m\_moorproc\_toolbox

**OSNAP-CLASS version**

**Please note: this Document is a Life Document and might not always reflect the most current changes of the toolbox**

Lewis Drysdale, Loic Houpert, Sam Jones, Kristin Burmeister, Yvonne Firing

Kristin.burmeister@sams.ac.uk

Contents

[Setup and version control 3](#_Toc108243078)

[1. Git 3](#_Toc108243079)

[1.1. Git setup 3](#_Toc108243080)

[1.2. Check the local and remote branches 4](#_Toc108243081)

[1.3. Making changes 4](#_Toc108243082)

[1.4. Push changes on the original remote repository 5](#_Toc108243083)

[1.5. More on Git 7](#_Toc108243084)

[2. Setup and run locally the toolbox 7](#_Toc108243085)

[2.1. Get the most recent version of the toolbox 7](#_Toc108243086)

[2.2. Create a branch for your development 8](#_Toc108243087)

[2.3. Set up the toolbox in your matlab path 8](#_Toc108243088)

[Real-time and delayed-time processing of microCAT data 8](#_Toc108243089)

[1. Introduction 8](#_Toc108243090)

[2. Set up directory structure 9](#_Toc108243091)

[2.1. Processing scripts: 9](#_Toc108243092)

[2.2. Shipboard calibration files 9](#_Toc108243093)

[3. Processing of the moored microcat 10](#_Toc108243094)

[4. Lowered microcat processing after the shipboard calibration casts 10](#_Toc108243095)

[5. Delayed mode processing of the microcat data. 11](#_Toc108243096)

[5.1. Set-up metadata files 11](#_Toc108243097)

[5.2. Calculation of accurate nominal depth of the mooring instruments, update of the metadata files 11](#_Toc108243098)

[5.3. Calculation of the pre- or post- deployment conductivity and temperature calibration coefficients 11](#_Toc108243099)

[5.4. Gridding of the data: creation of a lowpass filtered, regular gridded data set. 12](#_Toc108243100)

[5.5. merging of the several years of deployment (and if necessary moorings) 13](#_Toc108243101)

[Processing of moored current meter and ADCP data 15](#_Toc108243102)

[1. Introduction 15](#_Toc108243103)

[2. Nortek current meter data processing 15](#_Toc108243104)

[2.1. Stage0 – Data Download (done during the cruise $CRUISE) 15](#_Toc108243105)

[2.2. Stage1 – Conversion to standard RDB format and basic statistics 15](#_Toc108243106)

[2.3. Stage2 – Trimming of data, summary plots 16](#_Toc108243107)

[2.4. Stage3 – Data editing, speed of sound correction, magnetic declination correction, data filtering 16](#_Toc108243108)

[2.5. Stage 4 – Data gridding of individual deployment 18](#_Toc108243109)

[DeepSeapHOx Data Processing 19](#_Toc108243110)

[1. Stage 0 – Data Download 19](#_Toc108243111)

[2. Stage 1 – Conversion to standard RDB format 19](#_Toc108243112)

[Data reports tools 20](#_Toc108243113)

[Export to oceansites format [if required] 24](#_Toc108243114)

[Export to NetCDF for CLASS data working group 24](#_Toc108243115)

[Appendix: Historical and OSNAP CTD profiles close to the moorings 25](#_Toc108243116)

[1.1. RTEB1 25](#_Toc108243117)

[1.2. RTWB1 26](#_Toc108243118)

[1.3. RTWB2 28](#_Toc108243119)

[1.4. RTADCP1 30](#_Toc108243120)

# Setup and version control

Before using the toolbox and making any changes to branches it is essential that any user gains a basic understanding of git and github, which can be found here:

<https://docs.github.com/en/get-started>

## Git

This toolbox is managed using git. The master branch should be the last working version used, updated and revised since the last mooring cruise. The master branch is protected and changes can only be merge via pull requests which are then revised by the administrators of the repository. The following strategy is used to manage the repository.

### Updating the master branch with latest cruise branch

This should be normal text

1. Before each cruise, users of this mooring processing toolbox should request a new branch from the master branch. Before the users make any changes, they should rename their working branch to reflect the upcoming cruise. E.g. for the cruise JC238 the branch name should be 'JC238'.
2. To keep the master branch up to date, a pull request to merge the cruise branch with the master branch should be raised following good housekeeping practices (below)

a) Directly after the cruise to push any important on-cruise changes

b) After on land post-processing when the calibration of the single intruments are finalised and there will be no changes anymore.

1. One of the repository administrators who is not the creator of the branch will review the pull request and check the changes made.
2. If the reviewer finds no issues they will continue to 5. If there are issues or even merge conflicts the reviewer will block the pull request and raise either discussions within the pull request or issues to be addressed by the branch creator. These are open discussion and any contributor are encourage to take part in them to find the best solution. Once an agreement is reached and the creator implemented any nessecary changes and any merge conflicts are resolved steps 2-4 will be repeated.
3. If everything is ok the reviewer can verified the pull request and merge the cruise and master branch.
4. If the cruise branch is finalised and no further changes are expected, it will be locked by one of the administrators making it read only and protecting it from being deleted.

