Objectives

A proposal for a new ObsSpace data model

A proposal for a new ObsSpace data model

Contents

- Motivation/objective
- Lack of flexibility
  - Two examples
- Limited benefit of the multi-dimensionality
- Proposal: moving toward a flat and “unstructured” data model

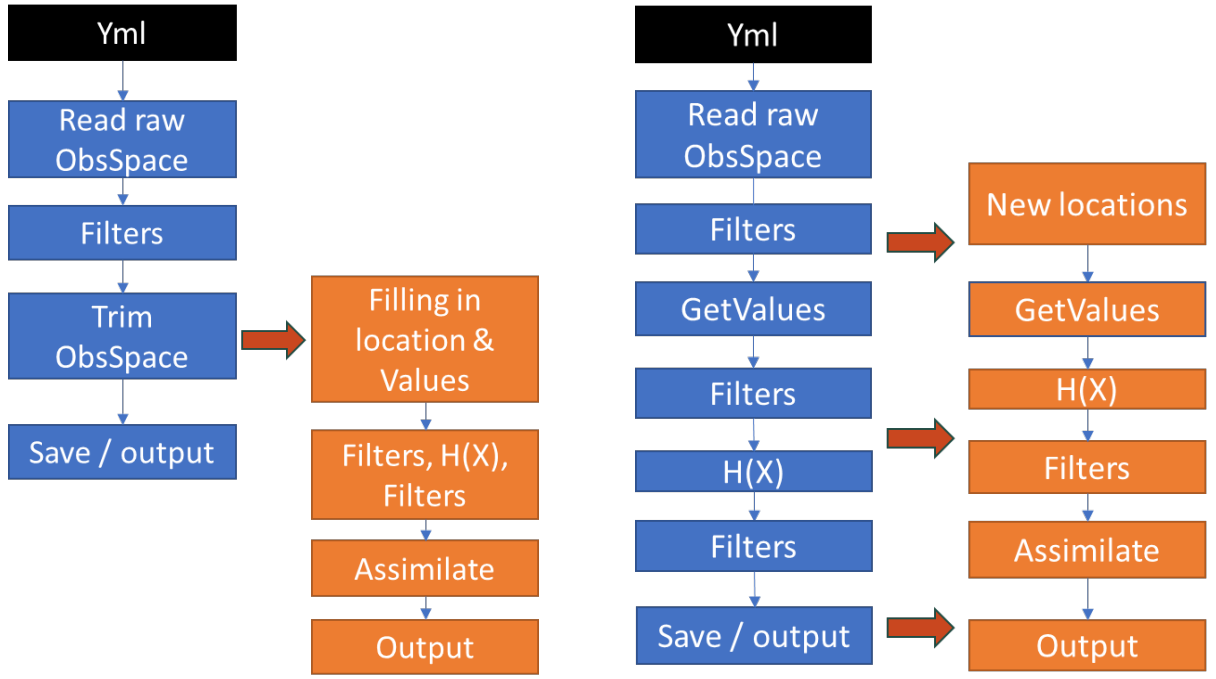
Motivation/objective

To enable more complex observation processing, we (Yannick, Stephen H. and I) have explored on many occasions how we could improve the functionality of the ObsSpace within UFO.

We can up with a reasonable wish list such are slicing, reshaping, etc.

However, during this exploration we found that the current definition of the obsSpace is an obstacle.

