Name ideas: RMDCS (raven mediated detection and classification system)

LFDCS: 1. Not optimized to reduce FPR, optimized to go fast

2. only on mac

3. interfaced through terminal, which makes it hard to look under the hood and tweak actual program

Mastor\_detecter

1. Optimized to reduce FPR, much slower computation
2. On PC, could probably port to mac
3. R script is interpretable and customizable by any researcher familiar with the language.

Autodetection tools are popular in passive acoustics to streamline manual analysis. Detector performance is commonly compared via TPR and FPR, but factors such as run time, ease of setup and use, and platform compatibility are also important considerations to performance. The LFDCS is a premier analysis tool in the field. It was designed to be compatible with near real-time detection on wave gliders so it is very computationally efficient. The trade off is that the learning method of the LFDCS (quadratic discriminant function analysis) only utilizes four features that do not contain information that allows for the discrimination of noise that can resemble positives for the target species. This can result in high FPR, particularly with environments and recorders that experience regular instrument noise, which is the case with AURAL recorders in the arctic. High FPR results in longer analysis time and more expensive analysis.

Despite not being optimized for this tradeoff from the perspective of acoustics labs working on archival data, the LFDCS is often used for this purpose. The LFCDS is also a Mac only implementation and interfaced through the command line, so ubiquity and customization for analysts is limited.

We submit an alternative approach for machine assisted analysis of low frequency sounds, optimized for efficient analysis time on archival data. This approach relies on a representative library of both the positives and negatives of a pitch tracker on ground truth reference data. It extracts measurements of each known and putative call to build and compare with random forest models. As you supply positive and negative detections, the models not only learn the identifying features of true positives, but also that of consistent types of false positives. In this way it ‘learns from its mistakes’. This is a flexible architecture that has been successfully applied to right whale upcalls and gunshots, as well as being able to discriminate from a variety of consistent sources of false positives due to noise. It is designed to be compatible with any stereotyped, distinct call type, and resilient in a variety of acoustic environments.

The detector is associated with Raven Pro 1.5 as part of its core functionality. The detector uses Raven Pro 1.5 Band Limited Energy Detectors (BLEDs) through API, and selection table outputs are formatted to be reviewed in Raven. The use of Raven software, as well as an implementation of the R language, make Mastor\_detecter comfortable to interface with and customize for scientists in the field.

Glossary

|  |  |
| --- | --- |
| Pitch tracker | Part of the detector that refers to the combined output of the Raven BLED suites and custom algorithm |
| BLED | Band Limited Energy Detector. Implemented in Raven |
| BLED suites | Group of BLEDs that is ran to produce MDs with high resolution time and frequency components |
| MD | Minidetections. Output from the BLED suites. |
| Custom algorithm | User designed algorithm that uses the time and frequency components of MDs to filter likely FPs. |
| AURAL |  |
| LFDCS |  |
| Mastor\_detecter | Name of described detection and classification system. |
| TPR | True positive rate, also known as sensitivity. Defined as TP/(TP+FN) |
| FPR | False positive rate, or precision of false positives. Defined as FP/(TP+FP). Note for myself: FPR may be a misnomer, seems to be calculated differently for technical definition, but we seem to be using it the same way as flightcallr paper. |
| TP | True positive detection. Defined as an autodetection that bounds the meantime of a manually annotated detection and/or has its own meantime bounded by a manually annotated detection. |
| FP | False positive detection. Defined as an autodetection that does not bound the meantime of a manually annotated detection and does not have its own meantime bounded by a manually annotated detection. |
| FN | False negative detection. Defined as an manually annotated detection that does not bound the meantime of an autodetection and does not have its own meantime bounded by an autodetection. |
| Meantime | Average time of a time by frequency box. = (start time + end time)/2 |
| Masking | Effect of higher signal noise obscuring desired signal when overlaid on same time and frequency. Note to self put in another section: The Raven BLEDs and feature extraction both respond poorly to masking. For the Raven BLEDs, recorder self-noise can interfere with calculations of background noise, and for the image analysis component of feature extraction, thresholding will only choose the loudest portion of an autodetection to measure and will miss attributes of more faint call. It is possible to do feature extraction with variable amplitude thresholding, where different ranges of amplitude for a given autodetection each had features extracted, but this would increase run time of script by roughly 2 \* # of amplitude ranges. |

Introduction:

AURAL recorders in the arctic region can suffer from a high degree of self-noise, making implementation of autodetectors challenging due to increased FPR. Popular autodetectors such as the LFDCS (which uses QDFA on 4 features: ) and Ishmael (which uses spectrogram correlation) are not resilient against false positives produced by intermittent noise as they don’t have the necessary features and learning system to distinguish mooring self-noise from positive calls. High FPR hurts the implementation of an autodetector as it either 1. increases the time of an analyst to verify detections or 2. reduces TPR due to having to set a lower threshold for time efficient analysis.

