**USV Detection Guide V 1.0**  
**Updated 8/17/14  
David J. Barker**

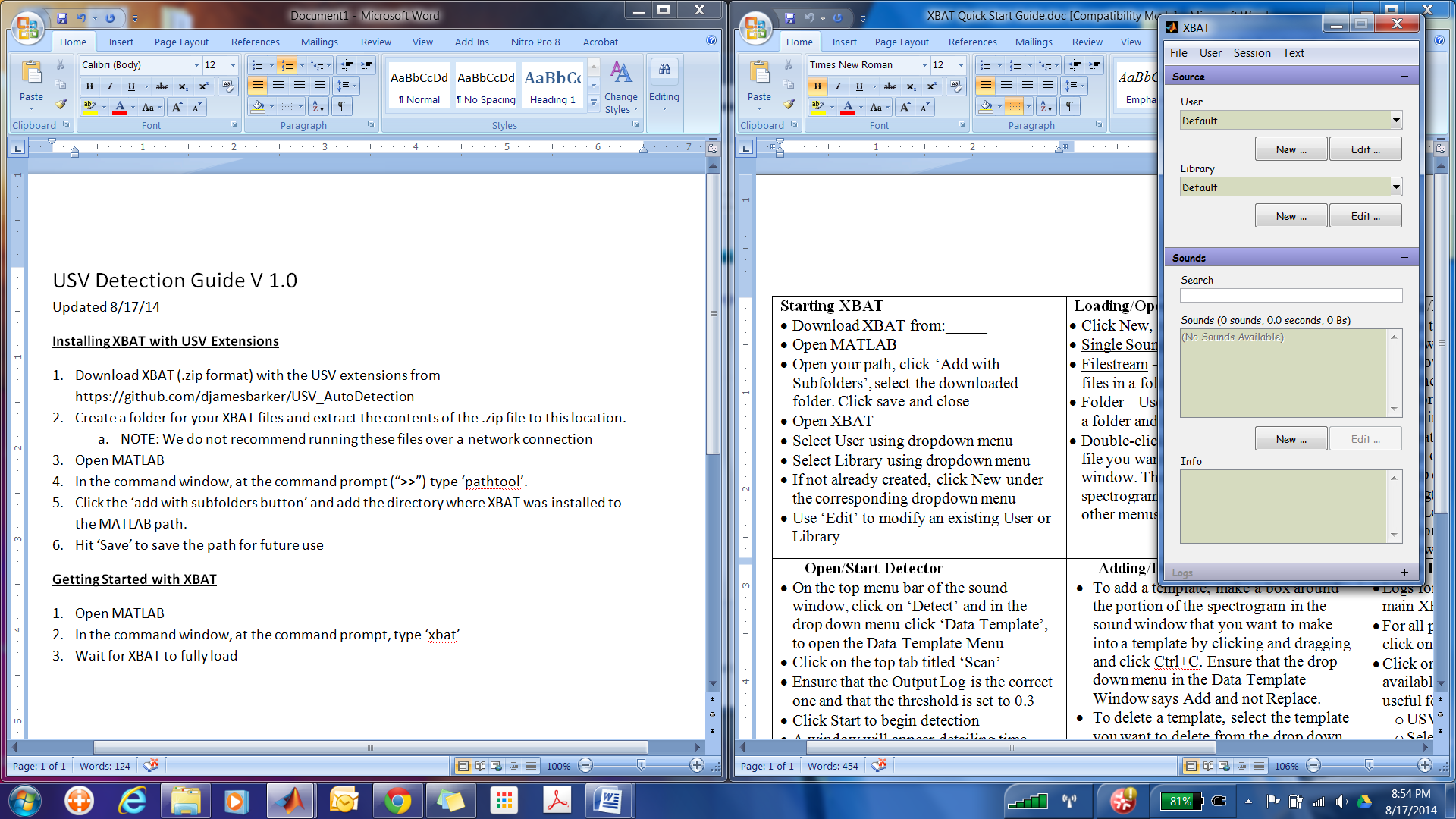
**Installing XBAT with USV Extensions:**

