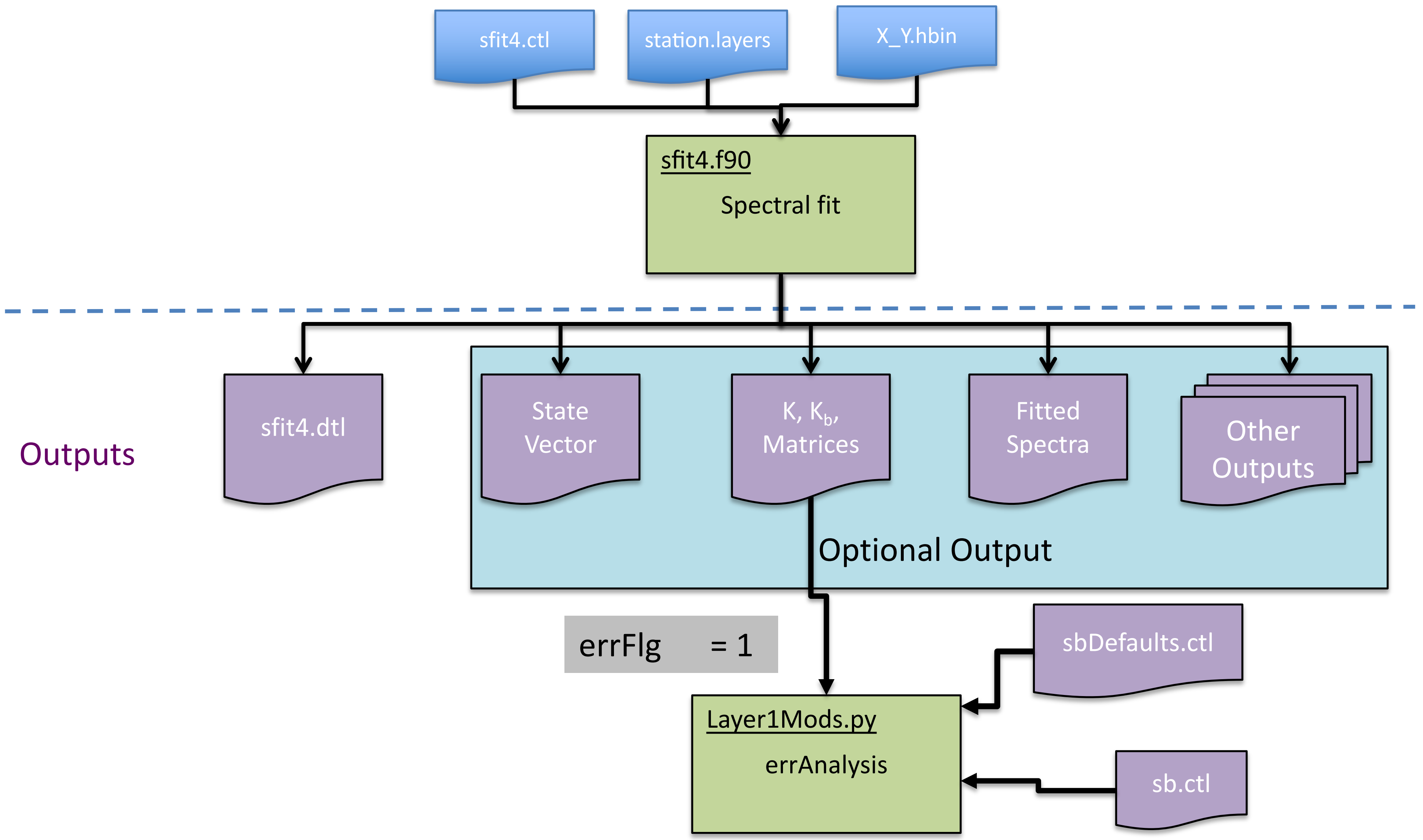
MLO (19 N)



Thule (76 N)



Error analysis in `sfit4Layer1.py`



Layer1Mods -- contains various modules used by sfit layer 1 processing. These modules include refMaker, t15ascPrep, and error analysis.

