Intermediary Data Format and Content Specification

OceanDataLab (ODL), France

Author	OceanDataLab
Version/Revision	1.5
Date of issue	2019/12/26

Revision Change log

Chronology			
Issue	**Date**	**Type**	**Change description**
0.0	2016/05/01	Initial draft	
1.0	2016/10/21	Version 1	
1.1	2016/11/14	Update	Add mandatory global attributes
1.2	2019/01/21	Update	Add details about data serialization and GCP density
1.3	2019/09/30	Change of document	Convert google document into latex
1.4	2019/10/02	Update	Add summary and illustrations
1.5	2019/12/26	Update	Clarification and SEAScope conformance check

Contents

Summary

The IDF (Intermediate Data Format) is a set of rules for formatting netCDF4 files meant to reduce the number of operations required for opening these files and interpreting their content as geolocated data.
These rules are based on conventions CF 1.6 or later with additional constraints to leverage the self-describing capabilities of netCDF files, allowing:

- compatibility (at least partial) with software that rely on CF convention compliance
- quick scanning of IDF files when searching for data matching a set of constraints, by filtering on the global attributes that summarize the content of the file and require little effort to read
- multiresolution handling for high resolution data
- low-latency data access due to the reduced overhead for reading from IDF files.

The difference between creating a classical netCDF file and producing an IDF file consists mainly in including the following mandatory information:

- few global attributes (section 3.2, table 6)
- additional location variables called GCP (Ground Control Point) which are defined in section 2.4
- Additional files that contain filtered variables to handle multiresolution (see section 2.3, required for high resolution data only).

Recommendations to build a proper IDF file are provided throughout this document, with a wide variety of examples listed at the end (netCDF headers for IDF files).

1 Introduction

1.1 Purpose and scope

The objective of this technical note is to define how data is described and stored in IDF files.
This format focuses on extracting file contents efficiently and fast enough to be loaded by client applications for interactive visualization, animation or on-the-fly data processing.
It is worth noting that this version of the IDF specifications focuses on complying with the CF and ACDD conventions while providing the information needed by the current version of the SEAScope software (more details below).
This software has been designed and implemented to display data for case studies that involve a limited amount of data, focussing on helping users with the development of new algorithms by offering the means to perform quick extraction/ analysis/visualization cycles.
While compliance with CF and ACDD conventions will be maintained in the future, major changes in the format are foreseen as the SEAScope requirements evolve and as new uses of IDF are discovered (e.g. machine learning benefiting from data format standardization).
IDF v1.5 and SEAScope are therefore a good choice for showcasing and analyzing a data set tailored for a study but the aforementioned limitations should be considered carefully before investing resources in the conversion of larger amounts of data. It is planned to address these limitations in v2.0 of the IDF specifications and with future versions of SEAScope.

1.2 Applicable documents

The following documents contain requirements and information applicable to this document:

[AD-1]	netCDF Climate and Forecast (CF) Metadata Conventions version 1.6 available from `http://cfconventions.org`
[AD-2]	COARDS Conventions available from `http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html`
[AD-3]	[AD-1] UDUNITS-2 package available from urlhttp://www.unidata.ucar.edu/software/udunits/udunits-2/udunits2.html
[AD-4]	[AD-2] ISO 8601, The International Standard for the representation of dates and times: `http://www.iso.org/iso/date_and_time_format`
[AD-5]	[AD-3] Unidata Attribute Conventions for Dataset Discovery (ACDD), available from `http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html`
[AD-6]	[AD-4] Current version (CF-1.6) of the standard name table can be found at `http://cfconventions.org/standard-names.html`
[AD-7]	[AD-5] NASA Global Change Master directory (GCMD) Science Keywords andAssociated Directory Keywords, available at `http://gcmd.nasa.gov/Resources/valids/archives/keyword_list.html`

1.3 Acronyms and abbreviations

AOML	Atlantic Oceanographic and Meteorological Laboratory
CF	Climate and Forecast
ECMWF	European Centre for Medium-Range Weather Forecasts
GCP	Ground Control Point
IDF	Intermediate Data Format
WMO	World Meteological Organization

1.4 Glossary

Collection	Ensemble of data that have the same geophysical variables, spatial and temporal granularity, i.e. a data set which can be considered homogeneous (e.g. all netCDF files from ECMWF wind model)
Granule	The smallest aggregation of data that share the same spatial and temporal coverage, with the same resolution
GCP	Geographical location associated with a pixel of the data matrix
IDF	NetCDF format with additional variables and attributes necessary to SEAScope
Product	Ensemble of data as defined by the provider.

2 IDF overview

2.1 Motivations

SEAScope is a 3D visualization application for satellite, in-situ and numerical model data (more information at `https://seascope.oceandatalab.com`). It offers advanced rendering functionalities that enable easy data synergy studies between several sources of observations and simulations. It also features tools to help users design and test algorithms on a large variety of data with immediate visual feedback. Thanks to its 3D rendering engine, SEAScope does not require any specific projection in its input: it uses the geographical coordinates of a few reference data points to map matrices of geophysical values with their footprint on the globe. While this method avoids the reprocessing required to transform data coordinates into a specific projection, direct access to raw data is still judged inadequate due to the heterogeneity of raw data formats: supporting a large variety of input data formats puts a toll on data loading performance. Some data may also need a time and resource consuming preprocessing (eg. resampling or scientific algorithm) for which it is preferable to store the result in a common format.
Insufficient data description, file structure complexity or the need of on-the-fly reprojection are sources of overhead that may discourage the use of netCDF files in applications that require fast data access (e.g. data streaming, search engine) and prevent the automation of some processes that rely on data homogeneity (e.g. classification).
These considerations lead to the definition of the Intermediate Data Format (IDF), a unique file format which standardizes geolocation patterns and implements a space-efficient strategy for storing geophysical variables. Future iterations of the format specifications will improve this strategy to make the IDF more suitable for streaming data over an Internet connection.
The definition of a unifying format is also an opportunity to provide data with a certain degree of self description and user friendliness in case of data distribution or sharing. During this process, we have paid a special attention to preserving the data geometry since no preliminary projection is required. Maintaining a good dynamic range of the geophysical variables in the IDF while keeping data as light as possible was also a main concern to allow subsequent calculations and manipulations.

2.2 NetCDF-4 classic file format

IDF data files use the NetCDF-4 Classic format. A major advantage to the use of NetCDF-4 format products from the producer's perspective is that no additional metadata records are required when using this format since one can easily extract it from the files without having to decompress the entire file. IDF data sets must comply with the Climate and Forecast (CF) Conventions, v1.6 or later because these conventions provide a practical standard for storing oceanographic data in a robust, easily-preserved for the long-term, and interoperable manner.

2.3 Multiresolution support

Multiresolution support is required for high resolution files in order to avoid loading (or resampling on the fly) a high amount of data when the SEAScope camera is far away from the data footprint and the number of values to display in this footprint exceeds the resolution of the screen, which would create noisy images. It also allows for nice transition effects as the application switches to the most suitable data resolution while zooming inside the data footprint.

The IDF supports multiresolution by generating one IDF file for each relevant spatial sampling derived from the high resolution granule. For each IDF file, the associated spatial spacing is indicated in meters as a global attribute (see `idf_spatial_resolution` in 3.2 Global attributes) in order for SEAScope to know which IDF file to use according to the viewport. These multiresolution IDF files are also arranged with an index, called here subsampling index, which goes from 0 (lowest spatial spacing) to n (highest spatial spacing). This index is specified in the IDF filename and as a global attribute (see `idf_subsampling_factor` in sections 3.1 File naming and 3.2 Global attributes). Although not mandatory for SEAScope use, the most simple/common case is to apply a downsampling factor of 2 between two successive IDF files in terms of spatial sampling. Then, the associated spatial spacing is each time multiplied by 2 and the subsampling index is equivalent to the exponent of 2 used from IDF file with the lowest spatial spacing (which is not necessarily the raw data spacing if resampling occurs during conversion into IDF).

For example, from an IDF file with a subsampling index of 0, a subsampling index of 1 indicates a downsampling by a factor 2, a subsampling index of 2 indicates a downsampling by a factor 4 and so on (illustrion on figure 1).

