Data processing for the RAVen project.

The data processing of the HDF5 files for the full domain includes following steps.

1. The files with data of the Jabbeke radar (scan 2) are combined in one hdf5 file. The final file includes the reflectivity (DBZH), the radial velocity (VRAD) and the specific width of the radial velocity (WRAD). In case of dual-pol data analysis the Correlation coefficient (RHOHV) is also included.
   For the on- and off-shore part of the domain specification the country borders are used. For the on-shore data the part of the data outside of land polygons were set to NaNs (255 in HDF data). For the off-shore data the part of the data over the land polygons were set to NaNs. These first part of processing is done by **ohfa.py** module. The full description of this module can be found below in the text.

2. The hdf5 files, generated as described above, are used in the bird detection algorithm Vol2Bird (https://github.com/adokter/vol2bird) described in Dokter et al., 2010. The 0.3.8 version of this algorithm was installed locally. The algorithm parameters can be found in **options.conf** file.

   For the RAVen project the algorithm was run with 5 km minimum range and 35 km maximum range. Standard operational bird cross section was used (11 cm^2). The heights between 0 and 4 km AMSL were analyzed and the volume data were assigned to 20 altitude layers having resolution of 200 m.

   The bird detection algorithm removes precipitation cells and estimates the bird density based on the remaining reflectivity and the radial velocity data (Dokter et al., 2010). The Vol2Bird processing of the radar data itself also consists of several steps.

   1) The precipitation cells are detected based on combined criterion using reflectivity and radial velocity information. Contiguous reflectivity areas with a high reflectivity factor (above 15 dBZ) and a low average nearest neighbor variance (σ_{cell}) in radial velocity data (less than 5 m s^{-1}) are assigned as precipitation cells. The low σ_{cell} variance is mostly observed for wind-borne scatters (such as hydrometeors or insects). Furthermore, insect contamination is mainly excluded from this analysis because it considers only nighttime migration.

   2) The volume data of multiple elevations are assigned to altitude intervals or layers.

   3) The average reflectivity is calculated for each altitude layer with a relatively high standard deviation in radial velocity data. The speed and flight direction for each altitude layer are estimated by fitting the clean velocity data to a constant velocity model (see equation A 1 in Dokter et al., 2010).

   The output of the Vol2Bird algorithm includes vertical profiles with among others horizontal speed (ff), direction (dd) speed component west to east(u) and speed component north to south (v).

3. The third part of the total processing uses the output of the Vol2Bird algorithm. The calculation of Migration Traffic Rate (MTR) is done by **mtr_vbird.py**. Based on the bird profiles we calculate the MTR's, mean speed and mean direction for the vertical intervals [200,800] m AMSL, [800,1400] m AMSL and a full interval [200,1400] m AMSL.

In the MTR's and mean speed values the mean for this scan is calculated first and after that the mean over all scans of particular hour are combined in an hourly mean value valid for this vertical interval. The mean direction is calculated by first taking the mean vectorial direction per altitude level of 200m based on u and v components from all scan performed during this hour. This is done, because the flight directions can change drastically from one altitude to another and with the bird's migration we expect less alternations in time. After the calculation of the mead u and v per hour, we calculate mean u and v per vertical interval and based on them calculate mean direction.

The same steps can be used for the on-shore and off-shore parts, but before the whole data processing starts, the proper land-mask should be generated by **on_off_shore.py**

The module generates land-mask for an input file (InputPathToHDF5File) based on the polygon from the shape file (-s option). The shape file should include a closed boundaries polygon. The output HDF5 file will be written to the output directory (OutputDirectory).

Input: [options] InputPathToHDF5File OutputDirectory

-v verbose,

-d debug,

-s shapefile (default ./data/bejab/bejab_poly_valid),

-c radar-code (default bejab)

Output: HDF5 file with the (0/1) mask and the same number of datasets as the input file.

This module requires some additional python libraries:

wradlib, cartopy and *shapely*.