### Working on board

* A copy of the cruise branch should be physically taken on the ship (i.e. osnap/exec/$cruise/) as the internet connection cannot be quarantied on the ship.
* The data (archived from previous cruise) is not kept on git and effort should be made to make sure that the most recent version is copied prior to the cruise/processing. Once the cruise is complete the cruise branch should be pushed to the remote repository (if not already happened during the cruise due to connection problems).

### Housekeeping

It might be nessecary to create feature branches from the cruise branches to solve certain issues (e.g. 'JC238-revise-CM-merging'). When merging the cruise branch into the master branch, any feature branches for the cruise branch should be merged with the cruise branch and deleted so that there is only one branch per cruise. After the final merging (2.b) the cruise branch will be locked by one of the administrators making is read only and no changes can be made.

### Git setup

If you want to set up a local copy of the project git repository to contribute to the project you should do the following steps:

1. First make sure that you have Git installed! See more details here: <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>. If you are using windows you may also want to install a nice terminal (e.g. <https://mobaxterm.mobatek.net>)
2. Open a terminal, move in your working directory (e.g. ~/Work) and then **clone the project** repository with the command*:*

git clone https://github.com/LewisDrysdale/m\_moorproc\_toolbox.git Enter the git repository:

cd m\_moorproc\_toolbox

Or, Fork the repository to your personal account. To do that just click the Fork button on the main repo page. **Add your fork as a remote.** This remote will be named after your github username for clarity. To do that, use the command: ”git remote add your-github-username fork-url”

For example, for a user named lhoupert, run the command:

git remote add lhoupert <https://github.com/lhoupert/m_moorproc_toolbox.git>

### Check the local and remote branches

To list the local and remote branches, use the the command “git branch -av ” . This command returns for example:

$ git branch -av

JC238 3782c45 update for jc238 caldips

\* JC238-dev 3782c45 update for jc238 caldips

master 4f85b54 Merge pull request #5 from ScotMarPhys/dy120\_seaphox

remotes/origin/HEAD -> origin/master

remotes/origin/JC238 3782c45 update for jc238 caldips

remotes/origin/JC238-dev 3782c45 update for jc238 caldips

remotes/origin/add-IBmoors f735df3 de-bug oceansites code

remotes/origin/dy120 c7578cb Merge pull request #7 from ScotMarPhys/dy120-netCDF-dev

remotes/origin/master 4f85b54 Merge pull request #5 from ScotMarPhys/dy120\_seaphox

We can see one local branch (master) and several remotes on the “origin” repository (corresponding to the repository cloned).

If you only have a local *master* branch (as in the example above) and you need to create a local copy of the other remote branches (“legacy”, “postdy120”, etc… ). You can do that with the one-time command:

git checkout -b postdy120 origin/postdy120

git checkout -b legacy origin/legacy

### Making changes

Before any changes, a branch should be created. **Best practices are to never commit to master branch**. It is better to work on a specific branch and then merge the branch onto the “main” or “ master” branch. Your terminal should be set up to display the current active branch.

**1) Update master.** Before you make any changes, first checkout master (or another branch if you are working on another branch) and pull in the latest changes. In the example below we want to work on the branch postdy120 so we do:

git checkout postdy120

git pull

*git pull* is here a shortcut for *git pull origin postdy120*

**2) Create a branch.** Make a branch name that is short, descriptive, and unique. Some examples of good branch names are fix-install, docs-cleanup, and add-travis-ci. Some examples of bad branch names are feature, fix, and patch. To create the branch, do :

git checkout -b branch-name

(e.g.: git checkout -b add-IBmoors)



**3) Make your changes and commit them.** Keep commits atomic <-> each commit should represent a single unit of change. Also write helpful commit messages, so that someone can understand what the commit does just from reading the message without having to read the diff.

a. To see what file have been changes locally run the command ”git status” in a terminal

b. To stage your changes, add the different files with the command ”git add filename [filename2 …]”

(e.g. ”git add OSNAP\_mooring\_processing.docx”)

Figure 1: From https://git-scm.com/about/staging-area

c. To commit the changes to your repository run the command: git commit -m “message explaining what the commit does” (e.g. git commit -m "add Git section to doc" )

**4) Push up your changes on your remote repository.** Once the changes are committed, they are only applied to the local repository. To propagate the changes of the remote repository, the user needs to “push” the changes. It can easily be done on your fork repository by running the command:

git push your-github-username branch-name (e.g. ”git push lhoupert add-IBmoors“)

### Push changes on the original remote repository

As owner of the “forked” github repository, it is easy to push the changes to it. However, because you are collaborating with other people, you will certainly want to push your changes back to the original repository so other users following changes on this repository can be aware of your modifications.

#### **Case 1: Through a pull request (best practices - available for anyone)**

**Make a pull request.** Once the changes are pushed on your forked repository, you can then go to your fork on GitHub. You should see a button to create a pull request from your branch. It will look something like this:



Then by selecting “comparing & pull request” , you will be presented a new page asking you more information about your request (name of the repository you want to merged your branch into, title of the request, description, etc…). You can find more information about the pull request workflow here: <https://www.asmeurer.com/git-workflow/>

#### **Case 2: By pushing changes directly on the origin repository without pull request (only for authorized users).**

When we started working on the postdy120 branch in November 2020, three people were working on the same files and therefore opening a “pull request” was not the most efficient way for everybody to have access to the same files in the last version of the file in real time.

