CS+AO Merged Data Product

Derivation of freeboard

1. Purpose

The purpose of this document is to describe the creation and format of the merged CSAO data product. We further described the software created to produce this data product and how it implement in this instance and how it can be adapted for future analysis

2. Existing Baseline D L2 data

The aim of the merged data product is allow for easy comparison and adaptation of existing Baseline-D Level 2 (Bl-D L2) data for further gridded data products and assessment of each.

2.1 Evaluation variables

There are three key processing components performed during the creation of the Bl-D L2 files that have been addressed by the CSAO project.

1. The retracking of the SAR and SARIN power waveforms to calculate the time delay from the satellite to surface
2. The correction of the time delay to take account of atmospheric conditions and fluctuations to surface due to non-static geophysical conditions, including ocean tides
3. The comparison of the satellite derived and corrected surface height to a mean sea surface, to allow for the consideration of surface height anomalies.

We evaluate these three components by comparing the project partner data at the following Bl-D L2 data variables

1. range_1_20_ku
2. The summed atm_corrections_sum_L2: (mod_wet_tropo_cor_to_20_ku , mod_dry_tropo_cor_to_20_ku , iono_cor_gim_to_20_ku , inv_bar_cor_to_20_ku , hf_fluct_total_cor_to_20_ku).
   The summed geo_corrections_sum_L2 (pole_tide_to_20_ku , solid_earth_tide_to_20_ku , load_tide_to_20_ku , ocean_tide_to_20_ku , ocean_tide_eq_to_20_ku).
3. mean_sea_surf_sea_ice_20_ku

Where data is available from a project partner for a particular method we create the following variables

1. range_1_20_ku_PARTNER and range_1_20_ku_ANOM_PARTNER
2. atm_geo_corrections_sum_PARTNER and atm_geo_corrections_sum_ANOM_PARTNER
3. mean_sea_surf_20_ku_PARTNER and mean_sea_surf_ANOM_partner

Where the data_array_ANOM_PARTNER = data_array_PARTNER - data_array_L2

This method allows us to add an ANOM variable for each differing processing method at each point in the Bl-D L2 procession chain and evaluate it's role in final data product. To compare the individual values of range_1_20_ku, atm_geo_corrections_summed and mean_sea_surf, anomalies must be ADDED for comparison.

2.2 Key final data products

Here we describe how to use the merged data product to evaluate each partner processing methods effect upon key observation products. First range_1_20_ku has to be processed to height_1_20_ku with

height_1_20_ku = alt_20_ku - (range_1_20_ku + corrections)

where

corrections = atm_corrections_sum_L2 + geo_corrections_sum_L2

Or

corrections = atm_geo_corrections_sum_(ISat,ISat_Ice,LEGOS_GPOD)

2.2.1 Dynamic Ocean topography

The dynamic ocean topography (DOT) is the part of the ocean surface that we account for due to surface currents. The DOT is evaluated as the difference between observed sea surface elevation and the static level of no-motion that is taken from a geoid. We take this from the default Bl-D L2 data as

DOT = height_1_20_ku - geoid_20_ku.

The measurements of height_20_ku has been corrected for geophysical and atmospheric effects and in this case is from ocean surfaces, therefore open ocean and leads.
As this equation contains the linear combination of two observations all cases of range_ANOM (within leads) and atm_geo_corrections_(ANOM) can be applied. In this case as height_1_20_ku = atl_20_ku - range_1_20_ku, the anomalies need to be SUBTRACTED for comparison. Additionally the various cases of flag_surf_type can be applied to show the role of changing lead classification algorithms.