1. Download XBAT (.zip format) with the USV extensions from https://github.com/djamesbarker/USV\_AutoDetection
2. On the C: Drive, create a folder named ‘Matlab’ with a subfolder named ‘XBAT\_R7’. Next, extract the XBAT files from github into this folder.
   1. NOTE: Github may create additional subfolders when extracting named ‘USV\_Autodetection-Master’. If this occurs, move the XBAT files and folders from under this directory and paste them directly into the ‘XBAT\_R7’ folder.
   2. NOTE: You may be able to install XBAT to a directory of your choosing. However, if you have problems, please follow the directions above.
   3. NOTE: We do not recommend running these files over a network connection.
3. Open MATLAB
4. In the command window, at the command prompt (“>>”) type ‘pathtool’.
5. Click the ‘add with subfolders button’ and add the directory where XBAT was installed to the MATLAB path.
6. Hit ‘Save’ to save the path for future use

**Getting Started with XBAT:**

XBAT is an open-sourced extensible tool which runs in the MATLAB environment. Users that are familiar with MATLAB should have few issues navigating this tool while those with little MATLAB experience may encounter more difficulties. For this reason, an introductory [MATLAB book](http://amzn.to/1lcatpX) might prove helpful for novice users. All users should anticipate that XBAT will function somewhat less fluidly than professionally developed software and be patient with errors that are encountered during the learning process. Eventually, one will find that the ability to use the XBAT detection features and develop custom scripts for analysis allows for more rapid data analysis and does save time.

The introductory documentation for XBAT can be found [here](https://dl.dropboxusercontent.com/u/4142063/build/home.html)

1. Open MATLAB
2. In the command window, at the command prompt, type ‘xbat’
3. Wait for XBAT to fully load
   1. NOTE: MATLAB will often display an error stating that XBAT function names overlap with built in function names and suggests renaming them. This error can be ignored.
4. If desired, add a new user. Continuing under the default user and library is easiest for single-user computers.

**Loading a Sound File:**

1. In the main XBAT window (right), under the sound section, click “New...”
2. Open a sound in one of the following ways:
   1. “File”- Opens a single sound file
   2. “File Stream”- Opens multiple sound files and combines them into a single ‘Sound’
   3. “Folder”- Compiles all sound files from one directory (folder and subfolders) into a single ‘sound’.
   4. “Variable”- Access sound data stored as a MATLAB variable and store it as a ‘sound’ in XBAT.
3. Once the sound has been added, it should be visible in the main XBAT window.

**Opening a Sound:**

1. After loading a sound file, double click on the selection under the sound tab of the main XBAT window to open a spectrogram of the sound.
   1. Note:
      1. Parameters of the spectrogram such as the FFT length, color scale, brightness, contrast etc. can be changed by clicking the “View” menu in the spectrogram window.
      2. Tools related to the control of the spectrogram, logging of events, etc. can be found under the “Window” menu.
      3. A list of valuable keyboard shortcuts can be found under Help>> KeyBoard Shortcuts (Ctrl+K).

**Logging sound events:**

1. Open a sound file as described above
2. Create a new log (3 methods):
   1. Ctrl + N
   2. File>>New Log
   3. Use the GUI under Window>>Log and click “New...” under the “File” subsection
3. Management of logs is best accomplished by using the Log tool under Window>>Log. Here, you can open previously created logs, enable or disable the display of certain logs, and determine which log is ‘active’ when multiple logs are present.
4. Once a log has been established, events can be manually logged by using the mouse to click and draw a square around a given event.
5. Once a border has been drawn around the event, it can be committed to the current, active log by hitting the “L” key.
6. Auditory confirmation of vocalizations can be accomplished via playback by hitting “P” on the keyboard. “Shift + P” will play back the current spectrogram window in its entirety. Playback settings can be adjusted under the Window>>Play tool.
7. If desired, annotations can be added in association with each logged event by using the menu bar to navigate to Annotate>>Active>>Simple. This allows a user to input a text code that might be used for the identification of call types etc.
8. The data for a given log can be exported to the MATLAB workspace by selecting Log>>Workspace from the menu bar and selecting the name of the relevant log in the menu.
   1. **Note: Understanding the structure of the logs may take some time. Most analysis will be based on the use of these logs, so becoming familiar with MATLAB structure variables and the log format is critical to success with XBAT.**

