# App development with audio applications from m-file to app

Daniel Jankovic

August 14, 2015

## 1 Part 1

*Voice Activity Detector* (VAD) is a technique used in signal processing to detect the presence of human voice in a signal. It can be an energy detector that indicates speech when the energy of the filtered signal exceeds a predefined threshold. Considered an important technology in speech based communication, today there are various types of applications that use it. Therefore a wide variety of VAD algorithms have been developed to provide the needed features.

There are different kind of stand-alone commercial baby monitors on the market today. From the most basic, that use one-way radio communication, to advance two-way communication monitors that use signal processing to transmit audio when a predefined threshold has been reached. It is also possible to find baby video monitors that broadcast both audio and video when the sensors notice movement. Since most of the monitor applications rely on radio signals to communicate between the units there is a probability that the signal will weaken or possibly not even reach the receiver because it needs to pass through multiple walls of varying thickness. The signal could also be effected by other applications. As the stand-alone monitor focuses on reliability (among other important sale strategies), little is known about the security features. It is possible to assume that the communication is uncrypted, at least in some products, and therefore introduces a potential risk for intrusion of peoples privacy.

To resolve the issues brought up above, an application such as the *Baby Activity Detector* (BAD) can can be made more portable, versatile and secure with the help of todays smart-phone technology and VAD. There are many VAD algorithms to choose from and they all have their strengths and weaknesses. Complex algorithms such as Linear Predictive coding (LPC), mel-frequency cepstrum (MFC) are very powerful but quite difficult to grasp and also to implement, they can be considered out of scope for this course. The following VAD algorithms are easy to implement and can be, when combined, quite robust for the task of a basic BAD. The simple short-time energy algorithm calculates the energy levels for each frame to detect voice, unvoiced or silenced regions. Voiced regions will have higher energy levels, however, the algorithm does not take unwanted noise into account which means that we can have false indication of voice detection. In order to remove the noise from the signal, spectral subtraction can be preformed. In the case of BAD, the threshold needs to be adjusted so that unforeseeable sound is not interpret as the infants cry. Zero-crossing rate (ZCR), is the rate at which a signal changes from plus to minus and back. The higher the rate the higher the frequency which indicates possible voice activity. According to [1] the cry sound that an infant makes has a fundamental frequency of 250-600 Hz (pitch). To be able to use ZCR together with the information above, it is necessary to extract the pitch from the signal in order to match the frequency interval.

The main task of BAD is to detect infant activity, an alternate algorithm is proposed in [1]. It describes an cry detection algorithm that is build up by three main stages. i) *VAD*, a statistical model-based detector [5] is used for detecting sections with sufficient audio acitivity. It also helps to reduce the power consumption. ii) *Classification*, uses k-nearest neighbours (k-NN) algorithm [6] to label each frame as either 'cry' (1), close enough, or 'no cry' (0). iii) Post-processing, validation stage in order to reduce false-negative errors. The idea of having devoted algorithm to detect infant cry is a winning concept for a BAD application, according to the authors it even had promising results in low SNR. Despite simplicity of the algorithm many of the features required to implement were mentioned earlier to be out of scope for this project.

An algorithm that might be of interest [3] suggests a new approach to speech enhancement, without the help of VAD technology. The signal is divided into multiple sub-bands and an noise floor level estimate is calculated simultaneously as the short-time average. The goal is to boost the sub-bands with high Signal-to-Noise Ration (SNR) instead of to suppress the lower. This algorithm has great potential to reduce the noise levels when analyzing incoming signals to the BAD application.

A quick search on the net gives significant amount of hits for smart-phone based BAD applications. The techniques vary, from bluetooth to Wi-Fi and 3-/4G solutions. The award winning application *Baby Monitor 3G* [7] is a feature rich cross platform application that solves most the of issues brought up in this report. It supports both Wi-Fi and 3G/LTE networks, ability to transfer high quality live video, adjustable microphone sensitivity, talk-back functionality and guarantees both reliability and privacy. It can be assumed that the Android based BAD application *Dormi* [8] offers similar features as Baby Monitor 3G, even if Baby Monitor 3G's feature list provides barely any deeper information. It is noteworthy that *Domri* have both *Smart noise detection* and *Adaptive audio enhancement* as sales pitch, which from a engineering point of view is very attractive.

