

# MERIDIAN Detection Viewer/Validator

Reference Manual

Wilfried Beslin

Oct. 28, 2021

## Table of Contents

1. INTRODUCTION .....	2
2. PROGRAM FILES .....	2
3. COMMANDS FOR RUNNING THE PROGRAM .....	2
4. INPUT .....	3
4.1 Autodetections Spreadsheet File.....	3
4.2 Folder of WAV Files.....	3
4.3 Parameters .....	4
<i>Spectrogram Settings</i> .....	5
<i>Plot Parameters</i> .....	5
<i>Marker Parameters</i> .....	7
5. OUTPUT.....	7
6. USAGE .....	9
6.1 Keyboard Inputs.....	10
6.2 Button Inputs.....	11
<i>SPECTROGRAM panel</i> .....	11
<i>DETECTION panel</i> .....	11
<i>PLAYBACK panel</i> .....	11
<i>VALIDATION panel</i> .....	11
<i>COMMENT panel</i> .....	11
<i>MISSED CALLS panel</i> .....	11
<i>ANNOTATIONS panel</i> .....	12
7. USING WITH LFDCS.....	12
8. REFERENCES .....	15

## 1. INTRODUCTION

This program started as a simple spectrogram viewing tool to inspect right whale upcall detections returned by the neural network detector being developed by MERIDIAN at Dalhousie University. Subsequently, it was developed into a validation tool emulating the *browse\_autodetections* module from Mark Baumgartner's LFDCS ([http://dcs.who.edu/lfdfs\\_manual/lfdfs\\_manual.html](http://dcs.who.edu/lfdfs_manual/lfdfs_manual.html)). Since then, it has grown to support various new features as well, including the marking of missed calls and general annotations within files where autodetections are present. A conversion script was also added to enable detections from LFDCS to be validated using this program. Inspiration for this program was taken from both LFDCS and JASCO's PAMlab.

This manual applies to version 20211028. This version requires at least MATLAB R2018b to run. Later releases of MATLAB may also work but this has not been tested - there is a risk that some functionality may break.

## 2. PROGRAM FILES

To use the program, you must have a folder in your MATLAB path with at least the following files:

- **validateMeridianDetections.m** – main program file
- **LFDCS2MERIDIAN.m** – script file for converting LFDCS autodetection result CSV files into XLSX files compatible with the validation program (see *USING WITH LFDCS*)
- **BrowserResources** folder containing the following:
  - **OutputTemplate.xlsx** – template file for the output spreadsheet (see *OUTPUT*).
- **PARAMS** folder containing one or more parameter files (see *INPUT > Parameters*)
  - The file *default\_params.txt* must at least be present here
- **+Classes** and **+Utilities** folders containing additional code files required for the main program to run



Unless you wish to modify the code, do not change the names or contents of any of these files or folders, except where otherwise noted. Doing so could cause the program to stop working correctly.

## 3. COMMANDS FOR RUNNING THE PROGRAM

The MATLAB command to run this program is its filename:

```
validateMeridianDetections
```

You can run the program in the simplest way by just typing this in the Command Window (provided all the above files are on your MATLAB path). This will start up the program with default settings.

However, it is possible to specify certain input arguments in the command line as well. The syntax for that is the following:

```
validateMeridianDetections('Name1',Value1, 'Name2',Value2, ...)
```

Where *Name1* is the name of one argument that is assigned *Value1*, *Name2* is the name of a second argument assigned *Value2*, and so on. You can specify as few or as many input arguments as you want, up to the total number of arguments defined by the program (currently three). They can also be in any order.

The actual arguments you can specify are covered in the following section.

## 4. INPUT

The program requires two to three pieces of information to run: a spreadsheet file of autodetections, a folder of raw audio (WAV) files, and optionally, a list of parameter values.

### 4.1 Autodetections Spreadsheet File

This can be either an initial CSV file produced by MERIDIAN, an XLSX file created using the *LFDCS2MERIDIAN* script, or a working XLSX file that has been previously created by the validation program (see *OUTPUT*). If you specify a working XLSX file, the program will automatically pick up where you left off. In the case of XLSX files produced by *LFDCS2MERIDIAN*, any validations you already made in LFDCS will also carry over.