Example of more complex observation processing



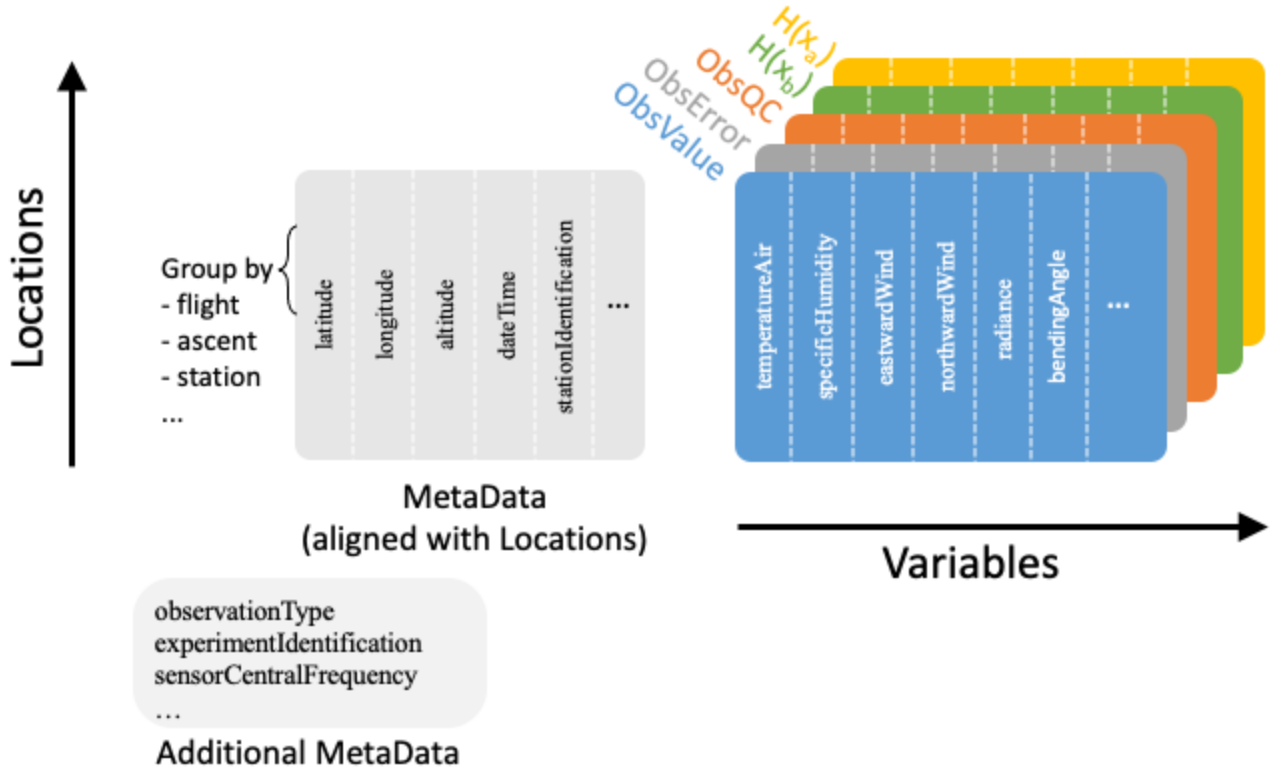
Simple reduction of obsSpace
Keep same locations

More complex exchange between obsSpaces
Different locations

ObsSpace

Basic principles of the obsSpace

- Based on netcdf/HDF5 (in memory)
- One compulsory dimension called ‘nloc’ that is associated to the observation location.
- One optional second dimension nChannel (e.g. nloc x nChannel)
- The observation value (and metadata) can map on to this one-dimension vector or remap onto a 2D matrix as it is currently done for radiances.



Lack of flexibility

The current ObsSpace structure can look attractive at first (especially for radiances) but is ultimately a limiting factor in development of UFO.

The main blocker is related to the lack of flexibility – It is difficult to extend or shrink at will.

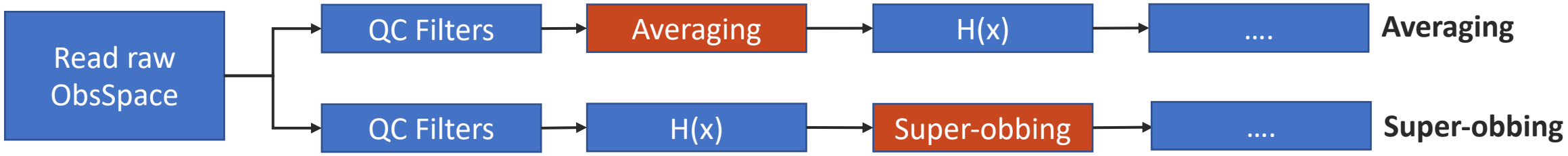
This is a problem - for example:

