# A guide to Argo Regional Quality Assessment in the Southern Ocean at BODC

Compiled by

Kamila Walicka and Clive Neil

Contents

[A guide to Argo Regional Quality Assessment in the Southern Ocean at BODC 1](#_Toc108593716)

[Useful documents 2](#_Toc108593717)

[1. General introduction to Argo regional quality assessment in the Southern Ocean 2](#_Toc108593718)

[2. Location of code directories 3](#_Toc108593719)

[3. Configure the processing environment 3](#_Toc108593720)

[4. Start using the SO quality assessment code 4](#_Toc108593721)

[4.1. Generating a list of d-mode floats from the Southern Ocean 4](#_Toc108593722)

[4.2. DMQC-PCM-main 4](#_Toc108593723)

[4.2.1 Setup configuration files 4](#_Toc108593724)

[4.2.2. Select floats for analysis and run the codes in software 6](#_Toc108593725)

[4.3. DMQC-PCM-Python 7](#_Toc108593726)

[4.3.1. Setup configuration files 7](#_Toc108593727)

[4.3.2. Select floats for analysis and run the codes in software 7](#_Toc108593728)

[Appendix I - Changes to the argopy configuration 8](#_Toc108593729)

# Useful documents

* Profile Classification Model: <https://github.com/euroargodev/DMQC-PCM>
* DMQC processing of Argo floats in Matlab: <https://github.com/ArgoDMQC/matlab_owc>
* DMQC processing of Argo floats in Python: <https://github.com/euroargodev/argodmqc_owc>

# General introduction to Argo regional quality assessment in the Southern Ocean

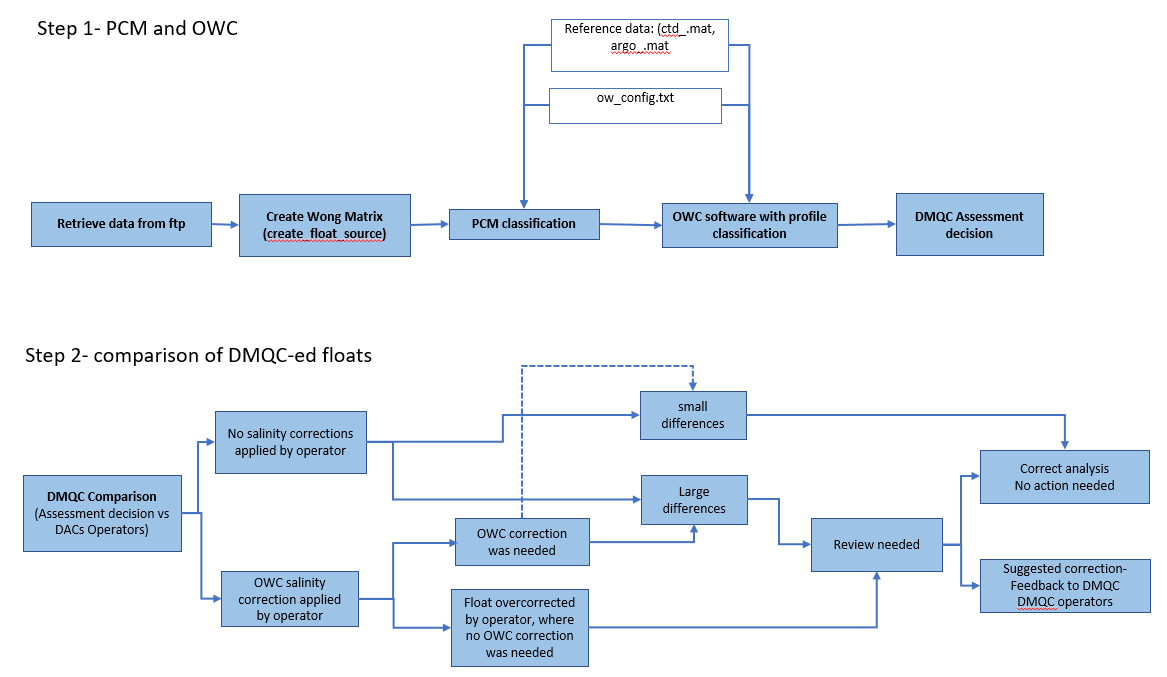
In BODC, the quality assessment of salinity data of Argo floats in the Southern Ocean Arc is performed for floats deployed in the Southern Ocean region (south of 40˚ S). The quality assessment method is used to verify the quality of Argo floats which have been already analysed by DMQC operators from various DACs and submitted to GDAC. A regular check of the quality of Argo float data in delayed mode is necessary to ensure the consistency of analysed salinity data within the basin. Moreover, the DMQC analysis is often **conducted by operators from various national data centres who have different experiences in analysis and oceanographic knowledge from specific regions. Another aspect is that the available reference datasets for DMQC analysis have evolved since the beginning of the Argo project, where a continuous increase in the amount of reference data gives a better opportunity in obtaining more robust results in verifying previous analyses.**

