Glider Toolbox

version 1.3.0

SOCIB DATA CENTER FACILITY

Quick Start

SOCIB Balearic Islands Coastal Observing and Forecasting System

GOVERN ILLES BALEARS

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

The glider toolbox (GTB) is a set of MATLAB/Octave scripts and functions developed at SOCIB to manage the data collected by a glider fleet. They cover the main stages of the data management and data processing both in real time and delayed time mode.

1. Requirements

- Linux or Windows OS
- Matlab 2012, Matlab 2014 or Octave.
- C++ compiler
- Imagemagick
- External libraries
  - mexnc and snctools (Commit #4053):
    NetCDF library preferences (Sourfoge/mexcdf). Octave users may install octcdf instead of snctools.
  - General Polygon Clipper library (GPC library) by Alan Murta
  - m_map 1.4 library: Mapping toolbox (m_map1.4.tar.gz)
    - GSHHS high-resolution coastline database v2.0 (download and follow instructions)
  - seawater 3.3 library: CSIRO Sea Water Library (seawater_ver3_3.1.zip)
  - m2html: inline html documentation generator
- Webb Research package for Slocum database
  (github: kerfoot/slocum for linux or coolcloud.mote/slocum for windows)
  - dba2asc: converts Slocum binary data to ascii format
  - dba_merge: merge ascii navigation and science files
  - dba_sensor_filter and dba2_time_filter: filter data based on sensor or timestamps
  - dba2_orig_matlab: converts ascii Slocum files to Matlab/Octave data files
- Matlab jar libraries:
  - Data base drivers 9.4 (postgresql-9.4.1212.jre6.jar). MATLAB interface with the PostgreSQL JDBC (Java Database Connectivity) driver. An Octave alternative would be needed. It may be installed from the distribution repositories as a package libpg-java (renamed to libpostgresql-jdbc- java in Debian repositories on March 2012).

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

- NetCDF 4.2 library (netcdfAll-4.2.jar from Unidata)

2. Installation

- Clone repository at https://github.com/socib/glider_toolbox.
  This creates a directory glider_toolbox with the following structure:
  - bin: binary scripts supporting the GTB
  - config: contains configuration files.
  - ext_lib: contains external libraries and binaries.
  - glider_data: contains files for local processing when using default configuration
  - m: contains Matlab/Octave code
  - documentation: contains schematics and manuals
- Download and uncompress external libraries, binary files and matlab jars. Users can run the gtb_install_extlibs under the bin folder. This script downloads and uncompresses these files automatically to the right locations.
  - External libraries to ext_lib/lib resulting in the following folders under ext_lib/lib: m2html, m_map, mexcdf/mexnc, mexcdf/snctools, seawater.
  - GPC library under m/mex_tools/gpcl (gpc.c and gpc.h). Notice that the installation script downloads the files under ext_lib and creates a soft link to m/mex_tools.
  - Webb Research binaries to ext_lib/bin
- Optionally, you can add the GSHHS high-resolution coastline database. The installation script included this database in the installation.
- Download jar libraries and configure Matlab. Users that run the installation script can find the jar files under ext_lib/matlab
  - Copy jar files to $MATLAB_ROOT/java/jarext
  - Add path to $MATLAB_ROOT/toolbox/local/classpath.txt
    - $matlabroot/java/jarext/netcdfAll-4.2.jar
    - $matlabroot/java/jarext/postgresql-9.4.1212.jre6.jar
- Setup Mex libraries from Matlab. Notice that GPC library must be added before this setup. Run matlab glider_toolbox and call the following commands.

```
>> cd m
>> mex -setup  (select 0 to answer prompt question)
>> configGliderToolboxPath
>> setupMexPosixtime
>> setupMexSFTP
>> exit
```


Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

- Create GTB library documentation: Run **make doc** from the glider_toolbox. Additionally, an online version of the of the documentation is available online.
- Edit configuration files for real time (config/configMainRT.txt) and delayed mode (config/configMainDT.txt).
  - Local and public paths
  - Database information
  - Dockserver

3. Processing

a. Real time Script

There are two ways to process glider data for each data mode (real or delayed time). First, users may process glider data by running the main_glider_data_processing_***X***t scripts where ***X*** defines "r" or "d" for real or delayed time respectively. These scripts use the configMain***X***R.txt configuration files to overwrite the default configurations set by the Matlab functions. The default configuration of the configuration files is enough to start processing except for the dockserver and/or database information. If necessary, users must complete the information of their database and dockservers before start processing. There are no edition required to the Matlab software as in previous versions of the GTB. In addition, users can call the gliderDataProcessing or the deploymentDataProcessing functions to run processing of glider data. The input variables of these functions include the information of the data sets, configuration and processing mode.

b. Delayed time Script

The main_glider_data_processing_rt script executes the realtime processing. The script uses most of the default configuration but custom configuration may be set by editing configMainRT.txt. The usual custom configuration are in the sample file from the repository. It includes the paths for the local and public locations, the database information access and the dockserver connection information. Notice that the processing_mode is defined in the configuration file but this value should remain to rt indicating real time processing mode.

c. Processing Function

The GTB recently includes a set of processing functions that allows its users to input custom information for specific runs. The first function deploymentDataProcessing takes the information of the deployment, the paths to the data, the configuration definition and the processing mode which are used to process a specific and single deployment. Deployment information may be input

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

directly into the function as explained in the Deployment Metadata section. A configuration file or structure can be input to define the processing configuration. Another difference with respect to the processing scripts is that users can choose what products are made by setting the appropriate paths in the input data_paths. For more information refer to the documentation of the code.

