CROAC frog call analysis technical report

University of Manchester - Advisory & Risk Management Solutions

Author: Richard Bamford

Date: 23/07/2018

CONTENTS

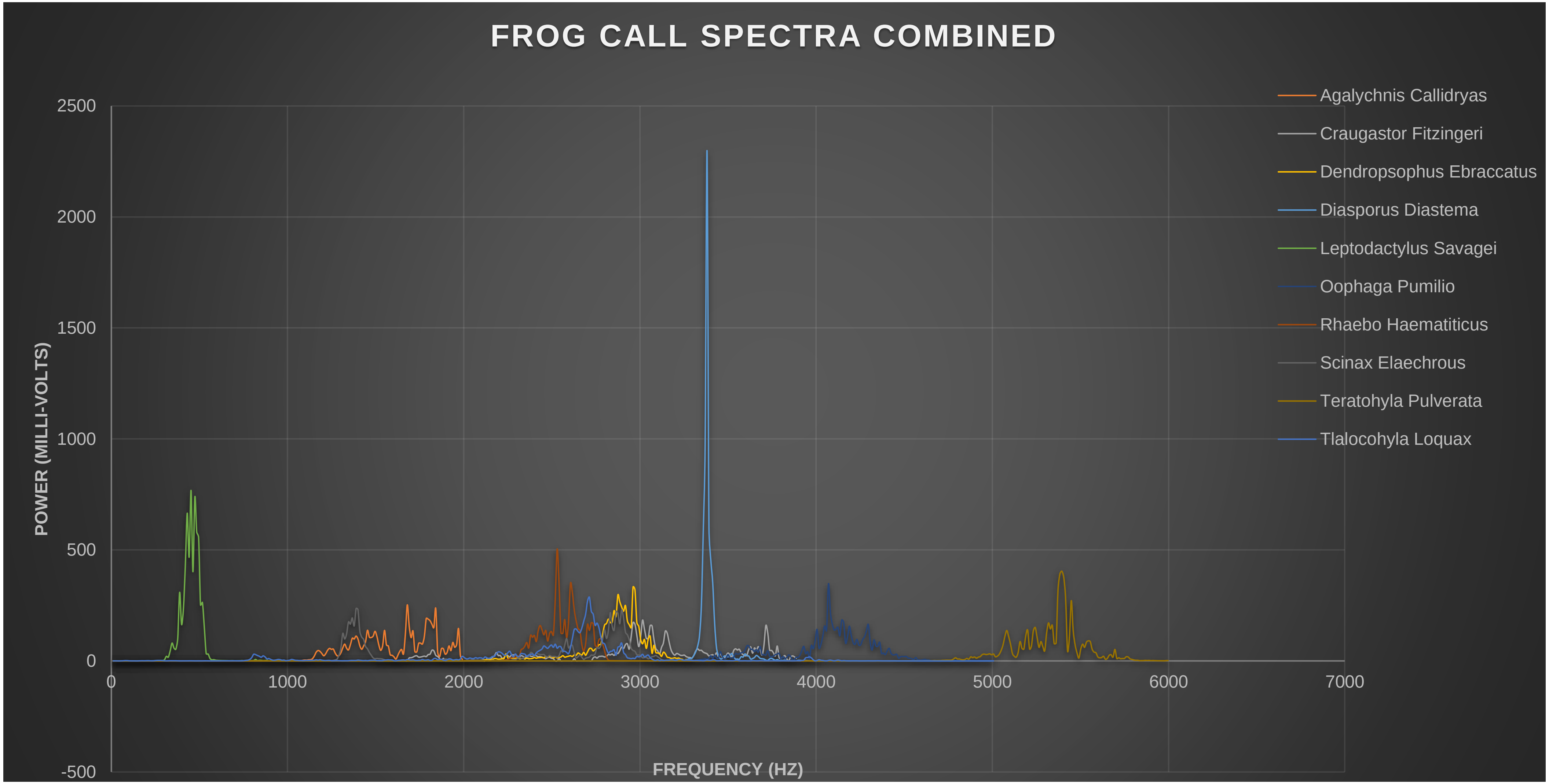
Introduction....2
Frog Spectra Graph....3
Frog Spectra Explained....4
Analysis Technique....5
Analysis Results Explained....6
Improvements....7
Code Metrics....8
Credits....9

INTRODUCTION

In this report I aim to get the reader to understand CROAC in its complexity and function. From how the data itself is transformed to how the correlation algorithm works.

After reading this report I hope that you will understand a signal analysis problem, signal to noise ratio, and how much code development it takes to implement a complex system like this.

FROG SPECTRA GRAPH



Richard Bamford
http://richardbamford.io

FROG SPECTRA EXPLAINED

The graph above displays the frequency bands that each frog call occupies. Using Audacity, I snipped the audio call into multiple single time series that each contain frog call information.

Afterwards I then programmed a Python pipeline which averaged the individual frog call snippets together to create a single averaged frog spectrum file for each frog. The pipeline consists of 276 lines of code and the result of the pipeline is what CROAC uses to run the correlation algorithm against.

