Glider Toolbox Quick Start

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.

# Requirements

* Linux or Windows OS
* Matlab 2012, Matlab 2014 or Octave.
* External libraries
  + mexnc and snctools (Commit #4053):   
    NetCDF library preferences ( [Sourfoge/mexcdf](https://sourceforge.net/projects/mexcdf/files/mexcdf/mexcdf.r4053.zip/download) )
  + m\_map 1.4 library: Mapping toolbox ([m\_map1.4.tar.gz](http://www.eos.ubc.ca/~rich/m_map1.4.tar.gz))
  + seawater 3.3 library: CSIRO Sea Water Library ([seawater\_ver3\_3.1.zip](http://www.marine.csiro.au/cgi-bin/marlin-dl/seawater/seawater_ver3_3.1.zip))
  + m2html: inline html documentation generator
* Webb Research package for Slocum database   
  (github: [kerfoot/slocum](https://github.com/kerfoot/slocum/tree/master/linux-bin) for linux or [coolcloud.mote/slocum](http://coolcloud.mote.org/glider_data/slocum_tools/windows/) 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](https://jdbc.postgresql.org/download/postgresql-9.4.1212.jar)).
  + NetCDF 4.2 library ([netcdfAll-4.2.jar](ftp://ftp.unidata.ucar.edu/pub/netcdf-java/v4.2/netcdfAll-4.2.jar) from Unidata)

# Installation

* Clone repository at <https://github.com/socib/glider_toolbox>.   
  This creates a directory glider\_toolbox with the following structure:
  + 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
  + notes: contains schematics and manuals
* Download external libraries and uncompress to ext\_lib/lib resulting in the following folders under ext\_lib/lib: m2html, m\_map, mexcdf/mexnc, mexcdf/snctools, seawater.
* Download Webb Research binaries to ext\_lib/bin
* Download and configure Matlab jar libraries:
  + 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. Run matlab glider\_toolbox and call the following commands.  
     >> cd m  
     >> mex -setup  
     >> configGliderToolboxPath  
     >> setupMexPosixtime  
     >> setupMexSFTP  
     >> exit
  + Create GTB library documentation: Run **make doc** from the glider\_toolbox. Additionally, an online version of the of the documentation is available [online](http://www.socib.es/users/glider/glider_toolbox/doc).
  + Edit configuration files for real time (config/configMainRT.txt) and delayed mode (config/configMainDT.txt).
    - Local and public paths
    - Database information
    - Dockserver

# Processing

There are two ways to process glider data for each data mode (real or delayed time). First, the processing can be run using the main\_glider\_data\_processing\_Xt script where X defines “r” or “d” for real or delayed time respectively. These scripts use the configMainXR.txt configuration files to overwrite the default configurations set by the config Matlab functions. Otherwise, there is little to define before running an automatic configuration as explained below. However, using the scripts forces the users to setup the dockserver and/or databases in order to successfully running them.

The second option uses the gliderDataProcessing or the deploymentDataProcessing functions to run processing based on the input variables which define the data sets, configuration and processing mode. In this running mode, these functions allow users to input the deployment and dataset information removing the need to setup a database.

## Real time script

1. The [main\_glider\_data\_processing\_rt](http://www.socib.es/users/glider/glider_toolbox/doc/m/main_glider_data_processing_rt) 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 initial 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.

## Delayed time script

1. The [main\_glider\_data\_processing\_dt](http://www.socib.es/users/glider/glider_toolbox/doc/m/main_glider_data_processing_dt) executes the delayed time processing. The script uses most of the default configuration but custom configuration may be set by editing configMainDT.txt. The usual custom configuration are in the initial file from the repository. It includes the paths for the local and public locations, the database information access and the format conversion option. Notice that the processing\_mode is defined in the configuration file but this value should remain to dt indicating delayed time processing mode. In this version of the main\_glider\_data\_processing\_dt script the indexes of the deployment ids are not edited in the matlab configuration file as it was required in previous versions. It must be added to the configMainDT.txt configuration file as db\_access.deployment\_ids with deployment numbers separated by the “|” symbol.

## Processing functions

The GTB recently includes a set of processing functions that allow users to input custom information for specific runs. The first function [deploymentDataProcessing](http://www.socib.es/users/glider/glider_toolbox/doc/m/processing_tools/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. In that way, deployments can be processed by knowing their information. 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.

The second function is [gliderDataProcessing](http://www.socib.es/users/glider/glider_toolbox/doc/m/processing_tools/gliderDataProcessing). It wraps deploymentDataProcessing and allows to manage datasets from multiple deployments. This function is specially useful if a metadata database is set with the information of the active and closed deployments as defined by the documentation. However, users can also defined a list of deployments by inserting an deployment\_list array with the list of deployments. As for deploymentDataProcessing, the configuration definition may be input as a structure or a filename.

# 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 the 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 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. 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. 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.

# Metadata Database

The GTB provides tools to retrieve metadata from a database that is used during the processing. The processing requires a structure with the fields shown below. By default, a table named deployment in a database called instrumentation is query to retrieve the keywords that match the fields of the deployment structure as follow:

|  |  |
| --- | --- |
| **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 |

Potentially, users can use any database, table and field names. However, this is not allowed yet using the configuration files. Therefore, they should edit the configDTDeploymentInfoQueryDB and/or configRTDeploymentInfoQueryDB accordingly at their own risk.