Therefore, we decided to work on the same “original” repository. To do that the owner of the original repository needed first to grant “pushing rights” on his repository. This can be done by opening the settings of the repository, selecting “manage access” and “invite a collaborator”:



Once the user is added as a collaborator, he/she can push changes on the “main” repository using the command: ”git push your-github-username branch-name”

For example: ”git push origin add-IBmoors”

### More on Git

* Tutorial about Git-workflow explained: <https://www.asmeurer.com/git-workflow/>
* More information on git commands: Run ”git help” or “git help -g“ in a terminal or have a look to the git website (e.g. <https://git-scm.com/docs/giteveryday>)
* Some notes on file manipulation with Git: <https://gist.github.com/lhoupert/01daa22e0b2a4171c20a291df14e661b>
* Git Cheat Sheet: <https://training.github.com/downloads/github-git-cheat-sheet.pdf>

## Setup and run locally

### Get the most recent version of the toolbox

If you don’t have a version of the git repository, get the last version of the code. Make sure you select the correct branch (e.g. *postdy120*, *master, etc..*), clone it or download the zip.

If you have already a git version of the repository set up following *Section 1. Git*, go in your git folder then run: git pull origin master (or another branch-name if you working on a specific branch)

This will update the local version of the repository with the latest version of the remote directory.

### Create a branch for your development

Before modifying the code, you should create a new branch. More information are given in the previous section, see 1.4)

### Set up the toolbox in your matlab path

Once the mooring processing toolbox is set up, copy the file in Documents/exampleofstartupfile.m in you matlab working directory and rename it (e.g. startup\_moor\_proc.m).

Open the startup file and edit the first two variables (pathosnap and pathgit) in order to set up the matlab paths properly.

* pathosnap should be pointing to the directory containing the OSNAP data

├── Documents

├── Figures

├── cruise\_data

├── data

├── exec

├── matlab\_diaries

└── users

* pathgit should be pointing to the mooring processing toolbox (the github repository). For example: pathgit = ‘/Users/locupe/Dropbox/Work/Python/Repos\_perso/m\_moorproc\_toolbox’

# Real-time and delayed-time processing of microCAT data

## Introduction

→ calibration of the instruments is essential to obtain accurate measurements and minimize errors in the transport calculation

→ calibration data obtain for each moored CTD during intercalibration cast: Each moored CTD is attached to a shipboard CTD rosette frame and a deep cast is carried out prior to deployment and after the recovery of the moored instruments. During the cast, 12 bottle stops of 5 mins each are performed, from the bottom to the surface, in order to cover the different range of pressure of the moored instruments. The comparison with the 1Hz-bin averaged shipboard CTD data gives calibration coefficients for each sensor (T, C, P) on each moored instrument.

Outline of the processing stages:

* Initial processing setup: creation of a data processing metadata control file.
* Raw data downloading and archiving
* Conversion from instrument format to standard Rapid Data Base (RDB) format
* Trimming of data record, basic statistics and summary plots
* Calibrating and quality control of the moored CTD data:
* Gridding of the data: creation of a lowpass filtered, regular gridded data set

**NOTE: the OSNAP data structure separates executable scripts (*osnap/exec/…*) from output files (*osnap/data/moor/…*). For clarity, file (or directory) names and paths for scripts are coloured red, input files are coloured blue, and output files are coloured green. The output files of some scripts may become the input of a later script. Hopefully this will make navigation of this document a little easier.**

## Set up directory structure

### Processing scripts:

If it doesn’t exist, create a directory *delayed\_processing\_script* in *osnap/exec/$cruise/* (from the previous cruise)

### Shipboard calibration files

#### **Cruise ctd files (pre and post deployment)**

1. Create the directory for the cruise data: *osnap/cruise\_data/$cruise*
2. Copy the cruise data from the archive directory of the cruise data (e.g. M:\Mar\_Phys\Cruises\CRXXX) ;
3. Look at the previous year cruise to copy the same type of ctd files ( \*\_1hz.nc, \*.ros and \*\_align\_ctm.cnv) for each caldip cast. The number of the caldip cast can be found in the cruise report or in *osnap\data\moor\proc\_calib\crxxx\cal\_dip (cast\*info.dat).*

#### **Microcat caldip data**

1. create a *cast???info.dat* file in *osnap/data/moor/proc\_calib/$cruise/cal\_dip/* for each caldip that summarizes information about each caldip CTD cast (location, time) and the serial numbers and the deployment periods of the lowered microcats. The deployment period number is used later (part 6b) to create a metadata file of the mooring deployed during the deployment period.
2. Create the directories that will host the raw microcat data files for each CTD cast, *osnap/data/moor/raw/$cruise/microcat\_cal\_dip/cast???/* (where *???* is the cast number), and move in this directory the raw data file (.hex, .cnv, .xml, .xmlcon, etc..)

