FOURIER & WAVELET METHODS FOR FINDING SPEECH ONSET LATENCIES

FOURIER & WAVELET METHODS FOR FINDING SPEECH ONSET LATENCIES

By IAN HORBATIUK, B.MATH

A Thesis Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements for the Degree
Master of Science

McMaster University MASTER OF SCIENCE (2011) Hamilton, Ontario (Psychology)

TITLE: Fourier & Wavelet Methods for Finding Speech Onset Latencies

AUTHOR: Ian Horbatiuk, B.Math (University of Waterloo)

SUPERVISOR: Professor Scott Watter

NUMBER OF PAGES: vii, 57

Abstract

Localization of speech onsets to determine onset latencies is a complicated problem with as many different methods for finding them as there are different areas which use such measurements. A majority of research performed in cognition uses a standard amplitude threshold voice key for estimating the speech onset latencies but a number of studies have shown that this method is incredibly inaccurate and can bias data or produce contradictory results. A number of alternative methods based on modifications to traditional voice-keys have been proposed to deal with the inconsistency although still show a number of deficiencies. Previous work has suggested that switching from the amplitude domain of a signal to the frequency domain a number of the issues present with voice keys can be overcome and when used in conjunction with a number of highly sensitive heuristics highly accurate onset latencies can be produced reliably under ideal conditions. This research is refined and paired with a new user interface to improve the ease of use and increase the adoption rate of this type of analysis. Recent work in the telecommunications industry also suggests that wavelet-based algorithms in conjunction with the Teager Energy Operator (TEO) can accurately detect speech even in the presence of noise. Four wavelet-based methods are investigated and tested; a simple wavelet transform test, and three methods using wavelet-packet transforms in conjunction with the TEO. Although these methods do not perform very well compared to traditional methods a number of potential issues with the implementation are discussed.

Acknowledgements

This project started off as the brain child of a student and his supervisor to avoid either tedious data coding or wildly unreliable results. After an initial effort resulted in an incredibly useful step forward before the project was shelved for a time to focus on more important research topics. The timing of this couldn't have been better – the recent economic downturn had just started to take hold and my restlessness with my job at the time had started to become apparent when my dear friend Peter came to me and put an idea in my head; an idea which he had also put in my soon-to-be supervisors head as well. Thankfully this incredibly interesting and challenging research problem had remained open for someone like me to come along and take up the gauntlet from where it had been left. So to (now) Dr. Peter Jansen, and Dr. Scott Watter whose creativity and curiosity started this project and the grants that helped fund it I owe the most profound of thanks for without them it would have remained a unrealized opportunity. They have both helped and supported me throughout the course of my time in the Cognitive Science lab as friends and mentors.

Of course I would be incredibly remiss if I neglected to thank the other members of my supervisory committee who have helped provide guidance over the course of development. Dr. Karin Humphreys and Dr. Tae-Jin Yoon have helped keep my research on track and asked many great questions to help keep me on track and thinking about what I was doing and what I would be doing in the future. I still don't quite know what I want to be when I grow up but I'm working on it.

To all the members of the lab – Amy-Beth, Sandra, Molly, Maria, Melena, Juliana, Esther, Kris, Lila and all the others who have helped along the way you have given me a place to bounce random, and usually bad, ideas off of, helped me design or test something or just given me a little distraction when I

hit the inevitable dead ends. With out your help and prodding I surely wouldn't have accomplished nearly as much as I have in the short time I had.

Like any good software project this has taken way more time than it ought to have, been over budget and is obviously full of bugs – features really. There is never enough time to get everything ironed out it seems; there is always something more to do, something to fix or something that could just be done better. Eventually development reaches a point where most of the major issues are worked out and someone could actually start using the software while the remaining issues get tagged as idiosyncrasies or a minor nuisances that simply have to be dealt with in the next version. That is unless you want to end up like Duke Nukem Forever – fifteen years in development, countless people involved in the project with far too much money to result in a harshly criticized gem. In the end you just hope that your hard work is appreciated by someone enough to use it once or twice. To those who do I hope it serves you well and please refer xkcd.com/730 and enjoy, I certainly did.

Table of Contents

Chapter 1: Introduction

Scientifically investigating the inner workings of the mind has posed a number of large difficulties that have required experimenters to be highly creative to overcome. Originally proposed by F.C. Donders in the 1860's, reaction time is one of the most commonly used measures of cognitive performance on a task but it was another 100 years before Sternberg (1969a) refined this method allowing experimental psychologists to began to use it as a dependent measure of cognitive process in earnest.

Response times are measured in a variety of different ways across many different domains in experimental psychology but one measure in common use is the latency of spoken responses. Most often this measurement is obtained from a voice key – a device connected to a microphone where the response time is generated when the amplitude of speech input exceeds a preset threshold. The voice key has been used to obtain response times in a number of different areas of behavioural research including, but not limited to, memory, attention, and emotion (Carroll and Young, 2005; De Houwer, 2004; Nino and Rickard, 2003). A study conducted by Rastle and Davis (2002) concluded that 95% of publications that used a spoken response latency measurement did so in conjunction with a voice key.

The voice key detection threshold is a delicate balance set by the experimenter to prevent false positive detections from occurring (possibly due to ambient noise such as computer equipment, ventilation, subject movement etc.) but still allow speech to be detected accurately. While in principle this seems like a good idea unfortunately voice keys have a fundamental flaw; speech onsets are not simple binary functions and do not abruptly cross a threshold. In most cases the speech onset will have occurred well before the voice key is actually triggered leading to onsets that are larger than they should be when compared to hand-coded onset latencies (Duyck, Anseel, Szmalec, Mestdagh, Tavernier, & Hartsuiker 2008; Kessler and Trieman, 2002).

A simple bias would be relatively easy for researchers to deal with when analyzing data but voice keys have been shown to be inconsistent – different phonemes will suffer different bias amounts and directions within and across speakers. These effects can dramatically change the results of a study and in the worst case can lead to conflicting interpretations as Rastle & Davis (2002) point out.

Although the problems associated with using voice keys have been known for many years (Kessler, Trieman & Mullennix, 2002; Pechmann, Reets & Zerbst, 1989; Rastle and Davis, 2002) simple, reliable methods for dealing with these issues have yet to gain widespread traction in the field. Some alternative approaches include modifications to

traditional voice keys – integrator voice keys, noise eliminating voice keys, and delayed trigger voice keys have been proposed. A number of software tools have also been proposed – Runword (Kello & Kawamoto, 1998), SayWhen (Jansen & Watter, 2008) as well as various schemes that monitors both amplitude and zero crossing rate (James, 1996) have been proposed as alternatives to a simple threshold voice key.

SayWhen is proposed as a step towards a more reliable solution for finding onset latencies that is easy to use and reliable. Instead of using amplitude thresholds or complex coding schemes, SayWhen uses a Fourier transform based analysis of the signal to determine onset latencies which has many benefits. As Jansen & Watter (2008) point out working within the frequency domain instead of the amplitude domain can allow low-amplitude signals (where the signal to be detected is on the order of the signal-to-noise ratio or SNR) that have strong peaks in the Fourier spectrum to be detected.

1.1 Phonemic Issues with Voice Keys

Voice keys are triggered when the sound pressure (amplitude) of input exceeds its threshold setting. Speech onsets are not simple – they are not a binary on/off condition and do not necessarily even rise continuously or linearly. This complexity has a large effect on the trigger point at which the voice key detects a signal and the resulting

response time. Kessler, Trieman & Mullennix (2002) present a compelling case for how phonemic differences alone can result in large differences in measured response times using standard voice keys. One primary reason for this is the manner in which sounds are articulated – the physical difference in how a sound is produced affects the time that it takes to reach sustained maximum amplitude. Among speakers producing the same sounds differences in the mechanical features (e.g. larynx, vocal tract) will result in acoustic differences that together act to produce large differences in phonemes; the average relative intensity of sounds in English can vary 22 dB (Fry, 1979).

One of the major problems of the voice-key detection methods is that they perform poorly on unvoiced speech where phonemes are not accompanied by vocal fold vibration (Rastle & Davis, 2002). As Rastle and Davis (2002) point out different categories of phonemes are detected with different levels of error – they are worst for fricatives (/f/, /s/, /v/, /z/), then plosives (/p/, /t/, /k/,/b/, /d/, /g/) and periodic consonants (/l/, /j/, /m/, /n/, /w/, /j/) , liquids, nasals, and semi-vowels. One might assume that this problem is limited to the first phoneme of a word but studies have shown that the situation is more complex than that and the second phoneme (among other factors) can influence when a voice key will trip on a particular word (Kessler, Trieman & Mullennix, 2002; Rastle & Davis 2002).

1.2 Voice Keys – Alternatives

Given that voice keys have a number of well documented issues that can drastically alter the conclusions of a research programme there have been a number of alternative methods proposed to produce more accurate response time data. One obvious solution that will produce the most accurate measurements is to visually inspect digital recordings of trial sessions and individually record the time it took for speech to begin following stimuli presentation. This gold standard of hand-coding would produce the most accurate times however it would do so at a potentially prohibitive cost in time. A number of alternatives have been proposed and are described below.

1.2.1 Integrator Voice Key

In an effort to research a curious discrepancy in results of two studies Rastle & Davis (2002) replicated a naming task to determine which was produced more quickly, words with simple or complex onsets. The discrepancy arose between two studies that used different methodologies to measure response times in naming studies. Frederiksen & Kroll (1976) used a voice key to determine that words with simple onsets (e.g. *sat*) were produced more quickly than words with complex onsets (e.g. *spat*) while a study by Kawamoto & Kello (1999) used a software algorithm to conclude the opposite.