The spreadsheet file can be specified in the command line as an input argument with the name *data\_file*. For example:

```
validateMeridianDetections('data_file','C:/Users/BeslinW/Desktop/ROB_2018_04/detections.xlsx')
```

Or another way using variables:

```
in_file = 'C:/Users/BeslinW/Desktop/ROB_2018_04/detections.xlsx';
validateMeridianDetections('data_file',in_file)
```

The command line argument is useful for quickly running the program with the same file multiple times. However, you are not required to use this syntax. If you run *validateMeridianDetections* without specifying the *data\_file* parameter, then you will be prompted to choose a file interactively instead.

### 4.2 Folder of WAV Files

This is the folder containing all WAV files referenced in the detection spreadsheet (i.e., all WAV files within which there are detections to view or validate). It is important that there be no missing files here.

As with the autodetections spreadsheet, the WAV folder can be specified in the command line, using the name *audio\_folder*. For example:

```
validateMeridianDetections('audio_folder','C:/Users/BeslinW/Desktop/ROB_2018_04/rec')
```

Or:

```
in_dir = 'C:/Users/BeslinW/Desktop/ROB_2018_04/rec';
validateMeridianDetections('audio_folder',in_dir)
```

Also as with the spreadsheet file, the audio folder does not necessarily need to be specified in the command line this way. If not explicitly specified, the user will be prompted to choose a folder.



The program can technically work with recordings of any sampling rate, but there are some caveats. Higher sampling rates such as 256 kHz may perform more slowly, as may sampling rates that are not powers of two. Audio playback may also not work at all for higher rates. If possible, downsampling to something like 8 kHz or lower is preferable, but this is not strictly necessary. Note also that sampling rate will limit the maximum frequency that can be displayed.

### 4.3 Parameters

The program has various parameters that you can adjust to customize how it looks and operates. These are specified in a TXT file, similar to LFDSCS. The folder *PARAMS* is a place where you can store as many custom parameter files as you want. Keeping parameter files here is not strictly necessary, but it does make things easier as will be shown below. The *PARAMS* folder also includes the file for default parameters, named *default\_params.txt*. Use this file as a template when creating custom parameter files. When making custom files, you can write whatever you want in the header and change the parameter values, but everything else should remain the same.



Do not remove or change the name of *default\_params.txt*, otherwise the program may not work. There is nothing stopping you from editing its contents (as long as you retain the correct format), but I do not recommend this. Custom parameter files are the preferred way of passing alternative parameter values.

To use a custom parameter file, you must specify it in the command line as an input argument. The argument name for parameter files is *params*. If your parameter files are stored in the *PARAMS* folder, then you only need to specify the name of the file:

```
validateMeridianDetections('params','my_params')
```

If you want to use a file that is not in the *PARAMS* folder, then you will need to specify the full path, as with the autodetections spreadsheet and audio folder:

```
validateMeridianDetections('params','C:/Users/BeslinW/Desktop/my_params')
```

The parameters themselves are described below.

## Spectrogram Settings

- WinSize** – Spectrogram window size, measured in seconds. Essentially, this is the amount of audio data over which to calculate the Discrete Fourier Transform (DFT) for each step. Choosing an appropriate value is a tradeoff between achieving high resolution in time and high resolution in frequency: Higher values capture more frequency information, but at the cost of reduced detail in time.  
 The default value of 0.256 is based on Kirsebom et al. (2020).
- StepSize** – Spectrogram stepping size, in seconds. It is the amount of time in which to step the window forward and calculate the DFT from the next chunk of data. This parameter can also be seen as the *perceived* time resolution; i.e., it is what determines where your “data points” will lie in the spectrogram on the time axis. Large values result in coarser-looking spectrograms, while smaller values result in a finer scale and are more computationally expensive. Note however that the *true* temporal resolution is dictated by *WinSize*. If you have a very large *WinSize* and a very small *StepSize*, it will *look* high-resolution, but the data itself will actually be very smoothed out (for example, rapid burst-pulse signals could appear continuous).  
 The amount of overlapping between subsequent windows is equal to  $WinSize - StepSize$ . *StepSize* should always be  $> 0$  and generally  $\leq WinSize/2$ . Values greater than  $WinSize/2$  may result in a loss of information depending on the function used for windowing (in this case, “hamming”). Values greater than *WinSize* would outright skip portions of the time series and should never be used.  
 The default value of 0.032 is based on Kirsebom et al. (2020).
- NFFT\_8kHz** – This is the equivalent number of points to use in the Fast Fourier Transform (FFT) algorithm used to calculate DFT, if the recording was sampled at 8 kHz. The actual number of points used will be adjusted for the data’s real sampling rate. NFFT is what determines the frequency step of the spectrogram, or *perceived* frequency resolution.  
 Using 8 kHz as a reference allows the frequency step to be conserved between sampling rates based on the simple calculation:  