2.2.2 Sea Surface Height Anomaly

The sea surface height anomaly (SSHA) is the difference between the present and long term mean sea surface elevation. This values is taken a physical metric of eddy activity within the ocean surface layer. The SSHA is evaluated as difference between a measurement of sea surface elevation and a given mean_sea_surface at all measurement locations. This measurement is taken over all surfaces for further sea ice freeboard evaluation, and gives a physical oceanography data product for the cases of ocean surface - open ocean and leads. We take this value from Bl-D L2 data as

SSHA = height_1_20_ku - mean_sea_surf_sea_ice_20_ku

The measurements of height_20_ku has been corrected for geophysical and atmospheric effects.
As this equation is a linear combination we can apply all cases of range_ANOM, atm_geo_corrections_ANOM and mean_sea_surf_ANOM for comparison. All these anomalies must be SUBTRACTED for comparison. Choosing between sea ice floe and lead/ ocean from flag_surf_type can give measurements for further freeboard evaluation or implications for a SSHA dataset.

2.2.3 Sea Ice Radar Freeboard

Sea ice Radar Freeboard is given as the difference between the SSHA for sea ice floes, and the SSHA interpolated between leads. This gives the height of the part of sea ice floes that protrude above the ocean surface that are in hydrostatic equilibrium floating upon the

ocean surface. This value is taken as the difference between the SSHA over sea ice floes and the interpolated SSHA taken from leads. These values in Bl-D L2 are

freeboard = ssha_20_ku - ssha_interp_20_ku.

Here ssha_20_ku is applied ONLY over sea ice floes

Due to the interpolation of SSHA between leads this measurements is only linear for certain cases, and even then care must be taken.

For the case of range_ANOM, these can be SUBTRACTED from ssha_20_ku over sea ice floes ONLY and evaluate the role of a sea ice floe specific retracker to final radar freeboard data. However in this case the resulting value may not be physically accurate. A constant bias is often applied to retracking methods, and this may well be required for the comparison value in range_ANOM. flag_surf_type type variations can also be applied here, making sure care is taken that the a floe specific retracting method has ben used at that location. For example using a different flag_surf_type for Bl-D L2 ssha_20_ku values can well introduce lead specific range measurments that will give a bias to the resultant freeboard. Such cases should not be evaluated.

For the case of range_ANOM in leads, it is likely that anomalies to SSHA will proceed non-linearly through the interpolation method used and thus evaluation of these measurements at freeboard level is not possible without also evaluating another interpolation method.

It is possible to investigate cases mean_sea_surf_ANOM, however care must be taken. As the interpolation of SSHA is taken from measurements of SSHA, the application of mean_sea_surf_ANOM, may well have non-linear effects through the interpolation process. As fields of mean_sea_surf are typically smooth this comparison may well still have merit. It is however that the same mean_sea_surf_ANOM is applied to both ssha_20_ku and ssha_interp_20_ku. Also note that values mean_sea_surf_ANOM musty be subtracted from ssha_20_ku to investigate.

Similarly values of atm_geo_corrections_ANOM can be applied (though added in this case), again making sure they are subtracted from both ssha_20_ku and ssha_inter_20_ku and being aware of possible non-linear effects of anomalies progressing through the interpolation method.

While it is not possible to accurately investigate range_ANOM within leads in the final freeboard measurement, we can look at the difference between ssha_20_ku and ssha_interp_20_ku, showing how much variation there is between measured SSHA within leads and the desired smooth ssha_interp_20_ku. Taking

freeboard = ssha_20_ku - ssha_interp_20_ku

Within Bl-D L2 classified lead ONLY, we can apply all range_ANOM. atm_geo_corrections_ANOM and mean_sea_surf_ANOM (mean sea surface anomalies to be subtracted) to the values of ssha_20_ku to see what linear effects these make to the smooth interpolated field. We can also apply them to ssha_interp_20_ku, taking care and being aware that we are not considering the non-linear effects these anomalies will make through the interpolation process.

LEGOS GPOD also supply two additional cases of ssha_(interp,smooth)_20_ku_LEGOS_GPOD_(Ice,Lead) that are intended to be used for freeboard measurements. The ‘smooth’ value are those to be used when evaluating freeboard with

freeboard = ssha_smooth_20_ku_LEGOS_GPOD_Ice - ssha_smooth_20_ku_LEGOS_GPOD_Lead
Alternatively ssha_interp_20_ku_LEGOS_GPOD_Ice or ssha_20_ku_LEGOS_GPOD_All (using the flag_surf_type_20_ku_LEGOS_GPOD to select sea ice floe points) can be used.

