# Harmonisation Framework Configuration File Parameters

## job\_id

This is just a two digit number of to separately identify jobs

## dataset\_dir

This is the path of the directory that contains the match-up data to harmonise

## parameter\_path

This is the path of a comma delimited csv file containing the input harmonisation parameters for the sensors (and only these sensors) is the dataset above. This is the parameters used to start the solver and also as the reference in the plotting.

Each sensors gets a new line and the ordering required for this is a bit funny:

The sensors rows should be in the order the sensors first appear in the pairs as they are listed in the directory. For example the AVHRR dataset you should them ordered as, n15, n16, n17, n18, n19, n08, n07, n09, n10, n11, n12, n14. The file for AVHRR\_RSIM\_3 is v3/Data/parameters\_AVHRR\_RSIM\_3.csv.

## output\_dir

Directory to save the output files in

## data\_reader\_path

This is the path of the input harmonisation reader required for the dataset. In an ideal world this wouldn’t have to be specified as it could always be harm\_data\_reader.py

For the AVHRR data we need a special case files:

* harm\_data\_reader\_AVHRR\_4.py – for time-dependent AVHRR data, capability to deal with additional variables in input files for the averaging uncertainties
* harm\_data\_reader\_AVHRR\_3.py – for time independent AVHRR data, capability to deal with additional variables in input files for the averaging uncertainties and to ignore columns 4 and 9 of data arrays
* harm\_data\_reader\_AVHRR\_3\_sample.py – same as harm\_data\_reader\_AVHRR\_3.py but only reader the first 1000 match-ups for quick testing

Hopefully we’ll move towards a solution where special case readers aren’t required eventually.

## sensor\_funcitons\_path

Path to my sensor functions python file.

For the AVHRR again we have:

* sensor\_functions\_AVHRR\_4.py – for time-dependent AVHRR data
* sensor\_functions\_AVHRR\_3.py – for time independent AVHRR data

In general (and presumably soon for MW and HIRS) the sensor functions file should contain two functions as follows:

def sensor\_model(a, X):

# Code

Return R, J

def adjustment\_model(R):

# More code

Return B, J

Where a are the sensor harmonisation parameters, X are the sensor state variables (annoyingly this is currently in rows, rather than columns as for H), R is the radiance and B is the adjusted radiance for K = B\_i – B\_j (which currently does nothing).