In addition to a traditional simple voice key, this study used a device called an integrator voice key to produce response times for comparison. Results were also verified by visual inspection of waveforms. The integrator voice key adds a component which tracks the amplitude of a signal in addition to its duration so that signals that are below the nominal detection threshold but continue for an extended duration are still detected once their cumulative amplitude exceeds the detection threshold.

While the integrator voice key used in the study did show improvements in response times generated compared to a simple threshold voice key there was still a large difference compared to the visual inspection times.

1.2.2 Delayed Trigger Voice Key

Tyler, Tyler, and Burnham (2005) introduce a device called the Delayed Trigger Voice Key (DTVK) designed to overcome some of the short comings of a standard voice key. The strategy pursued was one designed to decrease the detection threshold of the voice key to be more comparable to the ambient noise level and incorporate additional triggering criteria to eliminate spurious detections.

In order to eliminate the early detections that result from lowering the detection threshold the delayed trigger voice key incorporates two additional detection criteria. First is a minimum signal detection – in order for the voice key to trip the signal must be above the detection threshold for a specified duration which can be anywhere from 5 ms to 200 ms. The second parameter is a minimum silence duration which is required because audio signal are oscillatory in nature and constantly switch between positive and negative values and any signal would almost immediately drop below the critical detection threshold and not satisfy the minimum signal duration. In order to compensate for this the signal is permitted to fall below the detection threshold for a certain amount of time which is controlled by the minimum silence parameter before returning above threshold; this duration can be anywhere from 2-30 ms. In order to reduce the below detection threshold period the signal is full-wave rectified which allows lower minimum silence duration values and should allow non-speech sounds to be rejected more successfully.

The DTVK produced results that were near hand-coded values for a number of different onsets (including simple and complex onsets) but did not perform as well on fricatives and was only tested with monosyllabic words – not a spontaneously generated speech set.

1.2.3 Noise Elimination Voice Key

Duyck et al. (2008) propose modifications to the standard voice-key measurement scheme to improve the accuracy of onset detection. One of the goals of their voice-key modifications was to eliminate any possible operator induced biases that may occur from user selectable equipment settings (e.g. detection threshold level) and towards this end created the Noise Elimination Voice Key (NEVK). The NEVK does not have any user manipulable parameters in it as detection scheme used begins by first amplifying a speech signal by a factor of 20, then filtering this with high and low pass filter stages before being massively amplified (by a gain of about 2000). Following these processes the NEVK examines the duration of signal spikes operating on the principle that noise (clicks, lip smacks etc.) will be of short duration (50 ms or less) and can be ignored so any signal which has oscillations longer than this window will be considered signal.

This hardware solution to the problem appears to perform quite well across a number of different conditions that traditional voice-keys do not although the general category of fricatives still appears to be the largest deviations from the hand-coded times.

1.2.4 SayWhen

As an alternative to traditional voice key methods Jansen and Watter (2008) proposed an offline, software based, solution to the problem; SayWhen. SayWhen is a nearly complete software package for analyzing continuous speech production experiment data that allows users to view and inspect trials to produce accurate onset latencies. The algorithm used is composed of a number of processes that first work to detect a signal using Fourier analysis, and then determine if it is a signal of interest to the experimenter.

SayWhen showed large improvements over traditional voice-key measurements using a different method for analysis from the other solutions presented and provided useful feedback on the accuracy of the response times all within a relatively easy to use environment.

Chapter 2: SayWhen & SayWhen 2

Although discussed above in a general sense the SayWhen algorithm is described in detail below. It is composed of a number of processes that first work to detect a signal using Fourier analysis, and then determine if it is a signal of interest to the experimenter. The first stage of this process is a candidate signal detection, followed by a short-duration noise detection heuristic, a discontinuous phonation detection stage, frequency look-back-through-time, and finally problem trial tagging.

2.1 The Fourier Transform

The Fourier transform and its computational implementations (most notably the Fast Fourier Transform, or FFT) has been covered extensively in any number of easily accessible places and for brevity will not be repeated here in detail. The Fourier transform hinges on the principle that any signal can be reconstructed exactly using an infinite series of sums of sine and cosine functions. This result is certainly a groundbreaking and important result in mathematics but it poses a certain practical problem in that it is not possible to use an infinite number of functions to represent a signal in any real applications; approximations must be used. The sine and cosine basis functions used in

the transform have properties which can be useful in analysing signals – these functions have precise frequency values and amplitudes that can help us understand what a signals content is.

The understanding associated with these properties comes at a price however as a problem associated with the Fourier transform is that although it can perform frequency localization quite well it does not provide very good information about the temporal location of an event. The sine and cosine functions the Fourier transform uses as its basis have a precise frequency but do not have a precise location; the sine and cosine functions persist across all of time or space. This characteristic flaw is typically compensated for by performing the short-time Fourier transform (STFT) – a discrete Fourier transform limited to a windowed subset of our signal. In the STFT the time support of the transform is equal to the length of the window but this comes at a cost of frequency resolution; the shorter the window the less able it is to discriminate frequency separation.

2.2 The SayWhen Algorithm

The SayWhen algorithm begins with a candidate signal detection process before passing through a number of heuristics to refine the signal position. Candidate signal detection begins by converts the input signal from the amplitude domain to the frequency domain

using the Fourier transform (as implemented by the Fast Fourier Transform, as the STFT) at which point the algorithm then begins scanning the input in small windows for any 'interesting' frequency peaks. This peak-searching behaviour is much like a voice key in that it requires that the input pass a threshold before proceeding on to subsequent stages. Once this occurs the algorithm reduces the size of the scanning window and continues scanning the input until the window size reaches a preset lower bound (~ 16 samples or 1/3 msec) and the input still exceeds the detection threshold. The index of the candidate input signal is then marked for the remaining heuristics to process which are all designed to help classify a signal as a speech onset or noise.

The short-duration noise detection heuristic functions on the principle that most phonemic onsets in speech signals have quickly increasing amplitudes that are sustained for large temporal regions where transient noise is often present for only a very short time window. This heuristic sums the total change in the amplitude of a signal over a small temporal distance and compares the result to a preset noise threshold – if the total amplitude change exceeds this threshold then it determines that the current input is part of a speech signal and not noise.

Discontinuous phonation detection is designed to detect onsets that may have been previously classified as noise – in particular low-amplitude gradual onsets such as /s/ or

/f/ and some velar onsets such as /g/. To do this the temporal location of the candidate signal is compared to the locations of other potential speech components initially classified as noise. If the noise precedes, and is sufficiently close to the candidate signal then the algorithm will re-tag the detected onset location using the earlier onset.

The final step in the detection algorithm is to use a reverse scanning technique similar to the initial detection phase to refine the final onset position. Using a small window the algorithm scans backwards in time from the detected onset while computing the Fourier spectrum for each iteration and comparing these results to the spectrum of the initial onset. The final onset position is determined at the point when the frequency peaks that were being monitored drop below the threshold for detection.

The final stage is tagging any trials where the speech onset may not have been detected correctly. These are cases where the onset is either extremely short or extremely long or the trial had a low amplitude onset (determined by the average amplitude over the input following the onset for some short time and comparing to a threshold that is typically twice that of the short-duration noise threshold).

2.3 SayWhen 2

The SayWhen 2 algorithm is based on original SayWhen algorithm but includes a few changes in implementation and support new modes of use. One of the major goals with this version of the software was to support data formats where the input file contains only a single trial with no onset marker as well as the same multi-trial session format where individual trials are separated by the same onset marker used in Jansen & Watter (2008) – a 1 kHz square wave injected into the left channel of a stereo recording.

2.3.1 *Finding the Onset Marker*

The trial onset marker used by SayWhen is also used for detection in SayWhen 2 – a square wave with a 1000Hz fundamental frequency. SayWhen 2 finds these markers using a power spectrum matching method. As the program walks through an audio file it checks the left channel audio data power spectrum to see if a given section contains any signal that consists primarily of the 1000 Hz fundamental and the first few harmonics of the square wave (3000 Hz, 5000 Hz, 7000 Hz, etc) by ensuring that the largest peaks in the power spectrum occur at these (approximate) frequency values.

2.3.2 Finding the Signal

Prior to beginning the signal detection portion of the algorithm SayWhen 2 uses the first ~

100 ms of the file to produce estimates for the detection parameters. Rather than using a single set of parameters for all speakers and sessions (where noise conditions will almost certainly vary) updating the signal detection parameters should allow the signal detection process to be more accurate. This process produces unique detection parameters based on the background noise in the trial based on the settings within the SayWhen 2 interface. The estimated parameters are background noise estimate, detection threshold, minimum detection threshold and a frequency check-back detection estimate.

In the 'no onset marker' mode signal search begins at the file beginning, otherwise it begins shortly after the detected onset marker position. The signal search algorithm is a threshold – however as in the original SayWhen this thresholding occurs in Fourier space. Beginning from the search start location the algorithm looks at chunks of data (in powers of 2, starting with 1024 samples or ~23 ms of audio per channel) and performs a Fourier Transform (a STFT) on each chunk using a rectangular window. The absolute values of the transform output are then searched for any which exceed a preset initial detection threshold value.

If the window contains any transform values that exceed the initial detection threshold than the search algorithm pauses at the current location and rescans the current window with a reduced window size. Given the a priori knowledge of an initial detection, if no

detection is found in the first rescan pass then the detection threshold is reduced and the window rescanned again. This process continues until either a potential signal is detected at the minimum window size (16 samples) or the minimum detection threshold is reached and no potential signal found. This process is essentially the same as in the original SayWhen.