**Automatic detection of 50-kHz USVs:**

1. Open a sound file as described above
2. From the menu bar select Detect>>Data Template
3. Select the Scan tab in the Data Template window (top)
4. Selected the correct output log under the Scan tab
5. In the Data Templates window, select Presets>> USV\_Template from the menu bar.
6. On the bottom, select the Correlation tab and ensure the threshold is set to 0.3.
   1. NOTE: These settings can be adjusted as desired, but optimization analysis has determined that correlations of ~0.3 work best for the USV template detection procedure
7. Click “Start” on the Scan tab to begin the scan.
8. Once the scan has finished, open the main XBAT window and select the log used for detection from within the log subsection. Right click on the log and click Actions>>USV\_Clean.
   1. NOTE: In order to best detect complex vocalizations (e.g., those with multiple steps) the detection procedure will tag each vocalization using multiple templates. The goal of the USV\_Clean action is to determine overlapping templates and combine these templates into one selection containing all parts of the USV.
9. A new window will pop up. Click OK and then save the cleaned version of the log. We recommend keeping the log from each stage by saving logs with the suffixes ‘Detected’, ‘Cleaned’ and ‘Reviewed’ (see below).
10. Once the log has been cleaned, it is ready for human review. To do this, right click the cleaned log in the main XBAT window and then select Actions>>Selection Review.
11. Adjust the settings if desired, otherwise the default settings should be sufficient for USV review. Once the settings have been adjusted click OK.
12. A second prompt should appear to ask the user about custom key bindings. Users have the option to accept USVs and reject artifacts using the ‘Y’ and ‘N’ keys or may also use custom keys to label specific call types by providing a list of custom keys and call labels. An example file showing importable key bindings can be found in the XBAT installation folder under the \Extensions\Actions\Log\Selection Review folder. This file includes the traditional USV subtypes: FF, FM, and Trill.
    1. **Note: The ‘Y’, ‘N’, ‘Esc’, and right and left arrow keys are permanently bound and should not be used.**
13. Users may now review the automatically detected events. Pressing ‘Y’ will accept an event and advance to the next detection, while the ‘N’ Key will reject and eventually remove a detection. Within our selection review window, users have the ability to change color maps, play back vocalizations, plot the position of the peak frequency/time of each call etc. Users may return to previous detections using the left arrow and may advance to subsequent detections using the right arrow. A counter at the upper left shows the users position within the group of detections and a window on the right displays the status of previously reviewed events (e.g., ‘Accept’, ‘Reject’, ‘FF’, ‘FM’, or ‘Trill’).
14. Once all USVs have been reviewed, the user may exit the selection review using either the ‘Esc’ key or by exiting the window using the ‘X’ in the upper right. The user will then be prompted to save a new log with the reviewed data. Note, calls that are rejected using the ‘N’ key will be removed from the log at this stage in time. For this reason, saving over previous logs is not recommended.

**NOTE:** While our auto detection method has been tested, improvements are still possible. Thus, users are encouraged to use auto detection as well as to manually score a subset of files for comparison (~10-20%). Early studies should include the inter-rater reliability between the detector and human scoring for 50-kHz USVs and the correlation between the number of USVs detected by human scorers and the detector.

**Log Actions:**

Along with the USV\_Clean and Selection Review procedures, other log actions have been created by our laboratory for use with XBAT. We also encourage other users to develop tools and share them within the USV community (see below). Users familiar with MATLAB might begin programming by using the log actions we have developed as templates.