3. CSAO Software

The software develop for this project is written as object orientated python. The language and code structure was chosen to allow for the rapid development and adaptation required to work with all the team processed comparison data

3.1 Post processing of merged data

We have included three ipython notebooks that display how to read, plot and post process the merged and combined CSAO data products.

3.1.1 Merge_track_open.ipynb

This notebook will fist search through a given structure of a target month and find all merged data tracks within. The example data provided as part of the GitHub repository is given as an example.
We use the same code structure to both merge the data and to access the merged files. The CS2_track object (described in 3.2) is intialised using a merged data track file, and all variables within are listed. A selection of variables are then read using a number of search strings. The list_vars method returns a list of all variables that match the given list of search strings.
Following this are a selection of codes that plot: range anomalies, all anomalies from an institution, all non-range anomalies, all cases of altitude-range, all interpolated SSHA variables.

3.1.2 Combine_track_open.ipynb

The notebook accesses a combined data files for the entire Antarctic region. Not that for the example data provided on Github, there is limited variables and spatial coverage due to file size limitations.
Fist of all a cartopy map projection is defined (South Polar Stereographic) and my grid_set module is used to create a 50km spaced grid on this projection.
Similar to 3.1.1, the available variables are listed, and a selection are read. Additionally a directory matching the combined data filename is created to store figures in.
A large adaptable plotting and binning script is provided. There are sections of code here that can evaluate DOT, and varying surface classification.
The beginning part of the code
For the GitHub example data the various range anomalies are binned and plotted on the previously defined map projection. Alongside each map is a histogram of the collected along-track data.

3.1.3 Combine_track_region_open.ipynb

This notebook is similar to that described in 3.1.2, but focusses upon a combined data file that considers that data within a restricted region. The main difference to 3.1.2 is that the map

projection and binning grid is an Orthographic projection defined by the limits of the data in the file. The example data given is for the west Weddell sea.
The binning and plotting script is the same as described in 3.1.2, though the example gives the role of changing range measurements upon the Bl-D L2 freeboard evaluation.

3.2 CS2_Track object

This is the python class the whole processing code is based upon. In the case of merging tracks the class is initialised with an existing L2 Baseline-D satellite track NetCDF file. Initialisation options include longitude/latitude lints (for restricting the area of interest) and the option to read in variable attributes from the NetCDF file for future information.
If data is found within the area of interest, essential time, longitude and latitude data are read. This information is obtained within the python class object.

As our merged dataset mirrors the format of the existing L2 data, we can then use the same code to read the merged and combined data formats.

3.2.1 list_vars, add_vars and add_vars_1Hz

The list_vars method will print out all the variables within the file, filtered by any string. For example including 'range' will list all the variables within the file that have the word range in the title.
Taking the output of list vars (or a custom list), add_vars will read these variables. This works for all the data on the 20Hz time dimension
For data that is given on the 1Hz time dimension, add_vars_1Hz will read these data, and then transfer them to the 20Hz time dimension using the L2 'ind_first_meas_20hz_01' variable.

3.2.2 add_data_time

This is the essential method for comparing the additional term processed data. This data takes time data from a companion data set, along with a list of variables to be added to the CS2_Track object. The time vector from the Bl-D L2 file is compared to the companion time data, with each data point to be added, accurately inserted into the CS2_Track object. The average time difference between the new and companion time vectors is printed to show the accuracy of data insertion. Doing this data comparison Time point by time point ensures we are comparing exactly coincident data.

3.2.3 add_data_lat

This method works indentically to add_data_time, except that we compare the Bl-D L2 latitude vector with a companion latitude vector. This method allows for the inclusion of regridded static data (such as the NSIDC NASA ice concentration).