2.3.3 Short-duration Noise Detection

After a potential signal is found the SayWhen algorithm performs a short duration noise check by computing the average signal level out to a short distance from the detected potential signal. If the average signal in this window is below the noise threshold level then the algorithm makes note of the potential signal location as probable noise and continues searching for another potential signal beginning from the previous detection. If the average value exceeds the noise detection threshold then the search algorithm is terminated and the algorithm has classified the current set of detections as a signal whose position needs to be refined.

The final signal position is determined following a number of refinement heuristics beginning with a glottal stop check which goes back through the list of previous potential detections to determine if the current detection is part of an earlier detection that may

have been classified as noise. If there is a closely grouped set of potential signal detections the heuristic will successively move the onset location earlier until it reaches a large break (1250 samples, approximately 30 ms) or the first detection location.

After the glottal stop check is completed the final step is to perform a look-back through time from the current signal detection to find the precise onset location. In a similar fashion to the initial detection method the algorithm works its way backwards in small steps (32 samples) checking the Fourier spectrum of the signal for components above the check-back detection threshold. Once there are ten consecutive transformed windows where the maximum observed Fourier coefficient is below the threshold the signal is classified as having faded and the initial location of the fade becomes the final detected onset location.

One key difference between the search algorithm for multi-trial sessions and single trial sessions is that the multi-trial search proceeds through the file sequentially searching for an onset marker and then searches a limited window for a signal determined by the upper limit of what constitutes an 'acceptable' but 'slow' response time to a trial; the default value is set to 2500 ms – the window searched is approximately double this size. This can lead to situations where no response is found for a given trial if the response occurs outside of this window.

For situations where no onset marker is used in the recording a similar process is followed; the onset marker detection stage is skipped and the beginning of the recording is used as a marker and the response time is calculated relative to this.

2.4 Experimental Results and Performance Evaluation

The original SayWhen software was evaluated using experimental data generated in the Cognitive Science laboratory at McMaster University where subjects performed speeded spoken free word association with a visually presented word stimuli. Response times generated by the software were compared to those generated by a standard voice key as well as hand coded times produced by trained undergraduate research assistants from digital audio recordings of the trials. This dataset was also used to verify SayWhen 2's performance.

24 subjects (5 male, 19 female) were used as a performance verification for our Fourier algorithm comprised of approximately 168 individual trials each or 4017 total trials. Of these trials ~ 40.1 % (or 1608 of 4017) of all trials were within +/- 5 ms of the hand coded response time, while a further ~15% (or 585 of 4017) were within +/- 10 ms of the true

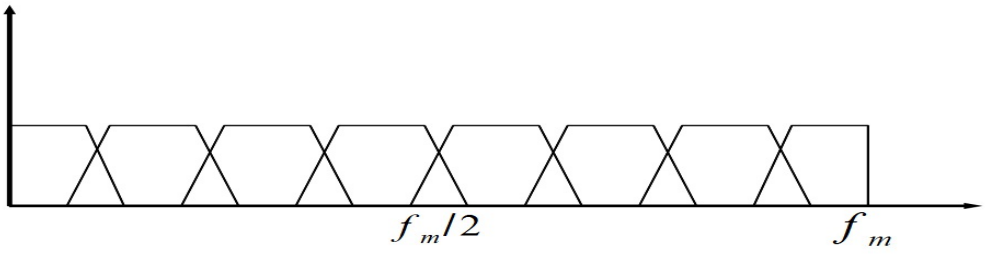

Figure 1: Frequency separation of a 3-level wavelet packet decomposition

onset latency (~55% total within +/- 10 ms). The percentage of RT's that fall within 5 ms of the true onset varies from 24% to 55% among individuals. Approximately 30% of all trials disagreed with the hand coded time by more than 25 ms here which is nearly double that of Jansen & Watter (2008); approximately 13% (537 trials) were between 100 and 1000 ms and a further 5% (or 180 trials) were a "No signal found" condition which was not present previously.

This fully automatic mode performance presented here is not quite as robust as the original SayWhen presented by Jansen & Watter (2008) although still represents an improvement over traditional voice keys where accuracy was ~33% of all trials tagged within 5 ms and 41.7% fell within 10 ms of the hand coded value.

Excluding trials flagged (1595 trials, flagged as either being too short, too long, not found

or low amplitude onsets) does not significantly change the distribution of response times generated. Of approximately 1600 trials initially tagged as being within 5 ms of the hand coded time approximately 600 of these were flagged, while 200 of the approximately 600 trials between 5 ms and 10 ms were removed via this method.

Of the approximately 800 trials flagged as being more than 100 ms different from the hand coded response time a majority of these (~65%) were identified as having initial phonemes of either an unvoiced stop or fricative /s/, /h/, /f/, /t/, /k/, and /p/ [1]. Phonemes of these types are characterized as being produced without any periodic phonation occurring in an absence of vocal cord vibration and are similar to noise. This leads to a period of near silence typically in the range of 70 – 140 ms long (Fry, 1979).

SayWhen 2 also features the capability to analyze individual trials that do not contain trial markers. This feature was verified using a subset of the dataset described above. A total of 220 individual trials were cut from 12 subjects datasets and hand-coded before being analyzed. These data showed an interesting trend when compared to the multi-trial set for the same parameter settings – typical response times in the single trial schema were approximately 50 ms earlier than the hand-coded values however the distribution of these

1 Phonemic transcriptions were completed using the UWA MRC Psycholinguistic Database (available online : http://www.psy.uwa.edu.au/mrcdatabase/uwa_mrc.htm, August 2011)

trials remained the same. This would necessitate a small adjustment to the detection parameters to decrease the sensitivity; this difference can likely be accounted for due to more accurate local noise estimates obtained at the beginning of every trial in this scenario when compared to the multi-trial scenario.

A note should be made that 3 subjects initially showed abnormal response time results when compared to hand coded times resulting from the parameter estimate stage of the Fourier algorithm. The initial parameter estimates are taken from the first ~ 100 ms of audio which are assumed to be silence/background noise; if there is an abnormal amount of noise at this stage (a cough, heavy breathing, an onset marker etc) then it may affect normal trial detection. If the recording begins with instructions these should be carefully trimmed to leave at least 100 ms of 'quiet' audio before the first trial begins. Adjusting the initial portion of the audio file to remove the noise produces a normal response time profile for the session.

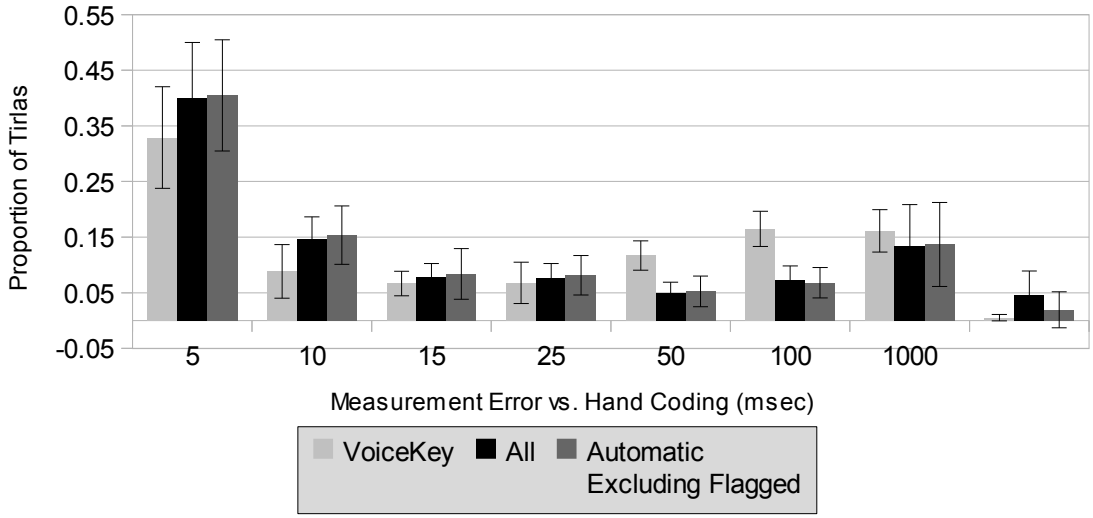

Figure 2: Histogram of proportions of speech onset measurement error vs hand coded times. A standard amplitude threshold voice-key (light grey), SayWhen 2 fully automatic mode (all trials, black), SayWhen 2 automatic excluding all flagged trials (dark grey).

Chapter 3: Wavelet Based Analysis

Wavelet methods are being explored for a wide variety of applications in telecommunications systems – some of the uses include signal compression, speech detection and speech recognition. The wavelet transform can be computationally simple and provide an adaptable time/frequency resolution that the Fourier transform cannot which leads to a number of exciting analysis possibilities. Following a brief description of the wavelet transform and some of its useful mathematical properties a few exploratory algorithms based upon results from telecommunications literature will be outlined and experimental results on a subset of the SayWhen 2 data set will be presented.

3.1 The Wavelet Transform

The wavelet transform is an alternative to the Fourier transform which inherently has the ability to address one of the major short-comings of the standard STFT – a fixed frequency resolution for a given window size. The wavelet transform has a multi-resolution capability which allows low frequency but high temporal accuracy or high frequency but low temporal accuracy and is at the heart of its utility.

