EGO gliders NetCDF format reference manual

NetCDF conventions
Reference tables and files distribution

Version 1.12

February 6th, 2023

doi http://dx.doi.org/10.13155/34980

Table of contents

History

Version	Date	Comment
0.9	08/10/2012	TC: initialization of the document based o EGO user's manual version 1.2
0.99	15/10/2012	TC : updates after Paris Groom meeting §2.3.3 : Profile, phase and direction management §2.3.4 : Positioning method §2.3.5 : add a coordinates attribute to <PARAM> §2.4.4 : Configuration parameters §2.4.6 : calibration : note on derived parameters such as PSAL, note on pre-deployment calibrations §2.5 : Gliders technical data §5 : Glossary, definitions
0.999	23/10/2012	GB : use TIME:units = "days since 1970-01-01T00:00:00Z"; instead of 1950 for compatibility with old version software such as ferret
0.9999	12/12/2012	TC : use TIME variable (seconds since 01/01/1970) and JULD (days since 01/01/1950) Remove <PARAM>_dm Remove uncertainty as an attribute Remove qc_indicator_attribute Remove EGO glider catalogue Remove configuration parameters chapter DERIVATION instead of CALIBRATION Add a data distribution chapter 7 Add a chapter 1.8 on CTD thermal lag error Manage technical data as standard variables
1.2	12/04/2016	JPR: updated to fit with the format (V1.2) generated at Coriolis.
1.2	23/06/2016	JPR, CG, TC: minor updates, creation of a DOI.
1.2	09/05/2017 11/05/2017	JPR: minor updates after Steve LOCH reviewed the document.
1.3	05/09/2017	JPR: updated to separate SENSOR an PARAMETER in glider metadata.
1.3	15/12/2017 03/01/2018 01/02/2018 22/02/2018	JPR: updated after Victor TURPIN reviewed the document.
1.3	27/04/2018 07/09/2018	JPR: checked and updated the list of mandatory variables. JPR: updated after Thierry CARVAL and Victor TURPIN reviewed the document.
1.3	21/02/2019	JPR: finalized.
1.3	31/01/2020	JPR: "IO", "TU" and "IM" added to reference table 4.
1.4	27/08/2020 03/11/2020 30/03/2021	JPR: Add "license" in global attributes Modify "distribution_statement" global attribute content Modify the list of global attributes that can be set by the user JPR: Reference to unit mapped to P06 SDN vocabulary (instead of P061 one) SeaDataNet sea areas referenced to C19 (instead of C16) Epoch time is "Time of the measurement in seconds since midnight, 1970-01-01." (not "noon"). JPR: Add the following global attributes: data_processing_chain_name data_processing_chain_version data_processing_chain_uri
1.5	13/09/2021	JPR: added a list of parameters specific to the EGO format (as part of Reference Table 3).
1.6	24/02/2022	JPR: updated Ref. Table 10.2 positioning methods. JPR: updated Ref. Table 27 sensor models (specific to EGO project).
1.7	24/03/2022	JPR: updated Ref. Table 27 sensor models (specific to EGO project).
1.8	09/05/2022	JPR: updated Ref. Table 27 sensor models (specific to EGO project).

1.9	20/07/2022	JPR: updated Ref. Tables 3, 25 and 27 (for URANINE and RHODAMINE parameters).
1.10	22/09/2022	JPR: updated Ref. Tables 3 and 27 (with BETA_BACKSCATTERING_SCALED parameter and SEAOWL_UV_A sensor model).
1.11	18/01/2023	JPR: definition of EGO profile file format V2.0.
1.12	06/02/2023	JPR: updated Ref. Table 24 (added UNIVERSITY_OF_WASHINGTON).

1 EGO gliders data-management principles

1.1 About EGO

Everyone's Gliding Observatories - EGO is dedicated to the promotion of the glider technology and its applications.

The EGO group promotes glider applications through coordination, training, liaison between providers and users, advocacy, and provision of expert advice.

We intend to favor oceanographic experiments and the operational monitoring of the oceans with gliders through scientific and international collaboration. We provide news, support, information about glider projects and glider data management, as well as resources related to gliders.

All EGO data are publicly available. More information about the project is available at: http://www.ego-network.org

1.2 About this document

This document specifies the NetCDF file format of EGO-gliders that is used to distribute glider data, metadata and technical data. It documents the standards used therein; this includes naming conventions as well as metadata content.

It was initiated in October 2012, based on OceanSITES, Argo and ANFOG user's manuals.

1.3 EGO data management structure and data access

The data flow within EGO is carried out through four organizational units: glider operators, PIs, DACs and GDACs.

The **glider operators** are the group of people in charge of the preparation, deployment, piloting and recovery of the glider. They deliver raw data and metadata to the PI and/or Data Assembly Center (DAC).

The **Principal Investigator (PI)**, typically a scientist at a research institution, maintains the observing platform and the sensors that deliver the data. He or she is responsible for providing the data and all auxiliary information to a **Data Assembly Center (DAC)**.

The **DAC** assembles EGO-compliant files from this information and delivers these to the two **Global Data Assembly Centers (GDACs)**, where they are made publicly available.

The **GDAC** distributes the best copy of the data files. When a higher quality data file (e.g. calibrated data) is available, it replaces the previous version of the data file.
The user can access the data at either GDAC, cf. section "GDAC organization".

1.4 User Obligations

A user of EGO data is expected to read and understand this manual and the documentation about the data as contained in the "attributes" of the NetCDF data files, as these contain essential information about data quality and accuracy.

A user of EGO data must comply with the requirements set forth in the attributes "distribution_statement" and "citation" of the NetCDF data files.

Unless stated otherwise, a user must acknowledge use of EGO data in all publications

and products where such data are used, preferably with the following standard sentence:

"These data were collected and made freely available by the international EGO project and the national programs that contribute to it."

1.5 Disclaimer

EGO data are published without any warranty, express or implied.

The user assumes all risk arising from his/her use of EGO data.

EGO data are intended to be research-quality and include estimates of data quality and accuracy, but it is possible that these estimates or the data themselves contain errors.

It is the sole responsibility of the user to assess if the data are appropriate for his/her use, and to interpret the data, data quality, and data accuracy accordingly.

EGO welcomes users to ask questions and report problems to the contact addresses listed in the data files or on the EGO internet page.

1.6 Further Information Sources and Contact Information

- EGO website: http://www.ego-network.org
- For further information about the benefits and distributing data onto the GTS, please refer to: http://www.jcommops.org/dbcp/gts or contact the EGO Project Office on webmaster@ego-network.org
- For information about unique numbering of EGO Gliders and Gliders on the GTS see: http://www.wmo.int/pages/prog/amp/mmop/wmo-number-rules.html

1.7 Useful links, tools

1.7.1 EGO data processing chain to generate EGO NetCDF files

The EGO glider data processing chain is available at:

- https://www.seanoe.org/data/00343/45402/

1.7.2 EGO file format checker

The EGO file format checker is a java software freely available at:

- http://www.coriolis.eu.org/Data-Products/Tools

2 Gliders NetCDF data format version 1.4

EGO uses the NetCDF (network Common Data Form) system, a set of software libraries and machine-independent data formats. Our implementation of NetCDF is based on the community-supported Climate and Forecast (CF) specification, which supplies a standard vocabulary and some metadata conventions.

EGO layers several more conventions above the CF standard. These are intended to make it easier to share in-situ data, to make it simpler for the GDACs to aggregate data from multiple sites, and to ensure that the data can be created and understood by the basic NetCDF utilities.

- EGO includes standard terms for the short name of both coordinate and data variables (measurements).
- File names are created using a standard, described in section 6.1.

An EGO data file contains measurements such as temperature and salinity, continuously performed at different levels on a glider, as well as engineering data recorded onboard and complete location, time informations.

The requirements are drawn almost exclusively from the NetCDF Style Guide:

- Units are compliant with CF/COARDS/UDUNITS;
- The time parameter is encoded as recommended by COARDS and CF;
- Parameters are given standard names from the CF table;
- Where time is specified as an attribute, the ISO8601 standard is used.

For more information on NetCDF, UDUNITS, COARDS, CF and ISO8601 see:

- NetCDF: https://www.unidata.ucar.edu/software/netcdf/docs/index.html
- CF: http://cfconventions.org
- UDUNITS: http://www.unidata.ucar.edu/software/udunits/
- COARDS: http://www.ferret.noaa.gov/noaa_coop/coop_cdf_profile.html
- ISO8601: http://en.wikipedia.org/wiki/ISO_8601

Note on format version

Since 2021, the EGO valid data format version is **1.4**.

The User's manual may be updated with clarifications, recommendations, additional optional attributes without changing the data format version.

2.1 Data file dimensions

EGO glider data are recorded as time-series. The TIME dimension is the main dimension for an EGO glider data file.

Name	Example	Comment
TIME	TIME = unlimited	Number of time steps.
TIME_GPS	TIME_GPS = 501	Number of GPS fixes.
TIME_CURRENT	TIME_CURRENT = 10	Number of sub-surface current estimates (this dimension is present only if such estimates are stored in the EGO file).
DATE_TIME	DATE_TIME = 14;	This dimension is the length of an ASCII date and time value. Date_time convention is : YYYYMMDDHHMISS YYYY : year MM : month DD : day HH : hour of the day (as 0 to 23) MI : minutes (as 0 to 59) SS : seconds (as 0 to 59) Date and time values are always in universal time coordinates (UTC). Examples : 20010105172834 : January 5th 2001 17:28:34 19971217000000 : December 17th 1997 00:00:00
STRING4096 STRING1024 STRING256 STRING128 STRING64 STRING32 STRING16 STRING8 STRING4 STRING2	STRING4096 = 4096; STRING1024 = 1024; STRING256 = 256; STRING128 = 128; STRING64 = 64; STRING32 = 32; STRING16 = 16; STRING8 = 8; STRING4 = 4; STRING2 = 2;	String dimensions from 2 to 4096.

2.2 Global attributes

The global attribute section of a NetCDF file contains metadata that describes the contents of the file overall, and allows for data discovery. All fields should be human-readable, and should be of character type, not numeric, even if the information content is a number. EGO recommends that all of these attributes be used and contain meaningful information unless there are technical reasons rendering this impossible. However, files that do not at least contain the attributes listed as "mandatory" will not be considered EGO-compliant. In EGO, global attribute names are in lower-case letters (except "Convention").

Global attributes can be thought of as conveying five kinds of information:

- What: what are the data in this data file;
- Where: the spatial coverage of the data;
- When: the temporal coverage of the data;
- Who: who produced the data;
- How: how were the data produced and made available.

The global attributes specification follows the recommendations of Unidata NetCDF Attribute Convention for Dataset Discovery, at:

http://www.unidata.ucar.edu/software/netcdf-java/formats/DataDiscoveryAttConvention.html

If the EGO format file is generated by the Coriolis processing chain some global attributes are set by the software. In the following table, the last column specifies the global attributes that can/must be filled by the user.

Name	Example	Definition	Set by
WHAT			
data_type	data_type="EGO glider time-series data"	Type of data contained in the file. The list of acceptable data types is in reference table 1. Example: "EGO glider time-series data" **This attribute is mandatory.**	software
format_version	format_version="1.4"	File format version Example: "1.4". **This attribute is mandatory.**	software
platform_code	platform_code="pytheas"	Glider unique code within EGO project. The use of lower case is recommended. Only basic ASCII letters, numbers and "_" (no accent) **This attribute is mandatory.**	user
date_update	date_update="2006-04-11T08:35:00Z"	File update or creation date (UTC). See note on time format below. **This attribute is mandatory.**	software
wmo_platform_code	wmo_platform_code="61864"	WMO (World Meteorological Organization) identifier. This platform number is unique within the EGO project. Example: "61864" for pytheas glider.	user
source	source="Glider observation"	Method of production of the original data. For EGO data, use one of the following: "Shipborne observation", "Glider observation"	software
license	License="https://creativecommons.org/licenses/by-nc/4.0/"	URL to the creative commons privacy policy.	software

history	history="2008-12-10T09:35:36Z Written by MATLAB script seagliderFV.m v1.2 2010-12-07T10:11:00Z data calibrated, controlled and sent to DAC, Laurent Mortier "	Audit trail for modifications to the original data. It should contain a separate line for each modification, with each line beginning with a timestamp, and including user name, modification name, and modification arguments. The time stamp should follow the format outlined in the note on time formats below.	software
data_mode	data_mode="R"	Indicates if the file contains real-time, provisional, mixed or delayed-mode data. The list of valid data modes is in reference table 19. Examples: - data_mode = "R" means that all parameters of the file have a PARAMETER_DATA_MODE = 'R' - data_mode = "P" means that all parameters of the file have a PARAMETER_DATA_MODE = 'R' or 'P' (with at least one of them with a PARAMETER_DATA_MODE = 'P') - data_mode = "A" means that all parameters of the file have a PARAMETER_DATA_MODE = 'R', 'P' or 'A' (with at least one of them with a PARAMETER_DATA_MODE = 'A') - etc… **This attribute is mandatory.**	software
quality_index	quality_index="excellent"	Code value valid for the whole data file: "unknown quality" "excellent" (no known problems, regular quality checking) "probably good" (occasional problems, validation phase) "extremely suspect", frequent problems	software
references	references="http://www.ego-network.org/"	Published or web-based references that describe the data or methods used to produce it. Include a reference to EGO and a project-specific reference if appropriate.	user
comment	comment="This deployment was performed during the Latex exercise"	Miscellaneous information about the data or methods used to produce it. Any free-format text is appropriate.	user
Conventions	Conventions="CF-1.4 EGO-1.4"	Name of the conventions followed by the data file.	software
netcdf_version	netcdf_version="3.6"	Netcdf version used for the data file	software
title summary	title="Pytheas glider data on Latex deployment" summary="Oceanographic glider data from Pytheas glider deployed in gulf of Lion, North-West Mediterranean sea, in 2010. Measured properties: temperature, salinity, oxygen, turbdity."	Free-format text describing the data file. The display of these two attributes together should allow data discovery for a human reader. "title": title of the data file. Use the file name if in doubt. "summary": a longer description of the data file. A paragraph of up to 100 words is appropriate.	user user

abstract	abstract="Glider Ocean observations have been collected by EGO since 2005 and are ongoing. EGO is Everyone's Gliding Observatory. The data are Slocum, SeaGlider or Spray gliders fitted with a wide range of sensors measuring temperature, salinity, oxygen, currents, chlorophyll, nitrate, cdom and other bio-geo-chemical data. This NetCDF file was created by EGO using the EGO filenaming convention version 1 and the EGO netCDF user's manual version 1.4"	Paragraph describing the data file: type of data contained, how it was created, who collected it, what instruments were used, what data formatting convention was used, etc.	user
keywords	keywords="Turbidity, Chlorophyll, Organic Matter, Oxygen, Fluorescence, Scattering, Water Temperature, Conductivity, Salinity"	Comma separated list of key words and phrases.	user
naming_authority id	naming_authority="EGO" id="GL_20100612_PYTHEAS_MooseT00_09_R.nc"	The "id" and "naming_authority" attributes are intended to provide a globally unique identification for each data file. For EGO data, use: naming_authority="EGO" and id=file name (without .nc suffix), which is designed to be unique. **Both attributes are mandatory.**	software software
cdm_data_type	cdm_data_type="Trajectory"	The "cdm_data_type" attribute gives the Unidata CDM (common data model) data type used by THREDDS. E.g. "Point", "Trajectory", "Station", "Radial", "Grid", "Swath". More: http://www.unidata.ucar.edu/projects/THREDDS/CDM/CDM-TDS.htm	software
WHERE			
area	area="North West Mediterranean Sea"	Geographical coverage Use vocabulary from SeaDataNet sea areas (C19). http://vocab.nerc.ac.uk/collection/C19/current/accepted/	user
geospatial_lat_min	geospatial_lat_min="59.8"	Southernmost valid latitude, a value between -90 and 90 degrees. This is calculated from the valid latitudes in the file. Decimal degrees	software
geospatial_lat_max	geospatial_lat_max="59.8"	Sorthernmost valid latitude, a value between -90 and 90 decimal degrees. This is calculated from the valid latitudes in the file.	software
geospatial_lon_min	geospatial_lon_min="-41.2"	The westernmost valid longitude, a value between -180 and 180 degrees. This is calculated from the valid longitudes in the file.	software
geospatial_lon_max	geospatial_lon_max="-41.2"	The easternmost valid longitude, a value between -180 and 180 decimal degrees. This is calculated from the valid longitudes in the file.	software

geospatial_vertical_min	geospatial_vertical_min="10.0"	Minimum valid depth or pressure for measurements. This is calculated from the valid depth or pressure in the file.	software
geospatial_vertical_max	geospatial_vertical_max="200"	Maximum valid depth or pressure for measurements. This is calculated from the valid depth or pressure in the file.	software
WHEN			
time_coverage_start	time_coverage_start="2010-07-01T00:00:00Z"	Start date of the data in UTC. See note on time format below.	software
time_coverage_end	time_coverage_end="2010-09-18T23:59:29Z"	Final date of the data in UTC. See note on time format below.	software
WHO			
institution	institution="CNRS-LOCEAN"	Preferably institution of the principal investigator.	user
institution_references	institution_references="http://www.nocs.uk"	References to principal investigator institution, the place to find all information on the data file (web-based, i.e. give URLs).	user
sdn_edmo_code	sdn_edmo_code="1042"	SeaDataNet EDMO code of the institution. EDMO is the "European Directory of Marine Organisations". http://seadatanet.maris2.nl/edmo/	user
authors	authors="Pierre Testor;Thierry Carval"	List of relevant persons involved in the creation of the data file (comma separated).	user
data_assembly_center	data_assembly_center="IF"	Data Assembly Center (DAC) in charge of this data file. The data_assembly_center are listed in reference table 4.	user
principal_investigator	principal_investigator="Laurent Mortier"	Name of the principal investigator in charge of the glider project.	user
principal_investigator_email	principal_investigator_email="Laurent.Mortier@upmc.fr"	Principal investigator's email address.	user
project_name	project_name=""	Name of the project which operates the profiling glider that performed the profile.	user
observatory	observatory="North west Mediterranean sea"	A geographical area monitored with a fleet of gliders.	user
deployment_code	deployment_code="MooseT00_19"	Deployment code. It is unique among EGO deployments. This code may be used as the local code in catalogues such as SeaDataNet Common Data Index (CDI).	user
deployment_label	deployment_label="Moose T00_19 summer 2010 deployment"	The deployment label, a free text to describe the deployment.	user
HOW			

distribution_statement	distribution_statement="EGO data are published without any warranty, express or implied. The user assumes all risk arising from his/her use of EGO data. EGO data are intended to be research-quality and include estimates of data quality and accuracy, but it is possible that these estimates or the data themselves contain errors. It is the sole responsibility of the user to assess if the data are appropriate for his/her use, and to interpret the data, data quality, and data accuracy accordingly. EGO welcomes users to ask questions and report problems to the contact addresses listed in the data files or on the EGO internet page."	Statement describing data distribution policy.	software
doi	doi="http://doi.org/10.17882/51141"	List of Data Object Identifiers (DOI) related to this data file (blank separated).	user
citation	citation="These data were collected and made freely available by the international EGO project and the national programs that contribute to it."	The citation to be used in publications using the data file.	user
update_interval	update_interval="daily"	Update interval for the file. The list of valid intervals is in reference table 6.	user
qc_manual	qc_manual="http://doi.org/10.13155/51485"	Contains the name of the manual that describes the quality control procedure. As of now, there is no separate QC manual, so the user's manual is the appropriate reference.	software
data_processing_chain_name	data_processing_chain_name = "EGO gliders data processing chain"	Contains the name of the processing chain used to generate the EGO file.	Software
data_processing_chain_version	data_processing_chain_version = "008a"	Contains the software version of the processing chain used to generate the EGO file.	Software
data_processing_chain_uri	data_processing_chain_uri = "https://doi.org/10.17882/45402"	Contains the Uniform Resource Identifier of the processing chain used to generate the EGO file.	Software

Note on time formats

Whenever time information is given in the global attributes, it ought to be a string of the format:

"YYYY-MM-DDThh:mm:ssZ" (i.e. year - month - day T hour : minute : second Z)

If higher resolution than seconds is needed, any number of decimal digits (“.s”) for the seconds is acceptable:

"YYYY-MM-DDThh:mm:ss.sZ"

In any case, the time must be in UTC. A capital “T” separates the date and the hour information. The string must end with a capital “Z”, an old indication of UTC. These formats are two (of many) described by ISO8601.

