argo data management

Argo data management
DOI: http://dx.doi.org/10.13155/41151

Argo DAC Profile Cookbook

Version 2
August 2019

ARGO
part of the integrated global observation strategy

Argo data management

Argo DAC profile cookbook

Authors: Megan Scanderbeg / Scripps Institution of Oceanography, Jean-Philippe Rannou / ALTRAN, Annie Wong /University of Washington , Birgit Klein, BSH, Claudia Schmid / AOML, Brian King / National Oceanography Centre, Esmee Van Wijk / CSIRO

How to cite this document

Megan Scanderbeg / Scripps Institution of Oceanography, Jean-Philippe Rannou / ALTRAN, Annie Wong / University of Washington, Birgit Klein, BSH, Claudia Schmid / AOML, Brian King / National Oceanography Centre, Esmee Van Wijk / CSIRO , **Argo DAC profile cookbook,** http://dx.doi.org/10.13155/41151.

Table of contents

History of the document

Version	Date	Comment
1.0	June 2015	Original version sent around for comment
2.0	August 2019	Added information about optional POSITION_ERROR and POSITION_COMMENT variables to be used with estimated positions Deleted section on how to reduce high density profiles for TESAC format on GTS since only send BUFR format as of July 2018

Preface

This document is still in progress. As such, there are highlighted sections of text throughout that need to be addressed. Yellow highlighting means this is a topic open to discussion - some things are known about this topic, but agreement needs to be reached. Green highlighting signifies a question that needs to be answered by a float expert or float manufacturer. Red highlighting means the issue needs a solution and nothing has been suggested yet.

1 Introduction

This DAC cookbook is to include instructions for the DACs on how to calculate different variables for the Argo profile files. This is especially true for some of the additional sensors that are being added to Argo floats. The extra variables increase the complexity of the profile file and extra instructions may be necessary, depending on float and sensor type. It is separate from other data manuals because users do not need to understand all these details, but that it is important that all DACs to be calculating the variables in the same manner.

There are instructions in this cookbook that apply to all floats and some that apply to only certain float types. Look through the table of contents for the specific float types included.

2 Profile files

2.1 Profile time and location

2.1.1 Profile time

The profile time determination depends on the float positiong system

For Argos floats:

The date of an ascending profile (JULD variable) should be set (in order of priority):

1. To the Ascent End Time (AET) of the current cycle,
2. To the Transmission Start Time (TST) of the current cycle,
3. To the First Message Time (FMT) of the current cycle.

The date of a descending profile (JULD variable) should be set (in order of priority):

1. To the JULD of descending profile (DST of current cycle) if the float reports this,
2. To the Transmission End Time (TET) of the previous cycle,
3. To the Last Message Time (LMT) of the previous cycle,
4. otherwise:
   a. If the previous cycle is missing: *Use (DST or TET or LMT of the current cycle) - CYCLE_TIME*
   b. If the current cycle is the first cycle, use:
      i. The LMT of the previous surface drift prior to the first dive, if any,
      ii. Otherwise, the launch time of the float.

See the DAC Trajectory Cookbook to find how to compute AET, TST, FMT, DST, TET and LMT times for each float type and version.

For Iridium floats:

The date of an ascending profile (JULD variable) should be set (in order of priority):

1. To the Ascent End Time (AET) of the current cycle,
2. To the time of the first GPS fix
3. To the time of the second GPS fix (if there is one)
4. To the Transmission Start Time (TST) of the current cycle,
5. To the First Message Time (FMT) of the current cycle.

The date of a descending profile (JULD variable) should be set (in order of priority):

1. To the JULD of descending profile (DST of current cycle) if the float reports this,
2. To the Transmission End Time (TET) of the previous cycle,
3. To the Last Message Time (LMT) of the previous cycle,
4. otherwise:
    c. If the previous cycle is missing: *Use (DST or TET or LMT of the current cycle) - CYCLE_TIME*
    d. If the current cycle is the first cycle, use:
        i. The LMT of the previous surface drift prior to the first dive, if any,
        ii. Otherwise, the launch time of the float.

See the DAC Trajectory Cookbook to find how to compute AET, TST, FMT, DST, TET and LMT times for each float type and version.

2.1.2 Profile Location

The location and location time (LATITUDE, LONGITUDE and JULD_LOCATION variables) determination depend on the float positioning system.

For Argos floats

The location and location time of a descending profile should be set to the last Argos fix of the previous cycle surface trajectory that succeeds the real time qc test #20 "Questionable Argos position test" (see ANNEX C: Implementation of the JAMSTEC trajectory quality control method)

If the previous cycle is missing, interpolation can be done in near-real-time or in delayed mode. It is set to fill value in real time. How many cycles with interpolated positions may exist between 2 cycles with a position?