(a) S2_20170214_00.nc
Subsampling index 0
Full resolution (20m)

(b) S2_20170214_01.nc
Subsampling index 1
Filtered at 40m

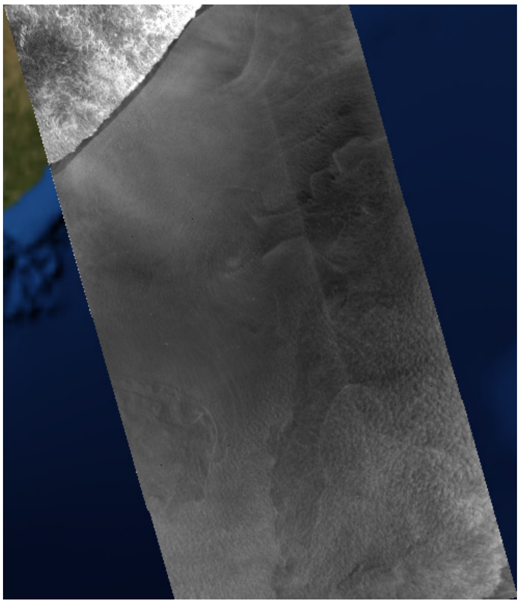
(c) S2_20170214_02.nc
Subsampling index 2
Filtered at 80m

(d) S2_20170214_03.nc
Subsampling index 3
Filtered at 160m

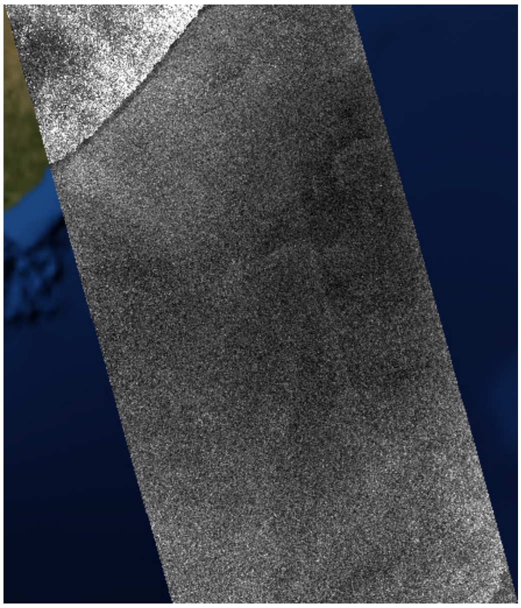
(e) S2_20170214_04.nc
Subsampling index 4
Filtered at 320m

Figure 1: Screenshots from SEAScope of the IDF from Sentinel 2 high resolution data at different zoom level showing different filtering length

Data that are not filtered properly may appear noisy when you zoom out as it is illustrated in figure 2.



(a) S1_20180811_05.nc
Subsampling index 5
Filtered at 640m



(b) S1_20180811_00.nc
Subsampling index 0
Full resolution (20m)

Figure 2: Illustration of the use of the multiresolution: Screenshots from SEAScope of the IDF from Sentinel 1 sar roughness data using a filtered view (left) and a non filtered one (right).

2.4 Geolocation with GCP

Ground Control Points (GCP) are a set of control points relating coordinates (typically latitude and longitude but also eventually depth or time) with their position in the array of the variable (ie. indices).
In most cases, data geolocation patterns can be sufficiently described with a subset of coordinates. Thus, a correct geographic mapping can be performed in SEAScope without transmitting coordinates for each data point. Moreover, subsetting coordinates reduces the size of IDF files.
GCPs are also intended to be used in the spatio-temporal indexation scheme of SEAScope. Currently, the concept of GCPs is not widely used: among geospatial tools we can cite GDAL which has defined GCPs in its model and among remote sensing data we can cite Sentinel-1 and Sentinel-3 L1 data format for which the full lat/lon coordinates are not given. As a consequence, no convention has been defined yet regarding the GCPs dimensions and variables naming. It will become a necessity as there is an increasing number of high resolution data. This document defines this naming for each conventional data model, see section 3.5 Coordinate variables. The main rules are the following:

- GCP dimensions are named from the main dimensions of the data model in use, with the suffix '`_gcp`'. As an example, for the swath data model, the main dimensions are '`row`' and '`cell`' then the GCP dimensions (subset of the main dimensions) are '`row_gcp`' and '`cell_gcp`'. The suffix '`_gcp`' allows to discriminate the main dimensions used for the geophysical variables and the GCP dimensions used for the subset of coordinates.
- GCP coordinate variables are named from the classic coordinates names with the suffix '`_gcp`': `lon_gcp`, `lat_gcp`, eventually `time_gcp` for time-dependent data model. It seems preferable to add this suffix in order to not mislead software or users expecting full coordinates in variables named `lon` or `lat`.
- GCP position in array variables are named from the main dimensions of the data model in use, with the prefix '`index_`' and the suffix '`_gcp`'. With the swath data model as an example, we obtain '`index_row_gcp`' and '`index_cell_gcp`'. These variables provide index location of GCPs in the data model dimensions.

GCP density shall be as minimal as possible to keep the IDF file size small but still to allow interpolation of the intermediate positions at a precision better than the IDF spatial spacing. It depends on data resolution, nature and geometry so it is not possible to provide a formula that computes the GCP density automatically. Here are some hints that can help with the tuning of GCP density:

- At least one gcp every degree is required for low resolution data.
- Start from a low number of GCPs (see previous recommendation), check that the granule is displayed at the right location and that it is not distorted by interpolation (use coastline and islands as reference if they are available). If not double GCP density and repeat.
- GCP density must never exceed the resolution of the data (it provides no meaningful information and makes the files heavier). There is an edge case for IDF files generated to be visualized in SEAScope as wind barbs or arrows: the symbols are placed above GCPs on the globe, so to increase the barbs/arrows density you have to increase the GCPs density too, even to the point wherein it might reach data resolution (but it cannot go beyond this limit).
- For trajectories, GCPs must have the same resolution/density as data.

Another important point to note is that GCPs must be placed between data points. SEAScope currently requires this to detect the outer shape of the data.

For trajectories (time-dependant data model) the GCPs are placed at the center of each point.
For all other data, the GCPs are placed at the edge of data points (see figure 3) as placing them at the center of the data points would generate a distorted rendering for the granule.

This constraint is not very intuitive and should be removed in future revisions of the specifications once the application evolves to compute the outer shape of the granules automatically.

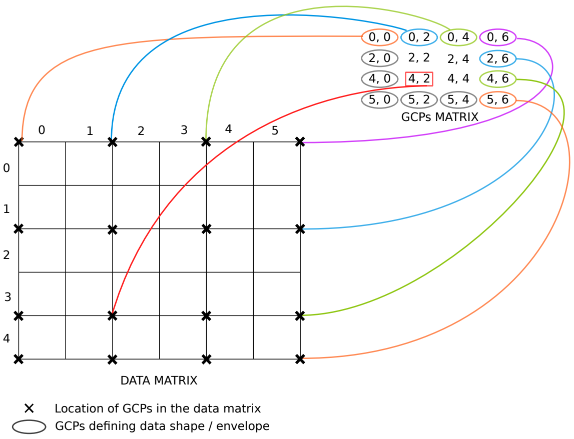


Figure 3: Location of Ground Control Points in the data matrix

3 IDF NetCDF-4 file structure

In the context of the IDF definition, a 'granule' is defined as the smallest aggregation of data which is independently managed (i.e. described, inventoried, retrievable). This concept is of importance to choose an adequate granularity for data partitioning.

3.1 File naming

3.1.1 Unique granule in the original file

If one granule is stored in the file to be converted, the IDF file name shall follow the original name of the file ending by the extension '`_idf_`' and a subsampling index. Keeping the original file name enables users to easily retrieve the original data. The subsampling index is a two digits starting at 00 for the higher resolution (more details can be found in section 2.3 Multiresolution support).

```
<Original File name>_idf_<Subsampling index>.nc
```

3.1.2 Several granules in one file

When the original file name cannot be used as a unique name (e.g. the file needs to be split into several time steps, orbits, platforms ...), the name of the IDF is not self explanatory and should be reconstructed following this standard name (file naming convention components are described in Table 5):

```
<Product_string>_<Indicative Date><Indicative Time>_<Processing Level>_v<Product Version>
_fv<File Version>_idf_<Subsampling index>.nc
```

For most drifter databases, one file contains all the available data produced by multiple platforms during their lifetime. There is one IDF file per platform, instrument and time range to be displayed (month for most drifters, day for drifters at high resolution like those from CARTHE experiment). For platforms having different onboard instruments, all data (from different on board sensor) can be stored in the same file if they have similar resolution and coordinate dimension. If the different instruments have different sampling times and/or different availability time ranges, they are stored in separate IDF files. If there is a significant interruption time (to be defined, more than one day for instance) or a significant change in the acquisition process (ex: maintenance change on the instrument, or replacement), then the data product for a specific platform/instrument may be split into two or more chronologically consecutive files within the current reference time range. A file containing the data from one platform (buoy, ship,...) during one specific time range will be named as follows:

```
<Product String>_<Start Date><Start Time>_<End Date><End Time>_<Platform>
_v<Product Version>_fv<File Version>_idf_<Subsampling index>.nc
```

A file containing data from a set of similar buoys (surface drifters, ARGO floats ...) during one specific time range will be split so that there is one file per buoy. The naming is similar to the one for fixed station/platform.
For along track L3 products where data are collated daily (several passes are stored in one file), the pass and the cycle is specified during the naming process of the IDF file so that there is a unique granule stored in each file.

```
<Product String>_<Start Date><Start Time>_<End Date><End Time>_c<cycle>_p<pass>
_v<Product Version_-fv<File Version>_idf_<Subsampling index>.nc
```

Name	Definition	Description
<Indicative Date>	YYYYMMDD	The identifying date for this data set, using the format YYYYMMDD, where YYYY is the four-digit year,MM is the two-digit month from 01 to 12, and DD is the two-digit day of month from 01 to 31. The date used should best represent the observation date for the dataset in UTC.
<Indicative Time>	HHMMSS	The identifying time for this data set in UTC, using the format HHMMSS, where HH is the two-digit hour from 00 to 23, MM is the two-digit minute from 00 to 59, and SS is the two-digit second from 00 to 59. The time used is dependent on the <Processing Level> of the data set: • L2P: start time of granule • L3U: start time of granule • L3C and L3S: centre time of the collation window • L4: nominal time of analysis All times should be given in UTC and should be chosen to best represent the observation time for this dataset. Note: providers should ensure the applications they use to determine UTC properly account for leap seconds.