Examples:

- 2005-10-24T08:00:00Z
- 2008-01-01T22:50:02.031Z

2.3 Variables

NetCDF variables include data measured by instruments, parameters derived from the primary measurements, and coordinate variables, which may be nominal values, such as values for depth for instruments that do not directly record depth. The variable names are written in CAPITALIZED letters. Each variable has a specific set of attributes, some of which are mandatory.

The mandatory variables or attributes are in **bold characters in the following tables**.

2.3.1 Coordinate variables

The coordinate variables orient the data in time and space. For this purpose, they have an "axis" attribute defining that they point in X, Y, Z, and T dimensions.

Default values are not allowed in coordinate variables.

All attributes in this section except the "comment" are mandatory.

The Z axis may be represented as pressure, if, for example pressure is recorded directly by an instrument and the calculation of depth from pressure would cause a loss of information. Depth is strongly preferred, since it allows data to be used more directly.

Name	Definition	Comment
TIME	double **TIME**(TIME); TIME:long_name = "Epoch time"; TIME:standard_name = "time"; TIME:units = "seconds since 1970-01-01T00:00:00Z"; TIME:_FillValue = 9999999999; TIME:valid_min = 0; TIME:valid_max = 90000; TIME:comment = "<X>"; TIME:axis = "T"; TIME:ancilliary_variable = "TIME_QC"; TIME:sdn_parameter_urn = "SDN:P01::ELTMEP01"; TIME:sdn_uom_urn = "SDN:P06::UTBB"; TIME:glider_original_parameter_name = "<Y>";	Time of the measurement in seconds since midnight, 1970-01-01. Example: July 25, 2001, 19:14:00 is stored as 996088440. <X>: Any optional comment. <Y>: Report the glider variable name for TIME.
LATITUDE	double **LATITUDE**(TIME); LATITUDE:long_name = "Measurement latitude"; LATITUDE:standard_name = "latitude"; LATITUDE:units = "degree_north"; LATITUDE:_FillValue = 99999; LATITUDE:valid_min = -90; LATITUDE:valid_max = 90; LATITUDE:comment = "<X>"; LATITUDE:axis = "Y"; LATITUDE:ancilliary_variable = "POSITION_QC"; LATITUDE:reference = "WGS84"; LATITUDE:coordinate_reference_frame = "urn:ogc:crs:EPSG::4326"; LATITUDE:sdn_parameter_urn = "SDN:P01::ALATZZ01"; LATITUDE:sdn_uom_urn = "SDN:P06::DEGN"; LATITUDE:glider_original_parameter_name = "<Y>";	Latitude of the measurements. Units: degrees north; southern latitudes are negative. Example: 44.4991 for 44° 29' 56.76" N <X>: Any optional comment. <Y>: Report the glider variable name for LATITUDE.

LONGITUDE	double **LONGITUDE**(TIME); LONGITUDE:long_name = "Measurement longitude"; LONGITUDE:standard_name = "longitude"; LONGITUDE:units = "degree_east"; LONGITUDE:_FillValue = 99999; LONGITUDE:valid_min = -180; LONGITUDE:valid_max = 180; LONGITUDE:comment = "<X>"; LONGITUDE:axis = "X"; LONGITUDE:ancilliary_variable = "POSITION_QC"; LONGITUDE:reference = "WGS84"; LONGITUDE:coordinate_reference_frame = "urn:ogc:crs:EPSG::4326"; LONGITUDE:sdn_parameter_urn = "SDN:P01::ALONZZ01"; LONGITUDE:sdn_uom_urn = "SDN:P06::DEGE"; LONGITUDE:glider_original_parameter_name = "<Y>";	Longitude of the measurements. Unit: degrees east; western latitudes are negative. Example: 16.7222 for 16° 43' 19.92" E <X>: Any optional comment. <Y>: Report the glider variable name for LONGITUDE.

Note on latitude and longitude WGS84 datum

The latitude and longitude datum is WGS84. This is the default output of GPS systems.

EGO uses the EPSG coordinate reference system to describe geographical positions; the coordinate reference frame corresponding to WGS84 is : "urn:ogc:crs:EPSG::4326".

More on EPSG : http://www.epsg.org/

Note on TIME

By default, the time word represents the center of the data sample or averaging period.

2.3.2 Coordinate quality control variables

The coordinate variables have the same quality control variables as the data variables. If the quality control values are constant, the information is given in attributes of the coordinate variables. For details, see <PARAM>_QC in the section on data variables, and the note on quality control therein.

Name	Definition	Comment
TIME_QC	byte **TIME_QC**(TIME); TIME_QC:long_name = "Quality flag"; TIME_QC:conventions = "EGO reference table 2.1"; TIME_QC:_FillValue = -128; TIME_QC:valid_min = 0; TIME_QC:valid_max= 9; TIME_QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; TIME_QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value"	Quality flag for each TIME value.
POSITION_QC	byte **POSITION_QC**(TIME) POSITION_QC:long_name = "Quality flag"; POSITION_QC:conventions = "EGO reference table 2.1"; POSITION_QC:_FillValue = -128; POSITION_QC:valid_min = 0; POSITION_QC:valid_max= 9; POSITION_QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; POSITION_QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value"	Quality flag for each LATITUDE and LONGITUDE value.

2.3.3 GPS variables

The following variables are used to store the GPS fixes.

Name	Definition	Comment
TIME_GPS	double **TIME_GPS**(TIME_GPS); TIME_GPS:long_name = "Epoch time of the GPS fixes"; TIME_GPS:standard_name = "time"; TIME_GPS:units = "seconds since 1970-01-01T00:00:00Z"; TIME_GPS:_FillValue = 9999999999; TIME_GPS:valid_min = 0; TIME_GPS:valid_max = 90000; TIME_GPS:comment = "<X>"; TIME_GPS:axis = "T"; TIME_GPS:ancilliary_variable = "TIME_GPS_QC"; TIME_GPS:sdn_parameter_urn = "SDN:P01::ELTMEP01"; TIME_GPS:sdn_uom_urn = "SDN:P06::UTBB"; TIME_GPS:glider_original_parameter_name = "<Y>";	Time of the GPS fix in seconds since midnight, 1970-01-01. Example: July 25, 2001, 19:14:00 is stored as 996088440. <X>: Any optional comment. <Y>: Report the glider variable name for TIME_GPS.
LATITUDE_GPS	double **LATITUDE_GPS**(TIME_GPS); LATITUDE_GPS:long_name = "Gps fixed latitude"; LATITUDE_GPS:standard_name = "latitude"; LATITUDE_GPS:units = "degree_north"; LATITUDE_GPS:_FillValue = 99999; LATITUDE_GPS:valid_min = -90; LATITUDE_GPS:valid_max = 90; LATITUDE_GPS:comment = "<X>"; LATITUDE_GPS:axis = "Y"; LATITUDE_GPS:ancilliary_variable = "POSITION_GPS_QC"; LATITUDE_GPS:reference = "WGS84"; LATITUDE_GPS:coordinate_reference_frame = "urn:ogc:crs:EPSG::4326"; LATITUDE_GPS:sdn_parameter_urn = "SDN:P01::ALATZZ01"; LATITUDE_GPS:sdn_uom_urn = "SDN:P06::DEGN"; LATITUDE_GPS:glider_original_parameter_name = "<Y>";	Latitude of the GPS fix. Units: degrees north; southern latitudes are negative. Example: 44.4991 for 44° 29' 56.76" N <X>: Any optional comment. <Y>: Report the glider variable name for LATITUDE_GPS.
LONGITUDE_GPS	double **LONGITUDE_GPS**(TIME_GPS); LONGITUDE_GPS:long_name = "Gps fixed longitude"; LONGITUDE_GPS:standard_name = "longitude"; LONGITUDE_GPS:units = "degree_east"; LONGITUDE_GPS:_FillValue = 99999; LONGITUDE_GPS:valid_min = -180; LONGITUDE_GPS:valid_max = 180; LONGITUDE_GPS:comment = "<X>"; LONGITUDE_GPS:axis = "X"; LONGITUDE_GPS:ancilliary_variable = "POSITION_GPS_QC"; LONGITUDE_GPS:reference = "WGS84"; LONGITUDE_GPS:coordinate_reference_frame = "urn:ogc:crs:EPSG::4326"; LONGITUDE_GPS:sdn_parameter_urn = "SDN:P01::ALONZZ01"; LONGITUDE_GPS:sdn_uom_urn = "SDN:P06::DEGE"; LONGITUDE_GPS:glider_original_parameter_name = "<Y>";	Longitude of the GPS fix. Unit: degrees east; western latitudes are negative. Example: 16.7222 for 16° 43' 19.92" E <X>: Any optional comment. <Y>: Report the glider variable name for LONGITUDE_GPS.

2.3.4 GPS quality control variables

Name	Definition	Comment
TIME_GPS_QC	byte **TIME_GPS_QC**(TIME_GPS); TIME_GPS_QC:long_name = "Quality flag"; TIME_GPS_QC:conventions = "EGO reference table 2.1"; TIME_GPS_QC:_FillValue = -128; TIME_GPS_QC:valid_min = 0; TIME_GPS_QC:valid_max= 9; TIME_GPS_QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; TIME_GPS_QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value"	Quality flag for each TIME_GPS value.
POSITION_GPS_QC	byte **POSITION_GPS_QC**(TIME_GPS) POSITION_GPS_QC:long_name = "Quality flag"; POSITION_GPS_QC:conventions = "EGO reference table 2.1"; POSITION_GPS_QC:_FillValue = -128; POSITION_GPS_QC:valid_min = 0; POSITION_GPS_QC:valid_max= 9; POSITION_GPS_QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; POSITION_GPS_QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value";	Quality flag for each LATITUDE_GPS and LONGITUDE_GPS value.

2.3.5 Profile, phase and direction management

A glider regularly performs various phases such as surface, descent, inflexion, subsurface drift.

For each time stamp, the following variables indicate the phase of the glider at that time.

During ascent or descent phase, the glider performs vertical profiles.

A number is associated with each phase.

The first phase of the deployment is number 0: it is the surface drift before the first dive.

The phase number is increased by 1 for each new phase.

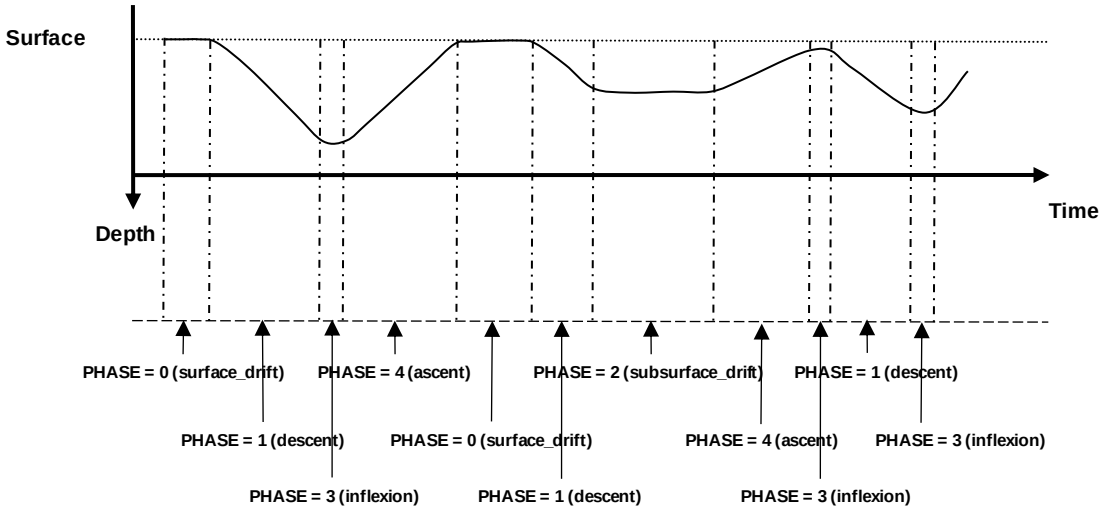

Figure 1: example of PHASE and PHASE_NUMBER assignment

Name	Definition	Comment
PHASE	byte **PHASE**(TIME); PHASE:long_name = "Glider trajectory phase code"; PHASE:conventions = "EGO reference table 9.2"; PHASE:_FillValue = -128; PHASE:flag_values = 0, 1, 2, 3, 4, 5, 6; PHASE:flag_meanings = "surface_drift descent subsurface_drift inflexion ascent grounded inconsistent"	Phase of the trajectory at that time, described in reference table 9.2.
PHASE_NUMBER	int **PHASE_NUMBER**(TIME); PHASE_NUMBER:long_name = "Glider trajectory phase number"; PHASE_NUMBER:_FillValue = 99999;	A number associated with the phase. The first phase number is 0. The phase number is increased at each phase change.

2.3.6 Positioning method

The positions reported in variables latitude and longitude are reported from various sources: GPS, Argos, interpolation.

Name	Definition	Comment
POSITIONING_METHOD	byte POSITIONING_METHOD(TIME); POSITIONING_METHOD:long_name = "Positioning method"; POSITIONING_METHOD:conventions = "EGO reference table 10.2"; POSITIONING_METHOD:_FillValue = -128; POSITIONING_METHOD:flag_values = 0, 1, 2; POSITIONING_METHOD:flag_meanings = "GPS Argos interpolated";	Positioning method at that time, described in reference table 10.2.

2.3.7 Data variables

Data variables contain the actual measurements and indicators about their quality, error, and mode through which they were obtained. The variable names are standardized in reference table 3; replace <PARAM> with any of the names indicated there. Mandatory attributes are marked in bold, however, EGO requests that all other attributes be used and contain meaningful information unless technical reasons make this impossible.

<PARAM> contains the raw values telemetered from the glider or obtained after it has been recovered or serviced (i.e. 'R' or 'P' data modes). **The values in <PARAM> should never be altered.**

<PARAM>_QC contains QC flags that pertain to the values in <PARAM>. Values in <PARAM>_QC are set initially by the automatic real-time tests. They are later modified in delayed mode at times where the QC flags are set incorrectly by the real-time procedures, and where erroneous data are not detected by the real-time procedures.

As EGO parameters can receive adjustments at different times, the variable PARAMETER_DATA_MODE is used to indicate the data mode of each <PARAM> (see reference table 19 for possible data modes of a parameter).

As detailed in reference table 3, <PARAM> can be classified into 3 groups:

a) Core parameters, C-EGO <PARAM>: these are parameters reported by the CTD sensor (i.e. PRES, TEMP, CNDC and PSAL);

b) BGC parameters, B-EGO <PARAM>: these are the ocean state BioGeoChemical (BGC) variables that will receive real-time qc tests, adjustment in real-time and delayed-mode adjustments;

c) Intermediate parameters, I-EGO <PARAM>: these are the intermediate biogeochemical variables. They will receive real-time qc tests and may receive adjustments.

C-EGO and B-EGO parameters could be adjusted. Consequently, each <PARAM> has 3 qc and adjusted variables that are used to record real-time qc test results and delayed-mode adjustment information:

- <PARAM>_ADJUSTED contains the adjusted values,
- <PARAM>_ADJUSTED_QC contains the QC flags set by the adjustment process,
- <PARAM>_ADJUSTED_ERROR contains the adjustment uncertainties

It's up to each DAC decision to consider that I-EGO parameters could be adjusted and thus to include or not these 3 additional variables for some or all I-EGO parameters.

When a parameter has PARAMETER_DATA_MODE = 'R' or 'P', no adjusted data are available. Hence, the adjusted section (<PARAM>_ADJUSTED, <PARAM>_ADJUSTED_QC and <PARAM>_ADJUSTED_ERROR) should be filled with FillValues.

When a parameter has PARAMETER_DATA_MODE = 'A', real time adjusted data are available in <PARAM>_ADJUSTED and their associated QC flags in

<PARAM>_ADJUSTED_QC. If not defined by the real time procedures, <PARAM>_ADJUSTED_ERROR could be let unset (filled with FillValues).

When a parameter has PARAMETER_DATA_MODE = 'D', <PARAM>_ADJUSTED, <PARAM>_ADJUSTED_QC, and <PARAM>_ADJUSTED_ERROR should be set.

Name	Definition	Comment
<PARAM>	float **<PARAM>(**TIME); <PARAM>:**standard_name** = "<X>"; <PARAM>:**units** = "<X>"; <PARAM>:**_FillValue** = <X>; <PARAM>:long_name = "<X>"; <PARAM>:valid_min = <X>; <PARAM>:valid_max = <X>; <PARAM>:comment = "<Y>"; <PARAM>:ancillary_variables = "XXX"; <PARAM>:cell_methods = "XXX"; <PARAM>:reference_scale = "XXX"; <PARAM>:coordinates = "TIME LATITUDE LONGITUDE PRES"; <PARAM>:sdn_parameter_urn = "XXX"; <PARAM>:sdn_uom_urn = "XXX"; <PARAM>:sdn_uom_name = "XXX"; <PARAM>:glider_original_parameter_name = "<Z>";	<PARAM> contains the original values of a parameter listed in reference table 3. Examples: PRES, TEMP, PSAL, DOXY. **These attributes are mandatory:** units and _FillValue. If a standard_name exists for this variable, **it is mandatory**. The other attributes are optional. <X> : standardized attributes listed in reference table 3 <Y>: Any optional comment. <Z>: Report the glider variable name for the concerned parameter. ancillary_variables. type char. Other variables associated with <PARAM>, e.g. <PARAM>_QC. List as space-separated string. Example: TEMP:ancillary_variables="TEMP_QC" cell_methods: type char. Specifies cell method as per CF convention. Example for instantaneous temperature measurements: TEMP:cell_methods="TIME:point". Values are listed in reference table 2.2. The boundary of the cells are described in the axis bound attribute (see CF convention for cell boundary, cell measure and cell method) . reference_scale: type char. For some measurements that are provided according to a standard reference scale specify the reference scale with this optional attribute. Example: ITS-90, PSS-78 sdn_parameter_urn: SeaDataNet parameter code sdn_uom_urn: SeaDataNet unit code
<PARAM>_QC	byte **<PARAM>_QC**(TIME); <PARAM>_QC:**long_name** = "Quality flag"; <PARAM>_QC:**conventions** = "EGO reference table 2.1"; <PARAM>_QC:**_FillValue** = -128; <PARAM>_QC:**valid_min** = 0; <PARAM>_QC:**valid_max**= 9; <PARAM>_QC:**flag_values** = 0, 1, 2, 3, 4, 5, 8, 9; <PARAM>_QC:**flag_meanings** = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value";	Quality flags for values of associated <PARAM>. The flag scale is specified in reference table 2.1, and is included in the flag_meanings attribute.