3.2.4 make_mask

This method creates an additional mask variable. Either by copying the mask attribute of and of the contain variable from the L2 file or comparison data, or by manually setting a logical array, such as selecting a particular surface type from 'flag_surf_type_class_20_ku'. This mask variable is used by future functions upon the CS2_Track object.

3.2.5 save_nc

This method saves all the read and added variables in the CS2_track object to a new NetCDF file. All variable data attributes are saved into the new file. With a call to the method check_attr required to allow for consistency to the NetCDF format.
The variables that have been read and added are save within the list names CS2_track.vars. By editing this list prior to calling save_nc if is possible to restrict the variables written to the file. If a mask attribute has been previously created, then only the data indicated by the mask will be saved.

3.3 CS_var_list and CS_track_list objects

These two objects work together to allow us to collect the data processed through the CS2_track object for saving as a combined data file.
The CS_track_list object aims to collect all data of interest from many CS2_track objects with each variable given as a CS2_var_list. The CS_track_list object is initialised with a name and a list of variables that will accumulated.
These objects can be used to create computationally efficient combined data files that collect together a limited list of variables to cover a particular time and space. These files greatly aid future analysis.

3.3.1 addtrack method

This method takes a CS2_track and a list of the data variables to to be added. The method uses the mask attribute of the CS2_track and adds all the data indicated by the mask to CS_track_list object.

3.3.2 save_nc method

This method works identically to the CS2_track.save_nc, though works over the data collected from many CS2_track objects. For variable attributes, these can be added to each individual CS_var_list from a CS2_track_objet (typically on the first step of a processing loop). With code “CS_var_list.attr = getattr(CS2_track,v+’_attr’)”

3.4 add_file object

This object standardises the file format of all processing comparison data. The object allows us to query a time point from a Bl-D L2 file and return either the comparison file to be read, or the variable slice from a combined data file (for example the MSSL data which is given as a large combined data file). As we are using python objects, these method calls can be generalised, and all the custom data handling required for each comparison data set can be performed ‘behind the scenes’.
The add_file object is initialised using a description name, the path to the partner data, and options as to whether we have single or combined files, how time and date are recorded on the files names, and additional words that need inclusion on the data files names. Also here we can set ‘tshift’ a value to make sure the time vectors align correctly, for example if different zero time point is used within the time vector.

3.4.1 get_files_times

This method searched through the given data path and produces either: a list of time points of the beginning of each file; or a list of Time points found from breaks in the time record, indicating a point where two satellite tracks have been combined. The combined NetCDF files is loaded in this case.

3.4.2 get_tp

This method takes in the first time point of a Bl-D L2 file and searches the comparison add_file object to see if any comparison data exists. This called is entirely generalised so can be performed in a loop over a list of add_file objects. A logical True or False is returned if data has been found or not. Within the object the file to read ('f_nc") and the slice within that file ('F_pt') are set.

3.4.3 load_nc and load_clean

If the get_tp method shows that comparison data is available, a call to load_nc will open this file if it needs to be open (a combined data can well be already open so we need to do nothing at this point). Once this method is called then variable within the file can be easily read using the add_file.f_nc attribute which is in the object. The slice of the file which we need is given as add_file.F_pt. Therefore to access a variable from the file to compare to the Bl-D L2 file of interest we use "add_file.f_nc['variable_name'][add_file.F_pt]" to get exact slice of data we wish to compare.
Once read and compared we call load_clean to close the file (if it needs to be closed).