Mastor\_detecter was created as a solution to the high FPR issue encountered with existing autodetectors on high self-noise recordings. While the idea to model true positives from a band limited energy detector against null data is not new (flightcallr, Mellinger 2004), mastor\_detector is the first detector that models the true and false positives of a pitch tracker result to identify and weed out consistent sources of false positives. Reducing the FPR of outputs enables higher TPR analysis for use cases like boxing or behavior studies, or for faster analysis at a lower set TPR for use cases like seasonal presence. As the models are spontaneously generated from a set of features in the ground truth library, this method should be applicable to other regions that experience high levels of self-noise on their recorders even if the composition and prevalence of the noise is different.

Mastor\_detecter has similar use limitations to existing pitch tracking detectors- it can be used to identify stereotyped, distinct calls that preferably feature FM, but is vulnerable to anything that will mask or obscure the shape of calls. It cannot use species presence or patterning of calls as a criteria for detection, so manual verification is required in cases where the call type in question may be attributable to multiple species (ie NPRW upcalls and bowhead, humpback, bearded seal upsweeps). For this reason, we refer to the detector that was designed for right whale upcalls as an upsweep detector, since there are multiple species in our system that will produce sounds that are indistinguishable from high probability upcalls.

Mastor\_detecter is easy to use, but hard to set up. BLEDs must be created manually in Raven, and the custom algorithm must be coded in R language. Mastor\_detector will never be an executable program, but rather a flexible infrastructure that can be customized to various call types. Due to high runtimes, it is not meant to be applicable to real time or near real time detection, but serves as an efficient method for reducing the analyst time needed for detecting or boxing desired calls from archival acoustic data.

We have successfully applied Mastor\_detector to right whale upcalls and gunshots and demonstrate results for these applications, as well as provide guidance for its implementation to any distinct, stereotyped signal of interest.

Methods:

Construction and design:

We identified the Raven BLEDs as a good starting place to assess the viability of applying energy detectors to our data, given the comfortability of the Raven interface. Upon finding an R implementation to call this algorithm from API, we decided to continue developing around Raven and the R language. It was determined that in environments where the amplitude of mooring noise was greater than that of individual calls, wide band limited energy detectors worked poorly for pulling out faint calls, whereas narrowband BLEDs were more likely to hit pieces of faint calls (hereafter referred to as minidetections (MDs)). With this knowledge, we pursued using a ‘suite’ of narrowband BLEDs throughout the frequency range of the desired call. Using the suite of detectors allowed for an additional ‘pitch-tracking’ filter, by comparing the frequency and time of each MD to assess FM of the signal and filter out some unlikely detections. Using only this pitch tracker method garnered too many FPs to make analysis feasible, so a machine learning solution using RF models inspired by the detector design in (flightcallr) was implemented to help weed out FPs. Image analysis was later introduced to improve the classification performance of the RF models. Parallelization is implemented while running the pitch tracking algorithm, extracting features, and building the models, and is highly recommended to improve run time. The infrastructure for these parallelized sections in R package ‘foreach’ is in theory portable to Azure given that Microsoft developed this package for compatibility with Azure (<https://cran.microsoft.com/web/packages/foreach/foreach.pdf>).

Much of the infrastructure was reworked for compatibility with multiple species detection in a single run of the detector. Included in this idea was the implementation of a more computationally efficient multiclass RF classifier, which seemed to have poor results compared to binary classification. For future attempts at multiclass detection, iterative binary detection and *post hoc* comparison is a safe approach if using RF models as a classifier. Use of different models besides RF may demonstrate better results at multiclass detection. This capability was scaled back as our timeline prioritized effective single species detection during development, but the looping structure is largely intact for this capability.