<PARAM>_ADJUSTED	float **<PARAM>_ADJUSTED(**TIME); <PARAM>_ADJUSTED:**standard_name** = "<X>"; <PARAM>_ADJUSTED:**units** = "<X>"; <PARAM>_ADJUSTED:**_FillValue** = <X>; <PARAM>_ADJUSTED:long_name = "<X>"; <PARAM>_ADJUSTED:valid_min = <X>; <PARAM>_ADJUSTED:valid_max = <X>; <PARAM>_ADJUSTED:comment = "<Y>"; <PARAM>_ADJUSTED:ancillary_variables = "XXX"; <PARAM>_ADJUSTED:cell_methods = "XXX"; <PARAM>_ADJUSTED:reference_scale = "XXX; <PARAM>_ADJUSTED:coordinates = "TIME LATITUDE LONGITUDE PRES"; <PARAM>_ADJUSTED:sdn_parameter_urn = "XXX"; <PARAM>_ADJUSTED:sdn_uom_urn = "XXX"; <PARAM>_ADJUSTED:sdn_uom_name = "XXX"; <PARAM>_ADJUSTED:glider_original_parameter_name = <Z>;	<PARAM>_ADJUSTED contains the adjusted values derived from the original values of a parameter. **<PARAM>_ADJUSTED is mandatory.** When no adjustment is performed, the FillValue is inserted. **These attributes are mandatory:** units and _FillValue. If a standard_name exists for this variable, **it is mandatory**. The other attributes are optional. <X> : standardized attributes listed in reference table 3 <Y>: Any optional comment. <Z>: Report the glider variable name for the concerned parameter. ancillary_variables. type char. Other variables associated with <PARAM>_ADJUSTED, e.g. <PARAM>_ADJUSTED_QC. List as space-separated string. Example: TEMP_ADJUSTED:ancillary_variables="TEMP_ADJUSTED_QC" cell_methods: type char. Specifies cell method as per CF convention. Example for instantaneous temperature measurements: TEMP_ADJUSTED:cell_methods="TIME: point". Values are listed in reference table 2.2. The boundary of the cells are described in the axis bound attribute (see CF convention for cell boundary, cell measure and cell method) . reference_scale: type char. For some measurements that are provided according to a standard reference scale specify the reference scale with this optional attribute. Example: ITS-90, PSS-78 sdn_parameter_urn: SeaDataNet parameter code sdn_uom_urn: SeaDataNet unit code
<PARAM>_ADJUSTED_QC	byte **<PARAM>_ADJUSTED_QC**(TIME); <PARAM>_ADJUSTED_QC:**long_name** = "Quality flag"; <PARAM>_ADJUSTED_QC:**conventions** = "EGO reference table 2.1"; <PARAM>_ADJUSTED_QC:**_FillValue** = -128; <PARAM>_ADJUSTED_QC:**valid_min** = 0; <PARAM>_ADJUSTED_QC:**valid_max**= 9; <PARAM>_ADJUSTED_QC:**flag_values** = 0, 1, 2, 3, 4, 5, 8, 9; <PARAM>_ADJUSTED_QC:**flag_meanings** = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value";	Quality flags for values of associated <PARAM>_ADJUSTED. The flag scale is specified in reference table 2.1, and is included in the flag_meanings attribute. **<PARAM>_ADJUSTED_QC is mandatory.** When no adjustment is performed, the FillValue is inserted.

<PARAM>_ADJUSTED_ERROR	float **<PARAM>_ADJUSTED_ERROR**(TIME); <PARAM>_ADJUSTED_ERROR:**long_name** = "Contains the error on the adjusted values as determined by the delayed mode QC process"; <PARAM>_ADJUSTED_ERROR**:_FillValue** = <X>; <PARAM>_ADJUSTED_ERROR:**units** = "<X>";	<PARAM>_ADJUSTED_ERROR Contains the error on the adjusted values as determined by the delayed mode QC process. **<PARAM>_ADJUSTED_ERROR is mandatory.** When no adjustment is performed, the FillValue is inserted. <X> : depends on the parameter.

Note on vertical axis

The PRES variable is usually a vertical axis. Its axis attribute is "Z" : PRES:axis="Z".

It has a "positive" mandatory attribute set to "down".

Example for sea temperature measurements and associated quality flags

```
        float TEMP(TIME);
                TEMP:standard_name = "sea_water_temperature" ;
                TEMP:units = "degree_Celsius";
                TEMP:_FillValue = 99999.f ;
                TEMP:long_name = "Sea temperature in-situ ITS-90 scale" ;
                TEMP:valid_min = -2.5f ;
                TEMP:valid_max = 40.f ;
                TEMP:comment = "" ;
                TEMP:ancillary_variable = "TEMP_QC";
                TEMP:sdn_parameter_urn = "SDN:P01::TEMPST01";
                TEMP:sdn_uom_urn = "SDN:P06::UPAA";
                TEMP:sdn_uom_name = "" ;
                TEMP:glider_original_parameter_name = "sci_water_temp" ;
                TEMP:cell_methods = "" ;
                TEMP:reference_scale = "" ;
                TEMP:coordinates = "TIME LATITUDE LONGITUDE PRES";
        byte TEMP_QC(TIME);
                TEMP_QC:long_name = "Quality flag" ;
                TEMP_QC:conventions = "EGO reference table 2.1" ;
                TEMP_QC:_FillValue = -128b ;
                TEMP_QC:valid_min = 0b ;
                TEMP_QC:valid_max = 9b ;
                TEMP_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b ;
                TEMP_QC:flag_meanings = "no_qc_performed good_data probably_good_data
bad_data_that_are_potentially_correctable  bad_data value_changed interpolated_value missing_value" ;
        float TEMP_ADJUSTED(TIME);
                TEMP_ADJUSTED:standard_name = "sea_water_temperature" ;
                TEMP_ADJUSTED:units = "degree_Celsius";
                TEMP_ADJUSTED:_FillValue = 99999.f ;
                TEMP_ADJUSTED:long_name = "Sea temperature in-situ ITS-90 scale" ;
                TEMP_ADJUSTED:valid_min = -2.5f ;
                TEMP_ADJUSTED:valid_max = 40.f ;
                TEMP_ADJUSTED:comment = "" ;
                TEMP_ADJUSTED:ancillary_variable = "TEMP_QC";
                TEMP_ADJUSTED:sdn_parameter_urn = "SDN:P01::TEMPST01";
                TEMP_ADJUSTED:sdn_uom_urn = "SDN:P06::UPAA";
                TEMP_ADJUSTED:sdn_uom_name = "" ;
                TEMP_ADJUSTED:glider_original_parameter_name = "sci_water_temp" ;
                TEMP_ADJUSTED:cell_methods = "" ;
                TEMP_ADJUSTED:reference_scale = "" ;
                TEMP_ADJUSTED:coordinates = "TIME LATITUDE LONGITUDE PRES";
        byte TEMP_ADJUSTED_QC(TIME);
                TEMP_ADJUSTED_QC:long_name = "Quality flag" ;
                TEMP_ADJUSTED_QC:conventions = "EGO reference table 2.1" ;
                TEMP_ADJUSTED_QC:_FillValue = -128b ;
                TEMP_ADJUSTED_QC:valid_min = 0b ;
                TEMP_ADJUSTED_QC:valid_max = 9b ;
                TEMP_ADJUSTED_QC:flag_values = 0b, 1b, 2b, 3b, 4b, 5b, 8b, 9b ;
                TEMP_ADJUSTED_QC:flag_meanings = "no_qc_performed good_data probably_good_data
bad_data_that_are_potentially_correctable  bad_data value_changed interpolated_value missing_value" ;
        float TEMP_ADJUSTED_ERROR(TIME);
```

TEMP_ADJUSTED_ERROR:long_name = "Contains the error on the adjusted values as determined by the delayed mode QC process" ;
TEMP_ADJUSTED_ERROR:_FillValue = 99999.f ;
TEMP_ADJUSTED_ERROR:units = "degree_Celsius" ;

2.3.8 Sub-surface current estimate variables

The following variables are used to store sub-surface current estimates.

They are present in the EGO file only if such data have to be stored.

Name	Definition	Comment
WATERCURRENTS_TIME	double WATERCURRENTS_TIME(TIME_CURRENT); WATERCURRENTS_TIME:long_name = "Epoch time of the sub-surface current estimate"; WATERCURRENTS_TIME:standard_name = "time"; WATERCURRENTS_TIME:units = "seconds since 1970-01-01T00:00:00Z"; WATERCURRENTS_TIME:_FillValue = 9999999999; WATERCURRENTS_TIME:valid_min = 0; WATERCURRENTS_TIME:valid_max = 90000; WATERCURRENTS_TIME:comment = <Y>; WATERCURRENTS_TIME:axis = "T"; WATERCURRENTS_TIME:sdn_parameter_urn = "SDN:P01::ELTMEP01"; WATERCURRENTS_TIME:sdn_uom_urn = "SDN:P06::UTBB";	Time of the current estimate in seconds since midnight, 1970-01-01. Example: July 25, 2001, 19:14:00 is stored as 996088440. <Y>: Any optional comment.
WATERCURRENTS_LATITUDE	double WATERCURRENTS_LATITUDE(TIME_CURRENT); WATERCURRENTS_LATITUDE:long_name = "Latitude of the sub-surface current estimate"; WATERCURRENTS_LATITUDE:standard_name = "latitude"; WATERCURRENTS_LATITUDE:units = "degree_north"; WATERCURRENTS_LATITUDE:_FillValue = 99999; WATERCURRENTS_LATITUDE:valid_min = -90; WATERCURRENTS_LATITUDE:valid_max = 90; WATERCURRENTS_LATITUDE:axis = "Y";	Latitude of the current estimate. Units: degrees north; southern latitudes are negative. Example: 44.4991 for 44° 29′ 56.76″ N
WATERCURRENTS_LONGITUDE	double WATERCURRENTS_LONGITUDE(TIME_CURRENT); WATERCURRENTS_LONGITUDE:long_name = "Longitude of the sub-surface current estimate"; WATERCURRENTS_LONGITUDE:standard_name = "longitude"; WATERCURRENTS_LONGITUDE:units = "degree_east"; WATERCURRENTS_LONGITUDE:_FillValue = 99999; WATERCURRENTS_LONGITUDE:valid_min = -180; WATERCURRENTS_LONGITUDE:valid_max = 180; WATERCURRENTS_LONGITUDE:axis = "X";	Longitude of the current estimate. Unit: degrees east; western latitudes are negative. Example: 16.7222 for 16° 43′ 19.92″ E
WATERCURRENTS_DEPTH	float WATERCURRENTS_DEPTH(TIME_CURRENT); WATERCURRENTS_DEPTH:long_name = "Depth of the sub-surface current estimate"; WATERCURRENTS_DEPTH:standard_name = "depth"; WATERCURRENTS_DEPTH:units = "m"; WATERCURRENTS_DEPTH:_FillValue = 99999;	Depth of the current estimate.
WATERCURRENTS_U	float WATERCURRENTS_U(TIME_CURRENT); WATERCURRENTS_U:long_name = "Eastward component of the sub-surface current estimate"; WATERCURRENTS_U:standard_name = "eastward_sea_water_velocity"; WATERCURRENTS_U:units = "cm/s"; WATERCURRENTS_U:_FillValue = 99999;	Eastward component of the current estimate.
WATERCURRENTS_V	float WATERCURRENTS_V(TIME_CURRENT); WATERCURRENTS_V:long_name = "Northward component of the sub-surface current estimate"; WATERCURRENTS_V:standard_name = "northward_sea_water_velocity"; WATERCURRENTS_V:units = "cm/s"; WATERCURRENTS_V:_FillValue = 99999;	Northward component of the current estimate.

2.3.9 History information

These dimensions are specific to history section.

Name	Definition	Comment
N_HISTORY	N_HISTORY =<int value>;	Number of history records.

This section contains history information for each action performed on the time-series by a data center.

Each item of this section has a N_HISTORY (number of history records) dimension.

A history record is created whenever an action is performed on a part of the time-series definied by history_start_time and history_stop_time.

The recorded actions are coded and described in the history code table from the reference table 7.

On the GDAC, multi-profile history section is empty to reduce the size of the file. History section is available on mono-profile files, or in multi-profile files distributed from the web data selection.

Name	Definition	Comment
HISTORY_INSTITUTION	char HISTORY_INSTITUTION (N_HISTORY, STRING2); HISTORY_INSTITUTION:long_name = "Institution which performed action"; HISTORY_INSTITUTION:conventions = "EGO reference table 4"; HISTORY_INSTITUTION:_FillValue = " ";	Institution that performed the action. Institution codes are described in reference table 4. Example : BO for BODC
HISTORY_STEP	char HISTORY_STEP (N_HISTORY, STRING4); HISTORY_STEP:long_name = "Step in data processing"; HISTORY_STEP:conventions = "EGO reference table 12"; HISTORY_STEP:_FillValue = " ";	Code of the step in data processing for this history record. The step codes are described in reference table 12. Example : ARGQ : Automatic QC of data reported in real-time has been performed
HISTORY_SOFTWARE	char HISTORY_SOFTWARE (N_HISTORY, STRING8); HISTORY_SOFTWARE:long_name = "Name of software which performed action"; HISTORY_SOFTWARE:conventions = "Institution dependent"; HISTORY_SOFTWARE:_FillValue = " ";	Name of the software that performed the action. This code is institution dependent. Example : WJO
HISTORY_SOFTWARE_RELEASE	char HISTORY_SOFTWARE_RELEASE (N_HISTORY, STRING4); HISTORY_SOFTWARE_RELEASE:long_name = "Version/release of software which performed action"; HISTORY_SOFTWARE_RELEASE:conventions = "Institution dependent"; HISTORY_SOFTWARE_RELEASE:_FillValue = " ";	Version of the software. This name is institution dependent. Example : «1.0»
HISTORY_REFERENCE	char HISTORY_REFERENCE (N_HISTORY, STRING64); HISTORY_REFERENCE:long_name = "Reference of database"; HISTORY_REFERENCE:conventions = "Institution dependent"; HISTORY_REFERENCE:_FillValue = " ";	Code of the reference database used for quality control in conjunction with the software. This code is institution dependent. Example : WOD2001

HISTORY_DATE	char HISTORY_DATE(N_HISTORY, DATE_TIME); HISTORY_DATE:long_name = "Date the history record was created"; HISTORY_DATE:conventions = "YYYYMMDDHHMISS"; HISTORY_DATE:_FillValue = " ";	Date of the action. Example : 20011217160057
HISTORY_ACTION	char HISTORY_ACTION (N_HISTORY, STRING64); HISTORY_ACTION:long_name = "Action performed on data"; HISTORY_ACTION:conventions = "EGO reference table 7"; HISTORY_ACTION:_FillValue = " ";	Name of the action. The action codes are described in reference table 7. Example : QCF$ for QC failed
HISTORY_PARAMETER	char HISTORY_PARAMETER(N_HISTORY, STRING16); HISTORY_PARAMETER:long_name = "Parameter action is performed on"; HISTORY_PARAMETER:conventions = "EGO reference table 3"; HISTORY_PARAMETER:_FillValue = " ";	Name of the parameter on which the action is performed. The parameter names are listed in reference table 3. Example : PSAL
HISTORY_PREVIOUS_VALUE	float HISTORY_PREVIOUS_VALUE(N_HISTORY); HISTORY_PREVIOUS_VALUE:long_name = "Parameter or flag previous value before action"; HISTORY_PREVIOUS_VALUE:_FillValue = 99999;	Parameter or flag of the previous value before action. Example : 2 (probably good) for a flag that was changed to 1 (good)
HISTORY_START_TIME_INDEX	int HISTORY_START_TIME_INDEX (N_HISTORY); HISTORY_START_TIME_INDEX:long_name = "Start time index action applied on"; HISTORY_START_TIME_INDEX:_FillValue = 99999;	Start time index the action is applied to. Example : 100
HISTORY_STOP_TIME_INDEX	int HISTORY_STOP_TIME_INDEX (N_HISTORY); HISTORY_STOP_TIME_INDEX:long_name = "Stop time index action applied on"; HISTORY_STOP_TIME_INDEX:_FillValue = 99999;	Stop time index the action is applied to. Example : 150
HISTORY_QCTEST	char HISTORY_QCTEST(N_HISTORY, STRING16); HISTORY_QCTEST:long_name = "Documentation of tests performed, tests failed (in hex form)"; HISTORY_QCTEST:conventions = "Write tests performed when ACTION=QCP$; tests failed when ACTION=QCF$"; HISTORY_QCTEST:_FillValue = " ";	This field records the tests performed when ACTION is set to QCP$ (qc performed), the test failed when ACTION is set to QCF$ (qc failed). The QCTEST codes are describe in reference table 11. Example : 0A (in hexadecimal form)

The usage of history section is described in §5 “Using the History section of the EGO NetCDF Structure”.

2.4 Gliders metadata

2.4.1 Dimensions and definitions

These dimensions are specific to metadata items.

Name	Definition	Comment
N_SENSOR	N_SENSOR=<int value>;	Number of sensors mounted on the glider and used to measure the parameters. Examples: (CTD_PRES, CTD_TEMP, CTD_CNDC): N_SENSOR = 3 (CTD_PRES, CTD_TEMP, CTD_CNDC, BACKSCATTERINGMETER_BBP700): N_PARAM = 4
N_PARAM	N_PARAM=<int value>;	Number of parameters measured or calculated during the glider deployment. Examples: (PRES, TEMP, PSAL): N_PARAM = 3 (PRES, TEMP, PSAL, BETA_BACKSCATTERING700, BBP700): N_PARAM = 5
N_DERIVATION	N_DERIVATION=<int value>;	Maximum number of derivation and calibrations for a parameter.
N_POSITIONING_SYSTEM	N_POSITIONING_SYSTEM=<int value>;	Number of positioning systems.
N_TRANS_SYSTEM	N_TRANS_SYSTEM=<int value>;	Number of transmission systems.

2.4.2 Glider characteristics

This section contains the main characteristics of the glider.