3.5 Track_merge script

This script is used to build the combined track files. It works on all the classes and methods in previous section

3.5.1 User options

tstart and n_days, the data start point for processing and the number of days to process
MERGE_TRACKS logical option, are the tracks written to file or not

Plotting options - we can just plot the merged data. This can make a lot of plots
PLOT_RANGE - plots all the versions of the retracked range to surface. We plot altitude - range.
PLOT_RANGE_ANOM - plots all the range anomalies to the Bl-D L2 range_1_20_ku
PLOT_EXTRA_ANOM - plots all the other anomalies - typically mean sea surface and correction differences to Bl-D L2
PLOT_INTERP - plots all cases of interpolated sea surface height anomaly, also plotted is the L2 sash anomaly over sea ice for comparison

check_single_track logical option. If this is set, after a single track is processed, variable attributes and interquartile ranges are printed to screen.
check_filter = '_' (print all variables)
check_filter = 'ANOM' (print only variables that contain the phrase 'ANOM', so only the anomalies)

merge_dir - where to save the merged data. The year/month directory structure is created by the script
fig_dir - where to save all the figures

lons/lats ranges for longitude/latitude for merging tracks. Only data within the longitude and latitude ranges are considered

COMBINE_FILES logical option, creates a combined data files many tracks collected together
Combine_name - the name of the combined file

Comb_ST() this is a user set function that operates on any CS2_track object. I needs to return a logical mask vector showing which data to include. Examples are given selecting only sea_ice/lead/ocean for example.
Combine_vars - a list of the variable names to combine

There are 3 final option v_tload, v_tmask, v_save, that if set to true makes the code to print more information to screen when processing.

3.5.2 Processing method

The main processing loop considers all the data available for each day. We then proceed looping through the number of days set using the user options

Within the loop, first we find all the Bl-D L2 files available for that day. This is performed by searching through the Bl-D directory structure and looking at individual file names. The file and the start time for each file is recorded in a list and sorted.
Next we set the variables to read from the Bl-D L2 file, a separate list for the 20Hz and 1Hz dimension data.

Next we initialise an add_file object for each partner comparison data. CLS, MSSL, DTU and IsardSat can be initialised using all the inbuilt object methods.The LEGOS GPOD add_file object has to be constructed manually as the tracks are collect by cycle rather than by month. After each object is initialise we define the variable names as they are in partner comparison data files, and then again as what we wish to call these data within the merged data.

Additionally we load and set up the NSIDC NASA ice concentration data and it's regridding object for along track data interpolation. This step uses the python basemap library and my own grid_set gridding library and associated 'NSIDC_gs_SH.npz' grid file, both of which are included in the GitHub. It is possible to use a cartopy projection here instead if required.

Next starts the track by track sub-loop. This work as a while loop letting it access all Bl-D L2 files. When accessing the file we check if it is within the lon/lat area of interest. For each Bl-D L2 file, the get_tp method is performed on each partner add_file object. This lets us know if any data is to be merged.
If any of the partner data is present, the Bl-D files are loaded, ice concentration is added, and then the partner data is added using the add_file objects and add_data_time methods. If no partner data is available for this Bl-D L2 track, then empty vectors are added instead. This ensures we have merged data file with data arrays for all variables within it, for any case where at least one comparison data set is available.

Next we perform variable conversions to ensure all Bl-D L2 and compassion data is of the same format.
We create summed geophysical and atmospheric correction arrays for Bl-D L2 data. The MSSL and DTU range data is converted to the Bl-D L2 range_1_20_ku variable definition.

The LEGOS GPOD sea surface height anomaly is back processed to a range by using the supplied corrections and mean sea surface.

Then we have two subsub-loops that takes all the comparison range_1_20_ku arrays, and calculates a range_1_20_ku_ANOM array for it. A similar subsub-loop is run for mean_sea_surf_20_ku.
There follows manual calculation of correction anomalies and the difference between the Bl-D L2 and and LEGOS GPOD ssha_interp_20_ku variables.
Finally we convert the various LEGOS GPOD flag arrays into a single flag_surf_type_20_ku array.

Then we have code that tidies up the data we have read, but don't wish to merge and the attributes read from comparison NetCDF files are checked for write compatibility.

We then have the code that writes the merged track to file (if desired) and combines the track into the combined track object (if desired).

Finally is the code that produces the automatic track figures. The left hand pane of these figures require the python basemap library which is longer support to allow the script to on our UCL CPOM server. It is possible to update this code to use the new well supported python cartopy library.