Following is a purposed algorithm for an BAD application

```
while(true){
    % Record audio can place it into the register %
    Get frame from the register
    Divide the signal into different sub-bands with FFT
    Calculate the total short-time energy average
    Calculate noise level for each sub-bands
    if energy average above threshold
        Calculate the gain for each sub-band
        Boost the sub-band with high SNR
        % Extract fundamental frequency (pitch)
        Count the ZCR under 1 seconds
            if the ZCR is within 250-600 Hz
                Possible infant activity detected!
                (Occurs only first time, and needs to be reseted)
% Send frames %
                Start broadcasting the sound to the receiver
            end
    else
        Reset ZCR
        Dismiss and get next frame from register
    end
}
```

## 2 Part 2

Three various baby and noise sounds were provided. The sample frequency for all of the recorded files are 8 kHz. In Figure 1 and Figure 3 the frequency spectrum is plotted for baby respective noise sounds. Because the energy levels are unproportionally low for two out of three audio recordings an enhanced plot is displayed in Figure 2. From the plots it can be seen that the baby sound spectrum is between $\sim$ 300-2500 Hz and the noise sound spectrum is between $\sim$0-200 Hz and $\sim$1300-2100 Hz. One way to work with only the desired frequencies is to create a bandpass filter with a band between 300-1300 Hz.