#### **Microcat deployment files**

1. creation of a data processing control file *$moorname$info.dat*  for each mooring in the directory *osnap/data/moor/proc/$moorname$*. The *$moorname$info.dat* file contains metadata for mooring position (lon, lat, waterdepth and mean magnetic deviation during the deployment), deployment period, nominal depths and serial numbers of each instrument. \*see previous cruise folder for an example.
2. copy the raw SBE37 files (with .cnv files) in the directory *osnap/data/moor/raw/$cruise/microcat/.* If the raw .cnv files are not named as *$serialnumber\_data.cnv*, rename them.

#### **Start and Set-up Matlab**

1. Open in a terminal and go in the root mooring processing directory (e.g.: *OSNAP\_mooring\_data\_processing/osnap/* ).
2. Check that in the top of the *startup.m* file a “cd” command is sending you in the cruise directory you want to process (e.g. *cd exec/pe400*).
3. Once matlab started under the *osnap/* directory the *startup.m* file should move you in the cruise directory you are interested in. In the cruise directory *osnap/exec/$cruise/*, another *startup.m* file should be present, generating the different path related to the cruise data.

## Processing of the moored microcat

1. Stage 1: Edit (paths to the data, mooring name, year of the first measurement) and run the script *osnap/exec/$cruise/stage1/microcat/mc\_call\_2\_$cruise.m* : convert the raw data to RDB formatted file .raw for an entire mooring. The processed data are stored in *osnap/data/moor/proc/$moor/microcat/*
2. Stage 2: Check that the deployment time and recovery time are accurate in the corresponding *$osnap/data/moor/proc/$moor/moor\_info.dat* file, then edit and run *osnap/exec/$cruise/ stage2/microcat/microcat\_raw2use\_003\_with\_ODO.m* . This script generates *.use* files (launching and recovery period removed) in *osnap/data/moor/proc/$moor/microcat/* . The script also creates data overview sheet including basic statistics, and produces summary plots, including 2-day low-pass plots.

## Lowered microcat processing after the shipboard calibration casts

The script *osnap/exec/$cruise/stage1/microcat/mc\_call\_caldip\_v4b.m* loads:

1. The raw microcat data located in *osnap/data/moor/raw/$cruise/microcat\_cal\_dip/$castnber/* ,
2. The shipboard CTD data *(if new calibrated ctd files are available after the cruise, the \*\_raw.nc and \*\_psal.nc have to be replaced)* files for the *$castnber* ( in *osnap/data/$cruise/)*
3. The caldip metadata file *castnnninfo.dat* file located in *osnap/data/moor/proc\_calib/$cruise/cal\_dip/*.

The script writes to a directory *~/osnap/data/moor/proc\_calib/cal\_dip/$cruise/microcat/$castnber/* which is created manually.Plots are generated for all microcat data for one CTD cast with the shipboard CTD data.

***Note that the raw microcat files have to be named as serialnumber\_cal\_dip\_data.cnv.***

The script *osnap/exec/$**cruise/stage1/microcat/mc\_caldip\_check\_$cruise.m* provides a quick quantitative comparison of Microcat cal-dip data with the SBE911 data from the CTD. Data obtained at the deepest bottle stops are used. For each instrument differences of conductivity, temperature and pressure between the instrument and the CTD sensor were calculated. The mean and standard deviation of the differences for each instrument are then presented in a tablein  *~/osnap/data/moor/proc\_calib/cal\_dip/$cruise/microcat/$castnber$/microcat\_check$castnber$.log*

## Delayed mode processing of the microcat data.

Before starting the delayed mode processing of the microcat, make sure that you have the final calibrated CTD dataset.

### Set-up metadata files

Manually add entries in two metadatabases: *cruise\_id.xls* lists the cruise id and unique number identifier; *microcat\_calib\_cruise.csv* lists the mooring name, the associated deployment and recovery cruises number. The metadata files are under *osnap/data/moor/cal\_coef/*

### Calculation of accurate nominal depth of the mooring instruments, update of the metadata files

If there are large differences between the planned instrument depth and the actual depth for most of the deployment (due to trawling etc), it is possible to calculate this from the observed pressures. Generally, we have not needed to do this stage but the following might be useful.

*Note this just updates the ‘nominal depth’ which is used for metadata and file organisation, but will not impact the final data product.*

The gridding algorithm pulls info directly from the pressure record for example. Run the function *ctd\_instrdpth2.m* for each mooring. (the function is in data/CRUISE/calibration). This will calculate accurate instrument depth from pressure record and update the *info.dat* file located in *osnap/data/moor/proc/$mooring/*

b) Manual creation of a *osnap/data/moor/cal\_coef/osnapXX\_deploymentdepths.dat* file, which lists all the instruments on every mooring deployed within the deployment period XX (by using the info.dat file for each mooring in *osnap/data/moor/proc/$mooring/* )

### Calculation of the pre- or post- deployment conductivity and temperature calibration coefficients

o) The raw microcat data files *.raw* have to be generated before starting this process (by doing the step III for example)

a) Before processing the mooring data, the calibration coefficient has to be calculated from the comparison between the calibrated shipboard CTD and the lowered mooring CTDs, this task is achieved by running the script *osnap/exec/$cruise/delayed\_processing\_script/mcat\_final\_calibration/*

*caldip\_coefficient\_calculation.m*. Several user parameters have to be edit in the beginning of the

*caldip\_coefficient\_calculation.m* script.