Name	Definition	Comment
PLATFORM_FAMILY	char PLATFORM_FAMILY(STRING256); PLATFORM_FAMILY:long_name = "Category of instrument"; PLATFORM_FAMILY:conventions = "EGO reference table 22"; PLATFORM_FAMILY:_FillValue = " ";	Category of instrument. See reference table 22. Example: COASTAL_GLIDER
PLATFORM_TYPE	char PLATFORM_TYPE(STRING32); PLATFORM_TYPE:long_name = "Type of glider"; PLATFORM_TYPE:conventions = "EGO reference table 23"; PLATFORM_TYPE:_FillValue = " ";	Type of glider. See reference table 23. Example: SEAGLIDER
PLATFORM_MAKER	char PLATFORM_MAKER (STRING256); PLATFORM_MAKER:long_name = "Name of the manufacturer"; PLATFORM_MAKER:convention = "EGO reference table 24"; PLATFORM_MAKER:_FillValue = " ";	Name of the manufacturer. See reference table 24. Example : WRC
GLIDER_SERIAL_NO	char GLIDER_SERIAL_NO(STRING16); GLIDER_SERIAL_NO:long_name = "Serial number of the glider"; GLIDER_SERIAL_NO:_FillValue = " ";	This field should contain only the serial number of the glider. Example 1679
GLIDER_OWNER	char GLIDER_OWNER(STRING64); GLIDER_OWNER:long_name = "Glider owner"; GLIDER_OWNER:_FillValue = " ";	The owner of the glider (may be different from the data center and operating institution). Example: "SCRIPPS"
OPERATING_INSTITUTION	char OPERATING_INSTITUTION (STRING64); OPERATING_INSTITUTION:long_name = "Operating institution of the glider"; OPERATING_INSTITUTION:_FillValue = " ";	The operating institution of the glider (may be different from the glider owner and data center). Example: "INSU", "MARS", "NACO".
WMO_INST_TYPE	char WMO_INST_TYPE(STRING4); WMO_INST_TYPE:long_name = "Coded instrument type"; WMO_INST_TYPE:conventions = "EGO reference table 8"; WMO_INST_TYPE:_FillValue = " ";	Instrument type from WMO code table 1770. A subset of WMO table 1770 is documented in the reference table 8.

POSITIONING_SYSTEM	char POSITIONING_SYSTEM(N_POSITIONING_SYSTEM, STRING8); POSITIONING_SYSTEM:long_name = "Positioning system"; POSITIONING_SYSTEM:convention = "EGO reference table 9.1"; POSITIONING_SYSTEM:_FillValue = " ";	Name of the positioning systems from reference table 9.1. Example : GPS
TRANS_SYSTEM	char TRANS_SYSTEM(N_TRANS_SYSTEM, STRING16); TRANS_SYSTEM:long_name = "Telecommunication system used"; TRANS_SYSTEM:convention = "EGO reference table 10.1"; TRANS_SYSTEM:_FillValue = " ";	Name of the telecommunication systems from reference table 10.1. Example: IRIDIUM.
TRANS_SYSTEM_ID	char TRANS_SYSTEM_ID(N_TRANS_SYSTEM, STRING32); TRANS_SYSTEM_ID:long_name = "Program identifier used by the transmission system"; TRANS_SYSTEM_ID:_FillValue = " ";	Program identifier of the telecommunication subscription. Use N/A when not applicable (eg : Iridium or Orbcomm) Example: 38511 is a program number for all the beacons of an ARGOS customer.
TRANS_FREQUENCY	char TRANS_FREQUENCY(N_TRANS_SYSTEM, STRING16); TRANS_FREQUENCY:long_name = "Frequency of transmission from the glider"; TRANS_FREQUENCY:units = "hertz"; TRANS_FREQUENCY:_FillValue = " ";	Frequency of transmission from the glider. Unit : hertz Example : 1/44
BATTERY_TYPE	char BATTERY_TYPE(STRING64); BATTERY_TYPE: long_name = "Type of battery packs in the glider"; BATTERY_TYPE:_FillValue = " ";	Describes the type of battery packs in the glider. Example: Alkaline, Lithium or Alkaline and Lithium
BATTERY_PACKS	char BATTERY_PACKS(STRING64); BATTERY_PACKS: long_name = "Configuration of battery packs in the glider"; BATTERY_PACKS:_FillValue = " ";	Describes the configuration of battery packs in the glider, number and type. Example: 4DD Li + 1C Alk
SPECIAL_FEATURES	char SPECIAL_FEATURES(STRING1024); SPECIAL_FEATURES:long_name = "Extra features of the glider (algorithms, compressee, pump change etc.)"; SPECIAL_FEATURES:_FillValue = " ";	Additional glider features can be specified here such as algorithms used by the glider (Ice Sensing Algorithm, Interim Storage Algorithm, grounding avoidance) or additional hardware such as a compressee (buoyancy compensator) or changing pump. Example : "Ice Sensing Algorithm"
FIRMWARE_VERSION_NAVIGATION	char FIRMWARE_VERSION_NAVIGATION(STRING16); FIRMWARE_VERSION_NAVIGATION:long_name = "Firmware version of the navigation controller board"; FIRMWARE_VERSION_NAVIGATION:_FillValue = " ";	The firmware version of the navigation controller board.
FIRMWARE_VERSION_SCIENCE	char FIRMWARE_VERSION_SCIENCE(STRING16); FIRMWARE_VERSION_SCIENCE:long_name = "Firmware version of the scientific sensors controller board"; FIRMWARE_VERSION_SCIENCE:_FillValue = " ";	The firmware version of the scientific sensors controller board.
GLIDER_MANUAL_VERSION	char GLIDER_MANUAL_VERSION(STRING16); GLIDER_MANUAL_VERSION:long_name = "Manual version of the glider"; GLIDER_MANUAL_VERSION:_FillValue = " ";	The version date or number for the manual for each glider. Example 110610 or 004
ANOMALY	char ANOMALY(STRING256); ANOMALY:long_name = "Describe any anomalies or problems the glider may have had"; ANOMALY:_FillValue = " ";	This field describes any anomaly or problem the glider may have had. Example: "the immersion drift is not stable."
CUSTOMIZATION	char CUZTOMIZATION (STRING1024); CUSTOMIZATION:long_name = "Glider customization, i.e. (institution and modifications)"; CUSTOMIZATION:_FillValue = " ";	Free form field to record changes made to the glider after manufacture and before deployment, i.e. this could be the customization institution plus a list of modifications.

DAC_FORMAT_ID	char DAC_FORMAT_ID(STRING16); DAC_FORMAT_ID:long_name = "Format number used by the DAC to describe the data format type for each glider"; DAC_FORMAT_ID:_FillValue = " ";	Format numbers used by individual DACs to describe each glider type and versison.

2.4.3 Glider deployment information

Name	Definition	Comment
DEPLOYMENT_START_DATE	charDEPLOYMENT_START_DATE(DATE_TIME); DEPLOYMENT_START_DATE:long_name = "Date (UTC) of the deployment"; DEPLOYMENT_START_DATE:conventions = "YYYYMMDDHHMISS"; DEPLOYMENT_START_DATE:_FillValue = " ";	Date and time (UTC) of deployment of the glider. Format : YYYYMMDDHHMISS Example: 20011230090500 : December 30th 2001 09:05:00
DEPLOYMENT_START_LATITUDE	double DEPLOYMENT_START_LATITUDE; DEPLOYMENT_START_LATITUDE:long_name = "Latitude of the glider when deployed"; DEPLOYMENT_START_LATITUDE:units = "degree_north"; DEPLOYMENT_START_LATITUDE:_FillValue = 99999; DEPLOYMENT_START_LATITUDE:valid_min = -90; DEPLOYMENT_START_LATITUDE:valid_max = 90;	Latitude of the deployment. Unit : degree north. Example : 44.4991 : 44° 29' 56.76" N
DEPLOYMENT_START_LONGITUDE	double DEPLOYMENT_START_LONGITUDE; DEPLOYMENT_START_LONGITUDE:long_name = "Longitude of the glider when deployed"; DEPLOYMENT_START_LONGITUDE:units = "degree_east"; DEPLOYMENT_START_LONGITUDE:_FillValue = 99999; DEPLOYMENT_START_LONGITUDE:valid_min = -180; DEPLOYMENT_START_LONGITUDE:valid_max = 180;	Longitude of the deployment. Unit : degree east Example: 16.7222 : 16° 43' 19.92" E
DEPLOYMENT_START_QC	byte DEPLOYMENT_START_QC; DEPLOYMENT_START_QC:long_name = "Quality on DEPLOYMENT_START date, time and location"; DEPLOYMENT_START_QC:conventions = "EGO reference table 2.1"; DEPLOYMENT_START_QC:_FillValue = -128; DEPLOYMENT_START _QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; DEPLOYMENT_START _QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value";	Quality flag on deployment start date, time and location. The flag scale is described in the reference table 2.1. Example: 1 : deployment start date, time and location seems correct.
DEPLOYMENT_PLATFORM	char DEPLOY_PLATFORM(STRING32); DEPLOY_PLATFORM:long_name = "Identifier of the deployment platform"; DEPLOY_PLATFORM:_FillValue = " ";	Identifier of the deployment platform. Example: L'ATALANTE
DEPLOYMENT_CRUISE_ID	char DEPLOYMENT_CRUISE_ID(STRING32); DEPLOYMENT_CRUISE_ID:long_name = "Identifier of the cruise that deployed the glider"; DEPLOYMENT_CRUISE_ID:_FillValue = " ";	Identifier of the cruise used to deploy the platform. Example : POMME2
DEPLOYMENT_REFERENCE_STATION_ID	char DEPLOYMENT_REFERENCE_STATION_ID(STRING256); DEPLOYMENT_REFERENCE_STATION_ID:long_name = "Identifier of stations used to verify the parameter measurements"; DEPLOYMENT_REFERENCE_STATION_ID:_FillValue = " ";	Identifier of CTD or XBT stations used to verify the first profile. Example : 58776, 58777
DEPLOYMENT_END_DATE	char DEPLOYMENT_END_DATE(DATE_TIME); DEPLOYMENT_END_DATE:long_name = "Date (UTC) of the glider recovery"; DEPLOYMENT_END_DATE:conventions = "YYYYMMDDHHMISS"; END_MISSION_DATE:_FillValue = " ";	Date (UTC) of the end of deployment of the glider. Format : YYYYMMDDHHMISS Example : 20011230090500 : December 30th 2001 09:05:00
DEPLOYMENT_END_LATITUDE	double DEPLOYMENT_END_LATITUDE; DEPLOYMENT_END_LATITUDE:long_name = "Latitude of the glider recovery"; DEPLOYMENT_END_LATITUDE:units = "degree_north"; DEPLOYMENT_END_LATITUDE:_FillValue = 99999; DEPLOYMENT_END_LATITUDE:valid_min = -90; DEPLOYMENT_END_LATITUDE:valid_max = 90;	Latitude of the recovery of the glider. Unit : degree north. Example : 44.4991 : 44° 29' 56.76" N
DEPLOYMENT_END_LONGIT	double DEPLOYMENT_END_LONGITUDE;	Longitude of the recovery of the

UDE	DEPLOYMENT_END_LONGITUDE:long_name = "Longitude of the glider recovery"; DEPLOYMENT_END_LONGITUDE:units = "degree_east"; DEPLOYMENT_END_LONGITUDE:_FillValue = 99999; DEPLOYMENT_END_LONGITUDE:valid_min = -180; DEPLOYMENT_END_LONGITUDE:valid_max = 180;	glider. Unit : degree east Example : 16.7222 : 16° 43' 19.92" E
DEPLOYMENT_END_QC	byte DEPLOYMENT_END_QC; DEPLOYMENT_END_QC:long_name = "Quality on DEPLOYMENT_END date, time and location"; DEPLOYMENT_END_QC:conventions = "EGO reference table 2.1"; DEPLOYMENT_END_QC:_FillValue = -128; DEPLOYMENT_ END_QC:flag_values = 0, 1, 2, 3, 4, 5, 8, 9; DEPLOYMENT_ END_QC:flag_meanings = "no_qc_performed good_data probably_good_data bad_data_that_are_potentially_correctable bad_data value_changed interpolated_value missing_value";	Quality flag on deployment end date, time and location. The flag scale is described in the reference table 2.1. Example : 1 : deployment end date, time and location seems correct.
DEPLOYMENT_END_STATUS	char DEPLOYMENT_END_STATUS; DEPLOYMENT_END_STATUS:long_name = "Status of the end of mission of the glider"; DEPLOYMENT_END_STATUS:conventions = "R: retrieved, L: lost"; DEPLOYMENT_END_STATUS:_FillValue = " ";	Status of the end of mission of the glider. R: retrieved L: lost
DEPLOYMENT_OPERATOR	char DEPLOYMENT_OPERATOR(STRING256); DEPLOYMENT_OPERATOR:long_name = "Name of the person in charge of the glider deployment"; DEPLOYMENT_OPERATOR:_FillValue = " ";	Name of the person in charge of the glider deployment

2.4.4 Glider sensor and parameter information

A **sensor** is a device used to measure a physical parameter. Sensor outputs are provided in parameter counts and need to be converted in parameter physical units using a calibration equation. This conversion can be done onboard the glider or during the decoding process.

A **parameter** is a measurement of a physical phenomenon; it can be provided by a sensor (in sensor counts or in physical units) or computed (derived) from other parameters.

A sensor can measure 1 to N parameter(s). A parameter can be measured by 1 or N sensor(s).

2.4.4.1 Glider sensor information

This section contains information about the sensors of the glider.

A list of standardised sensor names is given in reference table 25.

Name	Definition	Comment
SENSOR	char **SENSOR**(N_SENSOR, STRING32); SENSOR:long_name = "Name of the sensor mounted on the glider"; SENSOR:conventions = "EGO reference table 25"; SENSOR:_FillValue = " ";	Names of the sensors mounted on the glider Example: CTD_PRES, CTD_TEMP, CTD_CNDC, OXYGEN_OPTODE. See EGO reference table 25.
SENSOR_MAKER	char **SENSOR_MAKER**(N_SENSOR, STRING256); SENSOR_MAKER:long_name = "Name of the sensor manufacturer"; SENSOR_MAKER:conventions = "EGO reference table 26"; SENSOR_MAKER:_FillValue = " ";	Name of the manufacturer of the sensor. Example : DRUCK, SBE, AANDERAA. See EGO reference table 26.
SENSOR_MODEL	char **SENSOR_MODEL**(N_SENSOR, STRING256); SENSOR_MODEL:long_name = "Type of the sensor"; SENSOR_MODEL:conventions = "EGO reference table 27"; SENSOR_MODEL:_FillValue = " ";	Model of sensor. Example: DRUCK, SBE41CP, AANDERAA_OPTODE_3930. See EGO reference table 27.
SENSOR_SERIAL_NO	char **SENSOR_SERIAL_NO**(N_SENSOR, STRING16); SENSOR_SERIAL_NO:long_name = "Serial number of the sensor"; SENSOR_SERIAL_NO:_FillValue = " ";	Serial number of the sensor. Example : 2646 036 073
SENSOR_MOUNT	char SENSOR_MOUNT(N_SENSOR, STRING64); SENSOR_MOUNT:long_name = "Sensor mounting characteristics"; SENSOR_MOUNT:conventions = "EGO reference table 20"; SENSOR_MOUNT:_FillValue = " ";	Sensor mounting characteristics. See EGO reference table 20.
SENSOR_ORIENTATION	char SENSOR_ORIENTATION(N_SENSOR, STRING16); SENSOR_ORIENTATION:long_name = "Sensor orientation characteristics"; SENSOR_ORIENTATION:conventions = "EGO reference table 21"; SENSOR_ORIENTATION:_FillValue = " ";	Sensor orientation characteristics. See EGO reference table 21.

2.4.4.2 Glider parameter information

This section contains information about the parameters measured by the glider or derived from glider measurements.

A list of standardised parameter names is given in reference table 3.

Name	Definition	Comment
PARAMETER	char **PARAMETER**(N_PARAM, STRING64); PARAMETER:long_name = "Name of parameter computed from glider measurements"; PARAMETER:conventions = "EGO reference table 3"; PARAMETER:_FillValue = " ";	Names of the parameters measured by glider sensors or derived from glider measurements. The parameter names are listed in reference table 3. Examples : TEMP, PSAL, CNDC TEMP : temperature in Celsius PSAL : practical salinity in psu CNDC : conductvity in mhos/m
PARAMETER_SENSOR	char **PARAMETER_SENSOR**(N_PARAM, STRING128); PARAMETER_SENSOR:long_name = "Name of the sensor that measures this parameter"; PARAMETER_SENSOR:conventions = "EGO reference table 25"; PARAMETER_SENSOR:_FillValue = " ";	Names of the sensors that measured the glider parameters. See EGO reference table 25. Example: CTD_PRES, CTD_TEMP, CTD_CNDC, OPTODE_DOXY.
PARAMETER_DATA_MODE	char **PARAMETER_DATA_MODE**(N_PARAM); PARAMETER_DATA_MODE:long_name = "Data mode of the parameter"; PARAMETER_DATA_MODE:conventions = "EGO reference table 19"; PARAMETER_DATA_MODE:_FillValue =" ";	Describe the data mode of the individual parameter. The possible values are listed in reference table 19.
PARAMETER_UNITS	char PARAMETER_UNITS(N_PARAM, STRING32); PARAMETER_UNITS:long_name = "Units of accuracy and resolution of the parameter"; PARAMETER_UNITS:_FillValue = " ";	Units of accuracy and resolution of the parameter. Example : psu
PARAMETER_ACCURACY	char PARAMETER_ACCURACY(N_PARAM, STRING32); PARAMETER_ACCURACY:long_name = "Accuracy of the parameter"; PARAMETER_ACCURACY:_FillValue = " ";	Accuracy of the parameter. Example: "8 micromole/l or 5%"
PARAMETER_RESOLUTION	char PARAMETER_RESOLUTION(N_PARAM, STRING32); PARAMETER_RESOLUTION:long_name = "Resolution of the parameter"; PARAMETER_RESOLUTION:_FillValue =" ";	Resolution of the parameter returned by the sensor (note that this is not necessarily equivalent to the resolution of the parameter returned by the float through telemetry). Example : 0.001 micromole/l

2.4.5 Glider parameters derivation and calibration information

This section contains information about the parameter derivation and the parameter calibration.

A derived parameter is calculated from one or several parameters.Example: salinity is derived from conductivity, temperature and pressure.

Calibrations are applied to parameters to create adjusted parameters. Different calibration methods will be used by groups processing glider data. When a method is applied, its description is stored in the following fields.

This section contains calibration information for each parameter of the EGO file.

Each item of this section has a N_DERIVATION (number of derivations and calibrations), N_PARAM (number of parameters) dimension.

If no derivation or calibration is available, N_DERIVATION is set to 1, all values of the derivation section are set to fill values.

