

Processing application environment, Data Processing Glider Delayed Time Mode

SOCIB-Data Center Facility

Document type:	Internal procedure
Date:	2016-10-21
Ref. num.	GL_Glider_SOP-DatProc-07-1

Description:	Data processing procedure for Delayed Time Mode Glider data using the glider toolbox v1.2.0. Processing using other glider toolbox versions could be checked in Addendums of this document for the specific release.
Authors:	cmunoz, iruiz
Supervision:	ctroupin
Involved Personnel:	Data Center and Glider Facilities

DOCUMENT VERIFICATION LIST

Date:	Checked by (name)	SOCIB division	Ref.
24/10/2016	ctroupin	Data Center	

DOCUMENT DISTRIBUTION LIST

Date:	Distribution to:
21/10/2016	@datacenter, @glider
29/11/2016	@glider, @grogers (datacentre)

CHANGE RECORD

#	Date	Description	Author	Checked by
1.0	21/10/2016	First version document	cmunoz	ctroupin
1.2	29/11/2016	Revision after 1st processing try	ctroupin	cmunoz
2.0	21/08/2017	Second version document	iruiz	cmunoz

Table of contents

1. INTRODUCTION	4
1.1. Objective	4
1.2. Related Documents	5
1.3. Glider toolbox description	5
1.4. Database	6
2. REQUIRED FEATURES	6
3. DELAYED TIME PROCEDURE DEVELOPMENT	6
4.1 PREPARE DATA	7
4.1.1. Check Post-Mission Raw Data Files	7
4.1.2. Generate Data Processing Directories	10
4.1.3. Establish Symbolic Link in Binary Files	11
4.1.4. Rename Binary Files	12
4.2. ADJUST GLIDER TOOLBOX SETTINGS	13
4.2.1. Deployment ID	13
4.2.2. Binary files conversion	14
4.3. RUN THE MAIN PROCESSING	14
4.4. CHECK OUTPUT NETCDF RESULTS	15
4.5. CHECK OUTPUT FIGURES IN DAPP	15
4.6. SEND EMAIL TO GLIDER TECHNICIAN	16
5. TIPS AND TRICKS	16
5.1. Useful links	16
5.2. Delayed Time processing locally	17
5.2.1. Dependencies	17
5.2.2. Steps to run	17

1. INTRODUCTION

1.1. Objective

The aim of this document is to describe a standardized procedure to conduct the data processing from the Glider Delayed Time mode data using the Glider Toolbox developed by SOCIB (version 1.2.0) which is in the repository */home/glider/glider_toolbox/m*.

The SOCIB glider toolbox (Troupin et al, 2016) covers all stages of the data management process, including: metadata aggregation, raw data download, data processing, data correction and the automatic generation of data products and figures. It is designed to be operated either in real-time or in delayed mode, and to process data from two of the most widely used and commercially exploited glider platforms, Slocum gliders and SeaGliders. The SOCIB glider toolbox is ready to accelerate glider data integration and promote oceanographic discovery.

Ocean gliders are autonomous platforms which dive in a saw-tooth-sampling pattern in the ocean by changing their buoyancy. Depending upon configuration, gliders sample profiles of pressure, temperature, and conductivity as well as various ocean optical parameters. The gliders surface at regular intervals to transmit their observations over the Iridium Satellite.

The toolbox supports different glider models: Slocum (G1 and G2), Seaglider, and SeaExplorer. Figure 1 shows the actual SOCIB glider fleet.

The following sections describe the data processing procedure using as example the deployment SOCIB_ENL_CANALES_MAY2016_SDEEP00_GFMR0045 covering the period between April 2016 and June 2016.

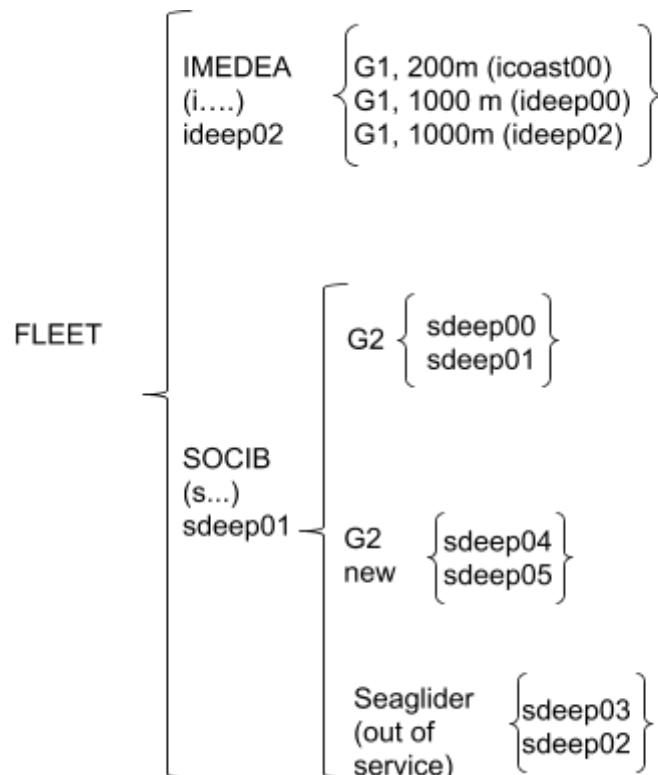


Figure 1. SOCIB Glider fleet.