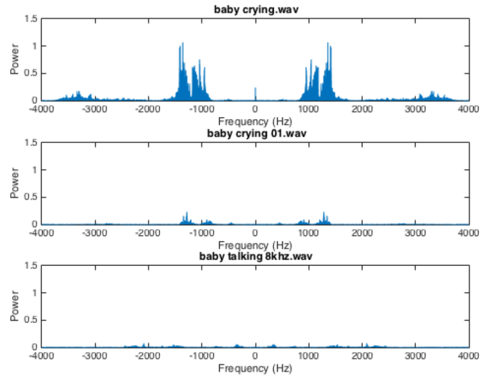


Figure 1: Frequency spectrum for baby sound

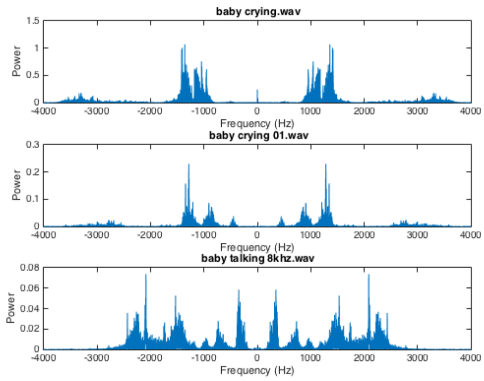


Figure 2: Frequency spectrum for baby sound, scaled y-axis

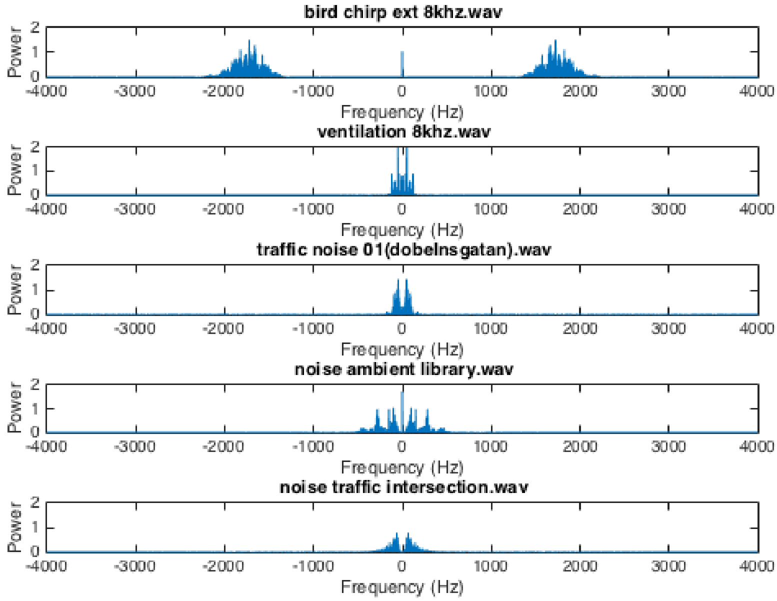


Figure 3: Frequency spectrum for noise files

## 2.1 The algorithms

A quick and simple algorithm for a BAD application is preferably an algorithm that measures the energy from the input signal. Another benefit is that the implementation in Android OS may be easier for a novice application developer. Mathematically described, the power equation, which is close relationship to energy, is given by the following formula:

$$P(n) = \frac{1}{N}\sum_{k=0}^{N-1} x^2(n-k)$$

Since the BAD application will be performing energy calculations of the input signal in real-time, it is undoubtedly impossible to implement the equation. A solution to the problem is to use the *recursive averaging* algorithm. It is a small and hardware friendly algorithm that calculates the energy levels of a given signal without using too much memory. Given the equation below,

$$P(n) = \alpha P(n-1) + (1-\alpha)x^2(n)$$

instead of performing the calculation for one sample at the time, a block of samples,referred to as frames, are squared and summed. Each frame represents

H

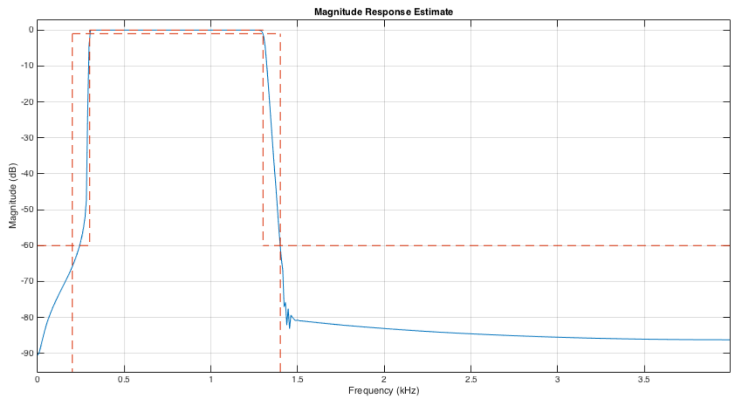


Figure 4: Butterworth 300-1300 Hz bandpass filter

$x^2(n)$. The result, $P(n-1)$, is saved to be used as $P(n-1)$. The $\alpha$ is a constant between 0 and 1 and is related to the following formula:

$$\alpha = \frac{1}{T_s F_s}$$

Despite that the $\alpha$ can be derived from the equation, to get the better results further tweaking and testing is required. The $\alpha$ value used in this research, 0.5, suggests that the new $P(n)$ is equally weighted between old and new calculations.

The advanced algorithm is based upon the same, *recursive averaging*, algorithm but before calculating the average power the signal is first filtered through a Butterworth bandpass filter to remove unwanted frequencies. Butterworth because is gives the least amount of ripple in the bandpass. By having the noise frequencies suppressed, more precision is acquired for finding baby activity. The MATLAB code for the filter can be found in Appendix A.

## 2.2 Evaluation and performance

To evaluate and test the performance of the simple and advance algorithm new sound files were created. The three provided baby sound files created the based. The goal was to make the alarm go off with the sound that the baby caused. For each baby sound file, three new sound files were created to mislead the algorithms. The three tests configurations, with different baby base, contained the following files:

1. clean, no without noise
2. simulate early mornings, bird and ventilation noise added

3. simulate noise environment, all noise files added
4. simulate 'extreme' noise environment, all noise files added and amplified

As can be seen from Figure 6, the green horizontal line indicates the threshold value and the red horizontal line suggest where a possible threshold could be placed. The red circle indicates when the alarm has been set off, when the algorithm has notified that there is baby activity.

