- SOCIB Glider Toolbox Tutorial -

v1.0 - April 2017 - grogers@socib.es

Whilst these instructions were written for the Ubuntu operating system, the toolbox can be configured to run on Windows, however building the necessary MEX files and general configuration is simpler within linux.

The tutorial uses Slocum Gliders as an example, but SeaGlider and SeaExplorer Gliders are also supported.

Basic Requirements

Matlab R2010, Ubuntu 16.04, SOCIB Glider Toolbox, gcc, libssh-dev (for RT processing), Git

Toolbox Dependencies

SNC Tools M_Map Package Oceanographic Toolbox
Sea Water Tools For older versions of MATLAB (R2010) Narginchk.m

The Slocum software tools as provided by TeleDyne

You can download all the dependences zipped in one file here

Installation & Setup

(1) From within your home directory, run: git clone https://github.com/socib/glider_toolbox (alternatively download directly as ZIP and extract)

Open MATLAB and add the glider toolbox directory to the path

(2) Decompress the contents of the Toolbox Specific Dependencies into a single directory (e.g. /home/user/glider_toolbox/dependencies)

Add this directory to the MATLAB path

(3) Edit /glider_toolbox/configuration/configDTPathsLocal.m

Find & replace '/path/to/glider_data/' with where the glider data files (DBD,EBD,etc) are stored (e.g. /home/username/glider_data/)

(4) Edit /glider_toolbox/configuration/configWRCprograms.m

Find & replace '/path/to/bin' with where slocum software tools are located (e.g. /home/user/glider_toolbox/dependencies/slocum/bin)

(5) Open and run the following M files:

/glider_toolbox/setupmex/setupMexPosixtime.m
/glider_toolbox/setupmex/setupMexSFTP.m (only necessary for RT processing)

You will need the linux packages gcc & libssh-dev and the MATLAB mex compiler installed

(6) Edit /glider_toolbox/database_tools/getDeploymentInfoScript.m

This script file needs to be populated with the various metadata variables associated with the deployment that is to be processed

The SOCIB Glider Toolbox is now ready for use!

A number of points to bear in mind:

Prior to processing, the Binary Glider files (DBD,EBD,etc) should be ran with the Slocum file renamer tool "rename_dbd_files" before processing

On newer versions of MATLAB, warnings may be given about the function NARGCHK. To disable these warnings run: warning('off','all')

A great deal of the processing time (specifically for Slocum data) is the conversion from Binary to Ascii format. Once this stage is ran once, it can be disabled for future DT processing by modifying configDTFileOptionsSlocum.m and changing the value of slocum_options.format_conversion from true to false

Running the toolbox with supplied sample data

Extract the contents of /glider_toolbox/documentation/glider_sample_data.zip into the location previously specified for glider data. (e.g. /home/username/glider_data/)

The contents of /glider_toolbox/database_tools/getDeploymentInfoScript.m is already configured to run this deployment.

Assuming all the previous steps have been followed, then the delayed time process is ready to be executed on this sample data with NetCDF files (L0, L1, L2) and figures exported to the glider data directory.

Adding Extra Sensors

Frequently it is desirable to add output from additional glider sensors that are not yet configured in the toolbox. A number of script files in the processing chain require modification to achieve this.

We take as an example, the additional of a Slocum G2 based triple channel scattering sensor to be added for delayed (DT) processing. This sensor outputs scalar data of optical scattering at 470nm, 532nm, and 660nm.

(1) L0 output - Sensor Metadata

The following files need to be populated with a description of the new variables to be added including information such as long name description, standard name and units.

./configuration/configDTOutputNetCDFL0Slocum.m

(2) Binary to ASCII Conversion

This will enable the new sensors to be added for the binary to ASCII conversion. Note the naming convention is case sensitive to allow only lower case letters; any uppercase letters will result in the sensor being ignored.

Edit configDTFileOptionsSlocum.m, and add the following sensor names to slocum_options.dba_sensors:

‘sci_bb3slo_b470_scaled’;
‘sci_bb3slo_b532_scaled’;
‘sci_bb3slo_b660_scaled’;

(3) L1 Output - Preprocessing

This stage will enable the sensors to be added to preprocessing data thus incorporating into the L1 data product.

Open ./configuration/configDataPreprocessingSlocum.m and add at the end of the file:

```
preprocessing_options.extra_sensor_list.bb3slo(1).scattering_470_scaled =
‘sci_bb3slo_b470_scaled’;
preprocessing_options.extra_sensor_list.bb3slo(1).scattering_532_scaled =
‘sci_bb3slo_b532_scaled’;
preprocessing_options.extra_sensor_list.bb3slo(1).scattering_660_scaled =
```

'sci_bb3slo_b660_scaled';

(4) L1 Figure output

To create L1 data figures for these extra sensors, open the file for editing:
./configuration/configFigures.m

In order to visualize these data over the distance travelled, we will want to generate a Transect Vertical section. Find a block of plotting code beginning:

```
figures_proc.temperature = struct();
...
figures_proc.temperature.prntopts.comment = 'Cross section of in situ measured
temperature.';
```

Copy this whole block and paste it in the same section (Do this once for each sensor you want to visualize). Find and replace 'temperature' with 'scattering_470_scaled'. Do this for the whole block. Ensure that the fields:

.cdata .clabel .title .filename .comment

All point towards the sensor you want to generate visualizations for. Repeat this process for any other sensors you wish to plot.

(5) L2 Output - Gridded files

This stage will add the sensor output to the gridded L2 data product.
Edit ./configuration/configDataGridding.m

Add the following sensors to the gridding_options.variable_list:

'scattering_470_scaled'
'scattering_532_scaled'
'scattering_660_scaled'

The addition of new sensors to the toolbox is now complete!