1.2. Related Documents

- [Glider Toolbox Description](#)

1.3. Glider toolbox description

The SOCIB glider toolbox is composed of a set of MATLAB scripts and functions designed to manage the data collected by a glider fleet.

The main stages of the data management process are covered, including metadata aggregation, data download, advanced data processing and the generation of data products and figures.

Two main scripts are available to perform real-time and delayed-mode data processing, *main_glider_data_processing_rt* and *main_glider_data_processing_dt* respectively, and the other tools are controlled and called from these main scripts. The repository that contains the last version of Glider toolbox is in: *home/glider/glider_toolbox/m*.

The SOCIB glider toolbox is built to output three levels of netCDF files, related to different levels of processing:

- **Level 0 (L0):** contains exactly the same data as the raw files. The information is very useful for pilots.
- **Level 1 (L1):** contains processed glider data with unit conversions and filters applied. Additional variables are derived from the existing ones, such as salinity and potential temperature. L1 files store the glider data in a format ready for scientific use, building the bridge between glider mission raw data outputs and research or operational applications. The structure of these L1 files does not depend on the type of glider.
- **Level 2 (L2):** contains gridded glider data, which means that the glider data are interpolated onto a user configured grid in the vertical and stored as vertical profiles. The profiles are obtained by interpolation of level 1 data to produce regular homogeneous and instantaneous profiles from each up or down cast, using the mean time and position of the corresponding cast for the profile location and time. L2 files are convenient for plotting or analysis purposes, as they allow the user to work with any tool or software designed to deal with vertical profiles.

1.4. Database

It is important to mention that the **Database** used is SCB-datserve. Glider toolbox, either in real or delayed time, will check this database for the different deployments to be managed.

2. REQUIRED FEATURES

- Desktop or laptop.
- Internet connection.
- Glider User access.
- DataProc User access.
- pgAdmin software.

3. DELAYED TIME PROCEDURE DEVELOPMENT

The operator has to connect to **SCB-GENPROC** with the **glider** user:

```
ssh glider@SCB-GENPROC
```

If the operator is not in SOCIB / IMEDEA or is using a machine not connected to the SOCIB network, they can access **GENPROC** through the portal.

All the Glider post-mission raw data files and the Glider Toolbox application are contained as

it is shown in Figure 2.

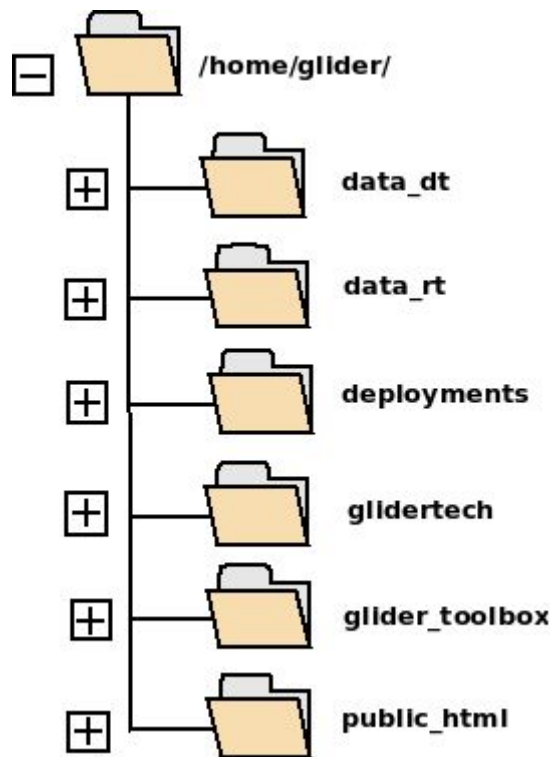


Figure 2. Directory structure for Gliders

4.1 PREPARE DATA

4.1.1. Check Post-Mission Raw Data Files

- Check that the Glider Technician properly stored the post-mission raw data in the **LOGS** and **SENTLOGS** directory contained in the following directory structure for both **nav** and **sci** data (Figure 3):

LOGS directories (one for **sci**, one for **nav**) should contain all the binary files corresponding to the data acquired by the glider.

SENTLOGS directories only contain the information that was sent through satellite communication.

Note: generally all the files of interest are only located in the **LOGS** directory. Nevertheless it is recommended to check for the files in the **SENTLOGS** folders.