The general concept of performing the quality assessment in the Southern Ocean is to use in the OWC analysis the beforehand classified Argo float and reference data belonging to similar water mass regimes and further to compare the result with currently submitted to GDAC Argo floats in d-mode. This will allow comparing only those profiles which come from the same zone with specified characteristics avoiding noise from other zones which could be underrepresented and under/overestimating the proposed correction of drifted or offset floats.

The classification of data is performed using the Profile Classification Model (PCM). **PCM** is a scientific analysis approach based on vertical profiles classification that can be used in a variety of oceanographic problems (e.g. front detection, water mass identification, natural region contouring, reference profile selection for validation). It is being developed at Ifremer/LOPS in collaboration with IMT Atlantique since 2015. The PCM software is written in Python and is compatible with both Python and Matlab OWC software versions. The OWC toolbox includes routines for calibrating profiling float conductivity sensor drift.

The general workflow of the quality assessment method shows figure below. In the first step, the PCM code retrieves Argo float data from the BODC scratch drive and generates the Wong matrix including all available profiles in d-mode. Next, the PCM method is run using the reference data and configuration information which are the same as for the OWC analysis. The output from the PCM classification is read by the OWC software producing the suggested salinity correction outputs and associated diagnostic plots used to decide if the Argo float requires any salinity corrections.

In the second step, the output from the OWC analysis is compared with the adjusted salinity data submitted to GDAC by the DMQC operator. In this step, the code is comparing the differences between both outputs and generates the comparison plot and list of analysed floats and associated comments from the comparison. Finally, for floats that show significant discrepancies from the initial analysis feedback is sent to the corresponding DMQC operators.



# Location of code directories

The quality assessment method code is located in the following directory

*>> cd /users/argo/dm\_qc/SO\_assesment*

The method consists of two software PCM and OWC. To perform the analysis firstly, the PCM code has to be used then the output from the PCM is used to run OWC software. The SO quality assessment software is designed to allow the user to select the OWC software language that he is currently using for DMQC analysis (Matlab and Python).

The directory of the SO quality assessment software compatible with OWC Matlab code is as follows:

*>> /users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main*

The directory of the SO quality assessment software compatible with OWC Python code is as follows:

*>> /users/argo/dm\_qc/SO\_assesment/DMQC-PCM-Python*

# Configure the processing environment

The SO quality assessment method operates using python. To run the Python program, open your NoMachine desktop. Then create a new terminal window and use *ssh* to open a connection to a suitable server, for example, *liviljobs5*:

**> ssh –X argo@livljobs5**

Load the python environment required for running the PCM code:

**> module add anaconda**

**> source activate so\_dmqc**

This will make the required Python libraries available for running the code.