An example of the call:

```
python on_off_shore.py –s ./data/bejab/bejab_poly_valid -v -c newbejab ./2016091423550500dBZ.pvol.h5 ./
```

Data processing steps in short

1. Prepare the merged HDF5 files

   1.1. The HDF5 files need to be merged by **ohfa.py** module to have all variables in one file and with the right mask (on- or off-shore, or without any mask for the full domain)

   The module **ohfa.py** is an Odim Hdf5 file Aggregator, which aggregates two or more Odim HDF5 files into a single one. It was programmed by Christophe Ferauge (2014) and modified for RAVen project by Maryna Lukach (2017).
   The modified module **ohfa.py** does the merging and applies the land mask if it is needed. The -o and -c options are added and masking of on- or off- shore parts of the data is done based on the

mask provided in -c option. In the processing the output was placed in onshore, offshore or full subdirectories of h5.merged directory of the main directory.

The call of the **ohfa.py** module:

ohfa.py

with correct parameters and input files.

Input: volume HDF5 files (dBZ, V, W) and the directory for the output

-v - verbose

-d - debug

-o – offshore (store True)

-c – coastline (string) the full path and the name of the land-mask generated based on the coastlines.

Output: one HDF5 file with all data placed in separate datasets will be written to the provided output directory

An example of the call:

```
python ohfa.py -v -d -o -c . /bejab_landcoast_validmask.hdf
./bin/2016101401450500dBZ.vol.h5   ./bin/2016101401450500V.vol.h5
./bin/2016101401450500W.vol.h5  ./h5.merged/offshore/
```

The run will combine the dBZ, V and W data from the input files in one HDF5 file, apply the mask and write the output file in the *./h5.merged/offshore/* directory.

2. Run the Vol2Bird algorithm over merged hdf5 files.

   The algorithm takes as input all the files from the directory provided by *–i* parameter. The output will be placed in the directory defined by *–o* parameter. Additional *–t* parameter gives a possibility to run the module only for the files in *–i* directory that start with a given YYYYmmDD (applicable only for the files having the timestamp at the beginning of their name).

   In the processing the outputs of the Vol2Bird algorithm were placed in full_vp, onshore_vp or offshore_vp subdirectories of directory h5.merged of the main directory.

   The processing for RAVen was done with the off- or on-shore part set to "nodata" and with the intern Vol2Bird parameter REQUIRE_VRAD set to True. (used options from the option.comfig file can be found in the apendix)

   The attribute "nodata" in OPERA data model is defined as "The raw value used to denote areas void of data (never radiated)". The parameter REQUIRE_VRAD makes Vol2Bird require from a range gate to have a valid radial velocity to contribute to the calculation.

With these settings, the total number of points (on- + off- shore), used for calculation of the bird reflectivity for each altitude level, equals the number of points used for the calculation on the whole domain.

This number of points can be found in "/dataset1/ data6" group of the Vol2Bird output.

The call of the ***vol2bird.py*** module with correct parameters and input files:

python ***vol2bird.py***

Input: all input files in the directory provided in –i parameter

–i - the input directory

–o – the output directory

–t - (YYYYmmDD format) - gives a possibility to run the module only for the files that start with a given YYYYmmDD (applicable only for the files having the timestamp at the beginning of their name).

Output: bird's vertical profiles

An example of the call:

python ***vol2bird.py*** *-i ./h5.merged/onshore -o ./h5.merged/onshore_vp -t 201609*

The algorithm will proceed through the onshore data of September 2016.

3. Calculate MTR's for vertical profiles from vol2bird algorithm

The module **mtr_vbird.py** calculates "specific MTRs " – MTRs per altitude level for 20 levels between 200m and 4000m a.m.s.l. The MTR per level is calculated as density*speed*3.6/5, where for onshore and offshore domains the speed is taken from the full domain and the density is calculated from the birds reflectivity values. These values are non-zero for all three datasets (full, on-shore, off-shore) and the density calculation is based on the same assumption of 11 cm^2 bird cross-section.