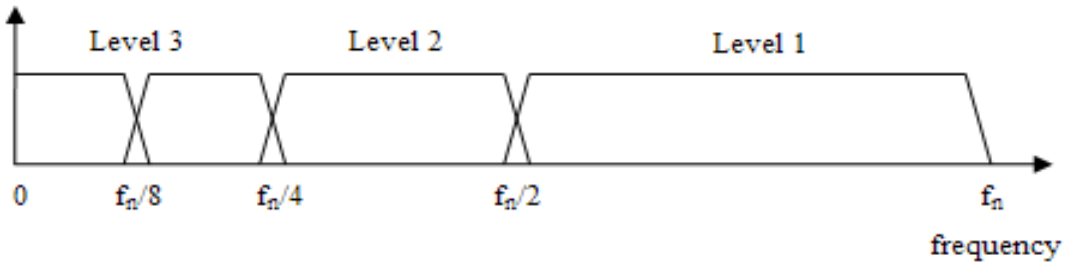

Figure 3: The frequency separation of a 3-level wavelet decomposition

A wavelet transform is the result of contracting or dilating a mother wavelet which adjusts the frequency response and then by shifting it over the signal to be analysed (Ghanbari & Karami-Mollaei, 2005). The wavelet transform can be efficiently implemented as a series of quadrature mirror filters as described by Mallat (1999). This filter bank implementation allows us to perform a wavelet transform by passing a signal through a high-pass and low-pass filter combined with a down-sampling operation to obtain a set of detail and approximation coefficients which represent the signal. These detail and approximation coefficients are the low-scale, high frequency and the high-scale, low frequency components of a signal respectively.

Processing the coefficients can be repeated a number of times to produce a finer scale frequency resolution representation of the signal. In a standard wavelet decomposition the approximation coefficients are successively broken down producing small frequency windows at low frequencies and larger windows at high frequencies. Because the signal is

down-sampled by a factor of 2 each time the length of the original input N should be a power of 2 and the maximum depth L which a signal can be decomposed satisfies $N=2^L$. A wavelet packet decomposition is similar to a standard wavelet decomposition but at each level both the approximation and detail coefficients are transformed which results in an even frequency range distribution between the wavelet coefficients.

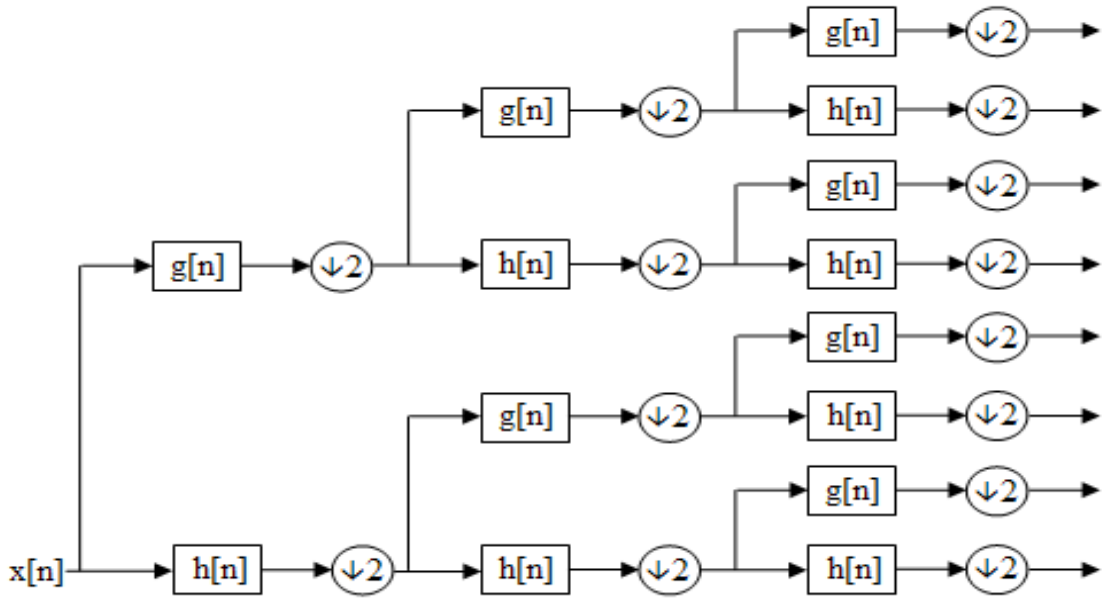

Figure 4: A wavelet packet decomposition over 3 levels of a signal x[n], where h[n] is the high pass detail coefficients, g[n] is the low pass approximation coefficients. The output of each filter is downsampled by a factor of 2 and used as the basis for another wavelet transform.

Burke-Hubbard (1998) points out that even with a large number of available wavelets most users stick to a few common ones – the Mexican hat, Morlets and Daubechies.

Choosing an appropriate wavelet to use can be difficult given the number of available possibilities. The frequency localization abilities of a wavelet is related to the number of vanishing moments the wavelet has; the more vanishing moments the wavelet has the better the frequency resolution. This comes at a cost however in that the more vanishing moments a wavelet has then the wavelet is less able to resolve similarities between the wavelet and the signal at low frequencies. The wavelet chosen to investigate the potential usefulness of this type of analysis was the Daubechies-4 (or D4 as it is sometimes noted) wavelet, one of the most commonly used wavelets that has compact support and relatively low number of vanishing moments (Daubechies wavelets D2, D4, D6 etc. have 1, 2 and 3 vanishing moments respectively due to their specific construction by Ingrid Daubechies). Compact support is a desirable mathematical property for a function to have that provides for 'nice' behaviour in a compact region and vanishes at infinity.

3.2 Wavelet Algorithms for Detecting Speech Onset Latencies

All wavelet based algorithms have roughly the same general structure: a wavelet transform is applied to an input signal, the resulting wavelet coefficients are processed somehow and then a VAD decision can be made on each frame or the coefficients can inverted to obtain a 'clean' signal frequently called the voice activity shape (VAS). This clean signal can then be searched to determine the presence of speech using a variety of

methods or simply returned as the desired output.

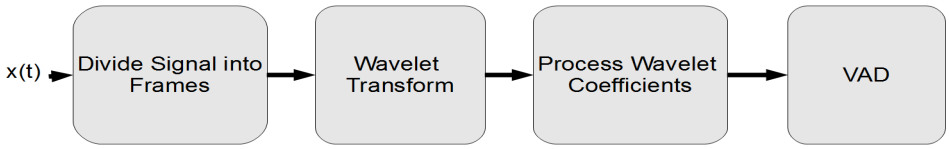

Figure 5: A general wavelet based VAD scheme

During analysis an input signal is typically divided into frames whose length varies from anywhere between 10 ms to over 30 ms (Bahoura & Rouat, 2001; Chiodi & Massicotte, 2009; Ghanbari & Karami-Mollaei, 2005) which are then classified either as containing speech or not. The relation between sampling rate and frame size means that using this methodology achieving a 5 ms accuracy at 44100 Hz would require ~ 256 samples as a frame size while at 16 kHz this is down to 80 samples. This may be an issue during the wavelet decomposition stage depending on the depth of the transform. Most of the works cited here are analyzing signals at a lower sample rate (8 kHz, typical telephony sampling rate and provides a bandwidth of 4 kHz) which contain less spectral 'depth' as a result of

the Shannon Sampling Theorem. The frequency resolution of a transform is related to the sampling frequency and the number of samples used:

$$(1/N)*f_s \quad (1)$$

where N is the number of samples in the transform and f_s is the sampling frequency. At this rate the standard wavelet based algorithms have a frequency resolution of ~ 31 Hz while a transform of 256 samples of audio sampled at 44.1 kHz has a frequency resolution of ~172 Hz. Wavelet analysis lets us adjust the resolution of the window used to capture different scale events from low frequency but highly accuracy in time or high frequency but low accuracy in time.

The algorithms considered here make use of either a wavelet packet transform or perceptual wavelet packet transform (PWPT), a variation on the wavelet packet transform. The PWPT is designed to match a psycho-acoustical model of hearing in creating critical frequency subbands (Carnero, Drygajlo, 1999; Pinter, 1996) in order to provide a better representation of the signal. In a standard WPT decomposition the subbands are all of equal sizes – equal number of wavelet coefficients in the signal representation whereas in the PWPT decomposition this is not the case; the number of coefficients depends on the subband.

Using an initial frame size of 256 points at a sampling rate of 8 kHz with a 5 level PWPT results in 17 total subbands ranging in size from 8 sample coefficients to 32 coefficients in the low to high frequency bands respectively while a 1024 point transform of a 44.1 kHz audio sample produces subbands that range in size from 32 to 128 sample points.

Typically these algorithms performance are compared to the telecommunications industry standards (e.g. G.729b or GSM) for the probability of correctly identifying a frame as containing speech vs the probability of false alarm or the probability of correctly detecting noise/inactive are considered. In speech enhancement tasks the level of noise reduction and speech preservation is often used although Bahoura & Rouat (2001) consider an improvement in SNR when compared to a noise corrupted signal.

One consideration in the design of these (and many other wavelet based algorithms) is that the noise is typically considered to be white or Gaussian white noise (Ghanbari & Karami-Mollaei, 2005; Shaojun, Haitao & Fuliang, 2004) which corrupts the entire signal whereas many real world noises are of short duration, high amplitude signals. While models are tested against real noise samples the models themselves are not usually built around assumptions regarding them. These assumptions would typically be embedded in the threshold calculation and result in a universal threshold for wavelet coefficients which

may work well for white noise but not for coloured or real noises (Wang, Chin & Tsai, 2009). Separation of results from real-world samples and those corrupted by white/Gaussian-white/pink noise is not usually considered.