<Processing Level>	The data processing level code	The data processing level code (L2P, L3U, L3C, L3S, or L4).
<Product String>	A character string identifying the current product set.	The string is used uniquely. The unique 'name' of the product line.
<Product Version>	nn.n	Version number of the product. A different version number must be associated with a product at each processor algorithm or configuration change. For example, '02.0'.
<File Version>	xx.x	Version number for the file, for example, "01.0".
<Subsampling index>	<byte> (ie 0)	Index indicating the spatial resolution of the IDF file. In most cases, it represents a subsampling of a factor exponent of 2 from the IDF file with the lowest spatial spacing.
<Platform>	The identifier of the measuring platform.	The identifier of the acquisition platform: if it as a WMO identifier, it shall be labelled WMO<identifier> otherwise it will use the provider identifier.

Table 5: File naming convention components

3.2 Global attributes

Table 6 below summarizes the global attributes that are mandatory or useful for IDF data files.
More details on the CF-mandated attributes (as indicated in the Source column) are available at: `http://cfconventions.org/Data/cf-conventions/cf-conventions-1.6/build/cfconventions.html#attribute-appendix` and information on the ACDD recommendations is available at `http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html`.
Some attributes (in orange) are required for compatibility with the SEAScope IDF loading mechanism.

Table 6: Mandatory global attributes for IDF data file: attributes in orange are required by SEAScope, attributes in blue can be displayed and thus are of use for SEAScope, attributes in purple can be displayed by SEAScope depending on the type of data, attributes in green are highly recommended for CF compliancy, all the others are optional.

Global Attribute Name	Format	Description	Source
`idf_version`	string	Version of the IDF specifications targeted when formatting the file	IDF
`idf_granule_id`	string	Text string, unique identifier for the granule in SEAScope. For most cases this attribute is the name of the IDF file without the subsampling index, idf suffix and file extension.	IDF
`idf_subsampling_factor`	int	Subsampling index integer	IDF
`idf_spatial_resolution`	float	Geophysical spatial resolution of the variable, used in SEAScope to determine the resolution to use as a function of the zoom. Should be specified in meter.	IDF
`idf_spatial_resolution_units`	string	Unit for spatial resolution, has to be meter ('m')	

time_coverage_start	string	Representative date and time of the start of the granule in the ISO 8601 compliant format of 'yyyymmddThhmmss.ssssssZ'. The exact meaning of this attribute depends the type of granule: • L2P: first measurement in granule (identical to 'time' netCDF variable) • L3U: start time of granule • L3C and L3S: representative start time of first measurement in the collation • L4: representative start time of the analysis (start_time and stop_time together represent the valid period of the L4 granule)	ACDD
time_coverage_end	string	Representative date and time of the end of the granule in the ISO 8601 compliant format of 'yyyymmddThhmmss.ssssssZ'. The exact meaning of this attribute depends the type of granule: • L2P: last measurement in granule • L3U: stop time of granule • L3C and L3S: representative stop time of last measurement in collation • L4: representative stop time of the analysis (start_time and stop_time together represent the valid period of the L4 granule)	ACDD
Conventions	string	A text string identifying the netCDF conventions followed. This attribute should be set to the version of CF used and should also include the ACDD. For example: 'CF1.4, Unidata Observation Dataset v1.0'.	CF
title	string	The "title" attribute gives a brief description of the collection. Its use is highly recommended and its value will be used by THREDDS. It therefore should be human readable and reasonable to display in a list of such names.	CF, ACDD
summary	string	The "summary" attribute gives a longer description of the data. Its use is highly recommended. In many discovery systems, the title and the summary will be displayed in the results list from a search. It should therefore capture the essence of the data it describes. For instance, we recommend this field includes information on the type of data, how the data was created (e.g., instrument X; or model X, run Y), the creator of the file, the project for which the data was created, the geospatial coverage of the data, and the temporal coverage of the data. This should just be a summary of this information, more details should be provided in the recommended creator attributes, the recommended geospatial attributes, and the recommended temporal attributes.	ACDD
references	string	Published or web-based references that describe the data or methods used to produce it.	CF
institution	string	Institution which the data originally come from. If a file is a simple reformatting without any modification the source institution is to be used.	CF, ACDD

`institution_abbreviation`	string	Abbreviation of the above full institution name.	CF, ACDD
`history`	string	The 'history' attribute provides an audit trail for modifications to the original data. It should contain a separate line for each modification with each line including a timestamp, user name, modification name, and modification arguments. Its use is recommended and its value will be used by THREDDS as a history-type documentation. The 'history' attribute is recommended by the NetCDF Users Guide and the CF convention.	CF, ACDD
`comment`	string	Miscellaneous information about the data or methods used to produce it.	CF, ACDD
`license`	string	Describe any restrictions to data access, use, and distribution. IDF data sets should be freely and openly available to comply with the R/GTS framework, with no restrictions. However, if a user should submit a simple registration via a web form, for example, the URL could be given here. Defaults to 'IDF protocol describes data use as free and open.'	ACDD
`id`	string	The unique character string for this granule.	ACDD
`naming_authority`	string	The `id` and `naming_authority` attributes are intended to provide a globally unique identification for each granule. The `id` value should attempt to uniquely identify the granule. The naming authority allows a further refinement of the `id`. The combination of the two should be globally unique for all time. We recommend using reverse-DNS naming for the naming authority.	ACDD
`product_version`	string	The product version of this product.	
`processing_software`	string	Name and version of the processing software	
`uuid`	string	A Universally Unique Identifier (UUID). Numerous, simple tools can be used to create a UUID, which is inserted as the value of this attribute. See `http://en.wikipedia.org/wiki/Universally_Unique_Identifier` for more information and tools.	
`netcdf_version_id`	string	Version of NetCDF libraries used to create this file. For example, '4.1.1 of Dec 22 2011 16:33:39'	
`date_created`	string	The date and time the data file was created in the form '`yyyymmddThhmmssZ`'. This time format is ISO 8601 compliant.	ACDD
`date_modified`	string	File last modification date in the form '`yyyymmddThhmmssZ`'. This time format is ISO 8601 compliant.	ACDD
`file_quality_level`	integer	A code value: • 0 = unknown quality • 1 = extremely suspect (frequent problems, e.g. with known satellite problems) • 2 = suspect (occasional problems, e.g. after launch) • 3 = excellent (no known problems)	
`spatial_resolution`	string	A string describing the approximate resolution of the product. For example, '1.1km at nadir'.	

`time_coverage_resolution`	string	The resolution provides an idea of the density of the data inside the time range and should be an ISO8601 duration string (e.g., 'P10D').	ACDD
`geospatial_lat_max`	float	Decimal degrees north, range -90 to +90.	ACDD
`geospatial_lat_min`	float	Decimal degrees north, range -90 to +90.	ACDD
`geospatial_lon_max`	float	Decimal degrees east, range -180 to +180.	ACDD
`geospatial_lon_min`	float	Decimal degrees east, range -180 to +180.	ACDD
`geospatial_vertical_min`	float	Use the min attribute to describe a simple vertical minimum depth or height. If `geospatial_vertical_units` is not used, vertical is assumed to be in meters above ground. Further refinement of the geospatial bounding box can be provided by using the units and resolution attributes.	ACDD
`geospatial_vertical_max`	float	Use the max attribute to describe a simple vertical maximum depth or height. If `geospatial_vertical_units` is not used, vertical is assumed to be in meters above ground. Further refinement of the geospatial bounding box can be provided by using the units and resolution attributes.	ACDD
`geospatial_vertical_units`	string	meters	ACDD
`geospatial_vertical_positive`	string	The `geospatial_vertical_positive` attribute indicates which direction is positive (a value of 'up' means that z increases upward, like units of height, while a value of 'down' means that z increases downward, like units of pressure or depth).	ACDD
`nominal_latitude`	float	Location latitude of acquisition platform. Only used in case of a fixed station (ex: moored buoy). This may differ from the actual GPS location provided in lat/lon variable. Decimal degrees north, range -90 to +90.	
`nominal_longitude`	float	Location longitude of acquisition platform. Only used in case of a fixed station (ex: moored buoy). This may differ from the actual GPS location provided in lat/lon variable. Decimal degrees east, range -180 to +180.	
`source`	string	Comma separated list of all source data present in this file. List current sources first,followed by Auxiliary sources.	CF
`source_version`	string	The version of the source data used, as provided by the original producer `wmo_id` string WMO identifier, if any. Use the source provider id if no WMO identifier exists for this platform.	
`buoy_network`	string	Identifier of the observation network to which the platform belongs to.	
`station_name`	string	Full name of the station (usually a location)	
`station_id`	string	Identifier of the station in network (this is the provider id which may be different from the WMO id).	
`sea_floor_depth_below_sea_level`	float	Depth as a positive value in meters.	
`site_elevation`	float	Elevation as a positive value in meters (for platforms on land such as lighthouses).	
`platform`	string	Satellite(s) used to create this data file. Select from the entries found in the Satellite Platform column of table 8 and provide as a comma separated list if there is more than one.	
`sensor`	string	Sensor(s) used to create this data file. Select from the entries found in the Satellite Sensor column of table 7 and provide as a comma separated list if there is more than one.	

