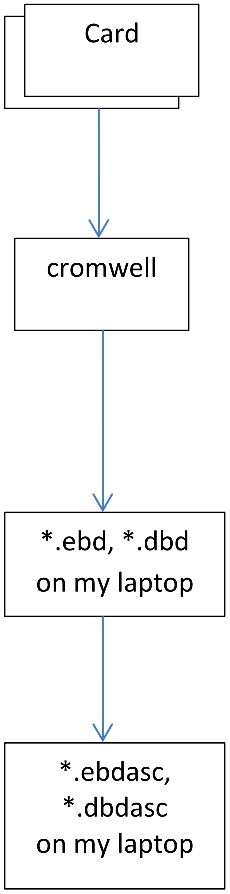
Dinkum binary to ascii (L0)

This section documents the flow from the original *.ebd, *.dbd to the ascii versions of the files (for both flight and science).

Flow Diagram



Save flight and science files under /seacoos/data

tar,compress
scp from cromwell to my laptop
uncompress, extract tar

Rename files using **rename_dbd_files**
After copying cache files to cache folder, run **dbd2asc**

…or instead of renaming (which introduces problem with order),
Run scripts **dbd2asc_Driver, ebd2asc_Driver** to process an entire set

Dependencies

The dependencies can be seen by looking through the code for the L0_Driver. This includes the following code directories:

```
% Addpaths
addpath('../gliderproc/MATLAB/util');
addpath('../gliderproc/MATLAB/matutil');
addpath('../gliderproc/MATLAB/plots');
addpath('../gliderproc/MATLAB/strfun');
```

as well as folders for the iinput (binary) data files and output (asci) data files. These will be custom, per your environment.

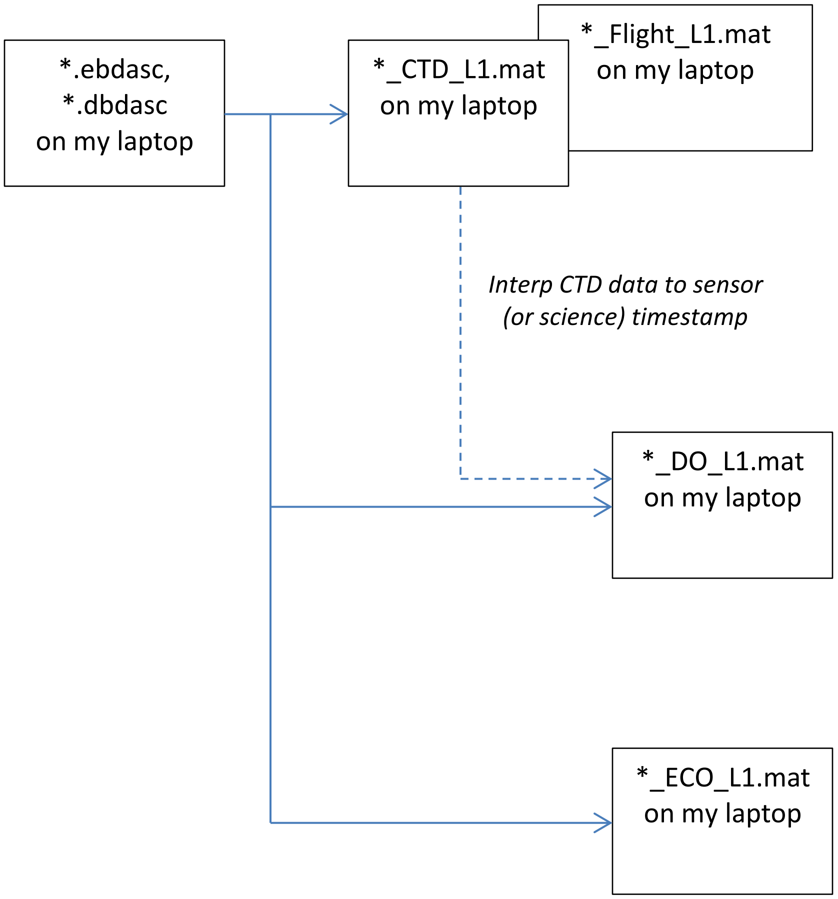
Note that the MATLAB folder has been zipped in the github repository, so you will need to unzip it.

Ascii to matlab (L1, L2)

This section documents the flow from the ascii to the matlab structures. In general, the L1 has corrections e.g.:

- correction for salinity spiking for CTD
- TSP correction, lag correction, and cal (fix) for DO.

Flow Diagram



Run **gliderCTD_Generate_L1_Data_Driver_???** where ??? is a label for a project.

Run **gliderOptode_Generate_L1_Data_Driver_???**.

Run **gliderECO_Generate_L1_Data_Driver_???**.