Note: Glider Technician will send an email to Data Center when the post-mission raw data are in the proper directory (example of subject will be: FileSet - GF-MR-0059)

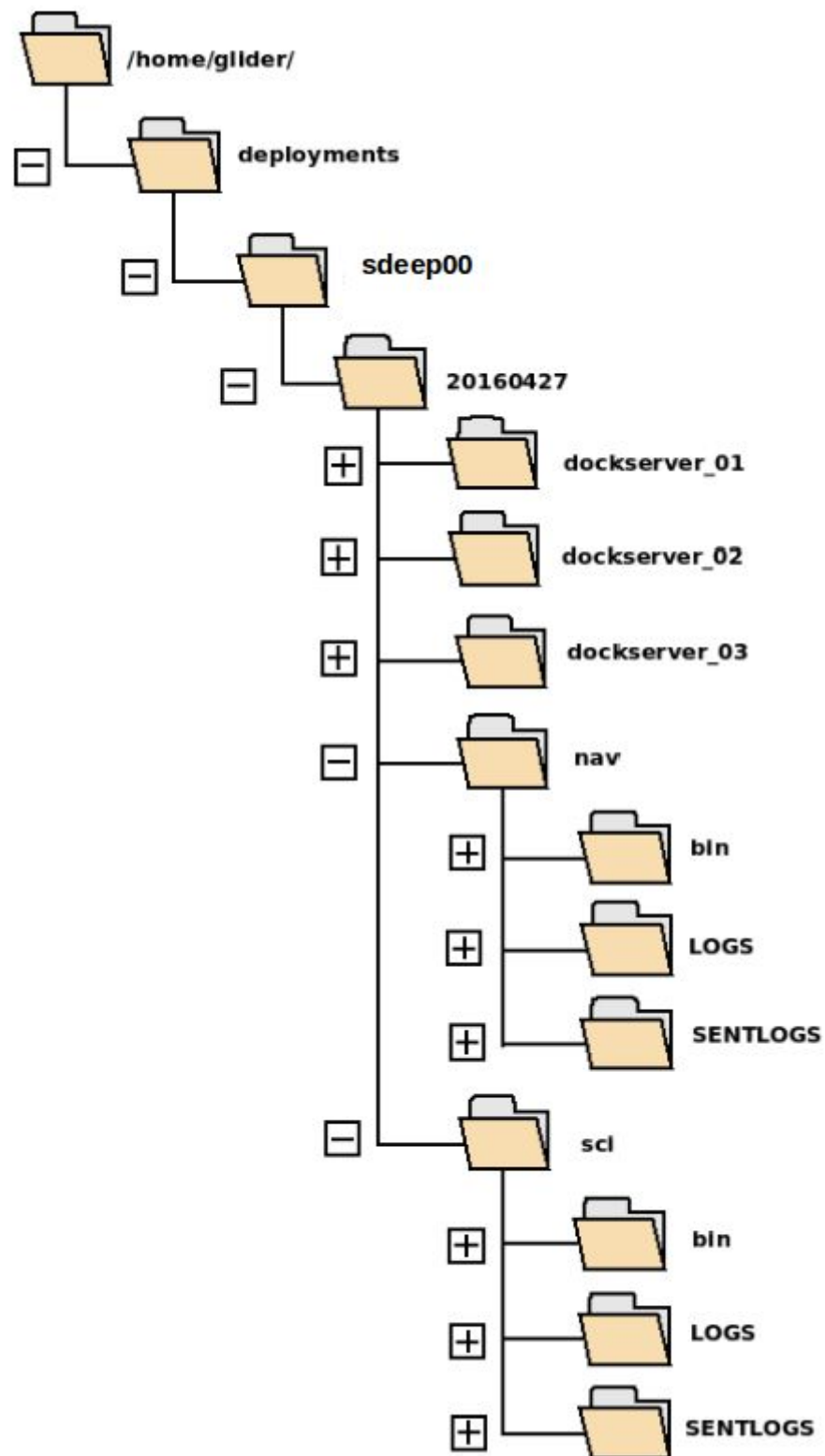


Figure 3. Directory structure for both nav and sci data

- There should be the following data formats for the **nav** directory: **DBD, MBD, SBD, MLG**.