If the current cycle is the first deep cycle: use the surface drift trajectory done by the float prior to its first dive, if any; otherwise use the float launch time and position.

For Iridium floats with available GPS fixes

The location and location time of an ascending profile should be set to the first GPS fix of the current cycle surface trajectory.

The location and location time of a descending profile should be set to the last GPS fix of the previous cycle surface trajectory.

If the previous cycle is missing, interpolation can be done in near-real-time or in delayed mode. It is set to fill value in real time. How many cycles with interpolated positions may exist between 2 cycles with a position?

If the current cycle is the first cycle: use the surface drift trajectory done by the float prior to its first dive, if any; otherwise use the float launch time and position. Typically, Iridium floats do not have a record of positions prior to the first dive.

For Iridium floats when no GPS fix is available

When no GPS fix is available, a weighted average of all Iridium fixes can be used. If the average CEPradius is more than 5km, a POSITION_QC flag of 2 should be assigned (see ANNEX B: How to calculate a profile position). In this case, the POSITIONING_SYSTEM variable should be set to 'IRIDIUM' instead of 'GPS'. If there are too few Iridium fixes available or they are of very poor quality, an interpolated position can be used. If this estimated position is based on two or more positioning systems (ie, a GPS fix on one side and an Iridium fix on the other), or if the estimation method does not rely on information from a positioning system, POSITIONING_SYSTEM should be set to 'NONE'.

Estimated positions for any transmission system

If a float has no positions or times at all for a surface interval, then an estimated position with a POSITION_QC flag of 8 is appropriate. This can happen often for ice floats or other floats that are unsuccessful at reaching a satellite while on the surface. Sometimes floats in sea-ice zones spend months under water without surfacing and upon surfacing send back many profiles without surface times or positions, but often with ascent end times. Currently, the usual method to estimate the position is to interpolate between known fixes from previous and later cycles. However, additional estimation methods may also be used, especially in delayed mode. To help give users more information on these estimated positions in the profile file, **two optional variables** can be added to the profile file: POSITION_ERROR and POSITION_COMMENT.

If DACs choose to add POSITION_ERROR, it should be filled with an uncertainty error in meters. If a DAC estimates the position using a method other than interpolation, it is highly suggested that the POSITION_COMMENT field be filled. When, after some cycles with no GPS fixes, we succeed in getting GPS fixes, should we replace positions computed from Iridium fixes with interpolated positions? If yes, this should depends on the number of interpolated cycles between 2 cycles with GPS fixes. At Coriolis we replace positions from averaged iridium fixes with GPS interpolated positions when the interpolated interval is less than or equal to 30 days.

2.2 Pumped and Unpumped CTD data collected by NKE floats

The management of pumped and unpumped CTD data is common to all NKE float versions but the reported information depends on instrument transmission type (Argos or Iridium). Moreover, with the new ProvBioII float configuration capabilities, the user can define the treatment type of the unpumped data.

In this chapter we explain how to separate pumped and unpumped (or partially pumped) data from a sampled CTD profile. This separation is mandatory to correctly:

- Flag the unpumped salinities (see "Near-surface unpumped CTD salinity test" in the Argo Quality Control Manual),
- Create a primary sampling profile (with the pumped CTD data) and a near-surface sampling profile (with the unpumped CTD data).

2.2.1 Needed information

In the algorithm provided bellow, one can determine for each received profile what should be known to separate pumped and unpumped CTD data for a NKE float. Depending on the available float reported information, the following items can be needed:

- For a ProvBioII float, to know the CTD sensor configuration for the current profile,
- For an Iridium float, to collect the 'subsurface point' measurement of the current profile,
- To know the CTD pump switch off pressure configuration parameter (Pcutoff),
- To know the bin size configuration parameter of the depth zone associated to Pcutoff,
- To know the software version of the float.

2.3.1.1 The treatment type of the CTD profile for the cycle

Each sensor mounted on a ProvBioII float can be configured for each of the 5 depth zones that can be defined.

One available configuration parameter is the treatment type applied to the data before transmission. This treatment type can be: 0: raw data are transmitted, 1: bin-averaged data are transmitted and 7: bin averaged data and associated median and standard deviation are transmitted.

2.2.1.2 The 'subsurface point' measurement pressure value of the profile

NKE Iridium floats provide, in the technical message, the raw value of the last (shallowest) pumped CTD (or CTDO) sample. This measurement is called 'subsurface point' in the float manuals.

A specific bin is created after the pressure of the 'subsurface point' ($P_{SubsurfacePoint}$).so that pumped and unpumped data are not mixed in the bin-averaged output values.

2.2.1.3 The CTD pump switch off pressure configuration parameter

The CTD pump switch off pressure (Pcutoff) is a technical configuration parameter of the NKE floats. The default value of PcutOff is 5 dbar, thus if the real configuration is unknown, use this default value.