$$NFFT_{Final} = Fs(NFFT_{8kHz}/8000)$$
 Where  $Fs$  is the real sampling rate.  $NFFT_{Final}$  should always result in a number of samples that is greater than or equal to the number of samples resulting from *WinSize*, that is:  

$$NFFT_{Final} \geq Fs \times WinSize$$
 Similar to *StepSize* for the time axis, values of  $NFFT_{Final}$  that are above  $Fs \times WinSize$  make the spectrogram look smoother along the frequency axis, but they do not improve the *true* frequency resolution, which is determined by *WinSize*. Ideally,  $NFFT_{Final}$  should be a power of two, as that is more computationally efficient.  
 The default value of 2048 was inferred from Kirsebom et al. (2020).

## Plot Parameters

- TSpanDefault** – This determines the time span, or duration, of the area that is plotted when you start the program or reset the plot. Measured in seconds. The time span will change as you zoom in/out across time (see *USAGE > Keyboard Inputs*)

- **FMaxDefault** – This is the upper frequency limit of the plot when you start the program or reset the plot. Measured in Hertz. The upper frequency limit will change as you zoom in/out across time (see *USAGE > Keyboard Inputs*)
- **TPanFactor** – Factor by which to step forward or back in time when panning across the spectrogram. This is measured relative to the current time span of the viewing window; so for example, if the panning factor is 0.75 and the current view spans 8 sec, then panning right will cause the view to shift forward by 6 sec. See *USAGE > Keyboard Inputs* for more on panning.
- **TZoomFactor** – Factor by which to increase the time span of the plot when zooming out in time. Zooming in will reduce the time span by  $1/TZoomFactor$ . So for example, if the current view spans 8 sec, zooming out with a factor of 2 will increase the time span to 16 sec, while zooming in will reduce it to 4 sec. See *USAGE > Keyboard Inputs* for more on zooming in time.
- **FZoomFactor** – Factor by which to increase the upper frequency limit of the plot when zooming out in frequency. Zooming in will reduce the upper frequency by  $1/FZoomFactor$ . So for example, if the current upper frequency is 1000 Hz, zooming out with a factor of 2 will increase it to 2000 Hz, while zooming in will reduce it to 500 Hz. See *USAGE > Keyboard Inputs* for more on zooming in frequency.
- **DefaultColormap** – The name of the colormap to use for the spectrogram on startup. The colormap can also be changed interactively (see *USAGE*). The available colormaps are listed below (modified from the MATLAB doc pages):

Colormap Name	Color Scale
parula	
hsv	
hot	
cool	
spring	
summer	
autumn	
winter	
gray	
bone	
copper	
pink	
jet	

## Marker Parameters

- **LineWidth** – Defines the normal thickness of the outline of the boxes marking detections or annotations.
- **FaceAlpha** – Defines the transparency of the inner face of the boxes marking detections or annotations. Must be between or equal to 0 and 1, where 0 is fully transparent (i.e., outline only) and 1 is fully opaque.
- **StandardFRange** – Frequency limits of the boxes marking detections/target calls. Measured in Hertz. Put this in brackets with the two values separated by a comma.
- **Color\_DetectedFocal** – Color of the box marking the detection in focus, specified as [R, G, B] where each RGB value is a number between 0 and 1.
- **Color\_DetectedOther** – Color of the boxes marking detections not in focus, specified as [R, G, B] where each RGB value is a number between 0 and 1.
- **Color\_MissedDefinite** – Color of the boxes marking definite missed calls, specified as an [R, G, B] vector or the string *auto*. If *auto* is specified, the color will be the same as the button for classifying Correct calls.
- **Color\_MissedPotential** – Color of the boxes marking potential missed calls, specified as an [R, G, B] vector or the string *auto*. If *auto* is specified, the color will be the same as the button for classifying Unknown calls.
- **Color\_Annotations** – Color of the boxes marking general annotations, specified as an [R, G, B] vector or the string *auto*. If *auto* is specified, the color will be the same as the Annotations button.