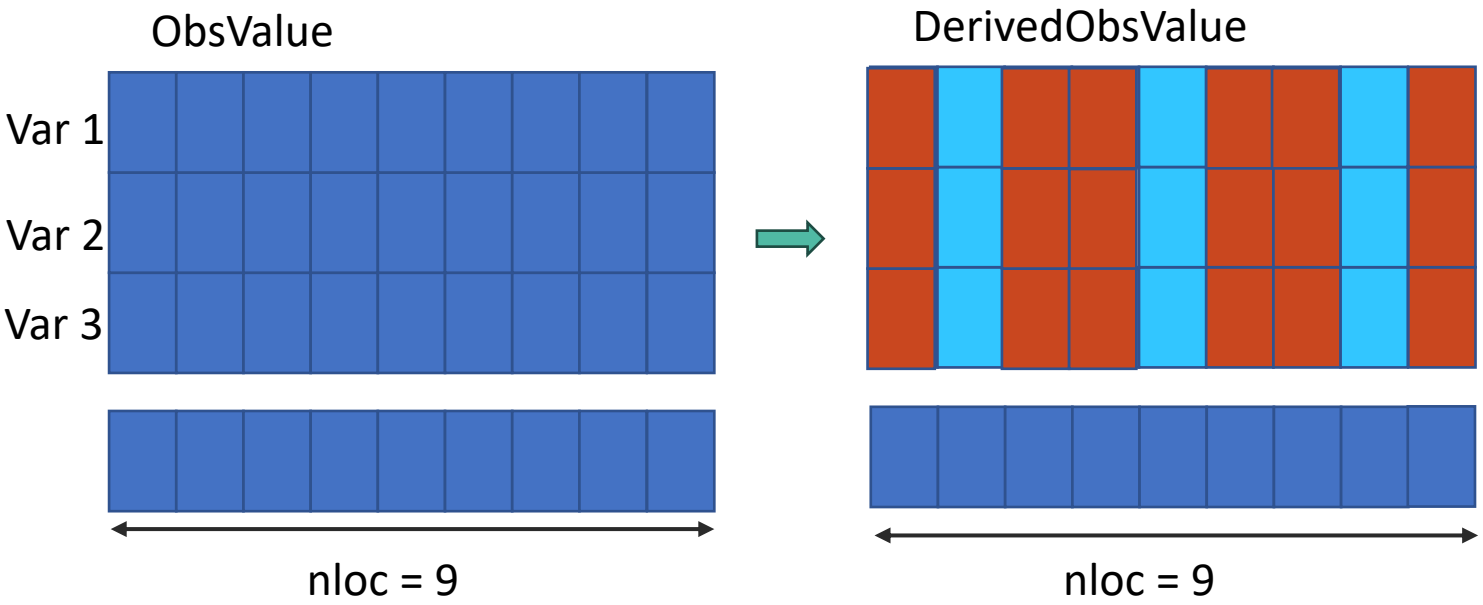
- averaging or super-obbing (see example)
- model-level averaging: extend obsSpace from *nloc* to *nloc*+ (*nb level***nb profile*)
- redrawing GeoVals

See the following examples

Lack of flexibility : Example 1

Super-obbing / Averaging





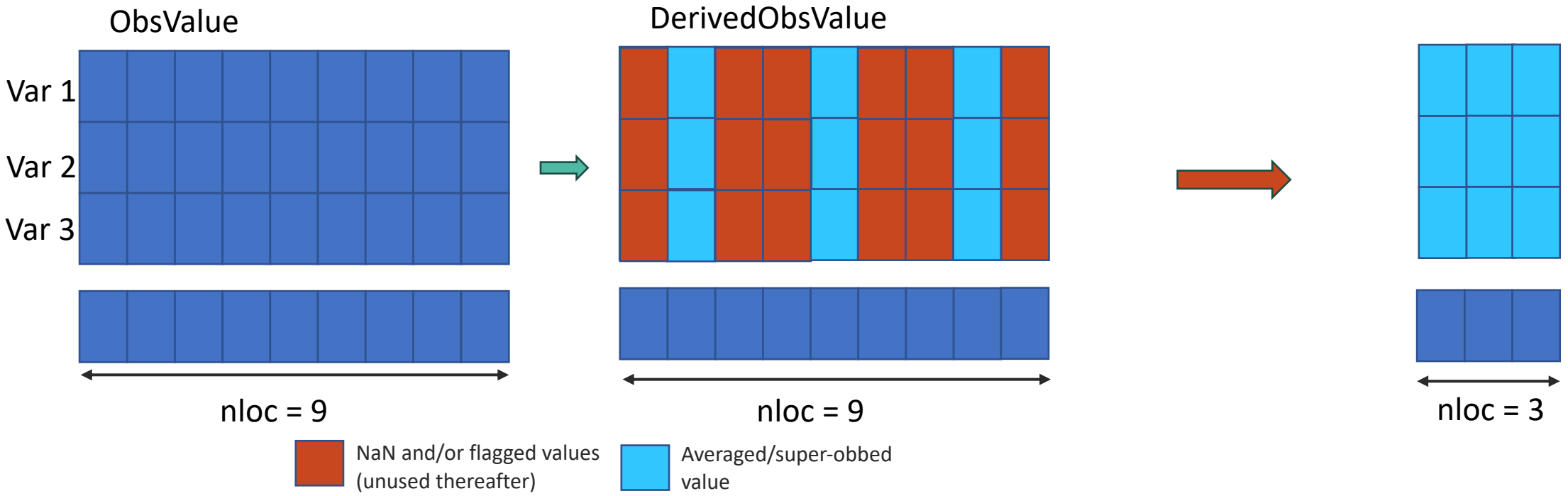
NaN and/or flagged values (unused thereafter)