Upsweep detector:

An upsweep detector was developed using the mastor\_detecter workflow. Segments of .wav data from seven different AURAL deployments that had high presence of RW were chosen to build the pitch tracker and train the RF models (Table 1). These data were decimated by a factor of 16, and combined into single sound files corresponding to mooring year. Upcalls in the data were boxed by hand in Raven. A suite of Raven BLEDs were designed by hand within a conservative frequency range (table 2) to best pick up upcall MDs. A custom algorithm was designed in R to identify likely groups of MDs as putative detections and weed out likely FPs (TABLE#). The Raven BLEDs and algorithm were tested by iteratively running the section of the script “runRavenGT” while changing structure and parameters of both. Performance was evaluated using visual examination of generated selection tables in Raven 1.5 and using autogenerated performance metrics on TPR and FPR.

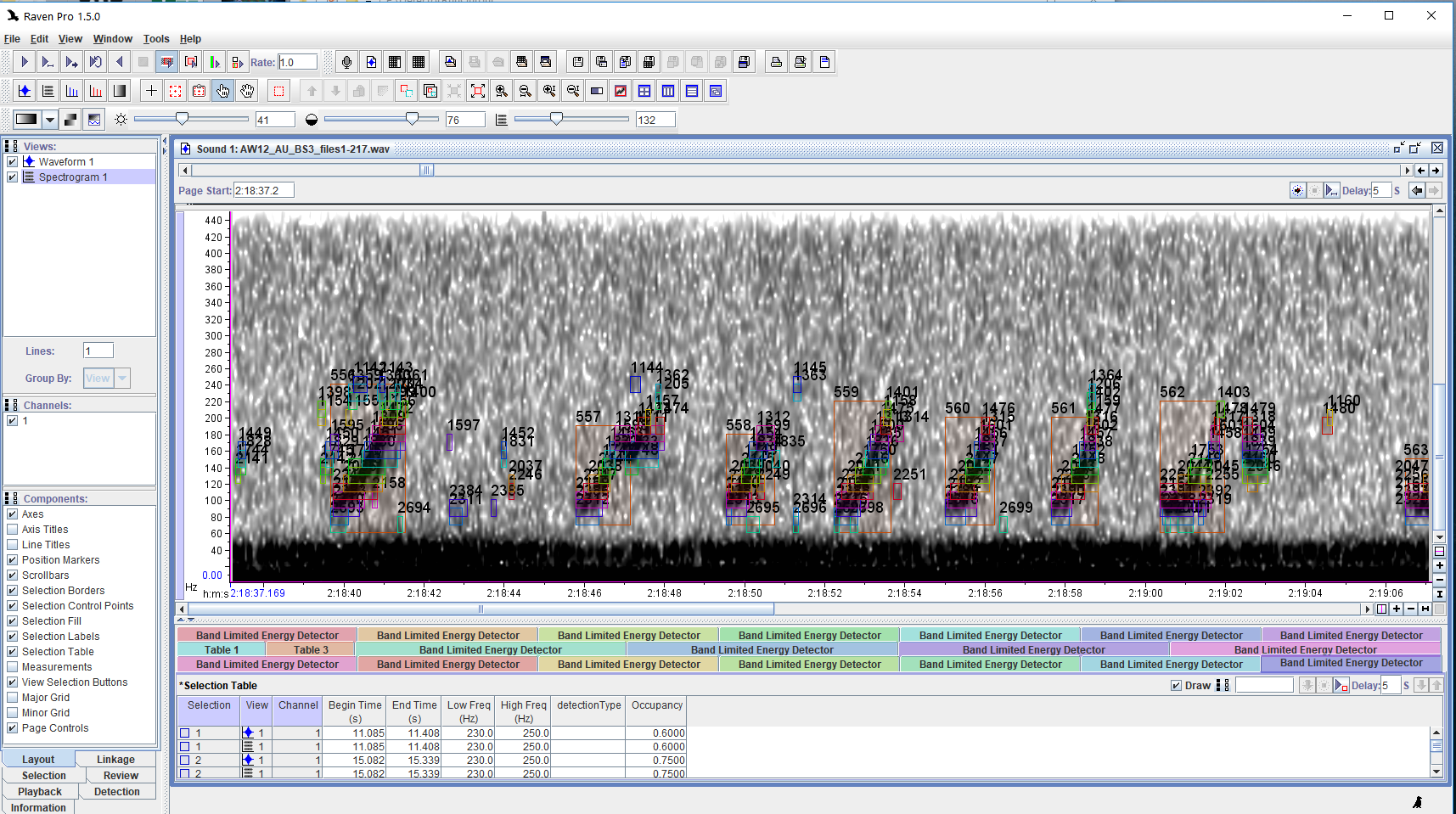
Once satisfactory results were generated (TABLE#) from the newly designed pitch tracker, we built a labelled dataset using the section of the script “runProcessGT”. This section of the script associates the generated selection tables with your ground truth selection tables to assign TP or FP labels for each box, and extracts features from each to build the training dataset that will inform the random forest models for the detector.