Name	Definition	Comment
DERIVATION_PARAMETER	char DERIVATION_PARAMETER(N_DERIVATION, N_PARAM, STRING64); DERIVATION_PARAMETER:long_name = "Name of parameter with derivation or calibration information"; DERIVATION_PARAMETER:conventions = "EGO reference table 3"; DERIVATION_PARAMETER:_FillValue = " ";	Name of the derived or calibrated parameter. The list of parameters is in reference table 3. Example : PSAL
DERIVATION_EQUATION	char DERIVATION_EQUATION(N_DERIVATION, N_PARAM, STRING4096); DERIVATION_EQUATION:long_name = "Derivation or calibration equation for this parameter"; DERIVATION_EQUATION:_FillValue = " ";	Derivation or calibration equation applied to the parameter. Example : Tc = a1 * T + a0
DERIVATION_COEFFICIENT	char DERIVATION_COEFFICIENT(N_DERIVATION, N_PARAM, STRING4096); DERIVATION_COEFFICIENT:long_name = "Derivation or calibration coefficients for this equation"; DERIVATION_COEFFICIENT:_FillValue = " ";	Derivation or calibration coefficients for this equation. Example : a1=0.99997 , a0=0.0021
DERIVATION_COMMENT	char DERIVATION_COMMENT(N_DERIVATION, N_PARAM, STRING4096); DERIVATION_COMMENT:long_name = "Comment applying to this parameter derivation or calibration"; DERIVATION_COMMENT:_FillValue = " ";	Comment about this derivation or calibration Example : The sensor is not stable
DERIVATION_DATE	char DERIVATION_DATE(N_DERIVATION, N_PARAM, DATE_TIME); DERIVATION_DATE:long_name = "Date (UTC) of derivation or calibration"; DERIVATION_DATE:conventions = "YYYYMMDDHHMISS"; DERIVATION_DATE:_FillValue = " ";	Date of the derivation or calibration. Format : YYYYMMDDHHMISS Example : 20011230090500 : December 30th 2001 09:05:00

2.5 Gliders technical data

The glider technical data are managed as variables.

TBC

3 Gliders NetCDF profile data format version 2.0

Vertical descending and ascending profiles are generated from EGO timeseries.

This chapter describes the format of the NetCDF files used to store EGO profile data. This format is the same as the Argo B profile file version 3.1.

One EGO profile file contains the data sampled during one descent or ascent of the glider.

For EGO profile file naming conventions, see §6.2.

Coriolis processing chain specificity:

This format is able to store more than one profile per descent or ascent (N_PROF dimension), typically one profile for each sensor mounted on the Glider. However, as in the Coriolis processing chain the (Slocum and SeaGlider) BGC measurements are reported to the levels of the CTD ones, the generated profiles have the N_PROF = 1 dimension.

3.1 Data file dimensions

Name	Value	Definition
DATE_TIME	DATE_TIME = 14	This dimension is the length of an ASCII date and time value. Date_time convention is : YYYYMMDDHHMISS YYYY : year MM : month DD : day HH : hour of the day (as 0 to 23) MI : minutes (as 0 to 59) SS : seconds (as 0 to 59) Date and time values are always in universal time coordinates (UTC). Examples : 20010105172834 : January 5th 2001 17:28:34 19971217000000 : December 17th 1997 00:00:00
STRING256 STRING64 STRING32 STRING16 STRING8 STRING4 STRING2	STRING256 = 256 STRING64 = 64 STRING32 = 32 STRING16 = 16 STRING8 = 8 STRING4 = 4 STRING2 = 2	String dimensions from 2 to 256.
N_PROF	N_PROF = <int value>	Number of profiles contained in the file. This dimension depends on the data set. A file contains at least one profile. There is no defined limit on the maximum number of profiles in a file. Example : N_PROF = 100

N_PARAM	N_PARAM = <int value>	Maximum number of parameters measured or calculated for a pressure sample. This dimension depends on the data set. Examples : (pressure, temperature) : N_PARAM = 2 (pressure, temperature, salinity) : N_PARAM = 3 (pressure, temperature, conductivity, salinity) : N_PARAM = 4
N_LEVELS	N_LEVELS = <int value>	Maximum number of pressure levels contained in a profile. This dimension depends on the data set. Example : N_LEVELS = 100
N_VALUES<i>	N_VALUES<i> = <int value>	Maximum number of parameter measurements sampled at a given pressure level (used only for sensors that provide multi-dimensional variables). Example: N_VALUES41 = 41
N_CALIB	N_CALIB = <int value>	Maximum number of calibrations performed on a profile. This dimension depends on the data set. Example : N_CALIB = 10
N_HISTORY	N_HISTORY = UNLIMITED	Number of history records.

3.2 Global attributes

The global attributes section is used for data discovery. The following global attributes should appear in the global section:

// global attributes:
:title = "Glider vertical profile";
:institution = "CORIOLIS";
:source = "Glider";
:history = " 2023-01-18T10:45:38Z creation; 2023-01-18T10:45:41Z last update (coriolis COQC software) ";
:references = "Generated by Coriolis EGO Glider decoder (version '011n')";
:comment = "free text";
:user_manual_version = "1.11";
:Conventions = "EGO-2.0 CF-1.6";
:featureType = "trajectoryProfile";

Global attribute name	Definition
title	A succinct description of what is in the dataset.
institution	Specifies where the original data was produced.
source	The method of production of the original data. If it was model-generated, source should name the model and its version, as specifically as could be useful. If it is observational, source should characterize it (e.g., "surface observation" or "radiosonde").
history	Provides an audit trail for modifications to the original data. Well-behaved generic NetCDF filters will automatically append their name and the parameters with which they were invoked to the global history attribute of an input NetCDF file. We recommend that each line begin with a timestamp indicating the date and time of day that the program was executed.
references	Published or web-based references that describe the data or methods used to produce it.
comment	Miscellaneous information about the data or methods used to produce it.
user_manual_version	The version number of the user manual
Conventions	The conventions supported by this file, blank separated
featureType	The NetCDF CF feature type.
decoder_version	Optional comment on decoder version

3.3 General information on the profile

This section contains information about the whole file.

Name	Definition	Comment
DATA_TYPE	char DATA_TYPE(STRING32); DATA_TYPE:long_name = "Data type"; DATA_TYPE:conventions = "EGO reference table 1"; DATA_TYPE:_FillValue = " ";	This field contains the type of data contained in the file. The list of acceptable data types is in the reference table 1. Example : EGO profile
FORMAT_VERSION	char FORMAT_VERSION(STRING4); FORMAT_VERSION:long_name = "File format version"; FORMAT_VERSION:_FillValue = " ";	File format version Example : "2.0"
HANDBOOK_VERSION	char HANDBOOK_VERSION(STRING4); HANDBOOK_VERSION:long_name = "Data handbook version"; HANDBOOK_VERSION:_FillValue = " ";	Version number of the data handbook. This field indicates that the data contained in this file are managed according to the policy described in the EGO data management handbook. Example : "1.0"
REFERENCE_DATE_TIME	char REFERENCE_DATE_TIME(DATE_TIME); REFERENCE_DATE_TIME:long_name = "Date of reference for Julian days"; REFERENCE_DATE_TIME:conventions = "YYYYMMDDHHMISS"; REFERENCE_DATE_TIME:_FillValue = " ";	Date of reference for Julian days. The recommended reference date time is "19500101000000" : January 1st 1950 00:00:00
DATE_CREATION	char DATE_CREATION(DATE_TIME); DATE_CREATION:long_name = "Date of file creation"; DATE_CREATION:conventions = "YYYYMMDDHHMISS"; DATE_CREATION:_FillValue = " ";	Date and time (UTC) of creation of this file. Format : YYYYMMDDHHMISS Example : 20011229161700 : December 29th 2001 16 :17 :00
DATE_UPDATE	char DATE_UPDATE(DATE_TIME); DATE_UPDATE:long_name = "Date of update of this file"; DATE_UPDATE:conventions = "YYYYMMDDHHMISS"; DATE_UPDATE:_FillValue = " ";	Date and time (UTC) of update of this file. Format : YYYYMMDDHHMISS Example : 20011230090500 : December 30th 2001 09 :05 :00

3.4 General information for each profile

Each item of this section has a N_PROF (number of profiles) dimension.

Name	Definition	Comment
PLATFORM_NUMBER	char PLATFORM_NUMBER(N_PROF, STRING8); PLATFORM_NUMBER:long_name = "Glider unique identifier"; PLATFORM_NUMBER:conventions = "WMO glider identifier : A9IIIII"; PLATFORM_NUMBER:_FillValue = " ";	WMO glider identifier. WMO is the World Meteorological Organization. This platform number is unique. Example : 6900045
PROJECT_NAME	char PROJECT_NAME(N_PROF, STRING64); PROJECT_NAME:long_name = "Name of the project"; PROJECT_NAME:_FillValue = " ";	Name of the project which operates the glider that performed the profile.
PI_NAME	char PI_NAME (N_PROF, STRING64); PI_NAME:long_name = "Name of the principal investigator"; PI_NAME:_FillValue = " ";	Name of the principal investigator in charge of the glider.
STATION_PARAMETERS	char STATION_PARAMETERS(N_PROF, N_PARAM, STRING64); STATION_PARAMETERS:long_name = "List of available parameters for the station"; STATION_PARAMETERS:conventions = "EGO reference table 3"; STATION_PARAMETERS:_FillValue = " ";	List of parameters contained in this profile. The parameter names are listed in reference table 3. Examples : TEMP, PSAL, CNDC TEMP : temperature PSAL : practical salinity CNDC : conductivity
CYCLE_NUMBER	int CYCLE_NUMBER(N_PROF); CYCLE_NUMBER:long_name = "Glider descent or ascent number"; CYCLE_NUMBER:conventions = "1...N"; CYCLE_NUMBER:_FillValue = 99999;	Glider descent or ascent number.
DIRECTION	char DIRECTION(N_PROF); DIRECTION:long_name = "Direction of the station profiles"; DIRECTION:conventions = "A: ascending profiles, D: descending profiles"; DIRECTION:_FillValue = " ";	Type of profile on which measurement occurs. A : ascending profile D : descending profile
DATA_CENTRE	char DATA_CENTRE(N_PROF, STRING2); DATA_CENTRE:long_name = "Data centre in charge of glider data processing"; DATA_CENTRE:conventions = "EGO reference table 4"; DATA_CENTRE:_FillValue = " ";	Code for the data centre in charge of the glider data management. The data centre codes are described in the reference table 4. Example : "ME" for MEDS
DC_REFERENCE	char DC_REFERENCE(N_PROF, STRING32); DC_REFERENCE:long_name = "Station unique identifier in data centre"; DC_REFERENCE:conventions = "Data centre convention"; DC_REFERENCE:_FillValue = " ";	Unique identifier of the profile in the data centre. Data centres may have different identifier schemes. DC_REFERENCE is therefore not unique across data centres.
DATA_STATE_INDICATOR	char DATA_STATE_INDICATOR(N_PROF, STRING4); DATA_STATE_INDICATOR:long_name =	Degree of processing the data has passed through. The data state indicator is described in the

	"Degree of processing the data have passed through"; DATA_STATE_INDICATOR:conventions = "EGO reference table 5"; DATA_STATE_INDICATOR:_FillValue = " ";	reference table 5.
DATA_MODE	char DATA_MODE(N_PROF); DATA_MODE:long_name = "Delayed mode or real time data"; DATA_MODE:conventions = "R : real time; D : delayed mode; A : real time with adjustment"; DATA_MODE:_FillValue = " ";	Indicates if the profile contains real time, delayed mode or adjusted data. R : real time data D : delayed mode data A : real time data with adjusted values
PARAMETER_DATA_MODE	char PARAMETER_DATA_MODE(N_PROF, N_PARAM); PARAMETER_DATA_MODE:long_name = "Delayed mode or real time data"; PARAMETER_DATA_MODE:conventions = "R : real time; D : delayed mode; A : real time with adjustment"; PARAMETER_DATA_MODE:_FillValue = " ";	Describe the data mode of the individual parameter. R : real time data D : delayed mode data A : real time data with adjusted values
PLATFORM_TYPE	char PLATFORM_TYPE(N_PROF, STRING32); PLATFORM_TYPE:long_name = "Type of glider"; PLATFORM_TYPE:conventions = "EGO reference table 23"; PLATFORM_TYPE:_FillValue = " ";	Type of glider listed in reference table 23. Example: SEAGLIDER
GLIDER_SERIAL_NO	char GLIDER_SERIAL_NO(N_PROF, STRING32); GLIDER_SERIAL_NO:long_name = "Serial number of the glider"; GLIDER_SERIAL_NO:_FillValue = " ";	Serial number of the glider. Example 1679
FIRMWARE_VERSION	char FIRMWARE_VERSION(N_PROF, STRING64); FIRMWARE_VERSION:long_name = "Instrument firmware version"; FIRMWARE_VERSION:_FillValue = " ";	Firmware version of the glider. Example : "013108"
WMO_INST_TYPE	char WMO_INST_TYPE(N_PROF, STRING4); WMO_INST_TYPE:long_name = "Coded instrument type"; WMO_INST_TYPE:conventions = "EGO reference table 8"; WMO_INST_TYPE:_FillValue = " ";	Instrument type from WMO code table 1770. A subset of WMO table 1770 is documented in the reference table 8.
JULD	double JULD(N_PROF); JULD:long_name = "Julian day (UTC) of the station relative to REFERENCE_DATE_TIME"; JULD:standard_name = "time"; JULD:units = "days since 1950-01-01 00:00:00 UTC"; JULD:conventions = "Relative julian days with decimal part (as parts of day)"; JULD:resolution = X; JULD:_FillValue = 999999.; JULD:axis = "T";	Julian day of the profile. The integer part represents the day, the decimal part represents the time of the profile. Date and time are in Universal Time. The Julian day is relative to REFERENCE_DATE_TIME. Example : 18833.8013889885 : July 25 2001 19:14:00

JULD_QC	char JULD_QC(N_PROF); JULD_QC:long_name = "Quality on date and time"; JULD_QC:conventions = "EGO reference table 2.1"; JULD_QC:_FillValue = " ";	Quality flag on JULD date and time. The flag scale is described in the reference table 2.1. Example : 1: the date and time seems correct.
JULD_LOCATION	double JULD_LOCATION(N_PROF); JULD_LOCATION:long_name = "Julian day (UTC) of the location relative to REFERENCE_DATE_TIME"; JULD_LOCATION:units = "days since 1950-01-01 00:00:00 UTC"; JULD_LOCATION:conventions = "Relative julian days with decimal part (as parts of day)"; JULD_LOCATION:resolution = X; JULD_LOCATION:_FillValue = 999999.;	Julian day of the location of the profile. The integer part represents the day, the decimal part represents the time of the profile. Date and time are in Universal Time. The Julian day is relative to REFERENCE_DATE_TIME. Example : 18833.8013889885 : July 25 2001 19:14:00
LATITUDE	double LATITUDE(N_PROF); LATITUDE:long_name = "Latitude of the station, best estimate"; LATITUDE:standard_name = "latitude"; LATITUDE:units = "degree_north"; LATITUDE:_FillValue = 99999.; LATITUDE:valid_min = -90.; LATITUDE:valid_max = 90.; LATITUDE:axis = "Y";	Latitude of the profile. Unit : degree north This field contains the best estimated latitude. The latitude value may be improved in delayed mode. Example : 44.4991 : 44° 29' 56.76" N
LONGITUDE	double LONGITUDE(N_PROF); LONGITUDE:long_name = "Longitude of the station, best estimate"; LONGITUDE:standard_name = "longitude"; LONGITUDE:units = "degree_east"; LONGITUDE:_FillValue = 99999.; LONGITUDE:valid_min = -180.; LONGITUDE:valid_max = 180.; LONGITUDE:axis = "X";	Longitude of the profile. Unit : degree east This field contains the best estimated longitude. The longitude value may be improved in delayed mode. Example : 16.7222 : 16° 43' 19.92" E
POSITION_QC	char POSITION_QC(N_PROF); POSITION_QC:long_name = "Quality on position (latitude and longitude)"; POSITION_QC:conventions = "EGO reference table 2.1"; POSITION_QC:_FillValue = " ";	Quality flag on position. The flag on position is set according to (LATITUDE, LONGITUDE) quality. The flag scale is described in the reference table 2.1. Example: 1: position seems correct.
POSITIONING_SYSTEM	char POSITIONING_SYSTEM(N_PROF, STRING8); POSITIONING_SYSTEM:long_name = "Positioning system"; POSITIONING_SYSTEM:_FillValue = " ";	Name of the system in charge of positioning the glider locations from reference table 9.1. Examples : GPS
PROFILE_<PARAM>_QC	char PROFILE_<PARAM>_QC(N_PROF); PROFILE_<PARAM>_QC:long_name = "Global quality flag of <PARAM> profile"; PROFILE_<PARAM>_QC:conventions = "EGO reference table 2.1a";	Global quality flag on the PARAM profile. PARAM is among the STATION_PARAMETERS. The overall flag is set to indicate the percentage of good data in the profile as described in reference table 2.1a.

	PROFILE_<PARAM>_QC:_FillValue = " ";	Example : PROFILE_TEMP_QC = A : the temperature profile contains only good values PROFILE_PSAL_QC = C : the salinity profile contains 50% to 75% good values
VERTICAL_SAMPLING_SCHEME	char VERTICAL_SAMPLING_SCHEME(N_PROF, STRING256); VERTICAL_SAMPLING_SCHEME:long_name = "Vertical sampling scheme"; VERTICAL_SAMPLING_SCHEME:conventions = "EGO reference table 16"; VERTICAL_SAMPLING_SCHEME:_FillValue = " ";	Use the vertical sampling scheme to differentiate and identify profiles from a single descent or ascent with different vertical sampling schemes. See reference table 16.
CONFIG_MISSION_NUMBER	int CONFIG_MISSION_NUMBER(N_PROF); CONFIG_MISSION_NUMBER:long_name = " Unique number denoting the missions performed by the glider"; CONFIG_MISSION_NUMBER:conventions = "1...N, 1 : first complete mission"; CONFIG_MISSION_NUMBER:_FillValue = 99999;	Unique number of the mission to which this profile belongs.

Note: how to sort STATION_PARAMETERS variable

The parameters listed in STATION_PARAMETERS should be sorted in the same order within a given glider's data file.

3.5 Measurements for each profile

This section contains information on each level of each profile.

Each variable in this section has a N_PROF (number of profiles), N_LEVELS (number of pressure levels) dimension.

<PARAM> contains the raw values transmitted by the glider.

The values in <PARAM> should never be altered. <PARAM>_QC contains QC flags that pertain to the values in <PARAM>. Values in <PARAM>_QC are set initially in 'R' and 'A' modes by the automatic real-time tests.

They are later modified in 'D' mode at levels where the QC flags are set incorrectly by the real-time procedures, and where erroneous data are not detected by the real-time procedures.

Each parameter can be adjusted (in delayed-mode, but also in real-time if appropriate). In that case, <PARAM>_ADJUSTED contains the adjusted values, <PARAM>_ADJUSTED_QC contains the QC flags set by the adjustment process, and <PARAM>_ADJUSTED_ERROR contains the adjustment uncertainties.

When a profile has DATA_MODE = 'R', no adjusted data are available. Hence the adjusted section (<PARAM>_ADJUSTED, <PARAM>_ADJUSTED_QC and <PARAM>_ADJUSTED_ERROR) should be filled with FillValue.

When N_PROF > 1, DATA_MODE for each profile can be assigned differently. This is because when there are multiple profiles, delayed-mode or near real-time adjustments can become available at different times.

The adjusted section for each N_PROF should then be filled independently according to its DATA_MODE.