`band`	string	Name of the sensing band used. Can be: • a band name for microwave sensors (C, Ku, Ka, L,...) • a wavelength for optical or infra-red sensors • a derived product such as Chl-a, SST, SPM, ...	
`sensor_description`	string	Free description of the instrument	
`sensor_manufacturer`	string	Name of the company responsible for the instrument construction.	
`sensor_serial_number`	string	Serial number of the instrument, following the manufacturer convention.	
`sensor_install_date`	string	Date and time at which the instrument was installed on the platform (buoy or ship), in the ISO 8601 compliant format of 'yyyymmddThhmmssZ'.	
`sensor_height`	string	Height (positive) or depth (negative) above sea surface at which the instrument is fixed, in meters.	
`sensor_sampling _period`	string	Sampling period of the instrument, in seconds. Applicable for instance to anemometer and wave sensors.	
`sensor_sampling_rate`	string	Sampling rate of the instrument, In Hz	
`sensor_calibration _date`	string	Date and time at which the sensor was last calibrated, in the ISO 8601 compliant format of 'yyyymmddThhmmssZ'.	
`sensor_history`	String list	List of events or interventions on the sensor in the time interval covered by the data file. Specify a date and time in the ISO 8601 compliant format of yyyymmddThhmmssZ' for each entry in this table. Includes for instance any maintenance operation or malfunction event.	
`Metadata_Conventions`	string	Unidata Dataset Discovery v1.0	ACDD
`metadata_link`	string	Link to collection metadata record at archive	ACDD
`keywords`	string	The 'keywords' attribute lists key words and phrases that are relevant to the product. Its use is highly recommended. The values in the list may be taken from a controlled list of keywords (e.g., the AGU Index list or the GCMD Science Keywords). If a controlled list is used, the 'keywords_vocabulary' attribute may be used to identify the list.	ACDD
`keywords_vocabulary`	string	The 'keywords_vocabulary' attribute identifies the controlled list of keywords from which the values in the "keywords" attribute are taken. If you are following a guideline for the words/phrases in your 'keywords' attribute, put the name of that guideline here. The use of this attribute is recommended and its value will be used by THREDDS to identify the vocabulary from which the keywords come. 'NASA Global Change Master Directory (GCMD) Science Keywords' as defined in [AD-7] are recommended.	ACDD
`standard_name _vocabulary`	string	'NetCDF Climate and Forecast (CF) Metadata Convention'	ACDD
`acknowledgement`	string	A place to acknowledge various type of support for the project that produced this data.	ACDD
`creator_name`	string	Provide a name for the most relevant point of contact at the producing organization or agency.	ACDD
`creator_email`	string	Provide the email relevant to the creator name	ACDD
`creator_url`	string	Provide the url relevant to the creator name and this data set	ACDD
`publisher_name`	string	Name of the distributing agency if different from institution attribute	ACDD

`publisher_url`	string	URL of the distributing agency if different from institution url attribute	ACDD
`publisher_email`	string	Email of help desk or contact point of the distributing agency	ACDD
`processing_level`	string	GHRSST definitions are the options: L2 L3U, L3C, L3S, L4	ACDD
`cdm_data_type`	string	Data feature type (point, station, swath, grid,...) as defined by Unidata CDM model (`https://www.unidata.ucar.edu/software/netcdf-java/v4.6/CDM/index.html`)	

The following tables provide the reference codes for some of the above attributes (for which areference is made to these tables).

Sensor code	Description
altimeter	Onboard Jason-x, Envisat, Cryosat-2,...
sar	Synthetic Aperture Radar: onboard ERS, Envisat, Sentinel-1,...
infrared radiometer	Onboard NOAA-xx, MSG-xx, METOP-x,...
microwave radiometer	AMSR-E, AMSR2, GMI, SMOS,...

Table 7: list of currently identified types of satellite sensors

Platform code	Description
leo satellite	Low earth orbit satellite, including for instance all polar orbitingsatellites: ERS, EnviSAT, METOP, ...
geostationary satellite	Satellite in geostationary orbit:
moored buoy	
drifting buoy	Surface drifters from Coriolis or AOML database
ship	
argo	Float from ARGO program

Table 8: list of currently identified types of observation platforms

3.3 Variable attributes

Table 9: Variable attributes for IDF data files. Variable attributes in orange are required for SEAScope platform, those in green are highly recommended. All the others are optional and depend on the variable.

Attribute Name	Format	Description	Source
`_FillValue`	Must be the same as the variable type	A value used to indicate array elements containing no valid data. This value must be of the same type as the storage (packed) type; should be set as the minimum or maximum value for this type. Required for the majority of variables except coordinates, gcps, mask and flags.	CF
`units`	string	Text description of the units, preferably S.I., and must be compatible with the Unidata UDUNITS-2 package. For a given variable (e.g. wind speed), these must be the same for each granule within a collection. It is not required for quality, flags or mask quantities.	CF, ACDD
`scale_factor`	float	multiplication factor to get real data value in units.	

`add_offset`	float	number to be added to the data to get real data value in units. If both `scale_factor` and `add_offset` attributes are present, the data are first scaled before the offset is added.	
`long_name`	string	A free-text descriptive variable name.	CF, ACDD
`valid_min`	Expressed in same data type as variable	Minimum valid value for this variable (in storage type). The fill value should be outside the [`valid_min`, `valid_max`] range. Values outside of `valid_min` and `valid_max` will be treated as missing values. Required for all variables except variable time.	CF
`valid_max`	Expressed in same data type as variable	Maximum valid value for this variable (in storage type). The fill value should be outside the [`valid_min`, `valid_max`] range. Required for all variables except variable time.	CF
`standard_name`	string	Where defined, a standard and unique description of a physical quantity. For the complete list of standard name strings, see[AD-6].	CF, ACDD
`comment`	string	Miscellaneous information about the variable or the methods used to produce it.	CF
`axis`	string	For use with coordinate variables only. The attribute 'axis' may be attached to a coordinate variable and given one of the values 'X', 'Y', 'Z', or 'T', which stand for a longitude, latitude, vertical, or time axis respectively. See: `http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.4/cf-conventions.html#coordinate-types`	CF
`positive`	string	For use with a vertical coordinate variable only. May have the value 'up' or 'down'. For example, if an oceanographic netCDF file encodes the depth of the surface as 0 and the depth of 1000 meters as 1000 then the axis would set positive to 'down'. If a depth of 1000 meters was encoded as -1000, then positive would be set to 'up'.	CF
`coordinates`	string	Identifies auxiliary coordinate variables, label variables, and alternate coordinate variables. See the section 3.5 on coordinate systems. This attribute must be provided if the data are on a non-regular lat/lon grid(map projection or swath data).	CF
`flag_meanings`	string	Space-separated list of text descriptions associated in strict order with conditions set by either `flag_values` or `flag_masks`. Words within a phrase should be connected with underscores.	
`flag_values`	Must be the same as the variable type	Comma-separated array of valid, mutually exclusive variable values (required when the bit field contains enumerated values;i.e., a 'list' of conditions).	CF
`flag_masks`	Must be the same as the variable type	Comma-separated array of valid variable masks (required when the bit field contains independent Boolean conditions; i.e., a bit 'mask'). Note: CF allows the use of both `flag_masks` and `flag_values` attributes in a single variable to create sets of masks that each have their own list of `flag_values` (see `http://cf-pcmdi.llnl.gov/documents/cf-conventions/1.5/ch03s05.html#id2710752` for examples), but this practice is discouraged.	CF
`depth`	string	Use this to indicate the depth for which the current data are valid. Be consistent with the global attribute `positive`.	

`height`	string	Use this to indicate the height for which the wind (or any atmosphere) data are specified.	