2.2.1.4 The bin size configuration parameters

The bin size of each defined depth zone is a mission configuration parameter of NKE floats.

Three depth zones can be defined (2 for older float versions and 5 for the ProvBioII float), we are interested in the bin size (bin_size) associated to the depth zone of Pcutoff.

2.2.1.5 The software version of the float

The ‘old’ version numbers are versions < 5816A00 for PROVOR floats and < 5605A00 for ARVOR floats. This information needs to be known in case of bin-averaged data.

2.2.2 Possible algorithm

To separate pumped and unpumped (or partially pumped) measurement of a received CTD profile, the algorithm can be the following.

```
If $P_{SubsurfacePoint}$ is available for the current profile

    If it is a ProvBioII float

        If the treatment type of the depth zone associated to $P_{SubsurfacePoint}$ is ‘raw’
            All data measurements, collected after the **first** crossing of $P_{SubsurfacePoint}$ are
            unpumped
        Else
            Data are unpumped when $PRES < P_{SubsurfacePoint}$
        End

    Else
        Data are unpumped when $PRES < P_{SubsurfacePoint}$
    End

Else

    If it is a ProvBioII float

        If the treatment type of the depth zone associated to Pcutoff is ‘raw’
            All data measurements, collected after the first crossing of (Pcutoff + 0 dbar
            (see note [1])) are unpumped.
        Else
            Data are unpumped when $PRES \leq Pcutoff + 0.5$
        End

    Else if it the data are spot-sampled

        Data are unpumped when $PRES \leq Pcutoff$

    Else if the data are bin-averaged

        If it is an ‘old’ version software
            Data are unpumped when $PRES \leq Pcutoff + bin\_size/2$
        Else
            Data are unpumped when $PRES \leq Pcutoff + 0.5$
        End

    End

End
```

[1] In that case the float transmits the SBE CTD output. We set Pcutoff + 0 dbar assuming that there is no delay for a SBE CTD configured to switch off its pump at Pcutoff dbars (to be confirmed by SBE).

2.2.3 Information storage

The cut-off pressure of the CTD pump is stored in the meta.nc file with the configuration name CONFIG_CTDPumpStopPressure_dbar.

The cut-off pressure of the ascent profile is stored in the meta.nc file with the configuration name CONFIG_CTDPumpStopPressurePlusThreshold_dbar.

The P, T and S measurements of the ‘subsurface point’ are stored in the tech.nc file with the names PRES_LastAscentPumpedRawSample_dbar, TEMP_LastAscentPumpedRawSample_degC and PSAL_LastAscentPumpedRawSample_psu respectively.

The PTS(O) measurements of the ‘subsurface point’ are stored in the traj.nc file with the MEASUREMENT_CODE = 599.

2.3 Guidelines for Argos message selection

2.3.1 Argos float message selection

Ideally, every DAC should use the same method for Argos message selection for each float type. Some floats are transmitting a CRC (Cyclic Redundancy Check) done on board the float and others are not. Additionally, not all CRC have the same reliability. Recommendations were issued at ADMT 10, but inconsistencies still exist between DACs.

Message selection strategy for each float type will be listed below:

Argos message selection done at Coriolis for PROVOR/ARVOR

Technical message selection

1. If only one technical message is received with a good CRC, use it,
2. If more than one technical message is received, all with good CRCs, use the"first received one"
3. If no technical message is received with a good CRC, no technical message is used. In this case, times provided by the float are missing and, consequently, the order of the drift CTD measurements cannot be determined.

CTD data message selection

Received messages are processed by type (type 4: "descent profile CTD message", type 5: "submerged drift CTD message" and type 6: "ascent profile CTD message").

For each type, the Id of the received message is computed.

- For type 4 or type 6 messages, the Id is defined by the date and the pressure of the first CTD measurement of the message,
- For type 5 messages, the Id is defined by the date and the time of the first CTD measurement of the message.

The selection process must lead to (at most) one message for a given Id.

For a given type, all messages of a given Id are processed:

1. If only one message is received with a good CRC, use it,
2. If more than one message is received all with good CRCs, use the "first received one",

3. If no message is received with a good CRC:
   a. If 1 or 2 copies of the message has been received, no message is used for this Id,
   b. If more than 2 copies of the message have been received:
      i. If an even number of copies of the message have been received, reject the "first received one",
      ii. The possibly emitted message is computed from received copies (each bit of the message is defined by selecting the "most redundant" received one),
      iii. A CRC check is done on this "reconstructed" message:
         1. If it succeeds, use this "reconstructed" message,
         2. If it fails, no message is used for this Id.