# Start using the SO quality assessment code

## Generating a list of d-mode floats from the Southern Ocean

The quality assessment of salinity data of Argo floats in the Southern Ocean is performed for floats deployed in the Southern Ocean region (south of 40˚ S). The code allowing the selection of the floats in d-mode is:

**> master\_argo\_index\_reader.py**

This code is reading the *ar\_index\_global\_prof.txt* file with a list of all available Argo floats from the location from a local BODC copy of Argo profiles from GDAC */scratch/argo/gdac\_mirror/.*This local copy is automatically synchronised with the GDAC Ifremer every week.

To run the code to generate a list of floats for SO quality assessment type in the terminal window:

**> python master\_argo\_index\_reader.py**

The output file is the excel *SO\_argo\_floats.xlsx* file including the following information: DAC, float WMO number, mode, profile number, profile date, latitude and longitude. Use this list to extract the Argo floats from the region from which you are going to perform the analysis.

## DMQC-PCM-main

The quality assessment method of Argo floats is compatible with the OWC software written in both Matlab and Python software. To perform the quality assessment of floats using the OWC Matlab code use the following steps.

### Setup configuration files

* ***pcm-config.txt***

The configuration file needed to setup for the DMQC-PCM analysis is located in */users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main/OWC-pcm/matlabow/*

This configuration file contains the locations of the local BODC GDAC mirror and four processing parameters required by the BIC processing. The BIC ([Bayesian Information Criteria](https://en.wikipedia.org/wiki/Bayesian_information_criterion)) can be used to **optimize the number of classes** in the model, trying not to over-fit or under-fit the data. To compute this index, the model is fitted to the training dataset for a range of K values from 0 to 15. A **minimum** in the BIC curve will give you the optimal number of classes to be used. Moreover, for each K range run, a subset of the training dataset is randomly selected to use independent profiles. Indeed, the ocean exhibits spatial correlations that reduce the real information contained in the training dataset. This has to be considered. The dataset is sub-sampled into several subsets of uncorrelated profiles, finally allowing us to compute several times each K range run and hence to compute a standard deviation on the BIC metric.

You can adjust these parameters, however, it is recommended to use the default values, where:

*MAX\_DEPTH = 1000 (maximal* **Interpolation depth).** The PCM cannot deal with “*NaN”* values, so the reference dataset is interpolated on standard depth levels and the profiles shallower than the ***MAX\_DEPTH***, defined above, are dropped out. A max depth of 1000 m can be enough; however, you should find a compromise between keeping a sufficient number of reference profiles and having a comprehensive representation of the oceanography in the region. You should also consider the depth of the float profiles: if they are shallower than the max depth, they will be dropped out, and they will not be classified by the PCM. In such cases, a lower value of max depth is recommended.

*CORR\_DISTANCE = 50 (*Correlation distance in km)

*NUMBER\_RUNS = 10 (*Number of runs in BIC for each K)

*NK = 15 (*Max number of classes to explore)

* ***ow\_config\_linux\_ctd\_argo.txt***

The following file is located in: */users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main/OWC-pcm/matlabow/.* This file is used to set up all mapping parameters used for the OWC processing. For more details look at: *M:\BODC\_DOC\Temp\Argo\DMQC\_guide\_new\_v2.docx*

To best represent the dynamic condition in this region and for further comparison of output with the DM data the constant values of the objective mapping parameters have been used (Table 1).  The floats which are going through the SO quality assessment analysis are firstly run using the first “SO1” configurations. The other sets of configurations are used if the specific float requires more iterations.

The new functionality in this file is the use of the PCM which allows you to select the profiles from climatology characteristics for the analysed float profile.