Name	Definition	Comment
<PARAM>	float <PARAM>(N_PROF, N_LEVELS); <PARAM>:long_name = "<X>"; <PARAM>:standard_name = "<X>"; <PARAM>:_FillValue = <X>; <PARAM>:units = "<X>"; <PARAM>:valid_min = <X>; <PARAM>:valid_max = <X>; <PARAM>:C_format = "<X>"; <PARAM>:FORTRAN_format = "<X>"; <PARAM>:resolution = <X>;	<PARAM> contains the original values of a parameter listed in reference table 3. <X> : this field is specified in the reference table 3.
<PARAM>_QC	char <PARAM>_QC(N_PROF, N_LEVELS); <PARAM>_QC:long_name = "quality flag"; <PARAM>_QC:conventions = "EGO reference table 2.1"; <PARAM>_QC:_FillValue = " ";	Quality flag applied on each <PARAM> value. The flag scale is specified in table 2.1.
<PARAM>_ADJUSTED	float <PARAM>_ADJUSTED(N_PROF, N_LEVELS); <PARAM>_ADJUSTED:long_name = "<X>"; <PARAM>_ADJUSTED:standard_name = "<X>"; <PARAM>_ADJUSTED:_FillValue = <X>; <PARAM>_ADJUSTED:units = "<X>";	<PARAM>_ADJUSTED contains the adjusted values derived from the original values of the parameter. <X> : this field is specified in the reference table 3. When no adjustment is performed, the

	<PARAM>_ADJUSTED:valid_min = <X>; <PARAM>_ADJUSTED:valid_max = <X>; <PARAM>_ADJUSTED:C_format = "<X>"; <PARAM>_ADJUSTED:FORTRAN_format = "<X>"; <PARAM>_ADJUSTED:resolution= <X>;	FillValue is inserted.
<PARAM>_ADJUSTED_QC	char <PARAM>_ADJUSTED_QC(N_PROF, N_LEVELS); <PARAM>_ADJUSTED_QC:long_name = "quality flag"; <PARAM>_ADJUSTED_QC:conventions = "EGO reference table 2.1"; <PARAM>_ADJUSTED_QC:_FillValue = " ";	Quality flag applied on each <PARAM>_ADJUSTED value. The flag scale is specified in reference table 2.1. When no adjustment is performed, the FillValue is inserted.
<PARAM>_ADJUSTED_ERROR	float <PARAM>_ADJUSTED_ERROR(N_PROF, N_LEVELS); <PARAM>_ADJUSTED_ERROR:long_name = "Contains the error on the adjusted values as determined by the delayed mode QC process"; <PARAM>_ADJUSTED_ERROR:_FillValue = <X>; <PARAM>_ADJUSTED_ERROR:units = "<X>"; <PARAM>_ADJUSTED_ERROR:C_format = "<X>"; <PARAM>_ADJUSTED_ERROR:FORTRAN_format = "<X>"; <PARAM>_ADJUSTED_ERROR:resolution= <X>;	<PARAM>_ADJUSTED_ERROR Contains the error on the adjusted values as determined by the delayed mode QC process. <X> : this field is specified in the reference table 3. When no adjustment is performed, the FillValue is inserted.

Note on vertical axis associated to PRES

The variable PRES (pressure) is the vertical axis. The PRES declaration contains the variable attribute

PRES:axis = "Z";

3.6 Calibration information for each profile

Calibrations are applied to parameters to create adjusted parameters. Different calibration methods will be used by groups processing EGO data. When a method is applied, its description is stored in the following fields.

This section contains calibration information for each parameter of each profile.

Each item of this section has a N_PROF (number of profiles), N_CALIB (number of calibrations), N_PARAM (number of parameters) dimension.

If no calibration is available, N_CALIB is set to 1, PARAMETER is filled with the list of parameter names, and all values of the calibration section are set to fill values.

Name	Definition	Comment
PARAMETER	char PARAMETER(N_PROF, N_CALIB, N_PARAM, STRING64); PARAMETER:long_name = "List of parameters with calibration information"; PARAMETER:conventions = "EGO reference table 3"; PARAMETER:_FillValue = " ";	Name of the calibrated parameter. The list of parameters is in reference table 3. Example : PSAL
SCIENTIFIC_CALIB_EQUATION	char SCIENTIFIC_CALIB_EQUATION(N_PROF, N_CALIB, N_PARAM, STRING256); SCIENTIFIC_CALIB_EQUATION:long_name = "Calibration equation for this parameter"; SCIENTIFIC_CALIB_EQUATION:_FillValue = " ";	Calibration equation applied to the parameter. Example : Tc = a1 * T + a0
SCIENTIFIC_CALIB_COEFFICIENT	char SCIENTIFIC_CALIB_COEFFICIENT(N_PROF, N_CALIB, N_PARAM, STRING256); SCIENTIFIC_CALIB_COEFFICIENT:long_name = "Calibration coefficients for this equation"; SCIENTIFIC_CALIB_COEFFICIENT:_FillValue = " ";	Calibration coefficients for this equation. Example : a1=0.99997 , a0=0.0021
SCIENTIFIC_CALIB_COMMENT	char SCIENTIFIC_CALIB_COMMENT(N_PROF, N_CALIB, N_PARAM, STRING256); SCIENTIFIC_CALIB_COMMENT:long_name = "Comment applying to this parameter calibration"; SCIENTIFIC_CALIB_COMMENT:_FillValue = " ";	Comment about this calibration Example : The sensor is not stable
SCIENTIFIC_CALIB_DATE	char SCIENTIFIC_CALIB_DATE (N_PROF N_CALIB, N_PARAM, DATE_TIME) SCIENTIFIC_CALIB_DATE:long_name = "Date of calibration"; SCIENTIFIC_CALIB_DATE:conventions = "YYYYMMDDHHMISS"; SCIENTIFIC_CALIB_DATE:_FillValue = " ";	Date of the calibration. Example : 20011217161700

3.7 History information for each profile

This section contains history information for each action performed on each profile by a data centre.

Each item of this section has a N_HISTORY (number of history records), N_PROF (number of profiles) dimension.

A history record is created whenever an action is performed on a profile.

The recorded actions are coded and described in the history code table from the reference table 7.

Name	Definition	Comment
HISTORY_INSTITUTION	char HISTORY_INSTITUTION(N_HISTORY, N_PROF, STRING4); HISTORY_INSTITUTION:long_name = "Institution which performed action"; HISTORY_INSTITUTION:conventions = "EGO reference table 4"; HISTORY_INSTITUTION:_FillValue = " ";	Institution that performed the action. Institution codes are described in reference table 4. Example : ME for MEDS
HISTORY_STEP	char HISTORY_STEP(N_HISTORY, N_PROF, STRING4); HISTORY_STEP:long_name = "Step in data processing"; HISTORY_STEP:conventions = "EGO reference table 12"; HISTORY_STEP:_FillValue = " ";	Code of the step in data processing for this history record. The step codes are described in reference table 12. Example : ARGQ : Automatic QC of data reported in real-time has been performed
HISTORY_SOFTWARE	char HISTORY_SOFTWARE (N_HISTORY, N_PROF, STRING4); HISTORY_SOFTWARE:long_name = "Name of software which performed action"; HISTORY_SOFTWARE:conventions = "Institution dependent"; HISTORY_SOFTWARE:_FillValue = " ";	Name of the software that performed the action. This code is institution dependent. Example : WJO
HISTORY_SOFTWARE_RELEASE	char HISTORY_SOFTWARE_RELEASE(N_HISTORY, N_PROF, STRING4); HISTORY_SOFTWARE_RELEASE:long_name = "Version/release of software which performed action"; HISTORY_SOFTWARE_RELEASE:conventions = "Institution dependent"; HISTORY_SOFTWARE_RELEASE:_FillValue = " ";	Version of the software. This name is institution dependent. Example : «1.0»
HISTORY_REFERENCE	char HISTORY_REFERENCE (N_HISTORY, N_PROF, STRING64); HISTORY_REFERENCE:long_name = "Reference of database"; HISTORY_REFERENCE:conventions = "Institution dependent"; HISTORY_REFERENCE:_FillValue = " ";	Code of the reference database used for quality control in conjunction with the software. This code is institution dependent. Example : WOD2001
HISTORY_DATE	char HISTORY_DATE(N_HISTORY,	Date of the action.

	N_PROF, DATE_TIME); HISTORY_DATE:long_name = "Date the history record was created"; HISTORY_DATE:conventions = "YYYYMMDDHHMISS"; HISTORY_DATE:_FillValue = " ";	Example : 20011217160057
HISTORY_ACTION	char HISTORY_ACTION(N_HISTORY, N_PROF, STRING4); HISTORY_ACTION:long_name = "Action performed on data"; HISTORY_ACTION:conventions = "EGO reference table 7"; HISTORY_ACTION:_FillValue = " ";	Name of the action. The action codes are described in reference table 7. Example : QCF$ for QC failed
HISTORY_PARAMETER	char HISTORY_PARAMETER(N_HISTORY, N_PROF, STRING16); HISTORY_PARAMETER:long_name = "Station parameter action is performed on"; HISTORY_PARAMETER:conventions = "EGO reference table 3"; HISTORY_PARAMETER:_FillValue = " ";	Name of the parameter on which the action is performed. Example : PSAL
HISTORY_START_PRES	float HISTORY_START_PRES(N_HISTORY, N_PROF); HISTORY_START_PRES:long_name = "Start pressure action applied on"; HISTORY_START_PRES:_FillValue = 99999.f; HISTORY_START_PRES:units = "decibar";	Start pressure the action is applied to. Example : 1500.0
HISTORY_STOP_PRES	float HISTORY_STOP_PRES(N_HISTORY, N_PROF); HISTORY_STOP_PRES:long_name = "Stop pressure action applied on"; HISTORY_STOP_PRES:_FillValue = 99999.f; HISTORY_STOP_PRES:units = "decibar";	Stop pressure the action is applied to. **This should be greater than or equal to START_PRES.** Example : 1757.0
HISTORY_PREVIOUS_VALUE	float HISTORY_PREVIOUS_VALUE(N_HISTORY, N_PROF); HISTORY_PREVIOUS_VALUE:long_name = "Parameter/Flag previous value before action"; HISTORY_PREVIOUS_VALUE:_FillValue = 99999.f;	Parameter or flag of the previous value before action. Example : 2 (probably good) for a flag that was changed to 1 (good)
HISTORY_QCTEST	char HISTORY_QCTEST(N_HISTORY, N_PROF, STRING16); HISTORY_QCTEST:long_name = "Documentation of tests performed, tests failed (in hex form)"; HISTORY_QCTEST:conventions = "Write tests performed when	This field records the tests performed when ACTION is set to QCP$ (QC performed), the test failed when ACTION is set to QCF$ (QC failed). The QCTEST codes are described in reference table 11. Example : 0A (in hexadecimal form)

	ACTION=QCP$; tests failed when ACTION=QCF$"; HISTORY_QCTEST:_FillValue = " ";	

The usage of the History section is described in §5 "Using the History section of the EGO netCDF Structure".

4 Reference tables

This chapter gives the reference tables used by the EGO format.

4.1 Reference table 1: data types

The data_type global attribute should have the following value:

Data type
EGO glider time-series data

4.2 Reference table 2.1: variable quality control flag scale

This table is shared with the Argo project.

The quality control flags indicate the data quality of the data values in a file, and are normally assigned after quality control procedures have been performed. These codes are used in the <PARAM>_QC, TIME_QC, POSITION_QC variables to describe the quality of each measurement.

Code	Meaning	Comment
0	No QC was performed	-
1	Good data	All QC tests passed.
2	Probably good data	-
3	Bad data that are potentially correctable	These data are not to be used without scientific correction or re-calibration.
4	Bad data	Data have failed one or more tests.
5	Value changed	Data may be recovered after transmission error.
6	-	Not used.
8	Estimated value	Estimated value (interpolated, extrapolated or other estimation).
9	Missing value	-

4.2.1 Reference table 2.1a: overall profile quality flag

N is defined as the percentage of levels with good data where:

- QC flag values of 1, 2, 5, 8 are GOOD data
- QC flag values of 0 (no QC), 9 (missing) or " " (FillValue) are NOT USED in the computation
- QC flag values of 3, 4 are BAD data

The computation should be taken from <PARAM_ADJUSTED>_QC if available and from <PARAM>_QC otherwise.

N	Meaning
" "	No QC is performed or no usable flag values are present.
A	N = 100%; all profile levels contain good data.
B	75% <= **N** < 100%
C	50% <= **N** < 75%
D	25% <= **N** < 50%
E	0% < **N** < 25%
F	**N** = 0%; No profile levels have good data.

Example: a TEMP profile has 60 levels (3 levels contain missing values).

- 45 levels are flagged as 1
- 5 levels are flagged as 2
- 7 levels are flagged as 4
- 3 levels are flagged as 9 (missing)

Percentage of good levels = ((45 + 5) / 57) * 100 = 87.7%

PROFILE_TEMP_QC = "B".

4.3 Reference table 2.2: cell methods

From NetCDF Climate and Forecast (CF) Metadata Conventions, Version 1.2, 4 May, 2008. In the Units column, *u* indicates the units of the physical quantity before the method is applied.

Cell methods	Units	Description
point	u	The data values are representative of points in space or time (instantaneous).
sum	u	The data values are representative of a sum or accumulation over the cell.
maximum	u	Maximum
median	u	Median
mid_range	u	Average of maximum and minimum
minimum	u	Minimum
mean	u	Mean (average value)
mode	u	Mode (most common value)
standard_deviation	u	Standard deviation
variance	u2	Variance

4.4 Reference table 3: EGO parameter dictionary

This table is shared with the Argo project.

4.4.1 Convention for parameter names, standard names and units

- Parameter names should start with a code based on SeaDataNet-BODC parameter discovery vocabulary.
  They are not strictly standardized, however.
  When necessary, a parameter name has a suffix that designates secondary parameters . The suffix starts with the character "_".
- The NetCDF "standard_name" attribute contains the standardized parameter name from CF conventions.
- The NetCDF "units" attribute are compliant with UDUNITS as implemented in the CF/COARDS standards.

As the parameter names are not strictly standardized, one should use the standard_name attribute to query a particular measurement from different data files.
Relevant information on the parameter is recorded in the attributes of the parameter; _xxx in the parameter name is just a guide

Example

On a glider, sea temperature measured by a series of Microcat CTD is reported as TEMP, with a standard name of SEA_WATER_TEMPERATURE.

Secondary temperature measurement performed by an oxygen sensor is reported as DOXY_TEMP with a standard name of temperature_of_sensor_for_oxygen_in_sea_water.

For both measurements, the unit attribute is "degree_Celsius".

4.4.2 EGO parameter list

The EGO parameter list is based on Argo reference parameters, it is available from:

http://www.argodatamgt.org/Documentation

An additional list, for EGO parameters not managed in Argo, is provided below.

Parameter name	long_name	cf standard_name	unit	Core/bio/intermediate parameter
BBP650	Particle backscattering at 650 nanometers		m-1	b
BBP880	Particle backscattering at 880 nanometers		m-1	b
FLUORESCENCE_ VOLTAGE_CHLA	Chlorophyll-A signal from analogic fluorescence sensor			i
FLUORESCENCE_ URANINE	Uranine signal from fluorescence sensor		count	i
URANINE	Uranine		ppb	b

FLUORESCENCE_RHODAMINE	Rhodamine signal from fluorescence sensor		count	i
RHODAMINE	Rhodamine		ppb	b
BETA_BACKSCATTERING700_SCALED	Total angle specific volume from backscattering sensor at 700 nanometers with factory calibration		m-1 sr-1	b
BETA_BACKSCATTERING470_SCALED	Total angle specific volume from backscattering sensor at 470 nanometers with factory calibration		m-1 sr-1	b

4.4.3 References

The EGO standard names are taken from the CF standard names, available at:

- http://cfconventions.org/standard-names.htmlThe parameter names are based on SeaDataNet-BODC parameter discovery vocabulary available at:
- http://seadatanet.maris2.nl/v_bodc_vocab/welcome.aspx
  Select P021, " BODC Parameter Discovery Vocabulary"

The units are compliant with UDUNITS, as implemented by the CF standard; definitions are available at:

- http://www.unidata.ucar.edu/software/udunits **Reference table 4: DAC and institution codes**

Code	Data Assembly Centers and institutions
IF	Ifremer for Coriolis (French joint project for operational oceanography)
BO	British Oceanographic Data Center
NM	Norvegian Marine Data Center
DF	Department of Fisheries and Oceans
OG	Istituto Nazionale di Oceanographia e di Geofisica Sperimentale : OGS
SO	Sistema d'Observació Costaner i de Predicció
IO	Integrated Ocean Observing System
TU	Tallin University of Technology
IM	Integrated Marine Observing System

4.6 Reference table 5: data state indicators

Level	Descriptor
0	Data are the raw output from instruments, without calibration, and not necessarily converted to engineering units. These data are rarely exchanged
1	Data have been converted to values independent of detailed instrument knowledge. Automated calibrations may have been done. Data may not have full geospatial and temporal referencing, but have sufficient information to uniquely reference the data to the point of measurement.
2	Data have complete geospatial and temporal references. Information may have been compressed (e.g. sub-sampled, averaged, etc.) but no assumptions of scales of variability or thermodynamic relationships have been used in the processing.
3	The data have been processed with assumptions about the scales of variability or thermodynamic relationships. The data are normally reduced to regular space, time intervals with enhanced signal to noise.

Class	Descriptor	Subclass
A	No scrutiny, value judgement or intercomparisons are performed on the data. The records are derived directly from the input with no filtering, or sub-sampling.	- Some reductions or sub-sampling has been performed, but the original record is available. + Geospatial and temporal properties are checked. Geophysical values are validated. If not validated, this is clearly indicated.
B	Data have been scrutinized and evaluated against a defined and documented set of measures. The process is often automated (i.e. has no human intervention) and the measures are published and widely available.	- Measures are completely automated, or documentation is not widely available. + The measures have been tested on independent data sets for completeness and robustness and are widely accepted.
C	Data have been scrutinized fully including intra-record and intra-dataset comparison and consistency checks. Scientists have been involved in the evaluation and brought latest knowledge to bear. The procedures are published, widely available and widely accepted.	- Procedures are not published or widely available. Procedures have not undergone full scrutiny and testing. + Data are fully quality controlled, peer reviewed and are widely accepted as valid. Documentation is complete and widely available.

Data state indicator recommended use

The following table describes the processing stage of data and the value to be assigned the data state indicator (DS Indicator). It is the concatenation of level and class described above.

Processing Stage	DS Indicator
1. Data pass through a communications system and arrive at a processing centre. The data resolution is the highest permitted by the technical constraints of the floats and communications system.	0A (note 1)
2. The national centre assembles all of the raw information into a complete profile located in space and time.	1A (note 2)
3. The national centre passes the data through automated QC procedures and prepares the data for distribution on the GTS, to global servers and to PIs.	2B
4. Real-time data are received at global data centres that apply QC including visual inspection of the data. These are then distributed to users in near real-time	2B+ (note 3)
5. Data are reviewed by PIs and returned to processing centres. The processing centres forward the data to the global Argo servers.	2C
6. Scientists accept data from various sources, combine them as they see fit with other data and generate a product. Results of the scientific analysis may be returned to regional centres or global servers. Incorporation of these results improves the quality of the data.	2C+
7. Scientists working as part of GODAE generate fields of gridded products delivered in near real-time for distribution from the global servers. Generally, these products mostly will be based on data having passed	3B (note 4)

through automated QC procedures.	
8. Scientists working as part of GODAE generate fields of gridded products delivered with some time delay for distribution from the global servers. Generally, these products mostly will be based on data having passed through manual or more sophisticated QC procedures than employed on the real-time data.	3C

Notes

1. We need to have a pragmatic approach to what constitutes "original" or "raw" data. Despite the fact that an instrument may be capable of high sampling rates, what is reported from the instrument defines what is considered "raw". For example, Argo floats can certainly sample at finer scales than every 10 db, but because of communications, all we see for now is data at that (or worse) vertical resolution. Therefore the data "coming from the instrument" is "raw" output at 10dbar resolution.
2. The conversion of the raw data stream from the communications system into profiles of variables causes the data state indicator to switch from level 0 to 1.
3. Even though the data at global data centres use manual or semi-automated QC procedures, there is often not the intercomparisons to larger data collections and fields that would qualify the data state indicator to be set to class C. This is generally only provided by scientific scrutiny of the data.
4. The transition from class 2 to 3 occurs when assumptions of scales of variability are applied. During the course of normal data processing it is common to carry out some averaging and sub-sampling. This is usually done to exploit oversampling by the instrument, and to ensure good measurements are achieved. These are considered to be part of the geospatial and temporal referencing process.