The results from *Baby crying.wav*, Figure 5, test show that the alarm was set off in the same approximate time. The reason for this could be that there is a high energy baby sound in the beginning of all of the files and hence why it triggered early.

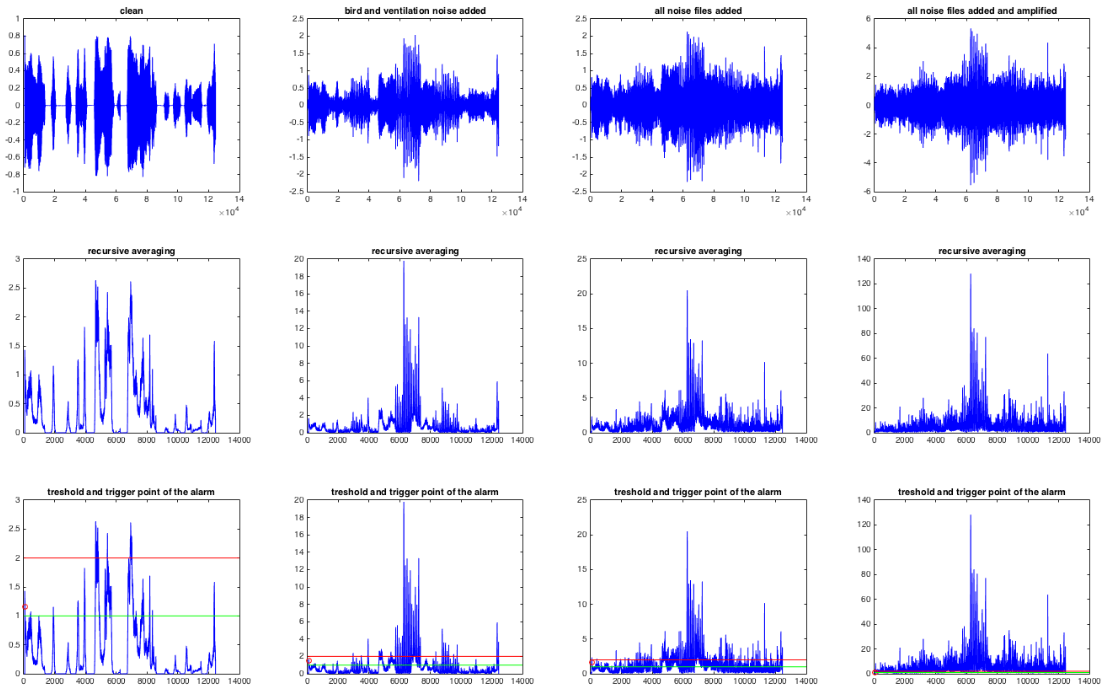


Figure 5: Baby crying.wav, simple algorithm

Figure 6 gives a doubt whether or not the noise could have set off the alarm.

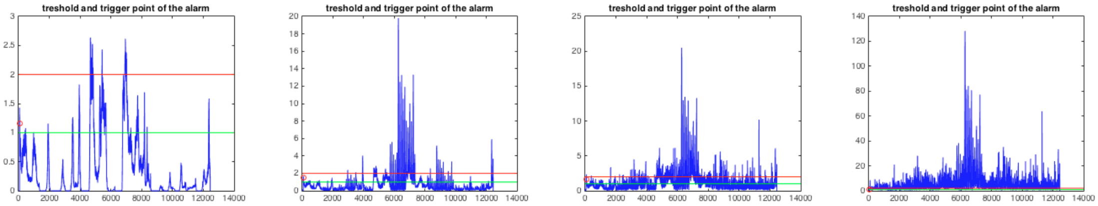


Figure 6: Baby crying.wav, simple algorithm

Both *Baby crying1.wav* and *Baby talking.wav* gave unsatisfactory results without any amplification, as can be seen in Figure 7 and Figure 8. Not remotely close to trigger the alarm in clean configuration, illustrated be the left most plots, the test were performed a second time but with amplified baby sound.