USE\_PCM=1

Where, 1=use PCM for profile selection, 0=don't use PCM for profile selection

**Table 1:** Configuration setup for the objective mapping parameters.

|  |  |  |  |
| --- | --- | --- | --- |
| **OWC CONFIGURATION** | **SO1** | **SO2** | **SO3** |
| CONFIG\_MAX\_CASTS  MAP\_USE\_PV  MAP\_USE\_SAF  MAPSCALE\_LONGITUDE\_LARGE  MAPSCALE\_LONGITUDE\_SMALL  MAPSCALE\_LATITUDE\_LARGE  MAPSCALE\_LATITUDE\_SMALL  MAPSCALE\_PHI\_LARGE  MAPSCALE\_PHI\_SMALL  MAPSCALE\_AGE  MAPSCALE\_AGE\_LARGE  MAP\_P\_EXCLUDE  MAP\_P\_DELTA    Reference database | 310  1  1  4  2.5  3  1.5  0.1  0.02  5  20  100  150    CTD +Argo | 310  1  1  4  2.5  3  1.5  0.1  0.02  5  20  100  150    CTD +Argo | 310  1  1  4  2.5  3  1.5  0.1  0.02  5  20  100  150    CTD +Argo |
| **Set\_calseries** |  | | |
| Constant on chosen levels  Max breaks | None  1 | **>1000**  1 | <1000  1 |

### Select floats for analysis and run the codes in software

The list of WMO numbers of floats which are intended to go threw the the SO quality assessment needs to be inserted in the following codes. After these edits, these codes need to be run.

* ***so\_dmqc\_master.py***

This code is located in the /*users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main/PCM-design*

This Python code (1) retrieves data from the local repository or GDAC (using [argopy](https://github.com/euroargodev/argopy) package), (2) automatically generates the source code for OWC analysis (using [argopy](https://github.com/euroargodev/argopy) package), (3) runs the BIC function which is estimating the most suitable number of classes for a training dataset to model, (4) runs the DMQC-PCM software and generate the output plots, model and class labels.

* ***ow\_calibration\_pcm.m***

This code is located in the */users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main/OWC-pcm/matlabow*

This Matlab code runs the OWC software including the class labels from the PCM and generates the diagnostic plots.

* ***dac\_comparison.m***

This code is located in the */users/argo/dm\_qc/SO\_assesment/DMQC-PCM-main/OWC-pcm/matlabow*.

The comparison is performed by calculating the differences between the corrections generated from the OWC software from the quality assessment method (from the cal\_series.m file) and the adjusted Argo float data in DM. The output plot visualising the difference  is presented in figure 4.1. Floats with a salinity  correction difference of +/- 0.01 are selected for more detailed checks. This includes multiple OWC iterations including different settings e.g. split time series, select pressure or theta levels. Any indication of the significant discrepancy between the analysed data is described in the generated report and feedback with suggested modifications to salinity data is sent to the operator responsible for this float.

The output list of floats is generated in the terminal window, while the plots are saved to *L:\users\argo\dm\_qc\SO\_assesment\DMQC-PCM-main\OWC-pcm\matlabow\data\float\_comparison\ ctd* directory.

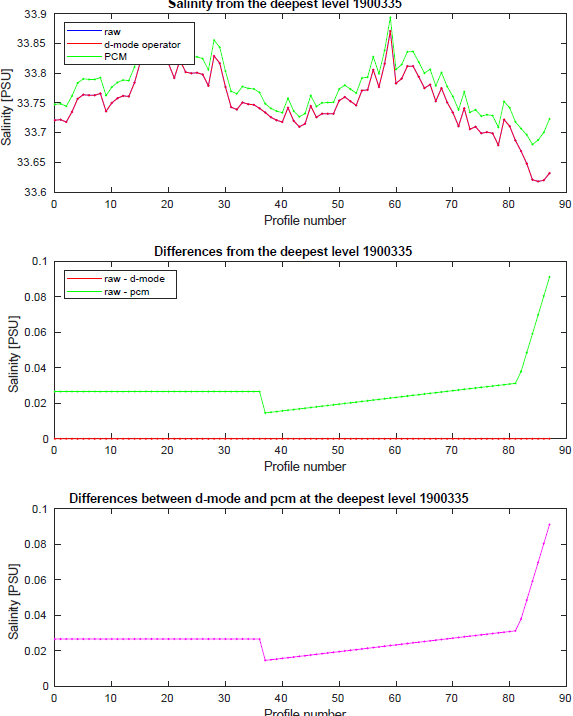


Figure 4.1: Comparison between the results from the quality assessment method of the floats in the Southern Ocean

