NOTEVIEW: A COMPUTER PROGRAM FOR THE ANALYSIS OF SINGLE-LINE MUSICAL PERFORMANCES

D. Gunawan[1] and E. Schubert[1]*

[1]Empirical Musicology Group, School of English, Media and Performing Arts,
University of New South Wales, NSW 2052, Australia
***e.schubert@unsw.edu.au**

Newly developed software, NoteView, is used to analyse the fundamental frequency (F0) and categorical as well as microtonal pitch from audio recordings of music performances from a single line (monophonic) instrument. The code is based on the SWIPE algorithm developed by Camacho and Harris. The features of the interface are described and a comparison of two performances of a familiar piece played by a professional French horn player is used as an example for the purpose of investigating the pitch stability of the two renditions. Results produced by NoteView indicate that pitch pairs across the two performances differed by a mean of 7 cents, and that within-note standard deviation was typically 6 cents. These results are examined using the various customisable views and statistics returned by the software. Some of the features and limitations of NoteView are discussed. The software is currently implemented in Matlab and is freely available from the UNSW Empirical Musicology web site http://empa.arts.unsw.edu.au/research-and-creative-practice/research-projects/empirical-musicology/.

INTRODUCTION AND RATIONALE

Research on acoustics of speech and music frequently involves analysis of F0 (the fundamental frequency of a periodic signal), and a range of software exists for analysing this property [e.g. 1, for a review of recent such software see 2]. However, the acoustic analysis of music, speech and noise at times require significantly different approaches. An important example is pitch, which is a property of subjective music perception that can be expressed as a logarithmic transform of F0, and is reported here in units of semitones or cents[1]. Pitch tends to be produced and perceived categorically in Western music but not in most Western speech [3, 4]. While efficient software exists for analysis of many musical features [5, 6], the present paper reports our attempt to provide a graphic representation of musical pitch that facilitates comparison of different performances of the same piece.

We sought to develop a freely available tool that could take as its input a sound recording of a single line instrument (in the present case a French horn), parse the notes of the performance into a list of events (that could be inspected in both tabular and graphic forms), and to provide a comparison of this event list with an event list of another performance (also reported via tables and graphs). We wanted to be able to answer questions like 'how close in pitch is player A playing piece X to that of player B playing the same piece', or 'how close in pitch is player A playing piece X to that of player A playing the same piece under a different circumstance'? While Dixon and Widmer's [6] MATCH software can perform such functions, we wanted to have a strong focus on graphic and tabular representation of pitch and pitch comparison, as well as a range of statistics on pitch related information. In addition, the algorithmic foundations of our coding is different to that of Dixon and Widmer, who apply positive spectral difference vectors, whereas we focus on pitch strength, according to the SWIPE [Sawtooth Waveform Inspired Pitch Estimator] algorithm. The former has advantages in identifying note onset times with efficiency and speed. And while we too were interested in identifying temporal position of note events such that they could be matched across two renditions of the same performance, detailed information about F0 or pitch was the more important consideration here. As a preliminary exercise we examined a recording of a horn player playing the same familiar piece twice.

NoteView overview

NoteView is a music signal analysis toolbox we developed to analyse and visualise music performances in the Matlab computing environment. Reading in a monophonic sound recording, NoteView's *nAnalyse* function begins by analysing the signal to determine the time-localised fundamental frequency and RMS power information. Onsets and offsets are then derived from the pitch and power information to form a series of audio *events*. For each event, various parameters are calculated, including various within-event fundamental frequencies, timing and RMS power parameters, in addition to

[1]A cent is a ratio of $2^{1/1200}$ between two F0s. One semitone in equal temperament corresponds to 100 cents. Other tuning systems can be represented with non-integer values of equal tempered semitones between intervals. An example is given in Table 1.

several statistical descriptions. These parameters are collated into a *signal* structure, which forms the basis for more complex analysis and visualisation.

NoteView also provides an automated comparative tool, which allows two *signal* structures to be compared to each other. This can be used to compare a particular performance against a template (e.g. one derived directly from a music score to exported audio, such as is available on many music notion software packages), or to compare two performances. This functionality is provided by the *nCompare* function, and is capable of automatically determining the most appropriate events from each signal structure to be matched and compared with each other. Having matched the appropriate event pairs, the *nCompare* function then provides statistical information regarding each pair of events and these are stored in a *compare* structure.