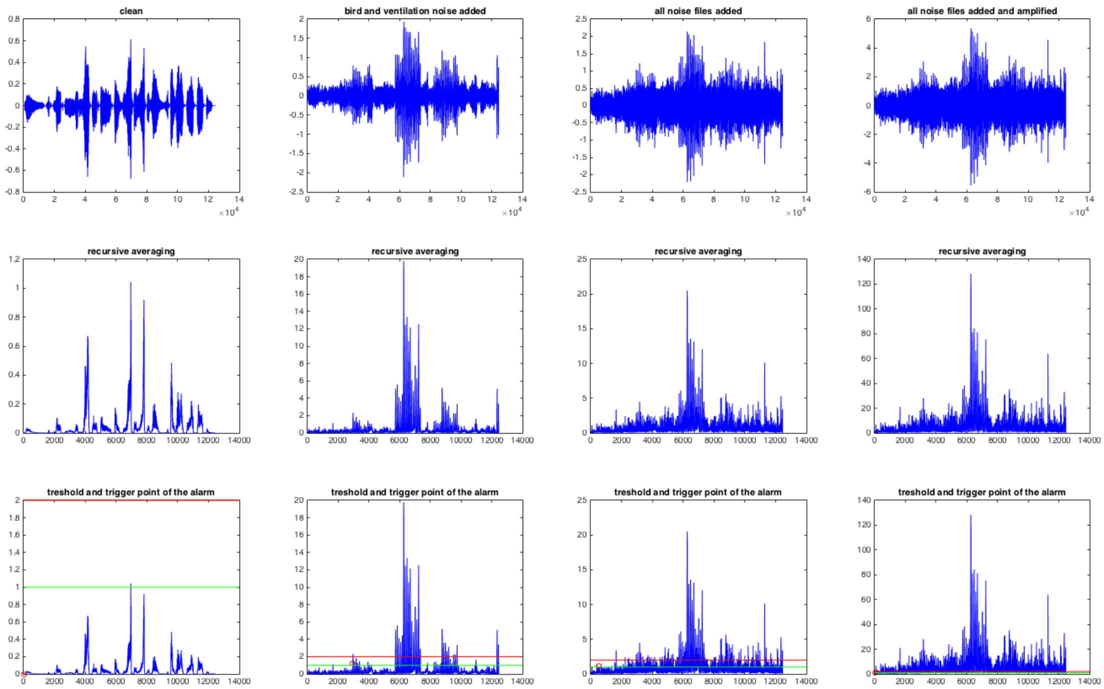


Figure 7: Baby crying1.wav, simple algorithm

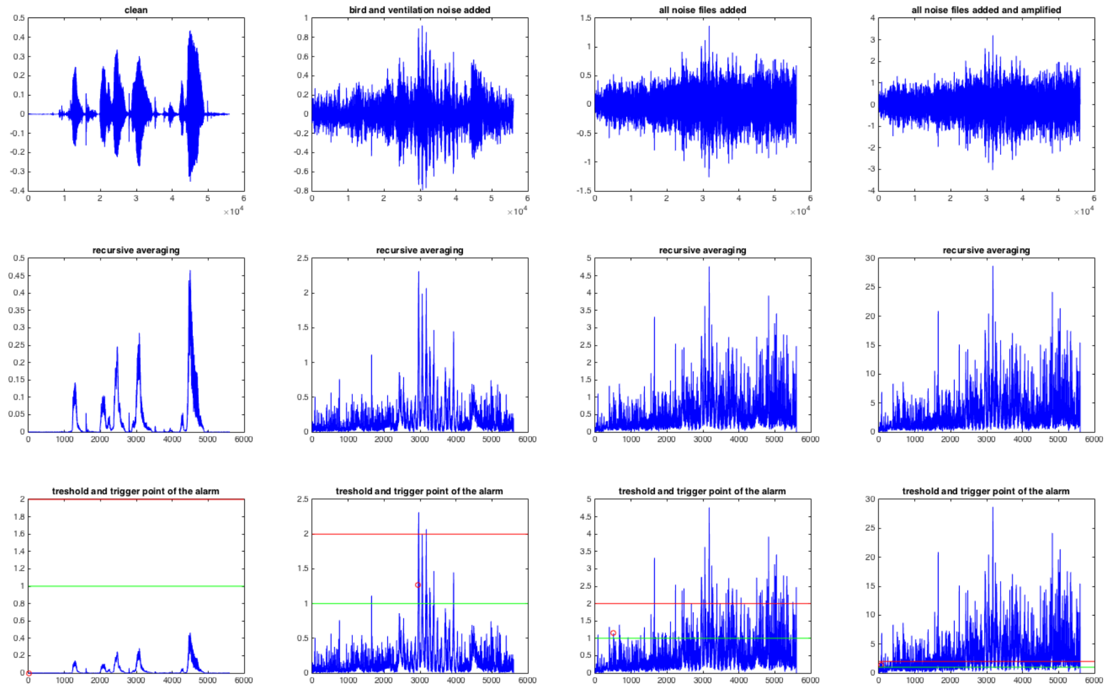


Figure 8: Baby talking.wav, simple algorithm

Figure 9 and Figure **??**

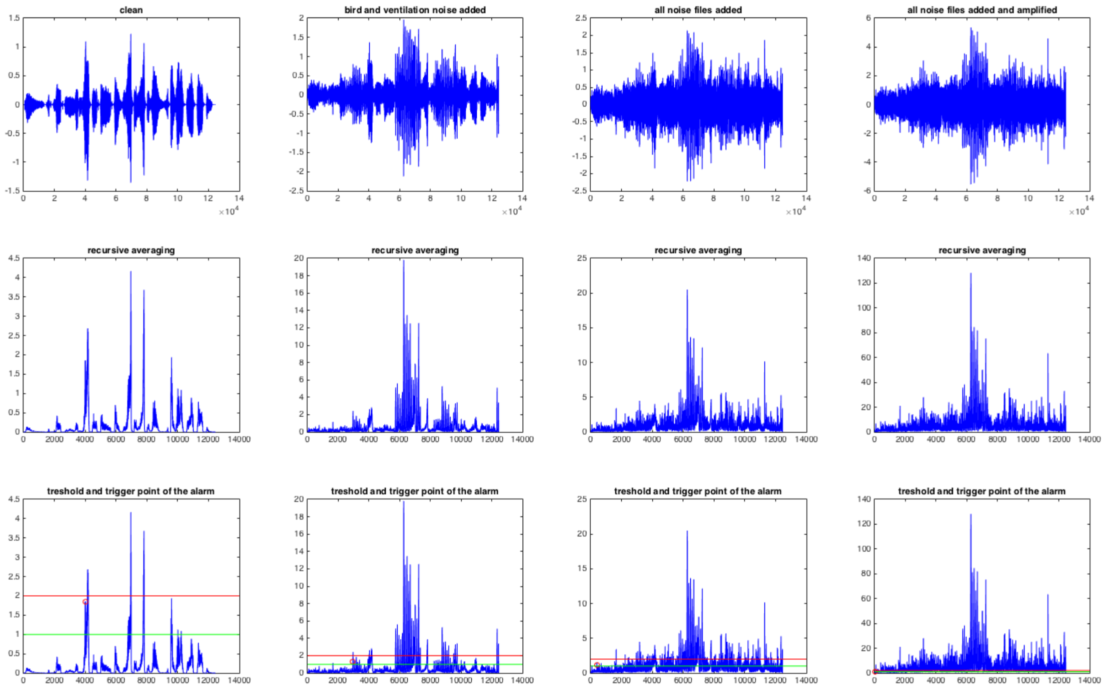


Figure 9: Amplified Baby crying1.wav, simple algorithm

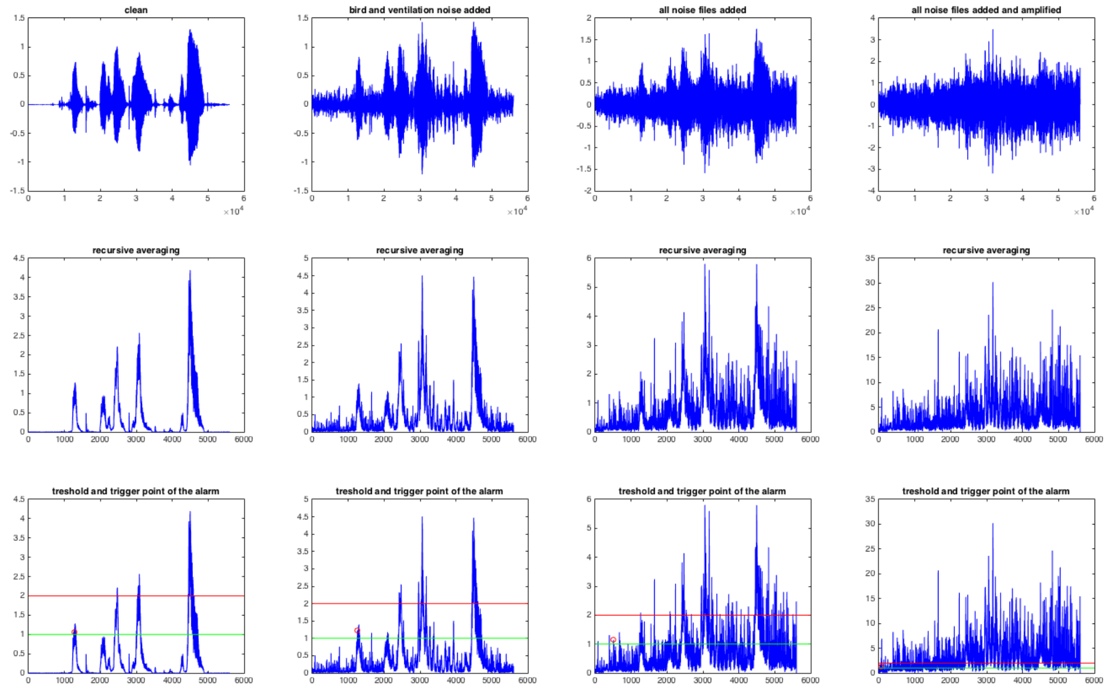


Figure 10: Amplified Baby talking.wav, simple algorithm

# 3 Appendix

## 3.1 A

```
% Values for the bandpass
A_stop1 = 60;
F_stop1 = 200;
F_pass1 = 300;
F_pass2 = 1300;
F_stop2 = 1400;
A_stop2 = 60;
A_pass = 1;