3 ANNEX A: Some definitions

Here are some definitions about elements mentioned in this document, if some of them remain unclear, please ask for a new or updated definition (argo@ucsd.edu, support@argo.net).

3.1 Definitions of Argos raw data contents

The following definitions can be found in the Argos User's manual (http://www.argos-system.org/manual/).

Let us consider Argos raw data provided in a PRV/DS command output format.

Argos float messages are collected by a given satellite during a satellite pass. A header of the satellite pass (in underlined bold in the two following examples) is added to the data by CLS.

In this header, one can find:

- The number of lines of data relative to the satellite pass header (including the header line),
- The name of the satellite,
- If a location has been computed from the data collected during the satellite pass (example 2):
    - The location class of the location,
    - The date of the location,
    - The latitude and longitude of the location.

The Argos float messages, collected during the satellite pass, follow the header.

For each we find:

- The Argos message date (time of reception of the message by the satellite),
- The Argos message redundancy,
- The Argos message content.

Example 1: A satellite pass without Argos location.

```
02412 63706 17 31 L
2007-04-24 02:40:16 2 64 A2 56 BA
B2 3C 8D 7C
AF 9F 85 AD
72 ED D5 4E
65 09 F7 5D
5C 1E B9 52
D0 CE AA 61
9A 30 00
2007-04-24 02:40:58 1 67 09 D5 CB
5F 31 75 7C
23 8D 3D 82
73 AA 30 8E
5C 46 A1 C7
68 D0 F9 91
D9 60 B9 7E
EB 38 00
```

Example 2: A satellite pass with Argos location.

```
02412 63706 33 31 D 2 2007-04-24 05:30:15 -32.189 11.405 0.000 401651871
2007-04-24 05:27:35 1 51 C9 1B A6
```

```
F4 0B 5B 5F
2E 83 F4 7F
DF E0 E4 06
1F 99 80 1E
94 6A 80 FD
FE 10 39 1E
A7 F4 00
2007-04-24 05:30:15 1 05 08 16 92
0F 83 AE 18
40 20 90 0A
20 00 19 9E
04 15 A6 00
0C 39 05 85
89 01 8E 04
C9 68 20
2007-04-24 05:30:52 1 58 A3 7D 66
F4 8B 16 DF
33 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00 00
00 00 00
2007-04-24 05:32:55 1 69 F5 58 B9
28 72 27 88
72 A5 59 91
54 B5 B4 2A
95 02 43 52
A8 49 2A 5C
E9 DD 4C F7
62 00 00
```

3.2 Cyclic Redundancy Check

All Argos floats (except SIO SOLO) have an error detection code embedded in their Argos messages. Checking this code, called Cyclic Redundancy Check (CRC), can theoretically enhance the reliability of the data by rejecting messages possibly corrupted by transmission error.

3.3 Float clock drift and clock offset

Some Argo float versions provide times for dated events or dated measurements. Over time, the float's clock may drift. Clock drift can be defined as the drift of the clock in hours/ minutes/ seconds per year. To correct for this, we must apply a clock offset where clock offset is defined as a measurement, done at a given time, of the offset of the clock due to clock drift. Thus a clock offset should be estimated for each of these float times.

Note that, in this document, float clock offset can also embrace a clock that has not been correctly set or a clock that has been set in local time. Of course, in these cases clock offset is not only revealing a drift of the float clock...

Float clock offset is defined as: Float clock offset = Float time - UTC time.

A good estimate of the clock offset can be obtained when the float transmits its Real Time Clock (RTC) time in the technical data. It can then be compared to the time from Argos of the corresponding message to compute a clock offset for all the float times of the concerned cycle.

Unfortunately this is not always the case, some floats do not transmit their RTC time and even if they do, this RTC time is not always received.

3.4 APEX Argos test/data messages

APEX Argos test messages are transmitted by an APEX float during the six hours period spent at the surface prior to its first dive. The test message contains programmed mission parameters and technical data.

After each deep cycle, APEX floats transmit the collected data in the ARGOS data messages.

3.5 APEX Deep Profile First floats

Some APEX floats are programmed to achieve their first profile shortly after deployment (for comparison to conventional CTD cast from the ship).

In the Deep Profile First (DPF) cycle the firmware is set to complete the first profile within 24 hours of deployment. The float descends to park, parks then descends to profiling depth. Then the profile commences. The duration of park is 5 minutes for Iridium floats and 60 minutes for ARGOS floats. The DPF also ignores any time of day setting. It is unclear how long the two descents are. Experience has shown that both the descent time out configuration settings are activated when the float fails to reach target depths.

3.6 APEX Time Of Day feature

Some APEX floats have the capability to schedule profiles so that the float surfaces at a particular Time Of Day (TOD).