Error analysis in `sfit4Layer1.py`

What has it changed in the latest **sfit4 - pre/post processing python package distribution?**

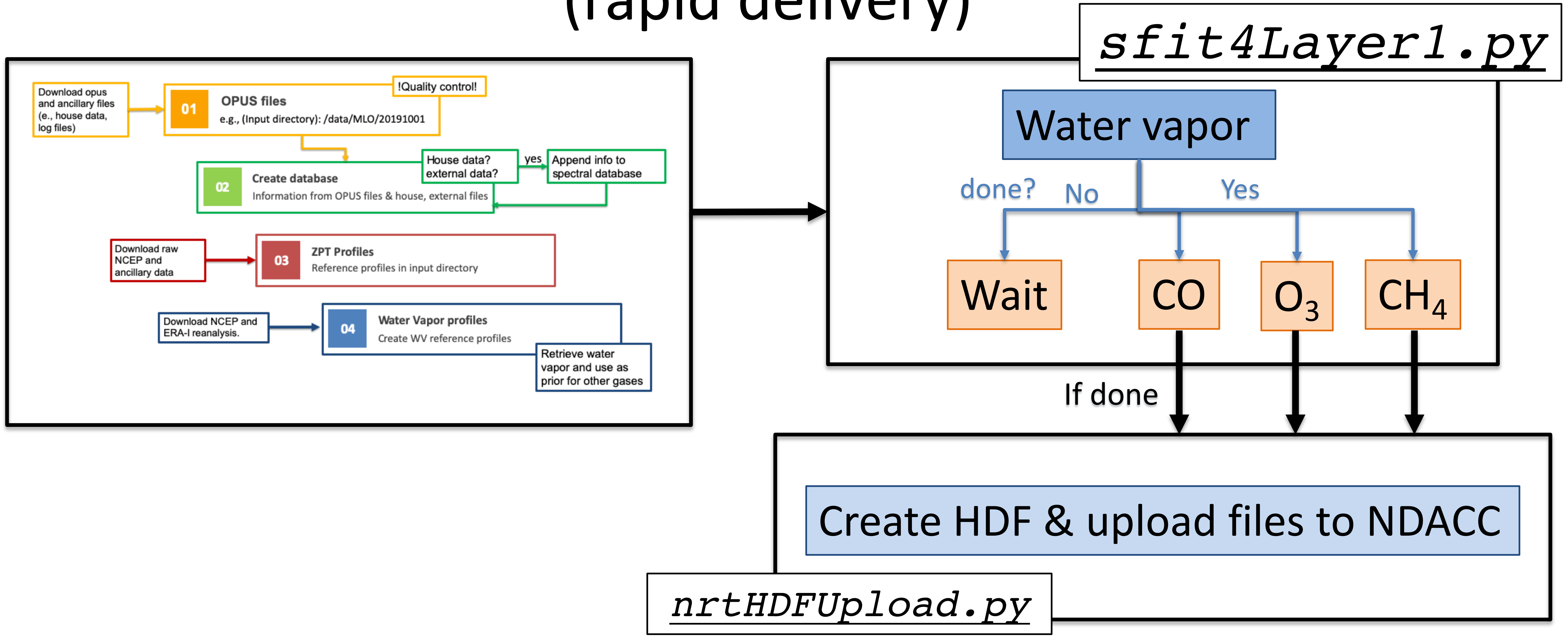
If running error analysis through Layer 1, the errFlg flags needs to be chose inside the input layer 1 file. Additionally, the Kb needs to be True in the sfit4 control file.

(1) The traditional single sb.ctl for each gas is not implemented. Instead, a single sb control file is used for all gases.

(2) For a harmonized IRWG error calculation, in particular spectroscopy uncertainties, there is a default control file that the sfit4 development team has been created. We suggets to use this file.
this file can be found in the Layer1 folder (called sbDefaults.ctl)

(3) To run error calculation the path to this file needs to be defined in the sfit4.ctl file as "file.in.sbdflt".

A few notes about near-real time analysis (rapid delivery)


Download opus and ancillary files (e., house data, log files)
01
OPUS files
e.g., (Input directory): /data/MLO/20191001
!Quality control!
02
Create database
Information from OPUS files & house, external files
House data? external data?
yes
Append info to spectral database
Download raw NCEP and ancillary data
03
ZPT Profiles
Reference profiles in input directory
Download NCEP and ERA-I reanalysis.
04
Water Vapor profiles
Create WV reference profiles
Retrieve water vapor and use as prior for other gases
sfit4Layer1.py
Water vapor
done?
No
Yes
Wait
CO
O_3
CH_4
If done
Create HDF & upload files to NDACC
nrtHDFUpload.py

The whole process is run in a shell and most processes are carried out using a screen - software program that can be used to multiplexes a physical console between several processes.

Final remarks

- Goal: put together a document that outlines the recommended retrieval processing.

- If you have feedback/recommendations let us know.