Fs = 8000;

% Design and create Bandpass filter
BandPassSpecObj = fdesign.bandpass(F_stop1, F_pass1, ...
    F_pass2, F_stop2, A_stop1,A_pass,A_stop2, Fs);
BandPassFilt = design(BandPassSpecObj,'butter');

% Graphical representaion of the bandpass filter
% Uncomment for usage, requires DSP toolbox for MATLAB
%  fvtool(BandPassFilt)
```

# References


[1] R. Cohen, Y. Lavner, *Infant Cry Analysis and Detection*, 2012

[2] E. Verteletskaya, K. Sakhnov, *Voice Activity Detection for Speech Enhancement Applications*, 2010

[3] N. Westerlund, M. Dahl, *Speech Enhancement using an Adaptive Gain Equalizer*, 2003

[4] R. Narayanam, *An Efficient Peak Valley Detection based VAD Algorithm for Robust Detection of Speech Auditory Brainstem Responses*, 2013

[5] J. Sohn, N.S. Kim, W. Sung, *A Statistical Model-Based Voice Activity Detection*, 1999

[6] O. Sutton, *A Statistical Model-Based Voice Activity Detection*, 2012

[7] TappyTaps, *https://www.babymonitor3g.com*,

[8] Sleekbit, *http://dormi.sleekbit.com/index.html*,

[9] D. Jankovic, M. Johansson, M. Lichota, *Adaptive Gain Control in Digital Signal Processors*, 2015