3.2.1 The Teager Energy Operator

After the initial decomposition stage typically the TEO, often touted as a 'powerful non-linear operator', is applied to the resulting signals (Bahoura & Rouat, 2001; Chen & Wang, 2002; Chiodi & Massicotte, 2009; Ghanbari & Karami-Mollaei, 2005). The TEO captures trends in the energy of a signal and is used to 'enhance the discriminability of speech' (Bahoura & Rouat, 2001; Chen & Wang, 2002;) and reduces non-periodic or transient trends in the signal while preserving periodic components (Chiodi & Massicotte, 2009).

The Teager Energy Operator for a discrete signal y(n) is defined as:

$$\psi(y(n)) = y(i)^2 - y(i-1) * y(i+1) \quad (2)$$

After the TEO is applied to the initial wavelet coefficient subbands a number of possible

operations can take place. Bahoura & Rouat (2001) apply a further smoothing mask and then apply a time-adaptive threshold to the smoothed signal before inverting to get back their enhanced signal. Wavelet coefficient thresholding (WCT) is a common operation where small coefficients in the output of a wavelet transform are set to zero. If the wavelet coefficients are then inverted the resulting signal is compressed from the original.

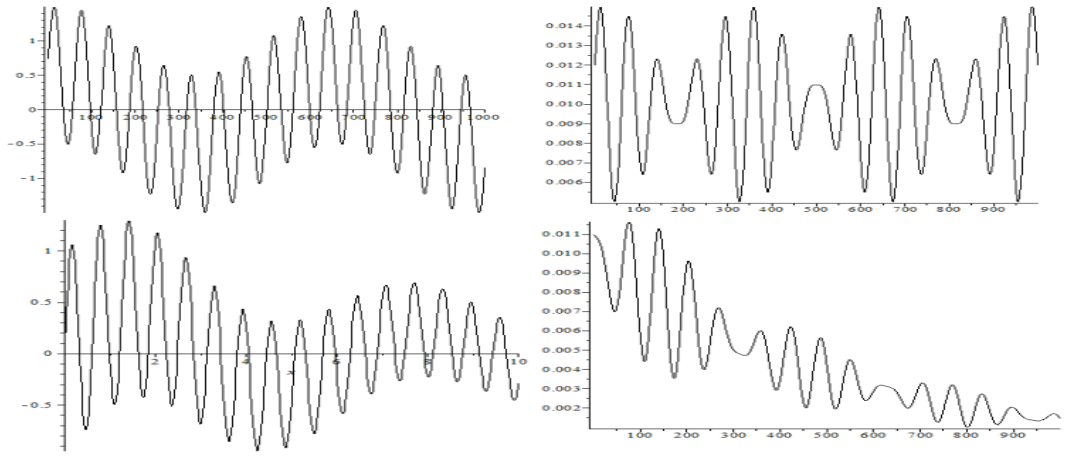

Figure 6: The TEO applied to two sinusoidal functions

3.2.2 Method 1:

As an initial investigation avenue a standard 1-level wavelet transform was considered

along with a rough density measure to determine the presence of speech. Using an initial frame size of 1024 samples a standard wavelet transform is applied to the signal and the resulting approximation coefficients undergoes a thresholding procedure to retain any that are 2.5 standard deviations above the frame mean. The density of a frame is computed as the number of nonzero coefficients remaining after this process; if the frame density is > . 5 then the frame is flagged as containing speech. The speech onset is then determined as the first nonzero entry in the frame and then converted to the original position in the signal. Using a Daubechies 4 wavelet the onset is determined within +/- 8 samples of the actual signal position with this method, or at 44100 Hz sampling rate this produces a possible error of < 1 ms difference.

3.2.3 Method 2:

Using a 5 level wavelet packet transform (with 1024 samples this would produce individual subbands of 32 wavelet coefficients) and then the TEO is applied to each of the 32 subbands of the transform. Following the noise removal process a threshold is computed according to eq. 15 from Ghanbari & Karami-Mollaei (2005). This threshold λ uses a statistical property computed on the wavelet decomposition called the Median Absolute Deviation (MAD) as a foundation

$$\lambda = \sigma_{(j,k)} \sqrt{(2 * \ln(N_j))} \quad (3)$$

where

$$\sigma_{(j,k)} = MAD_{(j,k)} / 0.6745 \quad (4)$$

and the Median Absolute Deviation is the median of the absolute deviations of a dataset from the dataset median and N_j is the number of coefficients in the j-th level .

Any values in the decomposition that are below the threshold λ are set to 0. The remaining coefficients are then inverted to obtain a VAS like curve for the frame. A signal detection threshold is determined using a rolling mean of the VAS curve, when the absolute value of the inverted sample exceeds approximately 100x the rolling mean it sets a flag indicating we may be in signal – when 100 (approximately) consecutive sample values exceed the threshold value the initial position where the flag was set is returned as the signal onset.

3.2.4 Method 3:

The third method attempts to use methods suggested in Wang et al. (2009) where energy measures are used to make a voiced/unvoiced/noise decision on a frame. During wavelet-

based compression schemes a threshold is chosen to allow wavelet coefficients to be zeroed which reduces noise and the amount of data required to represent a signal. Unfortunately unvoiced portions of the signal may have a larger number of coefficients reduced than desired due to spectral similarity to noise (Ghanbari & Karami-Mollaei, 2005) which removes components of speech and reduces the qualitative and quantitative quality when the speech is inverted.

Comparing the energy across different levels and subbands is proposed as a method to determine if a frame is either a voiced or unvoiced speech segment or if it contains noise/silence. If the ratio of energy in the level 1 approximation (low-pass) coefficients is more than ~90% of the full-band energy than the frame is considered to be a voiced speech component.

The unvoiced components are determined according to the following: compute the sum of the energy of the first four subbands, the sum of the energy of subbands 5-8, and subbands 9-15.

$$\begin{aligned} E_0 &= \sum e_{ij}, i = 1..4 \\ E_1 &= \sum e_{ij}, i = 5..8 \\ E_2 &= \sum e_{ij}, i = 9..15 \end{aligned} \tag{5}$$

where e_i is the energy of the i-th subband, computed as:

$$e_{ij} = \sum d_k * d_k \quad (6)$$

for d_k , the wavelet coefficients of the i-th subband in the j-th level of the wavelet transform.

If $E_2 > E_1 > E_0$ & $E_0 / E_2 < 0.99$ the segment is unvoiced otherwise it is noise.

Wang, Chin & Tsai (2009) use a perceptual wavelet packet decomposition which does produce a different set of wavelet coefficients corresponding to the difference in tree structure between the PWPT and the WPT however due to the conservation of energy principle (Walker, 2008) the energies are the same when the subbands are added together.

3.2.5 Method 4:

Chiodi and Massicotte (2009) use a typical PWPT and TEO scheme to decompose their signals for analysis and enhancement but instead of thresholding individual wavelet coefficients they compute a frame variance measure which is the sum of the subband variances. These variance measures are used to compute a VAS like curve for the signal where the minimum threshold for speech/non-speech decision in the VAS curve is based upon the maximum value of the first 10 VAS values which are assumed to be non-speech frames.

This scheme is modified slightly for this analysis to work within a standard WPT scheme and not a PWPT system. The potential speech signal is decomposed using a 5 level WPT and the frame variance measure is computed. An initial variance estimate is computed from the first 4096 sample values in a signal; any frame variance measure which exceeds the initial variance estimate by 2.5 standard deviations is selected as a speech signal, the first such incidence where this occurs is returned as the speech onset.

3.3 Experimental Results and Performance Evaluation

As mentioned previously the wavelet methods described are designed to detect speech vs. non-speech conditions and are not designed explicitly to detect speech onset latencies. The purpose of the investigation here was to determine in what ways (if any) they may be useful and what effort would be required to make them find precise latencies if possible.

The wavelet based methods performance was evaluated using a subset of the trials used to verify the Fourier analysis method. For methods 1, 2 and 4 response times were compared to the hand-coded values where for method 3 the goal was to determine if a distinction between voiced/unvoiced/noise/silence could be made and used to accurately locate onsets.

3.3.1 Method 1

Method 1 results were promising as an avenue for further research despite the simplicity of the method used. Approximately 20% of trials (42 trials) were detected within 5 ms of the hand-coded latency, a further 15% (35 trials) within 10 ms. The remaining trials were distributed with a decreasing linear trend (the regression equation for the set is given by: $y = 0.19522 - 0.01706 * x$) with a total of 35% of all trials differing by 25 ms or more. The effectiveness of this method is surprising given the simplicity of it; at this level the approximation coefficients of the transform easily represent nearly the entirety of the speech signal with the bandwidth of the coefficients for a 44.1 kHz signal being larger than the entire bandwidth of a typical telephony signal (8kHz). It should be noted that the position returned with this method is the position in the thresholded and inverted signal and should be considered exact.

3.3.2 Method 2

This method performs a wavelet-packet transform to produce 32 subbands which are 'cleaned' using the Teager Energy Operator before having subband thresholds computed and used to reduce the small coefficients further before being inverted where an adaptive

threshold is computed to determine where a signal may be present.

This method resulted in frequent 'no signal found' conditions that make it seem unsuitable for use in detecting speech, much less accurate onset latencies. Potentially this could result from a number of sources within the algorithm – first it may be possible that the TEO and subband thresholding operations simply remove too much signal with the noise to produce accurate detections. Another possibility is that adaptive threshold may not be appropriate. The choice of threshold was motivated by the observation that the inversion of zero-valued coefficients should produce very small (allowing for numerical error) signal values whereas the coefficients that survive the zeroing process should be several orders of magnitude larger.