Once we had built a pitch tracker and generated the labelled data, the detector is functional to be applied to any .wav file. To test the performance of the final detector, we compared it recursively to the labelled set to test performance (TABLE #) and evaluated results by visual inspection in Raven. Manipulations performed here to improve the model included adding features new features to the labeled data and adding post hoc filters to clean up issues with multiple boxes being assigned to single calls (adaptive\_compare).

Once the performance was found to be sufficient, we applied the detector to new data to test the generalization of the random forest model component. For this, we compared the performance of the model to new data iteratively to test for stabilization as more data were added. Unfortunately, the effect of adding new data was masked by the amount of variance in each individual dataset (FIG#), so an experiment was performed to isolate the effect of adding data from the variance of each dataset (SUPPLEMENTAL?) We found that….. (to be continued). Given the high computational requirements of this experiment, we recommend not to attempt to replicate this experiment as a step in detector construction, and suggest that the best method for evaluating detectors is visual inspection of generated selection tables on new data, providing data as needed in situations where the model is underperforming. Adding data from sections of data that have particularly high FPR or low TPR (unless in the case of masking) to the training set will make the model more sensitive to similar cases.

The same process was implemented for gunshot detection. Data were originally selected from moorings that had high incidence of gunshot (TABLE#). In this case the data were decimated by a factor of 8.

**Figure 1.** Screenshot of Raven Pro 1.5 on ground truth data showing MDs from Raven suite P11, superimposed with positive detections from the upsweep custom algorithm (larger orange boxes) that compare these MDs within the script. At detection #557 at 2:18:44, there is an example of a missed box- a positive detection that misses a part of the call. At present missed boxes still register as detections if they pass the criteria, and their prevalence is not quantified.



**Table 1.** Ground truthing effort for gunshot and upcalls

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Call type | Location | # high graded files | % analyzed | Data hours | Analysis start | Analysis end | Machine assisted\* | How data was  selected |
| Gunshot | BS3 | 1452 | 3.10 | 2.50 | 11/06/2015 | 11/14/2015 | n | Hand |
| Gunshot | M2 | 455 | 10.11 | 2.65 | 06/19/2012 | 07/20/2012 | n | Hand |
| Gunshot | BS3 | 583 | 12.18 | 3.96 | 11/05/2014 | 12/04/2014 | n | Hand |
| Gunshot | M4 | 289 | 30.45 | 4.79 | 08/15/2014 | 09/11/2014 | n | Hand |
| Gunshot | BS3 | 3764 | 6.64 | 13.77 | 08/11/2012 | 08/26/2012 | r | Hand |
| Gunshot | BS3 | 3764 | 1.17 | 2.38 | 10/07/2012 | 10/11/2012 | r | Hand |
| Gunshot | BS3 | 583 | 10.29 | 3.40 | 12/21/2014 | 12/27/2014 | r | Random |
| Gunshot | BS2 | 391 | 23.02 | 3.98 | 10/07/2015 | 10/09/2015 | r | Random |
| Gunshot | BS3 | 576 | 16.15 | 5.35 | 12/03/2016 | 07/01/2017 | r | Random |
| Gunshot | M2 | 1383 | 4.19 | 3.27 | 09/20/2012 | 09/23/2012 | r | Random |
| Gunshot | M4 | 166 | 44.58 | 4.15 | 04/02/2015 | 08/30/2015 | r | Random |
| Upcall | M2 | 194 | 53.61 | 5.88 | 06/22/2015 | 08/09/2015 | n | Hand |
| Upcall | M4 | 179 | 100.00 | 10.08 | 12/02/2014 | 09/20/2015 | n | Hand |
| Upcall | BS3 | 217 | 100.00 | 12.10 | 08/11/2012 | 09/12/2013 | n | Hand |
| Upcall | M4 | 304 | 100.00 | 16.85 | 09/18/2013 | 10/04/2014 | n | Hand |
| Upcall | M2 | 325 | 53.85 | 9.83 | 05/17/2016 | 09/08/2016 | n | Hand |
| Upcall | M2 | 62 | 100.00 | 3.44 | 10/01/2015 | 12/24/2015 | n | Hand |
| Upcall | BS3 | 443 | 36.12 | 9.00 | 10/20/2014 | 06/26/2015 | n | Hand |
| Upcall | BS1 | 158 | 100.00 | 9.06 | 09/26/2016 | 11/25/2016 | r | Random |
| Upcall | M2 | 58 | 100.00 | 3.15 | 07/03/2013 | 07/18/2013 | r | Random |
| Upcall | M5 | 13 | 100.00 | 0.75 | 09/29/2016 | 12/08/2016 | r | Random |
| Upcall | M4 | 540 | 21.48 | 6.56 | 08/15/2016 | 08/26/2016 | r | Random |
| Upcall | BS2 | 86 | 100.00 | 4.71 | 07/21/2015 | 09/25/2015 | r | Random |