3.4 Definitions of storage types within IDF

Computer storage types referred to in this document follow those used in netCDF. The preferred storage type is chosen to minimize the memory required to store a variable and therefore the size of the idf file: aside from space-time coordinates related to GCPs, all variables should be stored as unsigned bytes with `scale_factor` and `add_offset` attributes. More details can be found in the following section.
Note that the conversion in unsigned bytes may result in data degradation. To minimize this, the `scale_factor` and `add_offset` (derived from minimum and maximum values) must be chosen carefully for each granule.

3.5 Space Time coordinate variables

Data related to Earth Observation, whether they are measurements, analysis results or the output of a numerical simulation, can only be exploited if the values they contain can be located in space and time.
Associating data with their space and time coordinates while optimizing storage space is a core principle of the IDF.
Just like its backend, NetCDF, the IDF uses variables to provide coordinates in space and time for multidimensional data arrays.

Time

Adding the time dimension to variables facilitates aggregation of all files of a given dataset along the time axis with tools such as THREDDS and LAS. This applies to data with a geometry that does not evolve (fixed grids, moorings, etc...) but including a time dimension (axis along which time evolves) and a time variable (tick values along the time axis) in all data files provides a unified way for applications to associate data with a datetime.

For these reasons, IDF requires a time dimension and variable. The type for the time values must be double (64bits), defined as the number of seconds since 1970-01-01T00:00:00Z. Defining time as double instead of some flavor of integer allows the definition of sub-second times which is useful for sensors that acquire data at a high frequency.

The time dimension has to exist and must be an axis for the geophysical variables. For data which do not actually vary with time, such as snapshots, the size of the time dimension should be 1 and the associated value in the time variable should be the more meaningful reference time for the data (still expressed in seconds since 1970-01-01T00:00:00Z).

The following examples show how a sensible reference time can be chosen depending on the <Processing Level> of data:

- L2P: start time of granule;
- L3U: start time of granule;
- L3C and L3S: center time of the collation window;
- L4: nominal time of the analysis

Space

As explained in section 2.4, the IDF uses GCPs to locate data in space because they provide a flexible way to define geolocation for any type of geometry, with the additional benefit of being storage efficient when geolocation can be subsampled with a negligible precision loss.

In this version of the IDF, GCPs are only used to associate geographical coordinates (longitude and latitude) with data coordinates (indices in the data matrix). It is envisioned to include time and depth coordinates as well in the future.

Geographical coordinates are defined using two NetCDF variables, `gcp_lat` and `gcp_lon`, whose type is 32 bits float. As it is a good practice, the `units` attribute for these variables must always be defined.

Data coordinates are defined using variables whose number and name may differ depending on the nature and geometry of data, i.e. the data model. These data coordinates must be expressed as unsigned short integers (16 bits). Note that unsigned short integers (ushort) can only store values up to 65535 so if the size of the data matrix along any of its axes exceeds this number, the matrix will have to be split into chunks otherwise it will not be possible to express its coordinates as ushort.

The following data models are currently supported by the IDF:

- Regular latitude/longitude grids
- Non regular or unstructured grids
- Swath grids
- Time dependent variables in 1D

The structure of IDF files for each of these cases is described with more details in the subsections below.

Additional data models may be supported in future iterations of the format to extend the variety of data that can be stored in IDF files.

3.5.1 Regular latitude/longitude grids

This is the simplest case: the geographical (lat/lon) and data (matrix indices) axes are co-linear, meaning that:

- one line of the data matrix can be associated with a single latitude value
- one column of the data matrix can be associated with a single longitude value

Many L3 and L4 products as well as some geostationary L2 products are provided on a regular lat/lon grid.

Two dimensions are required for GCPs (`lat_gcp` and `lon_gcp`), as well as two variables for geographical coordinates (`lat_gcp` and `lon_gcp`) and two variables for data coordinates (`index_lat_gcp` and `index_lon_gcp`).

The `lat_gcp` and `index_lat_gcp` variables only vary along the `lat_gcp` dimension, whereas the `lon_gcp` and `index_lon_gcp` variables only vary along the `lon_gcp` dimension.
It is mandatory that the coordinates are monotonically ordered in these variables.

The time dimension must be specified as unlimited. Although the actual size of the time dimension should be 1 in L3 and L4 granules and the associated "time" variable only has one value (seconds since 1970-01-01), setting an unlimited size for time dimension will allow netCDF tools and utilities to easily concatenate (and average for example) a series of time consecutive granules.

The following CDL is provided as an example:

```
netcdf example {
dimensions:
        lat = 1801 ;
        lon = 3600 ;
        lon_gcp = 180;
        lat_gcp = 360 ;
        time = UNLIMITED ;
variables: ...
}
```

For regular lat/lon grid, the following dimension and coordinate variables are used:

```
dimensions:
       lon = 7200 ;
       lat = 521 ;
       lon_gcp = 720;
       lat_gcp = 52 ;
       time = UNLIMITED ;
   double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
       time:axis = "T" ;
   float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat:axis = "Y" ;
       lat:comment = "geographical coordinates, WGS84 projection" ;
   float lon_gcp(lon_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
       lon_gcp:axis = "X" ;
       lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
   int index_lat_gcp(lat_gcp) ;
       index_lat_gcp:long_name = "index of ground control points in lat dimension" ;
       index_lat_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                               value (end of last pixel)" ;
   int index_lon_gcp(lon_gcp) ;
       index_lon_gcp:long_name = "index of ground control points in lon dimension" ;
       index_lon_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                               value (end of last pixel)" ;
```

The dimensions of a variable is (time, lat, lon):

```
   ubyte variable(time, lat, lon) ;
       variable:_FillValue = 255UB ;
       variable:long_name = "variable long name" ;
       variable:standard_name = "variable_standard_name" ;
       variable:units = "variable Unit" ; variable:add_offset = offset (float);
       variable:scale_factor = offset (float) ;
```

3.5.2 Non-regular or unstructured grids (projection or finite element mesh)

In this data model, data (matrix indices) axes are not co-linear with geographical (lat/lon) axes but with the y and x axes of a well-known spatial system, meaning that:

- one line of the data matrix can be associated with a single y value
- one column of the data matrix can be associated with a single x value

Coordinates from the well-known spatial system can then be translated into lat/lon using a projection formula.

Two dimensions are required for GCPs (`y_gcp` and `x_gcp`), as well as two variables for geographical coordinates (`lat_gcp` and `lon_gcp`) and two variables for data coordinates (`index_y_gcp` and `index_x_gcp`).

The `index_y_gcp` only varies along the `y_gcp` dimension and the `index_x_gcp` only varies along the `x_gcp` dimension. On the other hand, the geographical variables (`lat_gcp` and `lon_gcp`) vary along both the `y_gcp` and `x_gcp` dimensions.

It is mandatory that values are monotonically ordered in the `index_y_gcp` and `index_x_gcp` variables.

The definition of the well-known spatial system must be included as CRS WKT using the guidelines provided by the CF convention (see section 5.6.1 of `http://cfconventions.org/Data/cf-conventions/cf-conventions-1.7/build/ch05s06.html`).

As in the previous model, the time dimension must be specified as unlimited. Although the actual size of the time dimension should be 1 in L3 and L4 granules and the associated "time" variable only has one value (seconds since 1970-01-01), setting an unlimited size for time dimension will allow netCDF tools and utilities to easily concatenate (and average for example) a series of time consecutive granules **given that they have similar dimension in x and y**.

The following CDL is provided as an example:

```
netcdf example {
   dimensions:
       x = 1801 ;
       y = 3600 ;
       x_gcp = 180;
       y_gcp = 360 ;
       time = UNLIMITED ;
   variables: ...
}
```

For projected `y/x` grid, the following dimension and coordinate variables are used:

```
dimensions:
   y = 7200 ;
   x = 521 ;
   y_gcp = 720 ;
   x_gcp = 52 ;
   time = UNLIMITED ;
variables:
   double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
       time:axis = "T" ;
   float lat_gcp(y_gcp, x_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
   float lon_gcp(y_gcp, x_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
```

```
        lon_gcp:units = "degrees_east" ;
        lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
    int index_y_gcp(y_gcp) ;
        index_y_gcp:long_name = "index of ground control points in y dimension" ;
        index_y_gcp:comment = "index goes from 0 (start of first pixel) to dimension value
                               (end of last pixel)" ;
    int index_x_gcp(x_gcp) ;
        index_x_gcp:long_name = "index of ground control points in x dimension" ;
        index_x_gcp:comment = "index goes from 0 (start of first pixel) to dimension value
                               (end of last pixel)" ;
```

The dimension of a variable is (time, y, x) :

```
    ubyte variable(time, y, x) ;
        variable:_FillValue = 255UB ;
        variable:long_name = "variable long name" ;
        variable:standard_name = "variable standard" ;
        variable:units = "variable unit" ;
        variable:add_offset = offset (float) ;
        variable:scale_factor = scale_factor (float) ;
        variable:valid_min = 0UB ;
        variable:valid_max = 254UB ;
```

3.5.3 Swath grids

In this data model, digital numbers are gridded following the sensor scanning pattern, no well-known spatial system can be used to map geographical and data coordinates so it is necessary to create a mesh of GCPs to map data to geographical coordinates.