Key parameters:

* ***P\_insitucal.interval\_move*** – this is the time intyerval n which data are selected for each bottle stop

b) Setup the paths to the data files in insitu\_cal\_osnap2.m (by adding an entry for the cruise in the if/else part). Make sure that

c) Run *caldip\_coefficient\_calculation.m* for each caldip cast, the user can adjust the options for the figure at the beginning of the script. The script produces plots and a table of the calibration coefficients in *osnap/data/moor/proc\_calib/$cruise/cal\_dip/microcat/$castnber/.* The calibration coefficients are then manually entered into 3 CSV format database in *osnap/data/moor/cal\_coef/* (*microcat\_cond.csv*, *microcat\_temp.csv* and *microcat\_pres.csv*). This process is quite arduous and ultimately we would like to automate. For a new cruise, a new set of 4 columns should be added in *microcat\_cond.csv*, *microcat\_temp.csv*. For *microcat\_pres.csv* , two set of 4 columns are added: one for the mooring which have been recovered during the cruise and one for the mooring that have been deployed. Note the format used in the csv tables for previous ‘before-after deployment’ cruise pairings and emulate. If you freeze the column and row headings (‘freeze panes’ in Excel) you can keep them visible while manually entering. Triple check!

**Entering pressure coefficients**

After each cruise, and creation of the deployment depths document, you must run the *caldip\_coefficient\_calculation.m* script again for the previous cruise but with the next deployment depths

*Each cruise has two sets of columns, one set for deployment and another set for recovery. Each set consist of a column for the mooring in the form MMMM\_DD\_YYYY, where MMMM is the mooring id (e.g. rteb1), DD is the deployment number (e.g 02), and YYYY is the year. In the recovery column you should enter the mooring that the instrument was recovered from including the deployment number and year of that deployment. In the recovery column, you should enter the mooring that the instrument was then deployed and the future deployment number and year. For example, instrument 1234 was recovered from rteb1\_01\_2020 and was deployed on ib3\_02\_2022.*

*dp is the interpolated and average pressure difference from bottle depths, dp\_ext is the extrapolated difference if the CTD is shallower than the instrument deployment depth*

d) Set-up the scripts that apply the calibration corrections ( in *osnap/exec/$cruise/delayed\_processing\_script/ mcat\_final\_calibration/microcat\_apply\_cal\_plus\_rbr\_idr\_osnap.m*).

Particularly:

* + the string formats for the reading of the .csv files has to be edited according to the number of columns in the csv files (variable *strformat* in the header of the script).
  + the reference CTD loaded for the QC plots. The cruises used for the plots are defined by the variable *ctd\_ref\_cruises*, then the user has to edit the end of the script (if/else section) by adding plotting instructions specific to each cruise.

e) Once the script is set-up, *microcat\_apply\_cal\_plus\_rbr\_idr\_osnap.m* applies the calibration coefficients for each time-series and bad data are removed. Constant offsets and conductivity pressure correction are applied if required.

* Average trend is
* General trend: to be applied when pre - or post cruise calibration is missing for individual instruments

The end result of this stage is the processed (but not gridded) microcat data in

*osnap\data\moor\proc\$mooring\microcat.*

Note that, if the script discovers a version of the processed data already exists in this directory, it will copy it, append its creation date and save it as a backup. Therefore the ‘to use’ data file is the one without any date appended, e.g. *rtwb1\_04\_2017\_001.microcat*

### Gridding of the data: creation of a lowpass filtered, regular gridded data set.

*Update the mooring file names and microcat order field in the script \osnap\exec\$cruise\stage3\gridding\MCAT\grid\_osnap\_mcat\_data.m*

* Check initialisation parameters listed in that script.
* Update the mooring file names and microcat order.
* If you want to run the gridding on un-calibrated data you must update the instrument order and the uncomment p\_hydrogrid.mc\_int uncomment variable later in the script.
* It calls the worker script: *hydro\_grid\_osnap\_linear\_interp.m* in the same directory which does the following:
  + -Load microcat data
  + -Fill pressure gaps
  + -Convert conductivity to salinity and despike
  + -Apply low-pass temporal filter
  + -Close big gaps in salinity
  + -Interpolate vertically
  + -Save and generate plots.

Data are saved as a mat file in *~\osnap\data\moor\proc\hydro\_grid\.* The gridded .mat files have the variable naming conventions listed in Table 1

### merging of the several years of deployment (and if necessary moorings)

First, we have to create 2 .dat files with instrument data from EB1 in one file. The same for WB1 and WB2 combined in the same file. Note, only instruments on WB2 that are at unique depth should be included. Note 2: in the .dat file, the 3rd columns correspond to the position of the mcat in the grid file created in 4) (can be found under *~/osnap/data/moor/proc/hydro\_grid/*

E.g.: *~\osnap\exec\$cruise\stage3\gridding\MCAT\ rtwb\_osnap\_03\_2016.dat*

Info can be found in:

*~\osnap\data\moor\proc\rtwb1\_03\_2016\rtwb1\_03\_2016info.dat*

Second, check and update paths and parameters under SET DIRECTORIES FOR MERGE and MERGE INITIALISATION (e,g, jg\_end and lastyeardata) headers in *grid\_osnap\_mcat\_data.m*

Last, update the scripts *merge\_osnap\_data\_west.m* and *merge\_osnap\_data\_east.m* by copy and pasting the last merge block section. In particular: create a new deployment by copying a previous one; change all the variable name (e.g. for osnap3, changes fileID2 -> fileID3, T2 -> T3, etc…); change the figure and titles (l326-331-336-341); and change concatenation.