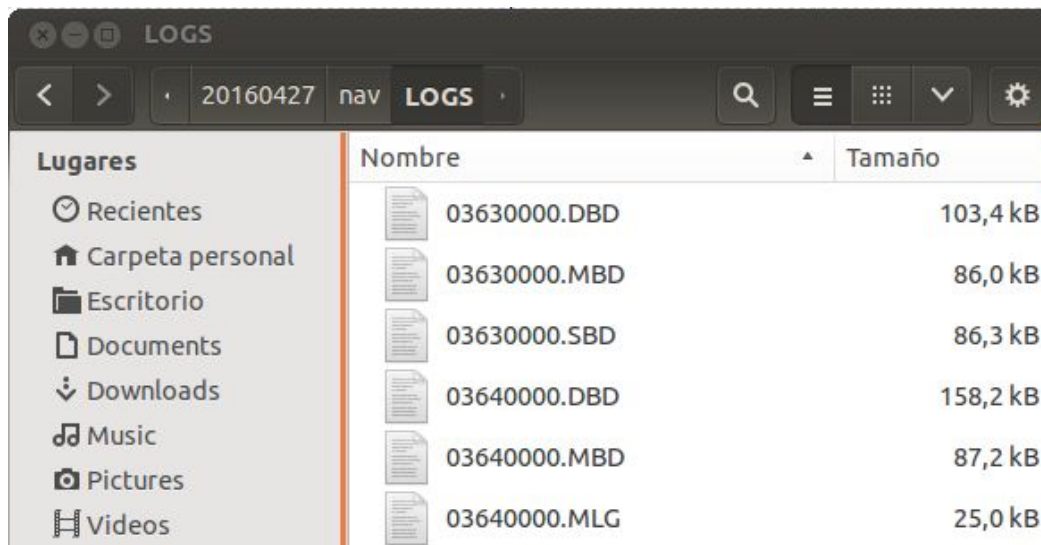


Figure 4. NAV directory.

- And the following formats for the **sci** directory: **TBD, EBD, NLG, NBD**.

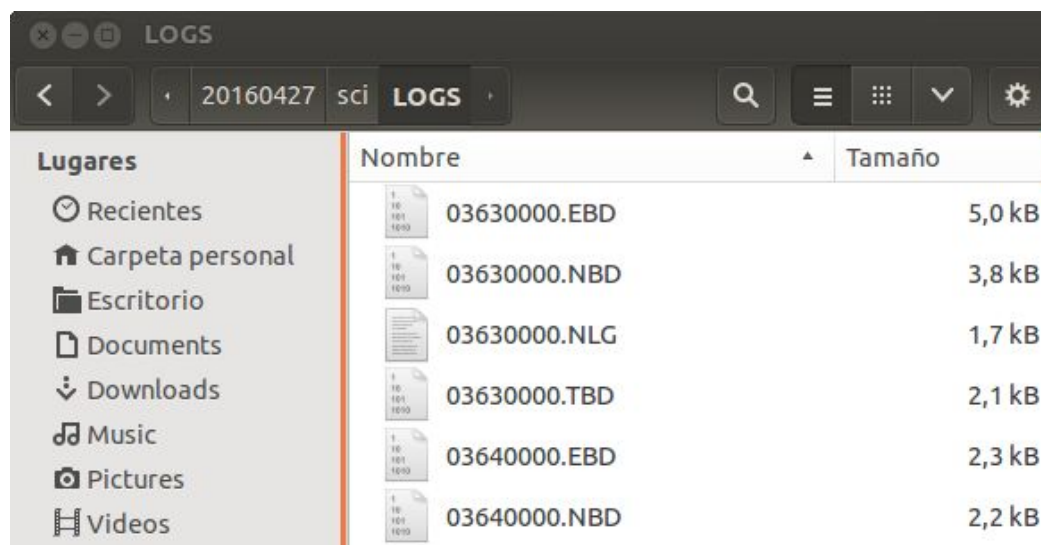


Figure 5: SCI directory.

Slocum glider deployments are organized missions, which themselves contain individual segments. The raw data files are named using the following 2 conventions:

- Glider-side:** Data files are named according to an 8.3 naming convention in which the first 4 numbers correspond to the sequential mission number and the last 4 numbers correspond to the sequential segment number of the current mission.

- **Shore-side:** The same data files, once transferred to shore, are typically renamed by the dockserver application according to the following convention:

gliderName_yyyy_ddd_mmm_sss.*bd

where *gliderName* is the name of the glider, *yyyy* is the current year, *ddd* is the (0-based) day of the year, *mmm* is the (0-based) mission number for the current day of the year and *sss* is the (0-based) segment number of the current mission.

4.1.2. Generate Data Processing Directories

1. Open a terminal (ctrl + Alt + T).
2. Log in using the following credentials (if not done before):
 - 2.1. Log in as “**glider**” user.
 - 2.2. Insert “**glider** password”

Note: Ask IT manager for the password.

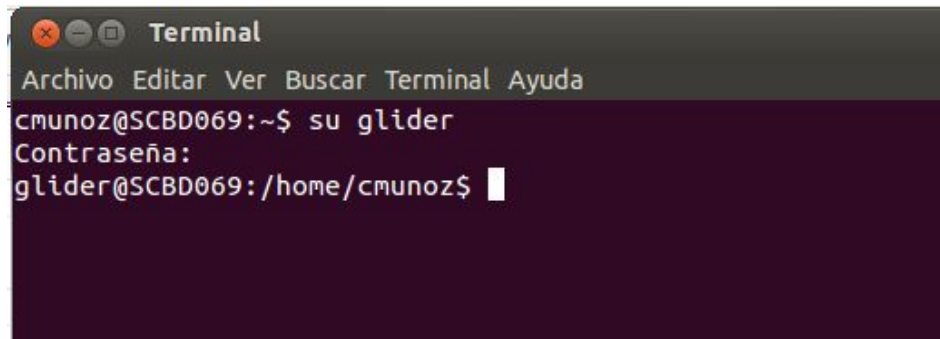


Figure 6. Log in as glider user.