In addition to its automated analysis and comparative tools, NoteView provides a set of tools to facilitate the manual editing of the computed structures. Events can be added, removed, split, swapped, and the onset and offset information can also be manually modified. Any manual edits also initiate the automatic recalculation of the statistical parameters, thereby ensuring that the information stored in the structures remains up to date.

While NoteView is capable of displaying the information contained in its structures in tab-delimited lists, it also provides visualisation tools, which can expedite the analysis of large sets of data. The information contained in *signal* structures can be visualised using the nView function, in either a frequency-time plot or a signal amplitude-time plot. Information contained in the *compare* structure is accessed using the *ncView* function. The *ncView* function displays the information of two signal structures concurrently in the frequency-time domain, with an option to time-align the onsets of matched events to facilitate visual comparison. An additional visualisation option allows up to 3 parameters to be plotted simultaneously in 3 dimensions, allowing many salient factors in musical performances to be identified.

NoteView Specifications

The *signal* structure acquires the F0 information in the current implementation using the SWIPE' algorithm [7], which is a sawtooth waveform inspired monophonic pitch estimator. It uses the spectrum of a sawtooth wave which is adjusted to best match the signal spectrum under investigation, and was the technique selected because of its computational efficiency and good performance compared to a range of other approaches [see 7 for details], and its compatibility with public domain audio analysis frameworks such as PsySound3 [5]. The F0 estimates are calculated for non-overlapping windows sampled at 100 Hz, potentially providing accuracy to less than a musical cent. The F0 estimates are formed into tracks (a time series containing an array of F0 estimates over the time for the audio file being analysed) based on frequency deviations over time, track length and pitch strength. The short-time RMS power is calculated for non-overlapping windows sampled at 50 Hz.

The events are then identified according to the following rules. The attack (event onset) portion of the event is defined as the time taken to reach 80% of the maximum pitch strength, and the end of the note defined as the point in time when the pitch strength drops below 20% of the median pitch strength for the note or F0 deviates by more than 40 cents from the median F0, whichever is the smaller. These thresholds are customisable, but our experiments have produced good results with these values. Pitch strength here refers to the salience of a pitch (as distinct from the more commonly understood property of height, which is measured by F0 and is commonly called 'pitch'[2]) [8, 9]. For example, a complex tone is likely to be perceived as having more pitch strength than the same tone with added narrow-band noise at the centre frequency of the tone, despite having identical pitch height.

The parameter that reports pitch height in NoteView is based on the MIDI note numbering system. F0 is converted to equal tempered semitone count according to the MIDI (Musical Instrument Digital Interface) protocol where A4 (defined as F0 with 440Hz) is assigned the value 69, C4 is assigned the value 60, C#4 61 and so on. By addition of two decimal places we are also able to represent pitch in units of cents (see footnote 1). For example 60.03 can be a quantification of an equal tempered middle C played three cents sharp, and so other tuning systems that require non-integer representation of cents could, in principle, be analysed with an accuracy of ±0.5 cents, provided that the note was sustained for a sufficiently long time. In NoteView, the summary value of pitch height reported for an event is the median of F0 estimates across windows within the event, reported in these MIDI units.

Pitch deviation of an event indicates the amount of variability in F0 during the event. It is measured as the standard deviation (SD) of the F0 estimates produced by the array of windows of the event and is reported in units of cents.