When the TOD feature is enabled, the float RTC is used to dynamically set the end of the DOWN TIME period to a (user programmed) number of minutes after midnight.

The time of day feature is ignored by Deep Profile First floats.

3.7 APEX Auxiliary Engineering Data

For some APEX floats, the remaining space of the last Argos data message is filled with Auxiliary Engineering Data (AED).

The AED are considered to be of lower priority and will never cause an additional Argos data message to be generated.

4 ANNEX B: How to calculate a profile position from a set of Iridium fixes

When a GPS fix is not available, a weighted average of all Iridium fixes should be used.

$$\text{Lat} = \frac{\sum_{i=1}^{N} w_i y_i}{\sum_{i=1}^{N} w_i}$$

$$\text{Lon} = \frac{\sum_{i=1}^{N} w_i x_i}{\sum_{i=1}^{N} w_i}$$

$$w_i = \frac{1}{R_i^2}$$

Lat: Calculated latitude

Lon: Calculated longitude

wi: weight

yi: Latitude of Iridium fix in a certain cycle

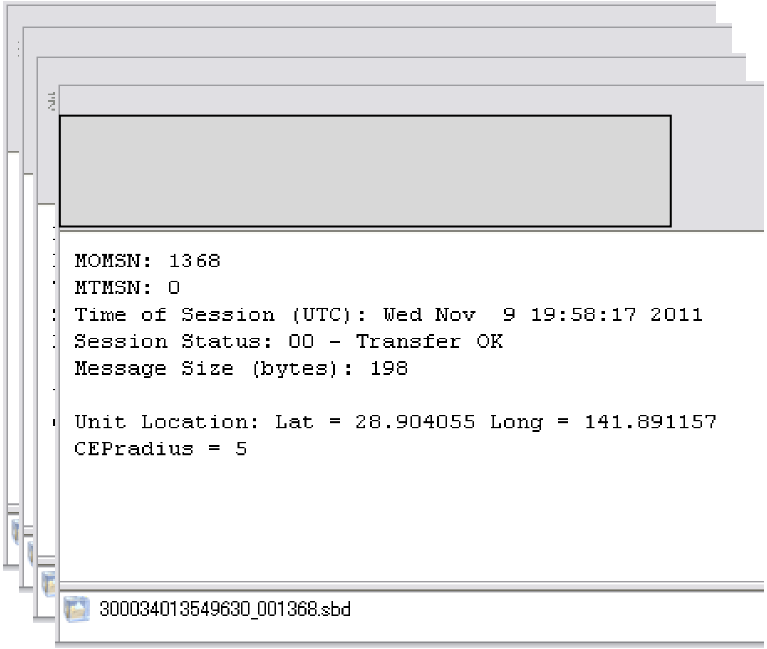
xi: Longitude of Iridium fix in a certain cycle

Ri: CEPradius in a certain cycle

N: number of all Iridium fixes in a certain cycle

4.1 JULD_LOCATION

JULD_LOCATION = minimum "Time of Session"



```
MOMSN: 1368
MTMSN: 0
Time of Session (UTC): Wed Nov  9 19:58:17 2011
Session Status: 00 - Transfer OK
Message Size (bytes): 198

Unit Location: Lat = 28.904055 Long = 141.891157
CEPradius = 5
```

4.2 POSITION_QC

When the average CEPradius (which is not a weight averaged CEPradius) is less than 5km,

POSITION_QC=1 (good data).

When the average CEPradius is more than 5 km,

POSITION_QC=2 (probably good data).

5 ANNEX C: Implementation of the JAMSTEC trajectory quality control method

The JAMSTEC trajectory quality control method is described in Nakamura et al (2008), "Quality control method of Argo float position data", JAMSTEC Report of Research and Development, Vol. 7, 11-18 (http://www.godac.jamstec.go.jp/catalog/data/doc_catalog/media/JAM_RandD07_02.pdf).

This method checks the surface trajectory of an Argos float by considering the speeds induced by the successive Argos fixes. The test can flag the surface position as '3' or '4'.

In the following, we propose a detailed description of the algorithm to implement.

5.1 Inputs

The inputs of the algorithm are:

- The surface trajectory to be checked (N Argos location dates, latitudes, longitudes and classes),
- The last good (flagged as '1') surface location of the (already checked) previous (received) cycle.

5.2 Algorithm

Assuming that the location dates have not been flagged as bad by the test #2 "Impossible date test", we first chronologically sort the surface positions.

The whole surface trajectory is used to initialize the (checked) current trajectory.

The current trajectory is processed in an infinite loop in which the following steps are performed:

5.2.1 Step 1

The subsurface drift speed is computed between the last good surface position of the previous cycle and the first position of the current trajectory.

If this speed is greater than 3 m/s, the first position of the current trajectory is flagged as '4', this position is then excluded from the current trajectory and a new iteration of the infinite loop starts.