Please note that the parameters are subject to change in future versions of the code

## 5. OUTPUT

When processing a new dataset from a MERIDIAN CSV file, the program will create an Excel spreadsheet file to keep track of your validations, comments and annotations. Where you save this file and what you name it is up to you – you will be prompted to specify this information whenever you load a new MERIDIAN CSV. Once created, the output file will be automatically updated as you work through the program.

XLSX files created using the LFDCS2MERIDIAN script already follow the output format, so a new file will not be created when importing one of these into the validation tool for the first time.



It is NOT recommended to modify the output spreadsheet manually. Doing so could have unexpected effects or may cause the program to stop recognizing it altogether. If you need to edit anything manually, create a separate copy. Furthermore, if the output file is open while the program is in use, the program will be unable to save its updates.

The file will have three sheets, with the following columns:

- **Detected** – *FileName, FileStart, SigStart, SigEnd, SigStartDateTime, Class\_LFDCS, Class\_MATLAB, ReasonForUNK, Comments*
- **Undetected** – *FileName, FileStart, SigStart, SigEnd, SigStartDateTime, Type, Comments*
- **Annotations** – *FileName, FileStart, SigStart, SigEnd, SigMinFreq, SigMaxFreq, SigStartDateTime, Comments*

The *Detected* sheet lists all candidate calls detected by the autodetector. The *Undetected* and *Annotations* sheet lists any missed calls or general annotations that you have marked yourself, respectively (see *USAGE > Button Inputs*).



Because the program deals with both detected and undetected sounds, I have opted to refer to candidate calls in general as “signals” rather than “detections”.

Information in each column is as follows:

- **FileName** – Name of WAV file containing the signal
- **FileStart** – Start time of the WAV file in seconds, since 1-Jan-1970 00:00:00
- **SigStart** – Start time of the signal in seconds, relative to file start
- **SigEnd** – End time of the signal in seconds, relative to file start
- **SigMinFreq** – [ANNOTATIONS ONLY] Lower frequency of the signal in Hertz
- **SigMaxFreq** – [ANNOTATIONS ONLY] Upper frequency of the signal in Hertz
- **SigStartDateTime** – absolute date/time of signal occurrence (e.g. 07-Jun-2015 18:45:04). Note that this is only accurate to the nearest second.
- **Class\_LFDCS** - [DETECTED ONLY] Manual signal classification made in LFDCS, in string format (i.e. ‘Correct’, ‘Incorrect’, ‘Unknown’, or ‘Unclassified’). A blank entry means there is no corresponding LFDCS detection. This is really only relevant for files that have been converted using *LFDCS2MERIDIAN*.
- **Class\_MATLAB** – [DETECTED ONLY] Manual signal classification made with this MATLAB validation tool. This uses the same format as *Class\_LFDCS*, except there is no explicit ‘Unclassified’ value. For output files created from MERIDIAN CSVs, signals that have yet to be