3. Access to the data_dt directory:

```
cd /home/glider/data_dt/sdeep00
```

4. Create a new directory and name it with the same date *yyyymmdd* than the date directory containing the post-mission raw files:

```
mkdir yyyymmdd
```

with the obvious substitutions.

```

Terminal
Archivo  Editor  Ver  Buscar  Terminal  Ayuda
glider@SCBD069:/home/cmuno$ cd /home/glider/
glider@SCBD069:~$ ls
data_dt      glidertech      main_glider_data_processing_dt.log
data_rt      glider_toolbox  main_glider_data_processing_rt.log
deployments  LEGACY          public_html
glider@SCBD069:~$ cd data_dt
glider@SCBD069:~/data_dt$ ls
icoast00  ideep01  sdeep00  sdeep02  sdeep04
ideep00   ideep02  sdeep01  sdeep03
glider@SCBD069:~/data_dt$ cd sdeep00
glider@SCBD069:~/data_dt/sdeep00$ ls
20120228  20130520  20130909  20140206  20150618  20151211
20121127  20130715  20131101  20140407  20150819  20160427
20130130  20130802  20131202  20140610  20151019
glider@SCBD069:~/data_dt/sdeep00$ mkdir 20160427

```

Figure 7. Creation of new directories.

5. Get into the new directory and create another new directory and name binary:

`mkdir binary`

4.1.3. Establish Symbolic Link in Binary Files

- **From the new binary directory**, create a symbolic link pointing to the files stored in both **LOGS** and **SENTLOGS** (Usually only LOGS) directories, **nav** and **sci** contained in **data_dt**. Only link the *.DBD and *.EBD files:

`ln -s /home/glider/deployments/sdeep00/20160427/NAV/LOGS/*.DBD .`
`ln -s /home/glider/deployments/sdeep00/20160427/SCI/LOGS/*.EBD .`

Note:

1. before to create the links, check if the files exist. If by mistake you create a file named *.DBD, you can delete it using:

`rm *.DBD`

2. It's possible to do the previous operations in only one line using **find... exec**. But I don't write it here ;)

~

```

Terminal
Archivo Editar Ver Buscar Terminal Ayuda
glider@SCBD069:~$ cd data_dt/
icoast00/ ideep01/ sdeep00/ sdeep02/ sdeep04/
ideep00/ ideep02/ sdeep01/ sdeep03/
glider@SCBD069:~$ cd data_dt/sdeep00/20160427/
ascii/
binary/
dep0018_sdeep00_scb-sldeep000_2016-04-27_data_dt.log
figures/
netcdf/
glider@SCBD069:~$ cd data_dt/sdeep00/20160427/binary/
glider@SCBD069:~/data_dt/sdeep00/20160427/binary$ ln -s /home/glider/deployments
/s1243/20160427/nav/LOGS/*.DBD .

```

Figure 8. Symbolic links in binary files.

4.1.4. Rename Binary Files

- Go in the home directory of glider binary files

```
cd
```

- Rename all the symbolic links using the deployment information and date:

```

./local/bin/rename-dbd-files data_dt/glider/20160427/binary/*
path_to_binary_short_files path_to_rename_dbd_files/rename_dbd_files *.EBD
path_to_binary_short_files path_to_rename_dbd_files/rename_dbd_files *.DBD

```

Using this, files will be renamed in the format [glidername]-YYYY-###-###-###.ext (Figure 8).

Note: if there are empty files, they are not be renamed because the script looks for deployment information in the file header and cannot find it.