**Table 2.** Parameters for suite of Raven BLEDs configured for upsweeps (P11). For the upsweep detector the parameters stay constant through the frequency range. See Raven Pro 1.4 users manual “Configuring Band Limited Energy Detectors” page 274 for in depth description of these parameters.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Suite ID | Low freq (Hz) | High freq (Hz) | Min dur (s) | Max dur (s) | Min sep (s) | Min % occupancy | SNR Threshold (dB) | Block Size (s) | Hop Size (s) | Percentile |
| P11a | 60 | 80 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11b | 70 | 90 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11c | 80 | 100 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11d | 90 | 110 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11e | 100 | 120 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11f | 110 | 130 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11g | 120 | 140 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11h | 130 | 150 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11i | 140 | 160 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11j | 150 | 170 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11k | 160 | 180 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11l | 170 | 190 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11m | 180 | 200 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11n | 190 | 210 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11o | 200 | 220 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11p | 210 | 230 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11q | 220 | 240 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
| P11r | 230 | 250 | 0.25 | 4 | 0.25 | 30 | 3.5 | 5 | 0.25 | 60 |
|  |  |  |  |  |  |  |  |  |  |  |

**Table 3.** Parameters for suite of Raven BLEDs configured for gunshots (Pg2). This detector has changing parameters over the frequency range to better account for the effects of propagation on perceived call duration. See Raven Pro 1.4 users manual “Configuring Band Limited Energy Detectors” page 274 for in depth description of these parameters.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Suite ID | Low freq (Hz) | High freq (Hz) | Min dur (s) | Max dur (s) | Min sep (s) | Min % occupancy | SNR Threshold (dB) | Block Size (s) | Hop Size (s) | Percentile |
| Pg2a | 750 | 850 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2b | 700 | 800 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2c | 650 | 750 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2d | 600 | 700 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2e | 550 | 650 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2f | 500 | 600 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2g | 450 | 550 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2h | 400 | 500 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2i | 350 | 450 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2j | 300 | 400 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2k | 250 | 350 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2l | 200 | 300 | 0 | 0.625 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2m | 150 | 250 | 0.125 | 1 | 0.125 | 50 | 3.5 | 1 | 0.25 | 65 |
| Pg2n | 135 | 165 | 0.125 | 1.75 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2o | 125 | 145 | 0.125 | 1.75 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2p | 115 | 135 | 0.125 | 1.75 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2q | 105 | 125 | 0.125 | 2.5 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2r | 95 | 115 | 0.125 | 3 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2s | 85 | 105 | 0.125 | 3 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2t | 75 | 95 | 0.125 | 3 | 0.5 | 20 | 4 | 6 | 1 | 65 |
| Pg2u | 65 | 85 | 0.125 | 3 | 0.5 | 20 | 4 | 6 | 1 | 65 |

There are many parameters that allow you to control and fine tune the script (parameters, supplemental). There are too many parameters to optimize automatically, so optimization is best fine tuned by hand.

Building detectors for a new signal requires constructing the detector suite in Raven, as well as an algorithm to relate the MD to one another and filter out likely negatives based on time and frequency criteria. New detectors must be built into the script, but it is compartmentalized such that minimal effort is needed to add support for additional detectors.