-Run *grid\_osnap\_mcat\_data.m* . This calls *merge\_osnap\_data\_west.m* or *merge\_osnap\_data\_east.m* depending on the mooring option you select. You are prompted to select the individual deployments to concatenate.

Output files are saved in: *~\osnap\data\moor\proc\hydro\_grid\_merged\.*

Table 1. The gridded and merged .mat files have the following variable naming conventions using temperature data as an example:

|  |  |
| --- | --- |
| **Time grid** | |
| Jd\_grid | julian day for ungridded data – 2 hour timesteps. NOTE the JD naming convention – currently the wrong way round! |
| jd | julian day for gridded data – interpolated onto 1/2-day timesteps |
| JG | Julian day |
| **Time grid** | |
| T,C,S,P | Temporal interpolated data (1/12 day) |
| T,S,Pf | Low pass filtered (on 1/12 day) |
| T,S,Pfs | Temporal interpolated filtered (1/2 day) |
| T,S,PGfs | Temporal interpolated (1/2 day) filtered gridded (20 m) |
| **Merge** | |
| T,S,Pfs | Temporal interpolated filtered (1/2 day) |
| T,S,PGfs | Merged temporal interpolated (1/2 day) filtered gridded (20 m) |
| T,SGfs2 | Merged de-spiked temporal interpolated (1/2 day) filtered gridded (20 m) horizontal interpolation |

# Processing of moored current meter and ADCP data

## Introduction

The purpose of this document is to describe the data processing applied to all current meter records collected from the SAMS contribution of the UK-OSNAP project, including initial quality control of the data, data editing, corrections applied, and filtering.

The current meters used on the 4 SAMS-OSNAP moorings for the period 2014-2015 consisted of Nortek Aquadopps for 3 moorings (RTEB1, RTWB1 and RTWB2) and a RDI-ADCP 75kHz on RTADCP1. The Nortek Aquadopps had recording intervals of 20 minutes, while the ADCP has a recording interval of 1 hour.

Note: ~ refers to *~\OSNAP\_mooring\_data\_processing*

*Note 2: the OSNAP data structure separates executable scripts (osnap/exec/…) from output files (osnap/data/moor/…). For clarity, file extensions for scripts are coloured red, input files are coloured blue, and output files are coloured green. The output files of some scripts may become the input of a later script. Hopefully this will make navigation of this document a little easier.*

## Nortek current meter data processing

### Stage0 – Data Download (done during the cruise $CRUISE)

Raw instrument data are downloaded from the instrument after the recovery of the mooring. Record keeping of the download is done on paper and for each instrument a download sheet is completed. After the download, the data are backup and transferred to the network drive on the cruise directory *$cruise/SAMS\_moorings/nortek .* Instrument setup details and data are sorted in different directory by mooring name.

Then the data (.dat, .aqd and .hdr files) are copied on the processing computer in the directory e.g ~\*osnap\data\moor\raw\$cruise\nortek*

### Stage1 – Conversion to standard RDB format and basic statistics

You must start by running *~\OSNAP\_mooring\_data\_processing\osnap\ startup\_$cruise.m*

A text file containing the serial numbers of the Nortek on the mooring and the filenames containing the data has to be created as follows:

*~\osnap\data\moor\raw\$cruise\nortek/rteb1\_03\_2016\_filenames.txt.*

The script *~\osnap\exec\$cruise\stage1\nor/process\_nors\_$cruise.m* performs stage1 and stage 2 processing on nortek data. It converts nortek data from raw to rodb format for an entire mooring. The user needs to modify the following information in the beginning of the script: cruise, the mooring name and operator.

*process\_nors\_$cruise* calls *nortek2rodb\_01*, which saves the files downloaded by the instrument software (stage 0) to the RDB formatted file *.raw (*in */osnap/data/moor/proc/rteb1\_03\_2016/nor.).* The script *nortek2rodb\_01* also create data overview sheet including basic statistics:

*~\osnap\data\moor\proc\rteb1\_03\_2016\nor\rteb1\_03\_2016\_Nortek\_stage1.log*

### Stage2 – Trimming of data, summary plots

The script *process\_nors\_$cruise* also performs stage 2 processing on nortek data by calling the script *nortek\_raw2use\_02.* This script uses the raw data file *mooring\_serialnum.raw* generated by stage1 and the <*mooring>info.dat* file:

*~\osnap\data\moor\proc\rteb1\_03\_2016*

and produces a *.use* file. It removes the launching and recovery period and produce summary and diagnostic plots (figure 1, 2), including filtered data for display and diagnostic purposes (figure3).