Finally, pitch stability is reported. This indicates how stable a played pitch remains for the duration of the event. In the current implementation of NoteView this stability is reported by comparing the temporally split (into two even halves) event. The pitch variance (in cents) is calculated for each half, and an F-test is conducted that compares the two variances. If the F test is not statistically significant at $p = 0.05$, a value of 0 is assigned to the stability change. If the F-test is significant, then the log of the F statistic is assigned to the stability change. A positive value indicates that the second half of the note (event) has statistically less pitch variability than the first half. A negative value indicates that the first half has statistically less pitch variability than the second half. The value is not indicative of the actual variance/standard deviations, just the amount by which one half changed relative to the other,

[2]We use the term pitch and pitch height interchangeably here.

as reflected by the F-test. Therefore it is possible to have, for example, an increased stability as the note unfolds (less stable to more stable, reflected by a positive value) while the overall variability of the event (reported by NoteView as SD) is very small. In such a case, the stability rate value may have limited utility. Because the F statistic is only reported when the difference is significant, the absolute value will generally be greater than 1 (and therefore its log greater than 0) but zero when not significant. It should also be noted that this value has some dependence on the length of the event (long events are more likely to be reported as having non-zero stability rates). This is an artefact of the statistic.

In summary, the parameters that the NoteView *signal* structure reports for the purpose of the current investigation are onset time, offset time, pitch strength, pitch height, pitch deviation and pitch stability change. Additional parameters that are variants of the above are also accessible for table reporting and visualisation, and plans for further expansion and flexibility may be considered in future versions of the software.

The *compare* structure calculates the frequency distance between each event pair across the 2 signal structures using a dynamic programming algorithm [10] to determine the optimum matching of events. Parameters returned by the compare structure include several temporal relations concerning the relative locations and amount of temporal overlap of the two notes (consider the case when a note played in one performance is slightly longer than the corresponding note played in another). In addition, and of particular relevance to the analysis we present in the following section, the *compare* structure also returns the difference in F0 medians between event pairs and interval difference relative to the previous event, each in MIDI units.

WORKED EXAMPLE

To observe some of NoteView's capabilities, the recordings of a professional horn player were evaluated, with the objective of determining his pitch production accuracy with respect to just intonation (JI) and between renditions. The player was a professional horn player and composer[3]. The player reported the intention of using a just intonation system, which formed the basis of some of our analyses.

The accuracy with which flautists and violinists can reproduce pitches depends on the playing condition, their expertise and several other factors, and is typically greater than 10 cents [11, 12]. Furthermore, Sundberg and colleagues [13] reported that professional singers had a mean difference of 7 cents across renditions. We did not find published literature on the pitch 'accuracy' of a horn player reproducing the same piece and decided to test the hypothesis of 7 to 10 cent accuracy for the task.

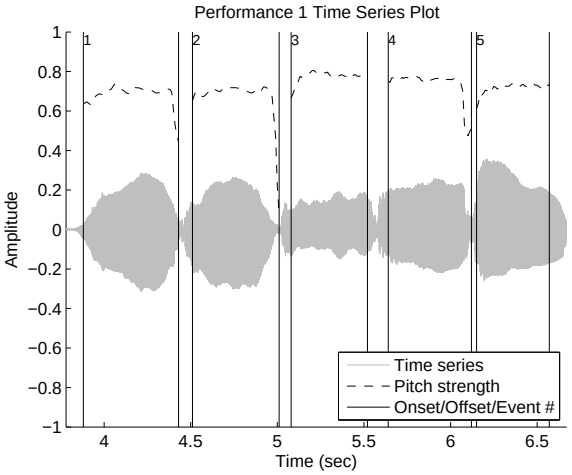

Figure 1. Time series plot of events 1-5 of performance 1 (first five notes of 'Twinkle Twinkle Little Star'), parameters relating to the event segmentation of the audio data.

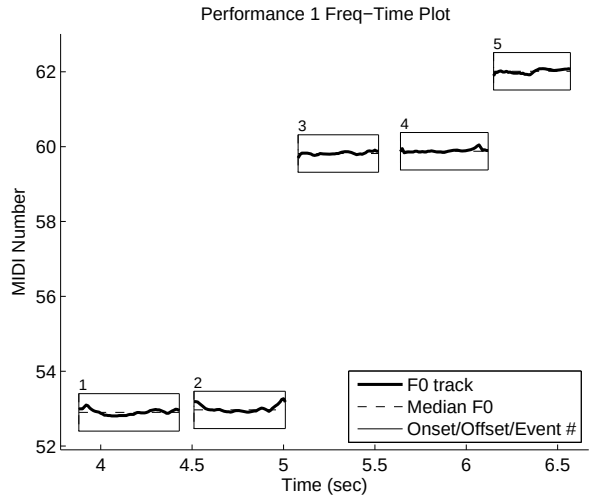

Figure 2. Frequency-time plot of events 1-5 for performance 1. The F0 track denotes the fundamental frequency (in MIDI units) as a function of time. The boxes outline the event number, encapsulating the event onset (left edge), offset (right edge), and ±50 cents either side of the median F0 (top/bottom edge).