## DMQC-PCM-Python

The Python version of the software is in the following directory:

**> cd /users/argo/dm\_qc/SO\_assesment/DMQC-PCM-Python/**

To perform the SO quality assessment of floats using the OWC Python code use the following steps:

### Setup configuration files

All necessary directories, constants values for PCM, and objective mapping parameters which are needed to run both PCM and OWC software can be set in one initial file below.  The configurations used are the same as in the DMQC-PCM-main software.

* *pcm\_ow\_config.ini*

### Select floats for analysis and run the codes in software

The list of WMO numbers of floats which are intended to go threw the SO quality assessment needs to be specified in the following codes. After these edits, these codes need to be run.

* ***so\_dmqc\_master.py***

In addition to the code from the DMQC-PCM-main this code is also performing the automatic OWC Python calculations.

* ***master\_dac\_comp.py***

This is a Python version of the dac\_comparison.m from DMQC-PCM-main.

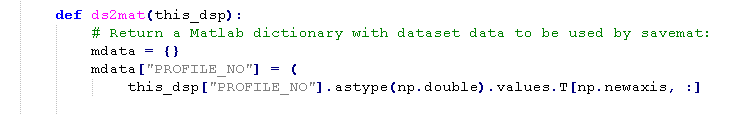
# Appendix I - Changes to the argopy configuration

Currently, the argopy python library that is part of the so\_dmqc environment is 0.1.11. If this library version updates in the future, the following python file needs editing:

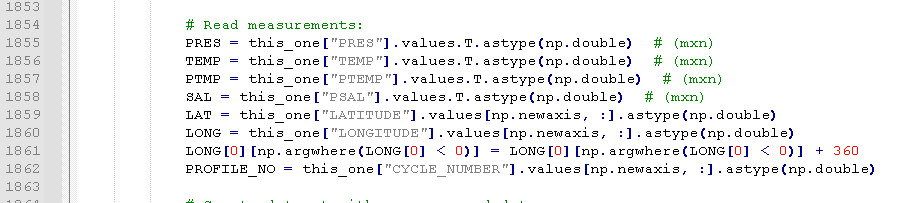
**\users\argo\.conda\envs\so\_dmqc\lib\python3.9\site-packages\argopy\xarray.py**

The data types for the variables saved into the Wong matrix in the create\_float\_source method have to be set to be of datatype “**np.double**”. On version 0.1.11 this is in lines 1737 and from 1835 to 1841 but this might be slightly different in a future release depending on other updates:

1737 change it to the following text



the same for lines 1855 to 1860 and 1862



In order to stop the deprecation warning for np.int the following two python files within the pyxpcm package needs changing:

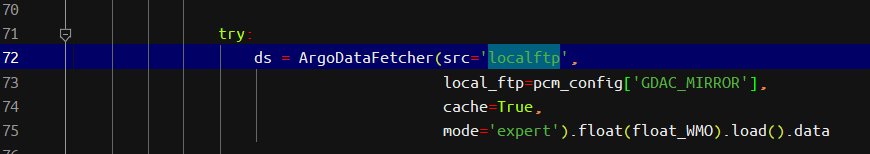
**\users\argo\.conda\envs\so\_dmqc\lib\python3.9\site-packages\pyxpcm\models.py**



**\users\argo\.conda\envs\so\_dmqc\lib\python3.9\site-packages\pyxpcm\plot.py**



The “localftp” method from the Data Fetcher is going to be depreciated in the next release of argopy so the call-in line 72 of the “so\_dmqc\_master\_v<>py” will need to be changed to “gdac”:



After this, the local mirror will not be able to be used. Please check for this on argopy documentation because the “localftp” parameter will be removed.