Averaged/super-obbed value

- **This reduces observation count**
  - Yet ObsSpace still has the same size
- **Once averaging (surper-obbing) has occurred, we are not interested in the original data.**
- **Use existing GeoVals**

Lack of flexibility : Example 1

Super-obbing / Averaging



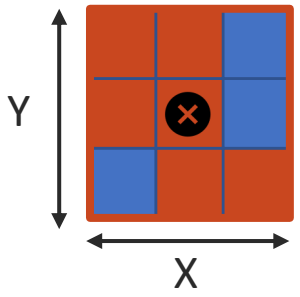
We should be able to:

- Create a new reduced obsSpace (free memory) to carry-on processing/DA
- Save (if necessary for monitoring) and destroy the original obsSpace

Lack of flexibility : Example 2

Super-obbing / Averaging

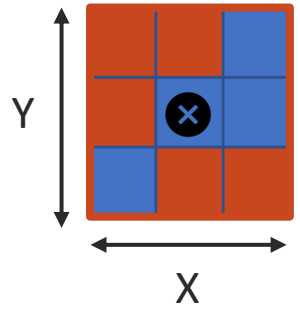
3x3 averaging box



One observation as the same location as the center (location) of the averaging box

→ The current ObsSpace enable us to get the right GeoVal for this averaging box

3x3 averaging box



No observation exist at the center (location) of the averaging box

→ The current ObsSpace does not enable us to get the right GeoVal for this averaging box
(Super-obbing: average of innovation added to the box centre GeoVal)

We should be able to:
- add a location (center of the box) and redraw the Geovals

Valid observation

No observation present or un-valid observation

Location of the resulting average

Limited benefit of the multi-dimensionality

The multi-dimensionality of the current data model has very limited benefit.

- Observation are not fully gridded dataset, and even in the most suitable cases a nloc x m structure require some adjustment such as array padding.
  - One example is the scatterometer. Not all the observations have always exactly the same number of solutions. Another is the radiosonde profile where each provide has different number of observations.
- The 2d structure also assume that the same shape and size can be applied to all the elements associated to the observation – This is often not the case (e.g. derived products from RTTOV/CRTM).
- Currently in a nloc x m structure, the second dimension is being predefined as nChannel (channel number for radiance) which make no sense outside radiances.

Proposal

Moving to a flat and "*unstructured*" representation similar to Python (pandas) DataFrames

Functionality and complexity

- Any data distribution/structure fit into a DataFrame
- DataFrame are a lot easier to *slice*/*expand*/*shrink* than netcdf/HDF5 structure.
  - The ObsSpace will be a lot more flexible and will enable more complex manipulation such as slicing and update.
- Code designed to process the ObsSpace may also end-up a lot simpler.
  - This is something what has happen with our generic monitoring when moving to DataFrame data model instead of netcdf.

representation similar to Python (pandas) DataFrames

Proposal

Moving to a flat and "*unstructured*" representation similar to Python (pandas) DataFrames

Memory consideration:

- Memory usage will also be improved (in some situation) with the ability to "resize/slice" the obsSpace.
  - After an averaging we may want to resize the obsSpace and only keep the averaged observation for the rest of the processing.
  - Remove from the obsSpace all observation that have been thinned.
- For a given set of observation the memory usage of the obsSpace based on dataframe compared to the one based on netcdf will be function of how the dataframe is organised, but there is no reason to believe that the dataframe approach will be more memory hungry.

DataFrame library

- We have explored few C++ library (Xframe / DataFrame / Apache Arrow / Atlas)
  - Is it still maintained? / Maturity?
  - How flexible is it with slicing, appending and updating values
  - Generality
  - How easy is it to morph the structure?
  - OpenMP / MPI capability