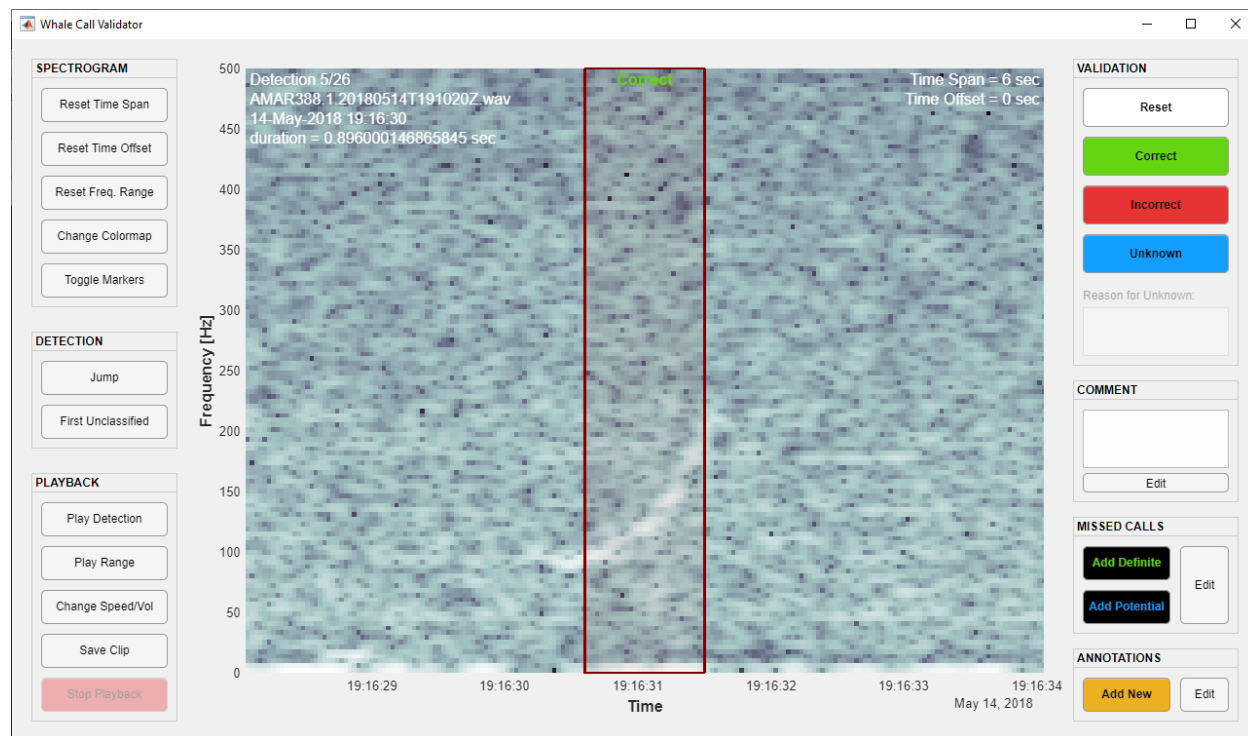
classified will have a blank entry; for files created using LFDCS2MERIDIAN, *Class\_MATLAB* will initially be set equal to the classifications made in LFDCS, if there are any.

- **ReasonForUNK** – [DETECTED ONLY] Description of why a signal has been classified as 'Unknown', if it has been classified as such
- **Type** – [UNDETECTED ONLY] Classification for missed calls. May be 'Definite' or 'Potential'.
- **Comments** – Any comments associated with a signal

## 6. USAGE

The program works similarly to the LFDCS browser: it focuses on one detection at a time and offers a number of options. Unlike LFDCS, it is possible to move forward and backward through the time series, and it is possible to manually add marks to other areas of the spectrogram (but only for audio files where there are detections). Many options are available via GUI buttons similar to LFDCS, but navigating through the plot/detections is generally done with keyboard inputs.

Here is what the main window looks like:



The user-assigned classification of the detection in focus always appears at the top centre and is colour-coded. The boxes marking signals on the spectrogram come in different colours which mean different things. The default marker colours can be changed using parameter files (see *INPUT > Parameters > Marker Parameters*).

Each button and keyboard action is described below.



The figure window must be in focus for the keyboard inputs to register. If you find that your key presses are not doing anything, try clicking in the figure area. Note also that keyboard navigation is not available in certain situations, like during audio playback or when marking undetected calls.

## 6.1 Keyboard Inputs

Most of the following operations are exclusive to the keyboard, though a few are available via GUI buttons as well.