Two dimensions are required for GCPs (`row_gcp` and `cell_gcp`), as well as two variables for geographical coordinates (`lat_gcp` and `lon_gcp`) and two variables for data coordinates (`index_row_gcp` and `index_cell_gcp`).
As a best practice, the `cell` dimension should refer to the across-track direction and the dimension `row` should refer to the along-track direction of a polar orbiting (or similar) satellite sensor swath.

The `index_row_gcp` variable only varies along the `row_gcp` dimension and the `index_cell_gcp` variables only varies along the `cell_gcp` dimension. On the other hand, the geographical variables (`lat_gcp` and `lon_gcp`) vary along both the `row_gcp` and `cell_gcp` dimensions.

It is mandatory that values are monotonically ordered in the `index_row_gcp` and `index_cell_gcp` variables.

Some metadata variables (such as incidence angle in SAR or view/sun angles in visible products) can be defined on the GCP grid (using gcp dimensions) rather than on the product variable grid.

For these cases, the following dimension and coordinate variables is used:

```
dimensions:
    time = 1 ;
    row = 2030 ;
    cell = 2361 ;
    row_gcp = 21 ;
    cell_gcp = 25 ;
variables:
    double time(time) ;
```

```
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
        time:axis = ‘‘T’’ ;
    float lat_gcp(row_gcp, cell_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
        lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
    float lon_gcp(row_gcp, cell_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
        lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
    int index_row_gcp(row_gcp) ;
        index_row_gcp:long_name = "index of ground control points in row dimension" ;
        index_row_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                 value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
        index_cell_gcp:long_name = "index of ground control points in cell dimension" ;
        index_cell_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                  value (end of last pixel)" ;
```

The dimension of a variable is (time, row, cell):

```
    ubyte variable(time, row, cell) ;
        variable:_FillValue = 255UB ;
        variable:long_name = "variable long name" ;
        variable:standard_name = "variable standard name" ;
        variable:units = "variable unit" ;
        variable:add_offset = offset (float) ;
        variable:scale_factor = scale factor (float) ;
        variable:valid_min = 0UB ;
        variable:valid_max = 254UB ;
```

3.5.4 Time dependent variables (in 1D)

This data pattern corresponds to any fixed platform, drifting buoy or along-track sensors such as altimeters. In this case where data are not gridded but vectorized along a time axis, the mapping between data and geographical coordinates is materialized by a 1D array of GCPs.

One dimension is required for GCPs (`time_gcp`), as well as two variables for geographical coordinates (`lat_gcp` and `lon_gcp`) and one variables for data coordinates (`index_time_gcp`).

The `lat_gcp`, `lon_gcp` and `index_time_gcp` variables only vary along the `time_gcp` dimension.

It is mandatory that values are monotonically ordered in the `index_time_gcp` variable (ascending chronological order highly recommended).

Latitude and longitude are defined for each time GCP for moving sensors.

For these cases, the following dimension and coordinate variables shall be used:

```
dimensions:
   time = 1231 ;
   station = 1;
   time_gcp = 123 ;
variables:
   double time(time_gcp) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z";
       time:calendar = "standard" ;
       time:axis = ‘‘T’’;
   float lat_gcp(time_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
       lat_gcp:comment = "geographical coordinates, WGS84 projection" ;
   float lon_gcp(time_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
       lon_gcp:comment = "geographical coordinates, WGS84 projection" ;
   int index_time_gcp(time_gcp) ;
       index_time_gcp:long_name = "index of ground control points in time dimension" ;
       index_time_gcp:comment = "index goes from 0 (first pixel) to value dimension" ;
```

The dimensions of a variable will then be (time):

```
   ubyte variable(time) ;
       variable:_FillValue = 255UB ;
       variable:long_name = ‘‘variable long name’’ ;
       variable:standard_name = ‘‘variable standard name’’ ;
       variable:unit = ‘‘variable unit" ;
       variable:add_offset = offset (float) ;
       variable:scale_factor = scale (float) ;
       variable:valid_min = 0UB ;
       variable:valid_max = 254UB ;
```

3.6 Geophysical variables

3.6.1 Naming convention

All these variables need to be uniquely and consistently named across all products if some products provide the same measured quantity. The rules behind the naming of these variables are as follows:

- avoid ambiguous naming such as 'u' and 'v' (for vector components). Explicit and unambiguous names are recommended.
- vectors should be stored as northward and eastward components (not module and direction). Explicit module variable can be provided in addition. For better integration in visualization tools, vector components must be named as:

  `northward_<geophysical quantity>`

eastward_<geophysical quantity>

- some instruments can only measure a single component of the current vector. Use one of the following prefixes:

  acrosstrack for a vector quantity measured along a satellite track (altimetry derived velocities)

  lineofsight for the radial component of the velocity (SAR derived velocities)

- avoid naming variables after the methodology ('mcc_current') or instrument('sar_doppler_current') used to retrieve it. Currents are a geophysical quantity and a proper naming needs to be found to describe the measured component.
- For geophysical quantity other than currents, use the conventions agreed in other projects (ex: GHRSST for sea surface temperature)
- use '_bias' and '_stddev' suffixes to express respectively the bias and standard deviation of a quantity
- use '_error' suffix to express the estimation error on a quantity

4 Examples of IDF

4.1 Lat/Lon grid datamodel

4.1.1 ECMWF wind

```
netcdf ECMWF_20141229T00Z_idf_00 {
dimensions:
   time = UNLIMITED ; // (1 currently)
   lat = 1409 ;
   lon = 2880 ;
   lat_gcp = 178 ;
   lon_gcp = 361 ;
variables:
   double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
   float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
   float lon_gcp(lon_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
   int index_lat_gcp(lat_gcp) ;
       index_lat_gcp:long_name = "index of ground control points in lat dimension" ;
       index_lat_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                value (end of last pixel)" ;
   int index_lon_gcp(lon_gcp) ;
       index_lon_gcp:long_name = "index of ground control points in lon dimension" ;
       index_lon_gcp:comment = "index goes from 0 (start of first pixel) to dimension
```

```
                                value (end of last pixel)" ;
   ubyte u10m(time, lat, lon) ;
       u10m:_FillValue = 255UB ;
       u10m:long_name = "u component of horizontal wind" ;
       u10m:standard_name = "eastward_wind" ;
       u10m:units = "m s-1" ;
       u10m:add_offset = -25.4 ;
       u10m:scale_factor = 0.2 ;
       u10m:valid_min = 0UB ;
       u10m:valid_max = 254UB ;
   ubyte v10m(time, lat, lon) ;
       v10m:_FillValue = 255UB ;
       v10m:long_name = "v component of horizontal wind" ;
       v10m:standard_name = "northward_wind" ;
       v10m:units = "m s-1" ;
       v10m:add_offset = -25.4 ;
       v10m:scale_factor = 0.2 ;
       v10m:valid_min = 0UB ;
       v10m:valid_max = 254UB ;
// global attributes:
   :idf_granule_id = "ECMWF_20141229T00Z" ;
   :time_coverage_start = "2014-12-28T22:30:00.000000Z" ;
   :time_coverage_end = "2014-12-29T01:30:00.000000Z" ;
   :idf_subsampling_factor = 0 ;
   :idf_spatial_resolution = 13875. ;
   : spatial_resolution_units = ‘‘m’’ ;
   :title = "ECMWF wind" ;
   :institution = "Cep;
   OceanDataLab" ;
   :comment = "" ;
   :id = "ECMWF_20141229T00Z_idf_00" ;
   :product_version = "1.0" ;
   :geospatial_lat_max = "88" ;
   :geospatial_lat_min = "-88" ;
   :geospatial_lon_max = "180" ;
   :geospatial_lon_min = "-180" ;
   :creator_email = ‘‘contact@oceandatalab.com’’
}
```