To determine the pitch production accuracy of the horn player, the musician was instructed to play 'Twinkle Twinkle Little Star'. The player was not told that he would be playing the piece twice, and was not told that the intention was to examine pitch accuracy. After the piece was recorded, the player was asked to play the piece again, resulting in two recorded performances. The recordings were made at the recording studios of the Australian Institute of Music, 1-51 Foveaux Street, Surry Hills, NSW, Australia using ProTools audio editing software, with recordings saved as wav files at 16 bit depth, 44.1kHz sampling rate, suitable for NoteView input. The two performances were then analysed using NoteView's *nAnalyse* function to generate *signal* structures

[3]We are grateful to Michael Dixon, who agreed to be named as the performer in this study.

for each performance separately. Fig 1 and Fig 2 illustrate the visualisations generated by the *nView* function given a signal structure, as a time series, frequency-time plot as well as an event list (Table 1).

Table 1. List view of events 1-5 of performance 1. F0 is the fundamental frequency in Hz to two decimal places (two decimal places are returned by the software, though here and in typical performance conditions the error is about ±1Hz) . On and Off are the note onset and offset times with respect to the time elapsed in the sound file. MIDI# is the pitch in MIDI units. The player reported the intent to use just intonation (JI). His starting tone (and musical tonic) was F3, with an empirical F0 of 174 Hz.. With this F0 for F3, the ideal JI tunings are 196, 217, 231, 260 and 289 Hz for G3, A3, Bb3, C4 and D4 respectively (204, 386, 498, 702 and 884 cents above F3 respectively). Error is the difference in cents between the played F0 and the ideal JI F0.

Event#	F0(Hz)	On(sec)	Off(sec)	MIDI#	Note	Error(cts)
1	173.62	3.88	4.43	52.90	F3	0
2	174.17	4.58	5.01	52.96	F3	6
3	259.16	5.08	5.52	59.84	C4	-8
4	260.06	5.64	6.12	59.90	C4	-2
5	291.24	6.15	6.57	61.86	D4	11

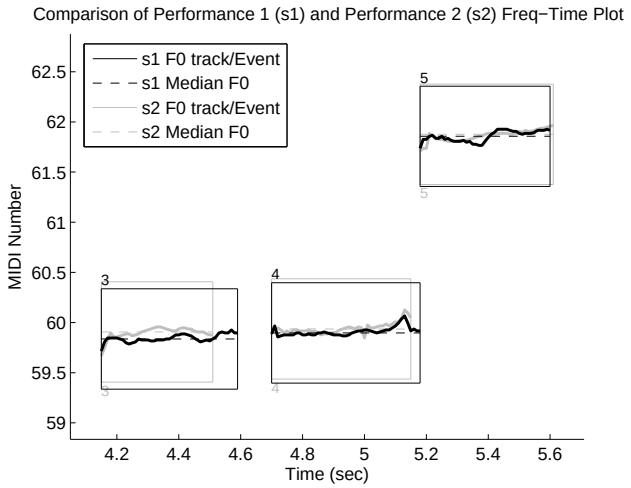

Figure 3. Frequency-time plot comparing the onset-aligned events 3-5 (third to fifth note of Twinkle Twinkle Little Star) of 2 performances.

The two performances were then compared using the *nCompare* function, generating a comparison structure matching events from the first and second performances. Fig 3 illustrates a comparative frequency-time plot generated using *ncView*, whose onsets are time-aligned to facilitate visual comparisons of the matched events. Fig 4 shows the deviation from ideal just intonation for each of the 42 events of the two performances (see caption for Table 1). The distribution of the events towards the top right side of the centre suggests that more pitches were played slightly sharp (above the median F0). Further, the occurrence of these points in the top right quadrant indicates a consistency of slightly sharp notes across the two performances. Visual inspection shows that these intonation variations fall within a boundary of ±20 cents with the calculated mean of the absolute value of the deviations being 7.2 cents, minimum of 0 and maximum of 25 cents.