This method worked because in a single audio file there were many waveforms of frog calls with background noise in-between, that means an accurate frequency signature is computed for each frog.

A disadvantage of this workflow was that it took a long time to develop the python pipeline and to manually gather the frequency data for each frog from multiple base audio files.

ANALYSIS TECHNIQUE

The aim of the code was to distinguish a frog from the audio data captured – which includes background noise and other animals (cicadas, wind, water and people to name a few).

Initially I attempted to run a correlation between the averaged frog call files and the entire audio captured by the mic.

Obviously, this was a bad idea since fast Fourier transform-ing a lot of data results in a high-level background noise compared to the level of significant call waveforms.

For example, a 1-minute recording would contain 20 seconds of call data and 40 seconds of irrelevant noise, therefore this is within the domain of signal processing.

After a meeting with Professor Stuart Jefferies on how signal-to-noise ratios can be reduced and correctly analysed we came up with this method.

It reduces the noise in each correlation by restricting the time domain. So for a 1 minute recording it is split into 6, 10 second windows which are then analysed. With windows containing a signal having a higher signal-to-noise ratio than before.

1) Split the audio file up into divisions:
    a. Each division is Fourier transformed.
    b. <u>Result is a timeline of frequency spectrums.</u>
2) For each frequency spectrum in the timeline:
    a. The averaged frog file data is scaled to match the same area as the divisions frequency signature.
    b. Pearson correlation ran on the averaged, scaled and trimmed frog file data.
    c. <u>Result is a timeline of correlation values.</u>
3) For each correlation value in the timeline.
    a. A moving average is performed which results in a global averaged R.
    b. The averaged R value is multiplied by 100 to produce a *"percentage match"* that is displayed.
    c. The result is a value between 0 and 100 which represents the similarity between the frequency time series of both the recorded audio and the averaged frog file data.

There are multiple theories to explain the inaccuracy of this method. As it represents a core domain of signal processing which is trying to distinguish a signal from noise.

However, the biggest issue for CROAC in particular is that – as shown in the main graph– the frequency ranges overlap. Note the frequency range 1000Hz to 4000Hz contains a lot of frogs making the detection frog calls within that range have a large margin of error, the frogs are correlated as a <u>false positive.</u>

ANALYSIS RESULTS EXPLAINED

Since the Pearson correlation accuracy depends on the signal to noise ratio in section 3, that means the resulting analysis values are relative to each other and the amount of irrelevant data.

For example, a recording where the top 3 frogs are correlated as 100% means that each frog has the same probability of being in the capture audio data. However, that doesn't mean that all 3 frogs are present with a 100% certainty, since it represents a probability when signal to noise and overlapping data is considered.

Additionally, if the top 3 frog results are 100%, 5% and 1% that doesn't mean that the frog at 100% correlation value is in the recording because it has a huge difference to the others. Most often it tells us some skewing of the correlation data is happening for example, the frog at 100% might be correlating strongly with background noise or other animals/environment.

To conclude, when reading the frog results, the order in which they appear does not really indicate which frog is correct, it just shows which frogs correlate the most. Therefore, the data which dominates wide frequency domains is correlated the most, meaning certain frogs appear almost all the time. Also, the background noise present in recordings were an issue, even when the data was averaged since it skewed every R value. The noise could be ignored to some degree since it skews all R values not individual correlation results, however certain animals are very loud e.g. Cicadas which produced large values at their frequencies consistently.

IMPROVEMENTS

Firstly, in the part of the algorithm described in section 5, part 3.A, we used a moving average to create an R value. This could be further statistically analysed that would improve upon a simple averaging.

Secondly, the frog spectra are overlapping. A great suggestion by Professor Alex Villegas is that we replace the frogs that overlap with others or introduce environmental filters which have distinct frequency ranges that don't overlap. This would be ideal since it fixes the core problem, which is the data.

In summary the analysis technique is watertight with only the development of statistical analysis of the correlation value timeline to improve. The faults in the app are generally part of the signal to noise issue discussed, overlapping data and some user error in recording irrelevant information.

Therefore, if given ideal data, this analysis technique would perform outstandingly as shown by real life detection of frogs in Costa Rica 2017/2018 field trips by University of Manchester.

CODE METRICS

Metric	Description	Value
Maintainability index	-	65 (green)
Cyclomatic complexity	Number of branches within the code.	200
Depth of inheritance	The greatest depth of OOP class inheriting.	6
Class coupling	Number of classes referenced.	139
Lines of code		814 (c#) 272 (python) 200 (nodejs) 200 (html, css & javascript)

CREDITS

From the start of this project my Dad, Dr Gary Bamford has provided 1 on 1 tuition from the basics of signal analysis to deriving our own method of “cleaning” the original file data to create a more accurate workflow. James Bamford for helping manage the business end of the situation and as a spokesperson.

Multiple researchers and scientists that have contributed to my knowledge of signal analysis and aided the creation of a specific system that has never been done on an Android phone before.

The University of Manchester, it has taken 2 years to develop this method and app CROAC, it has been a labour of love since day 1 and I am thankful that they have let me work on this project without pressure.