Using a threshold for the inverted signal of 10x the rolling mean as our trigger condition produced signal detection almost immediately in every trial while using a threshold of 100x the mean produced a 'no signal found' condition in nearly all of the trials. Even a threshold of 20x the rolling mean produced a high 'no signal found' condition with 15% of all trials producing this while nearly 60% of trials fell between 100 and 1000 ms different from hand-coded values.

3.3.3 Method 3

The attempt to calculate the voice/silence/noise condition performed extremely poorly, although it is unclear why, with nearly every frame was detected as being voiceless regardless of the actual frame content.

3.3.4 Method 4

This method calculates a frame variance measure in conjunction with the median absolute deviation thresholding. Any frames that contain speech should have high variance compared to frames that contain noise or silence. Initial performance of this method is quite poor with only ~10% of trials detected within 5 ms of the hand-coded time and a further 3% within 10 ms. A large majority of trials were detected as being between 100 and 1000 ms different, tending to be detected very early.

This is likely caused by the use of the Teager Energy Operator as a method for reducing noise due to its ability to decay signal components that are either transient or not periodic. While it seems initially counter-intuitive that the removal of non-periodic components of the signal would cause earlier detection, the TEO has the effect of reducing variance in the signal. This reduction results in the variance estimate that is calculated for the

detection threshold during the parameter estimation stage as being far too low. The variance and the standard deviation of the variance estimates are calculated using the same algorithmic process as the detection algorithm and this results in them being highly similar and typically quite small. In subsequent detection processes the variance threshold is surpassed very early while the input is still background noise.

3.4 Discussion

While the wavelet methods pose interesting research avenues there are several concerns which need to be addressed in future research. The full effect of the Teager Energy Operator on speech should be quantified as well as its ability to reduce transient noise – it seems likely that it has an unintended effect of removing any speech components that may be noise-like such as the problematic initial phonemes described during the Fourier analysis. The second major concern is addressing the accuracy of the methods; they are all designed to detect speech/non-speech frames and not where in the frame speech actually occurs. The relationship between analysis parameters and frame size is such that if we want to consider information in lower frequencies (such as those typical in speech) the frame must be larger than if we wish to consider frequencies in the high range. Our sampling rate of 44.1 kHz is quite high compared to typical telephony rates which means that to obtain comparable frequency resolutions we must resort to larger frame sizes

which in turn reduces the accuracy of these systems.

Chapter 4

SayWhen 2, the next stage of the SayWhen development process, is presented as a tool for use in the automatic detection of speech onset latencies. Both use a Fourier based analysis to produce highly accurate measurements of speech onset latencies from recorded audio data this version includes several new features and enhancements to improve its usability. A large portion of the software development was dedicated to producing a new user interface for the package to feature the ability to analyze individual trials in an experimental session in conjunction with modifications to the detection algorithm. The new interface also supports multiple operating systems giving researchers more freedom when it comes to analyzing their data while still providing the ability to inspect and adjust automatically produced response times.

The observed performance represents a step between the original SayWhen performance and that of a traditional voice-key although it is not fully clear as to why this is. In fully automatic mode performance within 10 ms was approximately 55% of all trials where in the original SayWhen it was closer to 70% and compared to 40% for a voice-key. Unfortunately when excluding flagged trials the overall performance does not increase as substantially as it did in the original SayWhen. Specifically a large number of trials in the

verification process were flagged as having low-amplitude onsets but this is based on a somewhat arbitrary notion of what low-amplitude is. The system used in SayWhen and SayWhen 2 both use a universal fixed threshold for what constitutes low-amplitude which cannot be appropriate for all speakers.

This means that individual trials are being tagged accurately despite being flagged for possible operator attention. If the base accuracy was higher this tendency would be a reassuring trait.

One possible source is the fundamental structure of the algorithm changed between the two versions – the original algorithm made several passes through the data – initially finding onset markers, then searching a small window following the onset marker for signals where as SayWhen 2 finds onset markers and signals successively using one pass through the audio recording. Alternatively there may be implementation specific details which can account for the differences between the performance; although its difficult to perform a detailed analysis of the implementation of the original SayWhen algorithm.

Wavelet-based methods were also investigated to assess their potential uses to increase the robustness to small noise fluctuations which may mistakenly be detected as signal. Wavelet algorithms can be computationally simple and provide an improved time-

frequency resolution when compared to Fourier methods which presents interesting options when it comes to analysis many of which have been explored through the telecommunications industry. Pursued here were methods designed to perform a voice-activity detection function on signals while simultaneously reducing the affect of noise that may be present in a signal. The four methods investigated were evaluated on their accuracy when compared to the hand-coded times however none of the methods used were particularly accurate. The method designed to detect voiced/unvoiced speech (method 3) using energy methods did not produce results that warranted further investigation as every frame of every signal tested produced an unvoiced frame detection.

Methods 2 and 4 used the Wavelet-Packet transform to decompose a signal into a number of evenly spaced frequency subbands for processing in different ways. Typical wavelet analysis methods use wavelet coefficient thresholding as a means to either reduce noise or save bandwidth; as part of the decomposition process used in these methods an attempt to reduce any transient or noise-like portions of the signal is made using the popular Teager Energy Operator. Performance in these two conditions was generally poor with typical differences between the true speech onset latency and the detected latency being on the order of hundreds of milliseconds. This likely arises from two sources within the algorithms used; first is the noise removal process that the signals undergo using the TEO

and the second is the significance conditions that needed to be satisfied before a signal detection was made.

The TEO process decays non-periodic components of an input signal which can certainly reduce the effect of background noise in the analysis however its impact on the actual speech portion is unclear. A potential future research avenue could include quantifying the total change in clean speech signals that the TEO can account for before using it to eliminate potential noises in speech signals.

Method 2 also used a popular statistical measure aimed at producing a robust threshold for coefficient thresholding which may not be appropriate. Most of the thresholds are designed to respond well to white noise while preserving speech which is not a condition that is typically of concern in the conditions that we are concerned with – a reasonably good recording environment with reasonably cheap modern recording equipment should produce relatively high quality recordings that do not suffer from high levels of white or coloured noise. The larger problem within method 2 likely comes from the VAD process – after being processed the wavelet coefficients are inverted to obtain an inverted signal which should only contain speech. In order to detect the onset a rolling average was used as an adaptive signal detection threshold which either found no signal or a very early signal. Of course redesigning this component could very likely produce drastically

different results and could be considered as an area for further research; the large question is how large should a value be above the rolling mean before it is considered significant. Of course using an absolute measure of size is difficult because it is impossible to a priori know the magnitude of the difference between large and small coefficients so a decision method for this needs to be determined.

Similarly the VAD decision in method 4 attempted to use the total frame variance calculated after TEO and thresholding operations as a means for detecting a speech onset operating on the observation that the variance of speech coefficients is higher than that of noise or silence. This likely failed due to the noise reduction methods which reduced the signal variance so much that the variance estimate used as a threshold was on the order of the signal floor; a modification to how the variance estimate is calculated may be sufficient to improve the detection performance using this methodology. Instead of calculating the variance measure of a cleaned signal as our estimate a more robust method may be to simply compute the variance measure of an uncleaned frame.

The simplest method attempted provided the best results although were not better than a voice key though showed a linear trend in decreasing error from the hand coded time. This method was more viewed as a proof-of-concept attempt as opposed to a serious attempt to obtain speech onset latencies accurately and used an arbitrary measurement of

signal density to determine the onset location. This density threshold could be refined to improve the accuracy; one possible method is to use some kind of neural network which would be trained to identify densities associated with speech onsets and in turn produce an intelligent density measure.

In summary while the updated SayWhen 2 interface provides more flexibility to users it unfortunately does not have quite the same performance as the original SayWhen at this time. Improvements to this implementation of the algorithm would be possible following a detailed analysis of implementation differences between the two versions, however the main area to focus on appears to be cases of unvoiced initial phonemes.

References

Bahoura, M., Rouat, J. (2001). Wavelet speech enhancement based on the teager energy operator. *IEEE Signal Processing Letters, 8(1), 10-11*.

Burke Hubbard, B. (1998). *The World According to Wavelets (2nd ed.)*. Natick, Ma: A K Peters.

Carnero, B., Drygajlo, A. (1999). Perceptual speech coding and enhancement using frame-synchronized fast wavelet packet transform algorithms. *IEEE Transactions on Signal Processing, 47(6),* 1622-1635.

Carroll, N.C., Young, A.W. (2005). Priming of emotion recognition. *Quarterly Journal of Experimental Psychology*, 58A, 1173-1197.

Chen, S., Wang, J. (2002). A Wavelet-based voice activity detection algorithm in noisy environments. *9Th International Conference on Electronics, Circuits and Systems, 2002* (pp. 995-998)

Chen, S., Wang, J. (2004). Speech enhancement using perceptual wavelet packet decomposition and teager energy operator. *Journal of VLSI Signal Processing, 36,* 125-139.

Chen S.H., Wu, H.T., Chang, Y., Truong, T.K. (2007). Robust voice activity detection using perceptual wavelet-packet transform and teager energy operator. *Pattern Recognition Letters, 28(11).* 1327- 1332.

Chiodi, R., Massicotte, D. (2009). Voice activity detection based on wavelet packet transform in communication nonlinear channel. *First International Conference on Advances in Satellite and Space Communications, 2009*. 54- 57.

De Houwer, J. (2004). Spatial Simon effects with nonspatial responses. P*sychonomic Bulletin & Review*, 11, 49-53

Donders, F.C, (1869). On the Speed of Mental Processes. In W.g. Koster (Ed.) *Attention and Performance II (1969) (412-431),* Amsterdam: North-Holland Publishing Company

Duyck, W., Anseel, F., Szmalec, A., Mestdagh, P., Tavernier, A., & Hartsuiker, R.J. (2008). Improving accuracy in detecting acoustic onsets. *Journal of Experimental Psychology: Human Perception & Performance, 34(5),* 1317-1326.