5.2.2 Step 2

Speeds are computed for the second position to the last position of the current trajectory. Each speed is computed between position #i-1 and position #i and affected to position #i.

In case of duplicated positions (i.e. if position #i-1 and position #i have the same latitude, longitude and date): the position #i is flagged as '4', it is then excluded from the current trajectory and a new iteration of the infinite loop starts.

In case of an erroneous cycle number of the position #i (i.e. if the times difference between position #i and position #i-1 is greater than one day): the position #i is flagged as '4', it is then excluded from the current trajectory and a new iteration of the infinite loop starts.

5.2.3 Step 3

The position #iMax is found as the position with the maximum speed.

If this maximum speed is greater than 3 m/s, the position #iMax is 'questionable' and the **speed test** (see §5.3) is performed on it over the current trajectory.

The **speed test** should lead to define position #iMax or/and position #iMax-1 as 'abnormal'.

5.2.4 Step 4

If the **distance test** (see §5.4) between position #iMax and position #iMax-1 is verified, the 'abnormal' position(s) is (are) flagged as '3'.

The 'abnormal' position(s) is(are) then deleted from the current trajectory (even when the distance test is not verified) and a new iteration of the infinite loop starts.

The infinite loop ends when no 'abnormal' position has been detected or when the current trajectory has less than 2 positions.

5.3 Speed test

The speed test is performed on a 'questionable' position over a given trajectory.

The 'questionable' position (called B in the following) can be all but the first position of the trajectory. The position which precedes B on the trajectory is called A in the following.

5.3.1 Case of different Argos classes

If positions A and B have different Argos classes, the position with the less accurate Argos class is defined as 'abnormal' by the speed test.

Remember that the accuracy of the Argos location classes is the following:

more accurate <= 3, 2, 1, 0, A, B, Z => less accurate

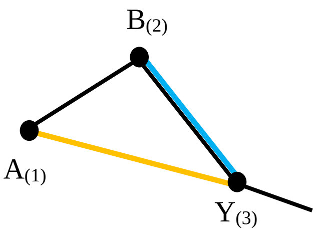
5.3.2 Case of identical Argos classes

If positions A and B have the same Argos classes:

- If the trajectory only comprises the two positions A and B, both positions are defined as 'abnormal' by the speed test,
- Otherwise the speed test depends on the position of the location B on the trajectory, 3 cases are possible.

Case 1: If B is the second position of the trajectory

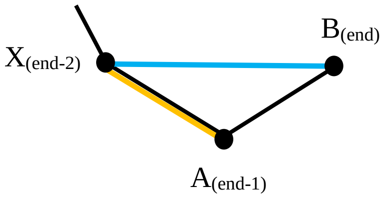
In this case: A is the first position, B the second one and there is a position Y following the position B on the trajectory.



Speeds on the segments A-Y (orange) and B-Y (blue) are computed: if speed$_{A\text{-}Y}$ is greater than speed$_{B\text{-}Y}$, the position A is defined as 'abnormal' by the speed test otherwise B is defined as 'abnormal' by the speed test.

Case 2: If B is the last position of the trajectory

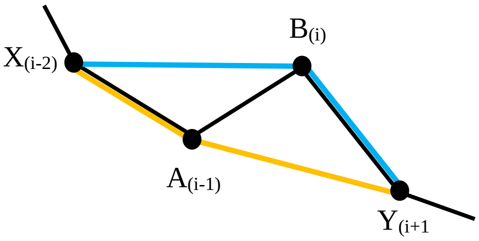
In this case: A is the last but one position, B is the last position and there is a position X preceding the position A on the trajectory.



Speeds on the segments X-A (orange) and X-B (blue) are computed: if speed$_{X\text{-}A}$ is greater than speed$_{X\text{-}B}$, the position A is defined as 'abnormal' by the speed test otherwise B is defined as 'abnormal' by the speed test.

Case 3: we are not in case 1 or 2

In this case: there is a position X preceding the position A on the trajectory and a position Y following the position B on the trajectory.



Speeds on the segments X-A-Y (orange trajectory) and X-B-Y (blue trajectory) are computed. If speed$_{X\text{-}A\text{-}Y}$ is greater than speed$_{X\text{-}B\text{-}Y}$, the position A is defined as 'abnormal' by the speed test otherwise B is defined as 'abnormal' by the speed test.

5.4 Distance test

The distance test is performed on two Argos locations A and B.

The distance test is verified if the distance between locations A and B is greater or equal to

$1.0 \times \sqrt{Er_A{}^2 + Er_B{}^2}$ where Er_A and Er_B are the radii of position error for locations A and B respectively.