Dependencies

The dependencies can be seen by looking through the code for the L1_Driver, including:

- Addpaths
- Data directories (ebddir, dbddir)

The addpaths are as follows

```
% add paths for required files...
addpath('../gliderproc');
addpath('../gliderproc/MATLAB/util');
addpath('../gliderproc/MATLAB/plots');
%addpath('MATLAB/matutil/');
addpath('../gliderproc/MATLAB/seawater');
addpath('../gliderproc/MATLAB/strfun');
%addpath('MATLAB/opnml/');
%addpath('MATLAB/opnml/FEM/');
% Add this in only if you want to use the NEW correctThermalLag (and make
% sure the old one does not get on the path first e.g. rename it).
addpath('../../glider_toolbox-master/m/processing_tools');
```

Note that the last one can be obtained from https://github.com/socib/glider_toolbox, it contains the new code to correct salinity spiking i.e. correctThermalLag.

Calibration

CTD

Calibration coefficients for the CTD are applied on the glider, not in matlab code.

ECO Puck

Calibration for ECO Puck is applied on the glider, using values in autoexec.mi.

Dissolved Oxygen

For LongBay_2012, calibration was performed at SkIO. The calibration was applied in **gliderOptode_Generate_L1_Data**, which is called by **gliderOptode_Generate_L1_Data_Driver_XXX.**

(For LongBay_2012, we had to revisit the calibration due to an offset between the internal temperature and the CTD. For SkIO calibration, see emails from Charles Robertson, where some content has been saved to Docs/from_Charles_Robertson. For calibration fix, see the document Docs/L2_DissolvedOxygen_Corrections_20150817.pdf.)

Corrections

CTD

To reduce salinity spiking, we use the new correctThermalLag function—the one that supports pumped CTDs. This is called in **gliderCTD_Generate_L1_Data**, which is called by **gliderCTD_Generate_L1_Data_Driver_???.** The correction parameters are in the driver **gliderCTD_Generate_L1_Data_Driver_???.**

ECO Puck

NA

Dissolved Oxygen

TSP corrections are applied in **gliderOptode_Generate_L1_Data**, which is called by **gliderOptode_Generate_L1_Data_Driver_???.**

Summary of code changes

This section summarizes the code changes that were made, relative to the code from Long Bay 2012.

1. The code was restructured so that glider- and deployment-specific code/settings were segregated. This makes it easier to reuse the core code for other projects (e.g. SECOORA_2016 and then PEACH).
2. For the CTD processing, we switched to a newer version of correctThermalLag (to reduce salinity spiking). This code now supports pumped CTDs.
3. One thing we'd like to do is to optimize the sample rate for the CTD. As a result, we may have a different sample rate per sensor. Therefore, the code was changed to support the capture and use of a timestamp per sensor [1](instead of relying upon the "science timestamp"). Per sensor (CTD, DO, ECO), one can now choose to use the science timestamp or the actual sensor's timestamp.

[1] On TWR forum, at https://datahost.webbresearch.com/viewtopic.php?f=4&t=154&p=430&hilit=timestamp#p430 TWR says that the most accurate timestamp is the sensor's timestamp.

Location/History of code

Code	Folder on my laptop (under Projects)	Where did I get it?
Rename_dbd_files	Glider\DataMgmt\Code\TWR	https://datahost.webbresearch.com/files.php?cwd=/glider/production/windoze-bin
dbd2asc	Glider\DataMgmt\Code\TWR	https://datahost.webbresearch.com/files.php?cwd=/glider/production/windoze-bin
dbd2asc_Driver ebd2asc_Driver	Glider\DataMgmt\Code\forPEACH\L0_Drivers	New
gliderCTD_Generate_L1_Data	Glider\DataMgmt\Code\forPEACH\gliderproc	Modified previous program (from Long Bay), in Glider\DataMgmt\fromHarveyAndSara~\gliderproc
gliderOptode_Generate_L1_Data	Glider\DataMgmt\Code\forPEACH\gliderproc	Modified previous program (from Long Bay), which is in my folder C:\Users\slockhar\Projects\LongBay\External\from_Chris_Calloway\gliderproc_modBySBL\gliderproc\glider_DO_20160119
gliderECO_Generate_L1_Data	Glider\DataMgmt\Code\forPEACH\gliderproc	Started with gliderCTD_Generate_L1_Data and then made changes, as I could not find the code that was used in Long Bay.
glider*_Generate_L1_Data drivers	Glider\DataMgmt\Code\forPEACH\L1_Drivers	

Issues

Short Descrip	Long Descrip	Resolution
Missing flbbcd proc	Where is the gliderproc code that generated the L1 data for chlorophyll, scatter, CDOM for Long Bay?	Chris Calloway does not have it, so I created one.
Calibration	Make sure calibration of DO takes into account the temperature offset and provides enough data in case we need to revisit the calibration curves.	
Clocks	Sample interval (science or CTD) can be negative when clock gets resynchronized. (It's usually positive. For SECOORA, there was only one negative value.)	By processing the files in the correct order, the issue is now visible. Hopefully, it doesn't happen frequently.
Clocks	CTD time can be zero (possibly at or near surface?).	Eliminate these samples, but know how many there are.
Clocks	CTD sample interval can be zero (or very close to it).	Eliminate these samples, but know how many there are.
Bounds	Each L1 dataset (mat file) has bounds as attributes of the data struct; however, the bounds are only applied in the DO dataset. Should we remove these attributes if they are not used?	