RSKtools for Matlab access to RBR data

Table of Contents

Clark Richards, PhD; clark.richards@rbr-global.com; RSKtools version 1.4; Updated 2016-04-05

Introduction

RBR instruments output data in an open database format known as SQLite. To facilitate direct access to the data in Matlab™, we created the `RSKtools` toolbox. `RSKtools` facilitates direct access to the data stored in `RSK` files by using the included `mksqlite` library, for which we have provided versions compiled for Windows (32/64bit), Linux (64bit) and Mac OSX (64bit). It may be necessary to compile your own version, using the source code provided at http://sourceforge.net/projects/mksqlite/.

`RSKtools` also provides some convenience functions for common data extraction (e.g. extracting profiles from a continuous dataset) and visualization (plotting individual profiles). For plans for future additions, see the Future plans section.

Installing

The latest stable version of `RSKtools` can be found at http://www.rbr-global.com/support/matlab-tools.

- Unzip the archive (to `~/matlab/RSKtools`, for instance)
- Add the folder to your path inside matlab (`addpath ~/matlab/RSKtools` or some nifty GUI thing)
- type `help RSKtools` to get an overview and take a look at the examples.

Examples of use

To work with an RSK file using `RSKtools`, a connection to the database must be made. This is done using the `RSKopen()` function. Note that `RSKopen` doesn't actually read the data, but reads a /thumbnail/ of the data which is typically about 4000 points long. The structure returned after opening an RSK looks something like:

```
file = '../testfiles/065583_20140612_0739.rsk';
rsk = RSKopen(file)
```

```
rsk =

              dbInfo: [1x1 struct]
            datasets: [1x1 struct]
    datasetDeployments: [1x1 struct]
        calibrations: [5x1 struct]
         instruments: [1x1 struct]
    instrumentChannels: [5x1 struct]
            ranging: [3x1 struct]
     instrumentSensors: []
            channels: [3x1 struct]
              epochs: [1x1 struct]
           schedules: [1x1 struct]
         appSettings: [1x1 struct]
         deployments: [1x1 struct]
       thumbnailData: [1x1 struct]
            profiles: [1x1 struct]
```

Note the structure element called `thumbnailData`. In order to read the actual data, we use the `RSKreaddata()` function, which if given with one argument (the variable name of the RSK object) will read the entire data set. Because RSK files can store a large amount of data, it may be preferable to read a subset of the data, specified using a start and end time (in Matlab `datenum` format, which is defined as the number of days since January 0, 0000).

```
t1 = rsk.thumbnailData.tstamp(1) + 0.5; % half a day after start
t2 = rsk.thumbnailData.tstamp(1) + 1.5; % 1.5 days after start
rsk = RSKreaddata(rsk, t1, t2);
```

Note that the data structure can be found in the object at

```
rsk.data


ans =

      tstamp: [442293x1 double]
      values: [442293x4 double]
    longName: {'Conductivity'  'Temperature'  'Pressure'  'Salinity'}
       units: {'mS/cm'  '°C'  'dbar'  'PSU'}
```

In this example, because the instrument was determined to be a "CTD"-type instrument, a new channel was created called `Salinity` (using the Practical Salinity Scale). The salinity calculation is performed by the TEOS-10]] package, which can be obtained from http://teos-10.org/software.htm.

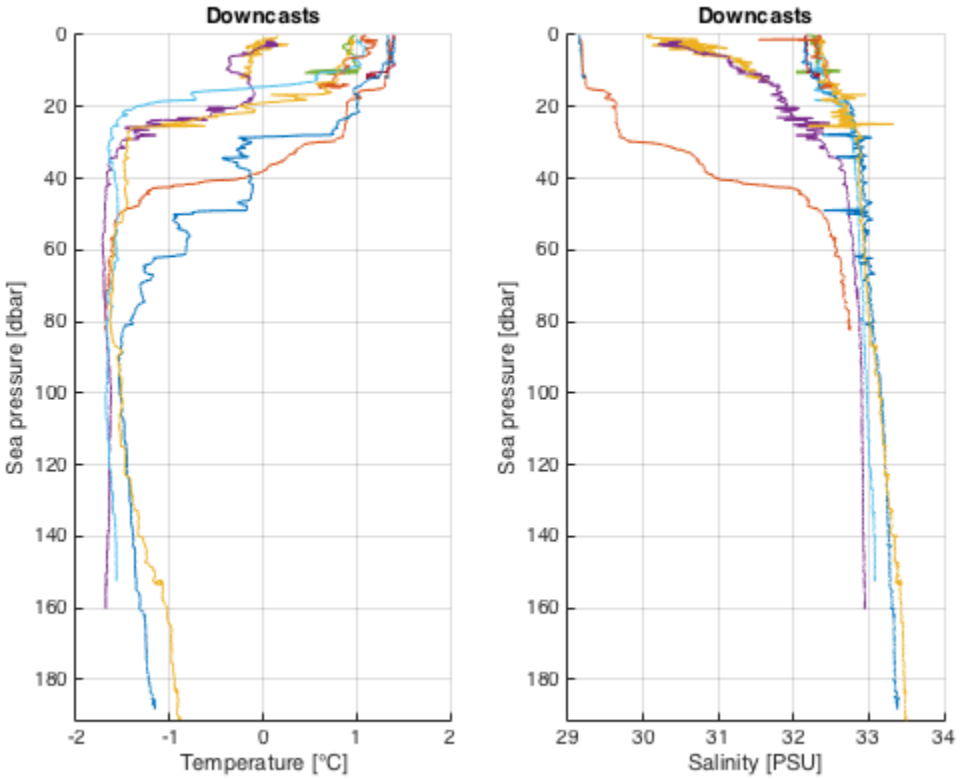
Working with profiles

Profiling loggers with recent versions of firmware contain the ability to automatically detect and log profile "events". These are denoted as "downcasts" and "upcasts", and the function `RSKreadprofiles()` can be used to extract individual profiles from the raw data, based on the previously identified events. Following this, quick plots of the profiles can be made using the `RSKplotprofiles()` function.

```
file = '../testfiles/065583_20140612_0739.rsk';
rsk = RSKopen(file);
```

```
% load the first 10 profiles
rsk = RSKreadprofiles(rsk, 1:10);

% plot the downcasts
subplot(121)
RSKplotprofiles(rsk, [], 'temperature', 'down')
subplot(122)
RSKplotprofiles(rsk, [], 'salinity', 'down')
```



Future plans

- Cast detection for datasets without profile events
- Wave processing functions
- Improved data processing functions (e.g. for CTD data)

About this document

This document was created using Matlab™ Markup Publishing. To publish it as an HTML page, run the command:

```
publish('RSKtools_vignette.m');
```

See `help publish` for more document export options.

Published with MATLAB® R2015a