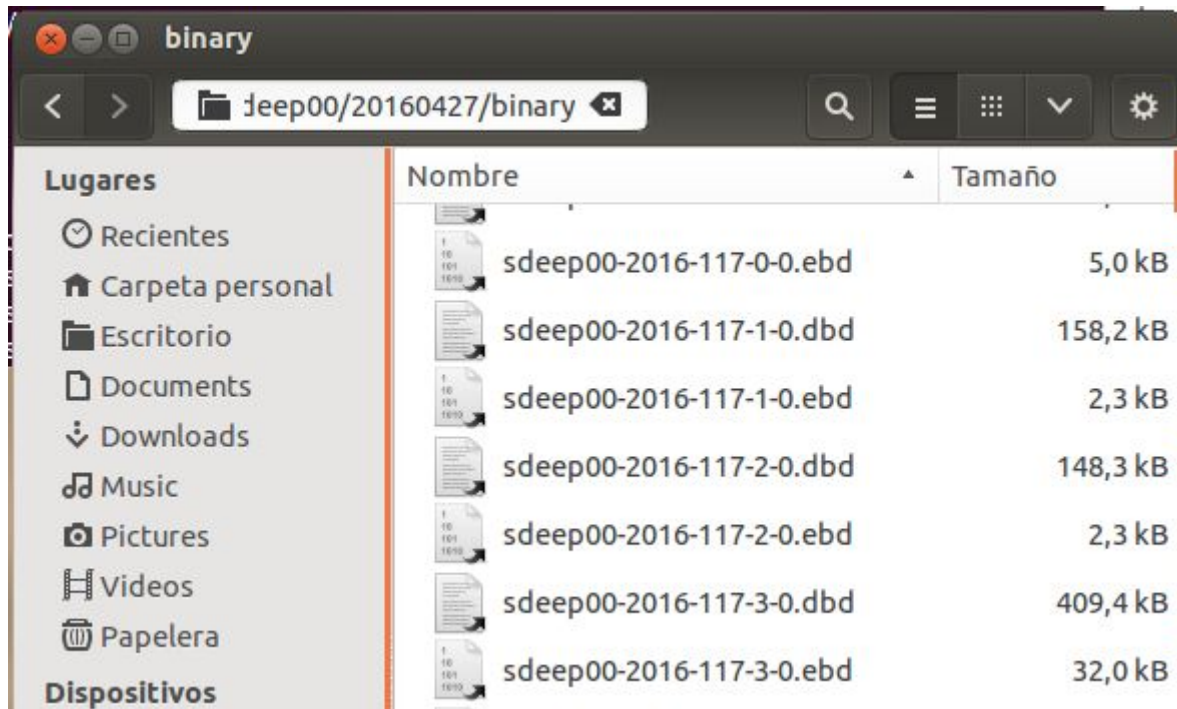


Figure 9. Shore-side convention name.

4.2. ADJUST GLIDER TOOLBOX SETTINGS

- Go to the toolbox directory:

```
cd /home/gliders/gliders_toolbox/m directory
```

4.2.1. Deployment ID

Edit the **configDTDeploymentInfoQueryDB.m** by changing the `deployment_ids` according to the deployment id that needs to be processed (Figures 10 and 11).

```
67 - deployment_ids = [598];
68 % Select the deployment fields.
69 % First column is deployment field
70 % Second column is column in data base table.
71 - fields_map = {
72     % Mandatory fields and fields required for paths.
73     'deployment_id'      'deployment_id'
74     'deployment_name'    'deployment_name'
75     'deployment_start'    'deployment_initial_date AT TIME ZONE ''UTC''
```

Figure 10. Changes deployment ID in configDTDeploymentInfoQueryDB.m

This number can be checked into apps.socib.es/instrumentation/ → Deployments → Check

the deployment info.