- **W** – Zoom in along the time axis (decrease time span)
- **S** – Zoom out along the time axis (increase time span)
- **A** – Go to previous detection
- **D** – Go to next detection
- **Shift+W/UpArrow** – Increase upper frequency limit
- **Shift+S/DownArrow** – Decrease upper frequency limit
- **Shift+A/LeftArrow** – Pan left. You can pan up to the beginning of the audio file in which the focal detection is located. Beyond that is nothing but the Void...
- **Shift+D/RightArrow** – Pan right. You can pan up to the end of the audio file in which the focal detection is located. Beyond that is nothing but the Void...
- **R** – Reset time span, time offset, and upper frequency limit to their defaults
- **T** – Toggle signal marker boxes
- **C** – Change color map of the spectrogram (see *INPUT > Parameters > Plot Parameters* for a list of available colormaps)
- **Q** – Close the application. You can also use the X button to do this. Either way, the program will attempt to save any changes before closing.
- **Esc** – Cancel the following operations:
  - marking/unmarking of undetected calls or annotations
  - editing of undetected call or annotation markers
  - audio playback
- **Return** – Exit editing of undetected call or annotation markers

## 6.2 Button Inputs

### SPECTROGRAM panel

- **Reset Time Span** – Reset time zoom to its default
- **Reset Time Offset** – Reset time translation to its default
- **Reset Freq. Range** – Reset frequency range to its default
- **Change Colormap** – change color map of the spectrogram (see *INPUT > Parameters > Plot Parameters* for a list of available colormaps)
- **Toggle Markers** – Activate/deactivate all boxes marking auto-detected or manually marked calls

### DETECTION panel

- **Jump** – Go to a specific detection
- **First Unclassified** – Go to the first detection in the list that has yet to be classified

### PLAYBACK panel

- **Play Detection** – Play audio of the target detection area
- **Play Range** – Play audio over an arbitrary range of your choosing. Specify the range by drawing a temporary box over an area of the spectrogram.
- **Change Speed/Vol** – Change playback speed and volume
- **Stop Playback** – Stop a currently playing clip. This button is active only when audio is playing.

### VALIDATION panel

- **Reset** – Clear classification for current detection, including “Reason for Unknown” if there is one
- **Correct** – Label detection as a true call
- **Incorrect** – Label detection as a false positive
- **Unknown** – Label detection as uncertain. You will be prompted to specify a reason why.

### COMMENT panel

- **Edit** – Enter or edit comment for current detection

### MISSED CALLS panel

- **Add Definite** – Mark a definite undetected call if you see one. After this button is pressed, you can draw a window over an area of the spectrogram to mark the start/end times of the call. You will also be prompted to enter a comment. Pressing “OK” after the comment will add your selection to the list of missed calls (sheet 2 in the output file).

- **Add Potential** – Mark a potential undetected call if you see one. This works the same way as for “Definite” calls, but will be labeled as “Potential” in the spreadsheet.
- **Edit** – Enter edit mode for missed calls. In this mode, you can click on missed call boxes and do the following:
  - Resize boxes by clicking on and dragging their edges (draggable edges will highlight when you hover the cursor over them). For missed calls, you can only edit the left and right edges.
  - Delete the box by pressing the **Delete** or **X** key. This will remove the entry from the spreadsheet.

### ANNOTATIONS panel

- **Add New** – Mark an arbitrary area on the spectrogram to indicate something of interest. This works similarly to the Missed Calls buttons in that you will be prompted to draw a box, but differs in that you can set the frequency limits as well. In other words, you can set all four edges of an annotation box to any position you want, unlike the Missed Calls boxes which have fixed upper and lower limits.
- **Edit** – Enter edit mode for annotations. This works similarly to the edit mode for missed calls – you can click on any annotation box and do the following:
  - Resize boxes by clicking on and dragging their edges (draggable edges will highlight when you hover the cursor over them). You can resize all four edges for annotations.
  - Delete the box by pressing the **Delete** or **X** key. This will remove the entry from the spreadsheet.

## 7. USING WITH LFDCS

The program can be used to view/validate autodetections from LFDCS. To do this, you will first need to convert a CSV file of autodetections exported from LFDCS into a compatible format. This is done using the *LFDCS2MERIDIAN* script included in the program folder. Once the spreadsheet has been converted, you can read it into the main program (see *INPUT*).

Running *LFDCS2MERIDIAN* is accomplished similarly to the main program – just type the script name in the MATLAB command window:

```
LFDCS2MERIDIAN
```

As with the main program, there are also a number of Name-Value pair input arguments you can specify with the syntax:

```
LFDCS2MERIDIAN('Name1',Value1, 'Name2',Value2, ...)
```

The input arguments are described below.

