DFO “Team Whale” Beaked Whale Detection Procedure

Wilfried Beslin, May 2019

Based on original document by Joy Stanistreet, Nov. 2018

Table of Contents

[1. INTRODUCTION 2](#_Toc9948417)

[2. WALKTHROUGH 2](#_Toc9948418)

[2.1 File Setup 2](#_Toc9948419)

[2.2. General Click Detection using Triton 2](#_Toc9948420)

[2.2.1 Batch Short Time Spectrum 2](#_Toc9948421)

[2.2.2 Batch High Res Clicks 3](#_Toc9948422)

[2.3. Automatic Event Detection 3](#_Toc9948423)

[2.4. Event Validation and Species Identification 4](#_Toc9948424)

[2.5. Post-Processing 5](#_Toc9948425)

# 1. INTRODUCTION

The purpose of this document is to provide a quick, step-by-step walkthrough of our process for detecting beaked whale events from acoustic recordings, using Triton and our in-house beaked whale detection code (BWD). For a comprehensive description of each feature in BWD, refer to the Reference Manual.

# 2. WALKTHROUGH

## 2.1 File Setup

The first steps in preparing to analyze a new dataset should be to do the following:

* Create a folder to store all WAV files to be analyzed. Move all WAV files within the period of interest to that folder.
* Create an output folder for the dataset to be analyzed. This can have any name you want, but best practice is to name it based on the deployment name (e.g. *MidGul2015*).
* Within this output folder, create another folder called *metadata*. This will store the output produced by Triton (.c and .cTg files).

## 2.2. General Click Detection using Triton

Initial click detection with Triton is done in two parts: “Batch Short-Time Spectrum”, which will find general locations of clicks, and “Batch High-Res Clicks”, which will find specific start and end times of clicks based on Teager-Kaiser energy estimation.

To run Triton, open MATLAB and type “triton” in the command window

* Note: if this does not work, make sure the Triton code folder is on the MATLAB path

### 2.2.1 Batch Short Time Spectrum

* Select *Tools 🡪 Detection 🡪 Batch Short Time Spectrum (STS)*
* In *Short Time Spectrum Detection Parameters*, select *load*, then enter the path to the *Triton\_detParams\_8dB.spec.prm* file (located in the BWD folder)
* In *Metadata*, enter the path to the *metadata* folder you created in the output directory
* In *Files*, click *Specific files*, then set *Base Folder* to the path containing your WAV files (do not check box to include subfolders)
* A complete list of WAV files should appear – this may take a few seconds to show up
* Hit large *Okay* button on bottom left
* Hit large *Okay* button on bottom left of next window (*Associate labels with files*)

Triton will now detect echolocation clicks and generate .c files with rough time stamps within the *metadata* folder.

### 2.2.2 Batch High Res Clicks

* Select *Tools 🡪 Detection 🡪 Batch High Res Click (STS Guided)*
* Click *Specific files*, then set *Base Folder* to the path containing your WAV files (do not check box to include subfolders)
* A complete list of WAV files should appear – this may take a few seconds to show up
* Hit large *Okay* button on bottom left
* On next window, set *Output Folder* to your *metadata* path, then hit *Okay*

This step runs the Teager-Kaiser energy detector based on the rough time stamps from the initial detections and produces .ctg files with precise start/end times of clicks.

## 2.3. Automatic Event Detection

* If Triton is open, close it by selecting *File* *🡪 Exit* in the *Control* window
* Open the script *BWD\_master.m* in the MATLAB editor (you can do this from the command window by typing *edit BWD\_master*; if this does not work (i.e. it creates a blank page), then the BWD directory is not on the MATLAB path)
* Edit the lines where prompted to set input parameters, according to the following table (Reference Manual **Table 4.6**):

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| depName | Name of the deployment/dataset |
| dirPath\_analysis | Full path to your output directory |
| dirPath\_audio | Full path to your WAV file directory |
| detProtocol | Name of the detection protocol you want to use (one of the folders within *DetectionCriteria*) |
| dirName\_detResults | Name of the results folder |
| segDur | This parameter controls segmentation of audio files for continuous data. Must be specified as a MATLAB duration object, which uses the syntax *duration(h,m,s)*.   For example, to divide a recording into 1-minute segments, type *duration(0,1,0)*.  To use recording durations as segments, type *duration(Inf,Inf,Inf)*. |
| nfft | Number of points to use for FFT |
| nMATClicksMax | Maximum number of clicks that can be stored in an output MAT file. If a recording has more clicks than this, it will be divided into multiple MAT files. |

* Run the script

This will create a results directory in your analysis folder, which will contain one or more Excel spreadsheets labelled “RawEvents”. The names and number of sheets depends on the detection protocol you selected.

## 2.4. Event Validation and Species Identification

* Open the script *identifySpecies\_master.m* in the MATLAB editor (again, you should be able to do this by typing *edit identifySpecies\_master* in the command window)
* Edit the lines where prompted to set input parameters, according to the following table (Reference Manual **Table 4.7**):

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| dirPath\_analysis | Full path to your output directory |
| dirName\_detResults | Name of the results folder |
| targetName | Name of target to analyze. There must be a *RawEvents* and/or *Validated* spreadsheet containing this target name. |
| dirPath\_analysis | Full path to your output directory.  Make sure this is consistent with what you entered in *BWD\_master*. |
| eventMergeOpt | A string that controls which definition to use for the start and end time of a detection event. Options are as follows:  ‘none’ – use raw segment durations as the basis for event durations.  ‘timegap’ – segments with positive detections that occur within a certain time period of one another will be merged into single large events  ‘calendar’ – use fixed calendar durations as the basis for event durations. A given calendar period will be considered a detection event if it contains at least one segment with beaked whale detections. |
| eventMergVal | Complementary parameter to *eventMergeOpt*. It represents different things depending on what you entered for *eventMergeOpt*:  If *eventMergeOpt* = ‘none’, then this parameter is not used.  If *eventMergeOpt* = ‘timegap’, then this parameter is a duration object representing the minimum separation that two segments with positive detections may have to be considered separate events.  If *eventMergeOpt* = ‘calendar’, then this parameter is a string specifying the calendar duration. Valid options are the same as for the *unit* argument of MATLAB’s *dateshift* function; this includes ‘hour’, ‘day’, ‘week’, ’month’, and ‘year’. |
| nClicksMax | Maximum number of clicks to include for event validation. Set to *Inf* to disable. |

* Run the script

This will start the interactive event validation/species identification application. On first run, this will produce a “Validated” Excel spreadsheet from the “RawEvents” sheet. The application will update this sheet directly as you work through it.

Among other options, the interactive application will let you to enter a code for each event specifying which species you think are present. For guidance on how to properly enter codes and how to use the application in general, consult the Reference Manual (section 3. Manual Event Validation and Species Identification).

## 2.5. Post-Processing

After the “Validated” spreadsheet has been filled out, there are a few more operations you can do:

* Use the script *subsetValidated.m* to isolate events of interest and create duplicates of the “Validated” sheet containing those events only. This can be useful for going back to unknowns.
* Use the script *createPresenceTable.m* to convert numeric species ID codes into presence/absence scores

These are discussed further in the Reference Manual (section 4.4 Additional Scripts).