4.7 Reference table 6: EGO file update interval

Interval	Meaning
hourly	
daily	
yearly	
void	Use "void" for delayed-mode or archive data that do not need continuous updating

4.8 Reference table 7: history action codes

This table is shared with the Argo project.

Code	Meaning
CF	Change a quality flag
CR	Create record
CV	Change value
DC	Station was checked by duplicate checking software
ED	Edit a parameter value
IP	This history group operates on the complete input record
NG	No good trace
PE	Position error. Profile position has been erroneously encoded. Corrected if possible.
QC	Quality Control
QCF$	Tests failed
QCP$	Test performed
SV	Set a value
TE	Time error. Profile date/time has been erroneously encoded. Corrected if possible.
UP	Station passed through the update program

4.9 Reference table 8: instrument types

The instrument type codes come from WMO table 1770.

Glider instrument codes should be requested from WMO.

As a default value, EGO uses the instrument type 830 : CTD.

Code number	Instrument
830	CTD

4.10 Reference table 9.1: positioning systems

This table is shared with the Argo project.

Code	Description
ARGOS	ARGOS positioning system
GPS	GPS positioning system
IRIDIUM	Iridium positioning system

4.11 Reference table 9.2: glider phases

A glider regularly performs surface, descent, inflexion, subsurface drift and ascent phases.

During ascent or descent phase, the glider performs vertical profiles.

Code	Meaning	Comment
0	surface drift	the glider is drifting on the surface layer
1	descent	the glider is descending
2	subsurface drift	the glider is drifting in subsurface
3	inflexion	the glider is changing its trajectory
4	ascent	the glider is ascending
5	grounded	the glider touched the ground or seafloor or onshore
6	inconsistent	the glider pressure is not consistent with the surrounding pressures

4.12 Reference table 10.1: transmission systems

Code	Description
IRIDIUM	Satellite transmission system (at sea)
FREEWAVE	Radio transmission system (deployment and recovery)

4.13 Reference table 10.2: positioning methods

The positions reported in variables latitude and longitude are reported from various sources such as: GPS, Argos, glider or interpolation.

Code	Meaning	Comment
0	gps	GPS positioning method.
1	argos	Argos positioning method.
2	interpolated	position derived from other positions.
3	glider	Glider internal estimated positions.

4.14 Reference table 11: QC test binary IDs

This table is shared with the Argo project.

This table is used to record the result of the quality control tests in the history section.

The binary IDs of the QC tests are used to define the history variable HISTORY_QCTEST, whose value is computed by adding the binary ID together, then translating to a hexadecimal number. An example is given on §5.3.

The test numbers and the test names are listed in the Argo Quality Control Manual:

- §2.1 "Argo Real-Time Quality Control Test Procedures on Vertical Profiles", and
- §2.2 "Argo Real-Time Quality Control Test Procedures on Trajectories"

See http://www.argodatamgt.org/Documentation .

Test number	QC test binary ID	Test name
1	2	Platform Identification test
2	4	Impossible Date test
3	8	Impossible Location test
4	16	Position on Land test
5	32	Impossible Speed test
6	64	Global Range test
7	128	Regional Global Parameter test
8	256	Pressure Increasing test
9	512	Spike test
10	*1024*	*Top and Bottom Spike test (obsolete)*
11	2048	Gradient test
12	4096	Digit Rollover test
13	8192	Stuck Value test
14	16384	Density Inversion test
15	32768	Grey List test
16	65536	Gross Salinity or Temperature Sensor Drift test
17	131072	Visual QC test
18	261144	Frozen profile test
19	524288	Deepest pressure test
20	1044576	Questionable Argos position test

4.15 Reference table 12: history steps codes

This table is shared with the Argo project.

Code	Meaning
ARFM	Convert raw data from telecommunications system to a processing format
ARGQ	Automatic QC of data reported in real-time has been performed
IGO3	Checking for duplicates has been performed
ARSQ	Delayed mode QC has been performed
ARCA	Calibration has been performed
ARUP	Real-time data have been archived locally and sent to GDACs
ARDU	Delayed data have been archived locally and sent to GDACs
RFMT	Reformat software to convert hexadecimal format reported by the buoy to our standard format
COOA	Coriolis objective analysis performed

If individual centers wish to record other codes, they may add to this list as they feel is appropriate.

4.16 Reference table 16: vertical sampling schemes

This variable differentiates the various vertical sampling schemes for multiple profiles from a single descent or ascent.
This variable can vary between cycles to accommodate gliders with two-way communication capabilities.
The profile with N_PROF=1 is required to be the Primary sampling profile. Other profiles will have N_PROF > 1 in any order. There can be only one Primary sampling profile, while other vertical sampling schemes can have more than one profile.

Code (STRING256) FORMAT → name: nominal measurement type [full description] [ ] indicates optional	N_PROF	Code Description
Primary sampling: averaged [description] or Primary sampling: discrete [description] or Primary sampling: mixed [description]	1	Primary CTD measurements and measurements from auxiliary sensors that are taken at the same pressure levels and with the same sampling method as the Primary CTD profile. For auxiliary sensor measurements it is not required that all pressure levels contain data.
Secondary sampling: averaged [description] or Secondary sampling: discrete [description] or Secondary sampling: mixed [description]	>1	Excluding "Primary sampling", this profile includes measurements that are taken at pressure levels different from the Primary CTD profile, or with sampling methods different from the Primary CTD profile. Measurements can be taken by the Primary CTD or by auxiliary sensors.
Near-surface sampling: averaged, pumped/unpumped [description] or Near-surface sampling: discrete, pumped/unpumped [description] or Near-surface sampling: mixed, pumped/unpumped [description]	>1	This profile includes near-surface measurements that are focused on the top 5dbar of the sea surface. (For the purpose of cross-calibration, this profile can extend deeper than the top 5dbar so as to overlap with the Primary sampling profile.) These measurements are taken at pressure levels different from the Primary CTD profile, or with sampling methods different from the Primary CTD profile. If the Primary sampling profile measures above 5dbar in the same manner as deeper data, there is no need to place the near-surface data here.
Bounce sampling: averaged [description] or Bounce sampling: discrete [description] or Bounce sampling: mixed [description]	>1	This scheme contains profiles that are collected on multiple rises/falls during a single cycle. The profiles are temporally offset from each other and/or the Primary sampling profile. They can be sampled with the Primary CTD or with auxiliary sensors.
Use the term 'averaged' if the data in the profile are pressure binned averages using multiple data measurements (pollings) from a sensor. Use the term 'discrete' if the data in the profile are from a single polling from a sensor. If both methods are used in the profile, use the term 'mixed'.		

4.17 Reference table 19: data modes

The values for the global attribute “data_mode” is defined as follows:

Value	Meaning
R	Real-time data. Data coming from the (typically remote) platform through a communication channel without physical access to the instruments, disassembly or recovery of the platform. Example: for a glider with a radio communication, this would be data obtained through the radio.
P	Provisional data. Data obtained after the instruments or the platform have been recovered or serviced. Example: for instruments on a glider, this would be data downloaded directly from the instruments after the glider has been recovered on a ship.
A	Rea-time adjusted data. Real-time or provisional data that have been adjusted by real-time automatic procedures.
D	Delayed-mode data. Data published after all calibrations and quality control procedures have been applied on the internally recorded or best available original data. This is the best possible version of processed data.
M	Mixed. This value indicates that the file contains data in more than one of the above states.

4.18 Reference table 20: sensor mount characteristics

The <PARAM>:”sensor_mount” attribute indicates the way a sensor is mounted on a glider.

The following table lists the valid sensor_mount attribute values.

Sensor mount
MOUNTED_ON_GLIDER

4.19 Reference table 21: sensor orientation characteristics

When appropriate, the <PARAM>:”sensor_orientation” attribute indicates the way a sensor is oriented on a glider.

The following table lists the valid sensor_orientation attribute values.

Sensor orientation	Description
DOWNWARD	Example : ADCP measuring from surface to bottom currents.
UPWARD	Example : In-line ADCP measuring currents towards the surface
FORWARD	
BACKWARD	

4.20 Reference table 22: glider categories

Glider category	Description
COASTAL_GLIDER	Operating up to 200 meters depth.
OPEN_OCEAN_GLIDER	Operating up to 1 000 meters depth.
DEEP_GLIDER	Operating below 1 000 meters depth.

4.21 Reference table 23: glider types

Glider type	Description
SLOCUM_SG1	Manufactured by Webb Research Corporation.
SLOCUM_SG2	Manufactured by Webb Research Corporation.
SLOCUM_SG3	Manufactured by Webb Research Corporation.
SEAGLIDER	Manufactured by Kongsberg.
SEAEXPLORER	Manufactured by Alseamar.
SPRAY	Manufactured by Bluefin Robotics.

4.22 Reference table 24: glider manufacturers

Code	Glider manufacturer	Comment
WRC	Webb Research Corporation	SLOCUM_SG1, SLOCUM_SG2, SLOCUM_SG3
KONGSBERG	Kongsberg	SEAGLIDER
ALSEAMAR	Alseamar	SEAEXPLORER
BLUEFIN_ROBOTICS	Bluefin Robotics	SPRAY
UNIVERSITY_OF_WASHINGTON	University of Washington	SEAGLIDER

4.23 Reference table 25: sensors

This table is shared with the Argo project.

For now, the EGO sensor list is based on the Argo one. Please note that this reference table is frequently updated to include new sensors. You can find the latest version of this list at: http://tinyurl.com/nwpqvp2.

On September 5th 2018, the table 25 content was:

SENSOR	Comment
UNKNOWN	to use when necessary
ACOUSTIC	
ACOUSTIC_GEOLOCATION	
CTD_PRES	
CTD_TEMP	
CTD_CNDC	
EM	The EM sensor measures U, V, and turbulence.
FLUOROMETER_CDOM	
FLUOROMETER_CHLA	
IDO_DOXY	
OPTODE_DOXY	
RADIOMETER_DOWN_IRR<nnn>	Radiometer measuring downwelling irradiance at wavelength <nnn>.
RADIOMETER_PAR	
RADIOMETER_UP_RAD<nnn>	Radiometer measuring upwelling radiance at wavelength <nnn>.
BACKSCATTERINGMETER_BBP<nnn>	Backscattering meter measuring backscattering at wavelength <nnn>.
BACKSCATTERINGMETER_TURBIDITY	
SPECTROPHOTOMETER_NITRATE	
SPECTROPHOTOMETER_BISULFIDE	
STS_CNDC	
STS_TEMP	
TRANSISTOR_PH	
TRANSMISSOMETER_CP<nnn>	Transmissometer measuring attenuation at wavelength <nnn>
FLOATCLOCK_MTIME	

An additional list, for EGO sensors not managed in Argo, is provided below.

SENSOR	Comment
FLUOROMETER_URANINE	
FLUOROMETER_RHODAMINE	

4.24 Reference table 26: sensor makers

This table is shared with the Argo project.

For now, the EGO sensor maker list is based on the Argo one. Please note that this reference table is frequently updated to include new sensor makers. You can find the latest version of this list at: http://tinyurl.com/nwpqvp2.

On September 5th 2018, the table 26 content was:

SENSOR_MAKER	Description	Comment
UNKNOWN	Unknown sensor maker	The sensor maker should not be unknown, the use of this value should be exceptional
AANDERAA		
AMETEK		
DRUCK		
FSI		
KISTLER		
PAINE		
SBE		
SEASCAN		
WETLABS	Wetlabs Inc.	
MBARI	Monterey Bay Aquarium Research Institute	
SATLANTIC		
JAC	JFE Advantech Co., Ltd	
APL_UW	University of Washington, Applied Physics Laboratory	
TSK	Tsurumi-Seiki Co., Ltd	
RBR	RBR Ltd	
KELLER		
MICRON		
SEAPOINT		
TURNER_DESIGN		

4.25 Reference table 27: sensor models

The EGO sensor model list is composed of the Argo sensor model list (first table below) and the EGO specific sensor model list (second table below).

Please note that Argo sensor model list is frequently updated to include new sensor makers. You can find the latest version of this list at: http://tinyurl.com/nwpqvp2.

The SENSOR_MODEL variable is standardized, i.e. we expect the manufacturer followed by the standard model number, i.e. SBE41CP or AANDERAA_3830. If there is a version number for a particular model then this is added at the end, i.e. SBE41CP_V1, SBE41CP_V1.2

On September 5th 2018, the Argo reference table 27 content was:

SENSOR_MODEL	Comment	Associated SENSOR
UNKNOWN	to use when necessary	UNKNOWN
Conductivity/temperature sensors		
FSI		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE37		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41_V2.5		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41_V2.6		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41_V3		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41_IDO_V1.0c	Seabird CTD module with IDO oxygen sensor	CTD_PRES, CTD_TEMP, CTD_CNDC,IDO_DOXY
SBE41_IDO_V2.0	Seabird CTD module with IDO oxygen sensor	CTD_PRES, CTD_TEMP, CTD_CNDC,IDO_DOXY
SBE41CP_IDO_V2.0b	Seabird CTD module with IDO oxygen sensor	CTD_PRES, CTD_TEMP, CTD_CNDC,IDO_DOXY
SBE41_IDO_V3.0	Seabird CTD module with IDO oxygen sensor	CTD_PRES, CTD_TEMP, CTD_CNDC,IDO_DOXY
SBE41CP_V1		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.1		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.2		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.2a		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.3		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.3b		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.4		CTD_PRES, CTD_TEMP, CTD_CNDC

SBE41CP_V1.5		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.7		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.8		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.9		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V1.9a		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V2		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V3		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V3.0a		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V3.0c		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V4.4.0		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V5.0.1		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V5.3.0		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V7.2.3		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41CP_V7.2.5		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41N		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41N_V5.3.0		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE41N_V5.4.0		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE61_V4.5.2		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE61_V4.5.3		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE61_V5.0.0		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE61_V5.0.1		CTD_PRES, CTD_TEMP, CTD_CNDC
SBE61		CTD_PRES, CTD_TEMP, CTD_CNDC
CTD_F01	TSK model	CTD_PRES, CTD_TEMP, CTD_CNDC
RBR	RBR model	CTD_PRES, CTD_TEMP, CTD_CNDC
RBRoem_V1.16	RBR model	CTD_PRES, CTD_TEMP, CTD_CNDC
Oxygen sensors		
SBE43_IDO	Seabird Electrochemical Dissolved Oxygen IDO sensor (volt output)	IDO_DOXY
SBE43I	configuration option	IDO_DOXY
SBE43F_IDO	Seabird Electrochemical Dissolved Oxygen IDO sensor (frequency output)	IDO_DOXY
SBE63_OPTODE	Seabird Optical Dissolved Oxygen Sensor	OPTODE_DOXY

AANDERAA_OPTODE		OPTODE_DOXY
AANDERAA_OPTODE_3830		OPTODE_DOXY
AANDERAA_OPTODE_3835		OPTODE_DOXY
AANDERAA_OPTODE_3930		OPTODE_DOXY
AANDERAA_OPTODE_4330		OPTODE_DOXY
AANDERAA_OPTODE_4330F		OPTODE_DOXY
AANDERAA_OPTODE_4831	Similar to AANDERAA_OPTODE_4330 for DO processing	OPTODE_DOXY
AANDERAA_OPTODE_4831F	Similar to AANDERAA_OPTODE_4330 for DO processing	OPTODE_DOXY
ARO_FT	JAC RINKO	OPTODE_DOXY
AROD_FT	deep housing	OPTODE_DOXY
Pressure sensors		
DRUCK_2900PSIA		CTD_PRES
DRUCK		CTD_PRES
DRUCK_10153PSIA		CTD_PRES
PAINE		CTD_PRES
PAINE_1500PSIA		CTD_PRES
PAINE_1600PSIA		CTD_PRES
PAINE_2000PSIA		CTD_PRES
PAINE_2900PSIA		CTD_PRES
PAINE_3000PSIA		CTD_PRES
AMETEK		CTD_PRES
AMETEK_3000PSIA		CTD_PRES
KISTLER		CTD_PRES
KISTLER_2900PSIA		CTD_PRES
KISTLER_10153PSIA		CTD_PRES
KELLER_PA8		CTD_PRES
SEASCAN_SSTD		CTD_PRES
MP40_C_2000_G	MICRON	CTD_PRES
Near Surface conductivity and temperature sensors		
SBE_STS	SBE Near Surface Conductivity and Temperature Module	STS_CNDC, STS_TEMP
Biogeochemical sensors (*)		

Spectrophotometers		
SUNA	UV absorption to derive nitrate and bisulfide (MBARI)	SPECTROPHOTOMETER_NITRATE, SPECTROPHOTOMETER_BISULFIDE
SUNA_V2	UV absorption to derive nitrate and bisulfide (SATLANTIC)	SPECTROPHOTOMETER_NITRATE, SPECTROPHOTOMETER_BISULFIDE
ISUS	Nitrate (MBARI)	SPECTROPHOTOMETER_NITRATE
ISUS_V3	Nitrate (SATLANTIC)	SPECTROPHOTOMETER_NITRATE
Transmissometers		
C_ROVER	Transmissometer (WETLABS)	TRANSMISSOMETER_CP<nnn>
pH Sensors		
DURA	pH (MBARI)	TRANSISTOR_PH
SEAFET	pH (SEABIRD)	TRANSISTOR_PH
Radiometers		
SATLANTIC_OCR504_ICSW	Multispectral radiometer 4 channels (SATLANTIC) with cosine detector to measure irradiance in water (at wavelength <nnn>)	RADIOMETER_DOWN_IRR<nnn>, RADIOMETER_PAR
SATLANTIC_OCR504_R10W	Multispectral radiometer 4 channels (SATLANTIC) with a 10° half-angle field of view to measure radiance in water (at wavelength <nnn>)	RADIOMETER_UP_RAD<nnn>
SATLANTIC_OCR507_ICSW	Multispectral radiometer 7 channels (SATLANTIC) with cosine detector to measure irradiance in water (at wavelength <nnn>)	RADIOMETER_DOWN_IRR<nnn>, RADIOMETER_PAR
SATLANTIC_OCR507_R10W	Multispectral radiometer 7 channels (SATLANTIC) with a 10° half-angle field of view to measure radiance in water (at wavelength <nnn>)	RADIOMETER_UP_RAD<nnn>