### Stage3 – Data editing, speed of sound correction, magnetic declination correction, data filtering

The script *proc\_stage3\_adcp\_nortek in*: *~\OSNAP\_mooring\_data\_processing\osnap\exec\$cruise\delayed\_processing\_script\adcp\*

is used to perform stage3 processing on current meter data, it calls the function *cm\_edit\_NOCS.m*. Once the data trimming procedure and an initial data control are completed, the data can be corrected for speed of sound and magnetic declination. Magnetic deviation value has to be defined in: ~\osnap\data\moor\proc\rteb1\_03\_2016\ *$mooring\_info.dat* file by adding a line like “*MagDeviation = -19.66”.*

*All Nortek current meters UK-OSNAP project use a fixed speed of sound (1500m/s), but this is still worth checking in the .cap files! The corrected sound speed is obtained by using measured values of pressure and temperature data from the Nortek and a local/regional value of salinity. For the OSNAP 2014 cruise, a mean salinity was defined for each Nortek current meter using the moored microcat and the CTD cruise profiles (Annex 2), and local salinity values are varying between 34.95 and 35.4. The corrected sound speed varies between 1491 and 1499 cm/s (Annex 3). Corrected velocities are then obtained by multiplying the uncorrected velocities with the ratio of the measured sound speed with a fixed sound speed.*

Typical processing responses are:

Was a fixed value of sound speed of 1500 [m/s] used when the instrument was setup? y/n y

Do you want to use nearby microcat data to correct the speed of sound? y/n n

What constant value should be used as fixed salinity (in PSU) for speed of sound correction? 35.1

Mean deployment value for the old speed of sound [m/s]: 1500.00

Mean deployment value for the new speed of sound [m/s]: 1492.82

u-component despiking:

Further manual editing required? 0=no, 1=yes: 0

v-component despiking:

Further manual editing required? 0=no, 1=yes:

Then the recorded velocity components are transformed into true east and north components using the local magnetic declination estimated on the median of the deployment and recovery times of each mooring from the NOAA's National Geophysical Data Centre (http://www.ngdc.noaa.gov/geomag-web/).

Then spikes from the current meter data are removed using the spike/mean ratio criterion of 10, and visual inspection can also be used to identify remaining spikes. If data gaps are less than 24 hours, they are interpolated linearly in time, otherwise flag values are inserted.

Despiked data are low pass filtered using a 40-hour Butterworth filter to remove tidal and inertial oscillations, and are then interpolated onto 12-hour time steps (figure 4).

Finally, corrected data files that have been de-spiked and interpolated onto 12 hour time steps are saved in *.edt* files in ~\osnap\data\moor\proc\rteb1\_XX\_XXXX\nor\.

### Stage 4 – Data gridding of individual deployment

A. Create new case and run the script:

*~\osnap\exec\$cruise\stage3\gridding\CM\velocity\_grid\_nor*

C. Data are saved as a mat file in *~\osnap\data\moor\proc\velocity\_grid\*

Stage 5 – Gridding and merging of the several years of deployment

Create dat files e.g

*~ \osnap\exec\$cruise\stage3\gridding\CM\ rteb\_CM\_osnap\_03\_2016.dat*

*Note that if data instruments were not available (e.g. an instrument failed) then that should not be included in the .dat file. Make sure number order of instruments is sequential!*

Copy old file e.g. *CM\_rteb\_merging\_02\_2015.m* and rename then edit as follows:

* Create new moor (e.g. moor3 = 'rteb\_CM\_osnap\_03\_2016')
* Rename directories
* Edit time variables **jg\_end** and **lastyeardata**
* Add a new cruise (e.g. 2c. OSNAP 3 (DY053 --> $CRUISE)), copy the relevant code and rename variables to reflect new cruise number (i.e. 3 > 4, 4 > 5 etc.)

Thoroughly check through the script for other instances that need to be changed with the addition of a new mooring.

# DeepSeapHOx Data Processing

## Stage 0 – Data Download

After download the data are backed up and transferred to the network drive, then copied onto the processing computer in the directory *osnap/data/moor/raw/$cruise/seaphox\_caldip*

## Stage 1 – Conversion to standard RDB format

The script *seaphox\_call\_caldip* performs stage1 processing on DeepSeapHOx data. It converts the data from raw to RDB format. The user needs to modify some information in the beginning of the script like the directory trees, the mooring name or caldip cast number. This script calls seaphox2rodb\_01, which saves the file downloaded by the instrument software (stage 0) to the RDB formatted file *.raw a*nd produce summary plots and statistics for each instrument.

Download DeepSeapHOx and transfer to the processing computer:

*~/osnap/data/moor/raw/$cruise/seaphox/ .* For the 2017-2018 deployment of the SeapHOx 117 the data appear to have been written on several files C00000\*.CSV, and several data files from the test on DY078 where still present on the memory of the DeepSeapHOx. After keeping a copy of all the original files, the data files were manually edited to make sure than only the data relevant for the 2017-2018 RTEB1 deployment were kept in this the seaphox raw directory.