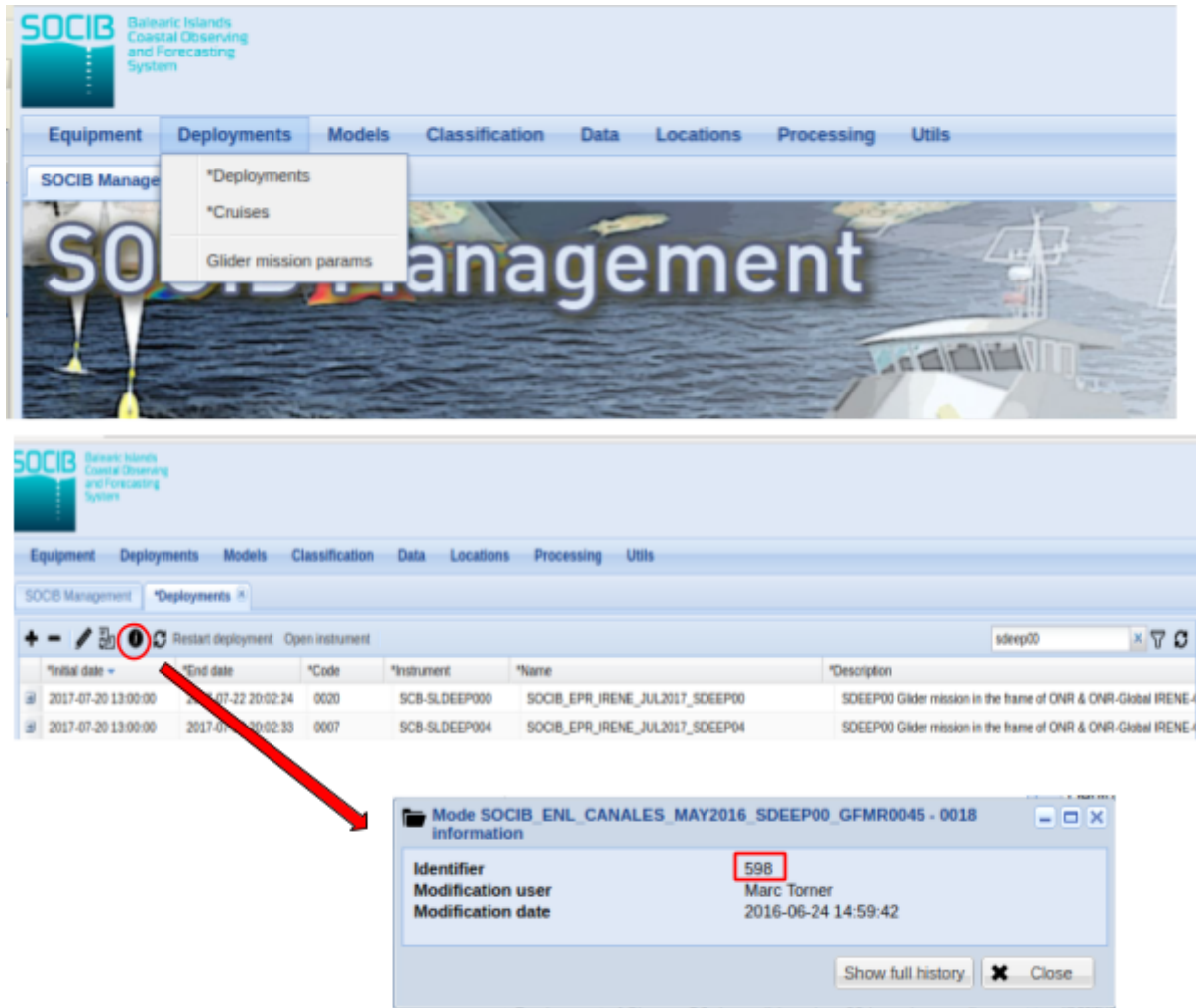


Figure 11. Deployment ID.

4.2.2. Binary files conversion

Check `configDTFileOptionsSlocum.m`:

```
66 % Binary file conversion to text format
67 % (disable it when reprocessing deployments with no new binary data):
68 slocum_options.format_conversion = true;
```

Figure 12. Binary file conversion option.

4.3. RUN THE MAIN PROCESSING

The file that will run the code is `main_glider_data_processing_dt.m`

You can either

1. Open a matlab session and type **main_glider_data_processing_dt** in the terminal
2. Run the tool from bash:

```
LANG=en_US.UTF-8      matlab      -nodisplay      -r      "try,      run  
/home/glider/glider_toolbox/m/main_glider_data_processing_dt.m,      exit(0),  
catch      exception,      disp(getReport(exception, 'extended')),      exit(1),      end" <  
/dev/null | tee main_glider_data_processing_dt.log
```

- LANG is used the [locale](#).
- [-nodisplay](#) makes matlab work without the desktop environment.
- [-r](#) indicates that what comes after will be executed by matlab
- [try ... catch](#) is used to execute a statement and catch the resulting errors
- [/dev/null](#) is a device file that discards all data written to it but reports that the write operation succeeded
- [<](#) is meant to redirects an input file-descriptor
- [tee](#) is to redirect output to multiple files.

4.4. CHECK OUTPUT NETCDF RESULTS

- Check that all the NetCDF files have been created in:
/data/current/opendap/observational/auv/glider
/home/glider/data_dt/{glider_name}/yyyymmdd/netcdf
- Check Thredds to ensure all data is properly displayed.



Figure 13. NetCDF files in Thredds.

4.5. CHECK OUTPUT FIGURES IN DAPP

Check that all the figures in DAPP have been updated from real time to delayed mode.