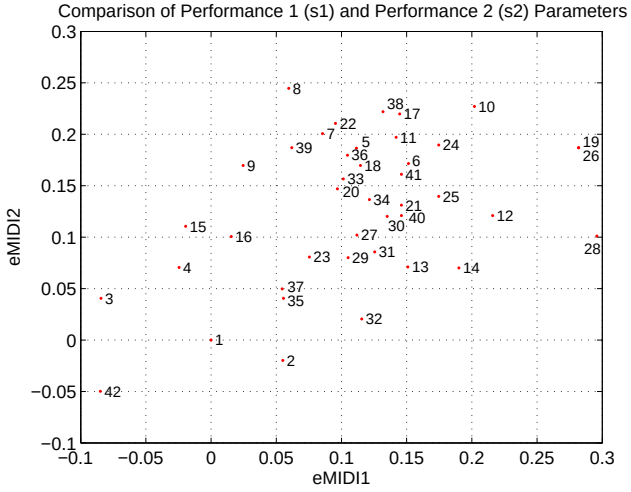

Figure 4. Deviation from just intonation pitch (in semitones) for performance 2 (y axis) plotted against deviation for performance 2, for the 42 notes in each performance.

In Fig 5 and Fig 6, the within-event variation of F0 is plotted. Fig 5 comparatively illustrates the standard deviations of the fundamental frequency for each event of the performances. We can see that the SDs of nearly all events fall within a square bounded by ±16 cents. The median of the within-note SD across the two performances was 6 cents with a maximum of 18 cents and a minimum of 2 cents. This means that, under the assumption of normal distribution, 68% (2 SDs) of notes vary by 12 (2 x 6) cents or less while the note is being played. This statistic is sensitive to the note onset and offset criteria, because including transients will inflate the standard deviation. The criterion for an event onset is the time at which the note has reached 80% of the pitch strength, as described above in NoteView Specifications.

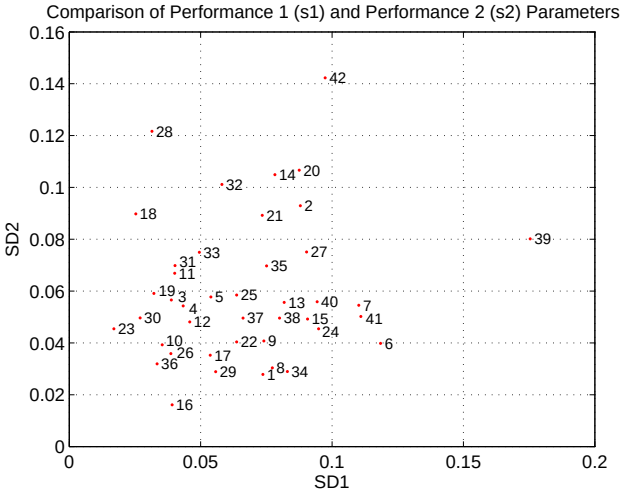

Figure 5. Standard deviation of F0 within each note in performance 2 plotted against that for each note in performance 1, for each of the 42 notes.

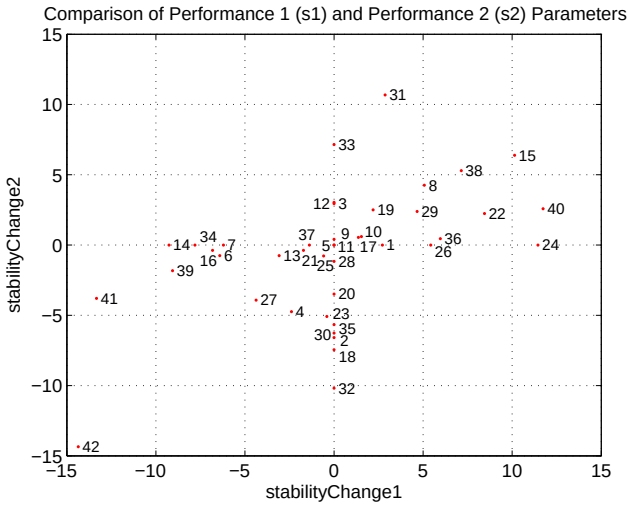

Figure 6. Stability change in performance 2 plotted as a function of that for performance 1. Units are log(F) when p =0.05, otherwise 0. Positive value denote variance decrease significantly in time from the first half to the second half of the event.