4.1.2 GlobCurrent current L4

```
netcdf \20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0_idf_00 {
dimensions:
   time = UNLIMITED ; // (1 currently)
   lat = 1280 ;
   lon = 2880 ;
   lat_gcp = 161 ;
   lon_gcp = 361 ;
variables:
   double time(time) ;
```

```
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(lat_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(lon_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_lat_gcp(lat_gcp) ;
        index_lat_gcp:long_name = "index of ground control points in lat dimension" ;
        index_lat_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                 value (end of last pixel)" ;
    int index_lon_gcp(lon_gcp) ;
        index_lon_gcp:long_name = "index of ground control points in lon dimension" ;
        index_lon_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                 value (end of last pixel)" ;
    ubyte eastward_geostrophic_current_velocity(time, lat, lon) ;
        eastward_geostrophic_current_velocity:_FillValue = 255UB ;
        eastward_geostrophic_current_velocity:long_name = "Absolute geostrophic velocity:
                                                           zonal component" ;
        eastward_geostrophic_current_velocity:units = "m s-1" ;
        eastward_geostrophic_current_velocity:add_offset = -5.08 ;
        eastward_geostrophic_current_velocity:scale_factor = 0.04 ;
        eastward_geostrophic_current_velocity:valid_min = 0UB ;
        eastward_geostrophic_current_velocity:valid_max = 254UB ;
    ubyte northward_geostrophic_current_velocity(time, lat, lon) ;
        northward_geostrophic_current_velocity:_FillValue = 255UB ;
        northward_geostrophic_current_velocity:long_name =  "Absolute geostrophic velocity:
                                                             meridian component" ;
        northward_geostrophic_current_velocity:units = "m s-1" ;
        northward_geostrophic_current_velocity:add_offset = -5.08 ;
        northward_geostrophic_current_velocity:scale_factor = 0.04 ;
        northward_geostrophic_current_velocity:valid_min = 0UB ;
        northward_geostrophic_current_velocity:valid_max = 254UB ;
// global attributes:
    :idf_granule_id = "20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0" ;
    :time_coverage_start = "2014-12-28T12:00:00.000000Z" ;
    :time_coverage_end = "2014-12-29T12:00:00.000000Z" ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 13875. ;
    :idf_spatial_resolution_units = "m" ;
    :title = "Globcurrent geostrophic surface velocity" ;
    :institution = "CLS; OceanDataLab" ;
    :comment = "Geostrophic current from mapped ADT altimeters ... " ;
    :id = "20141229000000-GLOBCURRENT-L4-CURgeo_0m-ALT_OI-v02.0-fv01.0_idf_00" ;
    :product_version = "1.0" ;
```

```
   :geospatial_lat_max = "90" ;
   :geospatial_lat_min = "-90" ;
   :geospatial_lon_max = "180" ;
   :geospatial_lon_min = "-180" ;
   :sensor = ‘‘HY-2A; Jason-2; Altika; Cryosat-2’’
   :creator_email = ‘‘contact@oceandatalab.com’’;
}
```

4.1.3 L4 SST Odyssea

```
netcdf \20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0_idf_00 {
dimensions:
   time = UNLIMITED ; // (1 currently)
   lat = 1750 ;
   lon = 2250 ;
   lat_gcp = 36 ;
   lon_gcp = 46 ;
variables:
   double time(time) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
   float lat_gcp(lat_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
   float lon_gcp(lon_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
   int index_lat_gcp(lat_gcp) ;
       index_lat_gcp:long_name = "index of ground control points in lat dimension" ;
       index_lat_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                             value (end of last pixel)" ;
   int index_lon_gcp(lon_gcp) ;
       index_lon_gcp:long_name = "index of ground control points in lon dimension" ;
       index_lon_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                             value (end of last pixel)" ;
   ubyte analysed_sst(time, lat, lon) ;
       analysed_sst:_FillValue = 255UB ;
       analysed_sst:long_name = "analysed sea surface temperature" ;
       analysed_sst:standard_name = "sea_surface_temperature" ;
       analysed_sst:units = "K" ;
       analysed_sst:add_offset = 271.05 ;
       analysed_sst:scale_factor = 0.15 ;
       analysed_sst:valid_min = 0UB ;
       analysed_sst:valid_max = 254UB ;
// global attributes:
   :idf_granule_id = "20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0" ;
```

```
    :time_coverage_start = "2014-12-27T12:00:00.000000Z" ;
    :time_coverage_end = "2014-12-28T12:00:00.000000Z" ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 2220. ;
    :idf_spatial_resolution_units = "m" ;
    :title = "ODYSSEA Sea Surface temperature Analysis in Aghulas" ;
    :institution = "Ifremer; OceanDataLab " ;
    :comment = "Analysed Sea Surface temperature computed using optimal interpolation with
                Odyssea processor 2.0" ;
    :id = "20141228-IFR-L4_GHRSST-SSTfnd-ODYSSEA-SAF_002-v2.0-fv1.0_idf_00";
    :product_version = "1.0" ;
    :geospatial_lat_max = "-10" ;
    :geospatial_lat_min = "-45" ;
    :geospatial_lon_max = "10" ;
    :geospatial_lon_min = "55" ;
    :creator_email = ‘‘contact@oceandatalab.com’’;
    :sensor = ‘‘AVHRR;AVHRR;MODIS_A;AATSR’’;
    :platform = ‘‘METOP-A; NOAA18;NOAA19;AQUA;ENVISAT’’;
}
```

4.2 Projected x/y grid datamodel

4.2.1 Sea ice concentration AMSR

```
netcdf asi-AMSR2-n6250-20141201-v5_idf_00 {
dimensions:
    time = UNLIMITED ; // (1 currently)
    y = 1792 ;
    x = 1216 ;
    y_gcp = 113 ;
    x_gcp = 77 ;
variables:
    double time(time) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(y_gcp, x_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(y_gcp, x_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_y_gcp(y_gcp) ;
        index_y_gcp:long_name = "index of ground control points in y dimension";
        index_y_gcp:comment = "index goes from 0 (start of first pixel) to dimension value
                               (end of last pixel)" ;
    int index_x_gcp(x_gcp) ;
```

```
        index_x_gcp:long_name = "index of ground control points in x dimension";
        index_x_gcp:comment = "index goes from 0 (start of first pixel) to dimension value
                              (end of last pixel)" ;
    ubyte sea_ice_concentration(time, y, x) ;
        sea_ice_concentration:_FillValue = 255UB ;
        sea_ice_concentration:long_name = "ASI Ice Concentration" ;
        sea_ice_concentration:units = "percent" ;
        sea_ice_concentration:add_offset = 0. ;
        sea_ice_concentration:scale_factor = 0.393700787401575 ;
        sea_ice_concentration:valid_min = 0UB ;
        sea_ice_concentration:valid_max = 254UB ;
// global attributes:
    :idf_granule_id = "asi-AMSR2-n6250-20141201-v5" ;
    :time_coverage_start = "2014-12-01T00:00:00.000000Z" ;
    :time_coverage_end = "2014-12-02T00:00:00.000000Z" ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 6250. ;
    :idf_spatial_resolution_units = "m" ;
}
```

4.3 Swath datamodel

4.3.1 SST L2 MODIS

```
netcdf A2014290131500.L2_LAC_SST_idf_00 {
dimensions:
    time = 1 ;
    row = 2030 ;
    cell = 2361 ;
    row_gcp = 21 ;
    cell_gcp = 25 ;
variables:
    double time(time) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(row_gcp, cell_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(row_gcp, cell_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_row_gcp(row_gcp) ;
        index_row_gcp:long_name = "index of ground control points in row dimension" ;
        index_row_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
```

```
        index_cell_gcp:long_name = "index of ground control points in cell dimension" ;
        index_cell_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                 value (end of last pixel)" ;
    ubyte denoised_sst(time, row, cell) ;
        denoised_sst:_FillValue = 255UB ;
        denoised_sst:long_name = "denoised sea surface temperature" ;
        denoised_sst:standard_name = "sea_surface_temperature" ;
        denoised_sst:units = "K" ;
        denoised_sst:add_offset = 283.200097855866 ;
        denoised_sst:scale_factor = 0.0511811023622047 ;
        denoised_sst:valid_min = 0UB ;
        denoised_sst:valid_max = 254UB ;
// global attributes: :idf_granule_id = "A2014290131500.L2_LAC_SST" ;
    :time_coverage_start = "2014-10-17T13:15:08.000000Z" ;
    :time_coverage_end = "2014-10-17T13:20:07.000000Z" ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 1000. ;
    :idf_spatial_resolution_units = "m" ;
 }
```

4.3.2 Sea Surface Roughness, Sentinel-1

```
netcdf s1a-ew-grd-hh-20141202t183501-20141202t183605-003544-0042c3-001-F64E_idf_00 {
dimensions:
    time = 1 ;
    row = 5339 ;
    cell = 5181 ;
    row_gcp = 23 ;
    cell_gcp = 21 ;
variables:
    double time(time) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
    float lat_gcp(row_gcp, cell_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(row_gcp, cell_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
        lon_gcp:units = "degrees_east" ;
    int index_row_gcp(row_gcp) ;
        index_row_gcp:long_name = "index of ground control points in row dimension" ;
        index_row_gcp:comment = "index goes from 0 (start of first pixel) to dimension
                                value (end of last pixel)" ;
    int index_cell_gcp(cell_gcp) ;
        index_cell_gcp:long_name = "index of ground control points in cell dimension" ;
        index_cell_gcp:comment = "index goes from 0 (start of first pixel) to dimension
```

```
                                value (end of last pixel)" ;
   ubyte sea_surface_roughness(time, row, cell) ;
       sea_surface_roughness:_FillValue = 255UB ;
       sea_surface_roughness:long_name = "sea surface roughness" ;
       sea_surface_roughness:units = "1" ; sea_surface_roughness:add_offset = 0. ;
       sea_surface_roughness:scale_factor = 0.0078740157480315 ;
       sea_surface_roughness:valid_min = 0UB ;
       sea_surface_roughness:valid_max = 254UB ;
// global attributes:
   :idf_granule_id = "s1a-ew-grd-hh-20141202t183501-20141202t183605-003544-0042c3-001-F64E
                    ";
   :time_coverage_start = "2014-12-02T18:35:01.251000Z" ;
   :time_coverage_end = "2014-12-02T18:36:05.342000Z" ;
   :idf_subsampling_factor = 0 ;
   :idf_spatial_resolution = 80. ;
   :idf_spatial_resolution_units = "m" ;
}
```

4.4 Trajectory datamodel

No subsampling has been performed yet on trajectories as spatial resolution is difficult to define. It will be done for the next version of the document.