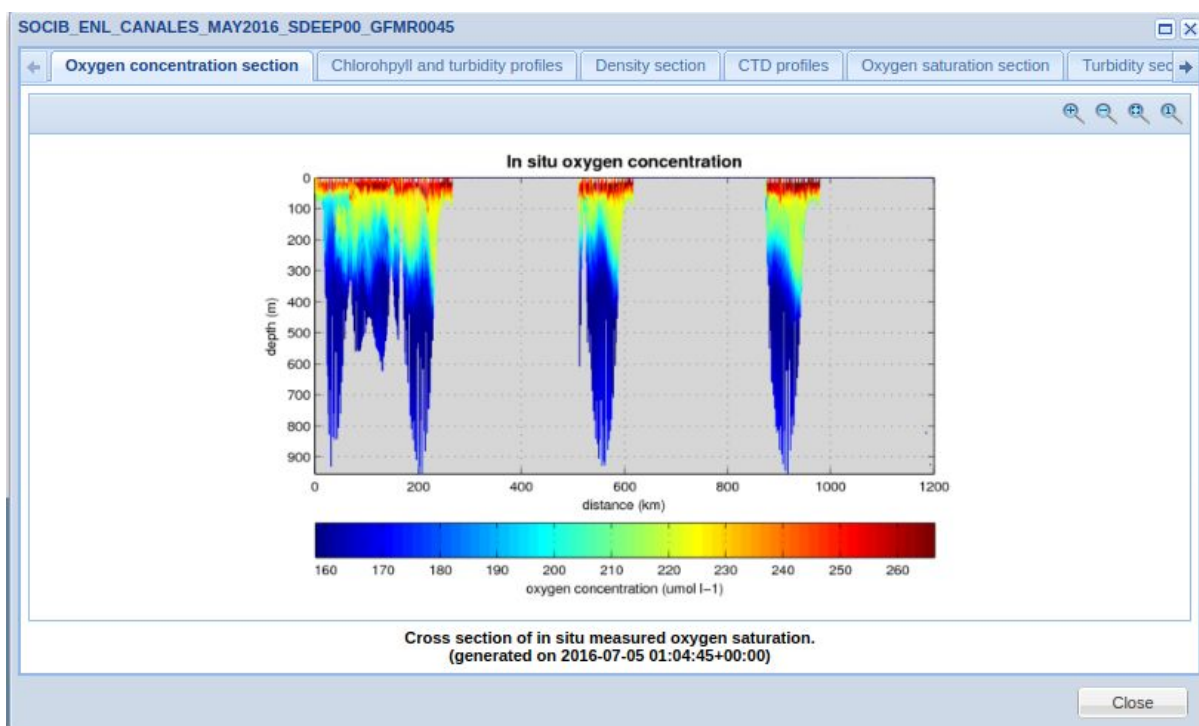


Figure 13. Public Figures generated by the Glider Toolbox.

4.6. SEND EMAIL TO GLIDER TECHNICIAN

Data Center has to send an email to Glider Technicians to confirm that delayed Mode postprocessing has been than.

DC will answer the mail that Glider Tech have sent before with the subject of FileSet - GF-MR-0059 (Following example in section 4.1, page 5).

5. TIPS AND TRICKS

5.1. Useful links

- How is working a glider?
<https://www.vistaalmar.es/hablame-del-mar/articulos/%C2%BFsabes-como-funciona-un-glider/>
- For Slocum Gliders, there is a web that give useful information about all the process involved:
<https://marine.rutgers.edu/~kerfoot/slocum/data/readme/>
- Slocum gliders: binary files header description
https://marine.rutgers.edu/~kerfoot/slocum/masterdata/source/masterdata_07_18.html

5.2. Delayed Time processing locally

5.2.1. Dependencies

In order to run it locally, some prerequisites/dependencies are needed:

- **SNC_Tools (mexcdf)**: SNCTOOLS is a collection of MATLAB codes that were written to provide read/write access to netCDF files. MEXNC is a mex-file interface to NetCDF files for MATLAB. Neither SNCTOOLS nor MEXNC are supported by the MathWorks.
- **mixing_library**: The Mixing Oceanographic Toolbox provides a framework to estimate the dissipation rate and diffusivity from Electromagnetic Autonomous Profiling Explorer (EM-APEX) float observational data.
- **m_map**: Mapping package for Matlab
- **seawater_ver3.31**: SEAWATER is a toolkit of MATLAB routines for calculating the properties of seawater
- **slocum**: Slocum software tools as provided by TeleDyne
- **postgresql-9.4.1212.jre6.jar**: object-relational database

5.2.2. Steps to run

1) Create the following structure (Figure 14) into home/user/projects/:

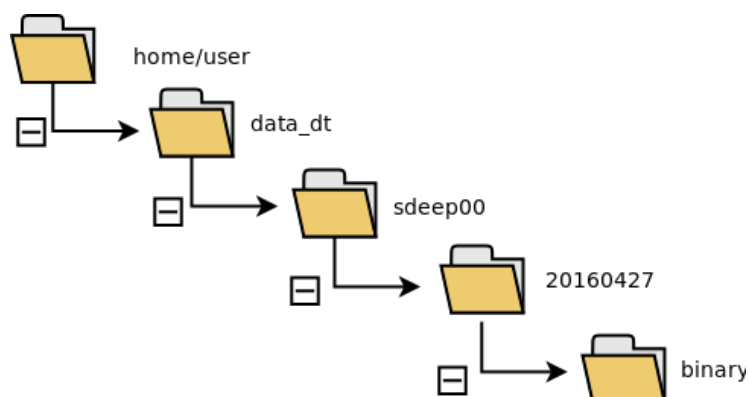


Figure 14. Structure to be created to run the Glider toolbox in Delayed Mode.

- 2) Establish symbolic link as it is explained in section 4.1.3.
- 3) Rename binary files as explained in section 4.1.4.
- 4) Copy glider toolbox into home/user/project.
- 5) Adjust the paths into section "Configure external programs path" in main_glider_data_processing_dt.m. For this purpose, edit the next scripts and change the paths.

- In *configDTPathsPublic.m*: Usually it should be like:
netcdf_basedir = '/data/current/opendap/observational/auv/glider';
figure_basedir = '/home/glider/public_html/dt';
- In *configDTPathsLocal.m*
base_dir = '/home/glider/data_dt/';

6) Configure the identifier of the deployment to process. Follow section 4.2.1.

7) Open a terminal and open Matlab with glider user:

```
sudo /LOCALDATA/MATLAB/R2012a/bin/matlab
```

Introduce glider password

8) When Matlab is opened, Set path, add folders with subfolders for the copied glider toolbox in 4)

9) If SOCIB data base is going to be used, In order to run in local computer, it has to be added to the Matlab Command Window and executed the next line:

```
javaaddpath('/opt/glider_toolbox_prerequisites/postgresql-9.4.1212.jre6.jar');
```

If this line is not executed, an error will appear "Unable to find JDBC driver".

10) Execute *main_glider_data_processing_dt.m*