1. Double Click on a sound file to open the spectrogram window
2. From the menu bar click EXT>>New>>Log Action
3. Fill in the relevant information in the popup
4. Close XBAT by typing ‘close all’ in the main MATLAB window at the command prompt
5. Re-open XBAT by typing ‘xbat’ at the command prompt. A new log action has now been created and will be present in the XBAT environment.
6. Right click on a log in the main XBAT window and click the name for your new action
7. In the popup window click EXT>>Edit>>Compute (not “Edit...”)
8. A MATLAB script window will now open. The compute function will be empty with the exception of two necessary lines of code.
9. Click cancel in the log action window to halt XBAT operations.
10. The first line of code calls information from the XBAT environment. To access this information, insert a breakpoint by typing ‘dbstop’ into the second line of the script.
11. Next, right click on a log in the log window and run your log action again.
12. If successful, the program will stop and you will find three inputs to the function in your MATLAB workspace. These variables contain information from the XBAT logs for automatic or manually scored detections.
13. At this point, the sky is the limit. Users can create their own GUI’s (see the Selection Review), data processing algorithms, etc. using the information contained in the logs. Please see the code from our laboratory for examples. You might also find reason to edit these codes for your own use.

*The following log actions, developed by our laboratory might prove useful for USV researchers:*

**AvisoftMarkerLabel-**Exports the starting and ending timestamp of each detection window as a text file that can be imported into Avisoft SASLab Pro.

**AvisoftTimeFreqLabel-** Exports the peak time and frequency into a text file. These data can be opened as markers in Avisoft SASLAB Pro.

**USV\_XBAT2Excel-**Exports a spreadsheet of data into a ‘.xls’ file. These data can be used in Microsoft Excel.

**USV\_Clean-** Described above. Combines multiple detections into a single detection.

**Selection Review-** Described above. Allows for user review of auto-detected events. This could also be used to determine inter-rater reliability among human scorers.

**Community sharing of detector progress:**

As the detector reaches more laboratories, improvements will be made and new analysis tools will be developed. Updates to the detector are welcome and can be submitted to the GitHub listed at the top of this document or by contacting Dr. David Barker via email. Submissions can come in any one of the following formats:

1. **USVs to be added as templates**- Some types of USVs may still be underrepresented by the current templates. If your testing suggests that a specific USV would make a valuable addition to the template library, please submit a (short) clip containing the USV with only a small buffer of ambient noise. Most clips will be ~1 second in length. Please also submit a brief description of the testing done to verify that this template is necessary.
2. **New Log Actions-** The analysis of USVs from the detector relies on log actions. We will continue to develop and share new tools for USV analysis and we encourage you to do the same. Those that are familiar with MATLAB programming may take it upon themselves to create new log actions for analysis. The MATLAB script for these log actions should be well-annotated and thoroughly tested. Tested scripts may then be submitted for inclusion with the GitHub download. Authors should also provide a readme file for how their tool can be used.
3. **New/Updated Template Libraries-** If more global improvements to the template library have been created, a whole new template library can be submitted for inclusion with the GitHub repository. Template libraries should include a description of changes/updates that have been made as well as a list of contributors and their respective institutional affiliations. New presets will be included in addition to the current library to allow for testing by other laboratories.
4. **Accuracy Reports-** We recognize that the accuracy of the detector may change depending on the recording circumstances. For this reason, we encourage all users to conduct assessments of ‘inter-rate reliability’ between automated detections and human scoring as well as to correlate results from human scoring with detector scoring. Indeed, for most applications, the detector does not need to be 100% accurate, but it does need to have good predictive validity. We encourage users to submit a report of these findings including the inter-rate reliability, correlation, notes about the recording conditions, etc. We will make every effort to make these findings publicly available in order to help improve detection.
5. **Other-** Anything not covered here that you believe should be shared with the field is welcome. Simply contact us about including your idea.