If the relevant data are present in several C000\*.CSV file, edit and run the script (with bash shell) *~/osnap/data/moor/raw/$cruise/seaphox/merge\_sort\_CSVfiles.sh.* This script will sort the lines of several CSV files generated by the SeapHOx and merge them into a single file. The name of this file, containing all the data, has to be indicated in the moor\_filenames.txt file (e.g. rteb1\_04\_2017\_filenames.txt).

Make sure that the relevant info.dat file contains the serial number of the DeepseapHOx (e.g. moor/proc/rteb1\_04\_2017/rteb1\_04\_2017 info.dat).

Run process\_seaphox\_$cruise.m producing timeseries for each instrument, converting raw data to RODB format and summary statistics.

# Data reports tools

A set of scripts can be used to quickly generated summary statistics and figures for each mooring. These scripts can be run for stage 2 data (.use files) or stage 3 data (.microcat and .edt files). The scripts and functions are under *osnap/exec/dy120/data\_report\_tool/ .* The wrapper script *batch\_all.m* can be run for a list of mooring and for a specific deployment period. It generated multiple figures which are saved under *osnap/Documents/datareports .* As an example, the list of figures generated for IB4 is indicated below.

ib4\_01\_2018\_conductivity\_overlay\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_currents\_stacked\_hordirection\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_currents\_stacked\_horspeed\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_currents\_stacked\_u\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_currents\_stacked\_v\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_currents\_stacked\_w\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_horcurrents\_overlay\_dir\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_horcurrents\_overlay\_speed\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_pden1000\_overlay\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_conductivity\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_pden1000\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_pressure\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_ptmp1000\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_salinity\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_plot\_stacked\_temperature\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_pres\_vs\_v.png

ib4\_01\_2018\_pressure\_overlay\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_pro\_vec\_proclvl\_2.png

ib4\_01\_2018\_ptmp1000\_overlay\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_sal\_vs\_u.png

ib4\_01\_2018\_salinity\_overlay\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_stick\_plot\_proclvl\_2\_unfilt.png

ib4\_01\_2018\_temperature\_overlay\_proclvl\_2\_unfilt.png

One of the main advantages of running *batch\_all.m* after processing the stage 2 data (.use) is the ability to see on a single figure a specific variable (temperature, conductivity, pressure, horizontal current) measured by all the mooring instruments. The subpanels for “stacked” figures are created from the info.dat file, therefore a missing instrument will clearly appear as a blank subpanel (Figure M3).

In addition, the wrapper script *batch\_all.m* is calling the function *stats\_table.m* which produces a summary table with statistics for every instrument(Figure M4). The output ascii files are saved under *osnap/Documents/datareports/stats/ .*

*batch\_all\_severalyr.m* is also a useful wrapper script as it allows to plot time-series of multiple deployments on a same figure (Figure M5). It can particularly be useful to check the coherence between several deployment (e.g conductivity) and spot any anomalies.



Figure M3: Figure of the meridional velocity at RTEB1 generated by the script *batch\_all.m*



Figure M4: Example of summary statistics file generated by the wrapper script *batch\_all.m*



Figure M5: Temperature time-series at RTEB1 generated by the script *batch\_all\_severalyr.m* . Figures are saved under *osnap/Documents/datareports/figs/rteb1\_04\_2017\_05\_2018/*

# Export to oceansites format [if required]

The mooring data are annually submitted to Feili Li (OSNAP) in Oceansites netcdf format. Currently we are generating this format (08/2019) though ultimately it may become BODC’s remit. Loic modified a suite of scripts to do this in *osnap\exec\$cruise\export\_Oceansites.*

The top-level script *osnap\exec\$cruise\export\_Oceansites\convert\_SAMS\_mooring\_to\_oceansites.m*

sets up directory paths and declares metadata to populate the netcdf file. This script has to be run once per mooring, note the commented out moorings on lines ~26-29. Scan through the script for any other tweaks necessary for the new mooring deployment. This script calls

*osnap\exec\$cruise\export\_Oceansites\rodb\_to\_oceansitesnetcdf*

which loads the Microcat and current meter datafiles and converts them to netcdf.

Note: the script searches for known instrument IDs (Microcat = 337 or 335, Nortek = 368 or 370) to populate its arrays. This caused some indexing issues when ODO Microcats (with oxygen sensors) were added to the array for the 04\_2017 mooring. Be aware that some careful checking will be required if another new instrument type is added to the moorings, including (but not limited to) lines ~56 – 83 and 86-98. This also applied to the gridding scripts (stages 4 and 5).

Data and plots are saved to

*osnap\exec\$cruise\export\_Oceansites\oceansites\_format*

These tend to be small files which can be emailed as a .zip file or similar.

# Export to NetCDF for CLASS data working group

Change the filenames in the script *osnap\exec\$cruise\export\_THREDDS/Export\_moor\_to\_NetCDF,m* to reflect the most recently processed data and run.

# Appendix: Historical and OSNAP CTD profiles close to the moorings

### RTEB1

|  |
| --- |
|  |
|  |

### RTWB1

|  |
| --- |
|  |
|  |

### RTWB2

|  |
| --- |
|  |
|  |

### RTADCP1

|  |
| --- |
|  |
|  |