Eshaghi, M., Mollaei, K.M.R., (2010). Voice activity detection based on using the wavelet packet. Digital Signal Processing, 20(4), 1102-1115.

ETSI: Draft Recommendation prETS 300 724: - GSM Enhanced Full Rate (EFR) speech codec, 1996.

Frederiksen, J.R., & Kroll, J. F. (1976). Spelling and sound: Approaches to the internal lexicon. Journal of Experimental Psychology: Human Perception & Performance, 2, 361-379.

Fry, D.B., (1979). *The Physics of Speech.* Cambridge, Cambridge University Press.

Ghanbari, Y., Karami-Mollaei, M.R., (2005). A new approach for speech enhancement based on adaptive thresholding of wavelet packets. *Speech Communication, 48, 927-940.*

Graps, A. (1995). An introduction to wavelets. *Computational Science & Engineerin, 2(2).* 50-61.

Haigh, J.A., Mason, J., S., (1993). Robust voice activity detection using cepstral features. *1993 IEEE Region 10 Conference Computer, Communication, Control and Power Engineering Proceedings, 3,* 321-324.

ITU-T: Draft Recommendation G.729, Annex B: Voice-Activity Detection, 1996

Jabloun, F, Cetin, A.E., Erzin, E. (1999). Teager energy based feature parameters for speech recognition in car noise. *IEEE Signal Processing Letters, 6(10),* 259-261.

James, C. (1996). A vocal response time system for use with sentence verification tasks. *Behaviour Research Methods, Instruments, & Computers, 28*, 67-75.

Jansen, P.A., Watter, S. (2008). SayWhen : An automated method for high-accuracy speech onset detection. *Behavior Research Methods*, *40*(3), 744-751.

Jeub, M., Kolossa, D., Astudillo, R.F., Orglmeister, R. (2009). Performance analysis of wavelet-based voice activity detection. *Proceedings of the International Conference on Acoustics, including the 35th German Annual Conference on Acoustics (DAGA).* Germany: Deutsche Gesellsch. f. Akustik

Kello, C.T., Kawamoto A.H. (1998). Runword: An IBM-PC software package for the collection and acoustic analysis of speeded naming responses. *Behavior Research Methods, Instruments, & Computers*, 30, 371-383.

Kessler, B., Trieman, R., Mullennix, J. (2002). Phonetic biases in voice key response time measurements. *Journal of Memory & Language*, 47, 145-171.

Mallat, S. (1999). *A Wavelet Tour of Signal Processing (2nd ed.).* San Diego, CA: Academic Press.

Nino, R.S., Rickard, T.C. (2003). Practice effects on two memory retrievals from a single

cue. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, 29, 373-388.

Pechmann, T., Reetz, H., Zerbst D. (1989). Kritik einer Meßmethode: Zur Ungenauigkeit von voice-key Messungen [Critique of a method of measurement: On the unreliability of voice-key measurements]. *Sprache & Kognition*, 8, 65-71.

Pinter, I. (1996). Perceptual wavelet-representation of speech signals and its application to speech enhancement. *Computer Speech & Language, 10(1),* 1-22.

Rabiner, L., R., Sambur, M.R., (1977). Voiced-unvoiced-silence detection using the Itakura LPC distance measure. *Acoustics, Speech and Signal Processing, IEEE International Conference on ICASSP '77.* 323-326.

Rastle, K., Davis, M.H. (2002). On the complexities of measuring naming. *Journal of Experimental Psychology: Human Perception & Performance*, 28, 307-314.

Shaojun, J., Haitao, G., Fuliang, Y. (2004). A new algorithm for voice activity detection based on the wavelet transformation. *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing* (pp. 222-225). 20-22 Oct. 2004.

Sheikhzadeh, H., Abutalebi, H. R. (2001). An improved wavelet-based speech enhancement system, *EUROSPEECH 2001*, 1855-1858.

Stegmann, J., Schroder, G. (1997) Robust voice-activity detection based on the wavelet transform. *IEEE Workshop on Speech Coding for Telecommunications Proceeding, 1997* (pp. 99-100). 7-10 Sep. 1997.

Sternberg, S. (1969a). The Discovery of Processing Stages: Extension of Donders Method. In W.g. Koster (Ed.) *Attention and Performance II (1969) (279-315),* Amsterdam: North-Holland Publishing Company

Tyler, M.D., Tyler, L., Burnham, D.K. (2005). The delay trigger voice key: an improved analogue voice key for psycholinguistic research. *Behavior Research Methods, 37(1),* 139-147.

Walker, J.S. (2008). *A Primer on Wavelets and their Scientific Applications (2nd ed.).* Boca Raton, FL: Chapman & Hall/CRC.

Wang, K.C., Chin, C.L., Tsai, Y.H. (2009). A wavelet-based denoising system using time-frequency adaptation for speech enhancement. *International Conference on Asian Language Processing, 2009. 7-9 December 2009, pp. 114-117.*

Appendix A:

SayWhen 2 Interface

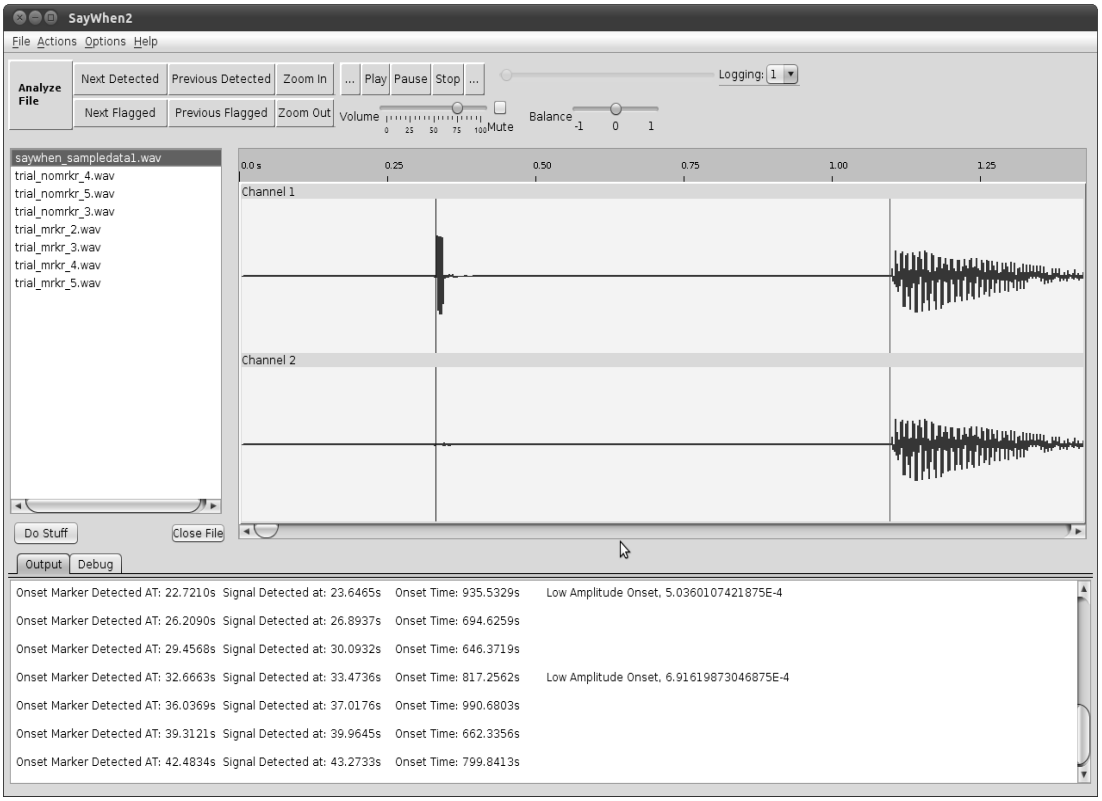

Figure 7: The main SayWhen 2 interface, showing an analyzed trial with output

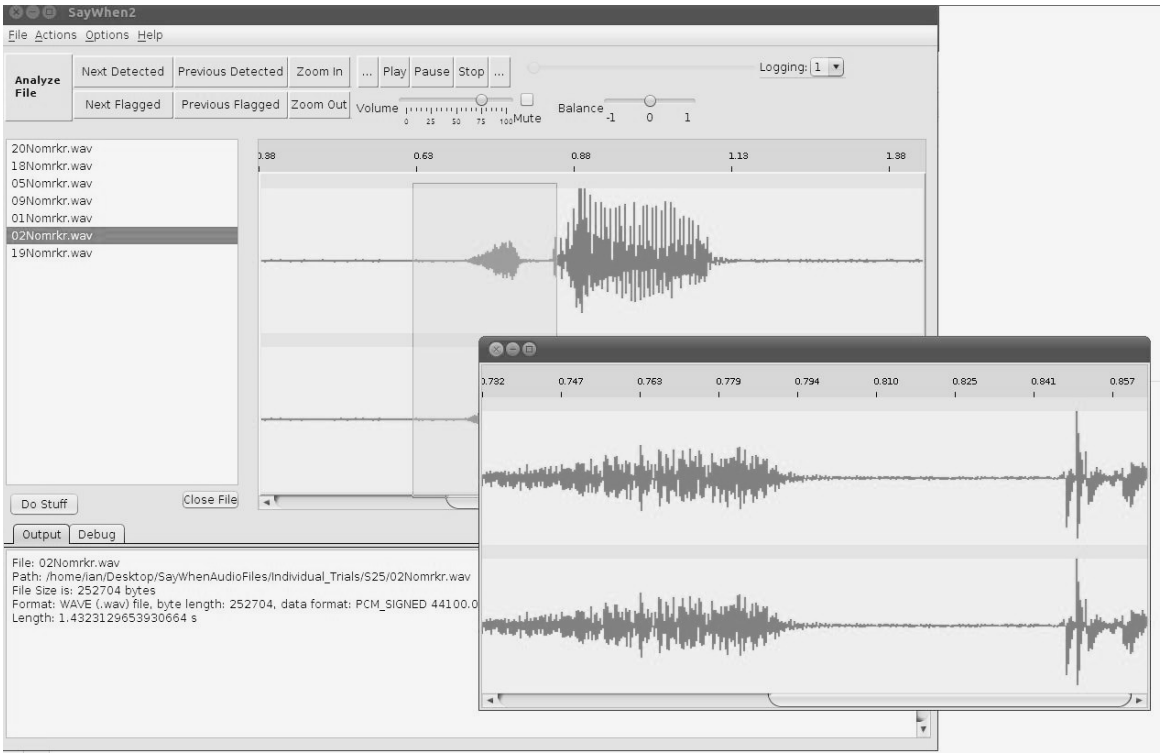

Figure 8: SayWhen 2's trial inspection capability

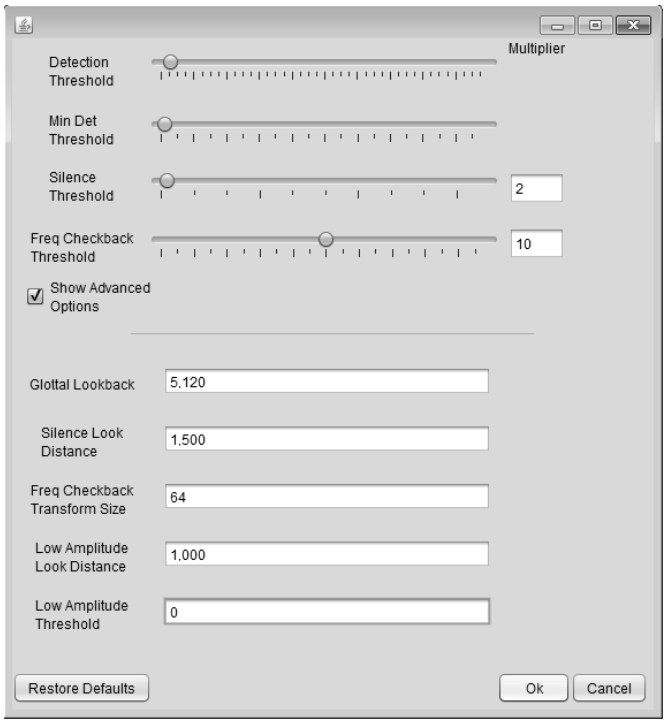

Figure 9: SayWhen 2 analysis parameters menu

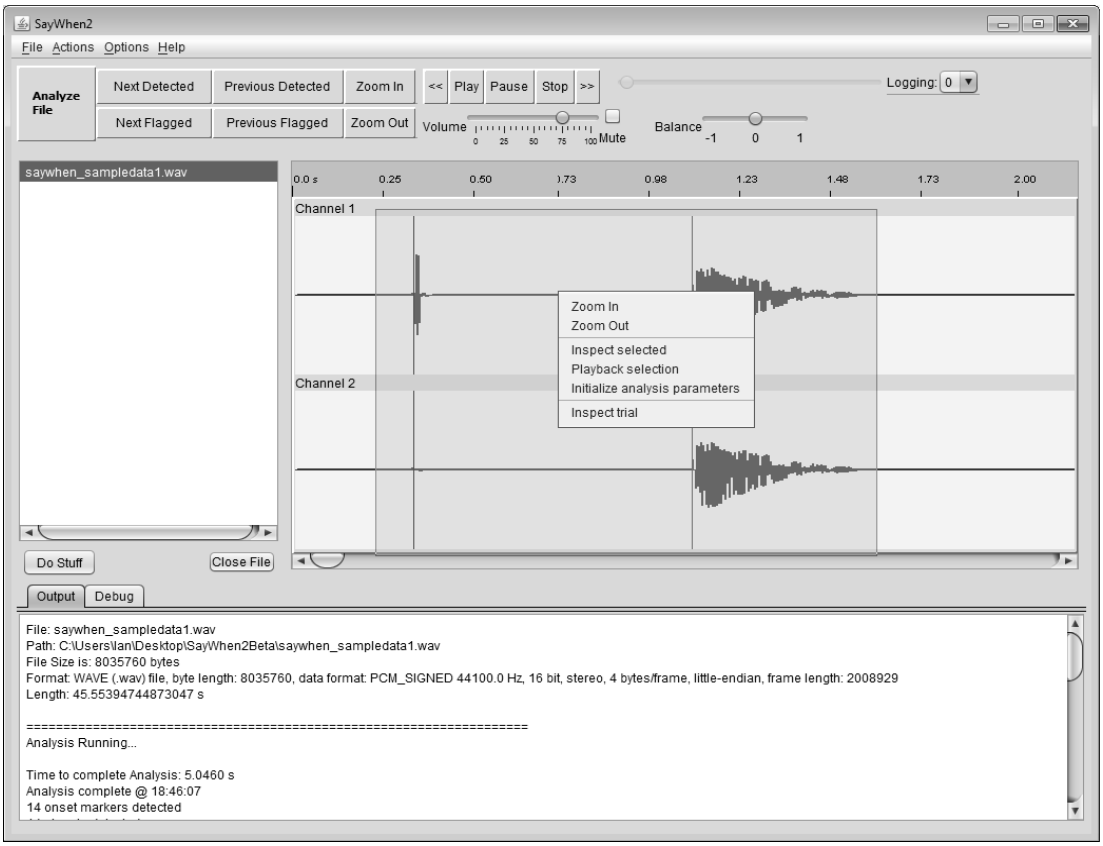

Figure 10: SayWhen 2's context menu

Appendix B:
SayWhen 2 Calibration Guidelines and Parameter Description

In order for the software to properly find speech onsets the search algorithm has to have several parameters set. This process begins by providing audio data that will be used to extract baseline search parameters. The audio data used for this should be a quiet example from your experimental sessions where the subject is not speaking and does not have to be long (< 1s).

This is done in the SayWhen interface by highlighting a section of the open audio waveform (drag the mouse on the waveform from left to right, holding down the primary (left) mouse button) and then right clicking within the highlighted selection.

1. Open a typical experiment wave file in the SayWhen interface
2. Highlight a quiet section of the recording (approximately 500 ms long is sufficient)
3. Right click in the selection and select "Initialize analysis parameters"

SayWhen2 Parameters:

Detection Threshold – primary detection parameter controls where an initial signal is detected.

Min Det Threshold – when a signal is detected the position is successively refined by reducing the window size and the detection threshold until their minimums are reached. If the signal exceeds the minimum detection threshold then the algorithm decides that a candidate signal has been detected.

Silence Threshold – controls the silence floor in the search detection algorithm. SayWhen checks to see if a signal detection is followed by a region of low amplitude signal. If the region is below this threshold then the algorithm adds the current detection to a candidate list and continues to search for more signal following the current position. When successive signal detections exceed this threshold SayWhen has found a signal.

Freq Checkback Threshold - when a signal is detected the position is refined by searching backwards in time and monitoring the peaks in the detected signal. When a certain number of successive peaks fall below the checkback threshold the signal is classified as faded and this position is returned as the final position of the signal.

Advanced Options

Glottal Lookback – the distance back the algorithm will check successive detections to determine if they are close enough to form part of the same utterance or if they could be considered noise.

Silence Look Distance – the distance out from a candidate signal the algorithm will search to check if is below the silence threshold.

Freq Checkback Transform Size – the Fourier transform size used during the frequency checkback. Half this size is the number of samples checked; if the value is 32 then 16 samples from each channel is used for the checkback.

Low Amplitude Look Distance – How far back the algorithm checks from the beginning of a signal to determine if it is a low-amplitude onset.

Low Amplitude Threshold – If the total amplitude of the signal is below this value then the signal is classified as having a low amplitude onset.

Determining the Detection Thresholds

When setting the four primary detection parameters an initial approach should be to start with a medium detection threshold, a low min. detection threshold, a low silence threshold and a medium frequency threshold.

Debug Output Sample:

1.
==========Beginning RT Search==========
beginning @: 593422
Dettime first detected as: -1 pos:0
Signal Onset at: -2.0 FilePos: 593420.0

This indicates one of the following – the initial detection threshold is too high and no candidate signals are found or that the minimum detection threshold is too high.

2.
==========Beginning RT Search==========
beginning @: 59856
Dettime first detected as: 66560 pos:0
2: Dettime Changed to: 0
Signal Onset at: 0.0 FilePos: 59856.0

This indicates that the frequency check-back parameter may be set too low – the dettime is always changed to 0.

3. ==========Beginning RT Search==========
beginning @: 1725408
Dettime first detected as: 71680 pos:0
.
.
.
Dettime first detected as: 296448 pos:296448
Dettime first detected as: 350208 pos:296960
Dettime first detected as: 350720 pos:350720
Signal Onset at: -2.0 FilePos: 1725406.0

Indicates that the silence parameter may be to high.

4.
==========Beginning RT Search==========
beginning @: 593424
Dettime first detected as: 31232 pos:0
2: Dettime Changed to: 21792
Signal Onset at: 43584.0 FilePos: 637008.0

This is a typical good detection profile.