- **input\_file** – Path of the LFDCS autodetections spreadsheet you want to convert. If not specified, you will be prompted to select the file interactively.
- **wav\_dir** – Path of the folder containing raw audio files from which the detections originate. If not specified, you will be prompted to select the folder interactively.
- **output\_file** – Path in which to write the converted autodetections file. If not specified, you will be prompted to save the file interactively.
- **sp\_codes** – Vector of numeric LFDCS validated species codes to keep in the converted spreadsheet. These should be limited to the following codes:
  - 9999 (Correct)
  - -9999 (Incorrect)
  - 0 (Unknown)
  - -32767 (Unclassified)

The converted spreadsheet will only retain detections with a manual validation code that corresponds to one of the entries in the list. For example, the following command will only keep calls classified as Correct and Unknown in the converted spreadsheet:

```
LFDCS2MERIDIAN('sp_codes', [9999, 0])
```

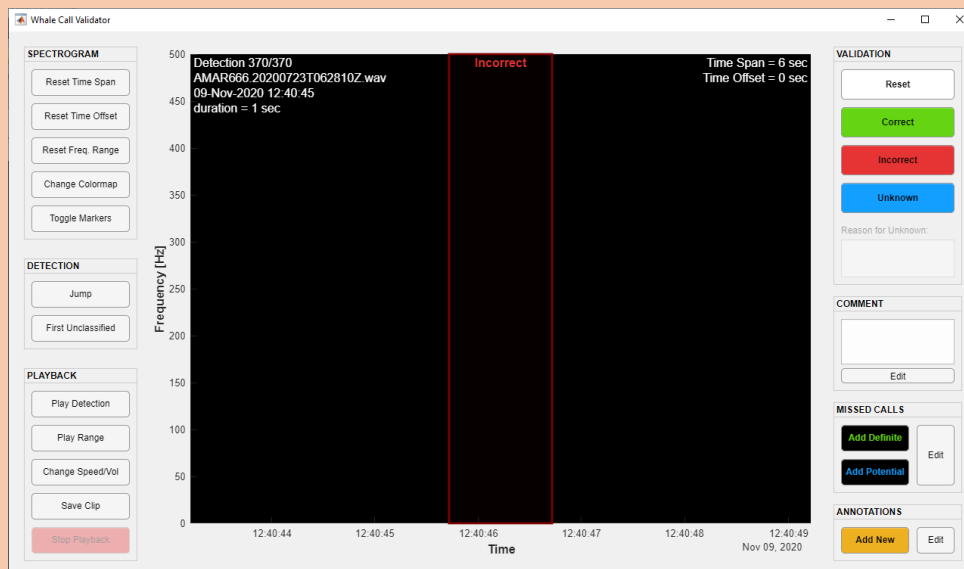
The converted spreadsheet will also translate these numeric codes into their corresponding descriptive strings above (except for Unclassified, which will be blank instead). If the vector is empty(i.e., []), then all detections will be retained – this is the default behaviour.

- **wav\_subfolders** – A Boolean value (true/false) specifying whether to search for audio files within any subfolders that may be nested within the root audio folder. Use this if your audio files are divided across multiple folders. Disable it if you have subfolders with irrelevant files. The default is True.



Audio files currently **MUST** have a timestamp embedded somewhere in the filename, with the format *YYYYMMDD.hhmmss* where *.* is any character. *LFDCS2MERIDIAN* will not work if it cannot find timestamps.

*LFDCS2MERIDIAN* determines which audio files each detection originates from by comparing the start times of each detection to the start times of each audio file. However, it currently does not evaluate the end time of the audio files. Therefore, it is very important that there be **NO MISSING AUDIO FILES**, especially in the middle or at the end of the sequence. If there are, then some detections may end up with the wrong file being assigned to them. In the validation app, this typically manifests as a detection marker appearing against a black background:



## 8. REFERENCES

Kirsebom, O.S., Frazao, F., Simard, Y., Roy, N., Matwin, S., and Giard, S. (2020). "Performance of a deep neural network at detecting North Atlantic right whale upcalls," *J. Acoust. Soc. Am.* **147** (4), 2636–2646