The MTR [birds/ km/ hour] is calculated from the volume density [bids/km^3], where the volume density is calculated the same way as before.
It is the bird reflectivity (eta [cm^2/km^3]) output of the vol2bird algorithm ("/dataset1/ data7") divided by the bird cross-section 11 cm^2.
The speed used for all datasets (on-, off-shore, and full) is the speed from the full domain output.
For the altitude intervals between 200m and 800m, between 800m and 1400m, between 200m and 1400m the total MTR per interval and the mean speed are calculated. Based on these "specific MTRs" all mean MTRs are calculated. The output of the module is saved to the panda's datasets and dumped to csv files.

In the processing the outputs of mtr_vbird algorithm were placed in MTR subdirectory of the main directory. The input for the module should be in onshore_vp, offshore_vp and full_vp subdirectories of the directory h5.merged of the main directory.

The hourly mean values can be found in bejab_vbird_avgmtralllevels_YYYYmm_*_vp.csv files.

*python **mtr_vbird.py***

Input: all vertical profiles in the directory provided in –i parameter

–i - the input directory

–o – the output directory

–t - (YYYYmmDD format) - gives a possibility to run the module only for the files with a given timestamp in the name of the file.

Output: .csv files with calculated means.

This module requires some additional python libraries:

Pandas

An example of the call:

```
python mtr_vbird.py -t 201609 -i  ./h5.merged/offshore_vp/ -o ./MTR/
```

Apendix

Options.conf file:

```
# the number of layers in an altitude profile

NLAYER = 20


# the width/thickness of a layer [m]

HLAYER = 200.0


# the minimum range [m] used for constructing the bird density profile

RANGEMIN = 5000.0


# the maximum range [m] used for constructing the bird density profile

RANGEMAX = 35000.0


# the minimum azimuth [degrees] used for constructing the bird density profile

AZIMMIN = 0.0


# the maximum range [degrees] used for constructing the bird density profile

AZIMMAX = 360.0


# the minimum scan elevation [degrees] used for constructing the bird density profile

ELEVMIN = 0.0


# the maximum scan elevation used for constructing the bird density profile

ELEVMAX = 90.0


# the wavelength [cm] of the radar

RADAR_WAVELENGTH_CM = 5.3
```

whether a static clutter map is used

USE_STATIC_CLUTTER_DATA = FALSE

print options to stderr

PRINT_OPTIONS = FALSE

print dbz to stderr

PRINT_DBZ = FALSE

print vrad to stderr

PRINT_VRAD = FALSE

print cell to stderr

PRINT_CELL = FALSE

print cell properties to stderr

PRINT_CELL_PROP = FALSE

print texture to stderr

PRINT_TEXTURE = FALSE

print clutter to stderr

PRINT_CLUT = FALSE

print profile data to stderr

PRINT_PROFILE = FALSE

whether or not to print the 'points' array

```
PRINT_POINTS_ARRAY = FALSE

# Whether or not to fit a model to the observed vrad
FIT_VRAD = TRUE

# Whether to export bird profile as JSON
EXPORT_BIRD_PROFILE_AS_JSON = FALSE

# Minimum Nyquist velocity
MIN_NYQUIST_VELOCITY = 5

# VVP Radial velocity standard deviation threshold 2.0
STDEV_BIRD = 3.0

# Bird radar cross section [cm^2] (was 11)
SIGMA_BIRD = 11.0

## Maximum mean reflectivity factor for cells containing birds
#DBZCELL = 15.0

## Maximum reflectivity factor for gates containing birds
#DBZMAX = 20.0

## reflectivity quantity to use, one of DBZH, DBZV, TH, TV
#DBZTYPE = DBZH

# for a range gate to contribute it should have a valid radial velocity
REQUIRE_VRAD = TRUE
```

dealias the radial velocities using the torus mapping method by Haase et al.

DEALIAS_VRAD = TRUE

sea bird analysis (new add 17.3.2017)

SEA_BIRD = TRUE

mask datafile (new add 30.3.2017)

MASK_FILE = "mask_file_jab.h5"

whether to use dual-pol moments for filtering meteorological echoes

DUALPOL = FALSE

correlation coefficients higher than this threshold will be classified as precipitation

RHOHVMIN = 0.9