Fig 7 shows a different visualisation of the difference in pitch between events, listing them in the order in which they were played. The difference in pitch between notes in the second and first performances are plotted in semitones (MIDI units). Of the 42 events, 95% were within ±15 cents.

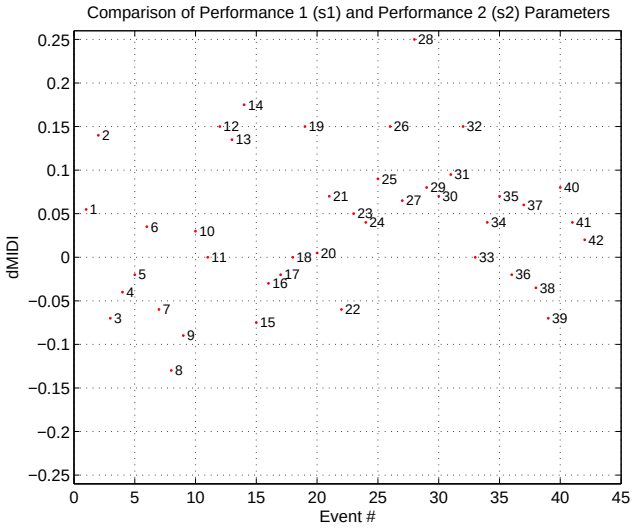

Figure 7. F0 in performance 1 minus F0 in performance 2 for each of the 42 notes, plotted in semitones.

A comparison of the within-event stability change is shown in Fig 6. 26% of events had no significant change in stability (each reported with a value of 0). 35% become significantly less stable (negative value) and 39% became more stable (positive value), with data pooled across performances. It needs to be kept in mind that some of these 'unstable' notes occur within the context of small overall variability within the note, and with the artefact of note duration affecting the calculation to some degree, as discussed above.

LIMITATIONS OF THE SOFTWARE

The current implementation of NoteView (Beta version 0.5) is limited to monophonic F0 detection between 30 and 5000 Hz. The event detection is generally quite robust, however there are issues when trying to track automatically events that have vibrato greater than 40 cents. In these circumstances, events can be manually edited using the *nEdit* function. There are also limits to the automated matching algorithm used in *nCompare*, which can skip a maximum of 2 consecutive events at a time. The system is expected to work for performances with legato and slurs (joining one event to the next event with little or no transient noise across the transition) when compared against a score, but otherwise is limited by the ability of the algorithm to identify such subtle transitions.

Additionally it should be noted that different instruments have different nuances that affect the pitch particularly around the note onsets and offsets. In the case of the horn, the inherent response time of the lip muscles can affect the pitch between notes and thus could warrant the manual editing of event onset and offset times to compensate for these physical limitations.

SUMMARY AND CONCLUSION

By applying a small set of the analytic tools available in NoteView we were able to examine the accuracy of the pitching of a professional horn player who was asked to play a simple piece twice without notice. The player was able to perform the two versions to ±7 cent accuracy of each other (pair by pair analysis). Both mean difference in pitch across versions and within-pitch variation were typically under 10 cents, with a mean of around 7 cents for both (paired comparison of median, and within event variation). We note that these data are in response to analysis of a simple piece played by a professional player, but are consistent with the literature we cited that investigated performance accuracy of other instruments including the human voice.

NoteView provides several statistics and allows visualisation of data in various, user-controlled ways [with examples shown in Figs 3-6]. Further, it calculates within-note pitch accuracy using standard deviation of pitch, and stability change statistics. It is able to deal with microtonally differentiated tunings such as equal temperament and just intonation.