The second processing function is gliderDataProcessing. It wraps deploymentDataProcessing and it allows managing datasets from multiple deployments. This function is specially useful if a metadata database is set with the information of the deployments. However, users can also defined a list of deployments by inserting an deployment_list array with the list of deployments or using the deployment file. See the Deployment Metadata section for more information about how to define deployment information.

4. Configuration files

Previous versions of the GTB required editing Matlab configuration functions to customize the configuration to the needs of the users. Currently, the approach of the GTB differs in the sense that the configuration files are set to a default configuration that allows the user to process data in a basic mode without the need to edit any code. For a more complex setup, users add or edit text files that are read by the processing scripts and functions to overwrite the default values. With this approach, different types of data processing are performed by running it using different configuration files. Configuration files may live in any place in the disk but it is recommended to use the config folder in the installation path for consistency. The configMainRT.txt and configMainDT.txt are used by the real and delayed time mode processing scripts. Custom files may be created in the config folder with no arbitrary name format. No parameter is required to be defined in the configuration file except for the database and dockserver information which are not set to any default value. Any parameter that is present in the configuration file will be used for the processing over default or input variables of the functions. The available parameters are defined in the configTemplate.txt file that also describe the parameters using commented lines. Users can use the “#” to write comments in their own configuration files.

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

5. Deployment Metadata

Metadata of the deployment may be input in three different ways. By setting to 1 the active parameter of the db_access configuration, users can retrieve the information from their database server. They must setup the server and user information in the db_access configuration. Additionally, deployment information can be input by using deployment configuration files. The approach for the deployment files is similar to the one of the configuration files. However, all the information must be set since there are no default values as for the configuration. Users can use the "#" to write comments in their own configuration files. Lastly, the deployment information can be input to the processing function as matlab structures. For all of the options, the fields that must be defined by the deployment are in the table 1.

Structure field	Database field
deployment_id	deployment_id
deployment_name	deployment_name
deployment_start	deployment_initial_date
deployment_end	deployment_end_date
glider_name	platform_name
glider_serial	instrument_serial
glider_model	instrument_model

Table1 - Deployment information

Additional fields may be described to overwrite the values of the global attributes of the NetCDF metadata. These fields must be named identically as the attribute to be filled.

a. Metadata Database

The GTB provides tools to retrieve the deployment metadata from a database. By default, a table (named deployment) in a database (named instrumentation) is query to retrieve the keywords that match the fields of the deployment structure as described by table 1. Potentially, users can use any database, table and field names. However, this is not allowed yet using the configuration files. Therefore, they

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

are required for this purpose to edit the configDTDeploymentInfoQueryDB and/or configRTDeploymentInfoQueryDB accordingly.

b. Metadata Deployment Structure

The deployment metadata may be input to the gliderDataProcessing function as an array of structures using the deployment_list option or as a structure to the deploymentDataProcessing function. Processing scripts do not provide this option. The required fields of the structure are deployment_id, deployment_name, deployment_start, deployment_end, glider_name, glider_serial and glider_model. Additionally, other fields may be used to overwrite values of the global attributes of the NetCDF metadata.

c. Metadata Deployment Text Files

Deployment information can also be read from a configuration file. The two processing scripts (main_glider_data_processing_dt and main_glider_data_processing_rt) read deployments from the config/deploymentDT.txt and config/deploymetnRT.txt files (respectively) if the db_access.active parameter is set to 0. Any custom file defining the deployments may be read by the gliderDataProcessing function when db_access.active is 0. Users should input the file name when calling the function. The deployment file should contain the information of the deployment (same as the structure) and any optional field to overwrite global attributes of the NetCDF metadata. The deployment file can contain any amount of deployments by using the index approach. Below we present an example of the definition of two deployments:

```
deployment_list(1).deployment_id           = 9
deployment_list(1).deployment_name         = NOV2017_TERESA_GFMR0064
deployment_list(1).deployment_initial_date = 7.3700e+05
deployment_list(1).deployment_end_date     = NaN
deployment_list(1).platform_name           = teresa
deployment_list(1).instrument_serial       = 518
deployment_list(1).instrument_model        = Slocum G2 Deep

deployment_list(2).deployment_id           = 10
deployment_list(2).deployment_name         = NOV2017_SDEEP04_GFMR0065
deployment_list(2).deployment_initial_date = 7.3700e+05
deployment_list(2).deployment_end_date     = NaN
deployment_list(2).platform_name           = sdeep04
deployment_list(2).instrument_serial       = 520
deployment_list(2).instrument_model        = Slocum G2 Deep
deployment_list(2).abstract                = Deployment for scientific tests
```


Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79

deployment_list(2).author = John Smith

Parc Bit. Ctra. Valldemossa, Km 7,4 · Edifici Naorte, Bloc A, Planta 2ª, Porta 3 · 07121 Palma (Illes Balears, Espanya) · Tel. +34 971 43 99 98 · Fax +34 971 43 99 79