These position errors, deduced from the position classes, are 150 m, 350 m and 1000 m for Argos class 3, 2 and 1 respectively. Moreover we have associated a position error of 1500 m, 1501 m, 1502 m and 1503 m for Argos classes 0, A, B and Z respectively.

5.5 Distance computation

As far as distance and speed are concerned in this trajectory QC method, we must specify an algorithm to compute distance between positions of the surface trajectory. This algorithm must be common to all the DACs so that the trajectory QC results will not depend on DAC's distance computation method.

We propose to use the distance algorithm from the *Laboratoire de Physiques des Océans* (*LPO*) at IFREMER.

This algorithm computes distance between points on the earth using the WGS 1984 ellipsoid, its Matlab implementation and some test points are provided below.

5.5.1 Matlab implementation of the LPO distance algorithm

```
function [range, A12, A21] = distance_lpo(lat, long)
%
% Computes distance and bearing between points on the earth using WGS 1984
% ellipsoid
%
% [range, A12, A21] = distance_lpo(lat, long) computes the ranges RANGE between
% points specified in the LAT and LONG vectors (decimal degrees with positive
% indicating north/east). Forward and reverse bearings (degrees) are returned
% in AF, AR.
%
% Ellipsoid formulas are recommended for distance d<2000 km,
% but can be used for longer distances.
%
% GIVEN THE LATITUDES AND LONGITUDES (IN DEG.) IT ASSUMES THE IAU SPHERO
% DEFINED IN THE NOTES ON PAGE 523 OF THE EXPLANATORY SUPPLEMENT TO THE
% AMERICAN EPHEMERIS.
%
% THIS PROGRAM COMPUTES THE DISTANCE ALONG THE NORMAL
% SECTION (IN M.) OF A SPECIFIED REFERENCE SPHEROID GIVEN
% THE GEODETIC LATITUDES AND LONGITUDES OF THE END POINTS
% *** IN DECIMAL DEGREES ***
%
% IT USES ROBBIN'S FORMULA, AS GIVEN BY BOMFORD, GEODESY,
% FOURTH EDITION, P. 122.  CORRECT TO ONE PART IN 10**8
% AT 1600 KM.  ERRORS OF 20 M AT 5000 KM.
%
% CHECK:  SMITHSONIAN METEOROLOGICAL TABLES, PP. 483 AND 484,
% GIVES LENGTHS OF ONE DEGREE OF LATITUDE AND LONGITUDE
% AS A FUNCTION OF LATITUDE. (SO DOES THE EPHEMERIS ABOVE)
%
% PETER WORCESTER, AS TOLD TO BRUCE CORNUELLE...1983 MAY 27
%

% On 09/11/1988, Peter Worcester gave me the constants for the
% WGS84 spheroid, and he gave A (semi-major axis), F = (A-B)/A
% (flattening) (where B is the semi-minor axis), and E is the
% eccentricity, E = ( (A**2 - B**2)**.5 )/ A
% the numbers from peter are: A=6378137.; 1/F = 298.257223563
% E = 0.081819191
A = 6378137.;
E = 0.081819191;
B = sqrt(A.^2 - (A*E).^2);
EPS = E*E/(1.-E*E);

NN = max(size(lat));
if (NN ~= max(size(long))),
   error('dist: Lat, Long vectors of different sizes!');
end

if (NN == size(lat))
   rowvec = 0;  % it is easier if things are column vectors,
else
   rowvec = 1; % but we have to fix things before returning!
end;

% convert to radians
lat = lat(:)*pi/180;
long = long(:)*pi/180;
```

```
% fixes some nasty 0/0 cases in the geodesics stuff
lat(lat == 0) = eps*ones(sum(lat == 0), 1);

% endpoints of each segment
PHI1 = lat(1:NN-1);
XLAM1 = long(1:NN-1);
PHI2 = lat(2:NN);
XLAM2 = long(2:NN);

% wiggle lines of constant lat to prevent numerical problems.
if (any(PHI1 == PHI2))
   for ii = 1:NN-1
      if (PHI1(ii) == PHI2(ii))
         PHI2(ii) = PHI2(ii) + 1e-14;
      end
   end
end
% wiggle lines of constant long to prevent numerical problems.
if (any(XLAM1 == XLAM2))
   for ii = 1:NN-1
      if (XLAM1(ii) == XLAM2(ii))
         XLAM2(ii) = XLAM2(ii) + 1e-14;
      end
   end
end

% COMPUTE THE RADIUS OF CURVATURE IN THE PRIME VERTICAL FOR EACH POINT
xnu = A./sqrt(1.0-(E*sin(lat)).^2);
xnu1 = xnu(1:NN-1);
xnu2 = xnu(2:NN);

% COMPUTE THE AZIMUTHS.
% A12 (A21) IS THE AZIMUTH AT POINT 1 (2) OF THE NORMAL SECTION CONTAINING THE POINT 2 (1)
TPSI2 = (1.-E*E)*tan(PHI2) + E*E*xnu1.*sin(PHI1)./(xnu2.*cos(PHI2));
PSI2 = atan(TPSI2);

% SOME FORM OF ANGLE DIFFERENCE COMPUTED HERE??
DPHI2 = PHI2-PSI2;
DLAM = XLAM2-XLAM1;
CTA12 = (cos(PHI1).*TPSI2 - sin(PHI1).*cos(DLAM))./sin(DLAM);
A12 = atan((1.)./CTA12);
CTA21P = (sin(PSI2).*cos(DLAM) - cos(PSI2).*tan(PHI1))./sin(DLAM);
A21P = atan((1.)./CTA21P);

% GET THE QUADRANT RIGHT
DLAM2 = (abs(DLAM)<pi).*DLAM + (DLAM>=pi).*(-2*pi+DLAM) + (DLAM<=-pi).*(2*pi+DLAM);
A12 = A12 + (A12<-pi)*2*pi-(A12>=pi)*2*pi;
A12 = A12 + pi*sign(-A12).*(sign(A12) ~= sign(DLAM2));
A21P = A21P + (A21P<-pi)*2*pi - (A21P>=pi)*2*pi;
A21P = A21P + pi*sign(-A21P).*(sign(A21P) ~= sign(-DLAM2));
% A12*180/pi
% A21P*180/pi

SSIG = sin(DLAM).*cos(PSI2)./sin(A12);

% At this point we are OK if the angle < 90 but otherwise
% we get the wrong branch of asin!
% This fudge will correct every case on a sphere, and *almost*
% every case on an ellipsoid (wrong handling will be when
% angle is almost exactly 90 degrees)
dd2 = [cos(long).*cos(lat) sin(long).*cos(lat) sin(lat)];
dd2 = sum((diff(dd2).*diff(dd2))')';
if (any(abs(dd2-2) < 2*((B-A)/A))^2),
   disp('dist: Warning...point(s) too close to 90 degrees apart');
end
bigbrnch = dd2>2;

SIG = asin(SSIG).*(bigbrnch==0) + (pi-asin(SSIG)).*bigbrnch;
A21 = A21P - DPHI2.*sin(A21P).*tan(SIG/2.0);

% COMPUTE RANGE
G2 = EPS*(sin(PHI1)).^2;
G = sqrt(G2);
H2 = EPS*(cos(PHI1).*cos(A12)).^2;
H = sqrt(H2);
TERM1 = -SIG.*SIG.*H2.*(1.0-H2)/6.0;
```

```
TERM2 = (SIG.^3).*G.*H.*(1.0-2.0*H2)/8.0;
TERM3 = (SIG.^4).*(H2.*(4.0-7.0*H2)-3.0*G2.*(1.0-7.0*H2))/120.0;
TERM4 = -(SIG.^5).*G.*H/48.0;

range = xnu1.*SIG.*(1.0 + TERM1 + TERM2 + TERM3 + TERM4);

% CONVERT TO DECIMAL DEGREES
A12 = A12*180/pi;
A21 = A21*180/pi;
if (rowvec),
   range = range';
   A12 = A12';
   A21 = A21';
end
```