4.4.1 Surface Drifter from AOML

```
netcdf AOML_2015_12_139922_idf_00 {
dimensions:
   time = 123 ;
   time_gcp = 123 ;
variables:
   double time(time_gcp) ;
       time:long_name = "time" ;
       time:standard_name = "time" ;
       time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
       time:calendar = "standard" ;
   float lat_gcp(time_gcp) ;
       lat_gcp:long_name = "ground control points latitude" ;
       lat_gcp:standard_name = "latitude" ;
       lat_gcp:units = "degrees_north" ;
   float lon_gcp(time_gcp) ;
       lon_gcp:long_name = "ground control points longitude" ;
       lon_gcp:standard_name = "longitude" ;
       lon_gcp:units = "degrees_east" ;
   int index_time_gcp(time_gcp) ;
       index_time_gcp:long_name = "index of ground control points in timedimension" ;
       index_time_gcp:comment = "index goes from 0 (first pixel) to valuedimension" ;
   ubyte current(time) ;
       current:_FillValue = 255UB ;
       current:unittype = "m/s" ;
       current:add_offset = 0.02351f ;
```

```
        current:scale_factor = 0.003844449f ;
        current:valid_min = 0UB ;
        current:valid_max = 254UB ;
    ubyte temp(time) ;
        temp:_FillValue = 255UB ;
        temp:unittype = "K" ;
        temp:add_offset = 302.565f ;
        temp:scale_factor = 0.006456693f ;
        temp:valid_min = 0UB ;
        temp:valid_max = 254UB ;
// global attributes:
    :idf_granule_id = "AOML_2015_12_139922" ;
    :time_coverage_start = "2015-12-01T06:00:00.000000Z" ;
    :time_coverage_end = "2015-12-31T18:00:00.000000Z" ;
    :platform_code = "139923" ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 1.e+07f ;
    :idf_spatial_resolution_units = "m" ;
    :title = "Temperature and curent from SVP data filtered by AOML" ;
    :institution = "AOML; OceanDataLaba " ;
    :comment = "Positions have been filtered using Lumpkin algorithm" ;
    :id = "AOML_2015_12_139922_idf_00" ;
    :product_version = "1.0" ;
    :geospatial_lat_max = "-9.00" ;
    :geospatial_lat_min = "-9.792" ;
    :geospatial_lon_min = "179.917" ;
    :geospatial_lon_max = "-178.3" ;
    :creator_email = ‘‘contact@oceandatalab.com’’;
    :platform_code = ‘‘139923’’;
}
```

4.4.2 L3 along track Jason 2 SLA

```
netcdf dt_global_j2_sla_vfec_20151201_20160610_c0273_p024_idf_00 {
dimensions:
    time = 1201 ;
    time_gcp = 1201 ;
variables:
    double time(time_gcp) ;
        time:long_name = "time" ;
        time:standard_name = "time" ;
        time:units = "seconds since 1970-01-01T00:00:00.000000Z" ;
        time:calendar = "standard" ;
  float lat_gcp(time_gcp) ;
        lat_gcp:long_name = "ground control points latitude" ;
        lat_gcp:standard_name = "latitude" ;
        lat_gcp:units = "degrees_north" ;
    float lon_gcp(time_gcp) ;
        lon_gcp:long_name = "ground control points longitude" ;
        lon_gcp:standard_name = "longitude" ;
```

```
        lon_gcp:units = "degrees_east" ;
    int index_time_gcp(time_gcp) ;
        index_time_gcp:long_name = "index of ground control points in timedimension" ;
        index_time_gcp:comment = "index goes from 0 (first pixel) to valuedimension" ;
    ubyte SLA(time) ;
        SLA:_FillValue = 255UB ;
        SLA:unittype = "m" ;
        SLA:add_offset = -0.451f ;
        SLA:scale_factor = 0.003622047f ;
        SLA:valid_min = 0UB ;
        SLA:valid_max = 254UB ;
// global attributes:
    :idf_granule_id = "dt_global_j2_sla_vfec_20151201_20160610_c0273_p024" ;
    :time_coverage_start = "2015-12-01T00:17:42.520577Z" ;
    :time_coverage_end = "2015-12-01T01:00:51.112577Z" ;
    :cycle = 273 ;
    :orbit = 24 ;
    :idf_subsampling_factor = 0 ;
    :idf_spatial_resolution = 1.e+07f ;
    :idf_spatial_resolution_units = "m" ;
    :title = "Sea Level Anomaly" ;
    :institution = "AVISO; OceanDataLaba " ;
    :comment = "" ;
    :id = "dt_global_j2_sla_vfec_20151201_20160610_c0273_p024" ;
    :product_version = "1.0" ;
    :geospatial_lat_max = "61" ;
    :geospatial_lat_min = "-61" ;
    :geospatial_lon_min = "-92.05706" ;
    :geospatial_lon_max = "-9.022436" ;
    :creator_email = ‘‘contact@oceandatalab.com’’;
    :sensor = ‘‘Jason2’’;
}
```

5 Annex

5.1 Open points

5.1.1 Flexibility to use non-IDF netCDF-4 data

Corresponding section: 2.2 NetCDF-4 classic file format
NetCDF-4 format is already used by some data producers and the number of NetCDF-4 formatted data will probably increase in the future. A direct streaming of such data instead of converting them into IDF would avoid data duplication and facilitate data ’ingestion’ into SEAScope. It is the general idea but it has to be seen case by case because of potential blocking points:

- Data need a special pre-processing preferably done off-line.
- High resolution data need multiresolution support
- Data geolocation pattern is outside the scope of IDF geolocation patterns

Some other blocking points need to be investigated along with the implementation of SEAScope data streaming service:

- Storage data type compliant with required fast access and streaming?
- Multiple granules in a file? Multiple files for a granule?

5.1.2 NetCDF-4 specific settings (chunk size, digit precision and data compression)

Corresponding section: 2.2 NetCDF-4 classic file format
NetCDF-4 format has some settings for optimizing data access and reducing file size. For now, no specific chunk size nor data compression are used and the geophysical variables are simply stored with 8 bits without digit precision (hence the use of a scale factor and an add offset). These settings will be investigated along with the implementation of SEAScope data streaming service.

5.1.3 Multiresolution guidelines

Corresponding section: 2.3 Multiresolution support
We should indicate clear rules for multiresolution:

- When is multiresolution needed?
- If needed, until which spatial sampling should IDF files be generated?
- How to define spatial spacing in meters? (eg. irregular trajectory like drifters)

5.1.4 Tiling from GCPs and GCPs density

Corresponding section: 2.4 Geolocation with GCP
GCPs serve two main purposes: data geolocation and virtual tiling. For now, GCPs have been used with a focus on data geolocation since SEAScope tiling is not effective yet. The use of GCPs for tiling will be addressed along with the implementation of SEAScope data tiling. Then, some guidelines about GCPs will be added in the document. It will probably add some constraints about where to position GCPs in the data space. Also, it makes more sense to wait for tiling before giving definitive rules about GCPs density.

5.1.5 Unique product identifier

Corresponding section: 3 IDF NetCDF-4 file structure
The product unique identifier is used to fill in the global attribute id described in table 6. If a product unique identifier is already defined in the original product, one should use it to redefine an id similar to:

`<Original id>-<Subsampling index>`

If no id is defined in the original file, one should define a product unique identifier using the following rules.

`<Producer>-<Processing Level>-<Product String>-v<Product Version>-<subsamplingindex>`

The definitions of the components match the definitions from the file naming convention, found in previous sections. The component `<Product Version>` is used to distinguish different versions of the same product and should be of the form x.y where x is the major and y is the minor version. The component `<Producer>` corresponds to the producing agency using the short name or acronym. Unique product identifier has not been defined yet in IDF global attributes. Recommended practices regarding its definition may evolve when this issue is tackled.

5.1.6 Data Model

Corresponding section: 3.5 Space Time Coordinate variables
Section 3.5 is giving an overview of possible datamodels. This section is going to evolve when various types of data are converted into IDF. New datamodel shall be defined such as Time Series for fixed sensor, as the only coordinate that varies is the time. In this case there is no need to store longitude and latitude as a function of time.