SATLANTIC_OCR507_ICSWR10W	Multispectral radiometer 7 channels (SATLANTIC) combination of cosine detector to measure irradiance and a 10° half-angle field of view to measure radiance in water (at wavelength <nnn>)	RADIOMETER_DOWN_IRR<nnn> , RADIOMETER_PAR, RADIOMETER_UP_RAD<nnn>
SATLANTIC_PAR		RADIOMETER_PAR
Backscatteringmeters and Fluorometers combination		
ECO_BB	Wetlabs Eco optical sensor packages with one backscattering meter (at wavelength <nnn>)	BACKSCATTERINGMETER_BBP<nnn>
ECO_FL	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR)	FLUOROMETER_CHLA
ECO_NTU	Wetlabs Eco optical sensor packages with one backscattering meter measuring turbidity	BACKSCATTERINGMETER_TURBIDITY
ECO_FLBB	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR) and one backscattering meter (at wavelength <nnn>)	FLUOROMETER_CHLA, BACKSCATTERINGMETER_BBP<nnn>
ECO_FLBB_AP2	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR) and one backscattering meter (wavelength specified in SENSOR) mounted on an Apex float	FLUOROMETER_CHLA, BACKSCATTERINGMETER_BBP<nnn>
ECO_FLBB_2K	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR) and one backscattering meter (at wavelength <nnn>) certified for applications down to 2000m	FLUOROMETER_CHLA, BACKSCATTERINGMETER_BBP<nnn>
ECO_FLNTU	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR) and one backscattering meter measuring turbidity	FLUOROMETER_CHLA, BACKSCATTERINGMETER_TURBIDITY

ECO_BB2	Wetlabs Eco optical sensor packages with two backscatteringmeters (at wavelength <nnn>)	BACKSCATTERINGMETER_BBP<nnn>
ECO_FLBBCD	Wetlabs Eco optical sensor packages with two fluorometers (type specified in SENSOR) and one backscatteringmeter (at wavelength <nnn>)	FLUOROMETER_CHLA, FLUOROMETER_CDOM, BACKSCATTERINGMETER_BBP<nnn>
ECO_FLBB2	Wetlabs Eco optical sensor packages with one fluorometer (type specified in SENSOR) and two backscatteringmeters (at wavelength <nnn>)	FLUOROMETER_CHLA, BACKSCATTERINGMETER_BBP<nnn>
ECO_BB3	Wetlabs Eco optical sensor packages with 3 backscatteringmeters (at wavelength <nnn>)	BACKSCATTERINGMETER_BBP<nnn>
MCOMS_FLBBCD	Wetlabs MCOMS optical sensor packages with two fluorometers CHLA, CDOM and one backscatteringmeter (at wavelength <nnn>)	FLUOROMETER_CHLA, FLUOROMETER_CDOM, BACKSCATTERINGMETER_BBP<nnn>
MCOMS_FLBB2	Wetlabs MCOMS optical sensor packages with one fluorometer CHLA and two backscatteringmeters (wavelengths specified in SENSOR)	FLUOROMETER_CHLA, BACKSCATTERINGMETER_BBP<nnn>
CYCLOPS_7_FLUOROMETER	Turner designs Optical sensor	FLUOROMETER_CHLA, BACKSCATTERINGMETER_TURBIDITY
SEAPOINT_TURBIDITY_METER	Seapoint turbidity meter	BACKSCATTERINGMETER_TURBIDITY
Other sensors		
RAFOS	Receiver mounted on some floats for geopositioning under ice using RAFOS sound sources in the array in the Weddell Sea (Location derivation is done in Delayed Mode for under ice floats equipped with RAFOS receivers).	ACOUSTIC_GEOLOCATION
PAL_UW	UW Passive acoustic listener	ACOUSTIC

EM	Electromagnetic sensor package to measure velocity	EM
FLOATCLOCK		FLOATCLOCK_MTIME

(*) Note that some biogeochemical sensors have different configurations, i.e. they are either in the pumped stream or not in the pumped stream. Sensor readings from those in the pumped vs unpumped stream can be very different. Some manufacturers do not distinguish this in the sensor model name.

The following sensor models are specific to EGO project.

SENSOR_MODEL	Comment	Associated SENSOR
Pressure sensors		
SBE_GPCTD	Glider Payload CTD	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_1111S	Glider Payload CTD, 20 dbar pressure sensor, fast pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_1200S	Glider Payload CTD, 100 dbar pressure sensor, standard pump speed, not compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_1201S	Glider Payload CTD, 110 dbar pressure sensor, standard pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_1300S	Glider Payload CTD, 350 dbar pressure sensor, standard pump speed, not compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_1301S	Glider Payload CTD, 350 dbar pressure sensor, standard pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_2411S	Glider Payload CTD, 600 dbar pressure sensor, fast pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_2500S	Glider Payload CTD, 1000 dbar pressure sensor, standard pump speed, not compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_2501S	Glider Payload CTD, 1000 dbar pressure sensor, standard pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_2600S	Glider Payload CTD, 2000 dbar pressure sensor, standard pump speed, not compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC
SBE_GPCTD_2601S	Glider Payload CTD, 2000 dbar pressure sensor, standard pump speed, compatible with an optional DO sensor.	CTD_PRES, CTD_TEMP, CTD_CNDC

SBE_CTD_SAIL	Free-flushed CTD, by Sea-Bird Electronics. It was the first science payload installed in the Seaglider.	CTD_PRES, CTD_TEMP, CTD_CNDC
Conductivity/temperature sensors		
RBR_LEGATO3	RBR model	CTD_PRES, CTD_TEMP, CTD_CNDC
Oxygen sensors		
AANDERAA_OPTODE_5013	Similar to AANDERAA_OPTODE_3835	OPTODE_DOXY
AANDERAA_OPTODE_5014	Similar to AANDERAA_OPTODE_3835	OPTODE_DOXY
AANDERAA_OPTODE_5015	Similar to AANDERAA_OPTODE_3835	OPTODE_DOXY
RBR_CODA_T_ODO	RBR model	OPTODE_DOXY
Fluorometer		
SEAOWL_UV_A		FLUOROMETER_CDOM
Fluorometers combination		
ECO_URRH		FLUOROMETER_URANINE, FLUOROMETER_RHODAMINE

5 Using the History section of the EGO NetCDF Structure

Within the NetCDF format are a number of fields that are used to track the progression of the data through the data system. This section records the processing stages, results of actions that may have altered the original values and information about QC tests performed and failed. The purpose of this section is to describe how to use this section of the format.

The next subsections provide examples of what is expected. The information shown in the column labeled "Sample" is what would be written into the associated "Field" name in the NetCDF format.

5.1 Recording information about the Delayed Mode QC process

The process of carrying out delayed mode QC may result in adjustments being made to observed variables. The table below shows how to record that the delayed mode QC has been done. Note that the fields HISTORY_SOFTWARE, HISTORY_SOFTWARE_RELEASE and HISTORY_REFERENCE are used together to document the name and version of software used to carry out the delayed QC, and the reference database used in the process. The contents of these three fields are defined locally by the person carrying out the QC.

Example: History entry to record that delayed mode QC has been carried out

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4.
HISTORY_STEP	ARSQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	WJO	This is a locally defined name for the delayed mode QC process employed.
HISTORY_SOFTWARE_RELEASE	1	This is a locally defined indicator that identifies what version of the QC software is being used.
HISTORY_REFERENCE	WOD2001	This is a locally defined name for the reference database used for the delayed mode QC process.
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	IP	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply (1)
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	FillValue	This field does not apply

Note

(1) The present version of delayed mode QC only tests salinity and as such it is tempting to place “PSAL” in the _PARAMETER field. In future, delayed mode QC tests may include tests for temperature, pressure and perhaps other parameters. For this reason, simply addressing the software and version number will tell users what parameters have been tested.

5.2 Recording processing stages

Each entry to record the processing stages has a similar form. An example is provided to show how this is done. Note that reference table 12 contains the present list of processing stages and there should be at least one entry for each of these through which the data have passed. If data pass through one of these steps more than once, an entry for each passage should be written and the variable N_HISTORY updated appropriately.

Some institutions may wish to record more details of what they do. In this case, adding additional “local” entries to table 12 is permissible as long as the meaning is documented and is readily available. These individual additions can be recommended to the wider community

for international adoption.

Example: History entry to record decoding of the data.

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4.
HISTORY_STEP	ARFM	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	IP	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_PRES	FillValue	This field does not apply
HISTORY_STOP_PRES	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	FillValue	This field does not apply

5.3 Recording QC Tests Performed and Failed

The delayed mode QC process is recorded separately from the other QC tests that are performed because of the unique nature of the process and the requirement to record other information about the reference database used. When other tests are performed, such as the automated real-time QC, a group of tests are applied all at once. In this case, instead of recording that each individual test was performed and whether or not the test was failed, it is possible to document all of this in two history records.

The first documents what suite of tests was performed, and the second documents which tests in the suite were failed. A test is failed if the value is considered to be something other than good (i.e. the resulting QC flag is set to anything other than "1"). An example of each is provided. If data pass through QC more than once, an entry for each passage should be written and the variable N_HISTORY updated appropriately.

Example: QC tests performed and failed.

The example shown here records that the data have passed through real-time QC and that two tests failed. The encoding of tests performed is done by adding the ID numbers provided in reference table 11 for all tests performed, then translating this to a hexadecimal number and recording this result.

Record 1: Documenting the tests performed

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4.
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	QCP$	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	1BE	This is the result of all tests with IDs from 2 to 256 having been applied (see reference table 11)

Record 2: Documenting the tests that failed

Field	Sample	Explanation
HISTORY_INSTITUTION	ME	Selected from the list in reference table 4.
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	QCF$	Selected from the list in reference table 7
HISTORY_PARAMETER	FillValue	This field does not apply
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	FillValue	This field does not apply
HISTORY_QCTEST	A0	This is the result when data fail tests with IDs of 32 and 128 (see reference table 11)

5.4 Recording changes in values

The PIs have the final word on the content of the data files in the EGO data system. In comparing their data to others there may arise occasions when changes may be required in the data.

We will use the example of recomputation of where the glider first surfaced as an example. This computation process can be carried out once all of the messages from a glider have been received. Not all real-time processing centers make this computation, but it can be made later on and added to the delayed mode data. If this is the case, we would insert the new position into the latitude and longitude fields and we would record the previous values in two history entries. Recording these allows us to return to the original value if we have made an error in the newly computed position. The two history entries would look as follows.

Example: Changed latitude

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4.
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	CV	Selected from the list in reference table 7
HISTORY_PARAMETER	LAT$	A new entry for reference table 3 created by institution CI to indicate changes have been made in the latitude.
HISTORY_START_TIME	FillValue	This field does not apply
HISTORY_STOP_TIME	FillValue	This field does not apply
HISTORY_PREVIOUS_VALUE	23.456	This is the value of the latitude before the change was made.
HISTORY_QCTEST	FillValue	This field does not apply

Notes

1. Be sure that the new value is recorded in the latitude and longitude of the trajectory.
2. Be sure that the POSITION_QC flag is set to "5" to indicate to a user that the value now in the position has been changed from the original one that was there.
3. Be sure to record the previous value in history entries.

It is also sometimes desirable to record changes in quality flags that may arise from reprocessing data through some QC procedures. In this example, assume that whereas prior to the analysis, all temperature values from 75 to 105 dbars were considered correct, after the

analysis, they are considered wrong. The history entry to record this would look as follows.

Example: Changed flags

Field	Sample	Explanation
HISTORY_INSTITUTION	CI	Selected from the list in reference table 4.
HISTORY_STEP	ARGQ	Selected from the list in reference table 12.
HISTORY_SOFTWARE	FillValue	This field does not apply
HISTORY_SOFTWARE_RELEASE	FillValue	This field does not apply
HISTORY_REFERENCE	FillValue	This field does not apply
HISTORY_DATE	20030805000000	The year, month, day, hour, minute, second that the process ran
HISTORY_ACTION	CF	Selected from the list in reference table 7
HISTORY_PARAMETER	TEMP	Selected from the list in reference table 3
HISTORY_START_PRES	75	Shallowest pressure of action.
HISTORY_STOP_PRES	105	Deepest pressure of action.
HISTORY_PREVIOUS_VALUE	1	This is the value of the quality flag on temperature readings before the change was made.
HISTORY_QCTEST	FillValue	This field does not apply

Notes

1. The new QC flag of “4” (to indicate wrong values) would appear in the <param>_QC field.

6 GDAC files distribution organization

There are two GDACs (global data assembly centers) for redundancy, which are the users' access points for EGO data. One GDAC is located in France (Coriolis, http://www.coriolis.eu.org). The GDACs handle EGO data, metadata, and index files on ftp servers. The servers at both GDACs are synchronized at least daily to provide the same EGO data.

The user can access the data at either GDAC's ftp site:

- ftp://ftp.ifremer.fr/ifremer/glider/v2/

From these root directories of the GDACs downward, the organization of the directories and files is:

- glider/FileName.nc
  site: EGO site code

The sites codes will be listed in the "EGO catalogue" document at either GDAC's root directory.

6.1 EGO file naming convention

The EGO file names use the following naming convention for data and metadata files.

YYY/YYY_XXX/YYY_ZZZ_T.nc

- YYY: platform code from the EGO catalogue
- XXX: deployment start day YYYYMMDD
- ZZZ: deployment code
- T: data Mode
  - R: real-time data
  - P: provisional data
  - D: delayed mode
  - M: mixed delayed mode and real-time.
- .nc: NetCDF file suffix

Example

- milou/milou_20150112/milou_mooseperseust02_08_R.nc

This file contains observations and metadata from the Milou glider, from the deployment performed in January 2015.

6.2 EGO profile file naming convention

The EGO profile file names use the following naming convention.

TVVV_XXX_NNN[D].nc

- T: data Mode
    - R: real-time data
    - P: provisional data
    - D: delayed mode
    - M: mixed delayed mode and real-time.
- VVV: glider Id. It could be WMO number (wmo_platform_code) if any, platform_code otherwise.
- XXX: deployment start day YYYYMMDD
- NNN: profile number (chronological number of the Yos that have produced one descent or/and ascent profile).
- D: final D indicates a descending profile (profile without this D are collected during ascent).
- .nc : NetCDF file suffix

Example

- milou/milou_20150112/profiles/R68951_20150113_032.nc

This file contains profile data, collected during the ascent of the 32th "useful" Yo of the Milou glider during the deployment performed in January 2015.

6.3 Index of glider deployments files

To allow for data discovery without downloading the data files themselves, an index file is created at the GDAC level, which lists all available data files and the location and time ranges of their data contents:

- The data index file is located at the root directory of the GDAC.
- The index file contains the list and a description of all data files available on the GDAC.
- There is a header section, lines of which start with # characters.
- The information sections are comma-separated values.
- Each line contains the following information:
    - file: the file name, beginning from the GDAC root directory
    - date_update: the update date of the file, YYYY-MM-DDTHH:MI:SSZ
    - start_date: first date for observations, YYYY-MM-DDTHH:MI:SSZ
    - end_date: last date for observations, YYYY-MM-DDTHH:MI:SSZ
    - southern_most_latitude, decimal degrees
    - northern_most_latitude, decimal degrees

- western_most_longitude, decimal degrees
- eastern_most_longitude, decimal degrees
- geospatial_vertical_min, decibar
- geospatial_vertical_min, decibar
- update_interval: M monthly, D daily, Y yearly, V void
- size: the size of the file in bytes
- gdac_creation_date: date of creation of the file on the GDAC, YYYY-MM-DDTHH:MI:SSZ
- gdac_update_date: date of update of the file on the GDAC, YYYY-MM-DDTHH:MI:SSZ
- data_mode: R, P, D, M (real-time, provisional, delayed mode, mixed; see reference table 19)
- parameters: list of parameters (standard_name) available in the file separated with blank

The fill value is empty: "".

GDAC data files index: EGO_files_index.txt

EGO FTP GLOBAL INDEX
FTP://FTP.IFREMER.FR/IFREMER/EGO
Contact: HTTP://WWW.EGO.ORG
Index update date YYYY-MM-DDTHH:MI:SSZ: 2008-03-30T18:37:46Z
#
#file,date_update,start_date,end_date,
southern_most_latitude,northern_most_latitude,western_most_longitude,eastern_most_longitude,
geospatial_vertical_min,geospatial_vertical_min,update_interval,size,gdac_creation_date,gdac_update_date,data_mode,
parameters
PYTHEAS/GL_PYTHEAS_201006_R_LATEX.nc,2008-04-12T08:05:00Z,2007-03-17T18:07:00Z,2008-04-
12T08:05:00Z,0,0,-170,-170,16.7,0,550,M,14178,2008-04-12T08:05:00Z,2008-04-12T08:05:00Z,R,sea_water_pressure
sea_water_temperature sea_water_salinity

7 Data distribution from DAC

7.1 DAC to GDAC data distribution

The Data Assembly Centers (DAC) collect data from glider operators (real-time) or from scientists (delayed mode data).

In real-time, each DAC converts glider data into EGO-NetCDF files. It applies the real-time quality controls on the NetCDF files.

The DACs push these quality controlled data files to the Global Data Assembly Centers (GDACs).

The role of the GDAC is to distribute the best versions of EGO NetCDF files.

7.2 DAC to GTS data distribution

The EGO glider data received in real-time are quality-controlled. The real-time quality control procedures are described in the EGO glider quality control manual. They are automatically applied, without human intervention to minimize the delay between data observation and data distribution.

For each active glider, the data that passed the real-time QC tests are distributed on GTS (the WMO data transmission system). Data distributed on GTS should be less than 30 days old. The target for distribution is within 48 hours of the observation time.

TESAC format distribution

The vertical profiles extracted from the glider time-series are distributed as TESAC messages.

Each vertical profile should have a vertical length greater or equal to 40 decibars.

Buoy format distribution

The glider time-series are distributed as BUOY format messages.

BUFR format distribution

In a near future, the glider time-series will be distributed in BUFR format. The glider BUFR template is under construction.

8 Glossary, definitions

This chapter gives a definition for the EGO items described in this manual.

8.1 Observatory

An observatory is a facility that manages a series of gliders.

8.2 Deployment

The deployment is the period between the launch and recovery or loss of a glider.

8.3 Glider

A steered and autonomous platform deployed in the sea that performs environmental monitoring.

8.4 Sensor

A device that measures environmental parameter but does not digitize data for transmission, it needs to be connected to an instrument to produce a data stream that a computer can read. Examples: Transmissiometer, Fluorometer, Oxygen sensor.

8.5 Parameter measured by the sensor

What was measured.

8.6 Calibration of the parameter measured by the sensor

Verification of Any operation measurement against independent measurements to derive a corrected value or a new parameter.

8.7 Principal Investigator (PI)

The **Principal Investigator (PI)**, typically a scientist at a research institution, maintains the observing platform and the sensors that deliver the data. He or she is responsible for providing the data and all auxiliary information to a **Data Assembly Center (DAC)**.

8.8 Global Data Assembly Center (GDAC)

The **GDAC** distributes the best copy of the data files. When a higher quality data file (e.g. calibrated data) is available, it replaces the previous version of the data file.
The user can access the data at either GDAC, cf. section “GDAC organization”.

8.9 Data Assembly Center (DAC)

The **DAC** assembles EGO-compliant files from this information and delivers these to the two **Global Data Assembly Centers (GDACs)**, where they are made publicly available.