5.5.2 Test points

The following table provides results of calculation distances from the LPO distance algorithm.

Test #	Longitude point #1	Latitude point #1	Longitude point #2	Latitude point #2	Distance (m)
1	59.137	81.450	132.862	-71.971	17452769.38
2	245.057	-75.309	331.764	-77.086	2110391.35
3	185.622	87.327	183.692	-17.999	11689986.02
4	182.640	20.009	49.196	5.048	14227739.39
5	150.579	41.603	208.973	39.188	4868529.07
6	0.000	0.000	332.341	19.629	3717195.47
7	356.228	79.610	254.896	-47.763	15364005.55
8	199.871	88.917	70.224	52.035	4312751.18
9	287.193	-35.107	200.803	52.926	12831368.01
10	102.486	-83.242	312.077	75.131	18753227.55
11	69.797	88.120	207.543	18.708	8087967.56
12	93.492	-16.942	304.265	20.978	16765984.94
13	199.115	-39.885	182.679	60.574	11263499.39
14	303.234	77.720	332.681	-0.149	8830419.21
15	152.391	-4.042	179.072	-21.859	3490115.84
16	38.772	-90.000	252.147	9.952	11097348.67
17	170.518	85.414	311.396	-28.009	13474193.18
18	83.708	44.039	273.558	48.297	9728568.10
19	325.393	4.457	60.402	-18.541	10702629.73
20	201.680	15.173	142.753	-29.194	8013018.54