The software described is a music performance analysis tool that provides information about individual signals as well as comparisons between signals using existing algorithms, with a strong focus on visual display of features concerning pitch. The system also provides timing information which can be used to investigate temporal aspects of a performance, such as tempo and articulation (ratio of note duration to inter-onset interval). Further, future versions of NoteView may apply different pitch extraction algorithms as required. The present algorithm does not do inherently well in identifying offsets and onsets when notes are played legato or slurred. Our approach mitigates the problem in two ways: (1) a score can be used

as one of the signal inputs to help the algorithm identify the likely location of a new event in the second signal input that was otherwise played in a connected manner with its preceding event, and (2) post analysis editing allows the user to manually adjust any notes that were incorrectly identified due to slurring or for any other reason. Coupled with a variety of visualisation options, NoteView provides flexible ways of accurately analysing performances to facilitate the investigation of music performance. The description in this paper refers to the Beta version 0.5, which is available for download from UNSW Empirical Musicology web site http://empa.arts.unsw.edu.au/research-and-creative-practice/research-projects/empirical-musicology/.

ACKNOWLEDGMENT

The research and software development reported in this paper is part of a larger project on microtonal music performance, supported by an Australian Research Council Grant (ARC-DP0773667) led by Greg Schiemer. The authors are grateful to Michael Dixon, Jonathan Jayanthakumar and the sound engineering team at the Australian Institute of Music. The authors also thank the anonymous reviewers for their comments and suggestions.

REFERENCES

[1] P. Loizou,, *COLEA: A MATLAB software tool for speech analysis* http://liceu.uab.es/~ joaquim/phonetics/fon_anal_acus/herram_anal_acus.html, Dept. Electrical Engineering, U. Texas at Dallas: Richardson, TX (1998-1999)

[2] A. Marsden, A. Mackenzie, A. Lindsay, H. Nock, J. Coleman, G. Kochanski, "Tools for Searching, Annotation and Analysis of Speech, Music, Film and Video A Survey". *Literary and Linguistic Computing*, **22**, 469-488 (2007)

[3] J. Wolfe, "Speech and Music: Acoustics, Signals and the Relation between them'", Proc. Inaugural International Conference on Music Communication Science (ICoMCS), 176-179 (2007)

[4] J. Wolfe, "Speech and music, acoustics and coding, and what music might be 'for'", Proc. 7th International Conference on Music Perception and Cognition (ICMPC) http://www.phys.unsw.edu.au/%7Ejw/ICMPC.pdf (2002)

[5] D. Cabrera, S. Ferguson, F. Rizwi, E. Schubert, "PsySound3: a program for the analysis of sound recordings", Proc. Acoustics 2008 – Joint conference of Acoustical Society of America and the European Acoustics Association (2008)

[6] S. Dixon, G. Widmer, "Match: A music alignment tool chest", Proc. 6th International Conference on Music Information Retrieval (ISMIR 2005), 492–497 (2005)

[7] A. Camacho,, J.G. Harris, "A sawtooth waveform inspired pitch estimator for speech and music'" *J. Acoust. Soc. America*, **124**, 1638-1652 (2008)

[8] H. Fastl, G. Stoll, "Scaling of pitch strength", *Hearing Research*, **1**, 293-301 (1979)

[9] J. W. Hall III, D.R. Soderquist,, "Encoding and pitch strength of complex tones", *J. Acoust. Soc. America*, **58**, 1257-1261 (1975)

[10] Ellis, D. *Dynamic Time Warp (DTW) in Matlab Web resource.* http://labrosa.ee.columbia.edu/matlab/dtw/ (2003)

[11] A. Botros, J. Smith, J. Wolfe, "The Virtual Boehm Flute-A Web Service that Predicts Multiphonics, Microtones and Alternative Fingerings", *Acoustics Australia*, **30**, 61-66 (2002).

[12] Pierce, J.R. *The science of musical sound*. Freeman, New York (1992)

[13] J. E. Sundberg, E. Prame,. J. Iwarsson, "Replicability and accuracy of pitch patterns in professional singers", in *Vocal fold psychology, controlling complexity and chaos* ed. P.J. Davis and N.H. Fletcher, Singular Publishing, San Diego (1996) pp. 291-306