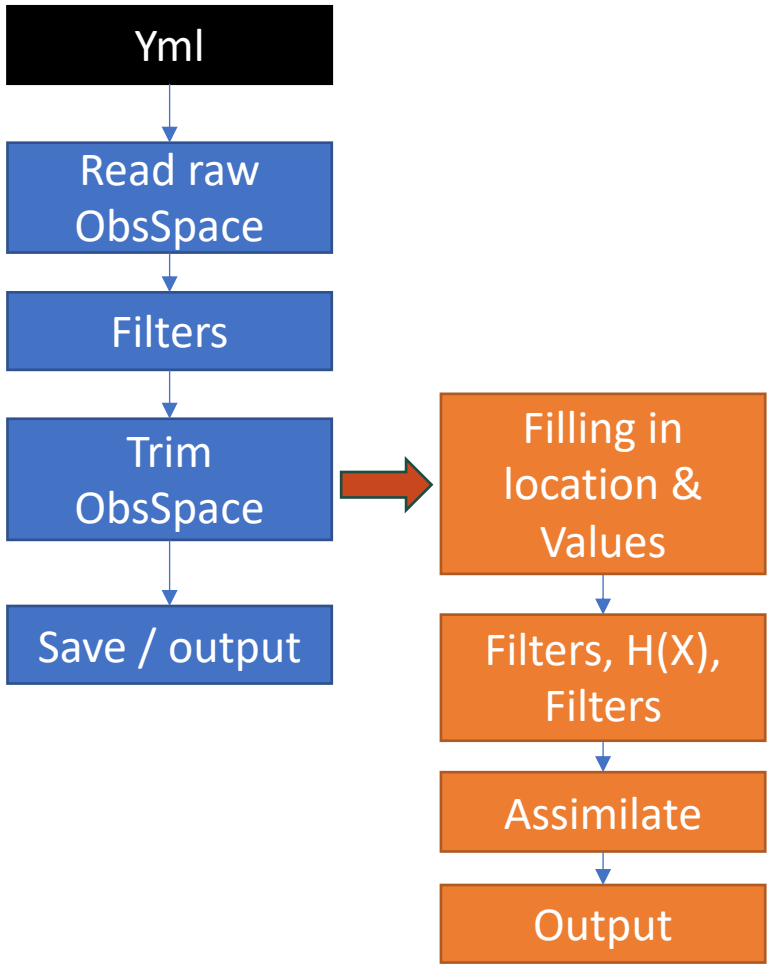
Overall, we found the *DataFrame* library to be the most hopeful.

- It was intuitive to use, and it seems rather extensible
- The documentation is very easy to follow
  - Each function has some example usage to go along with it.
- The library was clearly made with usability in mind.
- There is only a single leading developer (However he is open to update).

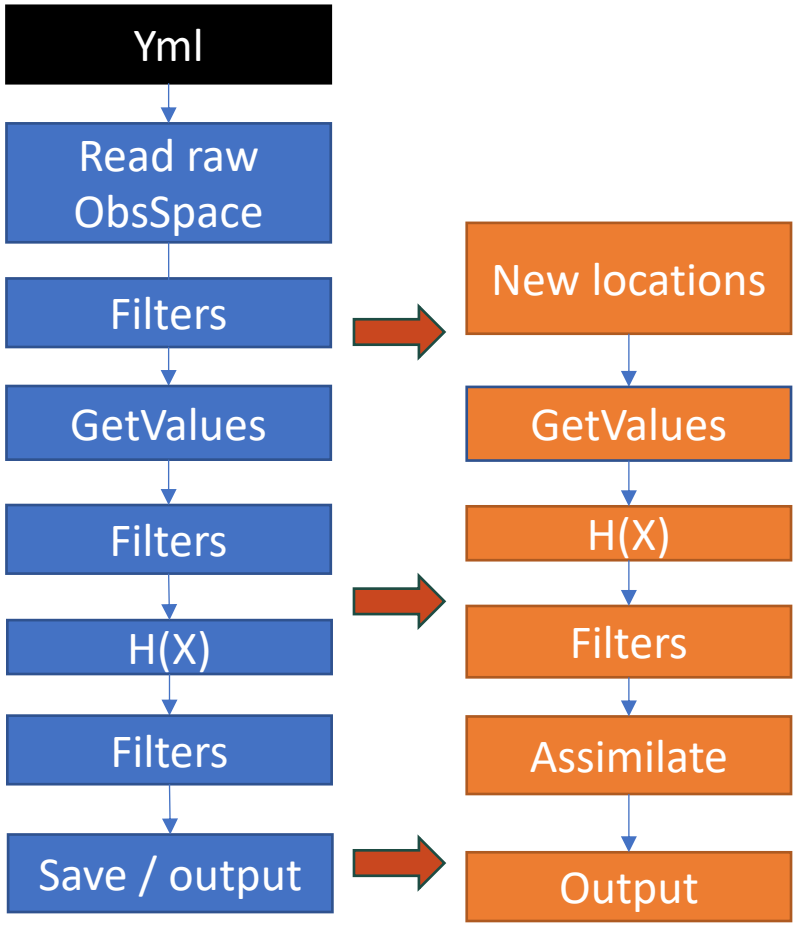
The Met Office will offer the resources for this work with a planned secondment to JCSDA.

Extra Slides

ObsSpace



Simple reduction of obsSpace
Keep locations



More complex exchange between two obsSpaces
Different locations